Embedded Systems Engineering

(Prof. Dr. Stefan Henkler)

Truck Platooning

Abhinandan Dinakar
Embedded System Engineering
Fachhochschule Dortmund
Dortmund, Germany

Nikhil Ganapathy Manjapura
Embedded System Engineering
Fachhochschule Dortmund
Dortmund, Germany

Shreya Manasali
Embedded System Engineering
Fachhochschule Dortmund
Dortmund, Germany

Abstract - Truck Platooning system is automation and cooperation system that leads a way for automated vehicle implementation. In this study we have considered platooning of lead truck with driver followed by other automated trucks. Addition to that we have considered many of the challenges regarding platooning system and possible effects and potential dangers in fully automated platooning system. We have discussed on what are the main modules that need to be considered for platooning implementation and the possible outcomes in case of failures. We conclude that each part of the system should be responsive to the stimuli of relevant reasons available..

I. INTRODUCTION

The vehicle dynamics can be separated into model and platoon types. The model types can be single integrator (Lin et al., 2012), second-order (Peters et al., 2014), third-order (Zheng et al., 2016b), SISO (Herman et al., 2015) and nonlinear (Zheng et al.,2017a) models. Moreover, platoons can be classified on other two terminologies, namely homogenous and heterogenous. Homogenous refers to platoons in which vehicles have same dynamical capabilities. In contrast, Heterogenous refers to platoons with vehicles that does not share identical dynamic capabilities. Here we concentrate on longitudinal models where lead truck is followed by follower trucks in single line manner.

II. MODEL SECTION

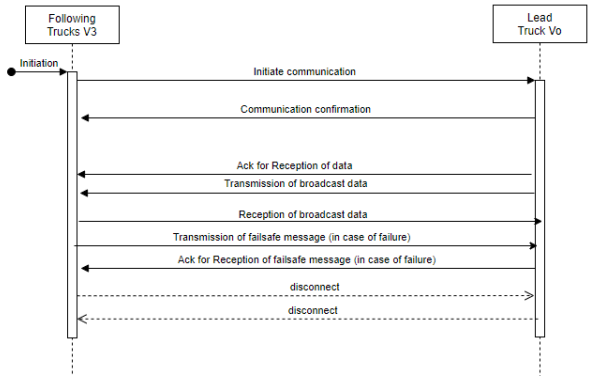


Figure 1: Sequence Diagram

Figure 1 image depicts the communication sequence between lead and follower trucks from initiation to data packets exchange between lead and follower trucks. Here we have used TCP/IP as communication protocol between trucks. Figure 2 and Figure 3 reprents the block diagrams of lead and follower truck platooning system modules that are taken into consideration in this study.

Figure 5 represents internal block diagram of Braking system with sensors, traction detector and brake modulator as internal modules. Figure 6 is internal block diagram of Communication system in which sensor and actuator data are communicated from one truck to another. Data is encrypted and decrypted at transmitter and receiver respectively.

