



# **CYBER PHYSICAL TRAFFIC CONTROL SYSTEM**

JACOB JOSE

SAI VARUN VAKA

KANAV SHARMA

# THE PROBLEM

Traffic control systems operate based on pre-defined time intervals.

## WHY IS IT A PROBLEM

- While this approach may be simple and cost-effective, it inevitably leads to excess congestion due to its inability to adapt to changing traffic patterns or road conditions.
- Manual preempt lanes for emergency vehicles.

# PROPOSED SOLUTION

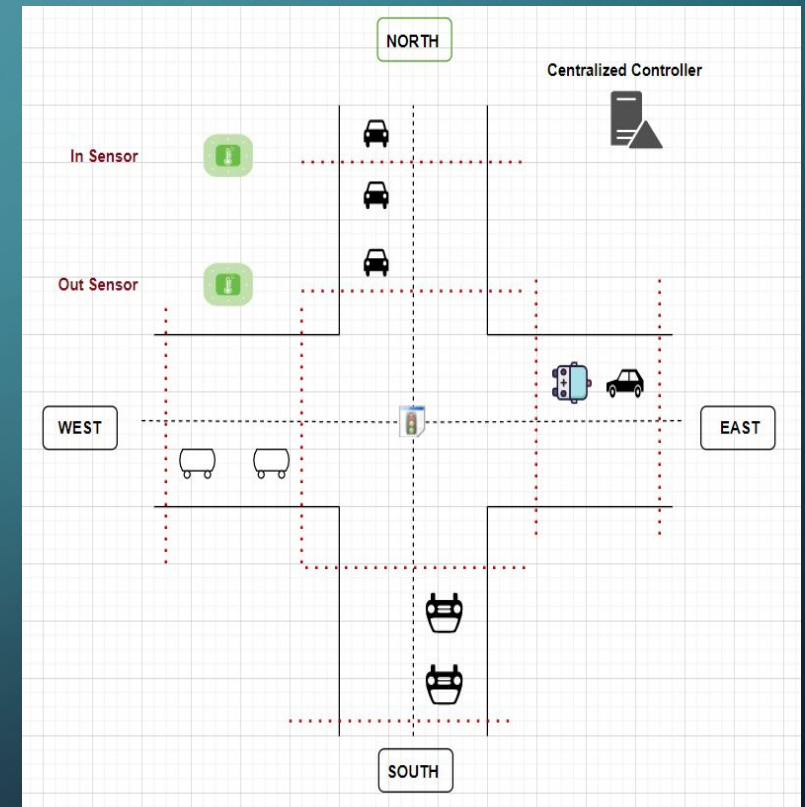
A CPS that utilizes a centralized traffic controller to make real-time traffic-light-change decisions based on data received from implemented road sensors.

- Input: Real time traffic data from road sensors.
- Controller: Processing data.
- Output: Changes to traffic lights.

System will be able to accommodate lane preemption for emergency vehicles.

# FEATURES

- Road sensors (one for each road) to collect count of incoming and outgoing vehicles.
- Emergency Vehicle sensors to detect EV.
- Traffic monitoring unit to collect the vehicle data & perform computing to make a decision.
- Webster Algorithm.



# ASSUMPTIONS

- All Vehicles will move in a straight road. No vehicle will take a left or right turn at the intersection.
- Vehicles do not break down while moving and will not take a U turn before the intersection.
- Each lane is having individual sensor and summation of cars calculated by each lane will be considered as input.
- Emergency Vehicles have a special radio frequency transmitter for identification. Radio frequency sensors will identify emergency vehicles through the transmitter.
- Only one emergency vehicle can be at the intersection at a time.
- The North-South bound traffic signal is Green at the initial state & East West bound traffic signal is Red at the initial state.

# HOW IS IT A CPS

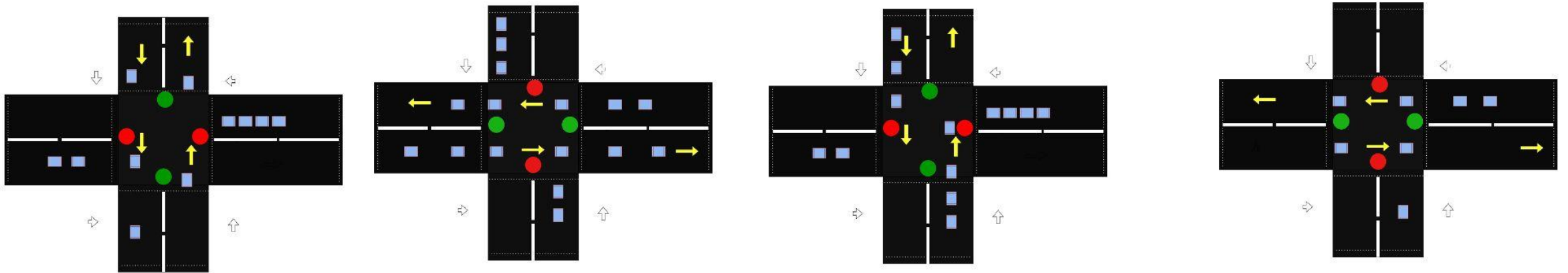
- Reactive - When a emergency vehicle arrives our program takes the input and sends an output to turn on the green light for emergency vehicle specific lane by interacting reactively via inputs and outputs.
- Concurrency - Vehicle count is calculated from one of the component while the other component checks for emergency vehicle and calculates green time.

