Version 1.4

()ALSET - Best Team

Dylan Edelman, Christopher Horn, Isabel Hughes, Steven Truong, Michael Mancino
24 March 2022
Team 2 CS 347-B

Table Of Contents

Section 1: Introduction

IoT (Internet of Things) is essentially a collection of sensors, mini processors and physical objects embedded with software/technologies that allow for the exchange and processing of information between other devices, systems and the internet. Consider that the majority of vehicle accidents are human error. Consider how often people go to mechanics to diagnose their vehicles only to leave still confused. Now imagine a system that could drive you from place to place in a safe and fast manner. A system that communicates with other vehicles to orchestrate lane changes and turns while also being able to see upcoming road conditions while having the capability to self-diagnose and attempt repairs using information straight from the manufacturer. All of this is possible when using IoT edge devices to capture information and IoT engines to process said information. As this is a mission critical real-time embedded system, availability, reliability and efficiency are of great importance to us.

As a self-organized team, one of our biggest goals in this project is to work efficiently and effectively towards a successful project completion in the time provided to us. Furthermore, we are committed to high quality software development, in which we are determined to create our software with an emphasis on security and debugging. Following the Agile Process Model, we first begin by understanding the problem and gathering all necessary requirements to communicate our collective vision for the project. After which, we will plan out our major features and devise their capabilities within the system, estimating the amount of work needed for integration. Next, we will model out concept designs and analyze real-world advantages and disadvantages to our proposed sensory features. Executing on our plan, we will test if our solution is one that effectively answers the problem of creating a safe and inexpensive vehicle by utilizing frequent and constructive customer feedback. If it is not a valid solution, we will return back to the drawing board to introspectively reflect on our previous work and reconstruct the software so it is better suited for our next iterative design. We will continue to repeat these steps until we, as a team, are unanimously confident with our project and that our primary goal, satisfaction from the customer, is reached as well. With a timeframe of February 1st to May 1st and our very different sets of qualifications, we can effectively move through the problem solving process, with our individual skills helping in each step.

- Isabel - People-Oriented, Diligent Worker, Coding
- Michael - Efficient, Coding, Goal-Oriented
- Steven - Analysis, Problem Solving, Reserved
- Christopher - Coding, Project Management, Task-Oriented
- Dylan - Communication, Detail-Oriented, Writing Skill

Altogether, we are oriented towards finishing our own individual tasks and reaching our goals in a manner where every detail is examined for flaws. We also are able to communicate effectively with one another and efficiently complete every aspect of our project. Isabel, Michael, and Christopher excel in programming and can ensure all software is working correctly in a technical mindset. Dylan is able to ensure our documentation is clear and communicates all aspects of the software that we have detailed. Lastly, Steven joins the entire team together and keeps us on track, while also providing solutions for parts of the project we may find challenging.

Features:

1. **Flexible Map routing**
    - While we are all well known of GPS apps, such as Google Maps or Waze, there are several downsides associated with them. First off is the distraction of the phone itself. By integrating the GPS routing into the vehicle, it allows for distraction less driving and keeps the driver's eyes on the road {HUD #3}. Secondly, GPS apps strive for the fastest arrival, which navigates drivers

through neighborhoods and side streets which are not meant to accommodate mass amounts of traffic. Our flexible map routing will take into consideration the volume of traffic, construction, weather conditions, and other environmental factors that would affect time of arrival. To combat the congestion of neighborhoods and other small streets, our service will attempt to avoid all of those roads and still get the driver there in an optimal amount of time. Throughout the trip if any of these statuses change, the driver will be informed and re-routed with the best possible course of action.