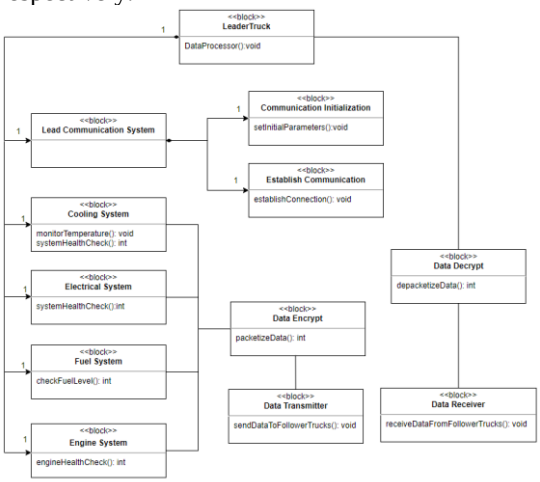


Figure 2: Block Diagram - Lead Truck

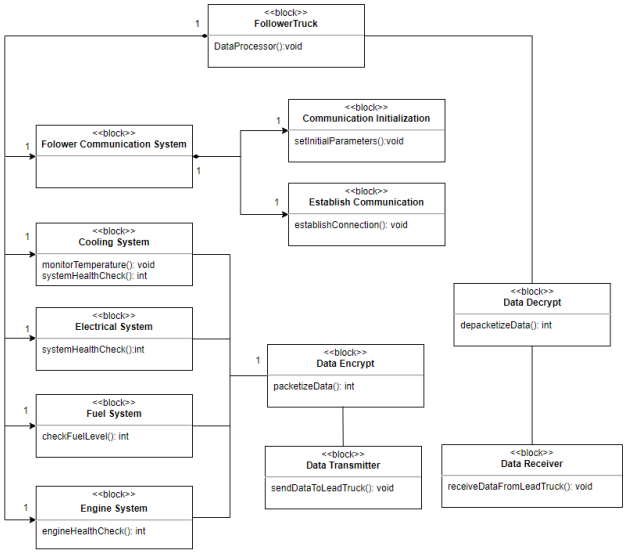


Figure 3: Block Diagram - Follower Truck

Figure 7 is internal block diagram of Cooling system with sensors and actuators. If there is variation in temperature beyond threshold, actuators like fan and coolant pumps shall be triggered and the same shall be indicated in the dashboard. Figure 8 represents the fule system, in which fule level is checked in all truck and the same shall be indicated to the driver in the lead truck.

