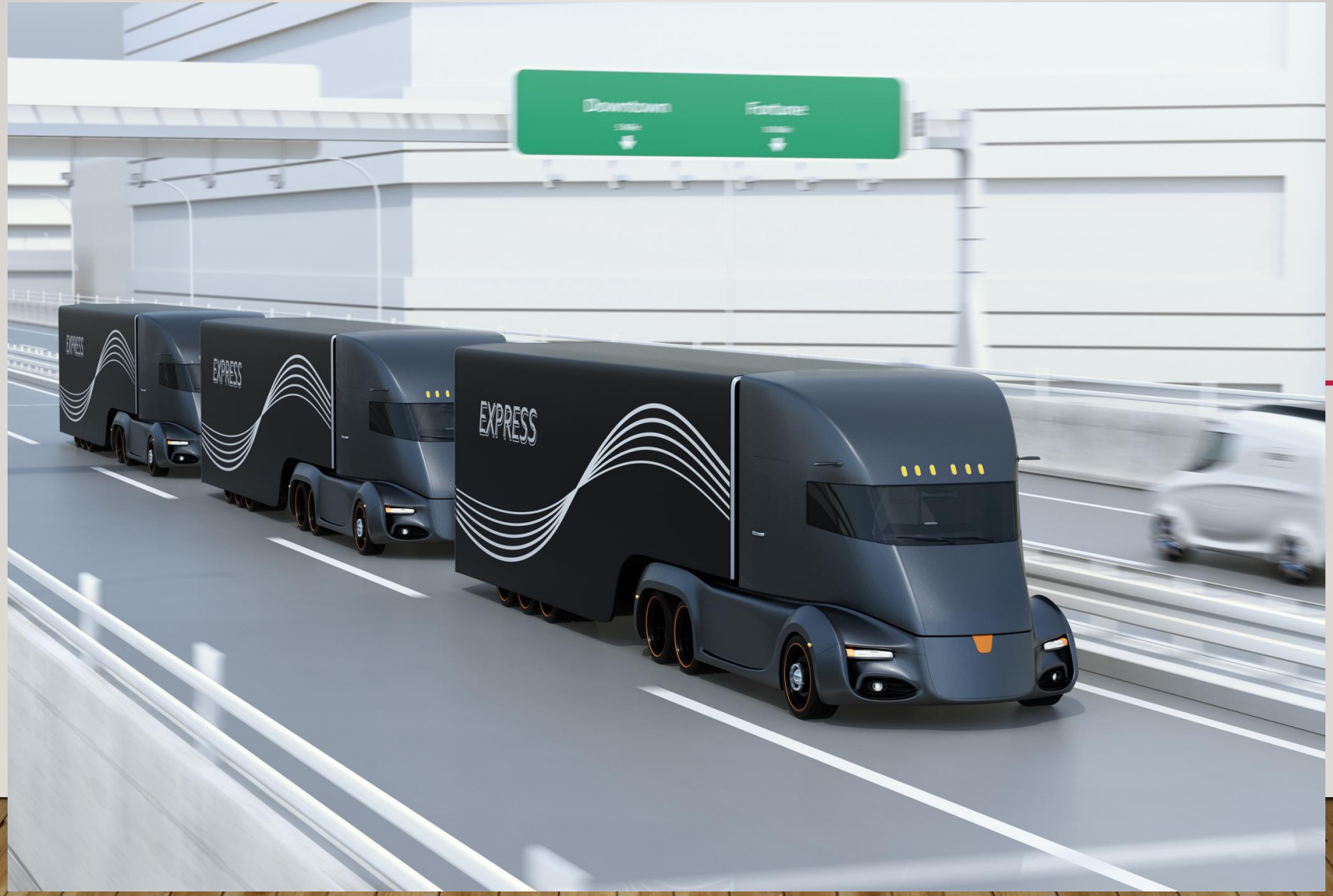
TRUCK PLATOONING



THE CHARACTERISTICS IN TRUCKS PLATOONING

- Reactive systems

The trucks keep detecting whether there are objects in the front or not. If the trucks detected objects, they will behave differently based on different objects.

- Real-time systems

In truck platooning, for example, the system can suddenly brake.

- Continuous systems

The trucks can increase or decrease the speed. But the speed will be continuously increasing or decreasing.

THE CHARACTERISTICS IN TRUCKS PLATOONING

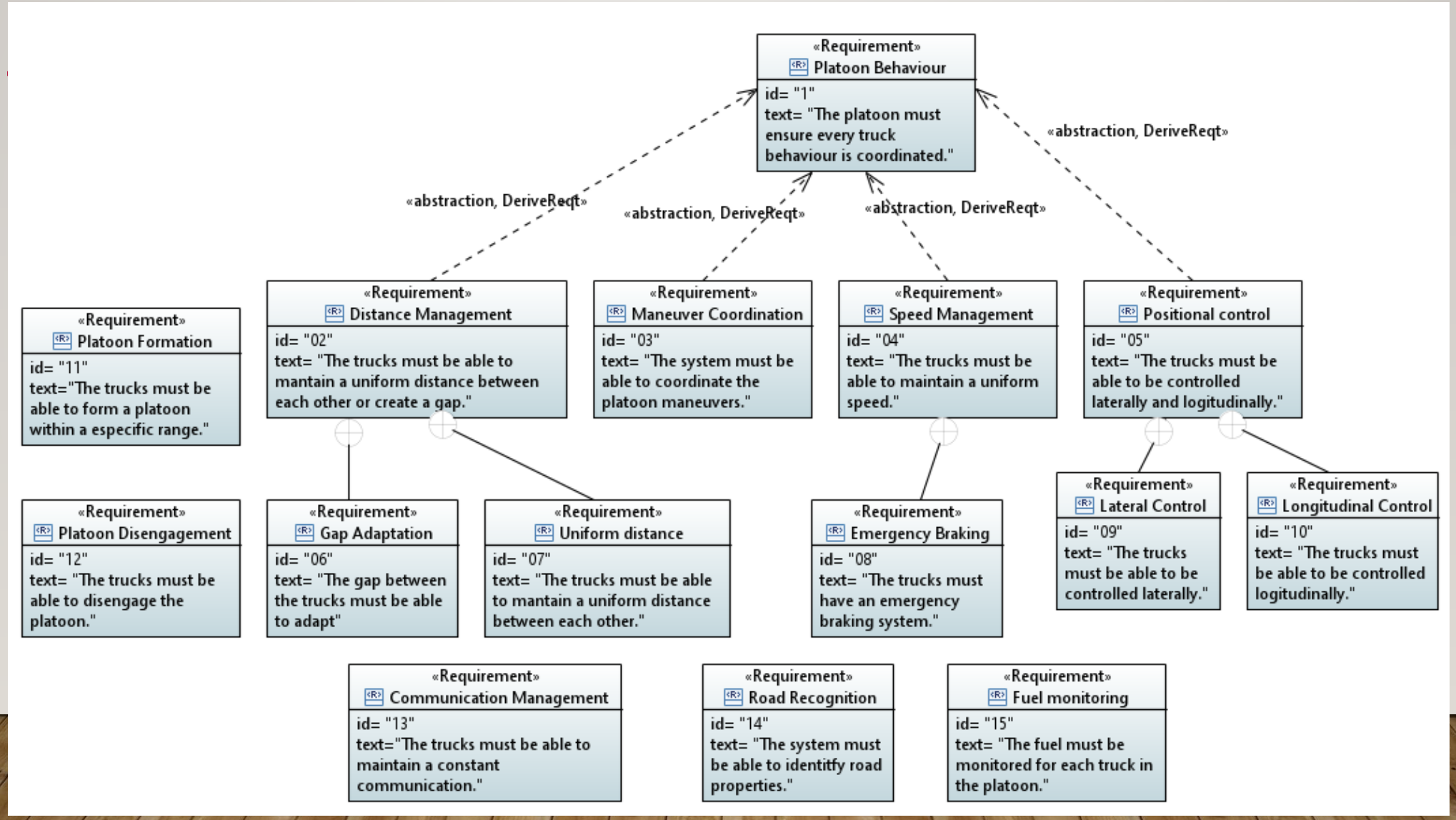
- Dependable systems

Because the truck platooning can save costs and improve the road safety, more and more companies now are applying this technique to their own products. Also, truck platooning is co-funded by the EU, which means is supported by the government.

- Distributed systems

While the trucks are driving, the trucks can still do other things such searching for the parking spaces, opening gap or closing gap at the same time.

SPECIFICATION OF THE ANALYSIS MODEL REQUIREMENTS DIAGRAM



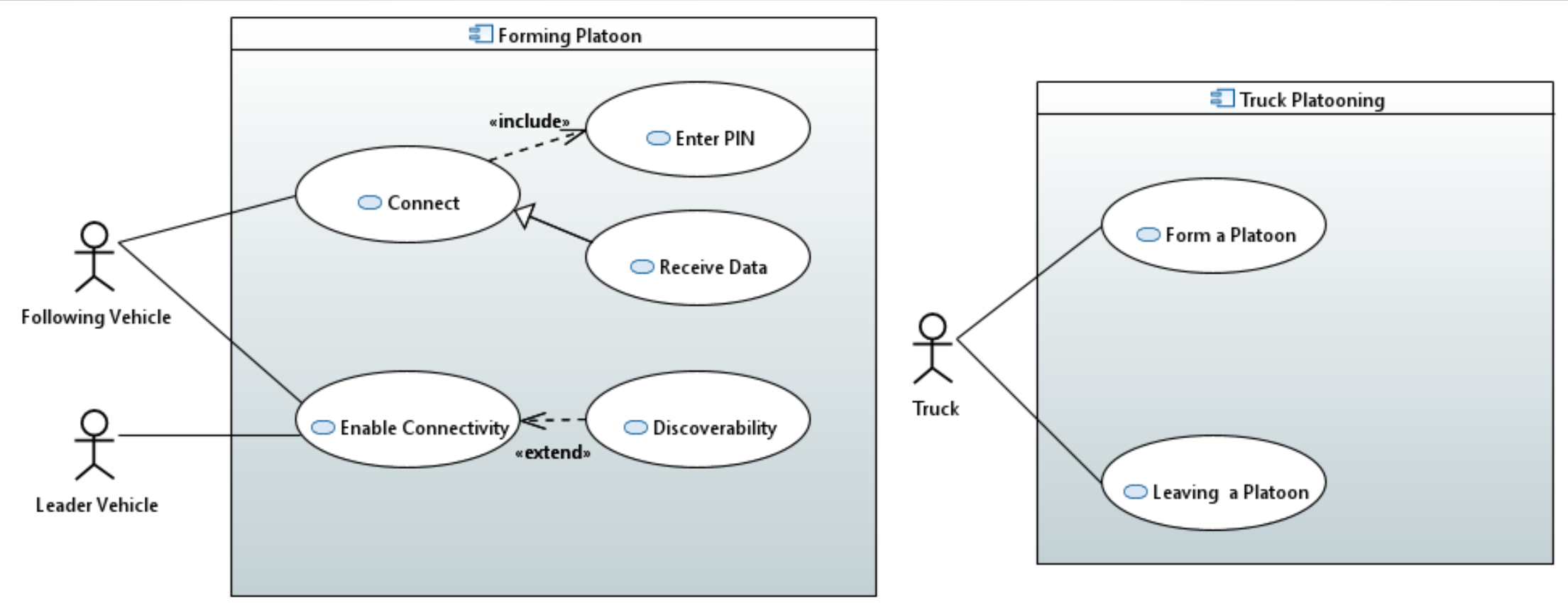
REFINED USE CASES

Refine three use cases with activity diagrams

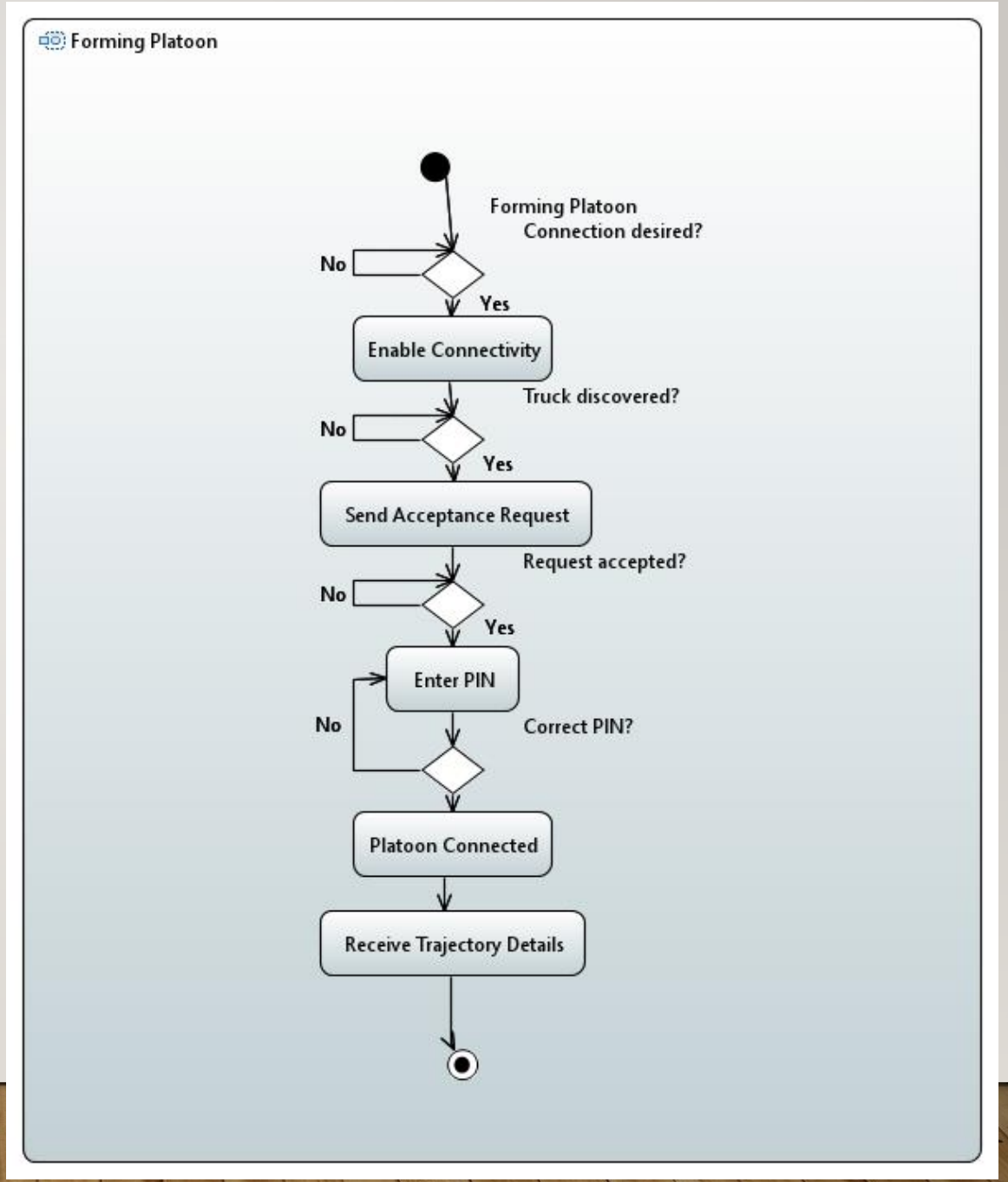
- Platoon Formation
- Gap Adaptation & Emergency braking using Object Detection
- Parking systems