2. **Lane mitigation**
    - Lane mitigation uses a detection system with forward and side facing cameras to continually monitor the lines of the lanes surrounding the vehicle. The vehicle usually produces visual and audible warnings with a display on the HUD and a distinct beeping noise to alert the driver that they have left the lane. In our vehicle, we are going to take lane departure warnings to a new level and implement lane keep and centering assistance. If the driver is ever to leave the lane, keep assistance will provide slight steering and/or braking to continue to keep the vehicle in its lane. Centering assistance will provide the same as keep assistance, but will also center the vehicle within the two lanes. As this safety feature can seem unnecessary to some, it will also be able to be disabled or re-enabled at any time.
3. **HUD**
    - Head-up display, or HUD, is software that projects the dashboard onto the windshield, just below the driver's line of sight. The features on the HUD include current speed, the speed limit of the road, navigation information, alerts from the vehicles running systems, and the state of the battery. The biggest reason to use a HUD is for safety. According to the National Highway and Traffic Safety Administration, in 2019, 3,142 people were killed due to distracted driving. By using the HUD, the driver can receive notifications from your phone by Bluetooth, and have navigation directions and speed directly in front of them. This allows for the overall and continual safety of the driver and those on the road with them.
4. **Virtual side mirrors**
    - Virtual side mirrors, while a newer technology, can extensively improve the safety of a vehicle. By using cameras on either side of the vehicle, the virtual side mirrors are projected on either side of the vehicle, and on the HUD if desired. These mirrors can also help in guidance with turning and adjusting the vehicle in a lane. Due to heated and water-resistant lenses, the e-mirrors will never freeze or be misty, with perfect side views at all times. The last feature of these mirrors is that they help in blind spot detection, by allowing the driver to move the camera's view around. This will also allow for a green overlay for safe

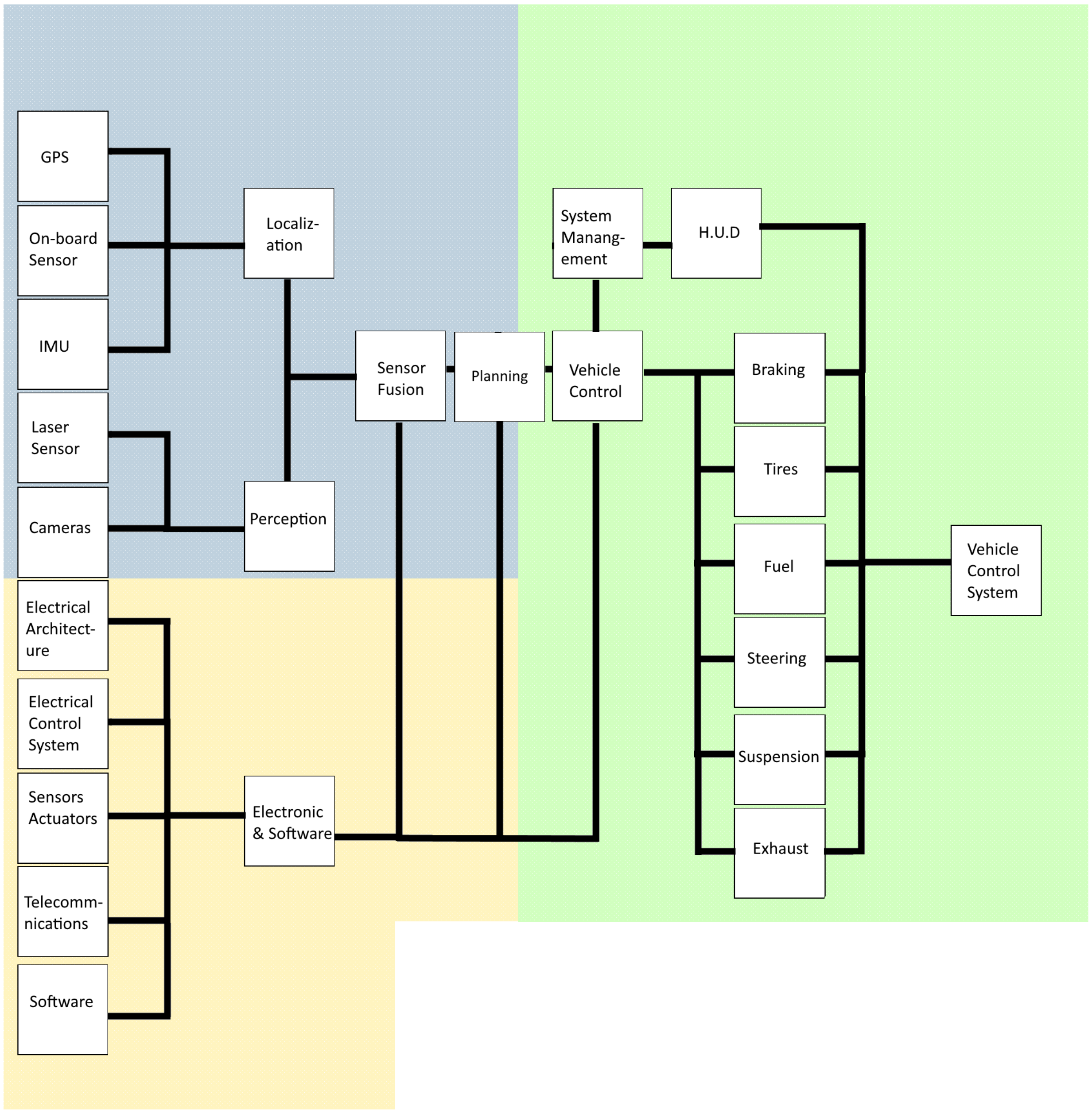
lane switching, and a red overlay for unsafe lane switching by using the AI sensors around the vehicle.