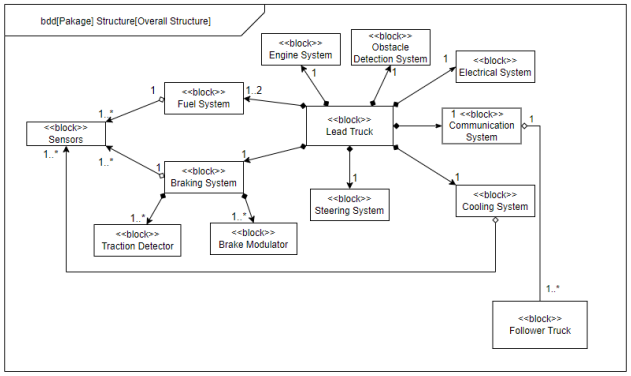


Figure 4: Main block diagram

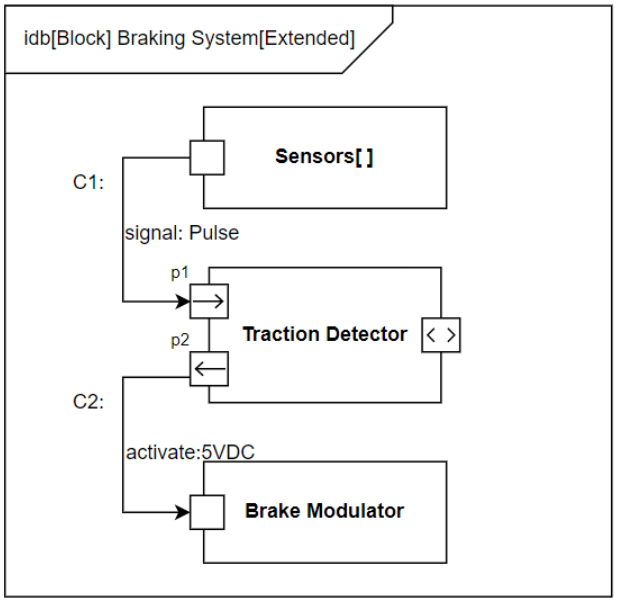


Figure 5: IBD - Brake System

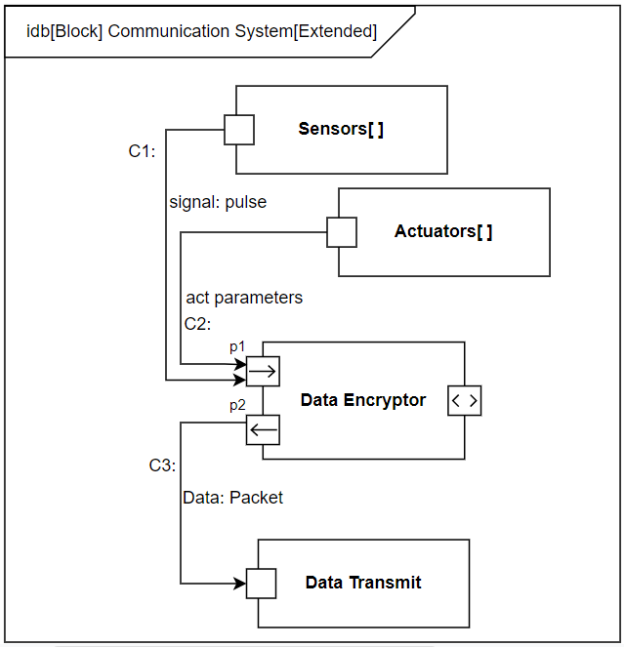


Figure 6: IBD - Communication System

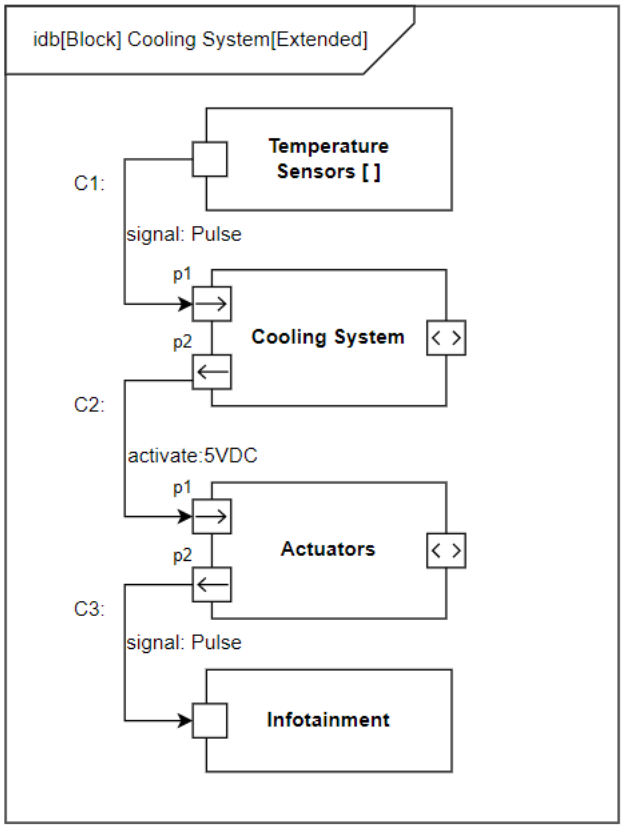


Figure 7: IBD - Cooling System

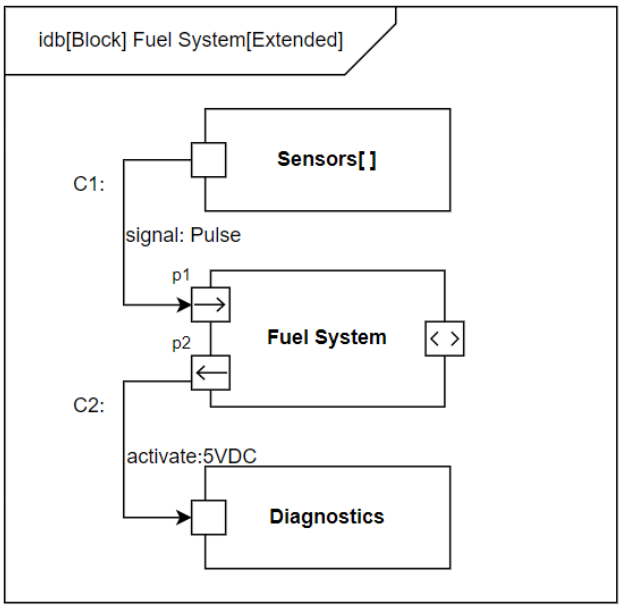


Figure 8: IBD - Fuel System

Figure 9 represents the internal block diagram of steering system. This system shall get the steering ratio and be monitored and plan the direction accordingly with lead truck.

