THE UNIVERSITY OF BUEA
P.O Box 63,
Buea, South West Region
Cameroon
Tel: (237) 3332 21 34/3332 26 90
Fax: (237) 3332 22 72

REPUBLIC OF CAMEROON

PEACE-WORK-FATHERLAND

FACULTY OF ENGINEERING AND TECHNOLOGY

Dean: Agbor Dieudonne Agbor PhD

Vice-Dean : Nde D. Nguti, PhD

Faculty Officer : Mme Ekpombang Patience Taku

Date: 19 May 2024

CEF440 – TASK 4:

SYSTEM MODELLING AND DESIGN

FOR A ROAD SIGN AND ROAD STATE MOBILE NOTIFICATION APPLICATION

Presented by

GROUP 4

SN	NAME	MATRICULE NUMBER
1	DJOUMESSI NGUEFACK IVAN	FE21A174
2	NJOYONG GODWILL	FE21A273
3	TETUH WIBINAG ENGONWEI	FE21A322
4	DJOUMESSI DONGMO RONSARD CARNEGIE	FE21A172
5	GATCHUINNE MOMO MURIELLE CYNTHIA	FE21A200

Valery Nkemeni, PhD

1.Introduction

Effective system modeling and design is a critical step in developing complex software applications. It involves creating visual representations and abstractions of the system's structure, behavior, and interactions to better understand and communicate the system requirements and architecture. The key objectives are to capture functional requirements through use case diagrams and user stories, model the system's static structure using class diagrams and ER diagrams, capture the dynamic behavior with activity diagrams and sequence diagrams, design the user interface, and define the high-level system architecture. By creating these visual models and designs, the development team can gain a shared understanding of the system, identify and resolve potential issues early, and communicate the system's structure and behavior to stakeholders more effectively. The subsequent sections of this document will cover the specific design artifacts for the road sign, road state, and notification application, including use case diagrams, class diagrams, and user interface designs.

1.1 Purpose

The purpose of this document is to design and model the requirements for a road sign, road state, and notification app that provides users with real-time traffic updates, location-based services, and other advanced features.

1.2 Scope

The scope of this project includes the development of a mobile application that offers the following key functionalities:

- Real-time traffic updates
- Location-based services
- Automatic start and stop
- Road sign recognition and display

- User feedback and collaboration
- Audio notifications
- Integration of current speed, GPS data, date, and time
- Navigation assistance
- Data visualization
- Route optimization
- Rerouting options
- Speed camera warnings
- Exposure compensation feature

2. User Requirements

2.1 User Roles

Drivers: The primary users of the application**.**

2.2 User Stories

Drivers want a comprehensive app that provides real-time traffic updates, location-based services, and advanced features to enhance their driving experience and safety. Key requirements include:

- Real-time updates on traffic conditions, construction, weather, road closures, hazards, and checkpoints to enable better route planning.
- Location-based services like nearby points of interest and route recommendations to improve navigation.
- Automated start/stop of updates based on driver preferences or driving detection for a seamless experience.
- Road sign recognition and display, with overlay on video feed, to stay aware of surroundings.
- Ability to provide feedback on update accuracy and report road conditions to improve the app.

- Audio notifications for important updates like traffic incidents or navigation instructions.
- Display of current speed, GPS data, date, and time for comprehensive driving information.
- Turn-by-turn navigation assistance with voice prompts, visual cues, and lane guidance.
- Route optimization based on real-time traffic analysis to minimize travel time.

3. System Design

3.1 System Architecture

The system architecture for the road sign, road state, and notification app will be based on a client-server model, with the mobile app as the client and a backend server handling the data processing and management.

The key components of the system architecture include:

Mobile App: The user-facing application that provides the core functionalities and user interface.
Backend Server: The server-side component responsible for data collection, processing, and storage.
Data Sources: External data sources, such as traffic data providers, map APIs, and road sign databases, that supply the necessary information for the app.
Communication Protocols: The protocols used for data exchange between the mobile app, backend server, and external data sources, such as HTTP, WebSocket, or API calls.

The component of the Model-View Controller include:

Model

Road State: Represents the current state of the road, including construction zones, weather, closures, hazards, and checkpoints.

Road Sign: Represents a road sign with its type, location, and other attributes.

User Location: Represents the user's current location from the device's GPS.

User Preferences: Represents the user's preferences for notifications.

Feedback Report: Represents user feedback on the accuracy of updates and reports of road conditions.

View

Road State View: Displays the current road state using data visualization.

Road Sign View: Displays recognized road signs in the app or overlays them on the video.

User Location View: Displays the user's location on a map and provides navigation assistance.

Notification View: Handles the display of audio and visual notifications based on user preferences.

Feedback View: Allows users to provide feedback and report road conditions.

Controller

Road State Controller: Manages the fetching and updating of real-time road state data.

Road Sign Controller: Manages the detection and recognition of road signs.

User Location Controller: Manages the user's location data and provides location-based services.

Notification Controller: Manages the delivery of notifications based on user preferences.

Feedback Controller: Manages the collection and processing of user feedback.

