EV Challenge Team Report

Team Information

Team Name	:	EPD-12
Team Members	:	Himanshu Singh, Aditya K Tiwari, Hasan Iqbal Khan
Mentor Name & Affiliation	:	Prof. Ganaraj P S, KIIT University

Abstract

This project demonstrates a decentralized orchestration system for electric vehicle (EV) charging using **CarMaker simulation**, **KUKSA VSS (Vehicle Signal Specification)**, and **UAgents for multi-agent communication**. The system allows an EV agent to dynamically detect reachable charging stations, negotiate for available charging spots, book one, and release it after charging is complete. The orchestration ensures efficient utilization of charging infrastructure and illustrates how simulation, vehicle data abstraction, and agent-based negotiation can be combined for smart mobility applications.

1. Introduction

Electric vehicles require reliable charging infrastructure for seamless long-distance travel. A key challenge lies in coordinating charging sessions when multiple stations and limited charging spots exist. Traditional centralized solutions lack scalability and flexibility.

This project proposes a **decentralized EV charging orchestration system** where the EV agent interacts with station agents through standardized vehicle signals (VSS) and negotiation protocols. By combining **CarMaker FMU-based supervisory control**, **KUKSA data exchange**, and **UAgents communication framework**, the system provides a scalable testbed for studying EV–charging station interactions.

2. Background

2.1 CarMaker

CarMaker is a vehicle simulation platform that provides realistic testing environments for autonomous and electric vehicle algorithms. In this project, it models the EV dynamics and supervisory energy management.

2.2 KUKSA VSS

KUKSA provides a **Vehicle Signal Server (VSS)** abstraction layer that standardizes data exchange between vehicle systems and external applications. It is used here to synchronize **FMU outputs** with **agent logic**.

2.3 UAgents Framework

UAgents enables **decentralized multi-agent communication**. Agents exchange structured messages (requests, replies, bookings) using predefined models. This supports negotiation between EVs and charging stations in a distributed manner.

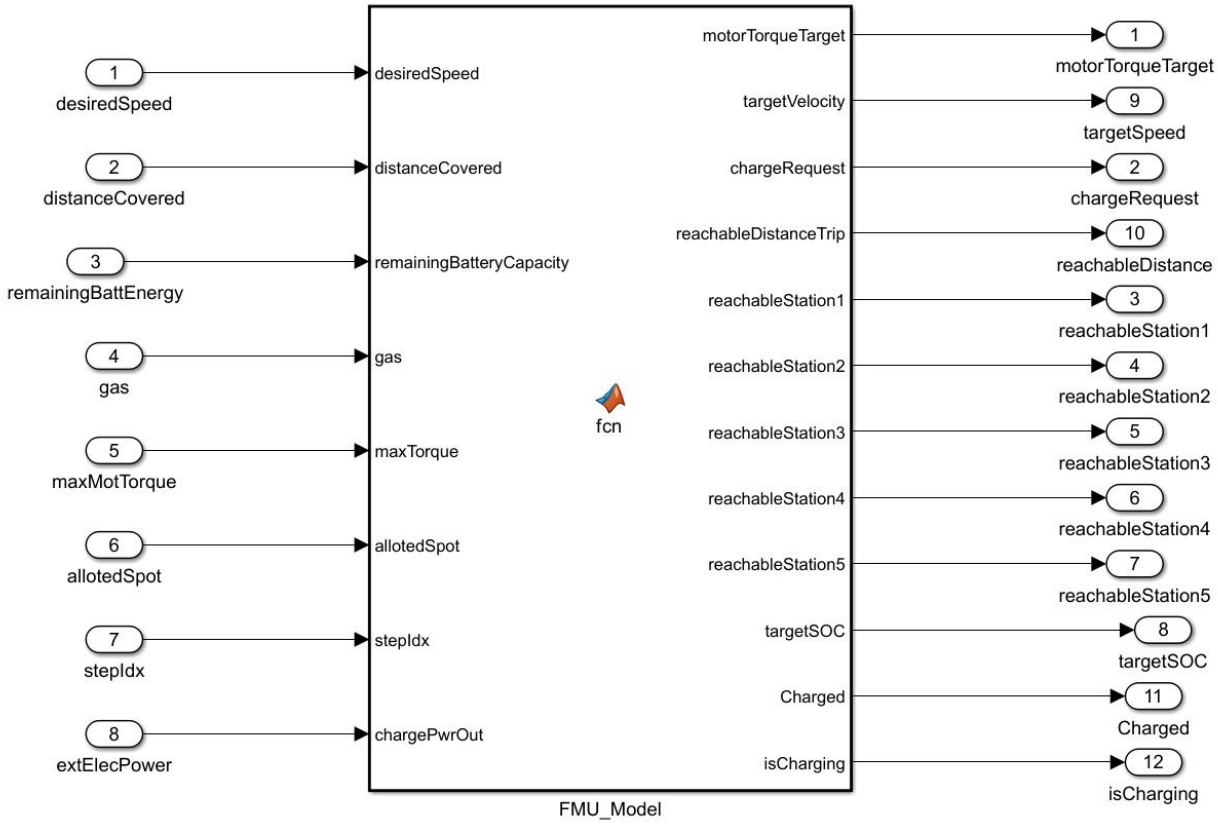
---

3. System Architecture

The system consists of three primary layers:

1. Simulation Layer (CarMaker FMU)

- Supervisory control FMU determines EV energy needs, reachable stations, and charging triggers.
- Outputs include chargeRequest, reachableStationX, Charged, and isCharging.



