BUE
The British University in Egypt
الجامعة البريطانية فى مصر

RTS

Real Time Systems

19CSSE03H - Final Phase

AUTOMOBILE MANAGEMENT SYSTEM - Group 7

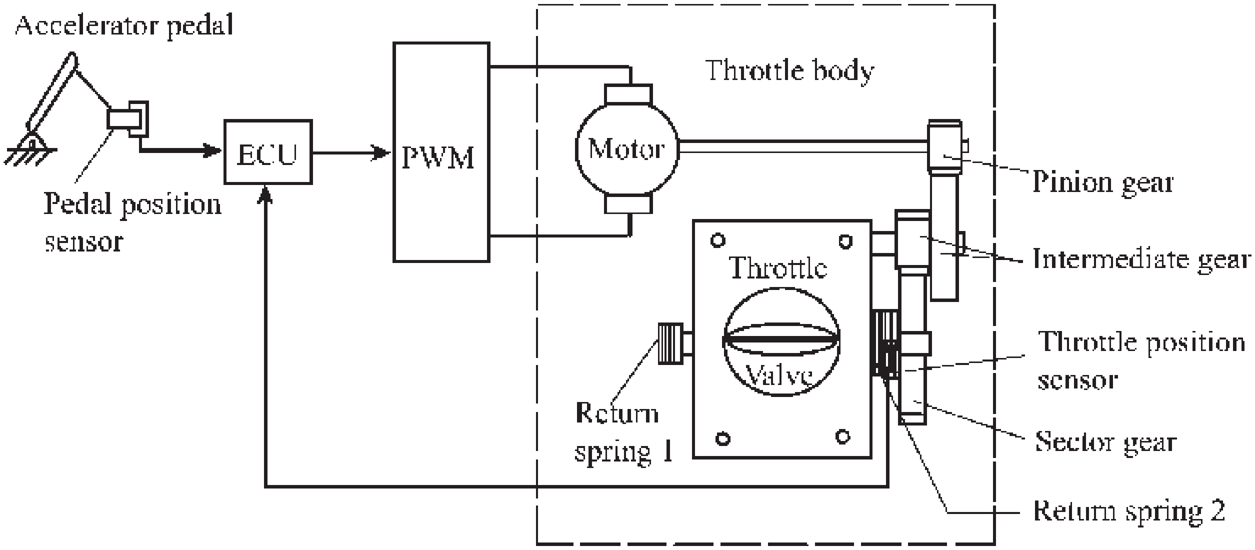
FINAL REPORT

Omar Khaled	161417
Mahmoud Magdy	160024
Samir Mohamed	162462
Moamen Tawfik	161469
Anas Mohamed	138778

Overview

The Automobile Management System is a real time system that consists of electrical sensors and mechanical parts composed together in order to work with one another to reduce the driver's strain while driving on long journeys. On the other hand, the system comes equipped with a numeric keyboard as an input, and CRT output monitor that tracks the vehicle's mileage and average fuel consumption, and the average speed in a certain trip upon user's selection. Therefore, Real Time Systems shall be taken into development and modeling conception as for this study.

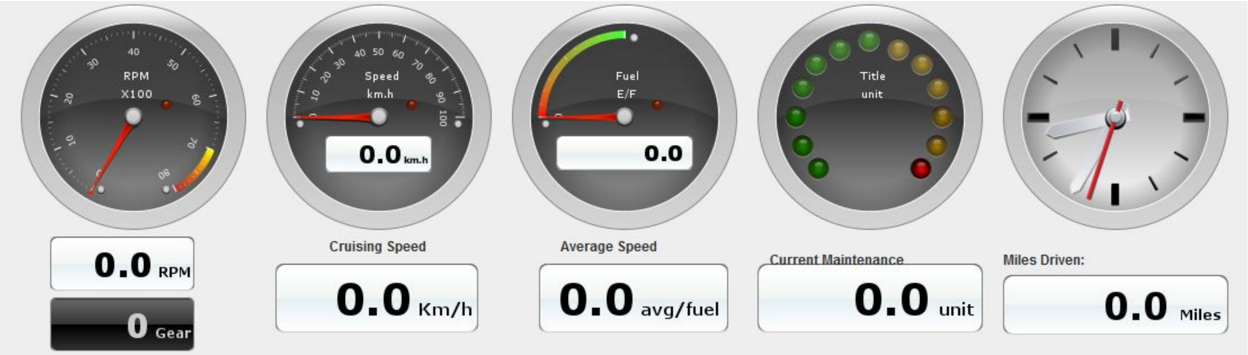
The Cruise Control System, is a system that maintains the vehicle's speed upon user's selection meanwhile the engine is running and its current gear is the top gear, it maintains the speed by controlling the throttle body through the actuator attached to it by a range of volts from 0 to 8 (closed – fully opened) in order to hold the car at that desired speed, as well as the vehicle's speed can be increased dynamically by depressing on the accelerator regardless of the current selected speed to be cruised on, but once the pedal is held out, the vehicle will be running back on the previously cruised speed. And vice versa for the braking pedal, until the speed reaches the previously selected cruising speed.



Throttle in action[1]

Also, the system will be outputting the speed on a speedometer that measures the speed and distance by counting the revolutions it receives from the drive shaft sensor to draw the actual speed after the effect of the tire size and wear.

The system will be monitoring the vehicle’s mileage, and notify the driver of determined factory scheduled maintenance, such as regular oil and filter changes, and such on.



Monitor displaying vehicle stats[2]*

CRT Monitor that displaying car’s current stats including the gauges, RPM showing the rotations of the engine, Speedometer showing the current speed of the vehicle, Fuel gauge, Maintenance gauge displaying the required maintenance “alert”, Clock, Vehicle’s drive shaft rotations “SRPM Shaft rotates per minute”.

The LCD displays show the RPM, current gear, the desired “selected” cruising speed, Average fuel used in 100m, upcoming maintenance, current miles driven.