3.3 UML Diagrams

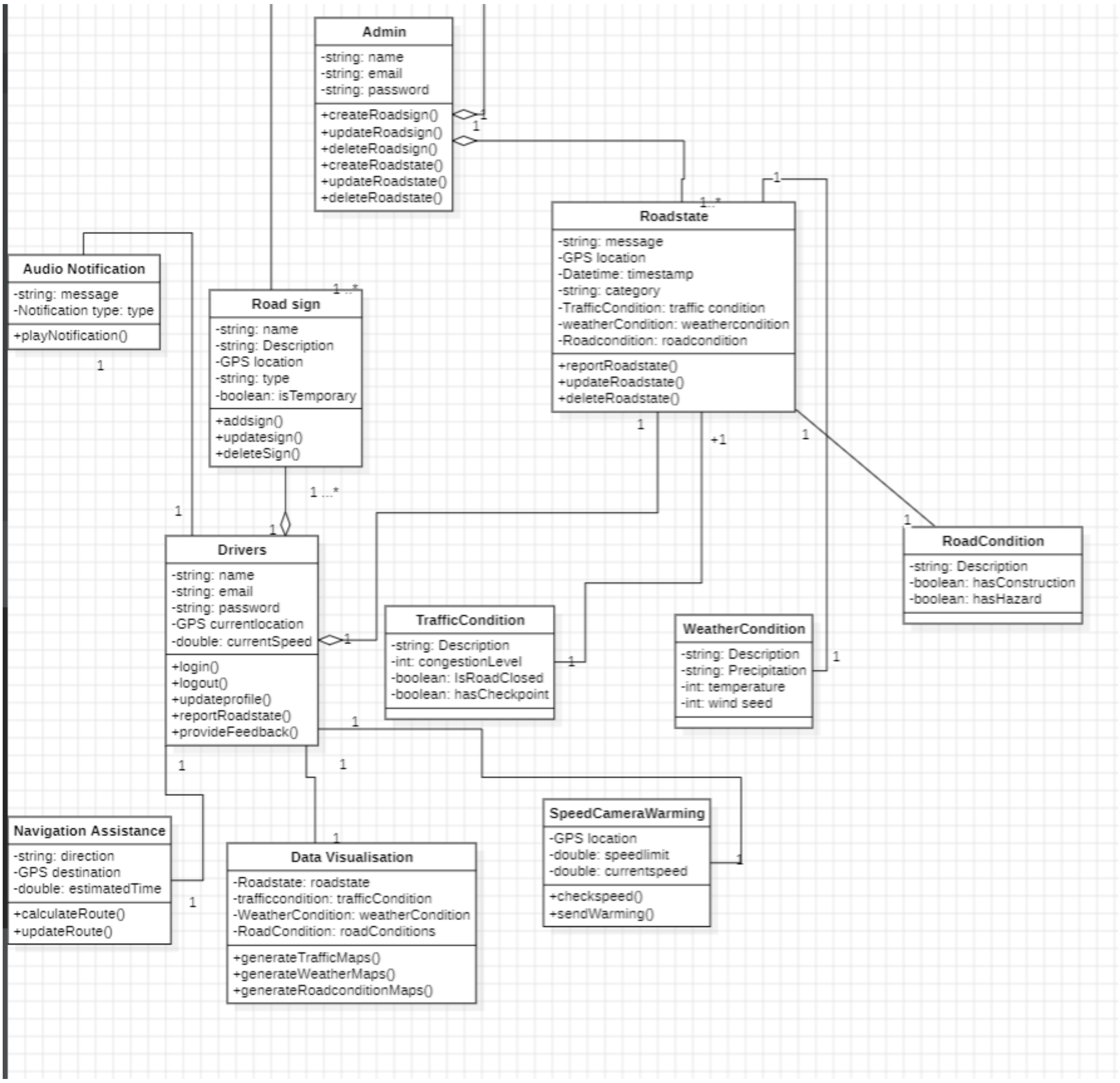
UML are categorized into the following:

1. **Structural Diagrams**:
   - Class Diagrams
   - Context diagram
   - Deployment diagram
2. **Behavioral Diagrams**:
   - Activity Diagrams
   - Use Case Diagrams
   - Sequence Diagrams

1 **Structural Diagram**

A structural diagram is a type of diagram that focuses on the static structure of a system, including its components, their properties, and the relationships between them.

1. Class Diagram



A class diagram is a type of structural diagram in software engineering that depicts the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

The following diagram works as follows:

Primary Actors

Pedestrians: Users of the app who are walking or using other modes of transportation besides driving.

Drivers: Users of the app who are driving vehicles.

Travel Agencies: Travel agencies that provide data on road conditions, traffic, and other relevant information.

Secondary Actors

GPS System: Provides location data to the app.

Data Providers: Third-party sources that supply real-time data on road conditions, weather, and other relevant information.

App Administrator: Responsible for maintaining the app, updating the database, and managing user feedback.

Use Case Relationships:

Association

Pedestrians and Drivers use the Road Sign, Road State, and Notification app.

State Agencies provide data to the app.

GPS System and Data Providers supply data to the app.

Include

The app includes the use cases for Real-time Traffic Updates, Location-Based Services, and Audio Notifications.