5. **Solar panels**
    - One of the bigger complaints about electric vehicles is the concept of phantom drainage. Phantom drain is the drain of energy which naturally occurs in batteries due to lack of use or driving in the cold. To combat this issue, solar cells will be placed into the roof of the vehicle to charge the battery when the vehicle is not in use. Another feature additional to the solar panels is a newer, Zinc-ion battery for the vehicle. Zinc-Ion batteries are lighter, more affordable, and more green than normal lithium ion batteries. They also increase safety as Zinc does not cause a heat reaction, which would combat phantom drainage when driving in a cold environment.
6. **AI sensors**
    - What good is a self-driving vehicle if it can't actually drive itself? Using our detection sensors, the vehicle will automatically accelerate and stop based on the current situation. In conjunction with our previous features, we can determine when to make turns, switch lanes and come to stops. Using map information and sensors, we will be able to stop at intersections and in conjunction with recognition software to see any crossing pedestrians. Information will be sent to us so we can improve our driving and mapping software.
7. **Auto parking/auto summon**
    - Even with parking cameras, drivers may still find it difficult to park in certain positions. Taking it to the next level, we will simply tell the vehicle to park itself instead of relying on the driver. Using sensors on the vehicle, we can determine valid parking positions and attempt to self park using a virtual representation of the current situation to calculate the actions and movements needed to park. Using an app or a feature implemented in the keys, we will allow the driver to automatically summon the vehicle. Simply using your phone's GPS coordinates, the vehicle will drive there safely and wait for the driver to unlock and enter the vehicle. Of course the vehicle will stay on the road and not just drive into a random building.
8. **Automatic emergency contact**
    - Should the vehicle get in a serious crash, the vehicle will automatically contact emergency services and send information such as coordinates, where the crash happened such as right & left side and estimation of the force of impact. Any people designated as emergency contacts will also be informed. To prevent false positives, if the driver fails to respond, emergency services will be contacted. Logs will also be sent to us to determine what exactly happened and how we can use this information to update our software.

9. **Self-diagnosis/repair**
    ○ The ability to self-diagnose problems and possibly attempt repairs is very handy for all drivers as not everyone will be savvy enough to understand the inner-workings of a vehicle. Unlike other vehicles with on-board diagnostics, we wish to inform the driver of the problem, possible solutions and urgency of said problem without a special reader. Though regulation may require us to have OBDII for now, we will still implement our own diagnostic system in conjunction with OBDII to collect vehicle metrics and determine abnormalities. If the vehicle is parked and there is a possibility of self-repair, the vehicle will attempt to do so when given permission. If self-repair is not feasible, the driver will be informed of the problem and be asked to take the vehicle to the nearest mechanic/dealer. Logs will be sent to use so we can determine error rates and so we can take steps to improve our systems.
10. **Internet integration**
    a. As this vehicle will use IOT, internet integration is extremely important in order to facilitate our previously mentioned features and keep our software updated. Using an embedded chipset and antenna allows for the vehicle to communicate with us to receive updates, diagnosis information, road/weather information and allow the driver to connect the internet for services like Spotify.