Alongside the buttons, Start/Stop Engine, Accelerate, Brake Start/Stop cruising.

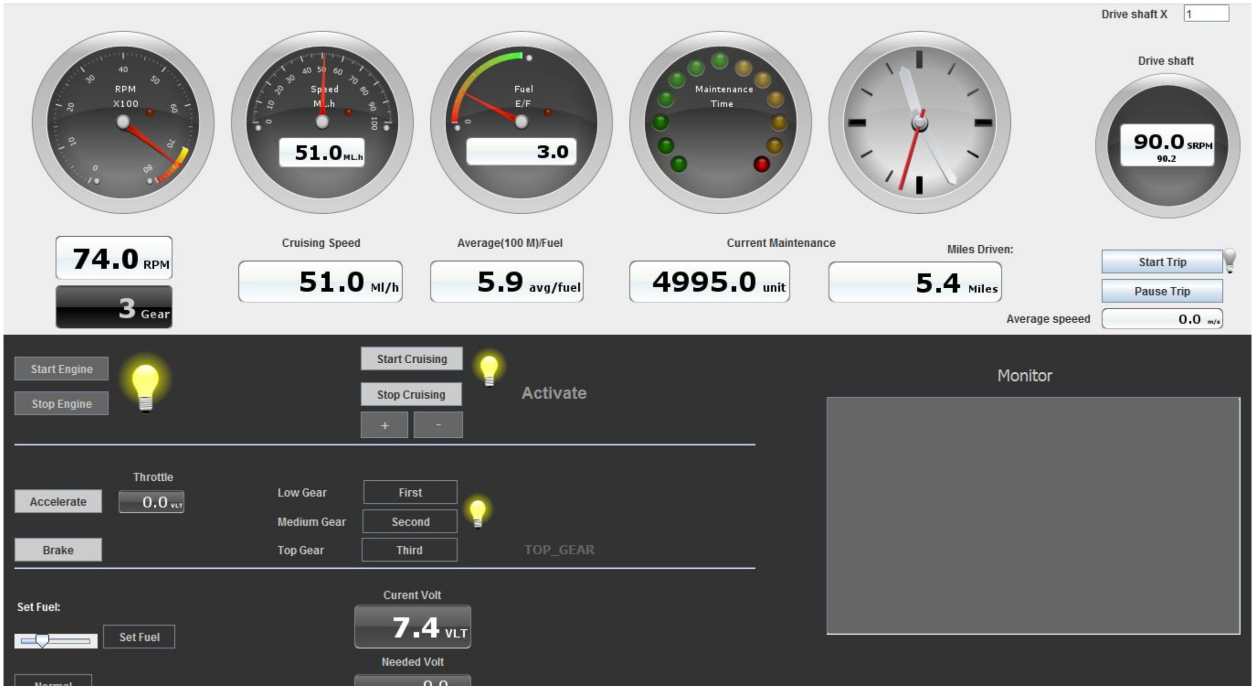
The Transmission “First/Second/Third” gears.

The fuel scaling.

Confirming the required maintenance.