PLATOON FORMATION

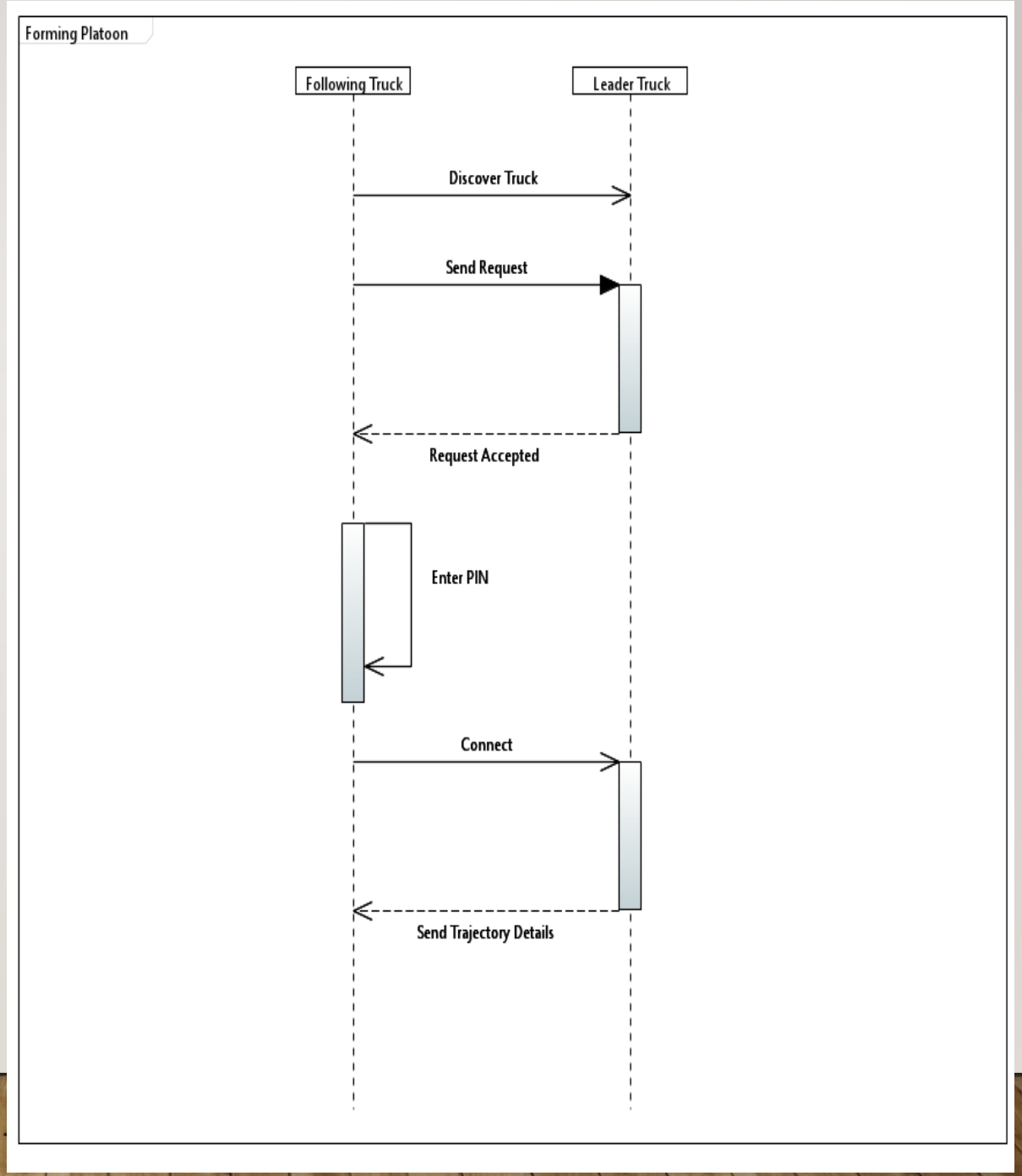
USE CASE DIAGRAM



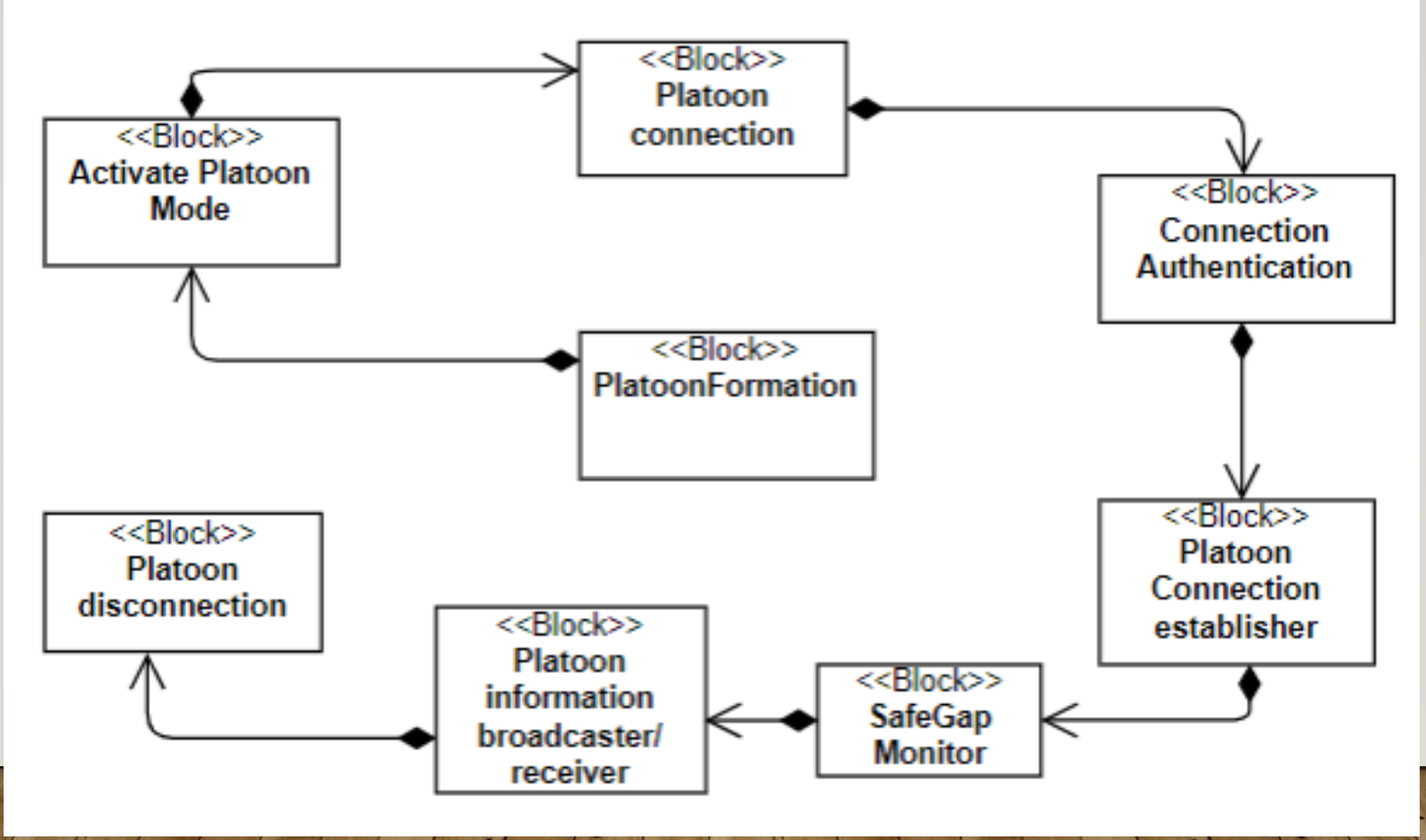
ACTIVITY DIAGRAM



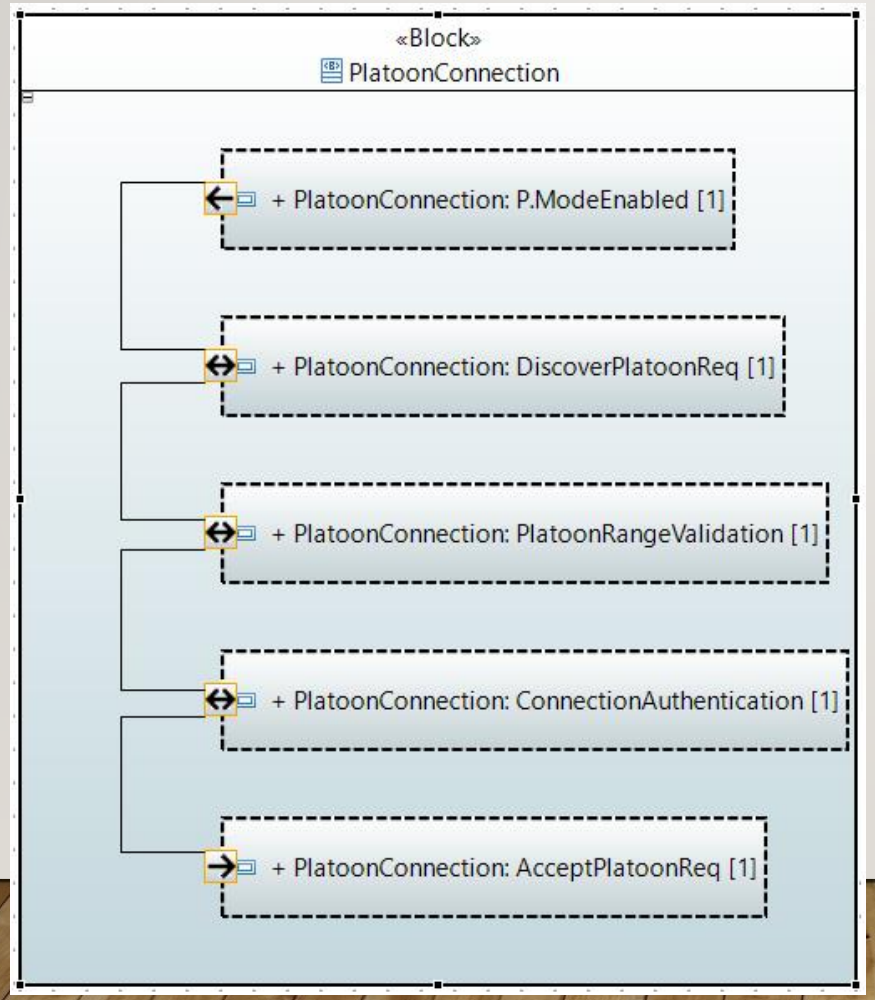
SEQUENCE DIAGRAM

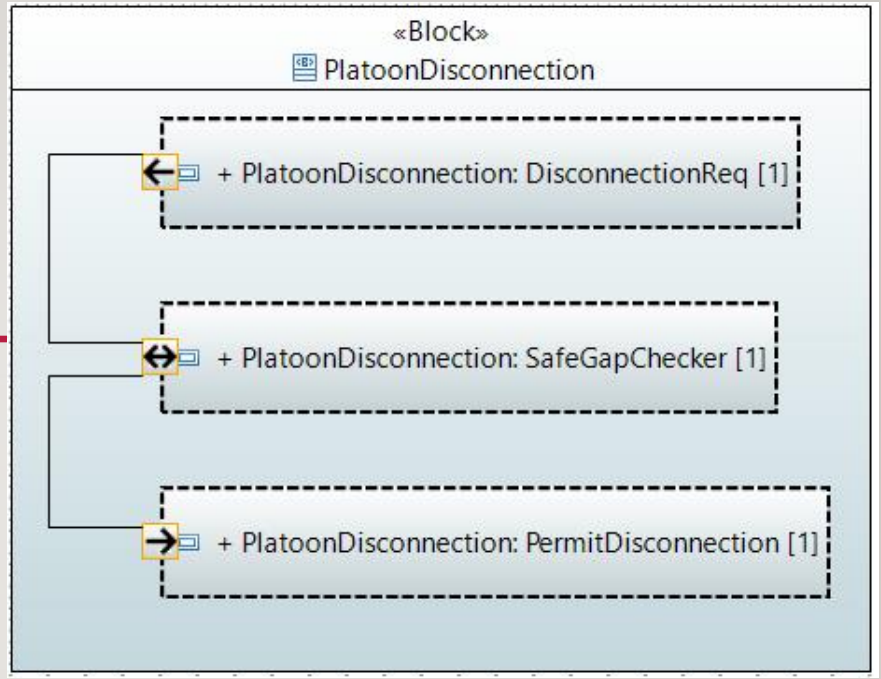


BLOCK DIAGRAM

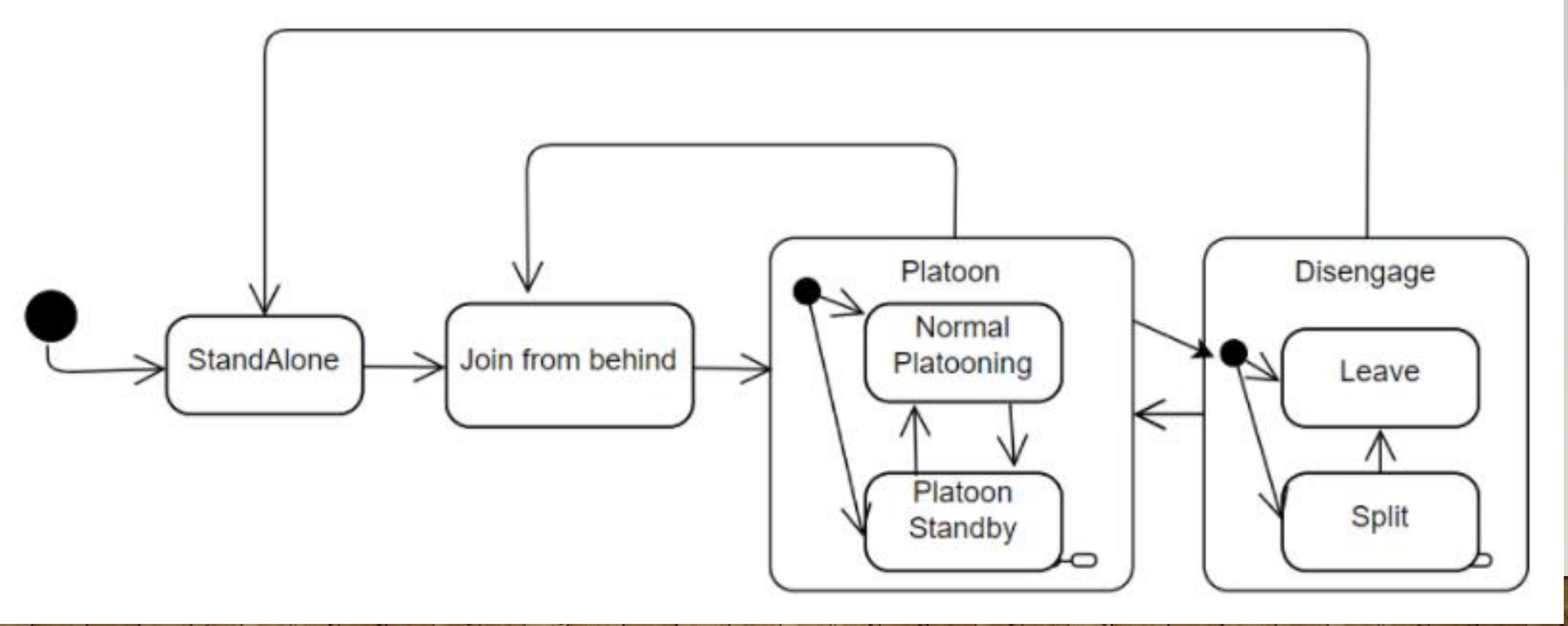


INTERNAL BLOCK DIAGRAM



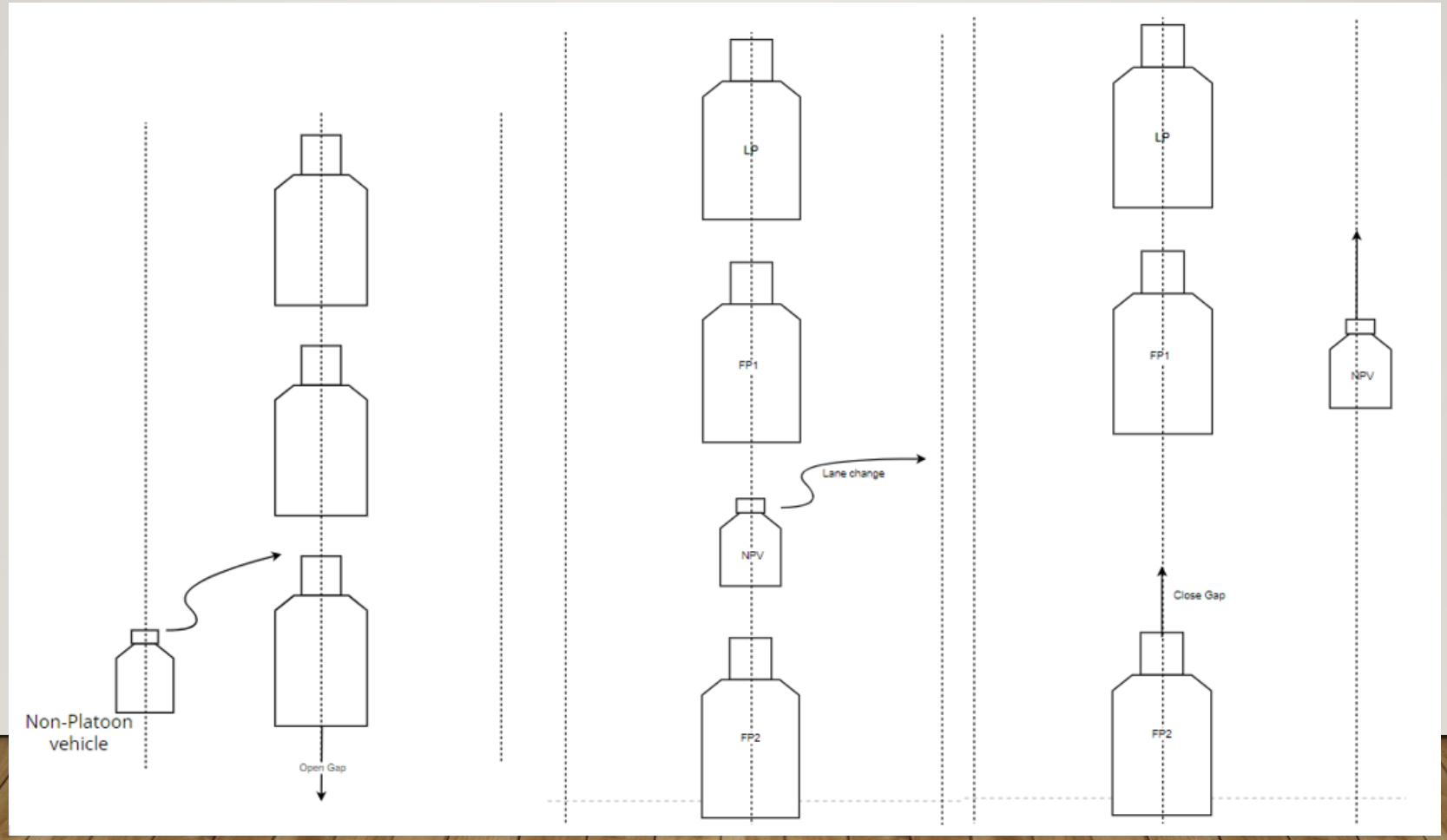