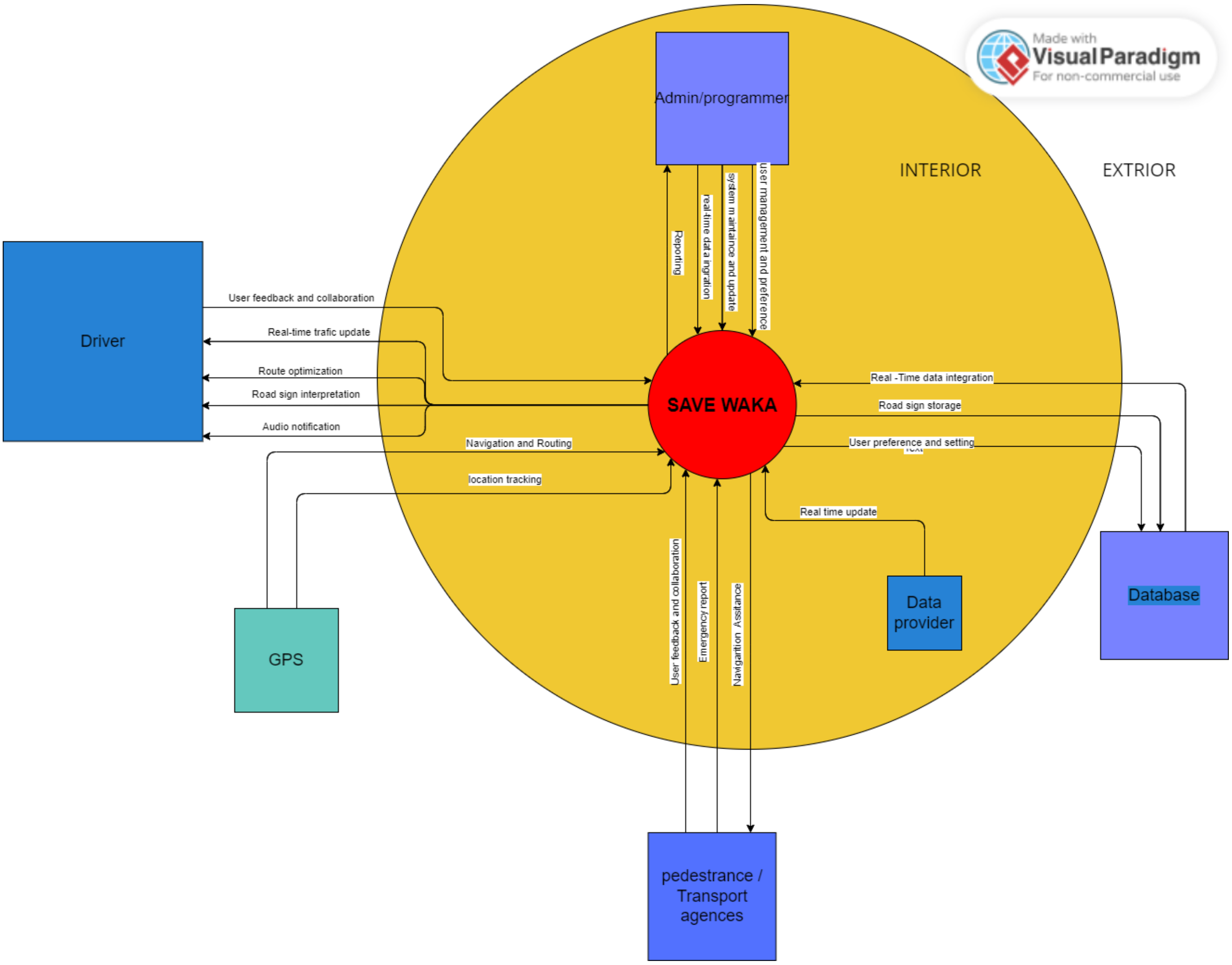
The app includes the use cases for Navigation Assistance, Data Visualization, and Route Optimization.

Generalization

Pedestrians and Drivers are both specialized types of Users of the app.

Multiplicities between different classes.

2. Context Diagram



Interpretation:

The context diagram for the "System - Safewaka" road state and road sign notification system provide an overview of the system's interactions with internal and external entities, showcasing the data flows and system boundaries.

Internal Entities:

Admin/Programmers: This entity is responsible for managing the system configuration, maintenance, and updates. They input configurations, updates, and system maintenance commands into the application server.

Data Provider: The data provider supplies real-time data on road conditions, traffic, and weather to the application server. They respond to requests for real-time data updates.

External Entities:

Drivers: Drivers interact with the mobile application, providing input preferences and receiving real-time road updates and notifications. The mobile application delivers road conditions, road sign information, and notifications to the drivers.

GPS: The GPS entity provides real-time location data to the mobile application, allowing it to deliver context-aware updates to drivers based on their current location.

Pedestrians/Transport Agencies: These entities access road sign and condition information through the mobile application, receiving updates and notifications. The application server delivers road sign and condition information to pedestrians and transport agencies.

Database: The database stores road sign information, user data, and system logs. The application server queries and updates road sign information in the database.

Data Flows:

Drivers to Mobile Application: Drivers input preferences and interact with the system through the mobile application, enabling them to receive personalized updates.