# HOW IS IT A CPS Contd...

- Feedback - Application interacts with the environment with the sensor by sending the input data and the component gives the output with the green time. When the green time ends, our application again checks for the count of vehicles by creating a feedback loop.
- Real Time - Our project relies on Webster's algorithm which takes the vehicle count on both the lanes and performs a computation in real time.
- Safety Critical - Application deals with the emergency vehicles, any wrong information or computation could result in major damage.

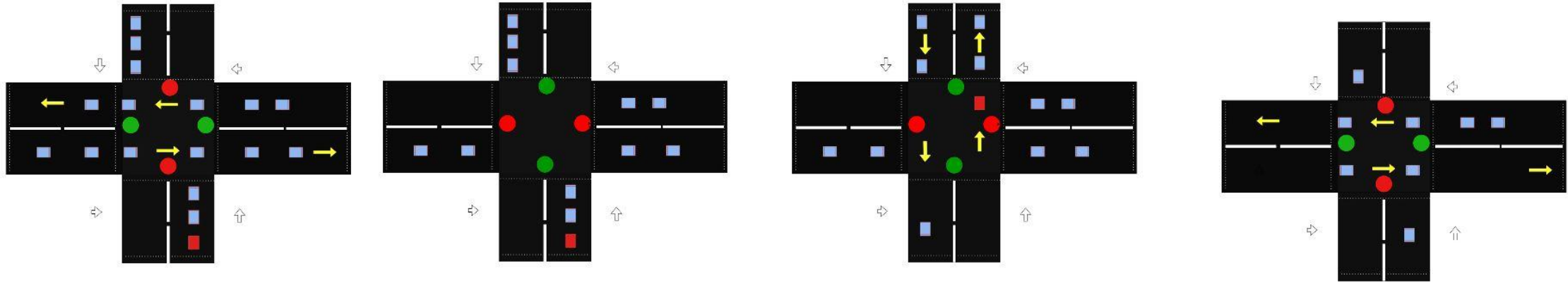


# Scenario - Normal Operation

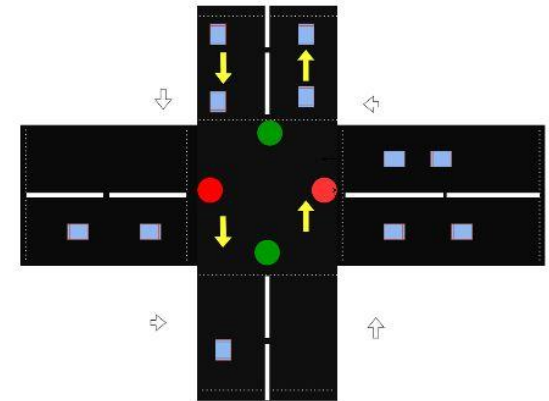
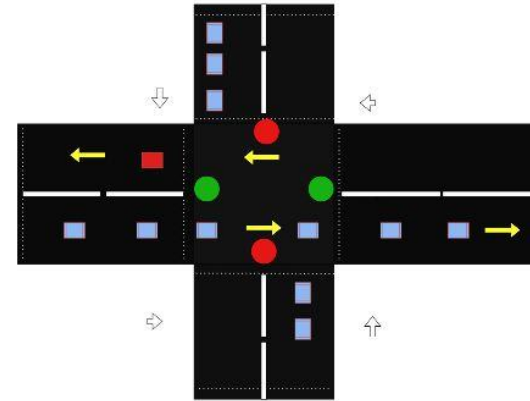
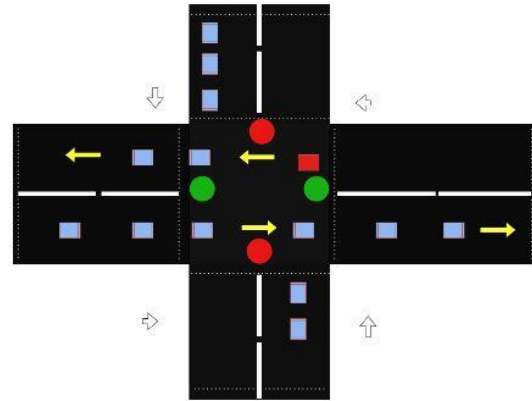
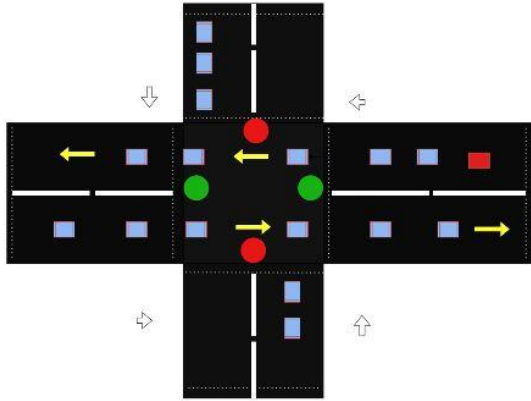