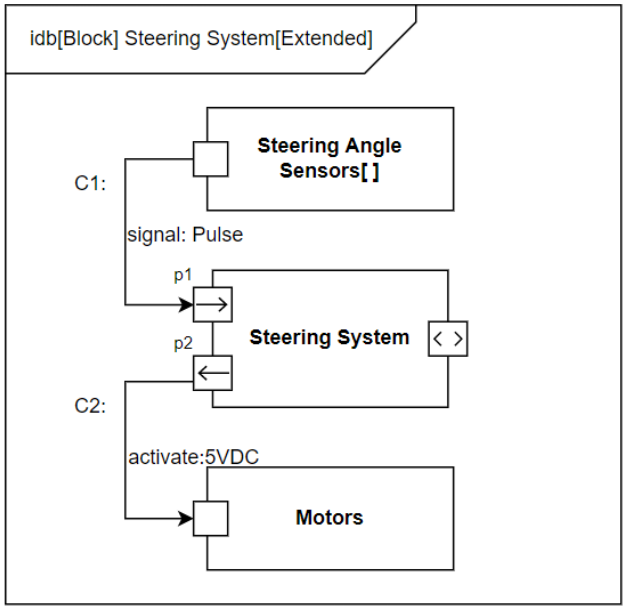


Figure 9: IBD -Steering System

Figure 10 represents the state machine diagram of obstacle detection system. This system shall monitor and notify the lead and following trucks about the obstacles if detected in their pathway. .

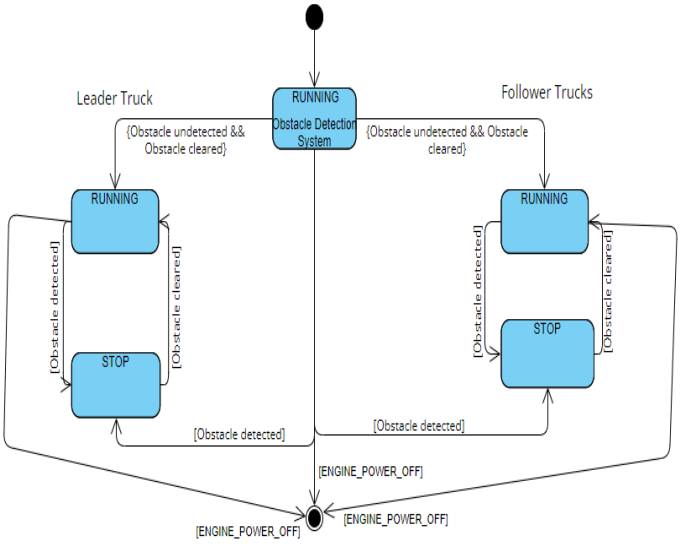


Figure 10: State machine -Obstacle Detection System

Figure 11 represents the state machine diagram of fuel level indicator system. This system shall notify lead truck driver about the fuel level in the tanks of the trucks. .

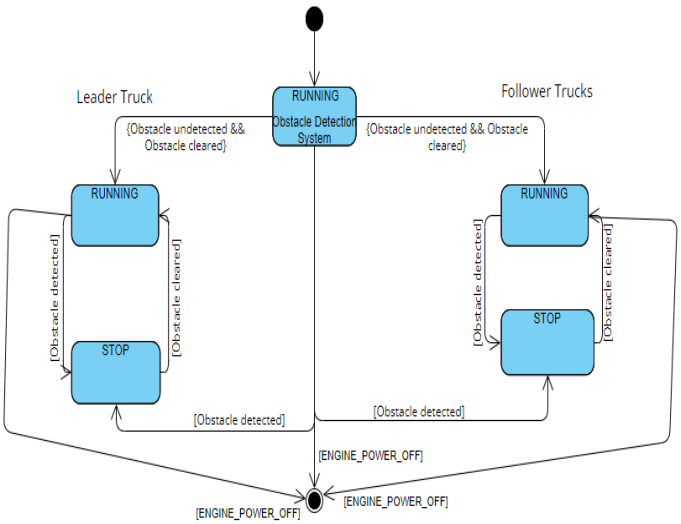


Figure 11: State machine -Fuel level Indicator System

Figure 12 represents the state machine diagram of Traction Control system. This system shall slow down the trucks in case of high spinning of wheels or slippery road surface.