Mobile Application to Drivers: The mobile application provides real-time road conditions, road sign information, and notifications to the drivers.

Mobile Application to GPS: The mobile application receives real-time location data from the GPS entity, allowing it to deliver location-based updates to drivers.

GPS to Mobile Application: The GPS entity supplies real-time location data to the mobile application for context-aware updates.

Mobile Application to Application Server: The mobile application sends requests for data and updates to the application server based on the user's location and preferences.

Application Server to Mobile Application: The application server processes the requests and sends back the relevant data and notifications to the mobile application for display to the drivers.

Application Server to Data Provider: The application server requests real-time data on road conditions, traffic, and weather from the data provider.

Data Provider to Application Server: The data provider supplies real-time data updates in response to the application server's requests.

Application Server to Road Sign Database: The application server queries and updates road sign information stored in the road sign database.

Road Sign Database to Application Server: The road sign database provides road sign information to the application server for user notifications.

Application Server to Admin/Programmers: The application server sends logs, system status, and maintenance data to the admin/programmers for monitoring and maintenance purposes.

Admin/Programmers to Application Server: The admin/programmers input configurations, updates, and system maintenance commands into the application server for system management.

Application Server to Pedestrians/Transport Agencies: The application server delivers road sign and condition information to pedestrians and transport agencies.

Pedestrians/Transport Agencies to Mobile Application: Pedestrians and transport agencies access road sign and condition information through the mobile application's interface.

3. Deployment Diagram

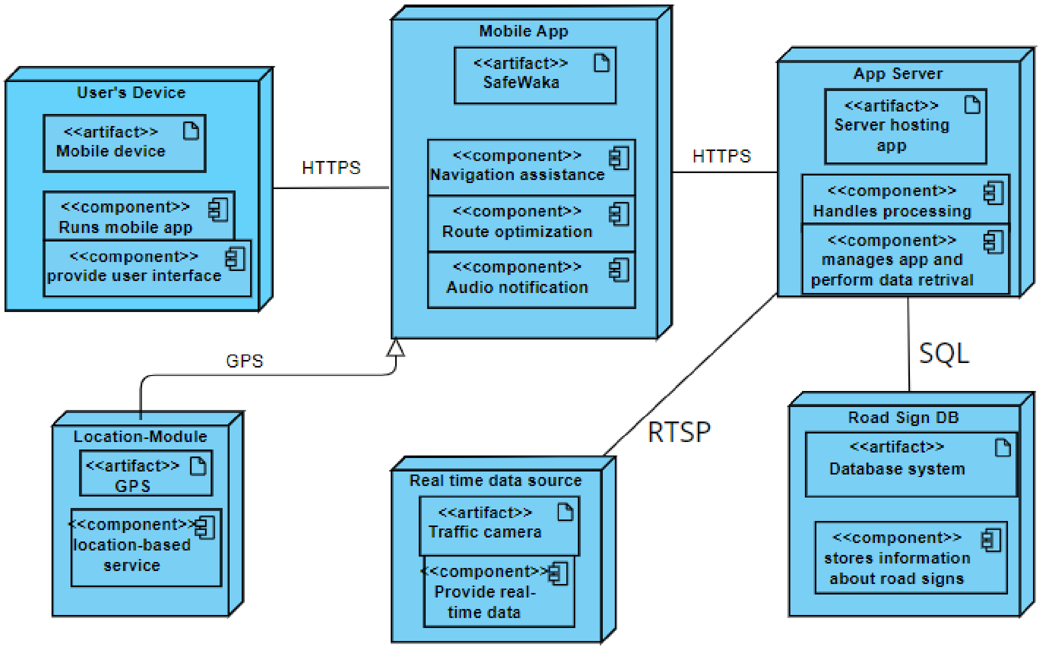
Deployment Diagram

The deployment diagram is a type of UML (Unified Modeling Language) diagram that represents the physical deployment of software components on hardware nodes. It illustrates how software artifacts (such as modules, classes, files, etc.) are allocated to different nodes (e.g., servers, computers, devices) in a system's architecture.

Key elements in a deployment diagram include nodes (representing hardware devices), artifacts (representing software components or files), and deployment relationships (showing how artifacts are deployed on nodes). Additional notations can be used to indicate communication paths, deployment constraints, and other relevant details

For a Safewaka, a road state, road sign notification app,the deployment model will show how the user's device interacts with the mobile application, which communicates with the application server. The application server retrieves data from various sources, including traffic cameras, weather sensors, and a road sign database. Each component uses specific protocols to facilitate communication and provide the intended functionalities.

