

A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is a light green color. They are positioned diagonally, with the blue one in front of the green one.

Truck Platooning

Team Outbreak



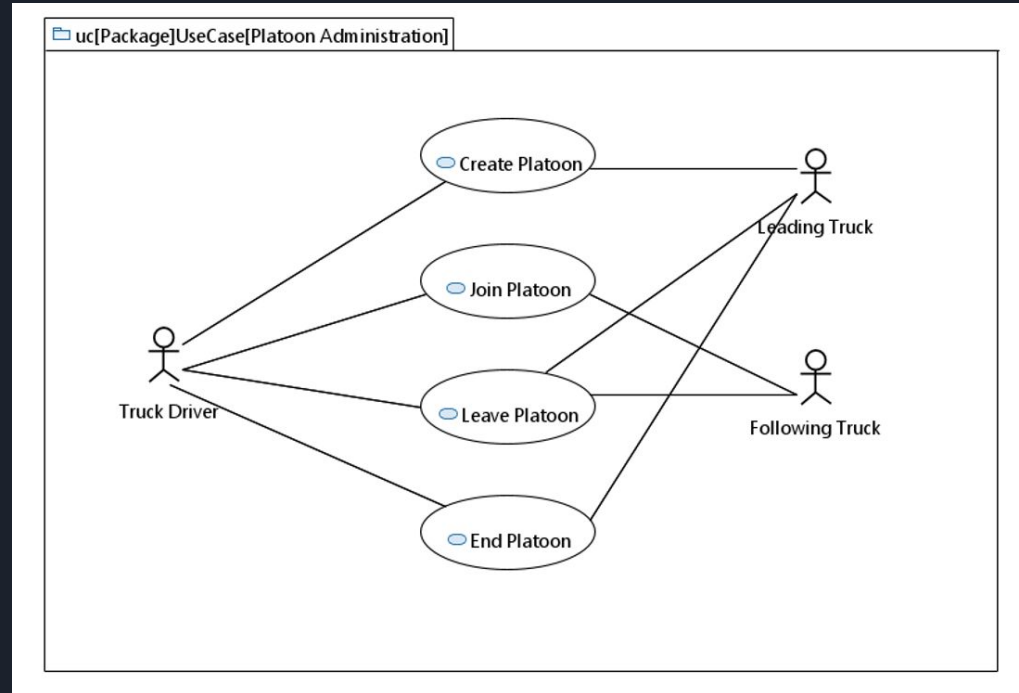
Requirement Diagram

- Requirement Elicitation and analysis where done before starting the requirement diagram.
- Main requirements are analysed and specified as

Use Case Diagram

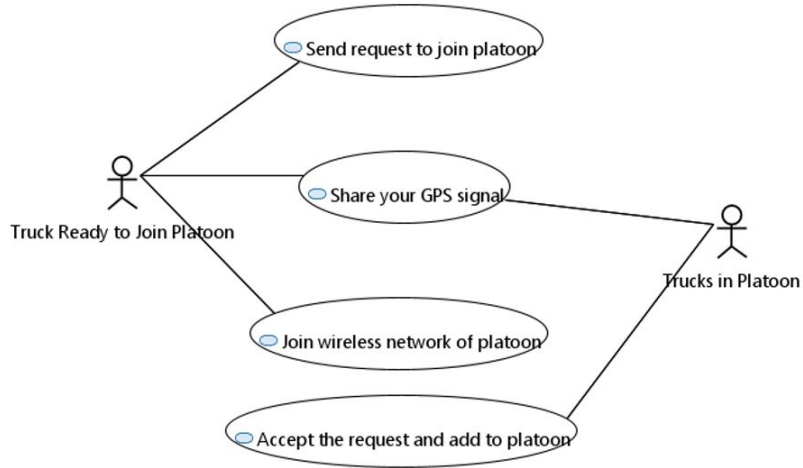
Use case diagram defines the operations that users want to perform on the system through a function.

It consists of four main elements: Actors, System, Use Cases and their relations.