* Screenshot from the actual implemented system

Panel -



Drive shaft X: 1

Drive shaft: 90.0 SRPM 90.2

74.0 RPM

3 Gear

Cruising Speed: 51.0 MI/h

Average(100 M)/Fuel: 5.9 avg/fuel

Current Maintenance: 4995.0 unit

Miles Driven: 5.4 Miles

Average speeed: 0.0 m/s

Throttle: 0.0 VLT

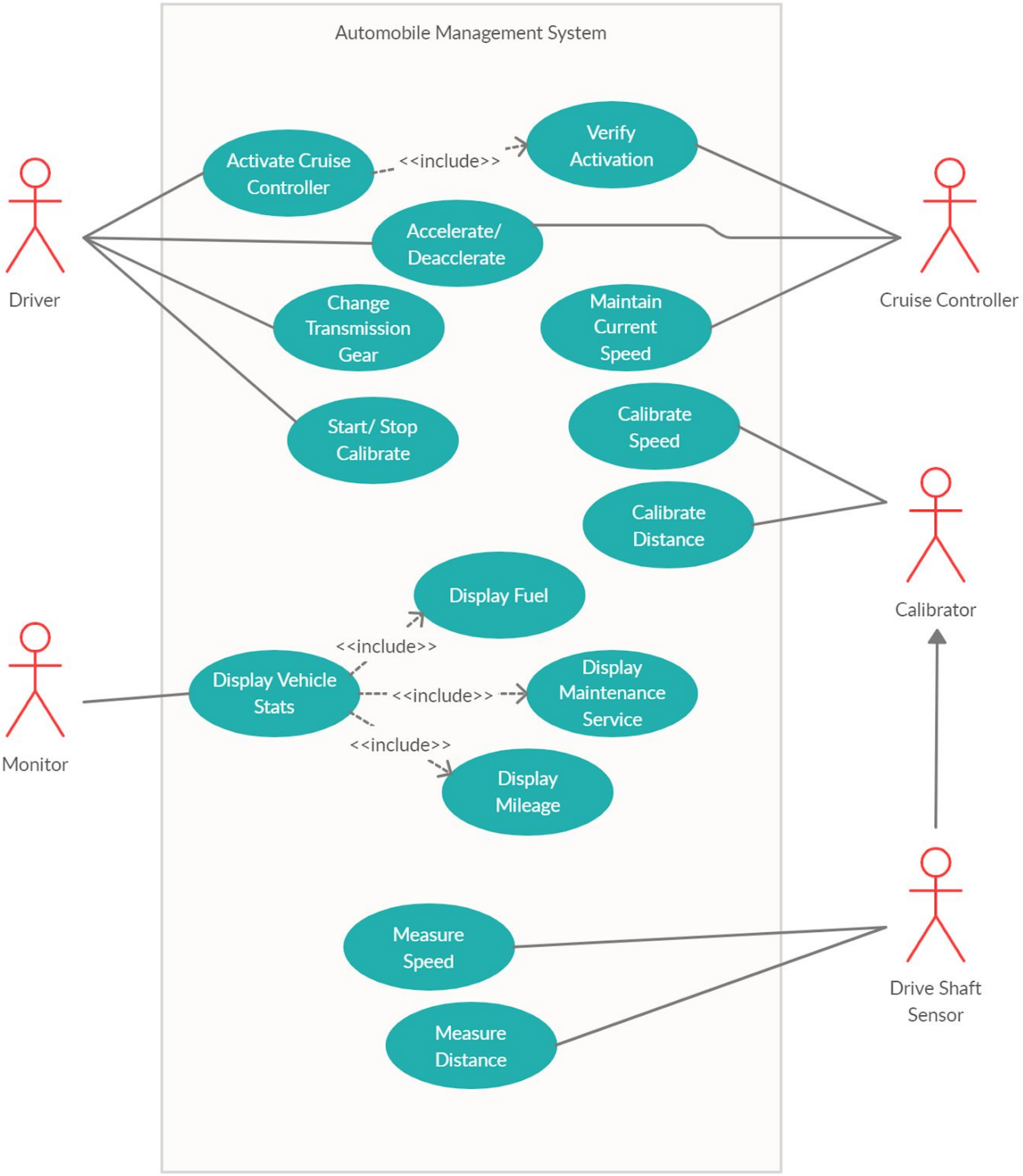
Curent Volt: 7.4 VLT

Needed Volt: 0.0

System GUI Panel[3]