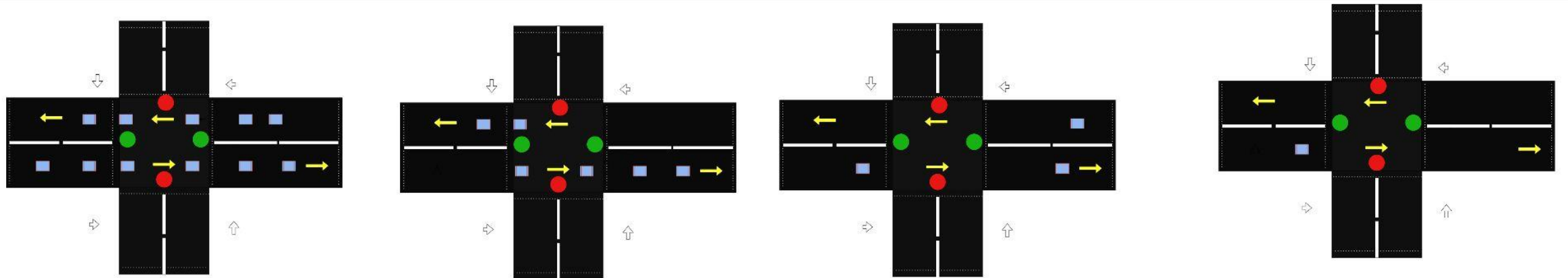
# Scenario - Emergency Vehicle from South