Section 2: Functional Architecture

Diagram:



2.1 Diagram Explanation

- When the vehicle is managing itself, whether in an auto-summon, lane mitigation, or driving autonomously, the blue side of the graph (sensor fusion) it activated to pick up the precise location of the vehicle and what it is doing, where the yellow side of the graph (electronics and software) provides the means to keep the vehicle in its action. For example, in auto-summoning, the sensor fusion displays the vehicles location in actuality to it's person and relays that to the software which will navigate the vehicle in the right direction, using sensors to avoid any obstacles. Together, sensor fusion and the software allow for the access of system management and vehicle control by the driver.

2.2 Localization

- GPS
    - Built specifically for Hug the Lanes, puts a greater weight on taking routes through high density roadways than through neighborhoods and backstreets to minimize traffic in those areas. Allows for greater flexibility in map routing.
- On-board Sensor
    - Detects passengers onboard and sends information to make sure the vehicle never makes any maneuvers that might cause harm to any of them.
- Inertial measurement unit
    - Perceives the rotational momentum and force of the vehicle and can detect when rates increase to dangerous levels as well as prevent them.

2.3 Perception

- Laser Sensors
    - Uses lasers to create virtual representation of space around the vehicle. Used in self-driving, auto-parking, and summoning.
- Cameras
    - Rearview, Front View, Blind spot detection, and Virtual side Mirrors.

2.4 Electronics and Software

- Electrical Architecture
    - Convergence of electrical hardware such as solar panels and dashboard touchscreen with software that ties everything together.
- Electrical Control System
    - The control unit of the vehicle. It’s responsible for all the other electrical components and lets them run as intended.
- Sensors/Actuators
    - The main reason the vehicle’s software has visualization capabilities.
- Telecommunication
    - Telecommunications and Internet Connections within the vehicle allows for the vehicle to receive software updates, also allowing the vehicle to self-diagnose issues and attempt to repair or contact for repairs. Lastly, telecommunications will alert emergency contacts and road assistance if an accident or emergency occurs with the vehicle. Utilizes network and cloud connection.
- Software
    - Handles driver input and sends it to vehicle control.

2.5 Sensor Fusion

- Combination of perception and localization in order to give the vehicle an accurate understanding of its surroundings.

2.6 System Management

- Collects information & data logs of systems inside the vehicle and sends them to Alset.

2.7 Vehicle Control

- Combines Sensor Fusion, Perception, Electronics & Software and System Management to decide the best course of action in different parts of the vehicle such as:
    - Braking and Steering - Allows the vehicle to stop or turn if it is needed, such as for sensing a vehicle in front of it going slower than it's current speed or that the road ends ahead and the vehicle must change its current trajectory.
    - Exhaust, Suspension, and Tires - Allows the vehicle to be able to tell if anything is going wrong with these systems, and will warn the driver or stop the vehicle if help is required.
    - Fuel - Allows the vehicle to check the amount of charge left in the battery and inform the driver if it is low and needs to be recharged.

2.8 HUD

- Uses Vehicle control and System Management to give driver information about the drive, such as road conditions and the state of the vehicle's battery.
- Can also display side view cameras to allow the driver to see blindspots.

2.9 Vehicle Driving System

- The combination of all of the features we listed, the end goal of a fully self-driving vehicle.
- Flexible Map Routing, Lane mitigation, HUD, Virtual Side Mirrors, Solar Panels, AISensors, Auto Parking/Auto Summon, Automatic Emergency Contact, Self-Diagnosis/Repair, Internet Integration, Self-driving capability.

Section 3: Requirements

3.1 Functional Requirements

3.1.1 Engaging the Cruise Control

Pre-conditions

- The vehicle must not be in park, neutral, or reverse
- The driver cannot have their foot on the brake pedal
- The vehicle's current speed is not over 90 mph or under 30 mph
- Self-Diagnostics detect no fault in the cruise control systems
    - Blown Fuse
    - Faulty Brake Pedal/Speed Sensor/Control Switch
- The set speed must be within 15 mph of the vehicle's current speed
- The speed sensor must confirm the current speed of the vehicle

Post-conditions

- The vehicle continues to cruise at the inputted speed, until compression from the brakes is received.