STATE MACHINE

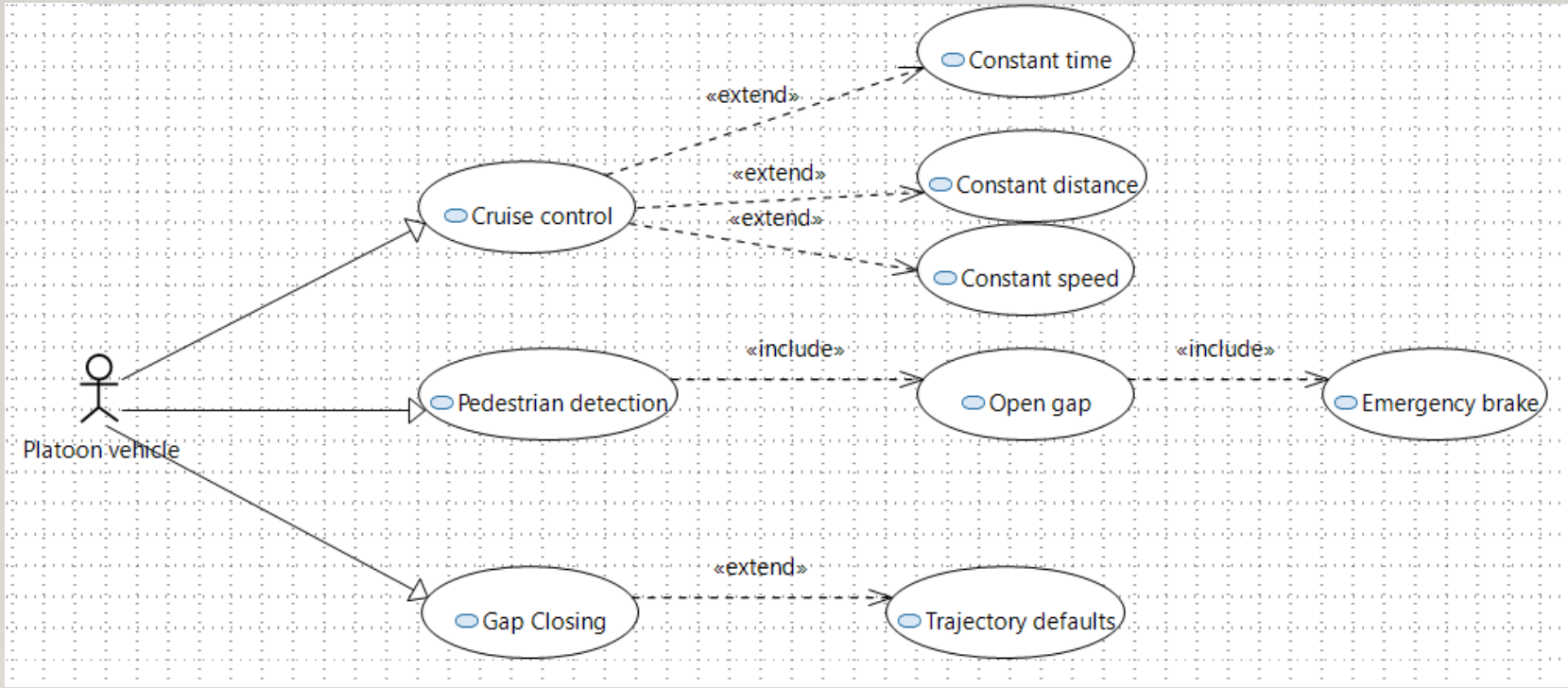


GAP ADAPTATION TECHNIQUES

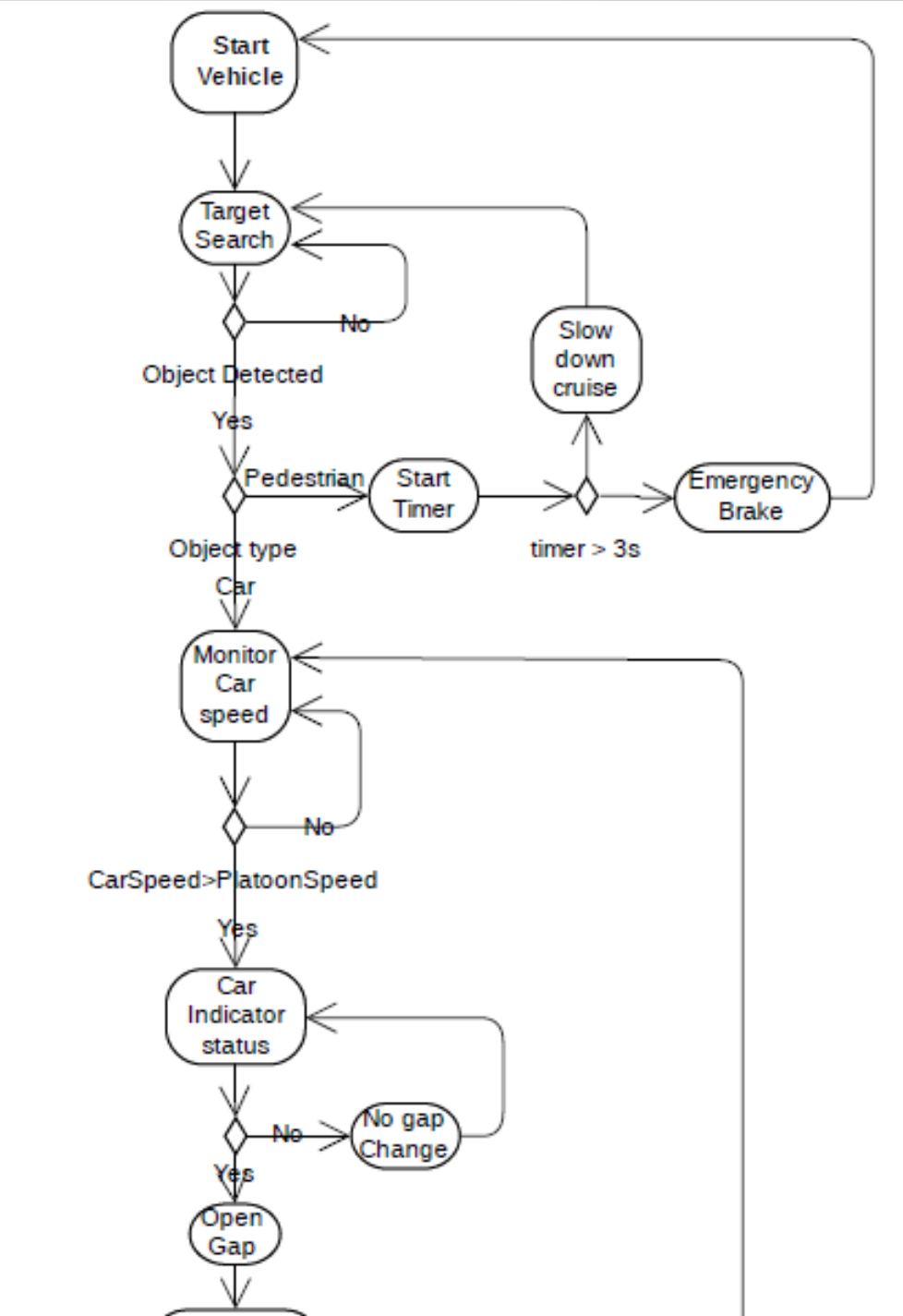
USE CASE OVERVIEW



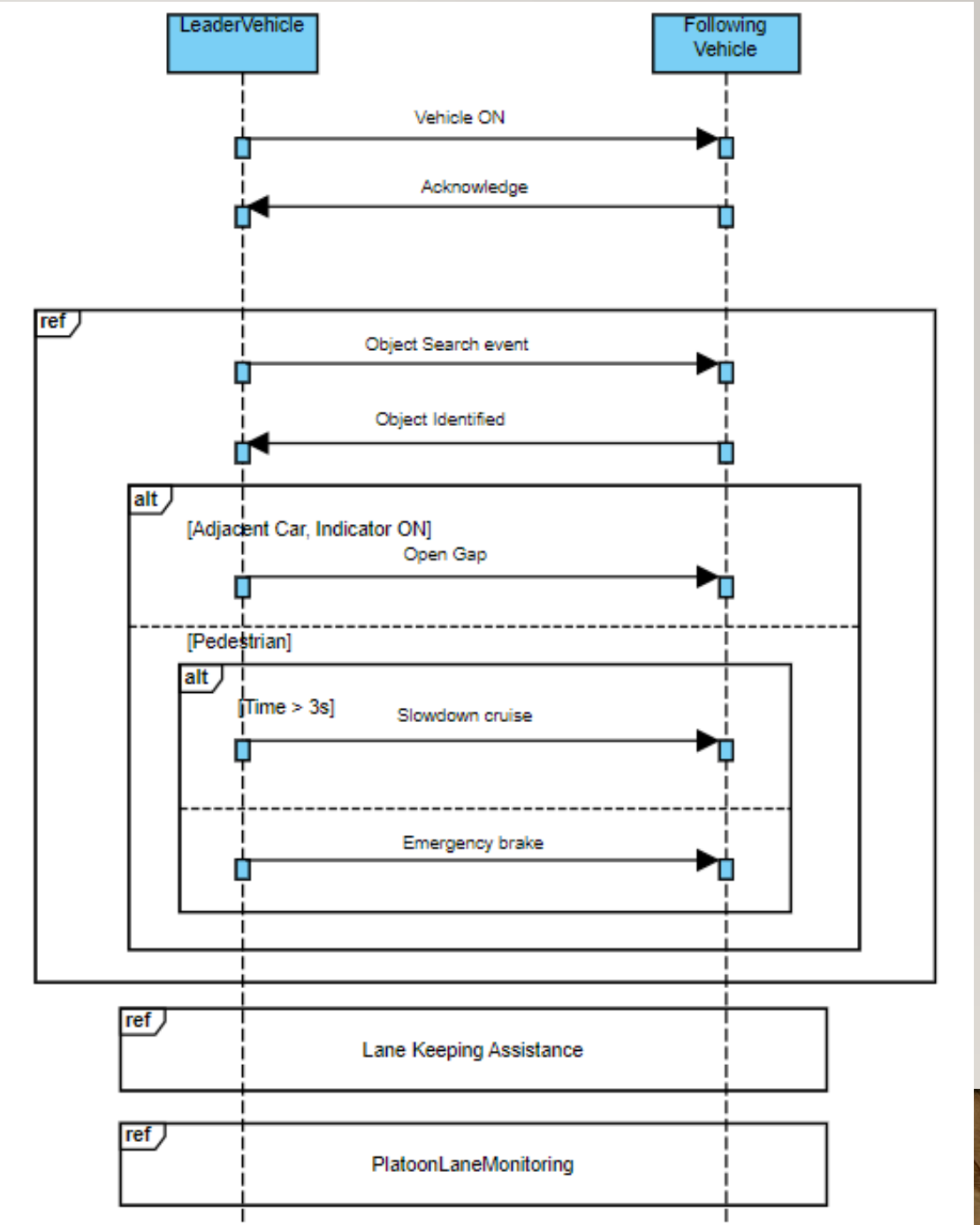
USE CASE DIAGRAM



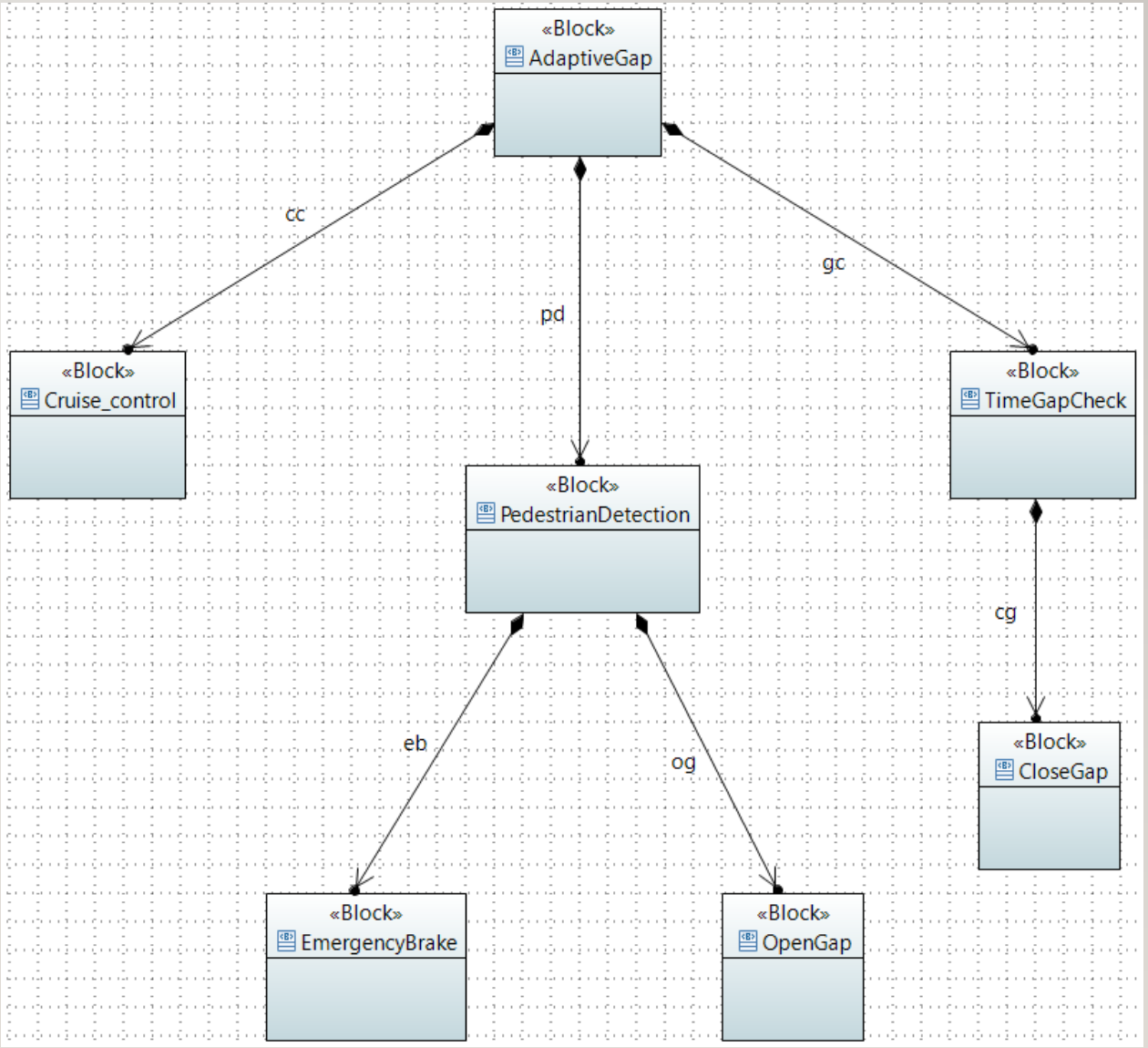
ACTIVITY DIAGRAM



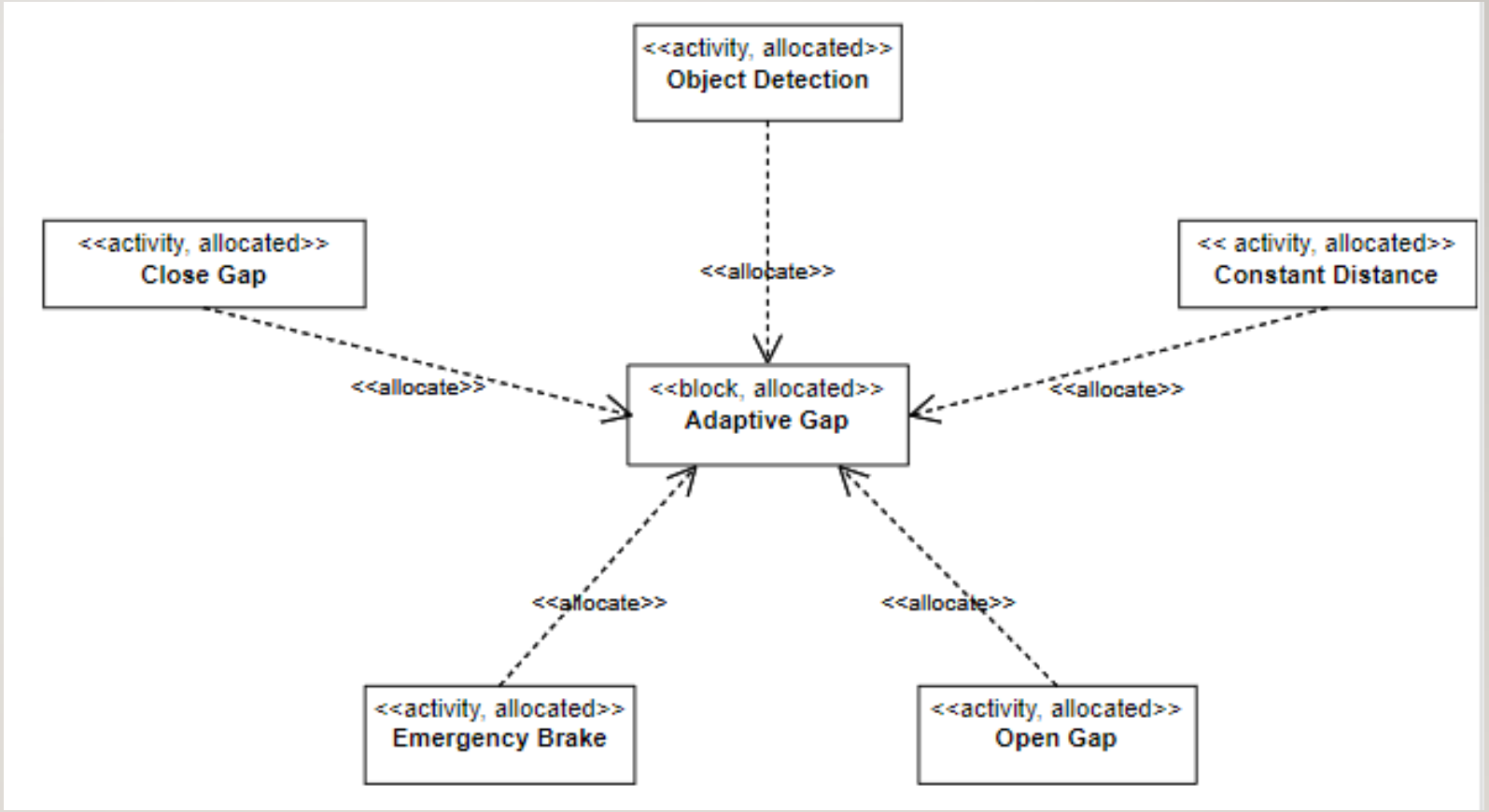
SEQUENCE DIAGRAM



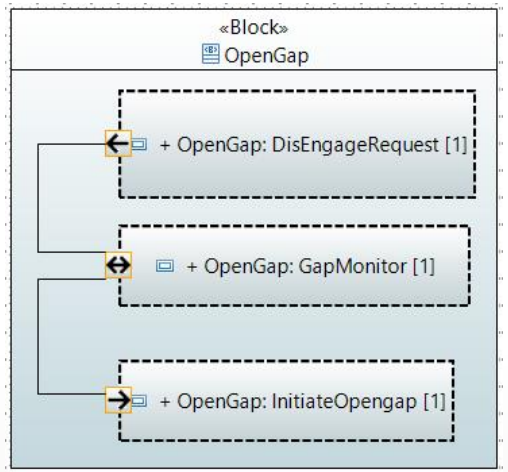
BLOCK DIAGRAM

