**Labs 08 & 09: Draw Sequence diagram (System S1)**

**Lab Learning Outcomes (LLO)**

By completion of the lab the students should be able to:

1. Identify the scenarios for which the sequence diagrams are needed.
2. Identify the sequence of interactions between the classes in the Traffic Violation Monitoring System.
3. Identify the exchange of messages between objects in the system.
4. Identify how to represent dynamic behavior using a sequence diagram.
5. Gain insights into how the suggested solution processes a traffic violation.

**Lab Requirements**

For example, [StarUML](https://staruml.io/), [Lucidchart](https://www.lucidchart.com/pages/), [penpot](https://penpot.app/) and [Figma](https://www.figma.com/) etc

**Tasks to Do:**

The focus of this lab will be on capturing the interactions between the system components during the process of detecting and handling a traffic violation. This will help students visualize the dynamic behavior of the system. This lab will provide you with experience in visualizing and analyzing system requirements using Sequence diagram modeling. You may start by considering the followings directions:

**1: Identify Interactions**

1. Review the problem description and suggested solution provided. (**Refer to previous labs**)
2. Identify the key Classes involved in the traffic violation system.
3. Determine the interactions and messages exchanged between these classes.

**2: Create a Sequence Diagram**

1. Select a suitable tool for creating sequence diagrams (as given above).
2. Create a sequence diagram for the process of detecting and handling a traffic violation.
3. Place the identified components (objects) on the diagram.
4. Represent the interactions between objects using lifelines and messages.

**3: Sequence of Interactions**

1. Begin the sequence diagram with the initiation of a traffic violation detection.
2. Show the messages exchanged between the TrafficCamera, TrafficData, Violation, Driver, and Notification components.
3. Include both synchronous and asynchronous messages to reflect real-world interactions.

**4: Include Conditions and Decisions**

1. Represent any conditions or decisions that determine the flow of interactions.
2. If needed, use combined fragments (such as alt or opt) to illustrate optional or alternative paths.

**5: Review and Documentation**

1. Review your sequence diagram to ensure the correctness of interactions and message flow.
2. Add proper annotations and descriptions to clarify the purpose of each message.
3. Document the diagram with a title, date, and brief explanation.

Here, we will explain a sequence diagram for the Traffic Violation Monitoring System based on the defined classes, attributes, methods, and relationships. Let's start by breaking down the sequence of interactions and messages between the classes. Following is a simplified scenario for the purpose of this example:

**Scenario:** Detection and Handling of a Traffic Violation

**Classes Involved: Interactions and Messages**

1. **TrafficCamera:** Initiates the violation detection process.

**Messages/Methods:**

* + **startDetection():** Initiates the detection process**.**
  + **captureImage():** Captures an image of the violating vehicle.

1. **TrafficData: Receives captured data from TrafficCamera.**

**Messages/Methods:**

* + **receiveData(image):** Receives the captured image from TrafficCamera.

1. **Violation:** Analyzes the captured data and determines if a violation occurred**.**

**Messages/Methods:**

* + **analyzeData(image):** Analyzes the captured image for violations.
  + **isViolation():** Checks if the captured data indicates a violation.

1. **Driver:** Represents the driver associated with the violating vehicle.

**Messages/Methods:**

* + **identifyDriver(image):** Extracts driver information from the captured image.

1. **Notification:** Notifies the driver about the violation.

**Messages/Methods:**

* + **notifyViolation(driver, violationType):** Sends a notification to the driver regarding the violation.

**Sequence of Interactions (Simplified):**

1. TrafficCamera initiates the detection process by calling the **startDetection()** method.
2. TrafficCamera captures an image using the **captureImage()** method.
3. TrafficData receives the captured image through the **receiveData(image)** method.
4. Violation analyzes the received data using the **analyzeData(image)** method.
5. Violation checks if a violation occurred by calling the **isViolation()** method.
6. If a violation is detected, Driver is identified by calling the **identifyDriver(image)** method.
7. Notification sends a notification to the Driver using the **notifyViolation(driver, violationType)** method.

**NOTE:**

1. Please note that the methods used in the sequence diagram should match the methods defined in the class diagram. If the above methods do not match the methods in the class diagram then either change the methods shown in the class diagram or include the above methods in the class diagram.
2. Please note that this is a simplified example for the purpose of understanding the sequence of interactions and messages. In a real-world scenario, there would be more details, conditions, and decisions involved in the interactions. **Use the outlined interactions and messages as a starting point to create your sequence diagram, adding more complexity and details as guided by the lab instructor**.

**QUESTIONS TO ANSWER:**

1. Draw the above sequence of interaction in the sequence diagram using the tools. You must provide the written description for your sequence diagram.
2. Extend the above sequence diagram to a scenario, “Driver Disputes a Violation” where the classes involved are Driver, Notification and Violation.
3. Show the following scenarios.
   1. Dispute is **accepted** by the relevant authority (may be represented by a class).
   2. Dispute is **rejected** by the relevant authority (may be represented by a class).
      * 1. The driver pays penalty, if the dispute is rejected by the relevant authority. You must show any additional classes, attributes and methods involved in the Class Diagram and then use in the sequence diagram. For instance, for the **Payment** mechanism, you will need an additional class. Let’s explore yourself the remaining.
4. Do you think, the above scenarios will have any change in their sequence diagrams for the following violations:
   1. Over speeding
   2. Crossing red lights
   3. Changing lanes abruptly or without indicators
   4. Not wearing seat belts
   5. Driving erratically or zigzagging
   6. Stopping beyond the stop line at red lights

If yes, reflect those changes in the diagram otherwise, justify your answer with proper reasoning.

**Examples**

You may also get help from the following samples produced by the students. These samples may not be complete and accurate, so **you are required to complete them and identify any mistakes/errors or space for improvements or changes required.**

**Sample 1**

A diagram of a diagram

Description automatically generatedA diagram of a process

Description automatically generated

**Sample 2**

Diagram

Description automatically generated with low confidence

Timeline

Description automatically generated