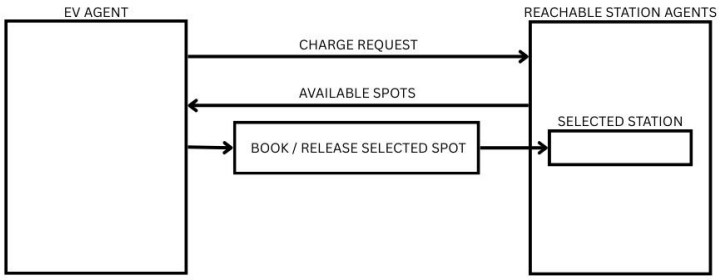
2. Middleware Layer (KUKSA + Python Bridge)

- pyCarmaker.py bridges CarMaker and KUKSA.
- Maps FMU signals ↔ VSS paths.
- Maintains synchronization between simulation state and agent decisions.

VSS Paths	FMU Outputs
Vehicle.CurrentLocation.GNSSReceiver.MountingPosition.X	Car.Charge_Request
Vehicle.CurrentLocation.GNSSReceiver.MountingPosition.Y	FMU.Abstract_Supervisory_Control2.Out.chargeRequest
Vehicle.ADAS.CruiseControl.IsError	Charged
Vehicle.ADAS.ABS.IsEnabled	FMU.Abstract_Supervisory_Control2.Out.reachableStation1
Vehicle.ADAS.ABS.IsEngaged	FMU.Abstract_Supervisory_Control2.Out.reachableStation2
Vehicle.ADAS.ABS.IsError	FMU.Abstract_Supervisory_Control2.Out.reachableStation3
Vehicle.ADAS.CruiseControl.IsActive	FMU.Abstract_Supervisory_Control2.Out.reachableStation4
Vehicle.ADAS.CruiseControl.IsEnabled	FMU.Abstract_Supervisory_Control2.Out.reachableStation5
Vehicle.ADAS.DMS.IsEnabled	FMU.Abstract_Supervisory_Control2.Out.isCharging

3. Agent Layer (UAgents)

- **EV Agent (**ev_agent.py**)**: Monitors charge requests, detects reachable stations, requests availability, books charging spots, and releases them after charging.
- **Station Agents (**stationX_agent.py**)**: Maintain spot availability, respond to requests, confirm bookings, and accept spot release.



4. Implementation

4.1 Supervisory Control (FMU)

- Computes reachable distance using battery state and efficiency.
- Identifies which charging stations are reachable.
- Issues charge requests when needed.
- Calculates target SOC based on remaining distance.

4.2 CarMaker–KUKSA Bridge

- Written in Python (pyCarmaker.py).
- Subscribes to CarMaker quantities, updates VSS values, and resets when simulation is idle.
- Synchronizes **selected spot ID** between KUKSA and CarMaker.

4.3 EV Agent (ev_agent.py)

- Periodically checks for charge requests.
- Detects reachable stations from VSS signals.
- Sends **ChargingRequest** messages to relevant station agents.
- Collects **ChargingReply**, selects the farthest reachable station with available spots, and books a spot.
- Updates KUKSA with the selected spot ID.
- Releases the booked spot when charging completes.

4.4 Station Agents (station1_agent.py, …)

- Each station maintains a list of available spots.
- Replies to EV requests with availability.
- Updates availability upon booking and release.

4.5 Message Models (models.py)

- ChargingRequest: Sent by EV → Station.
- ChargingReply: Station → EV with available spots.
- BookingRequest: EV → Station to confirm spot booking.
- ReleaseSpot: EV → Station to release spot after charging.

---

5. Results

- **Charge request detection**: EV agent correctly identifies when charging is required.
- **Reachable station identification**: Based on efficiency and range, the system dynamically filters feasible stations.
- **Decentralized negotiation**: EV agent receives replies from multiple station agents and selects the farthest reachable one with availability.
- **Booking and releasing**: Spots are allocated and released successfully, with updates reflected in KUKSA.
- **Simulation–agent integration**: Real-time synchronization achieved between CarMaker simulation, VSS abstraction, and agent negotiation.

Initial SOC	Allotted Station	Allotted Spot ID	Time Taken	SOC Left	Average Efficiency
30	Station-4	11	5hr 8min	~25	~17.76
40	Station-5	14	5hr 5min		
50	Station-5	14	5hr 5min		

---

6. Discussion

The system demonstrates the feasibility of integrating simulation-based EV energy management with decentralized agent-based charging coordination. Advantages include:

- **Scalability**: Multiple EVs and stations can be added without centralized control.
- **Interoperability**: Use of VSS standard enables modular integration.
- **Flexibility**: Decision-making logic (e.g., station selection criteria) can be adapted.

Limitations:

- Current implementation selects the farthest station; optimization could include **wait times, pricing, or SOC-based heuristics**.
- Scalability in multi-EV scenarios not yet tested in detail.
- Security and trust mechanisms between agents need further development.

---

7. Conclusion

This project successfully built a decentralized EV charging orchestration framework by integrating **CarMaker FMU supervisory control, KUKSA VSS middleware, and UAgents agent communication**. The system supports real-time simulation, negotiation, booking, and release of charging resources.

Future work includes extending to multiple EVs, incorporating real charging infrastructure data, and exploring advanced optimization strategies for station selection.

Submitted on: 17.08.2025 **Submitted by:** EPD-12