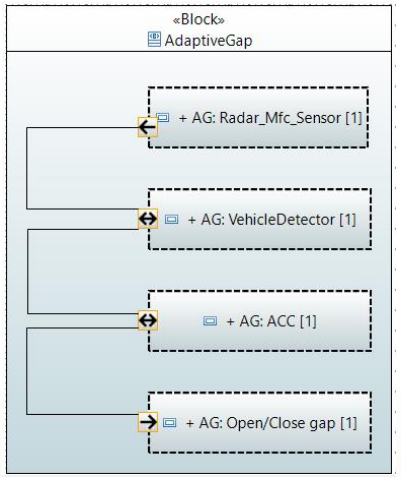


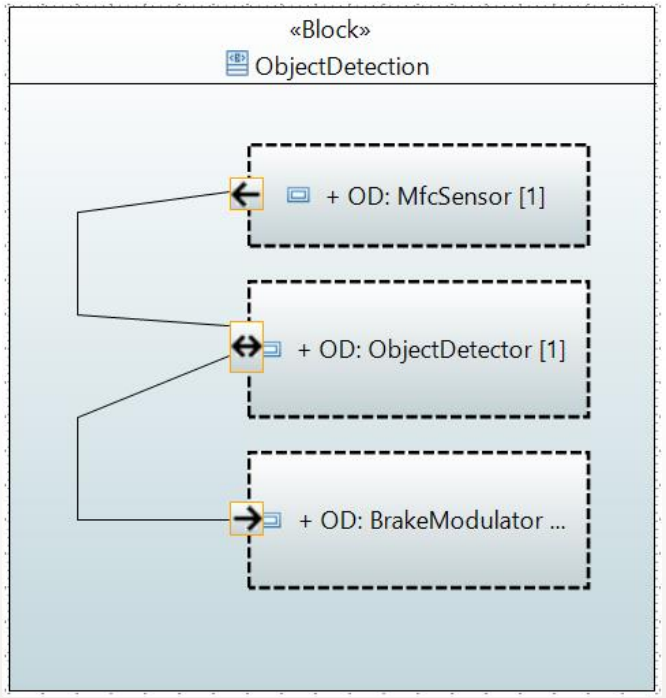
ALLOCATION DIAGRAM

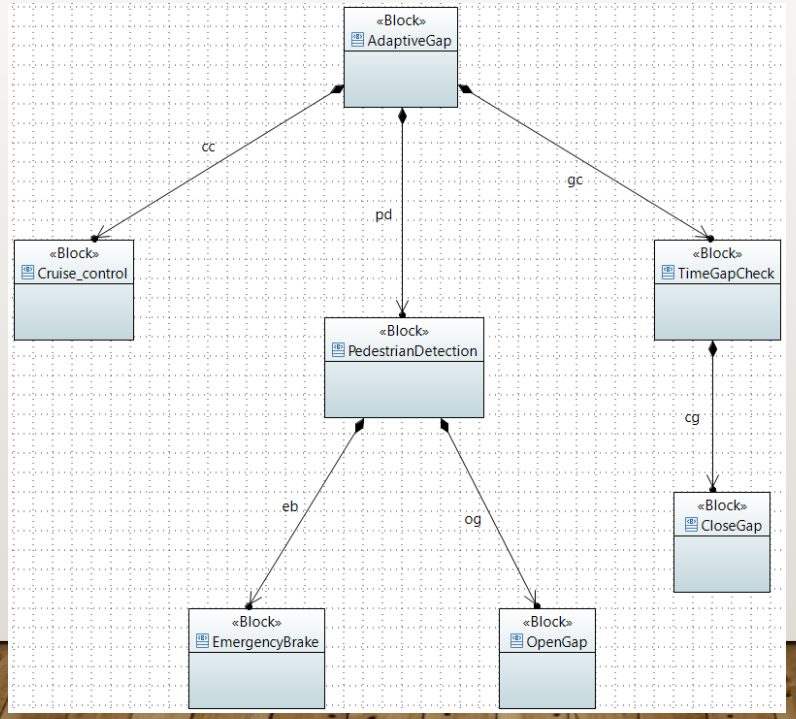


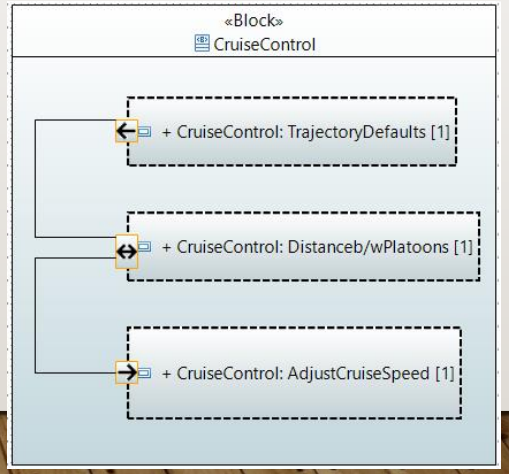
INTERNAL BLOCK DIAGRAM



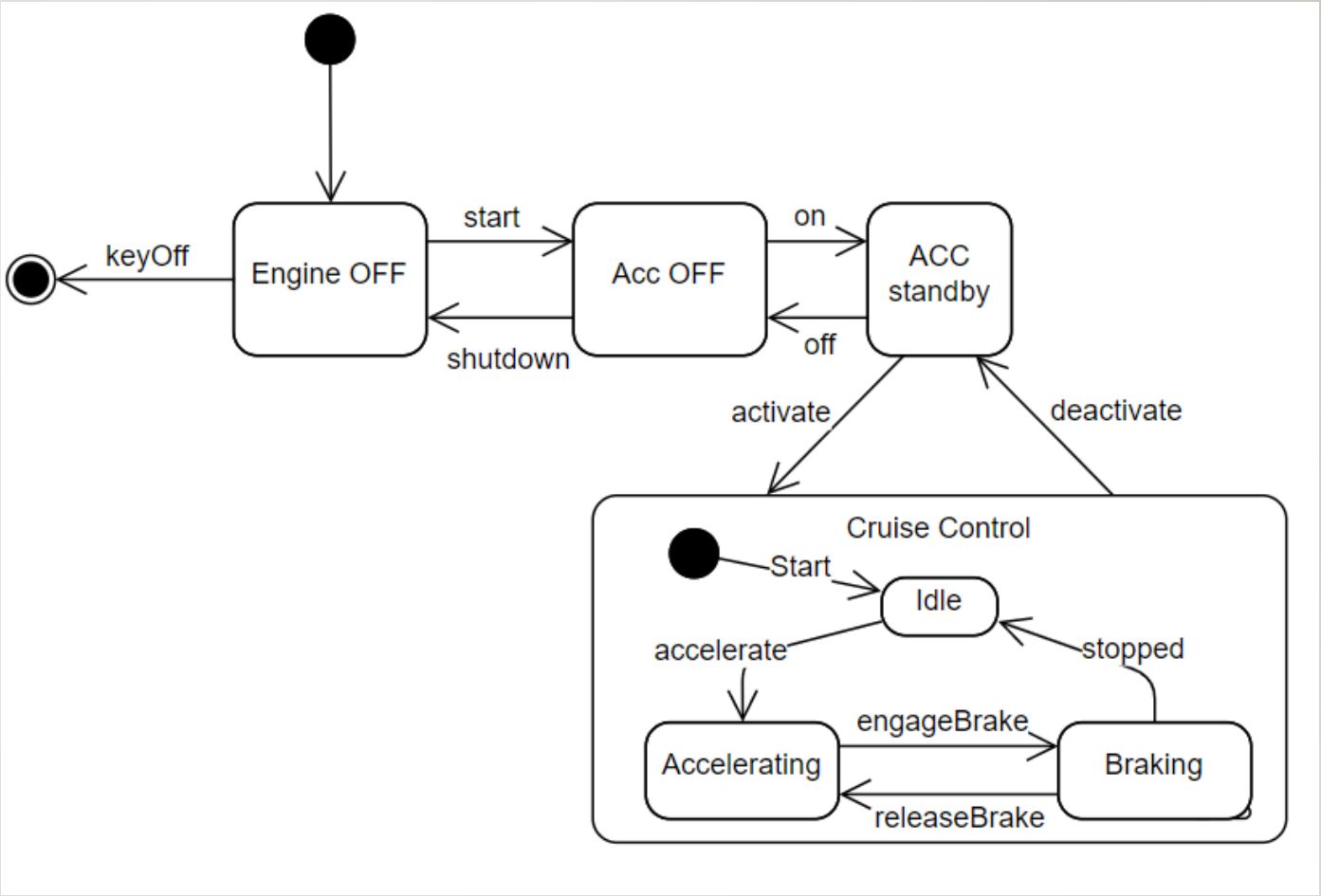






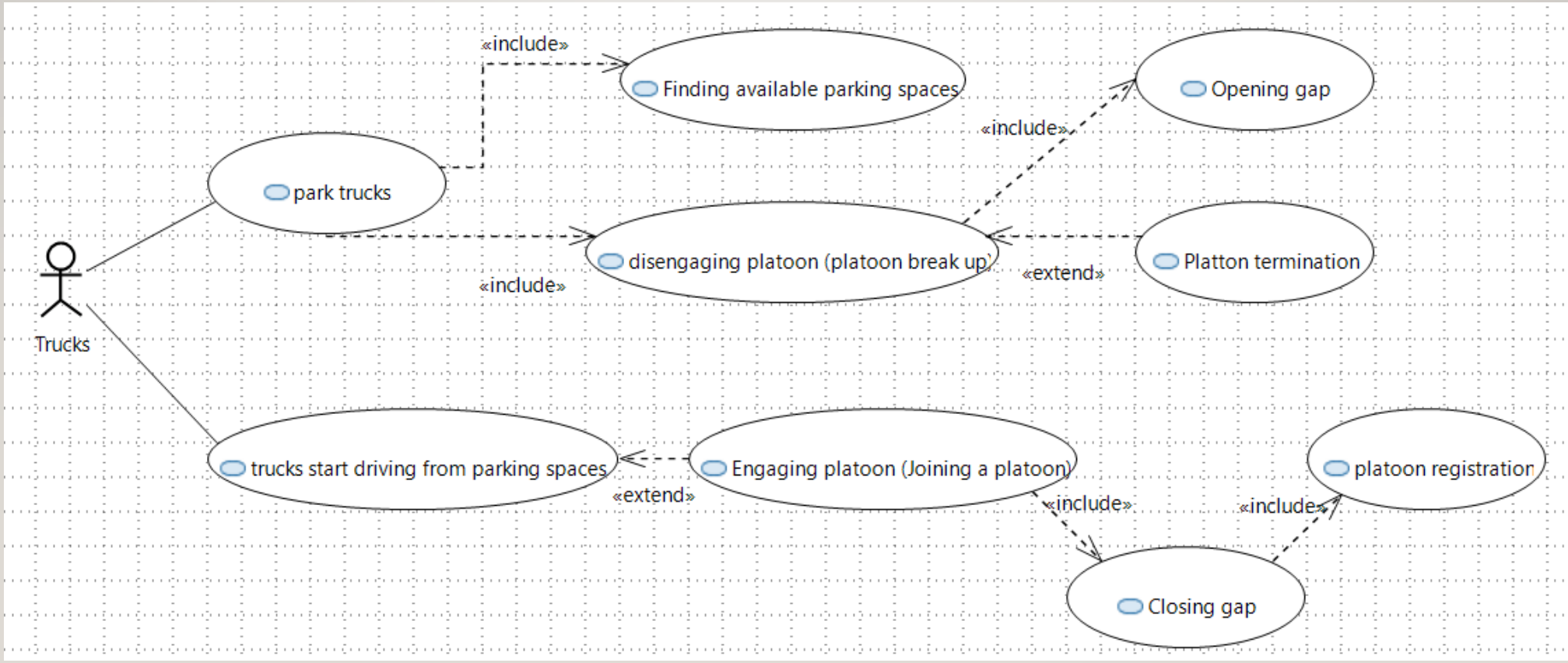


STATE-MACHINE DIAGRAM

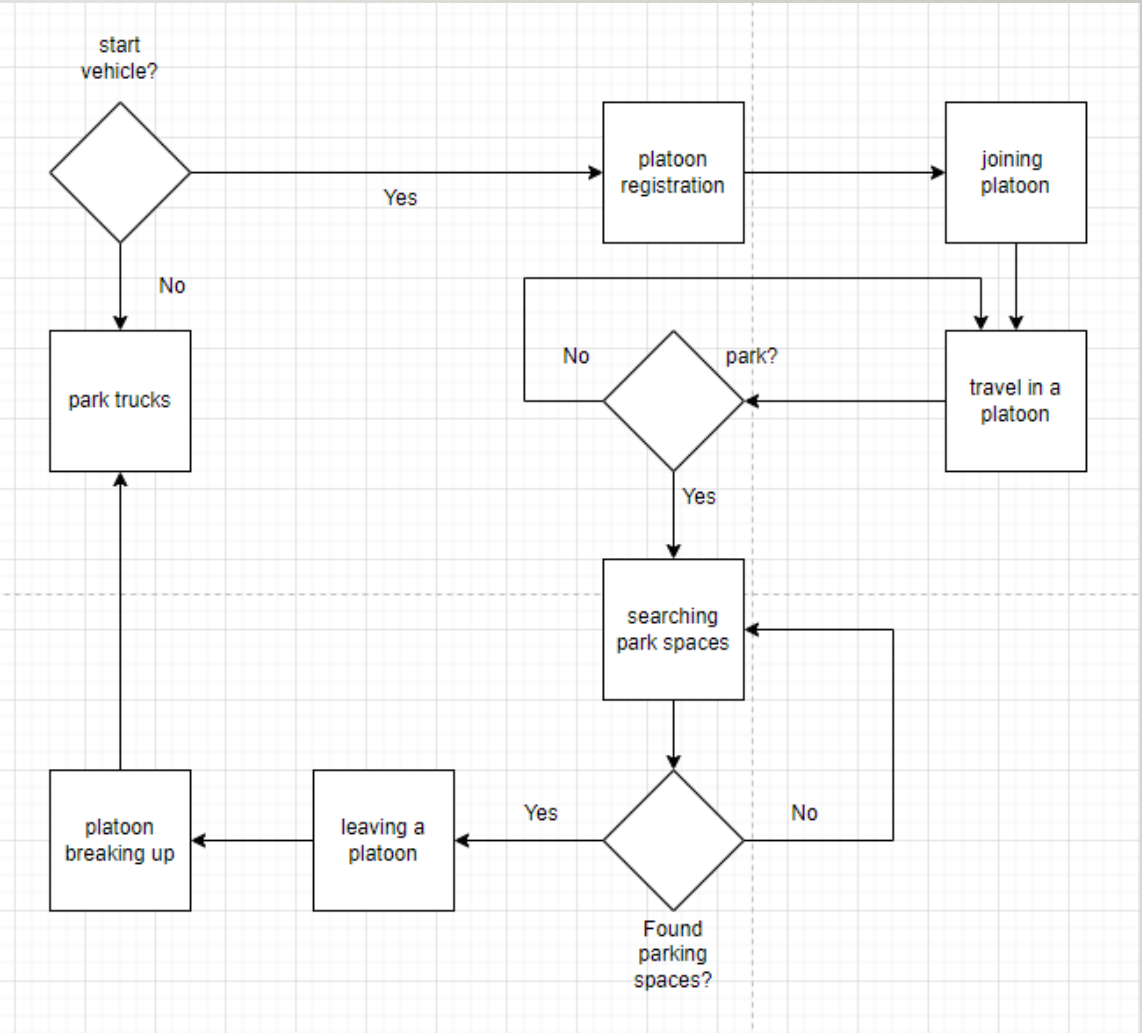


