Parking Lot Simulation - Design Documentation

March 10, 2025

1 Introduction

This document outlines the design architecture for the parking lot simulation project. It explains the UML diagram, describing each component, their relationships, and how they contribute to achieving the overall project goals. The design ensures that the simulation effectively models parking lot occupancy, vehicle arrival/departure events, and system efficiency.

2 UML Diagram Overview

The UML diagram consists of three main classes:

- **Simulation**
- **ParkingLot**
- **Vehicle**

Each component is structured to accurately represent real-world interactions between a parking system, its vehicles, and the simulation managing them.

3 Component Explanations

3.1 Simulation Class

Purpose: The simulation class manages the overall system, executing vehicle arrivals, parking, and departures.

Key Attributes:

- `parking_lot: ParkingLot`: - The Simulation owns the ParkingLot, meaning if Simulation is deleted, so is the ParkingLot.
- `vehicles: List[Vehicle]` - Stores generated Vehicle objects temporarily.
- `time: float` - Keeps track of the simulation clock.

Key Methods:

- `run_simulation(duration: float) -> None` - Runs the simulation over a period.
- `generate_vehicle() -> Vehicle` - Creates new Vehicle objects.
- `process_events() -> None` - Handles vehicle arrivals and departures.

Relationships:

- **Composition with ParkingLot** - The simulation owns and manages the ParkingLot.
- **Dependency with Vehicle** - It creates Vehicle objects but does not permanently manage them.

3.2 ParkingLot Class

Purpose: - Represents the physical parking lot where vehicles are parked.

Key Attributes:

- `capacity: int` - Total available parking spaces.
- `occupied_spaces: int` - Number of currently occupied spaces.
- `queue: Queue` - Manages waiting vehicles if the lot is full.

Key Methods:

- `is_full() -> bool` - Checks if the lot is at full capacity.
- `park_vehicle(vehicle: Vehicle) -> bool` - Attempts to park a vehicle.
- `remove_vehicle(vehicle: Vehicle) -> None` - Removes a vehicle when it departs.

Relationships:

- **Aggregation with Vehicle** - The parking lot contains multiple vehicles, but vehicles can exist independently.

3.3 Vehicle Class

Purpose: - Represents a vehicle that enters, parks, and exits the parking lot.

Key Attributes:

- `id: int` - Unique identifier for each vehicle.
- `arrival_time: float` - The time when the vehicle arrives at the parking lot.
- `departure_time: float` - The time when the vehicle leaves the parking lot.

Key Methods:

- `calculate_parking_duration() -> float` - Determines how long a vehicle stays.

Relationships:

- **Aggregated in ParkingLot** - Vehicles are contained within a parking lot but exist independently.
- **Created by Simulation** - Vehicles are dynamically generated and managed by the simulation.