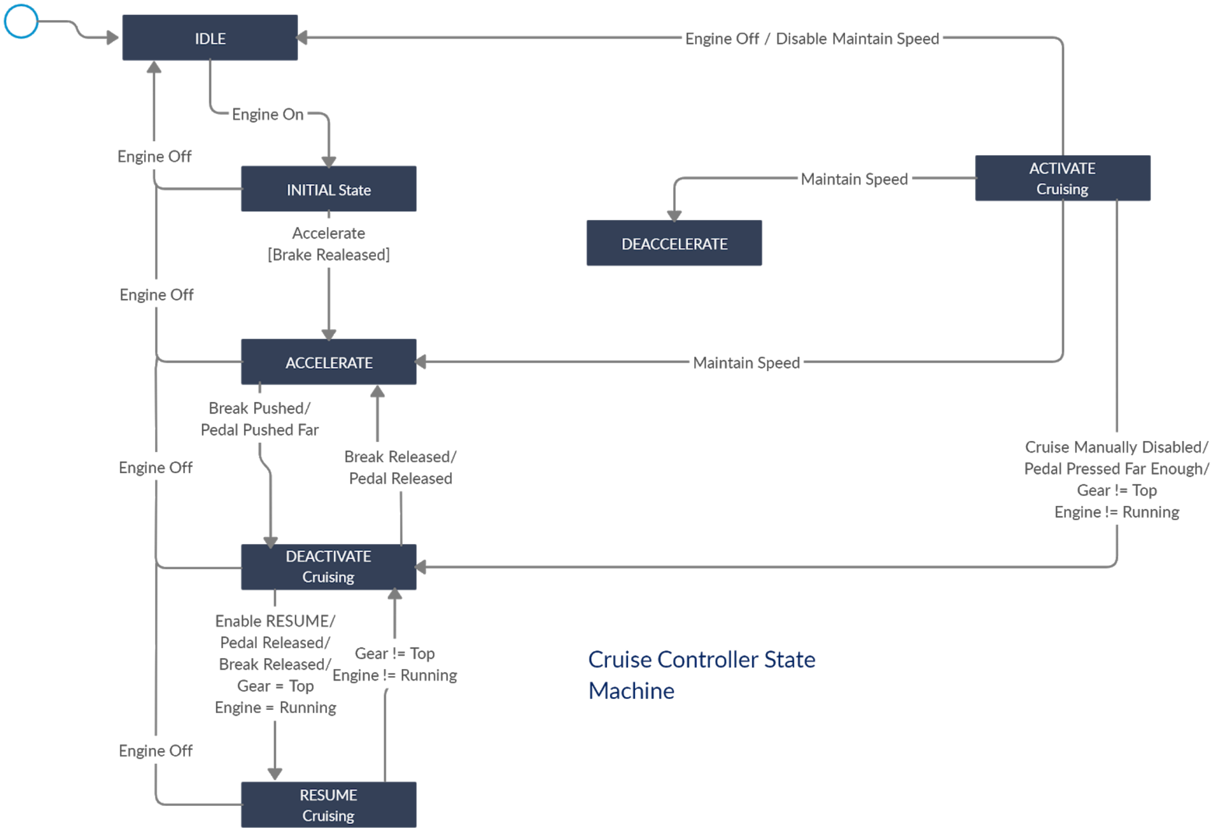
Project In-depth

Use Case Diagram -

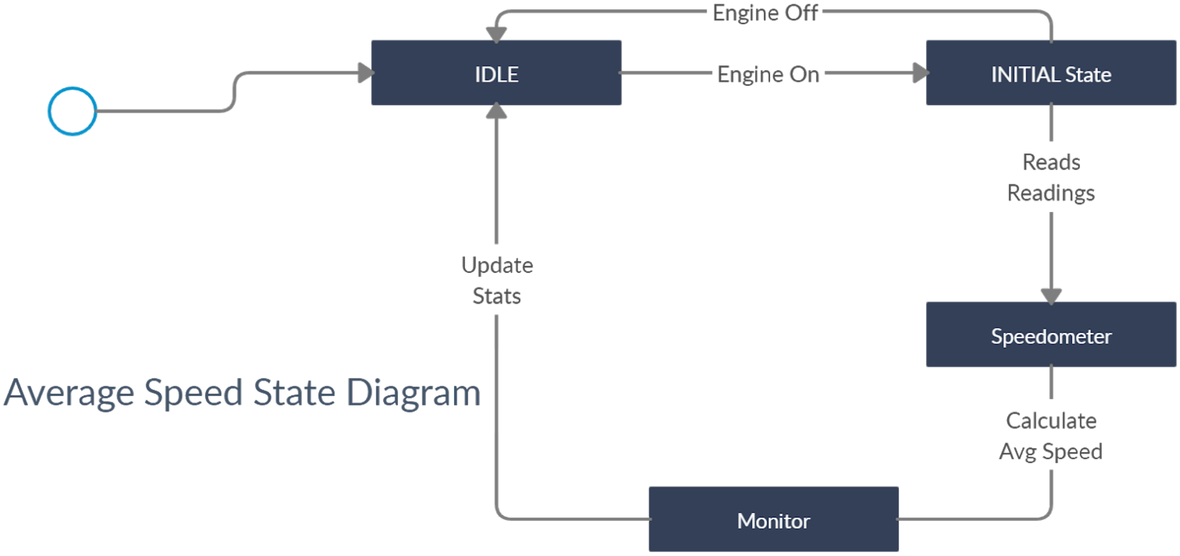


Use Case Diagram[4]

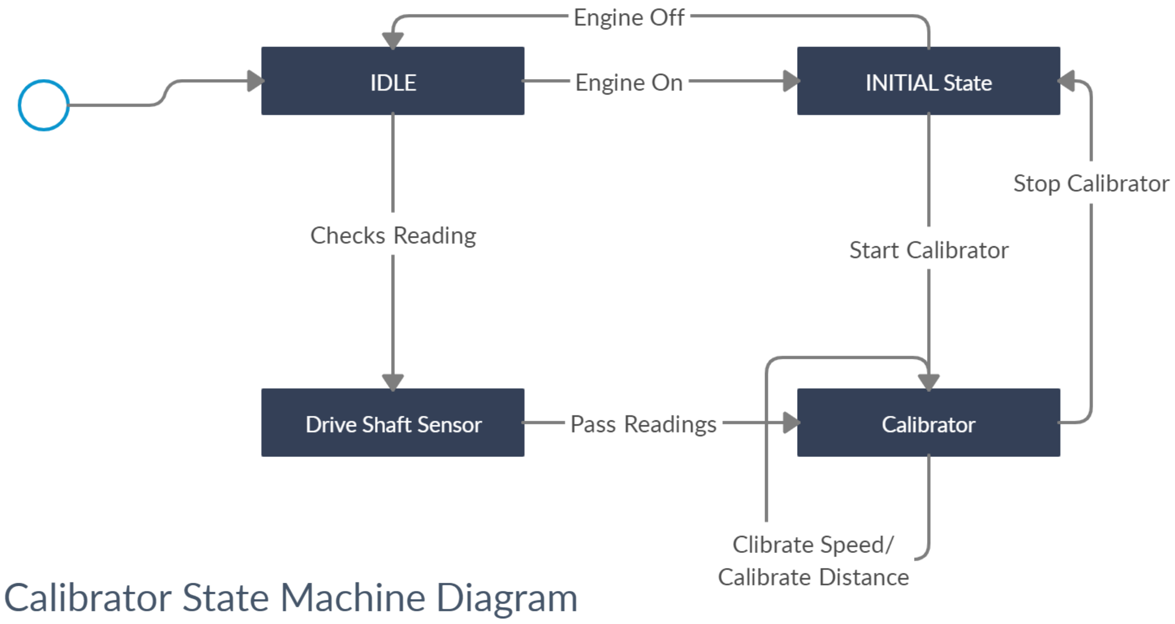
State Machine Diagrams -



Cruise Controller State Machine Diagram[5]

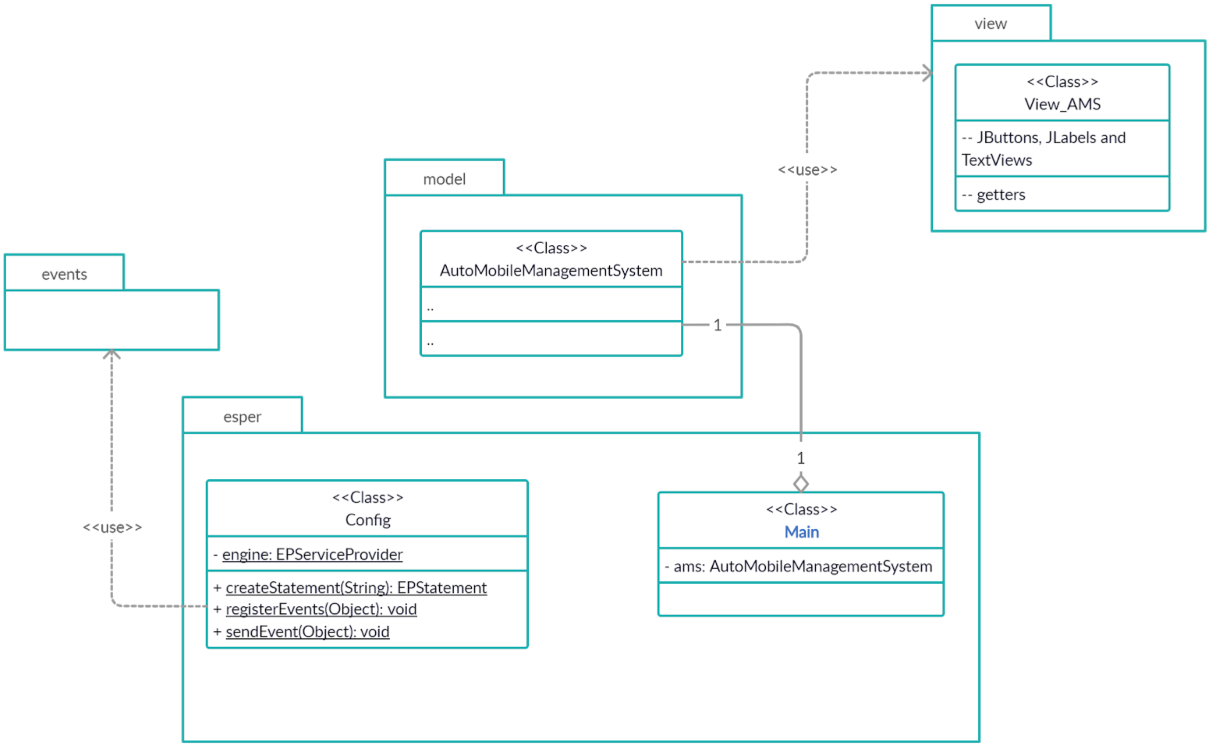


Average Speed State Machine Diagram[6]