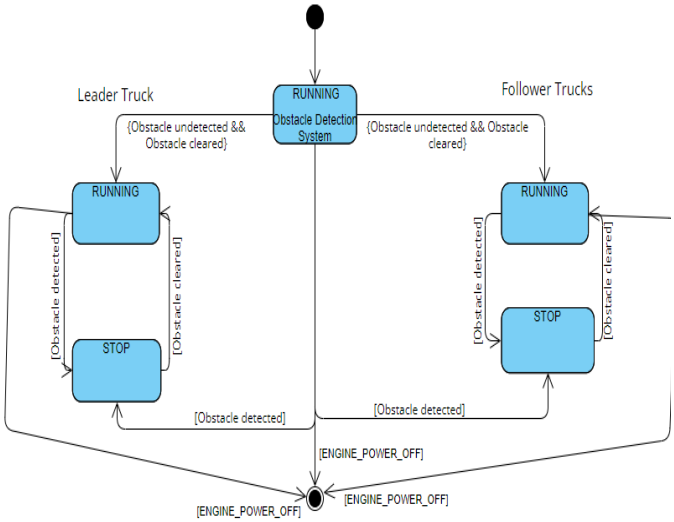


Figure 12: State machine -Traction Control System

III. IMPLEMENTATION

Main System that needs to be considered for longitudinal module is maintaining equidistance and obstacle detection system. Obstacle can be in front of lead truck or between the platoons. All the trucks are expected to detect the obstacle and act according to the situation.

Obstacle range information can be obtained from ladar (laser ranging), sonar (sound ranging) or vision based (video and image processing) techniques. In this study we are taking proximity sensors in the truck into consideration for detecting the obstacle or incoming traffic. We have taken Arduino as the GPU and Ultrasonic sensor as proximity sensor. When the obstacle is detected by lead truck in certain distance, farther than critical distance, it indicates an alert signal to lead driver and the signal to stop or slowdown will be transmitted to follower trucks. Likewise, when obstacle is sensed by follower truck, it slows down or stop and send information to lead truck and other following trucks behind it.

Image above depicts the two scenarios of obstacle detection. Firstly, obstacle in critical distance which alerts the system with stop signal and in another scenario, obstacle in not so critical distance which send just alert signal. Obstacle detection algorithm includes polling of sensor input to the GPU. Our approach is reflexive based on the instantaneous local perception of obstacle position. This system prioritizes the obstacle based on their distance from the truck and take actions to avoid obstacles and crash. Another system that is implemented in this study is fuel level monitoring and indication. All the trucks fuel level is measured and the status is indicated to lead truck.

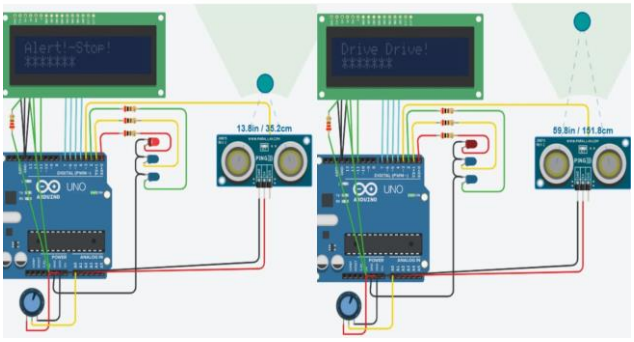


Figure 13: Module Implementation

IV. SCHEDULING

Figure 14 represents the Scheduling of Obstacle Detection System.

scheduling.xml

General | Scheduler | Processors | Tasks

id	Name	Task type	Abort on miss	Act. Date (ms)	Period (ms)	List of Act. dates (ms)	Deadline (ms)	WCET (ms)
1	T1	Periodic	Yes	0	30	-	20	1.99
2	T2	Periodic	Yes	0	25	-	20	5

Figure 14: EDF and RM Scheduling

```
Performing analysis started
Result:
Monitoring the obstacles by emitting sound waves: wcrt=6.990000, bcrt = 1.990000
Sending the notification to the trucks : wcrt=6.990000, bcrt = 5.000000
```