# Scenario - Emergency Vehicle from East



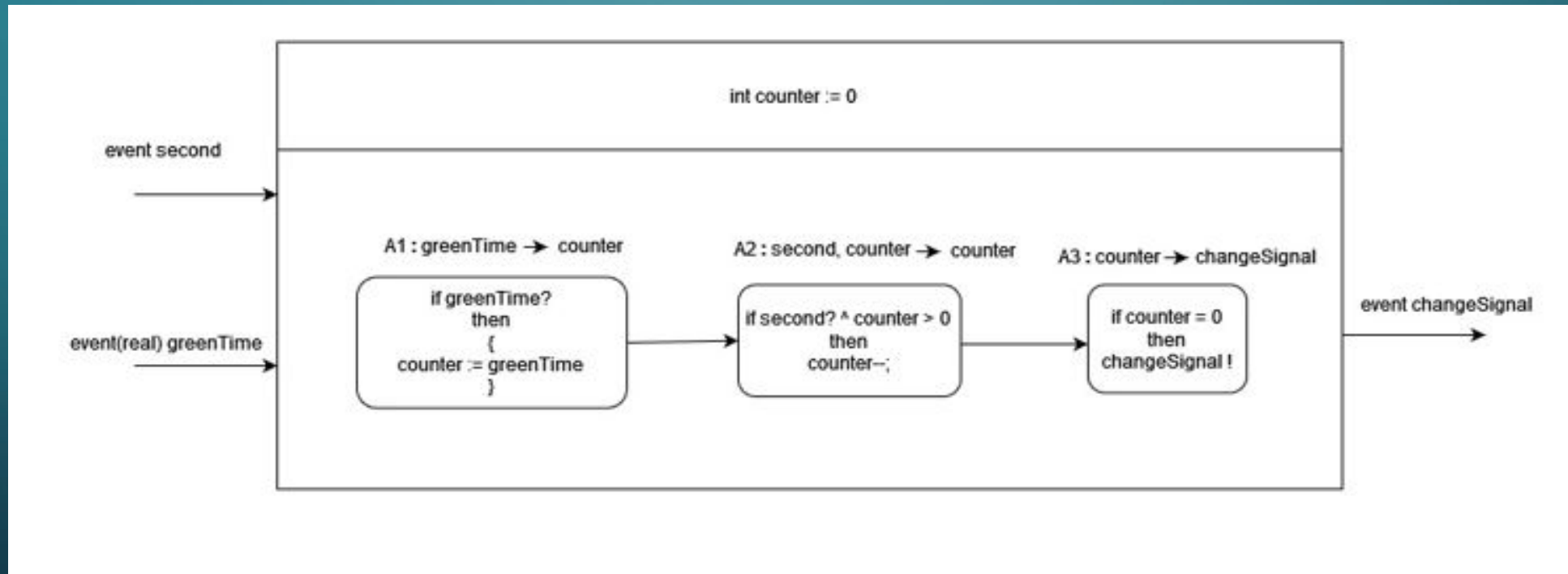
# Scenario - No traffic in North-South



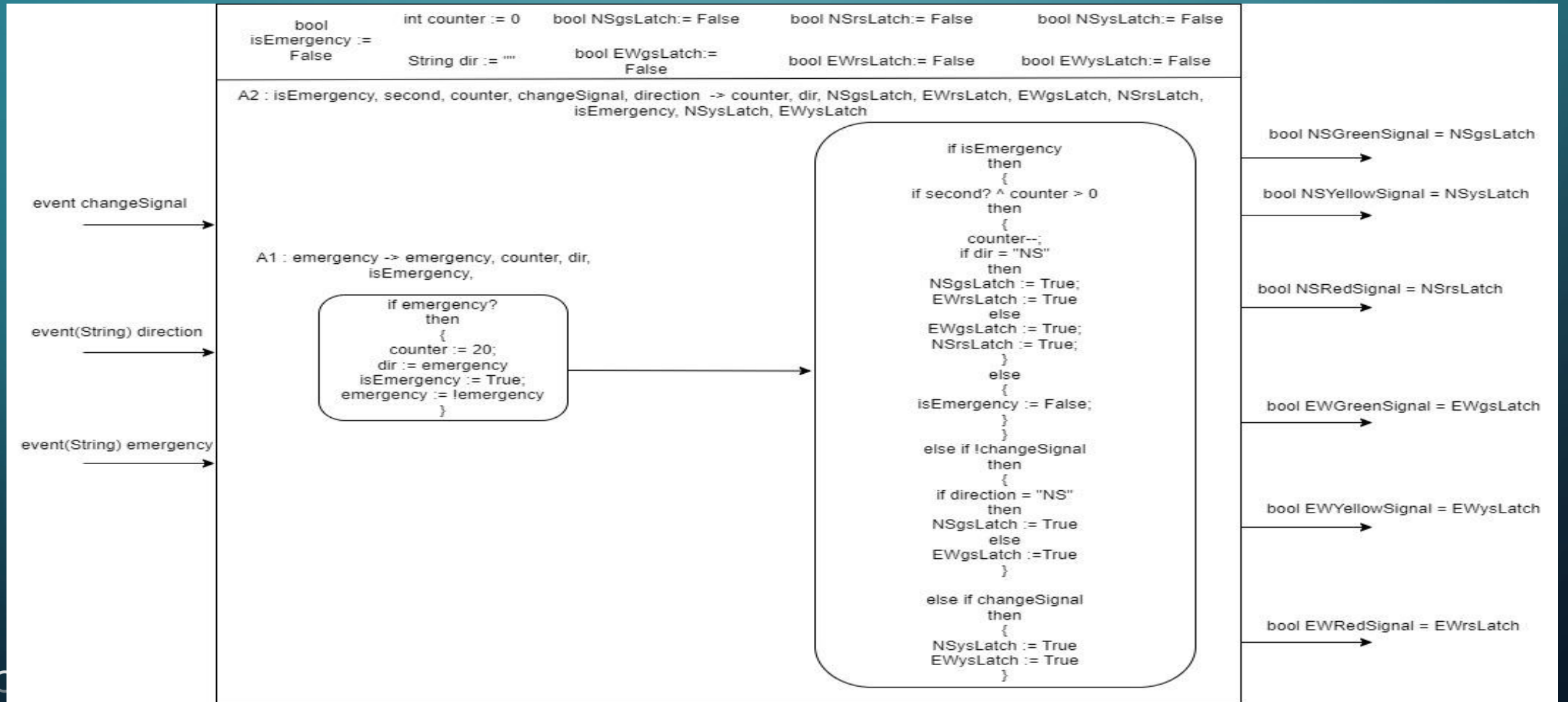
# BLOCK SRC

[View Block SRC](#)