3.1.1.1

- The driver engages cruise control, the button sends the inputted speed to Sensor Fusion

3.1.1.2

- Sensor Fusion sends speed request (1) to IOT HTL for cruise control.

3.1.1.3

- IOT HTL continually sends speed requests to the Vehicle Control System.

3.1.1.4

- IOT HTL sends a request to turn on the cruise control display on the HUD.

3.1.1.5

- IOT HTL logs the event in System Management and the vehicle continues to cruise at the speed given in the request.

3.1.2 Disengaging the Cruise Control

Pre-conditions

- Vehicle is driving with Cruise Control engaged. Ensured by the usage light on HUD. IOT is sending continual speed requests to the Vehicle Control System.

Post-conditions

- The cruise control disengages and the light turns off. The driver regains speed control of the vehicle.

3.1.2.1

- The driver compresses the brake, vehicle control sends a 0 signal to IOT HTL for sensor fusion to disengage the cruise control.

3.1.2.2

- Sensor fusion sends back a signal to vehicle control to turn off the cruise control display from the HUD.
- Sensor fusion stops sending signals to navigate the vehicle at the set speed *s*, allowing the driver to regain control of speed.

3.1.3 Flexible Routing

Pre-conditions

- Engine must be running but the vehicle must be stopped to ensure undistracted driving.
- Must be in a location where internet and location services are available.

Post-conditions

- Informs the driver that they have arrived at the destination.

3.1.3.1

- Driver uses touch screen or app.

3.1.3.2

- The IOT engine communicates with our servers to determine the optimal route.
- Planning constantly references GPS and map data to dynamically optimize routes depending on road conditions.

3.1.3.3

- When the GPS matches the selected location, it displays to the driver on the HUD or app that they have arrived at their destination.

3.1.3.4

- IOT HTL logs the event in System Management.

3.1.4 Lane Mitigation

Pre-conditions

- Vehicle must be driving on a road with visible lane markings.
- Turn signals must be disengaged.

Post-conditions

- If the steering wheel is turned, the system disengages to avoid over-alerting or or over-turning the steering wheel, causing the vehicle to cross over lane markings.

3.1.4.1

- Cameras send an alert to Perception that lane markings are in sight.

3.1.4.2

- Perception sends a request to sensor fusion to alert the driver with a sound.
- Sensor Fusion passes information to Planning.

3.1.4.3

- If continuous alerts of the lane markings are too close to the vehicle, Planning directs the driver's vehicle control and shifts the vehicle back into the lane.

3.1.4.4

- IOT HTL logs the event in System Management.

3.1.5 HUD

Pre-conditions

- The vehicle is on.

Post-conditions

- The HUD is turned, allowing the driver to see the information provided by it.

3.1.5.1

- HUD communicates with Self-Diagnostics to alert the driver about any issues and display options regarding self-repair.
- Display to driver a message that that vehicle should be taken to the repair center if the self-repair attempt failed or no self-repair options are available.

3.1.5.2

- Communicate with systems such as VCS and Electronics/Software to relay information to the driver. Includes information such as speed, battery charge, and more.

3.1.6 Virtual Side Mirrors

Pre-conditions

- The vehicle is on.

Post-conditions

- The virtual side mirrors are accessible by the driver, and are able to be used as a blind spot detector.

3.1.6.1

- When the vehicle is on, cameras activate and send their feed to Sensor Fusion.
- Cameras continually send their live feed to Sensor Fusion, so the driver is never put into an unsafe position.

3.1.6.2

- Sensor Fusion sends the live camera feed to Vehicle Control through Planning.

3.1.6.3

- Vehicle control sends the feed to System Management, where it can be viewed directly on either side of the HUD, and can be moved with the side mirror joystick.

3.1.7 Solar Panels

Pre-conditions

- Vehicle is turned off.
- Self-Diagnostics detect no fault in the Solar Panel System.
- The driver has not specified that they do not want the solar panels to activate.
- Solar panels are in sunlight ranges.
- The vehicle is not currently charging via its charging port.