Diagram


User's Device
<<artifact>> Mobile device
<<component>> Runs mobile app
<<component>> provide user interface
HTTPS
Mobile App
<<artifact>> SafeWaka
<<component>> Navigation assistance
<<component>> Route optimization
<<component>> Audio notification
HTTPS
App Server
<<artifact>> Server hosting app
<<component>> Handles processing
<<component>> manages app and perform data retrival
SQL
GPS
Location-Module
<<artifact>> GPS
<<component>> location-based service
RTSP
Real time data source
<<artifact>> Traffic camera
<<component>> Provide real-time data
Road Sign DB
<<artifact>> Database system
<<component>> stores information about road signs

Interpretation

- User's Device: Smartphones or tablets run the mobile application, providing an interface for users to interact with road sign information and receive real-time updates on road conditions.
- Mobile Application: Installed on the user's device, it allows access to road sign information, real-time updates, customizable notifications, and integration with navigation systems. It communicates with the application server for data retrieval and updates.
- Application Server: Hosts the mobile application, processing and managing logic. It retrieves data from traffic cameras, weather sensors, and a road sign database, processes user preferences, and responds to the mobile application.
- Location-based services: The location-based module leverages GPS and geofencing to provide real-time, context-aware notifications about road conditions and relevant road signs based on the user's current location. It integrates with navigation systems to enhance route guidance and safety.
- Real-time data sources: Provide real-time data on road conditions, traffic, accidents, and weather hazards. These devices transmit data via protocols like HTTP(S) or RTSP to the application server.

- Road Sign Database: Stores information about road signs on major highways, enabling instant access to their meanings and significance. It uses SQL or NoSQL protocols for data retrieval and storage.

Importance of the deployment diagram

The deployment diagram for a road state and road sign notification app offers several key benefits:

- System Visualization: It provides a visual representation of the system's architecture, helping stakeholders understand the deployment and interaction of components.
- Component Interaction: It illustrates information flow and communication between system components, crucial for accurate design and implementation.
- Scalability and Performance: It helps identify potential scalability and performance issues, allowing effective resource planning and optimization based on load and usage patterns.
- Integration Points: It highlights integration points with external systems, aiding in the identification of necessary protocols and interfaces for seamless data exchange.
- Fault Tolerance and Redundancy: It assists in designing fault-tolerant systems by visualizing redundancy and backup mechanisms, helping ensure system availability and reliability.
- Collaboration and Communication: It serves as a communication tool among project stakeholders, facilitating discussions and collaborative decision-making.

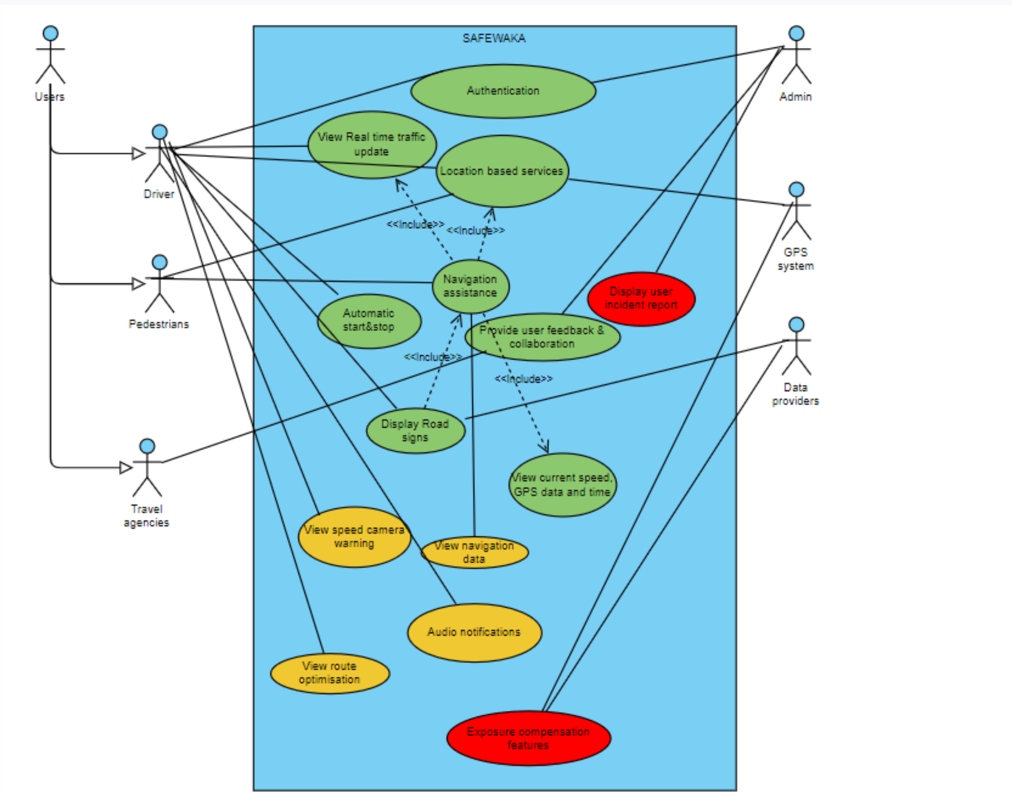
3.3.2 Behavioural Diagram

The behavioral diagrams are used to model the dynamic aspects of a system, focusing on the behavior of the system and its components over time.

A. Use Case

A use case diagram for a road state and road sign notification app illustrates the interactions between actors (users or external systems) and the system itself, showcasing the different use cases or functionalities provided by the app

- Pedestrians and car drivers are the primary users of the app, interacting with various use cases such as viewing road signs, receiving updates, receive road conditions, customizing preferences, and getting directions.
- Admin has additional use cases related to managing the app, such as updating road data and integrating GPS data.
- Travel agencies can use the app to access road state and road sign information for their clients.
- GPS system provides location-based services and data integration to the app.
- Data providers supply real-time road data, which is utilized by the app to provide accurate updates and information.