PARKING SYSTEM

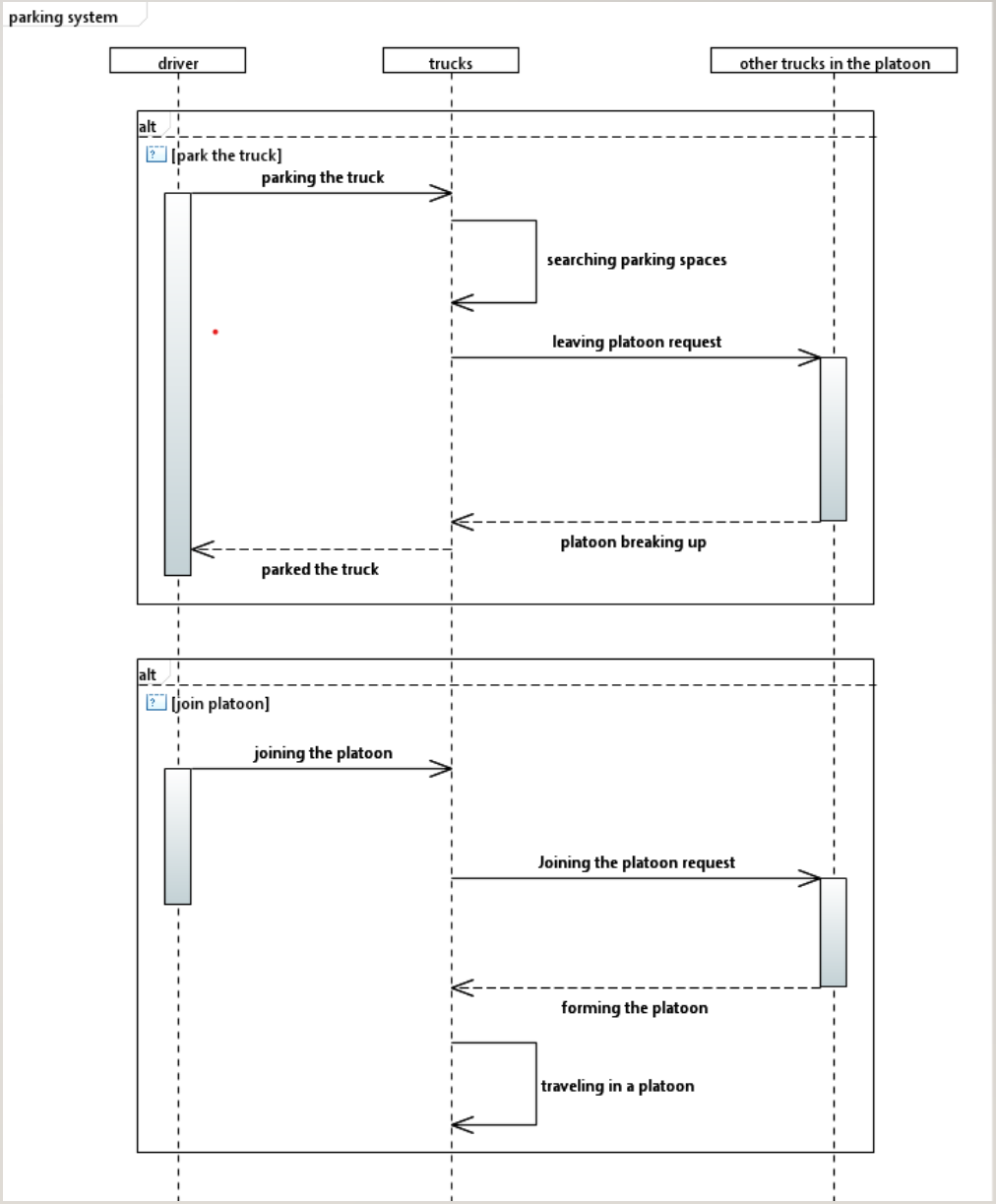
USE CASES


«include»
Finding available parking spaces
Opening gap
«include»
park trucks
disengaging platoon (platoon break up)
Platton termination
«include»
«extend»
Trucks
trucks start driving from parking spaces
Engaging platoon (Joining a platoon)
platoon registration
«extend»
«include»
«include»
Closing gap