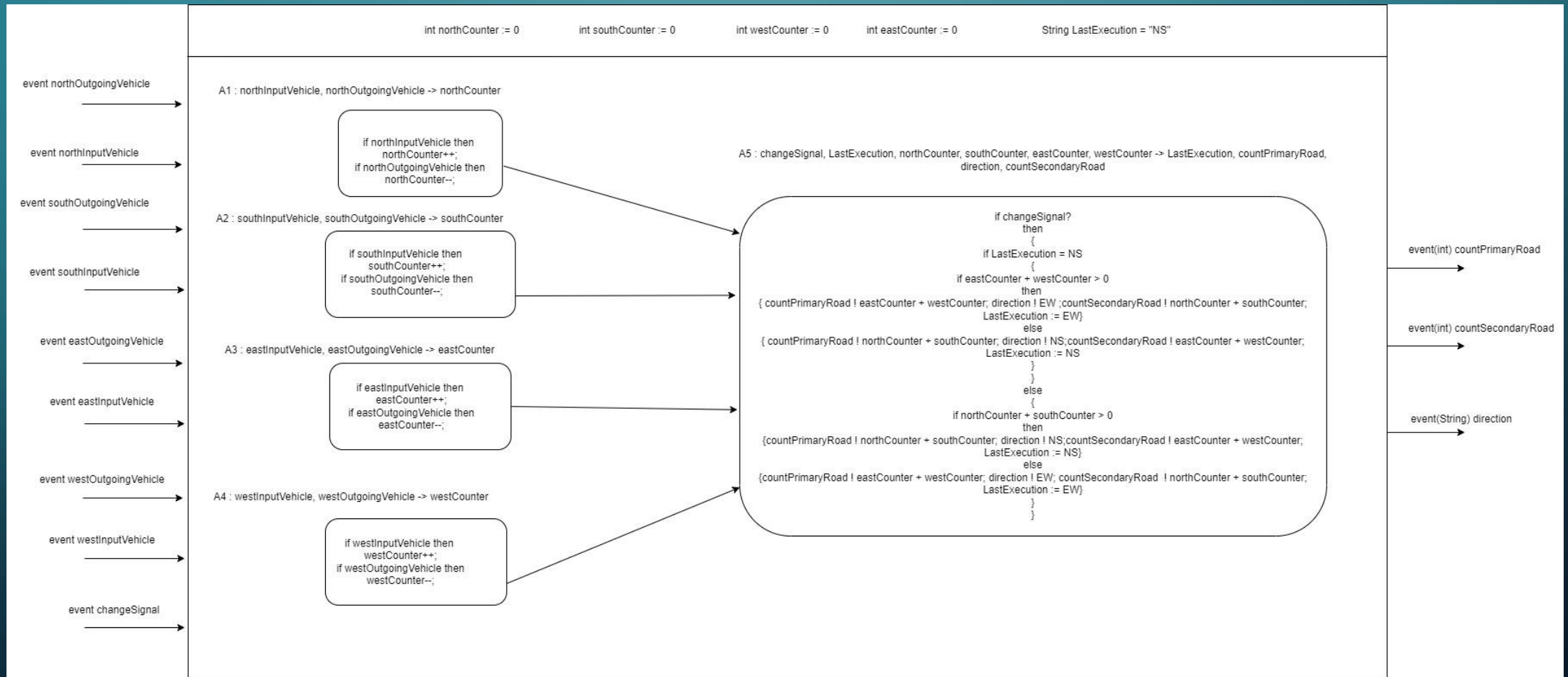
# ATOMIC SRC - DATA PROCESSING



# ATOMIC SRC - LIGHT PROCESSING



# COMPOSITE SRC - VEHICLE COUNTER