Post-conditions

- The vehicle gains *X*% of charge from the solar panels.

3.1.7.1

- When pre-conditions are met, the vehicle communicates with the IOT engine to determine weather conditions and determine if solar panels would be effective.
- Informs driver about current weather conditions if they are bad and confirms with the driver if the solar panels should be exposed via the hud.

3.1.7.2

- Planning tells electronic and software modules to expose the solar panels to the sky and collect energy.

3.1.7.3

- IOT HTL logs the event in System Management.

3.1.8 AI Sensor

Pre-conditions

- Self-Diagnostics detect no fault in any of the sensors.

Post-conditions

- Relays sensor information to necessary modules.

3.1.8.1

- While the vehicle is on, sensor data is recorded and sent to the Sensor Fusion module.

3.1.8.2

- Sensor Fusion will collect sensor data and the localization and perception modules will pass this information to the Planning Module.

3.1.8.3

- IOT HTL logs the event in System Management.

- This process repeats itself while the vehicle stays in use to ensure the AI sensors are always on unless they are functioning improperly.

3.1.9 Auto Parking/Summon

Pre-conditions

- Driver has enabled auto summon via app or keys or has turned on auto parking.
- For auto summoning, the vehicle must have the ability to find the location of the driver using our app or keys.
- For auto parking/summoning, the vehicle must be at a stop.
- Driver is inside the vehicle and does not have a foot on the accelerator pedal.
- Self-Diagnostics detect no fault in the auto park/summon systems or sensors.

Post-conditions.

- Vehicle has come to a complete stop at the designated location.

3.1.9.1

- Auto park request is sent to Planning.
- Keys or App will send auto summon requests to Planning.

3.1.9.2

- Planning will take the auto park request and use information from sensor data alongside the IOT engine to determine valid parking spots.
- Planning will take the auto summon request and use information from sensor data alongside the IOT engine to determine the driver's current position or designated summon location.
- Vehicle speed will not exceed 5 mph.

3.1.9.3

- VCS will manipulate the vehicle's speed, wheel angle and more to drive the vehicle into the valid parking spot or driver’s summon location.

3.1.9.4

- Planning and VCS will tell the vehicle to come to a stop and relay to the driver that the auto park/summon has been completed in either the HUD or app.

3.1.9.5

- IOT HTL logs the event in System Management.

3.1.A Self-diagnosis/Repair

Pre-conditions

- None.
- This feature will run in the background regardless of whether the driver is driving the vehicle or not.
- Feature will run unless the driver decides to turn the vehicle completely off.

Post-conditions

- Vehicle will either attempt to do vehicle repair and/or inform the driver of the diagnosis and to bring the vehicle to a repair shop. Create a log for repair attempt and diagnosis in System Management.

3.1.A.1

- Reads logs from System Management and will check any abnormalities or errors in various vehicle systems.
- Reports and reads data from modules if no log is available to read.
- If at any point, a system reports an error, Self-Diagnosis will immediately begin to assess the log and the error.
- If logs or data don't have any errors, repeat the process of checking other systems.

3.1.A.2

- If logs or data do have errors, determine the corresponding error codes and communicate with our servers for possible repair solutions using Electronic/Software.
- Informs driver on the HUD. Gives them the error code, brief description and depending on availability, will ask if the vehicle can perform self repair.

3.1.A.3

- If repair was selected, it will inform the user to drive the car to a stop at a safe location and once stopped. Then Planning will perform the necessary repair steps using information from our servers once the car has come to a stop. If this fails, inform the driver and suggest taking the vehicle to a repair center.
- If repair was selected or is unavailable, inform the driver to take the vehicle to a repair center.

3.1.A.4

- Keeps the error message on the HUD until the problem is fixed.
- IOT HTL logs the event in System Management.

3.1.B Vehicle Drive System

Pre-conditions

- Self-Diagnostics detect no faults in all other systems and modules
- Driver has enabled self driving via a button or HUD.
- Driver has selected a destination.

Post-conditions

- Vehicle navigates itself from point A to point B in a safe and efficient manner.