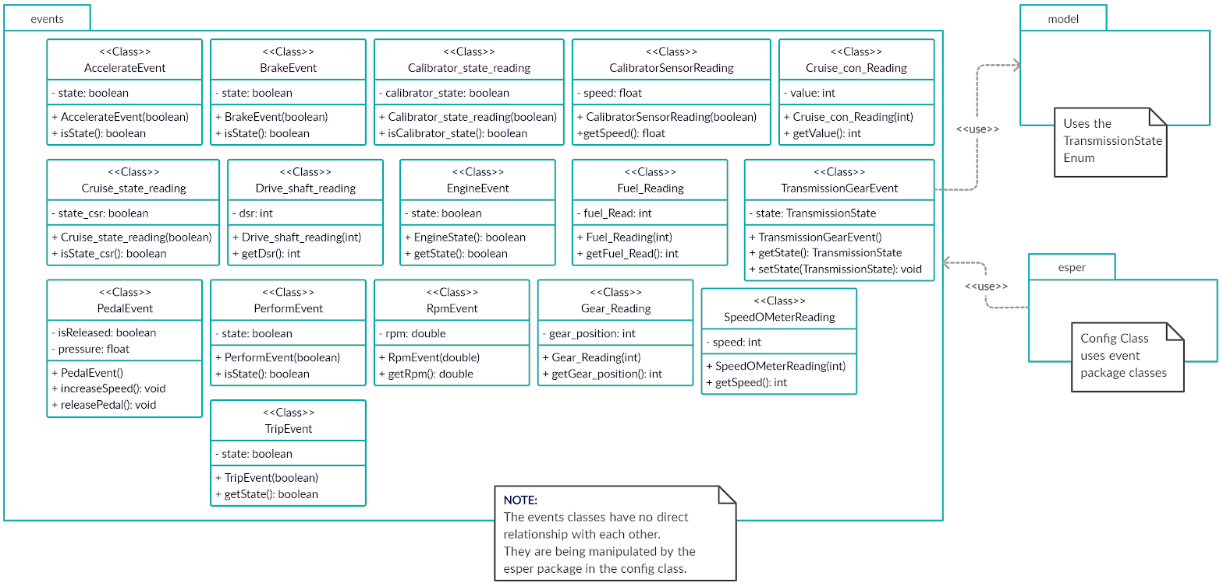


Calibrator State Machine Diagram[7]

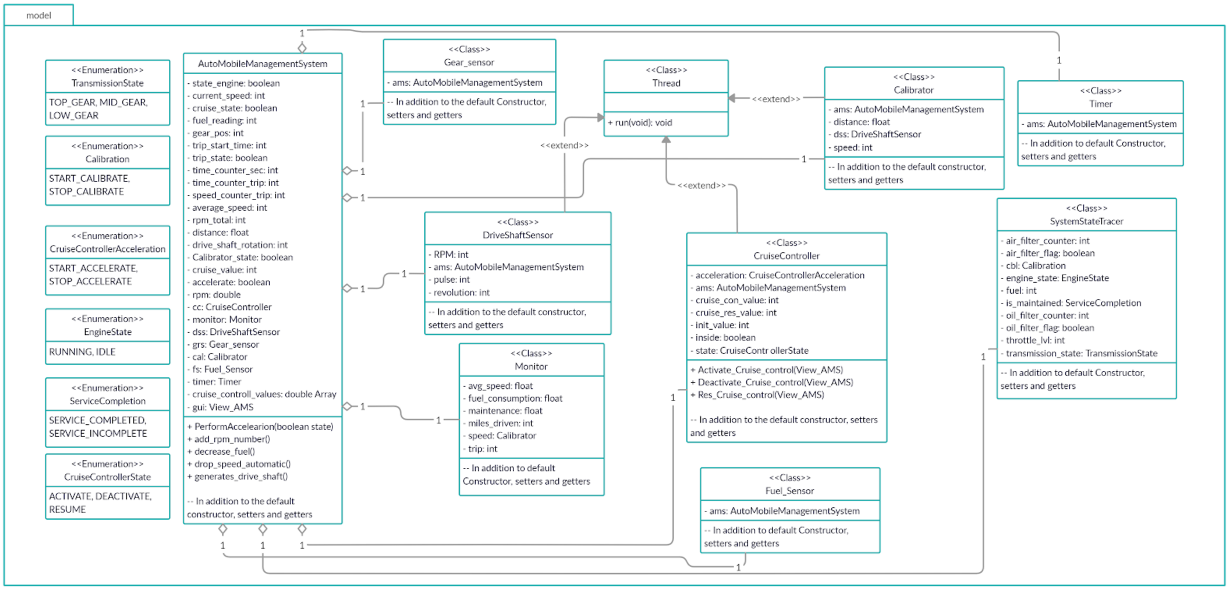
Class Diagram -


view
<<Class>>
View_AMS
-- JButtons, JLabels and TextViews
-- getters
<<use>>
model
<<Class>>
AutoMobileManagementSystem
..
..
1
events
esper
1
<<Class>>
Config
- engine: EPServiceProvider
+ createStatement(String): EPStatement
+ registerEvents(Object): void
+ sendEvent(Object): void
<<use>>
<<Class>>
Main
- ams: AutoMobileManagementSystem