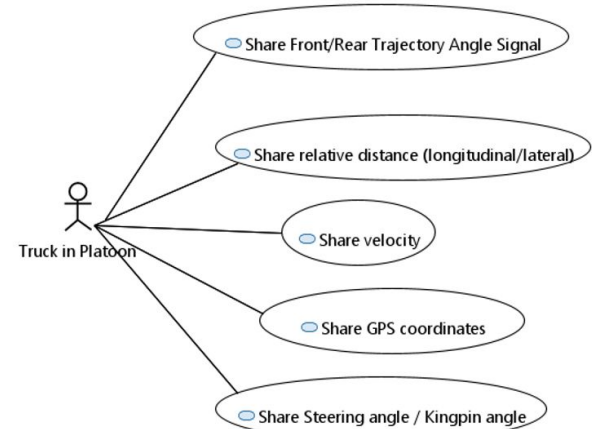


Use Case Diagram

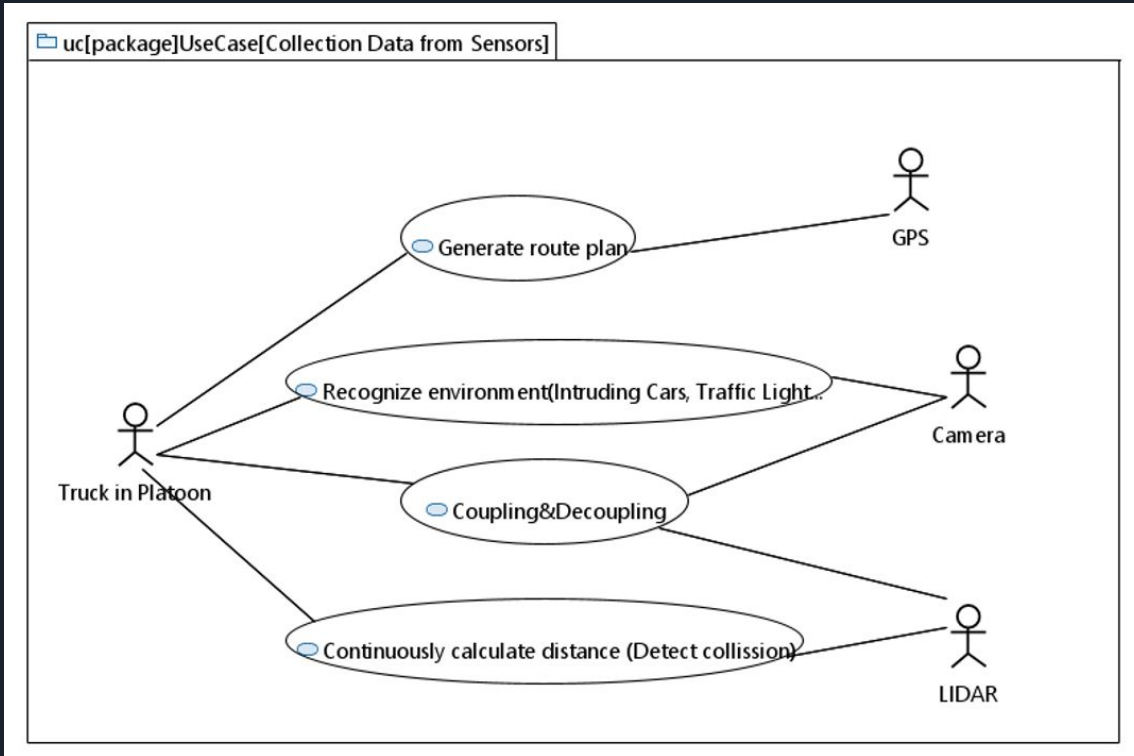
uc[package]UseCase[Joining to The Platoon]



uc[package]UseCase[Data Sharing in Platoon]



