Architecture Diagram

1. Architecture Design Analyse

Separation of Tasks: Each layer focuses on specific responsibilities - presentation layer handles interaction(GUI), business layer handles logic processing, interface layer handles protocol encapsulation.

Dependency Inversion: Higher-level modules depend on abstract interfaces of lower-level modules, not concrete implementations, reducing coupling.

Scalability: Modules within each layer are loosely coupled, making it easier to add or replace functional modules.

2. Detailed Layer Description

2.1 Presentation Layer(GUI)

Core Responsibility: Provides user interaction entry and data visualization display, receives user operations and transfer them to business logic layer, receives business layer data and updates interface.

2.2 Business Logic Layer

Core Responsibility: The core business logic is wrapped, coordinating work between the presentation layer and the interface layer, and handling data transformation and business rule validation.

2.3 Libtraci Interface Layer

Core Responsibility: It encapsulates the libtraci protocol, offers type-safe interfaces for entity operations, hides libtraci protocol specifics (like data formats and request/response handling), and provides a unified API for accessing entities to the business logic layer.

2.4 SUMO Core

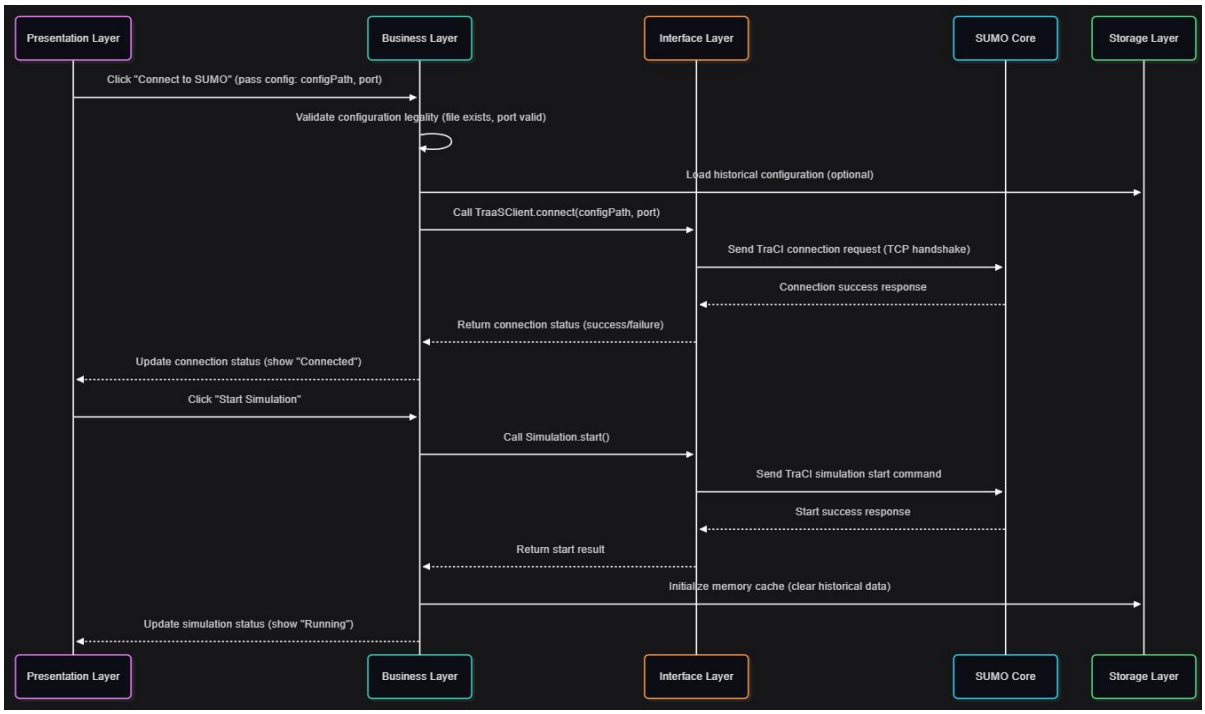
Core Responsibility: Provides core traffic simulation computing capabilities, exposes interaction interfaces through libtraci protocol.

2.5 Data Storage Layer

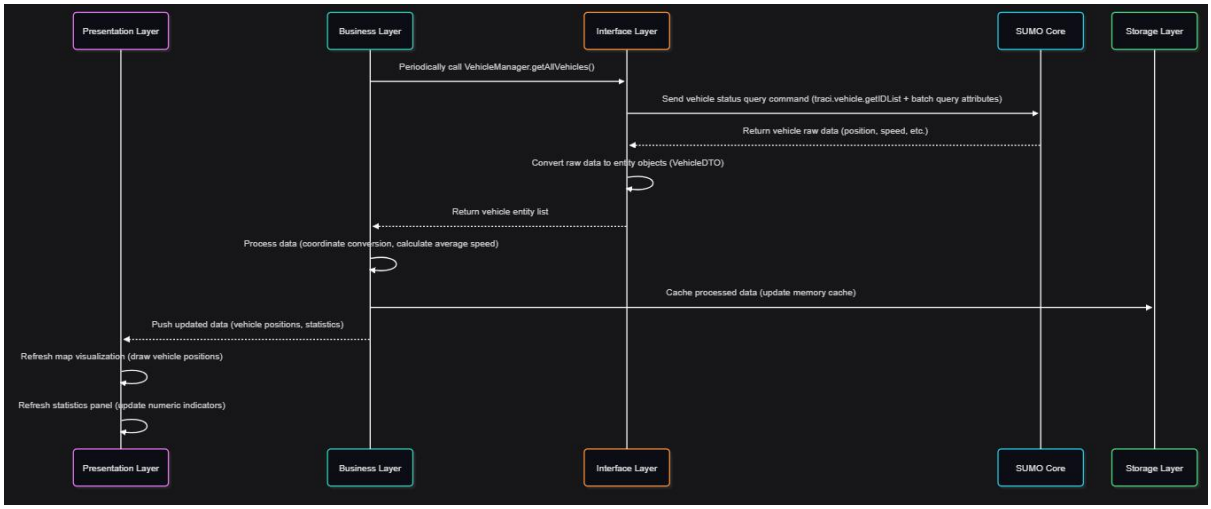
Core Responsibility: Responsible for persisting simulation data, managing storage and cache, supporting real-time data caching and querying historical data.

3. Interaction Flow

3.1 Simulation Start and Connection Flow



3.2 Entity Status Query and Interface Update Flow



3.3 Simulation Data Export Flow