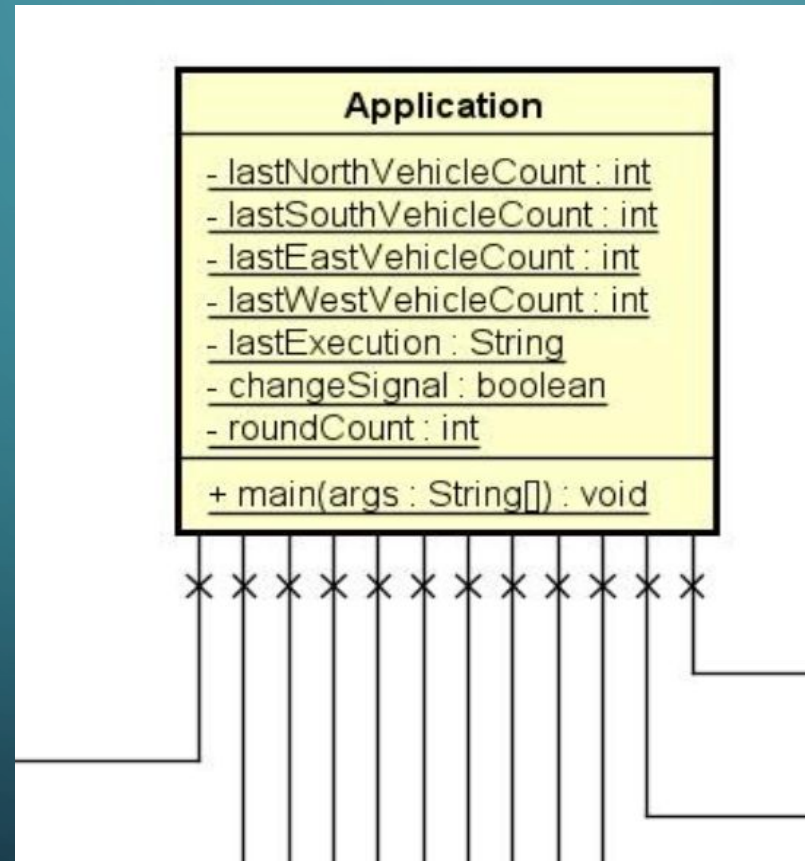




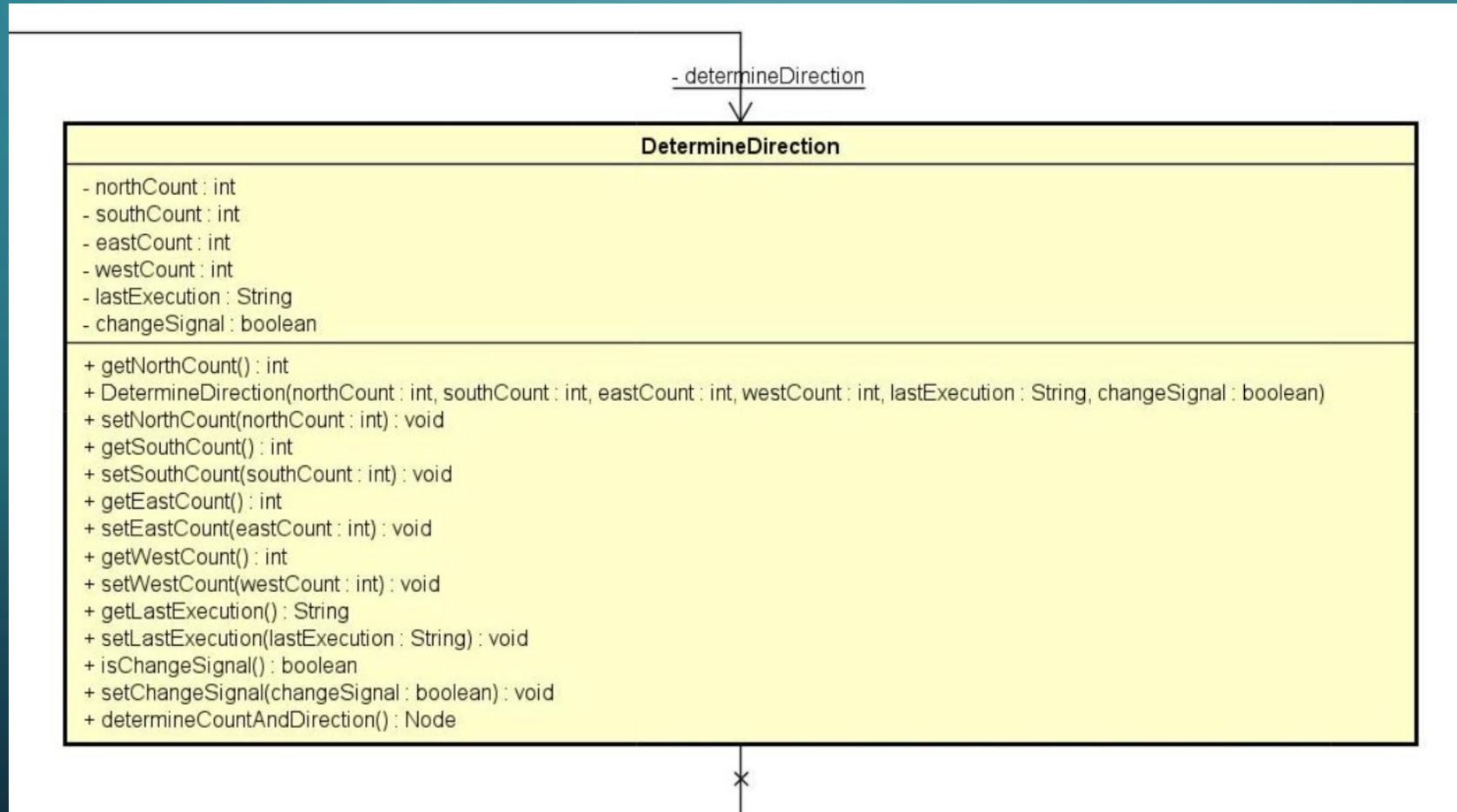
# UML Class Diagram - Composite

[View Composite UML](#)