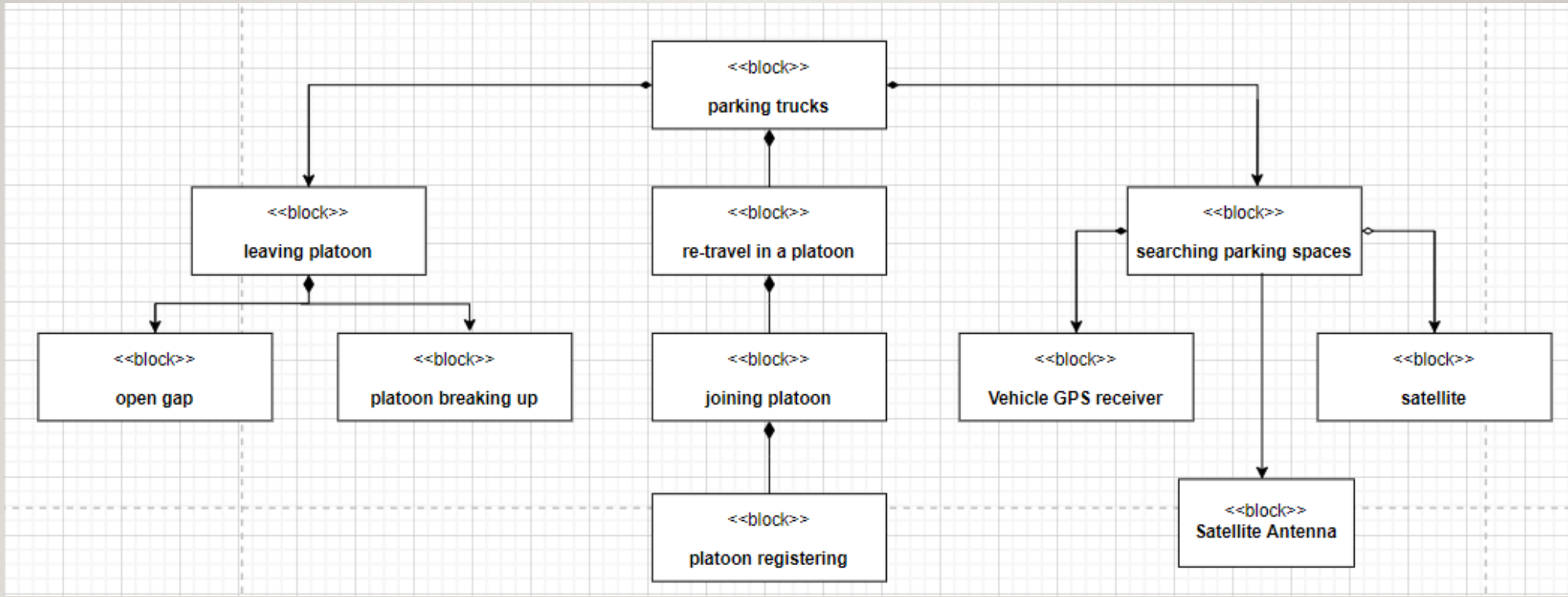
ACTIVITY DIAGRAM



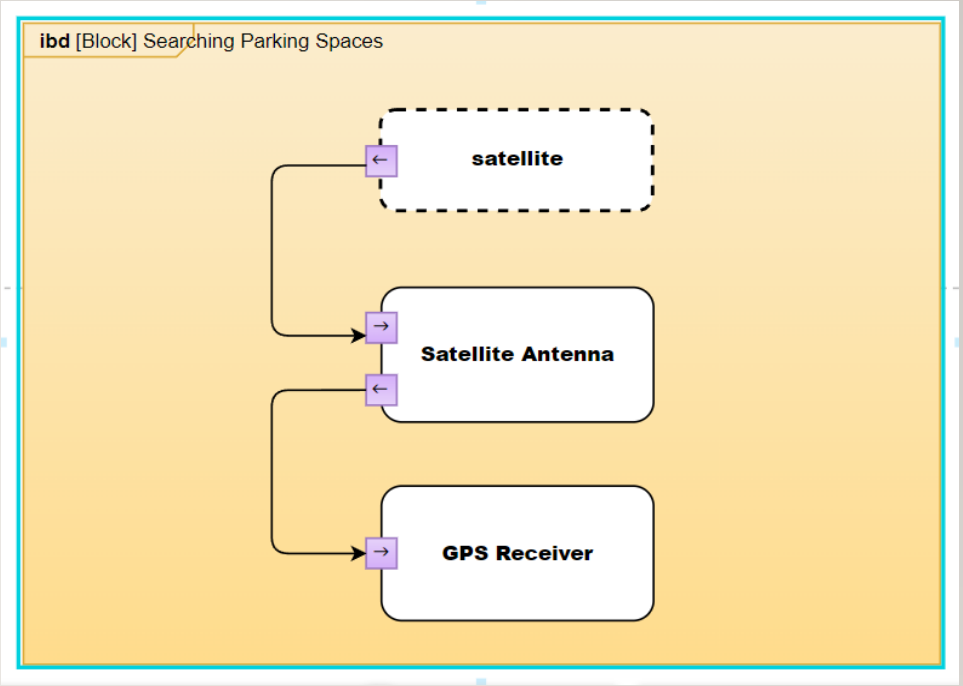
SEQUENCE DIAGRAM

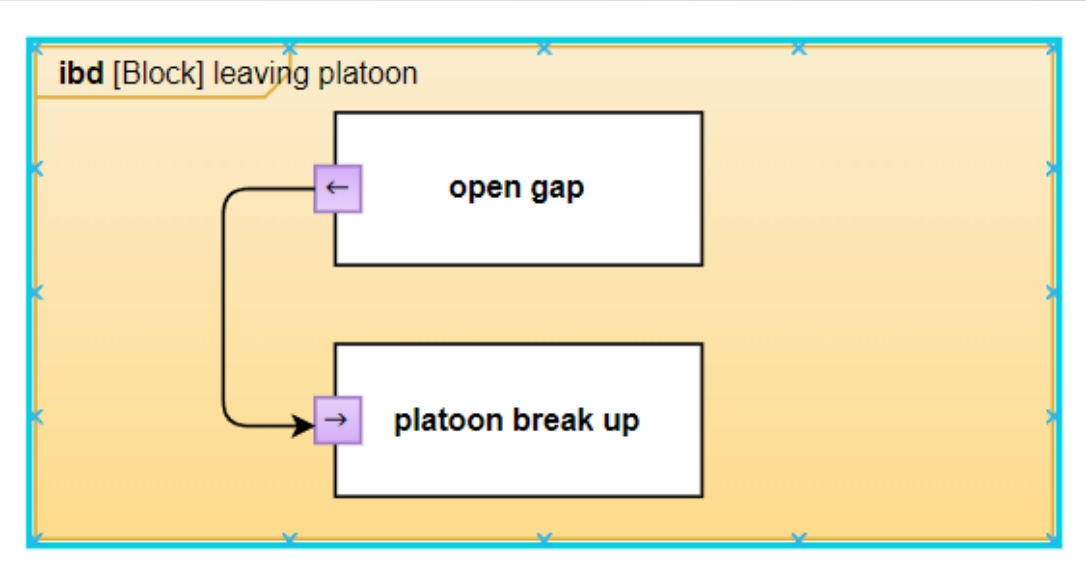


BLOCK DIAGRAM

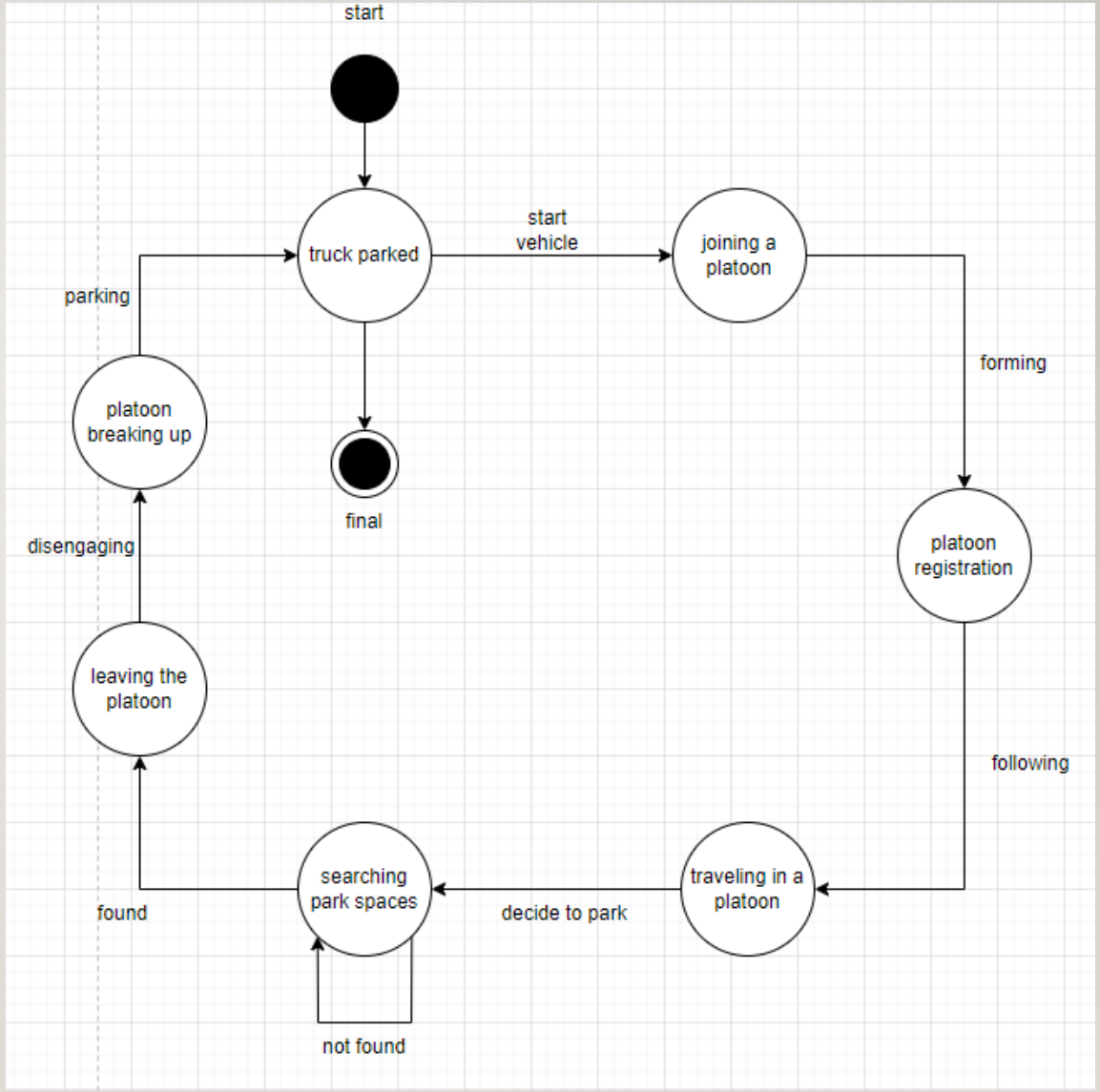


BLOCK DIAGRAM WITH INTERNAL BLOCK DIAGRAMS

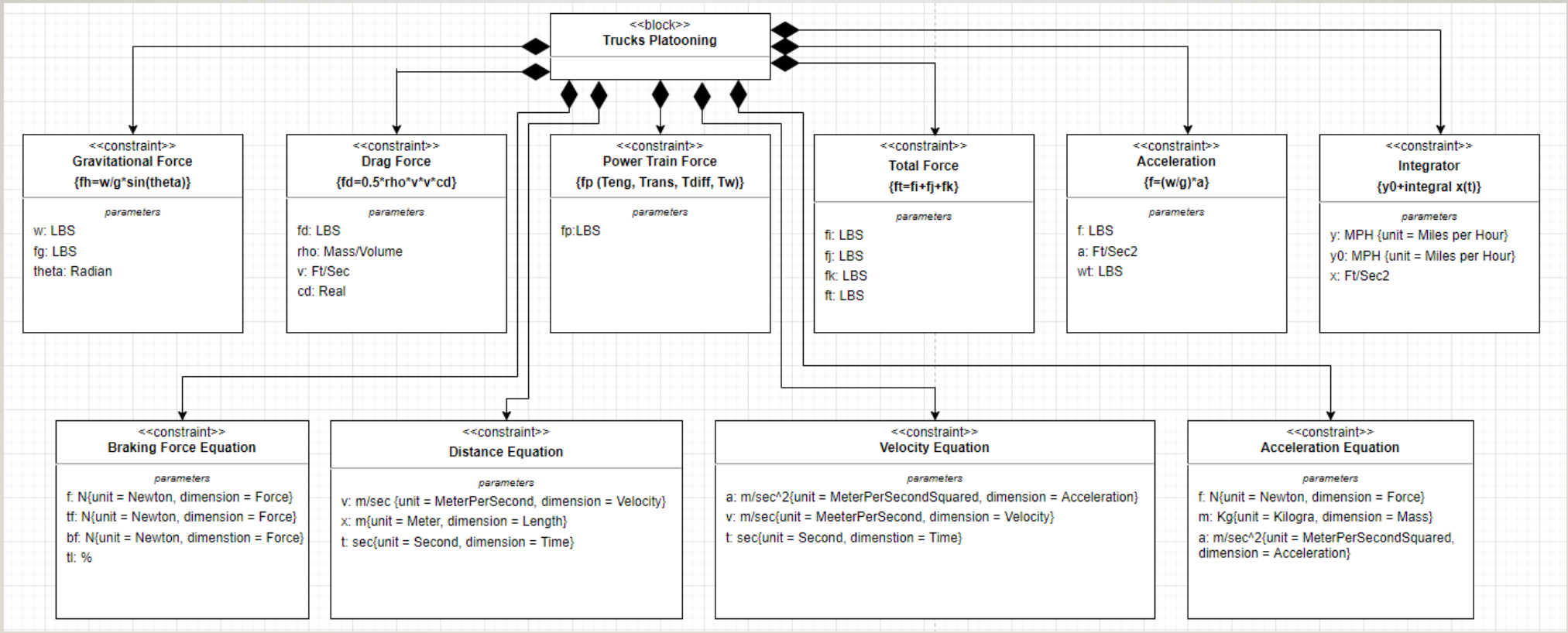




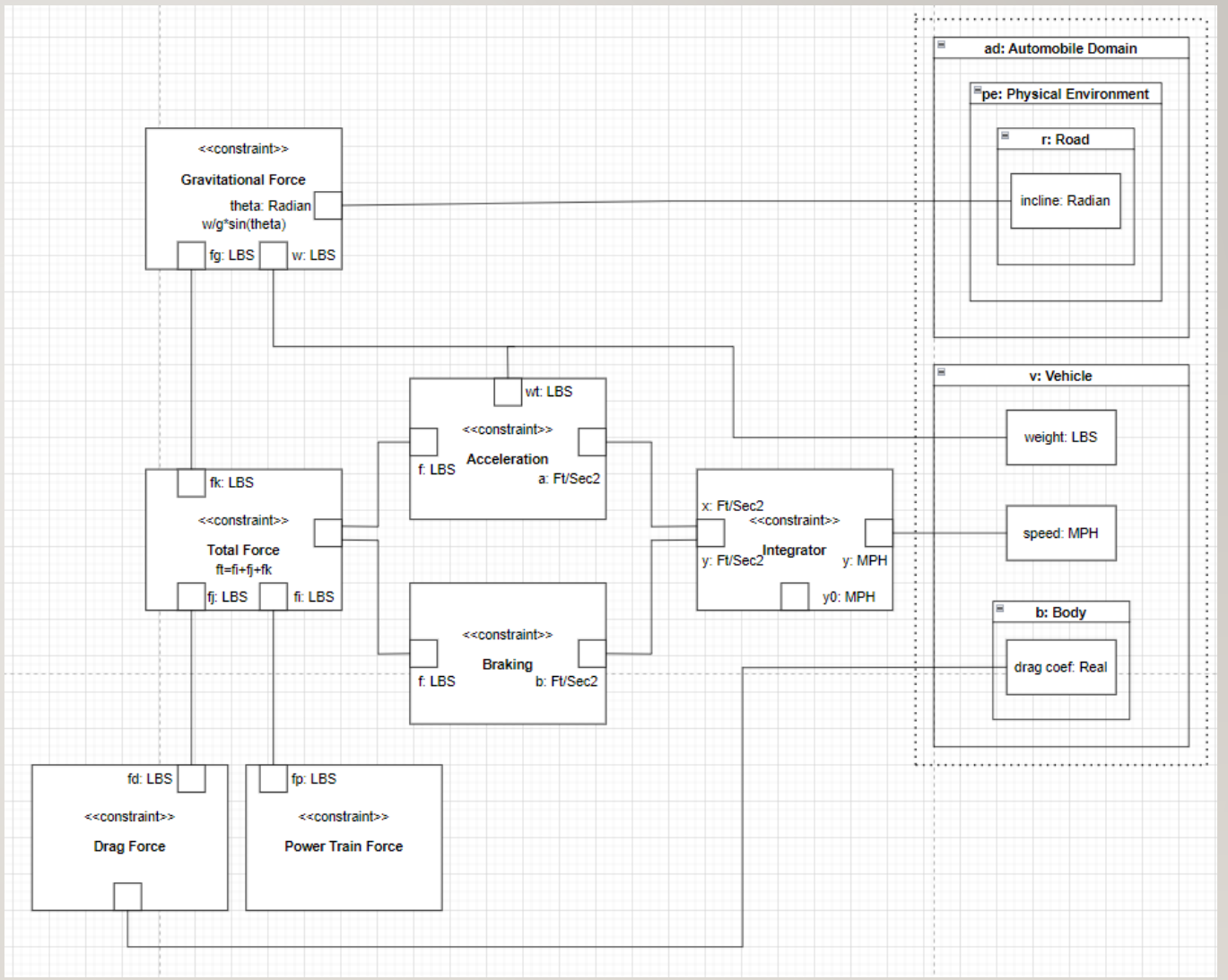
STATE MACHINE



PARAMETRIC CONSTRAINT DIAGRAMS



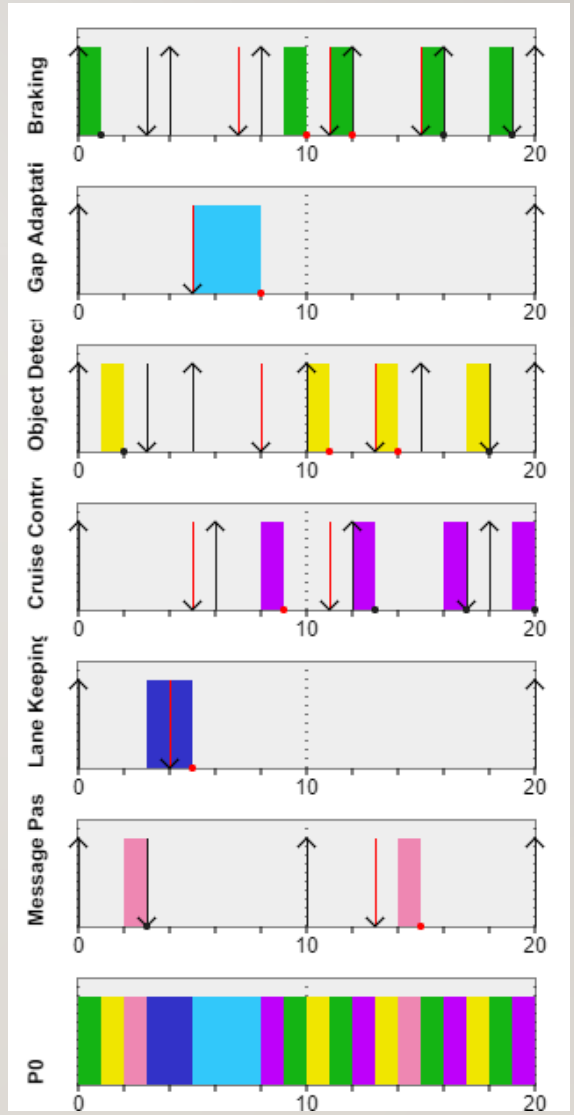