Use Case Diagram





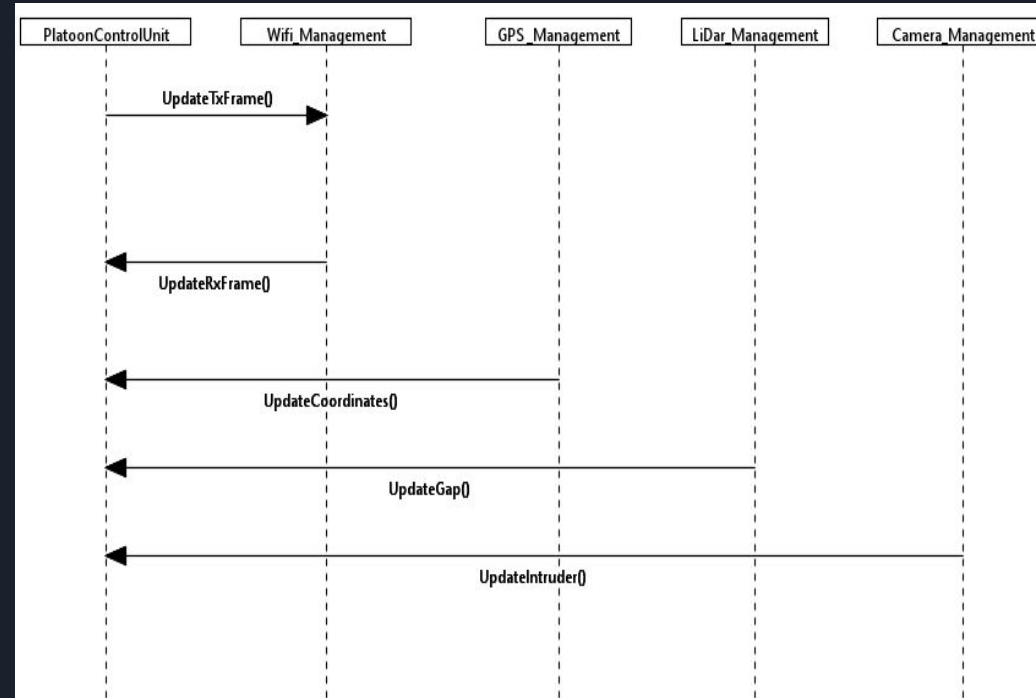
Activity Diagram



Block Diagram

Sequence Diagram

- UpdateTxFrame to update the Wifi credentials
- UpdateRxFrame to show the main controller the received output
- UpdateCoordinates is sent from Gps to update longitude and latitude of first truck
- UpdateGap is getting data from the Radar and updates distance between the trucks
- As there is intruder in the system PCU got informed by UpdateIntruder message