Figure 15: Response Time using pycpa

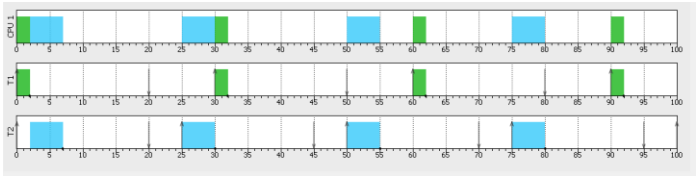


Figure 16: Timing Diagram for EDF

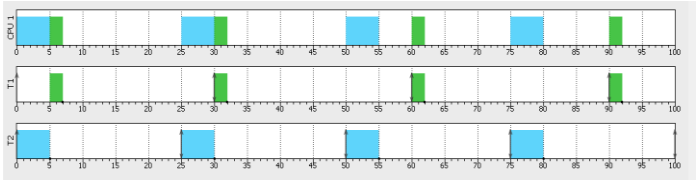


Figure 17: Timing Diagram for RM

V. TESTING

Google Test is a unit testing library for the C++ programming language, based on the xUnit architecture. The library is released under the BSD 3-clause license. It can be compiled for a variety of POSIX and Windows platforms, allowing unit-testing of C sources as well as C++ with minimal source modification [7].

```
TEST_F(TEST_FUNC, TestCase1) {

    cout << "-------------------------------------------------------------------------------------------------------------" << endl;
    cout << "SRS ID : PLATOON_SRS_0001" << endl;
    cout << "-------------------------------------------------------------------------------------------------------------" << endl;
    cout << "Requirement : Slave Trucks should notify Application whenever Reception Timeout Occurs, if not set Fault." << endl;
    cout << "Category : Integration Test Case" << endl;
    cout << "Test Case 1 : Check if any Fault Detected during Transmission and Reception of Data." << endl;
    cout << "Expected Result: 0" << endl;
    cout << "Actual Result: " << Fault.PlatoonSystem() << endl;
    ASSERT_EQ(Fault.PlatoonSystem(), 0);
    cout << "Test Case 1: Passed" << endl;
    cout << "Analysis : The TimeoutNotification to be configured for every Packet in the Application to trigger the notification when any Packet is not received within the Configured Time."
    cout << "-------------------------------------------------------------------------------------------------------------" << endl;
}
```

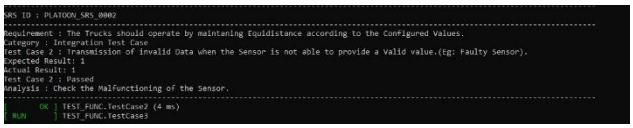
Figure 18: Test Case 1

```
SRS ID : PLATOON_SRS_0001
-------------------------------------------------------------------------------------------------------------
Requirement : Slave Trucks should notify Application whenever Reception Timeout Occurs, if not set Fault.
Category : Integration Test Case
Test Case 1 : Check if any Fault Detected during Transmission and Reception of Data.
Expected Result: 0
Actual Result: 0
Test Case 1: Passed
Analysis : The TimeoutNotification to be configured for every Packet in the Application to trigger the notification when any Packet is not received within the Configured Time.
-------------------------------------------------------------------------------------------------------------

      OK | TEST_FUNC.TestCase1 (0 ms)
[ RUN      ] TEST_FUNC.TestCase2
```

Figure 19: Result 1

```
TEST_F(TEST_FUNC, TestCase2) {

    Notification.State = TEST_FAILURE;
    cout << "-------------------------------------------------------------------------------------------------------------" << endl;
    cout << "SRS ID : PLATOON_SRS_0002" << endl;
    cout << "-------------------------------------------------------------------------------------------------------------" << endl;
    cout << "Requirement : The Trucks should operate by maintaning Equidistance according to the Configured Values." << endl;
    cout << "Category : Integration Test Case" << endl;
    cout << "Test Case 2 : Transmission of Invalid Data when the Sensor is not able to provide a Valid value.(Eg: Faulty Sensor)." << endl;
    cout << "Expected Result: 1" << endl;
    cout << "Actual Result: " << Fault.PlatoonSystem() << endl;
    ASSERT_EQ(Fault.PlatoonSystem(), 1);
    cout << "Test Case 2 : Passed" << endl;
    cout << "Analysis : Check the Malfunctioning of the Sensor." << endl;
    cout << "-------------------------------------------------------------------------------------------------------------" << endl;
}
```

Figure 20: Test Case 2



```
SRS ID : PLATOON_SRS_0002
-------------------------------------------------------------------------------------------------------------
Requirement : The Trucks should operate by maintaning Equidistance according to the ConFigured Values.
Category : Integration Test Case
Test Case 2 : Transmission of Invalid Data when the Sensor is not able to provide a Valid value.(Eg: Faulty Sensor).
Expected Result: 1
Actual Result: 1
Test Case 2 : Passed
Analysis : Check the Malfunctioning of the Sensor.
-------------------------------------------------------------------------------------------------------------

      OK | TEST_FUNC.TestCase2 (4 ms)
[ RUN      ] TEST_FUNC.TestCase3
```