Green use cases represents functionalities with low priority.

Yellow use cases represents functionalities with medium priority.

Red use cases represents functionalities with low priority.

3.3.3 Interaction Diagrams

These are diagrams that focus on modeling the interactions between objects or components within a system.

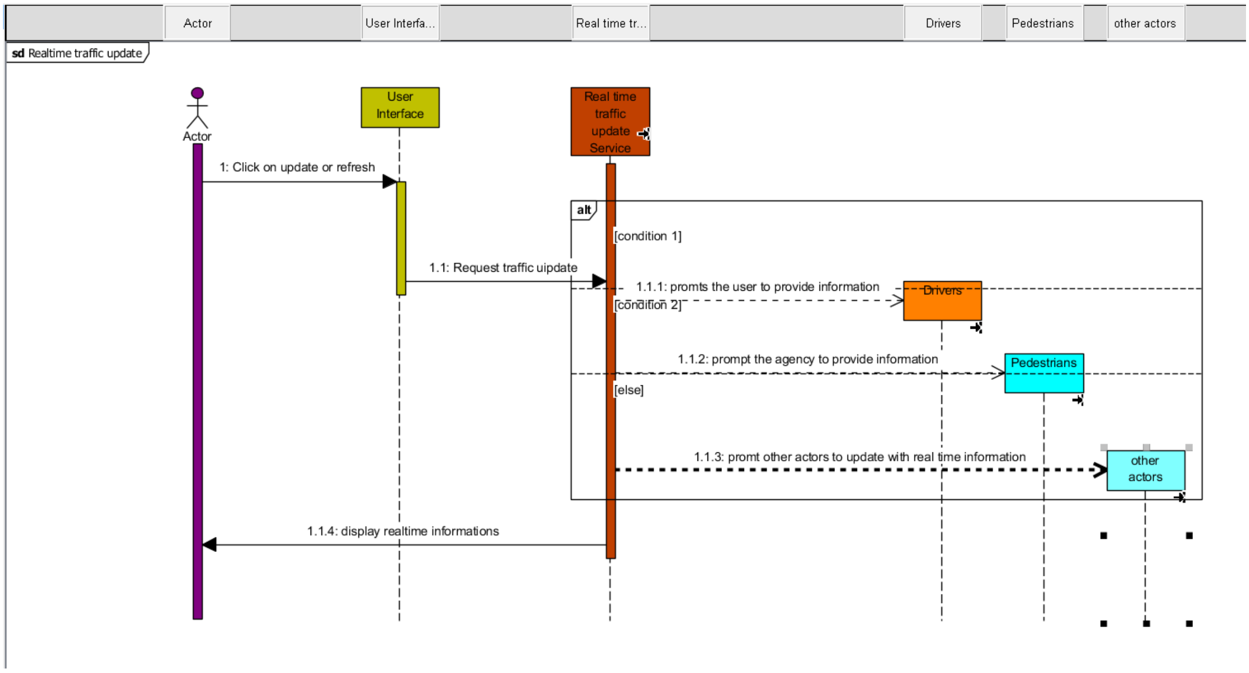
A. Sequence Diagram

Defintion

Why use a sequence diagram

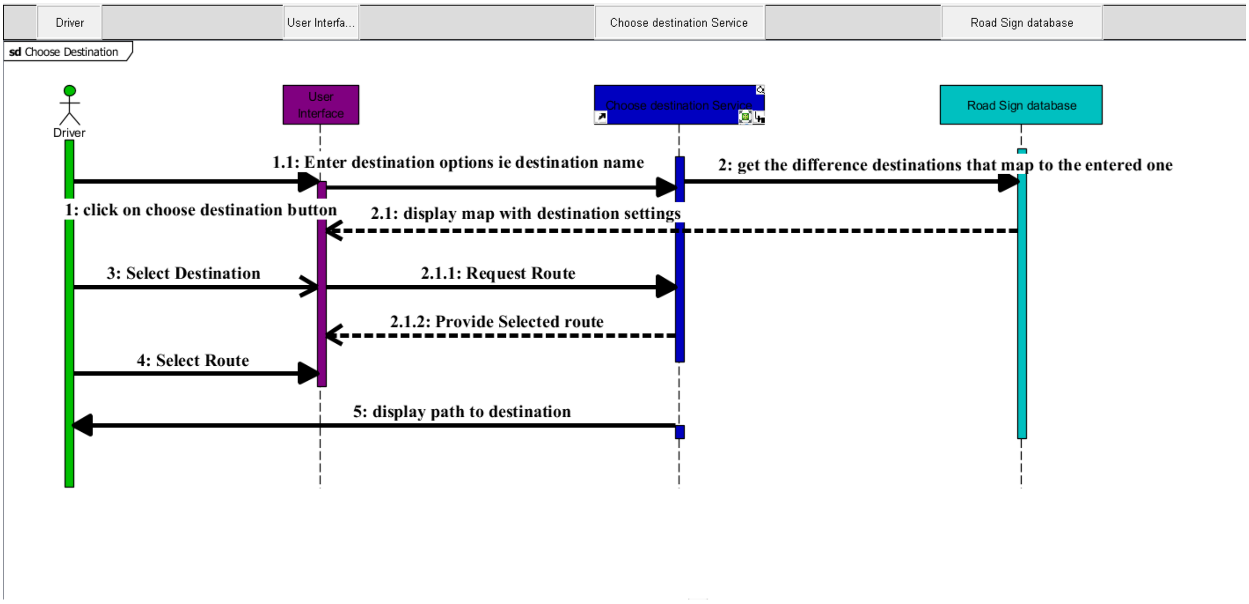
Various Sequence Diagrams in the system

Sequence Diagram 1: Get Real Time Traffic updates