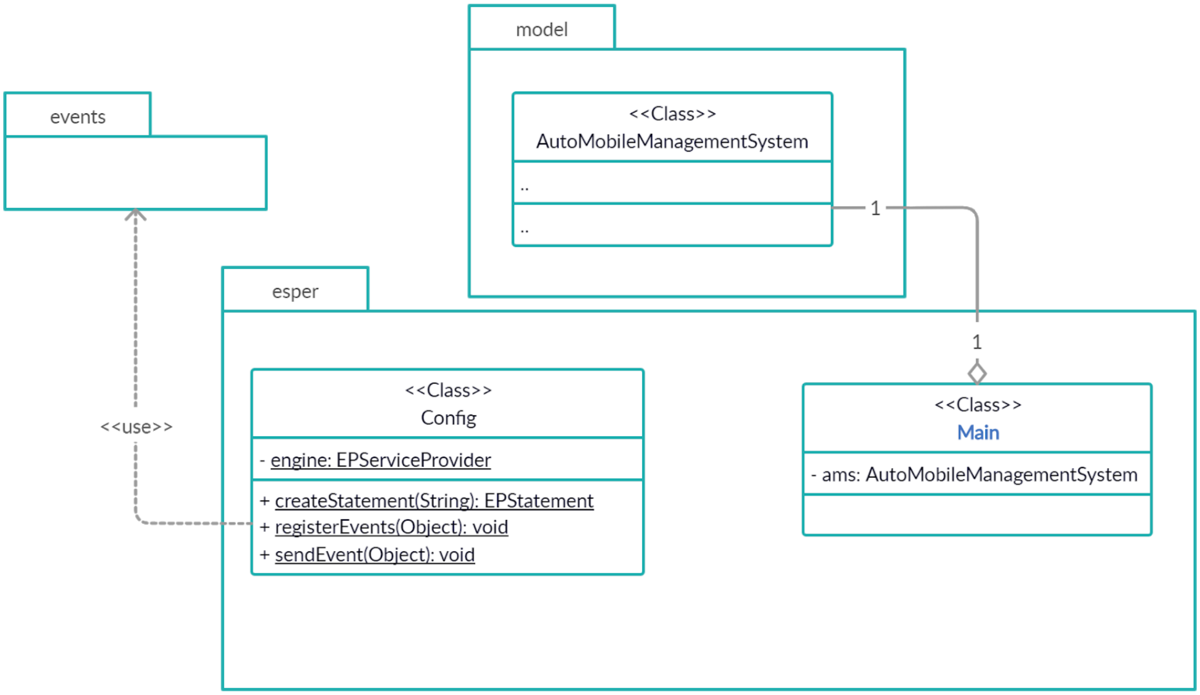
Packages relationships[8]



Events package[9]