3.1.B.1

- Grabs location requested from driver.

- Repeats location requests constantly in order to make sure the drive system is accurate.

3.1.B.2

- Communicates with Flexible Map Routing module to determine path.
- Gets traffic data from various possible routes to the destination and calculates which path results in the least amount of overall trip time, as well as whether or not the route contains tolls.
- Sends all the information and information from Flexible Map Routing to Planning.

3.1.B.3

- Communicates with Planning to drive the vehicle.
- Utilizes other functional requirements and processes previously integrated in order to effectively drive the vehicle to its destination in a safe and timely manner.

3.1.B.4

- Communicates with the Auto Park module to park the vehicle.
- Once securely parked, Flexible Map Routing, Line Mitigation, and other functionally used software is told to turn off in order to preserve battery life by limiting the running programs to only the necessary ones.

3.1.B.5

- IOT HTL logs the event in System Management.

3.2 Non-Functional Requirements

3.2.1 Performance

3.2.1.1

- All response and execution times will not take longer than 10 ns.

3.2.1.2

- IOT HTL will be able to process a thousand requests per second.

3.2.2 Reliability

3.2.2.1

- CC does not fail more than once per 300,000 hours during operation.

3.2.2.2

- Processors will not fail more than 3 minutes per year.

3.2.2.3

- Each System will not fail more than once per year.

3.2.2.4

- Total System Failure will be impossible excluding crashes or unforeseeable events.

3.2.3 Security

3.2.3.1

- Access to IOT HTL will be secured by driver ID passwords.

3.2.3.2

- Data sent through IOT HTL will be encrypted to protect against cyber attacks on the vehicle.

3.2.3.3

- App communicates with vehicle on an encrypted network. Outside drivers will not be able to eavesdrop or manipulate communication lines.

3.2.4 Software Update

3.2.4.1

- Software update by cloud services will make sure the vehicle is always up to date, so long as the connection is stable.

3.2.4.2

- Software updates will happen automatically as long as the vehicle is not being used.

3.2.4.3

- Software updates will be done with encryption. Update will have a verifiable signature to guarantee that the vehicle will not download a rogue update.

3.2.5 Network

3.2.5.1

- Connects to the IoT and allows the rest of the vehicle to function.

3.2.5.2

- Network must be reliable to facilitate software updates and vehicle features.

3.2.5.3

- Network must be fast to run our systems and ensure drivers can enjoy consistent access to the internet.

3.2.5.4

- Network must be encrypted.

3.2.6 User Interface

3.2.6.1

- Usability for the driver.
- Will not have too much abstraction so that the driver can access permitted information to assess and fix software and hardware issues should they arise.

3.2.6.2

- UI must be intuitive. Buttons must be labeled properly and sized & spaced for efficient usage.

3.2.6.3

- UI must be reactive and responsive. Needs to be clean, sleek and quick.

3.2.7 HUD/Display

3.2.7.1

- Used by the driver to see insights about the vehicle, but without means of distraction.

3.2.7.2

- Needs to be non-invasive. Needs to not distract or block the driver’s view of the road.

3.2.8 Hardware

3.2.8.1

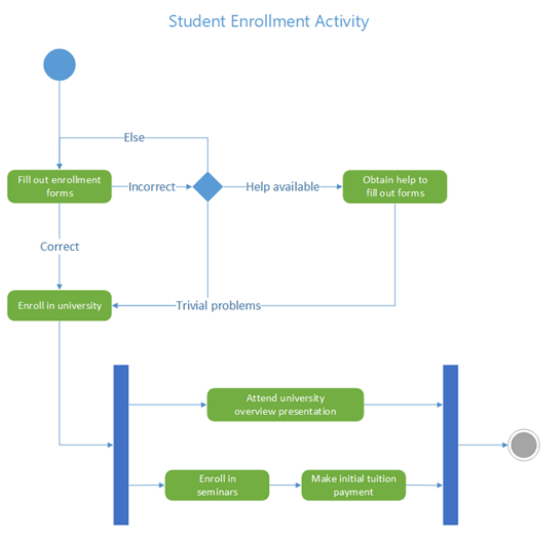
- Physical interface for technician access.