Figure 21: Result 2

Requirements Analysis for Integration

Requirements				Requirement Analysis			
Integration Requirements							
Sl.No.	SRS ID	Requirement Description	Requirement Type	Category	Test Case Description	Analysis Comment	Detailed Information
1	PLATOON_SRS_0001	Slave Trucks should notify Master whenever Reception Timeout Occurs, if not set Fault.	Functional Requirement	Integration Test Case	Check if any Fault Detected during Transmission and Reception of Data.	the TimeoutNotification to be configured for every Packet in the Application to trigger the notification when any Packet is not received within the Configured Time.	
2	PLATOON_SRS_0002	The Trucks should operate by maintaning Equidistance according to the Configured Values.	Functional Requirement	Integration Test Case	Transmission of Invalid Data when the Sensor is not able to provide a Valid value.(Eg: Faulty Sensor).	Check the Malfunctioning of the Sensor.	
3	PLATOON_SRS_0003	Transmission shall be stopped during the Power-Off of the ECU.	Functional Requirement	Integration Test Case	Master or Slave Truck face Power-Off Condition.	Trucks needs to Stop due to Hardware / Software Failure.	
4	PLATOON_SRS_0004	Failure in the Braking System due to Low levels in the Rear Reservoir, Broken Wheel Speed Sensors or System is turned Off.	Functional Requirement	Integration Test Case	To Check the Activation of ABS and UBS Systems.	The Hardware / Software part of the ABS and UBS needs to be Corrected.	
5	PLATOON_SRS_0005	Any failures detected in the Fuel System.	Functional Requirement	Integration Test Case	To Check the performance of Fuel Filter, Lubricator and Fuel Pump.	issues related to Fuel System needs to be Corrected.	
6	PLATOON_SRS_0006	Any failures detected in the Cooling System.	Functional Requirement	Integration Test Case	To Check the Working of Temperature Sensor.	Check the Malfunctioning of the Sensor.	

Figure 22: Test Specification

VI. CONCLUSION

Truck platooning will improve traffic mobility and safety, while reducing manpower required in transportation. Using existing infrastructure to increase logistical efficiency, truck platooning aims to increase productivity and decrease resource consumption. Platooning concepts are breaking barriers with the growth in automated vehicle market and the commercialization of 5G and C-ITS technologies. This paper aims to analyze, model and design a truck platoon system. Sysml a general-purpose architecture modeling language helps in designing a complex Embedded System in a efficient way. Further challenges in truck platooning will become evident with pilot projects that are being realized by major automotive firms..

VII. REFERENCES

Sai Ravela, Bruce A. Draper, Allen Hanson "A practical obstacle detection and avoidance system" Appplication of Computer Vision, 1994

André Souza Mendes, A.T. Fleury, Marko Ackermann, Fabrizio Leonardi "Heavy-duty Truck Platooning" 24th ABCM International Congress of Mechanical Engineering - Jan 2017

Peters, A.A., Middleton, R.H. and Mason, O., 2014. "Leader tracking in homogeneous vehicle platoons with broadcast delays". Automatica, Vol. 50, No. 1, pp. 64–74. doi:10.1016/j.automatica.2013.09.034.

Zheng, Y., Li, S.E., Li, K., Borrelli, F. and Hedrick, J.K., 2017a. "Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies". IEEE Transactions on Control Systems Technology, Vol. 25, No. 3, pp. 899–910. doi:10.1109/TCST.2016.2594588.

Zheng, Y., Li, S.E., Li, K. and Ren, W., 2017b. "Platooning of connected vehicles with undirected topologies: Robustness analysis and distributed h-infinity controller synthesis". IEEE Transactions on Intelligent Transportation Systems, Vol. PP, No. 99, pp. 1–12. doi:10.1109/TITS.2017.2726038.

Herman, I., Martinec, D., Hurák, Z. and Šebek, M., 2015. "Nonzero bound on fiedler eigenvalue causes exponential growth of h-infinity norm of vehicular platoon". IEEE Transactions on Automatic Control, Vol. 60, No. 8, pp. 2248– 2253. doi:10.1109/TAC.2014.2366980.

[7] https://en.wikipedia.org/wiki/Google_Test