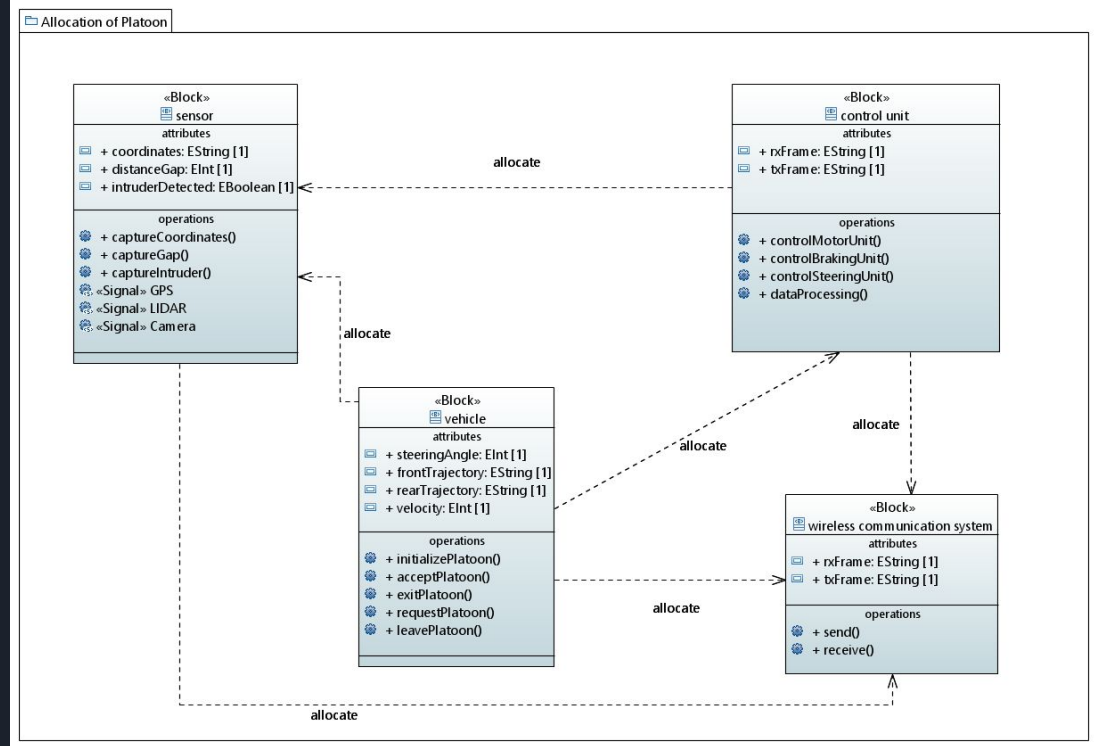




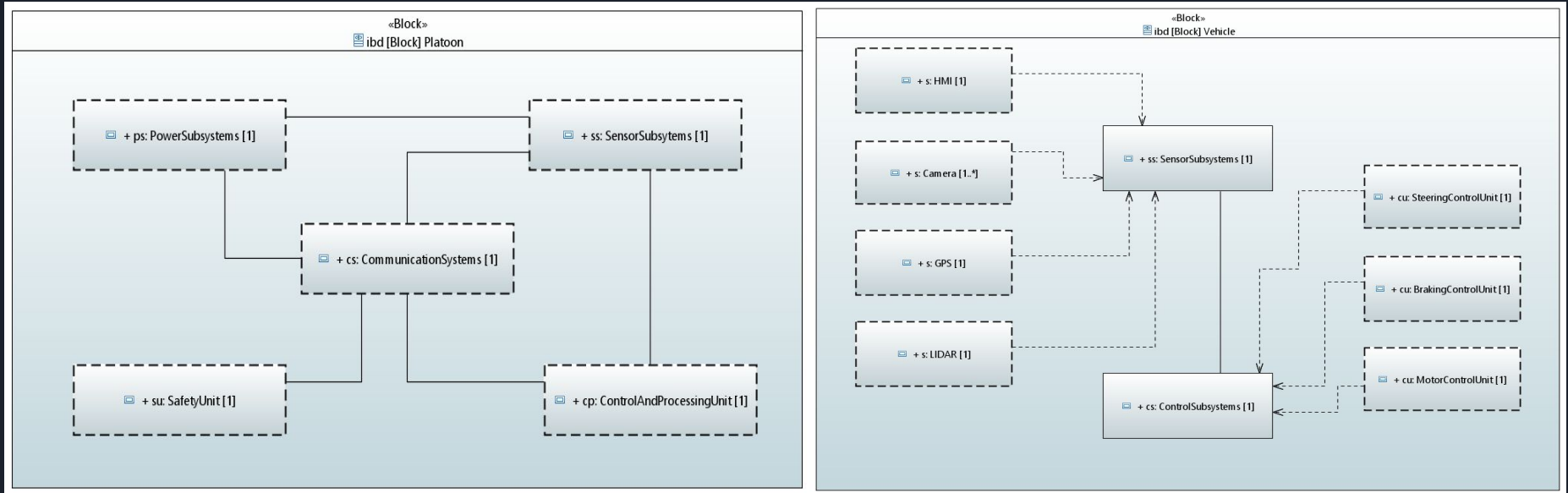
Parametric Constraint Diagram

Allocation Diagram

Allocation diagrams are used to define relationship of various parts of the model.



Internal Block Diagram





State machine Diagram



Implementation



Scheduling



Inspection

Unit testing

The screenshot displays the Microsoft Visual Studio IDE with the following components:

- Debug Console:** Shows the output of a unit test run. It lists five tests: `updateCoordinatesTest.PCUTest`, `truckSelection.PCUTest`, `detectIntruderTest.PCUTest`, `gapUpdateTest.PCUTest`, and a global test environment tear-down. All tests passed, and the total execution time was 4 ms.
- Solution Explorer:** Displays the project structure for `UnitTestFile`, including `packages.config`, `pch.cpp`, `pch.h`, and `test.cpp`.
- Output Window:** Shows the status of the debug session, indicating that the program `UnitTestFile.exe` has exited with code 0 (0x0).

```
test[-----] 1 test from updateCoordinatesTest
[ RUN      ] updateCoordinatesTest.PCUTest
[ OK       ] updateCoordinatesTest.PCUTest (0 ms)
[-----] 1 test from updateCoordinatesTest (0 ms total)

[-----] 1 test from truckSelection
[ RUN      ] truckSelection.PCUTest
[ OK       ] truckSelection.PCUTest (0 ms)
[-----] 1 test from truckSelection (0 ms total)

[-----] 1 test from detectIntruderTest
[ RUN      ] detectIntruderTest.PCUTest
[ OK       ] detectIntruderTest.PCUTest (0 ms)
[-----] 1 test from detectIntruderTest (0 ms total)

[-----] 1 test from gapUpdateTest
[ RUN      ] gapUpdateTest.PCUTest
[ OK       ] gapUpdateTest.PCUTest (0 ms)
[-----] 1 test from gapUpdateTest (0 ms total)

[-----] Global test environment tear-down
[=====] 5 tests from 5 test cases ran. (4 ms total)
[ PASSED  ] 5 tests.

C:\Users\mntun\source\repos\UnitTestFile\Debug\UnitTestFile.exe (process 4944) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .

26 |         void PCU::updateGap(int16_t gap)
27 |         {
28 |             measuredDistanceGap = gap;
29 |         }
30 |
31 |         void BRU::updateIntruder(bool detected, status)
```

Output

Show output from: Debug

The thread 0x3b9c has exited with code 0 (0x0).

The program '[4944] UnitTestFile.exe' has exited with code 0 (0x0).

Ready



Component Testing

There are 5 components in the system. PCU, Wifi, Lidar, Camera and Gps. PCU is main object so I tested other using PCU object functions

- Test Camera if it informs the PCU about intruder
- Test GPS whether it updates coordinates correctly
- Test Lidar to send distance data to PCU and check if PCU is responding in correct way
- Test Wifi via comparing output from it with input from PCU