PARAMETRIC CONSTRAINT DIAGRAMS



SCHEDULING

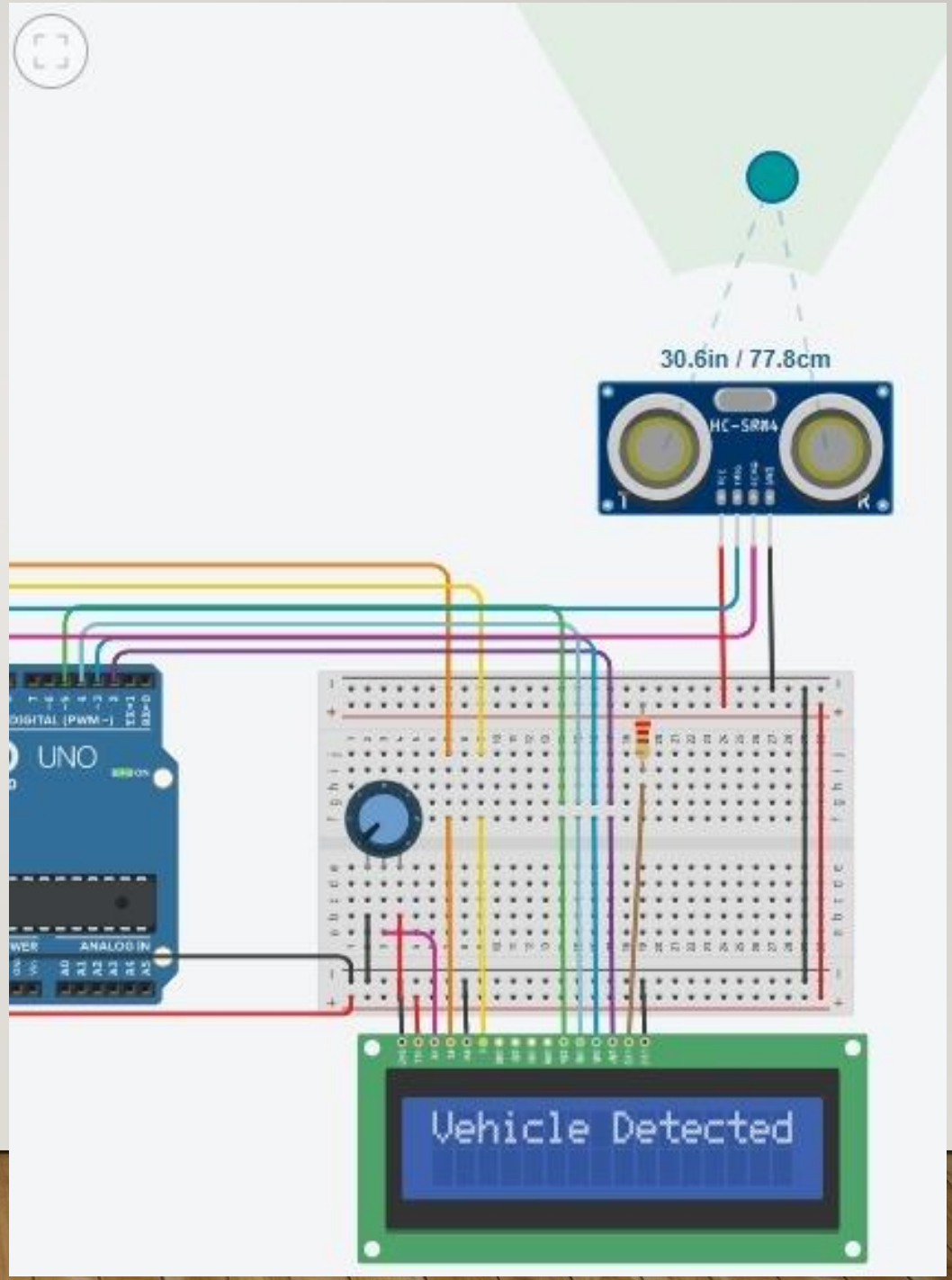
- Earliest deadline first principle is used
- Periodic and static real time scenario is considered

Task	WCET (Ci)	Deadline (Di)	Period (Ti)	Ci/Ti	Response Time
Braking	1	3	4	0.25	3.6
Gap Adaptation	3	5	20	0.15	8
Object Detection	1	3	5	0.2	3.75
Cruise Control	1	5	6	0.16	5.75
Lane Keeping	2	4	20	0.1	5
Message Passing	1	3	10	0.1	4