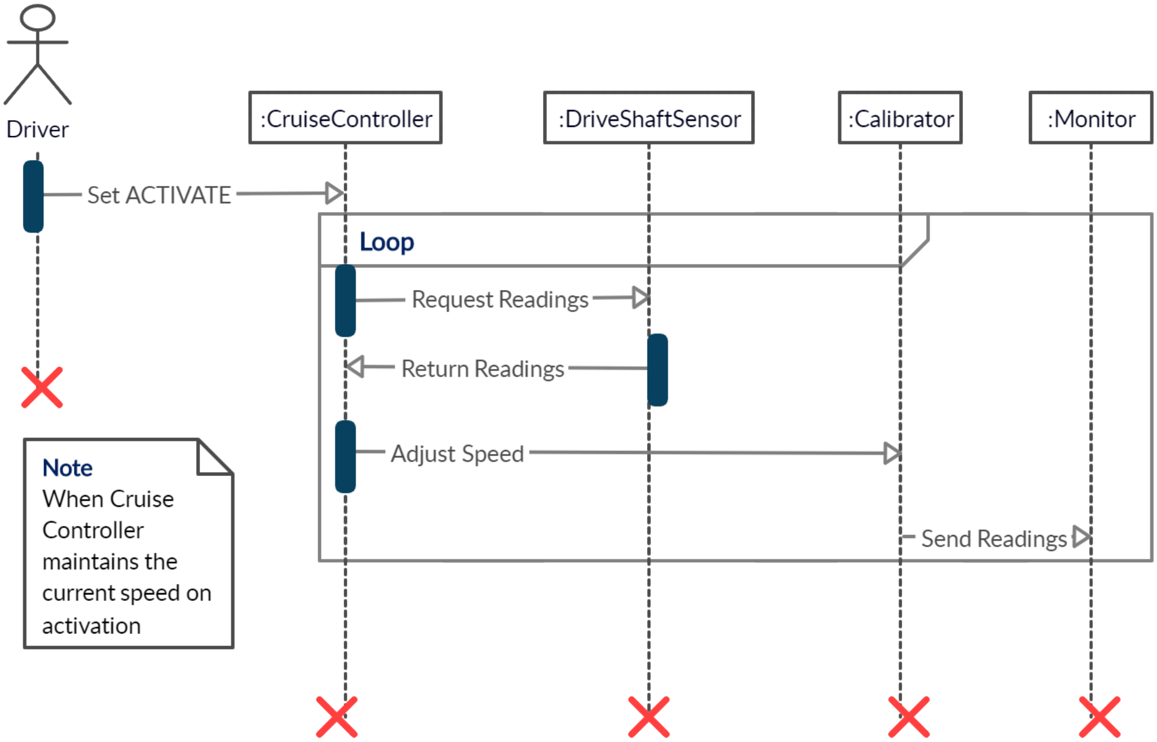

model
<<Enumeration>>
TransmissionState
TOP_GEAR, MID_GEAR, LOW_GEAR
<<Enumeration>>
Calibration
START_CALIBRATE, STOP_CALIBRATE
<<Enumeration>>
CruiseControllerAcceleration
START_ACCELERATE, STOP_ACCELERATE
<<Enumeration>>
EngineState
RUNNING, IDLE
<<Enumeration>>
ServiceCompletion
SERVICE_COMPLETED, SERVICE_INCOMPLETE
<<Enumeration>>
CruiseControllerState
ACTIVATE, DEACTIVATE, RESUME
AutoMobileManagementSystem
- state_engine: boolean
- current_speed: int
- cruise_state: boolean
- fuel_reading: int
- gear_pos: int
- trip_start_time: int
- trip_state: boolean
- time_counter_sec: int
- time_counter_trip: int
- speed_counter_trip: int
- average_speed: int
- rpm_total: int
- distance: float
- drive_shaft_rotation: int
- Calibrator_state: boolean
- cruise_value: int
- accelerate: boolean
- rpm: double
- cc: CruiseController
- monitor: Monitor
- dss: DriveShaftSensor
- grs: Gear_sensor
- cal: Calibrator
- fs: Fuel_Sensor
- timer: Timer
- cruise_controll_values: double Array
- gui: View_AMS
+ PerformAccelearion(boolean state)
+ add_rpm_number()
+ decrease_fuel()
+ drop_speed_automatic()
+ generates_drive_shaft()
-- In addition to the default constructor, setters and getters
<<Class>>
Gear_sensor
- ams: AutoMobileManagementSystem
-- In addition to the default Constructor, setters and getters
<<Class>>
Thread
+ run(void): void
<<extend>>
<<extend>>
<<extend>>
<<Class>>
Calibrator
- ams: AutoMobileManagementSystem
- distance: float
- dss: DriveShaftSensor
- speed: int
-- In addition to the default constructor, setters and getters
<<Class>>
Timer
- ams: AutoMobileManagementSystem
-- In addition to default Constructor, setters and getters
<<Class>>
DriveShaftSensor
- RPM: int
- ams: AutoMobileManagementSystem
- pulse: int
- revolution: int
-- In addition to the default constructor, setters and getters
<<Class>>
Monitor
- avg_speed: float
- fuel_consumption: float
- maintenance: float
- miles_driven: int
- speed: Calibrator
- trip: int
-- In addition to default Constructor, setters and getters
<<Class>>
CruiseController
- acceleration: CruiseControllerAcceleration
- ams: AutoMobileManagementSystem
- cruise_con_value: int
- cruise_res_value: int
- init_value: int
- inside: boolean
- state: CruiseContr ollerState
+ Activate_Cruise_control(View_AMS)
+ Deactivate_Cruise_control(View_AMS)
+ Res_Cruise_control(View_AMS)
-- In addition to the default constructor, setters and getters
<<Class>>
Fuel_Sensor
- ams: AutoMobileManagementSystem
-- In addition to default Constructor, setters and getters
<<Class>>
SystemStateTracer
- air_filter_counter: int
- air_filter_flag: boolean
- cbl: Calibration
- engine_state: EngineState
- fuel: int
- is_maintained: ServiceCompletion
- oil_filter_counter: int
- oil_filter_flag: boolean
- throttle_lvl: int
- transmission_state: TransmissionState
-- In addition to default Constructor, setters and getters
1

Model package[10]

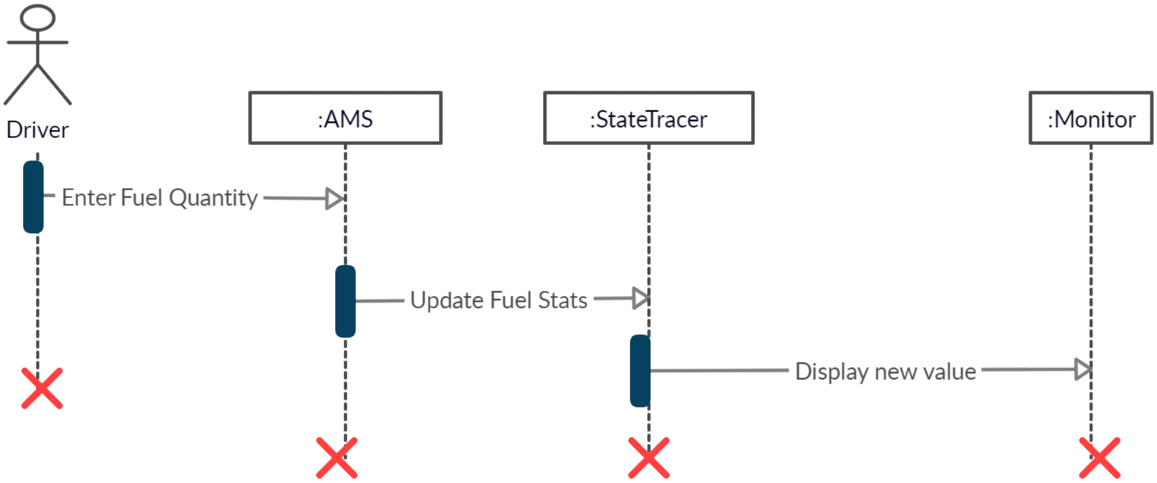

model
<<Class>>
AutoMobileManagementSystem
..
..
1
1
events
esper
<<use>>
<<Class>>
Config
- engine: EPServiceProvider
+ createStatement(String): EPStatement
+ registerEvents(Object): void
+ sendEvent(Object): void
<<Class>>
Main
- ams: AutoMobileManagementSystem

Esper package[11]