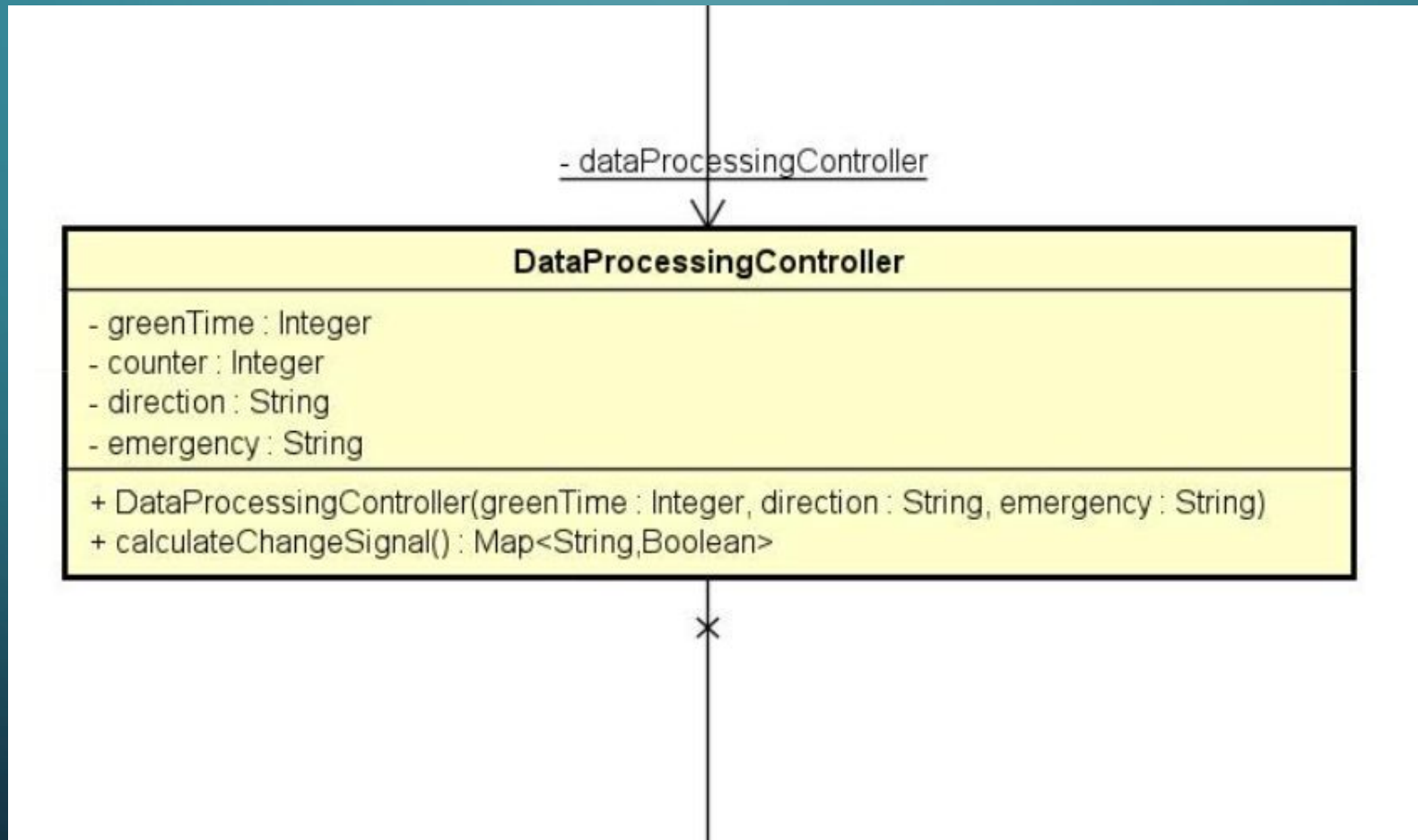
# UML - Application



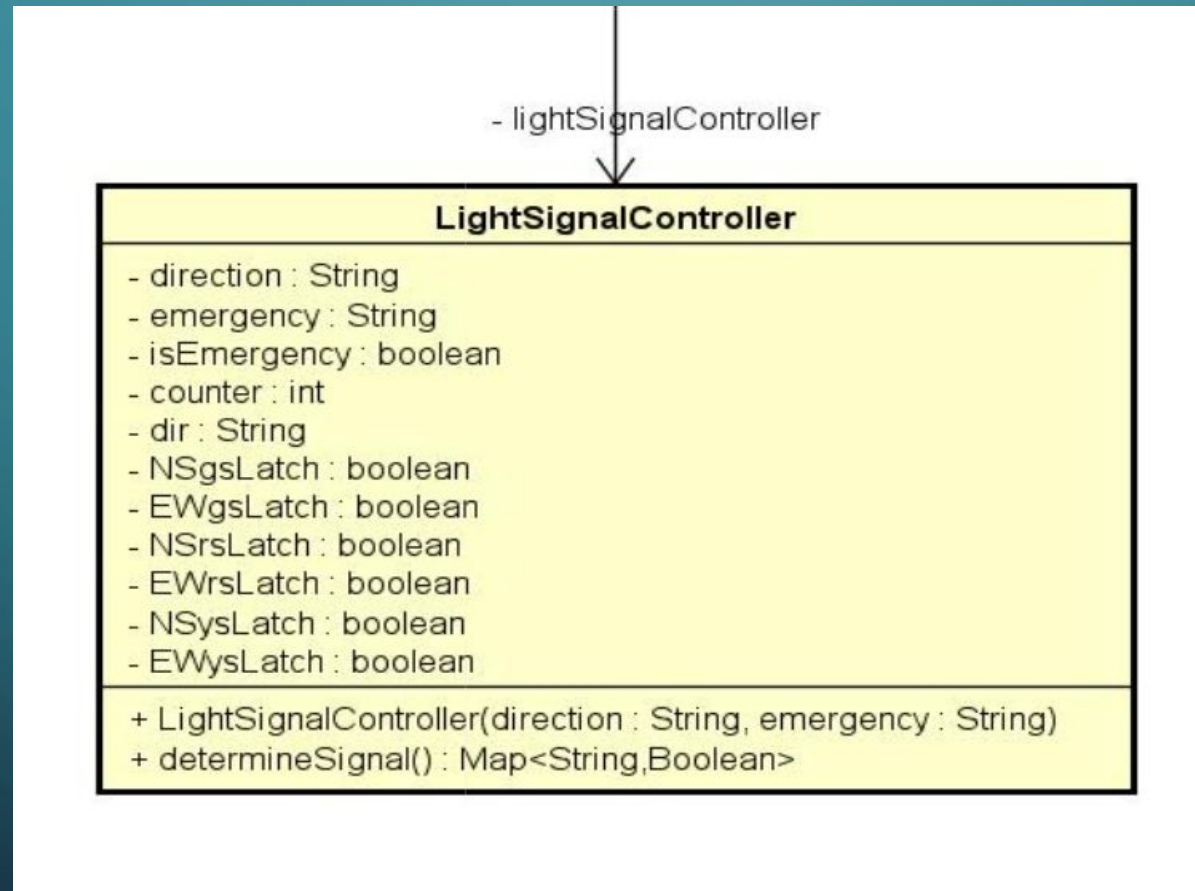
# UML - VEHICLE COUNTER



# UML - DATA PROCESSING

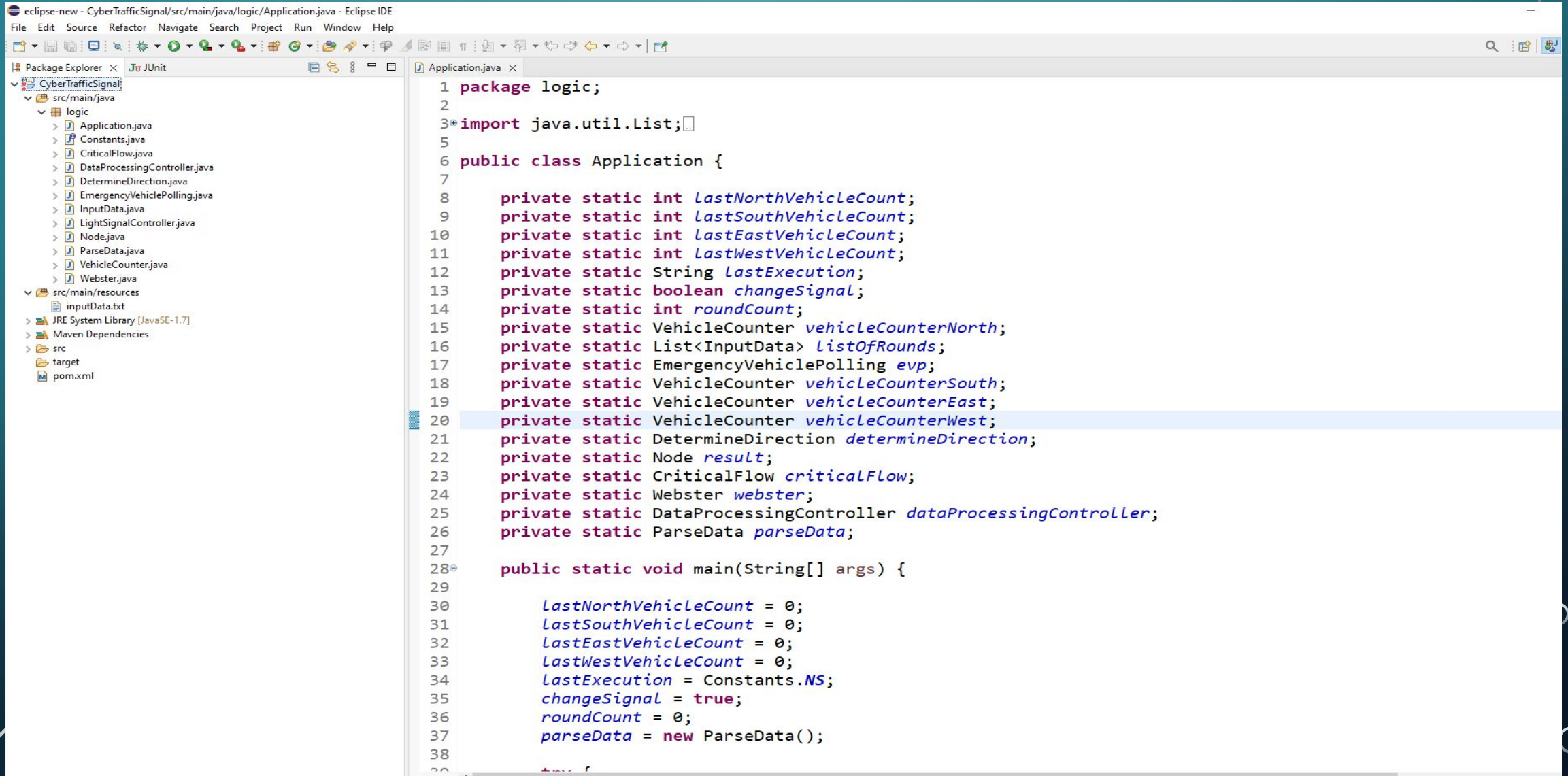


# UML - LIGHT PROCESSING





# IMPLEMENTATION



The screenshot displays the Eclipse IDE interface. The Package Explorer on the left shows the project structure for 'CyberTrafficSignal', including a 'logic' package with various Java files and a 'src/main/resources' directory containing 'inputData.txt'. The main editor window shows the code for 'Application.java'.

```
1 package logic;
2
3 import java.util.List;
4
5
6 public class Application {
7
8     private static int lastNorthVehicleCount;
9     private static int lastSouthVehicleCount;
10    private static int lastEastVehicleCount;
11    private static int lastWestVehicleCount;
12    private static String lastExecution;
13    private static boolean changeSignal;
14    private static int roundCount;
15    private static VehicleCounter vehicleCounterNorth;
16    private static List<InputData> listOfRounds;
17    private static EmergencyVehiclePolling evp;
18    private static VehicleCounter vehicleCounterSouth;
19    private static VehicleCounter vehicleCounterEast;
20    private static VehicleCounter vehicleCounterWest;