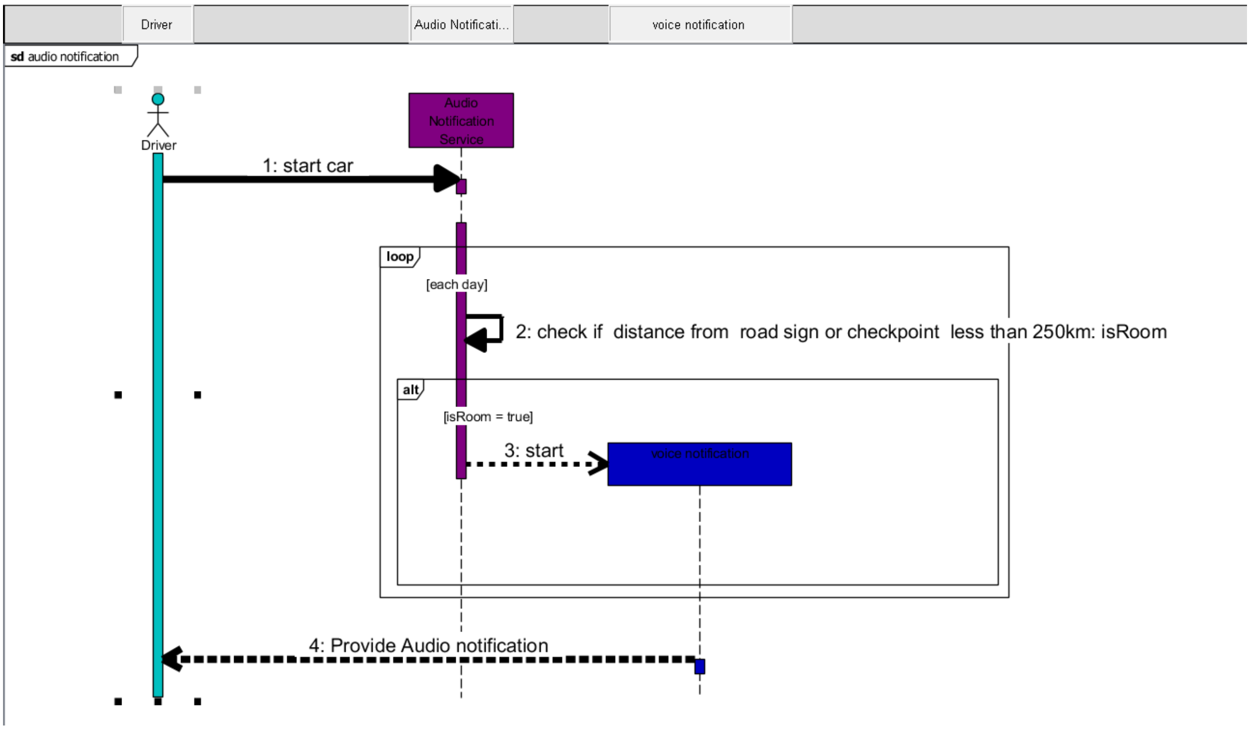

Actor
User Interfa...
Real time tr...
Drivers
Pedestrians
other actors
sd Realtime traffic update
Actor
User Interface
Real time traffic update Service
1: Click on update or refresh
alt
[condition 1]
1.1: Request traffic uipdate
1.1.1: promts the user to provide information
Drivers
[condition 2]
1.1.2: prompt the agency to provide information
Pedestrians
[else]
1.1.3: promt other actors to update with real time information
other actors
1.1.4: display realtime informations