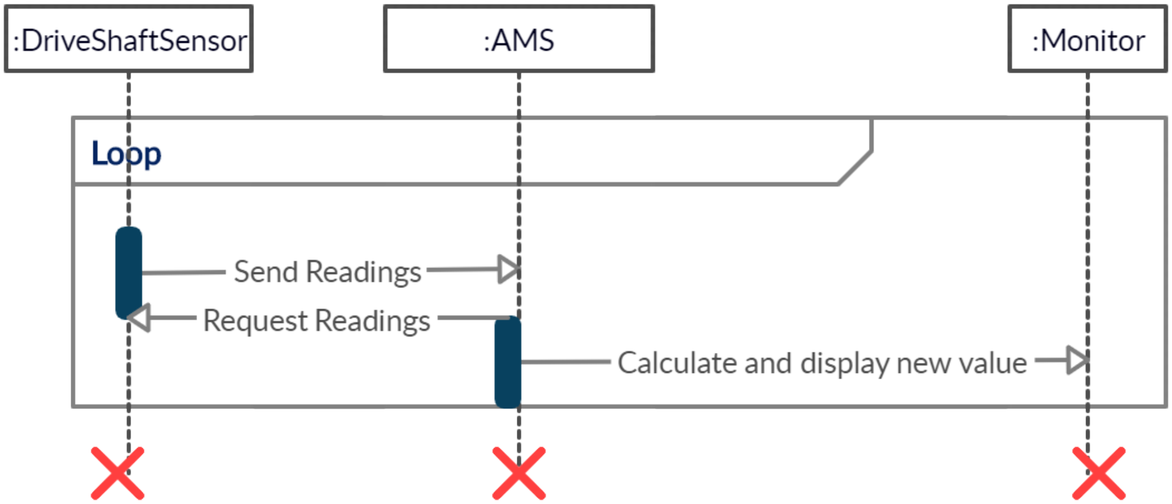
Sequence Diagrams -



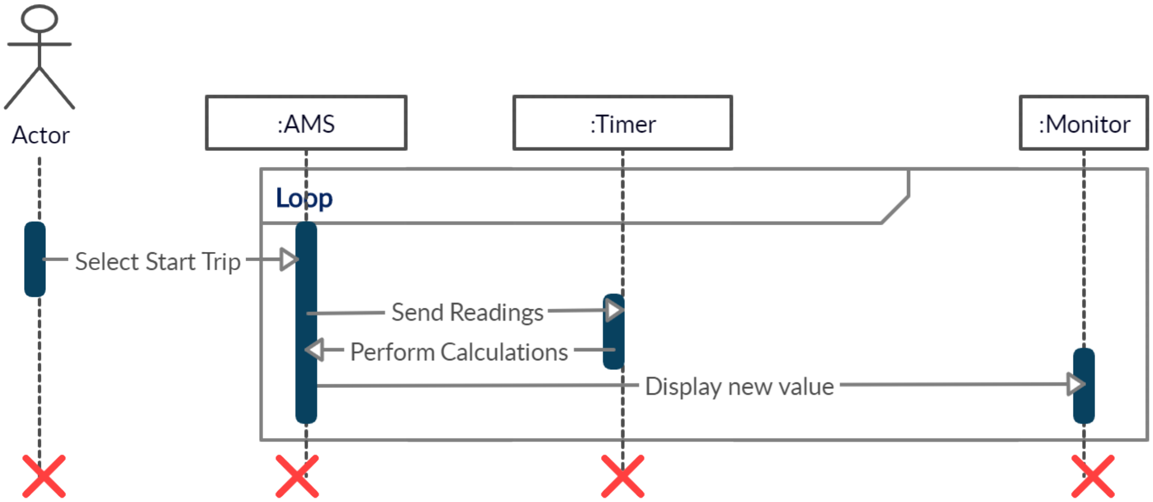
Activate cruise controller[12]



Update fuel quantity[13]



Mileage calculation[14]



Trip Average Speed[15]

Response Identification -

NUM	**Event**	**Repond**	**Duration [ms]**
1	*AccelerateEvent*	Reads acceleration current state	1
2	*Accelerate_cruise*	Reads current speed value and responds with setting the cruising to the current speed value.	900
3	*BrakeEvent*	Reads acceleration and sets it with false, and performs acceleration adversely	1
4	*CalibratorSensorReading*	Set distance and speed	1000
5	*Cruise_con_Reading*	Reads current cruising speed, and responds with matching the current speed to cruise speed.	1
6	*Cruise_state_reading*	Reads current cruise state and responds with setting the cruising bulb on/off and cruise value.	1
7	*Drive_shaft_reading*	Reads drive shaft rotation & displays on "View GUI"	250
8	*EngineEvent*	Start/Stop Engine & Light bulb.	1
9	*Fuel_Reading*	Reads the fuel and checks if no fuel, engine will not start, else it will start.	250
10	*Gear_Reading*	Reads current gear and responds with four different labels "Low/Mid/Top/Clutch" and turns gear bulb on.	250
11	*Maintainace_done*	Reads current maintenance condition and responds with a change to maintenance stages "	1
12	*PerformEvent*	Reads current engine state, and maps speed to RPM according to acceleration.	1
13	*RpmEvent*	Reads current RPM, and displays it on GUI.	1000

14	*SpeedOMeterReading*	Reads current speed, and responds with dropping the speed if cruising isn’t enabled, and prints Fuel/engine_state/GearPosition/Speed/ CruiseState and Value/drive shaft rotation.	1000
15	*TripEvent*	Reads current trip state, and responds with setting the trip bulb on & Time/Speed counters and sets the new state.	1
16	*Set_Fuel*	Reads current fuel, and responds with displaying the fuel on the gauge.	1
17	*Actuator_Reading*	Reads the current actuator state, and responds with three states either uphill, downhill, or normal road condition	1

Threads used -

No.	**Thread**	**Repond**	**Description**
1	*“Cruise Controller”*	cc	Cruise controller updates cruise control states “Activate/Deactivate/Resume” every second
2	*“Clibrator”*	ca	Calibrator measures speed and distance every second.
3	*“Fuel sensor”*	fs	The fuel sensor sets average fuel, decreases the fuel every 250ms.
4	*“Gear sensor”*	grs	Every 250ms it updates the transmission gear.
5	*“Monitor”*	mon	Monitor updates/notifies GUI every second.
6	*“Drive Shaft sensor”*	dss	Measure distance and speed every second.
7	*“Timer”*	Timer	Timer sets average speed and distance driven every second.