21    private static DetermineDirection determineDirection;
22    private static Node result;
23    private static CriticalFlow criticalFlow;
24    private static Webster webster;
25    private static DataProcessingController dataProcessingController;
26    private static ParseData parseData;
27
28    public static void main(String[] args) {
29
30        lastNorthVehicleCount = 0;
31        lastSouthVehicleCount = 0;
32        lastEastVehicleCount = 0;
33        lastWestVehicleCount = 0;
34        lastExecution = Constants.NS;
35        changeSignal = true;
36        roundCount = 0;
37        parseData = new ParseData();
38
39    }
```

# FRAMEWORKS & TOOLS

- Eclipse
- GIT
- Astah
- Draw.io
- Apache Maven
- IntelliJ IDE
- Java



# DEMO

CYBER PHYSICAL TRAFFIC CONTROL SYSTEM

JACOB JOSE

SAI VARUN VAKA

KANAV SHARMA

# REFERENCES

- <https://www.apsed.in/post/traffic-signal-design-webster-s-formula-for-optimum-cycle-length>
- R. Alur, (2015), Principles of Cyber-Physical Systems, MIT Press.
- G. Booch, et al., (2007), Object Oriented Analysis and Design (OOAD), 3rd Ed., Addison Wesley.
- Java Platform, Standard Edition (Java SE), (2019), <https://www.oracle.com/java/technologies/java-se.html>.
- OMG 2012. “Unified Modeling Language version 2.5.1”. <https://www.omg.org/spec/UML/2.5.1/>.

# Questions