HARDWARE IMPLEMENTATION

CIRCUIT SIMULATION

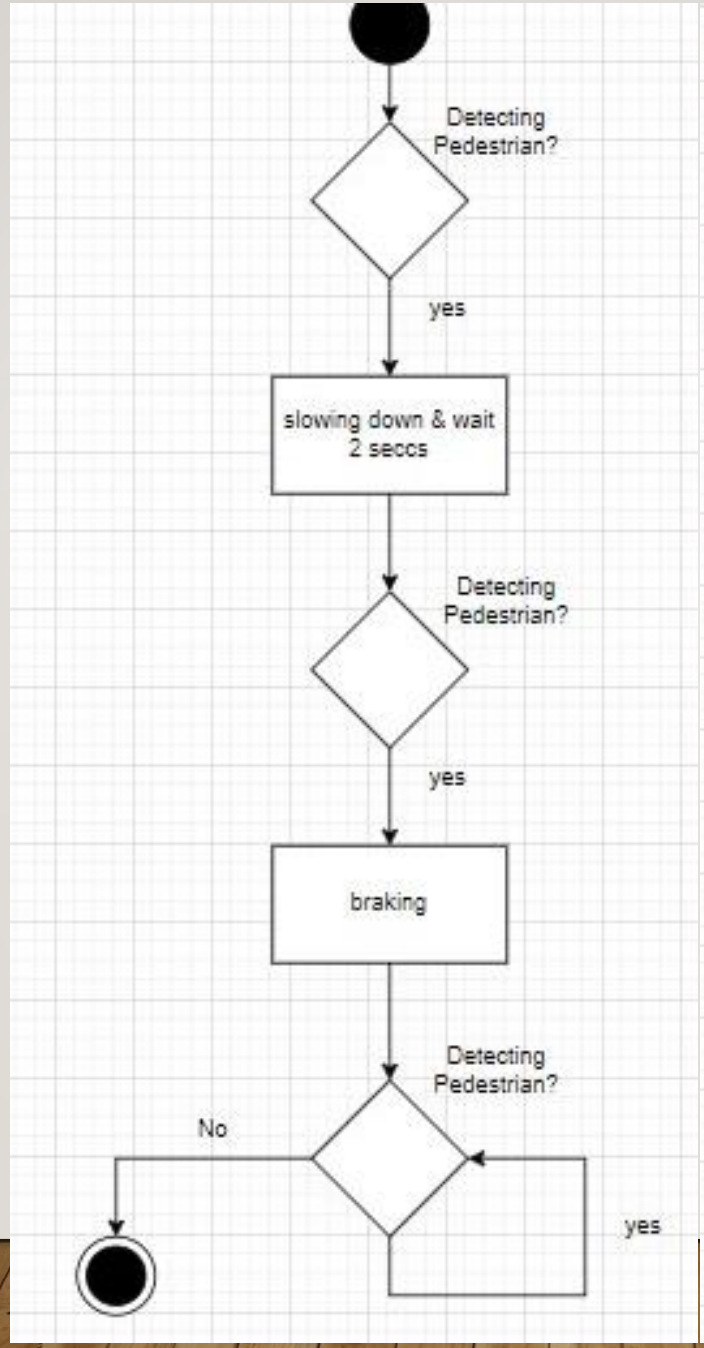


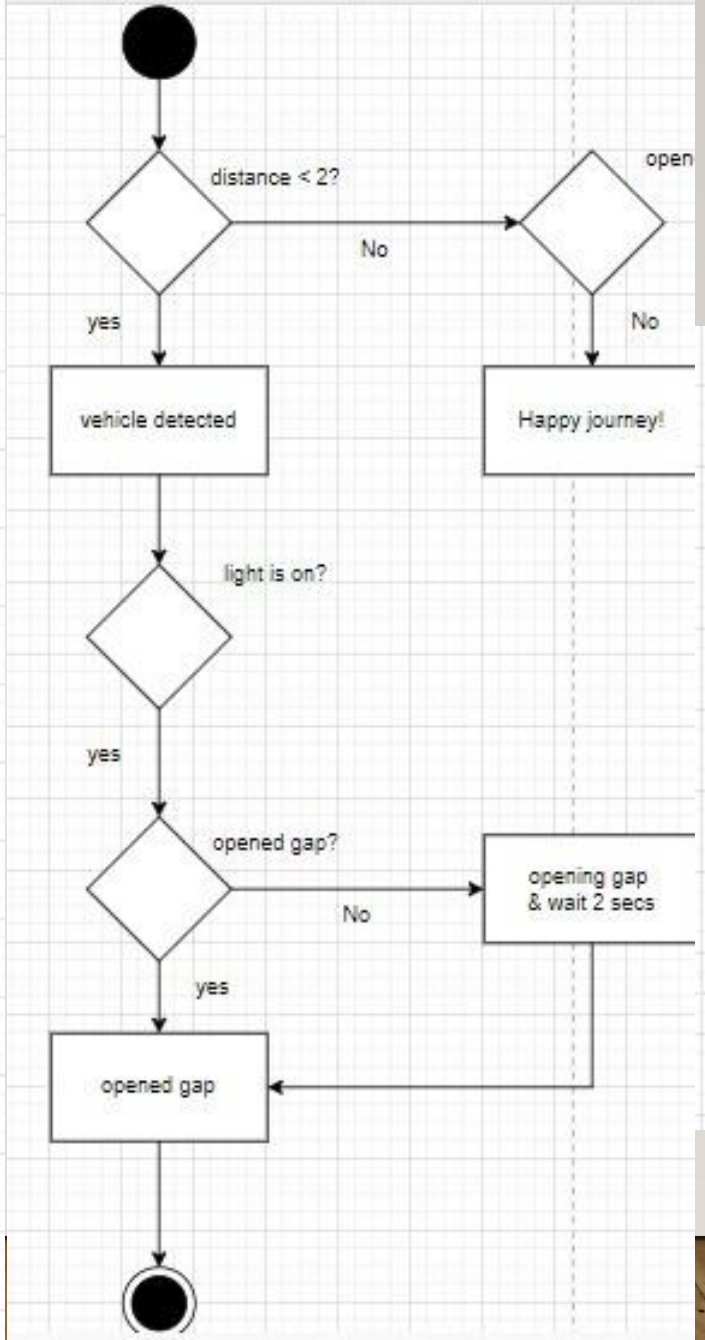
CODE FOR HARDWARE IMPLEMENTATION

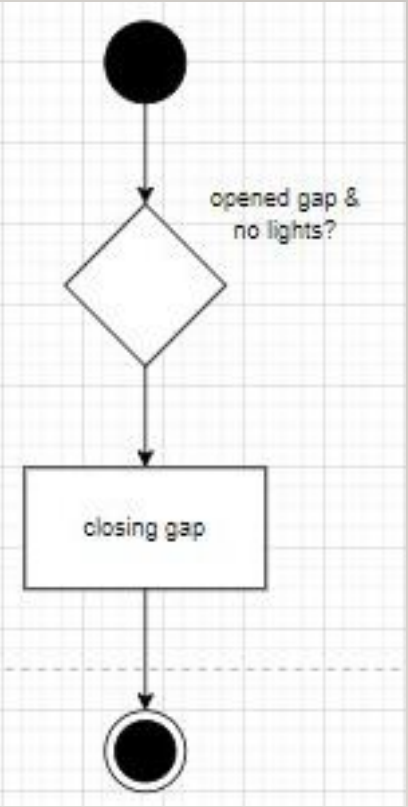
```
51 if(pedestrian){   // DETECTS PEDESTRIAN?
52   lcd.clear();
53   lcd.print("Slowing Down...");
54   delay(2000);                  // WAITS 2 SECS
55   if (Serial.available() > 0) {       // Character available?
56     if(Serial.read()=='3'){           // Read character
57       pedestrian = false;
58     }
59   }
60   if(pedestrian){ // STILL THERE?
61       lcd.clear();
62       lcd.print("Braking...");
63     while (pedestrian){
64       if (Serial.available() > 0) {       // Character available?
65         if(Serial.read()=='3'){           // Read character
66           pedestrian = false;
67         }
68       }
69     }
70   }
71 }
```

```
73 if(distance < 2){// DETECTS CAR WITH LIGHTS?
74   lcd.clear();
75   lcd.print("Vehicle Detected");
76   if(ligths){// DETECTS CAR WITH LIGHTS?
77     lcd.clear();
78     if(!opened){      // No opened Gap
79       lcd.print("Opening Gap...");
80       opened = true;
81       delay(2000);                 // WAITS 2 SECS
82     }
83     lcd.clear();
84     lcd.print("Opened Gap");
85   }
86 }
87 else if(!opened){     // No Gap
88   lcd.clear();
89   lcd.print("Happy Journey!");
90 }
91
92 if(opened && !ligths){ //Have open gap and No ligths
93     lcd.clear();
94     lcd.print("Closing Gap...");
95     delay(2000);
96     opened = false;
97 }
98
```

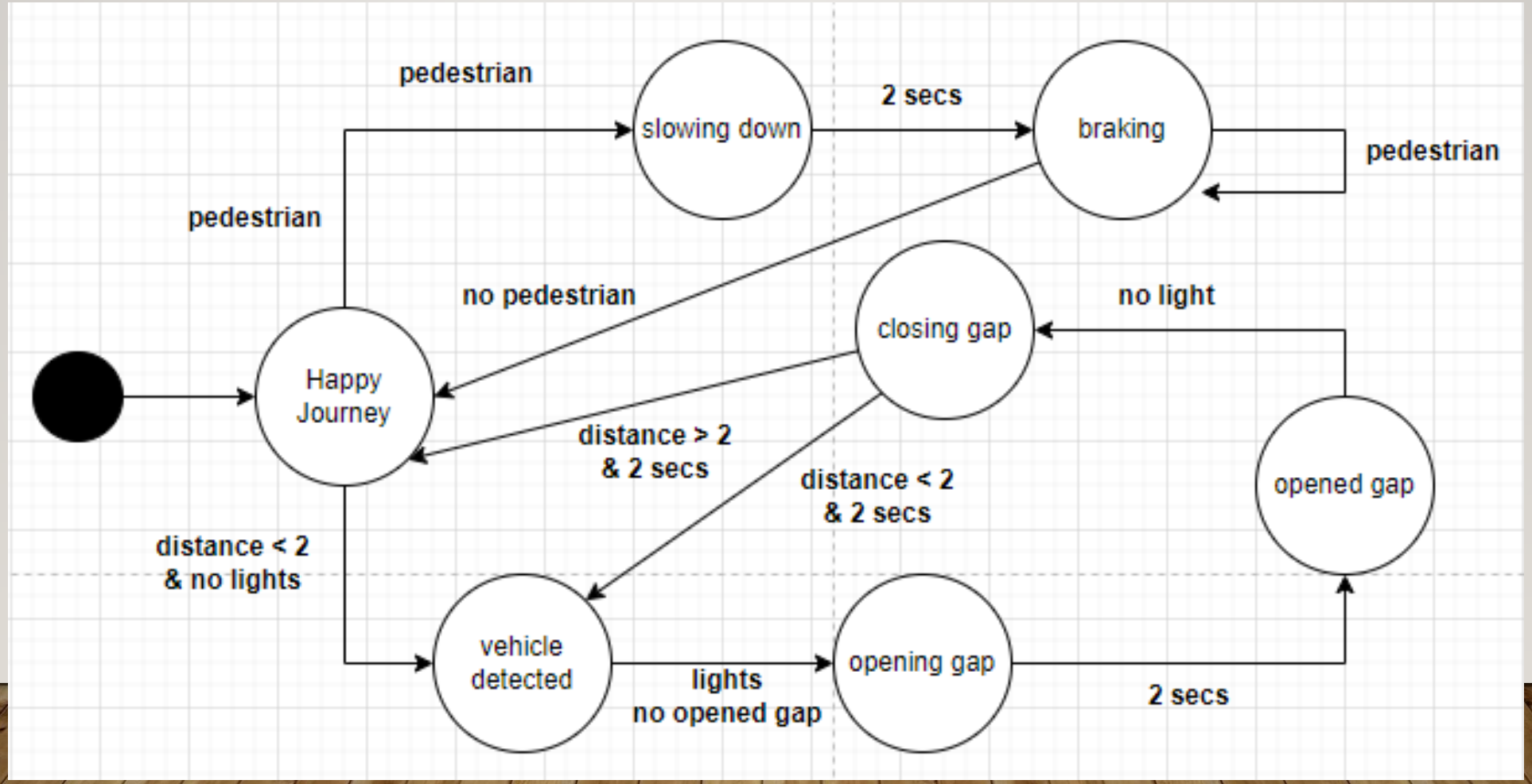
ACTIVITY DIAGRAMS







MESSAGES STATE MACHINE



JUNIT TESTING

PEDESTRIAN DETECTION

```
package TruckP1;

public class UnitTesting {
        private String result;
        public String pedestrian(boolean check1, boolean check2) {

                if(check1){                                          // DETECTS PEDESTRIAN?
                        result = "Slowing Down...";
                        if(check2){                                  // STILL THERE?
                        result = "Braking...";
                    }
                }
                else {
                        result = "Happy Journey!";
                }
                return result;
        }
```

VEHICLE AND LIGHTS DETECTION

```
public String vehicle(double distance, boolean lights, boolean opened) {

        if(distance < 2){                                   // VEHICLE DETECTED?
                result = "Vehicle Detected";
            if(lights){                                     // VEHICLE WITH LIGHTS DETECTED?
                if(!opened){                                // N0 OPENED GAP?
                  result = "Opening Gap...";
                  opened = true;
                }
                else{                                       // OPENED GAP
                        result = "Opened Gap";
                }
            }
        }
        else if(!opened){                                   // NO VEHICLE DETECTED AND NO OPENED GAP
                result = "Happy Journey!";
        }

        return result;

}
```

GAP DETECTION

```
public String closeGap(boolean lights, boolean opened) {
        if(!lights && opened){                          // NO LIGTHS DETECTED AND OPENED GAP
              result = "Closing Gap...";
        }else {
                result = "";
        }

        return result;

}
```

PEDESTRIAN DETECTION TESTS

```java
package TruckPl;

import static org.junit.Assert.*;

import org.junit.Test;

public class PedestrianTest {

	UnitTesting obj = new UnitTesting();
	String result;

	// Pedestrian detected once
	@Test
	public void SlowingDown() {
		result = obj.pedestrian(true, false);
		assertEquals("Slowing Down...", result);
	}
	@Test
	public void SlowingDown_Defect() {
		result = obj.pedestrian(false, true);
		assertNotEquals("Slowing Down...", result);
	}
```

```java
// Pedestrian detected twice
@Test
public void Braking() {
	result = obj.pedestrian(true, true);
	assertEquals("Braking...", result);
}
@Test
public void Braking_Defect() {
	result = obj.pedestrian(false, false);
	assertNotEquals("Braking...", result);
}

// Pedestrian never detected
@Test
public void HappyJourney() {
	result = obj.pedestrian(false, false);
	assertEquals("Happy Journey!", result);
}
@Test
public void HappyJourney_Defect() {
	result = obj.pedestrian(true, true);
	assertNotEquals("Happy Journey!", result);
}
```

VEHICLE AND LIGHTS DETECTION TEST

```java
package TruckPl;

import static org.junit.Assert.*;

import org.junit.Test;

public class VehicleTest {

	UnitTesting obj = new UnitTesting();
	String result;

	// Vehicle detected without lights before opening a gap
	@Test
	public void VehicleDetected() {
		result = obj.vehicle(1.5, false, false);
		assertEquals("Vehicle Detected", result);
	}
	@Test
	public void VehicleDetected_Defect() {
		result = obj.vehicle(1.5, true, true);
		assertNotEquals("Vehicle Detected", result);
	}
```

```java
// Vehicle detected with lights before opening a gap
@Test
public void OpeningGap() {
	result = obj.vehicle(1.5, true, false);
	assertEquals("Opening Gap...", result);
}
@Test
public void OpeningGap_Defect() {
	result = obj.vehicle(1.5, false, true);
	assertNotEquals("Opening Gap...", result);
}

// Vehicle detected with lights after opening a gap
@Test
public void OpenedGap() {
	result = obj.vehicle(1.5, true, true);
	assertEquals("Opened Gap", result);
}
@Test
public void OpenedGap_Defect() {
	result = obj.vehicle(1.5, false, false);
	assertNotEquals("Opened Gap", result);
}
```

```java
// No Vehicle detected and no gap
@Test
public void HappyJourney() {
	result = obj.vehicle(2.5, false, false);
	assertEquals("Happy Journey!", result);
}
@Test
public void HappyJourney_Defect() {
	result = obj.vehicle(2.5, true, true);
	assertNotEquals("Happy Journey!", result);
}
```

GAP DETECTION TEST

```java
package TruckPl;

import static org.junit.Assert.*;

import org.junit.Test;

public class GapTest {

	UnitTesting obj = new UnitTesting();
	String result;

	@Test
	// No lights detected after opening a gap
	public void ClosingGap() {
		result = obj.closeGap(false, true);
		assertEquals("Closing Gap...", result);
	}

	@Test
	public void ClosingGap_Defect() {
		result = obj.closeGap(true, true);
		assertNotEquals("Closing Gap...", result);
	}
}
```

TESTING RESULTS