Sequence Diagram 2: Choose Destination

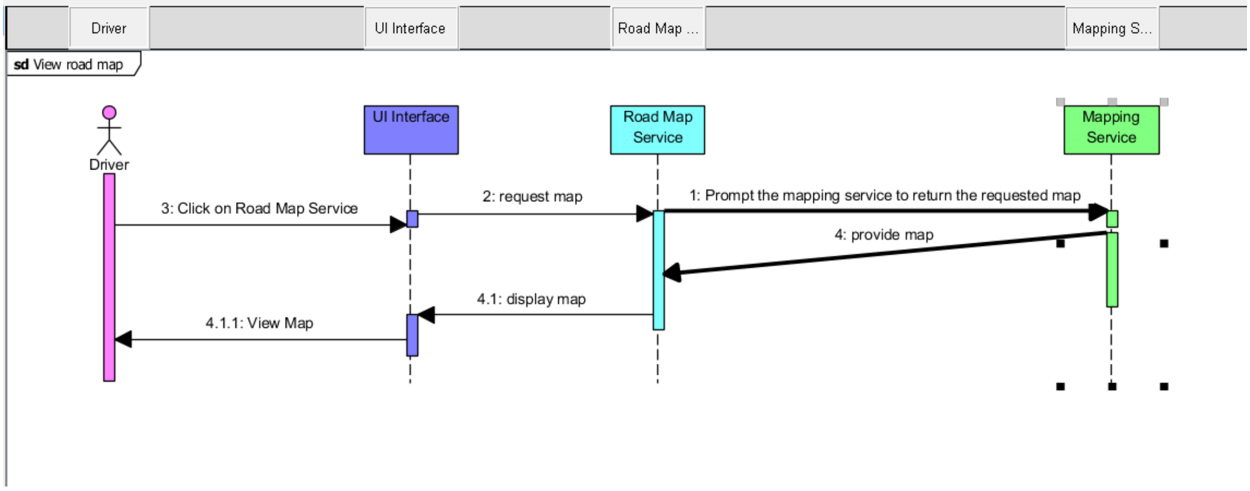
The sequence diagram below shows the interaction between the user and Safewaka in choosing the destination to reach his given destination


Driver
User Interfa...
Choose destination Service
Road Sign database
sd Choose Destination
Driver
User Interface
Choose destination Service
Road Sign database
1.1: Enter destination options ie destination name
2: get the difference destinations that map to the entered one
1: click on choose destination button
2.1: display map with destination settings
3: Select Destination
2.1.1: Request Route
2.1.2: Provide Selected route
4: Select Route
5: display path to destination

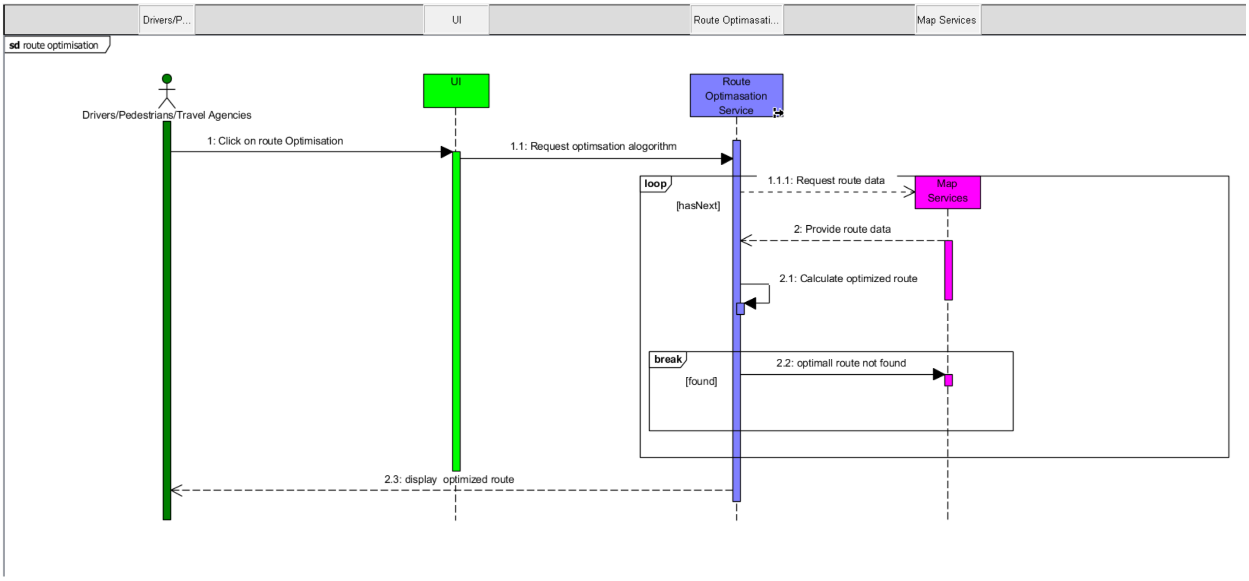
Sequence Diagram 3: Audio Notifications



Sequence Diagram 4: View Road map



Sequence Diagram5 : Route Optimisation


Drivers/P...
UI
Route Optimasati...
Map Services
sd route optimisation
Drivers/Pedestrians/Travel Agencies
UI
Route Optimasation Service
1: Click on route Optimisation
1.1: Request optimsation alogorithm
loop
[hasNext]
1.1.1: Request route data
Map Services
2: Provide route data
2.1: Calculate optimized route
break
[found]
2.2: optimall route not found
2.3: display optimized route

CONCLUSION

In summary, the comprehensive use of structural, behavioral, and interaction diagrams has been crucial in designing and documenting the road sign, road state, and notification app. The structural diagrams have defined the system's data model and core components, the behavioral diagrams have captured the intended dynamic behavior, and the interaction diagrams have provided insights into the collaboration and communication patterns between the different parts of the system. This holistic approach to system design ensures the final product will meet user requirements, adhere to best practices, and be readily maintainable and scalable over time.