Section 4: Requirement Modeling

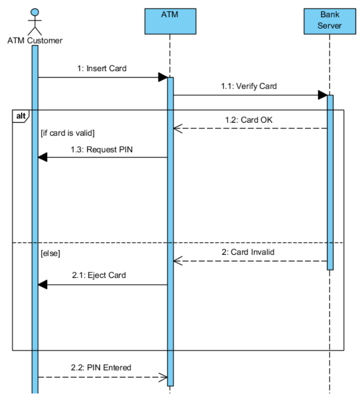
1.4.1: Use Case Scenarios: At least five features that are good representative of all functionalities of the system, so we understand the role of the Components (Sensors, Sensor Fusion, Planning, etc.), the data flows, the Actors, the Triggers, alternative paths, exceptions, etc. Do not repeat similar Use Cases. The purpose is requirement modeling. We don't needs models that are the same.

- Use Cases statements should be clear statements of the expected software behavior in proper time order. See Example from class discussion.
-

1.4.2: Activity Diagrams: should be based on Use Cases, shows the role of Actors (Objects) in hierarchical order, decision points, etc. Include alternative paths and Exceptions.

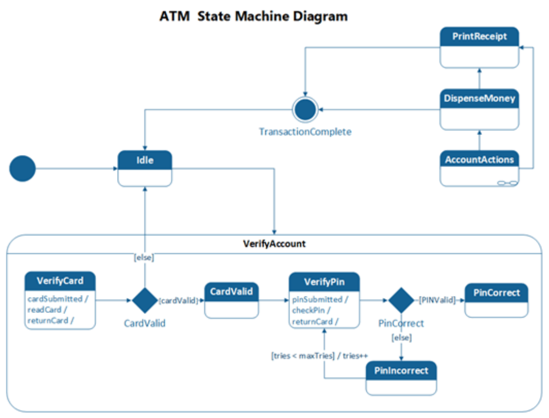

Student Enrollment Activity
Else
Fill out enrollment forms
Incorrect
Help available
Obtain help to fill out forms
Correct
Enroll in university
Trivial problems
Attend university overview presentation
Enroll in seminars
Make initial tuition payment

1.4.3: Sequence Diagrams: There should be one Sequence Diagram per Use Case.



1.4.4: Classes: Classes must be defined with meaningful attributes and methods that you observed in the expected software behavior (e.g. Use Cases)

1.4.5: State Diagrams: Nodes must be the states that the software creates, not processes (such as Normal State, Orange State, Red State of the software/car). The arrowed lines must be the triggers that causes state changes.



4.1 Cruise Control

4.1.1 Use Case - User Sets Cruise Control

- Pre-Condition: Vehicle is on, moving, and break is not being compressed.
- Post-Condition: Vehicle moves at set cruise control speed. HUD Display shows that cruise control is in use.

1. The driver presses the Cruise Control button in the vehicle.
2. Inputted speed x is requested from Electrictronics and Software to Sensor Fusion.
3. The Internet of Things (IOT) receives the request from Sensor Fusion, which prompts it to send a request for x speed to the Vehicle Control System.
4. The Vehicle Control System turns on the Cruise Control display with the current speed for the driver to see.
5. IOT continually sends requests to drive at speed x until the speed is changed or cruise control is disengaged.
6. All events are logged in System Management as to if the system is functioning properly
    a. If not, an alert is sent to the driver through the HUD and cruise control will not activate until self-diagnosis/repair is done.
    b. If yes, the cruise control will remain on at speed x.
7. The vehicle cruises at speed x safely.

4.1.2 Activity Diagram

-

4.1.3 Sequence Diagram

-

4.1.4 Classes

-

4.1.5 State Diagrams

-

4.2 Auto Park/Summon

4.2.1 Use Case

- Pre-Condition:
- Post-Condition:

1.
2.
3.
4.
5.
6.
7.
8.

4.2.2 Activity Diagram

-

4.2.3 Sequence Diagram

-

4.2.4 Classes

-

4.2.5 State Diagrams

-

4.3 Self-Diagnosis/Repair

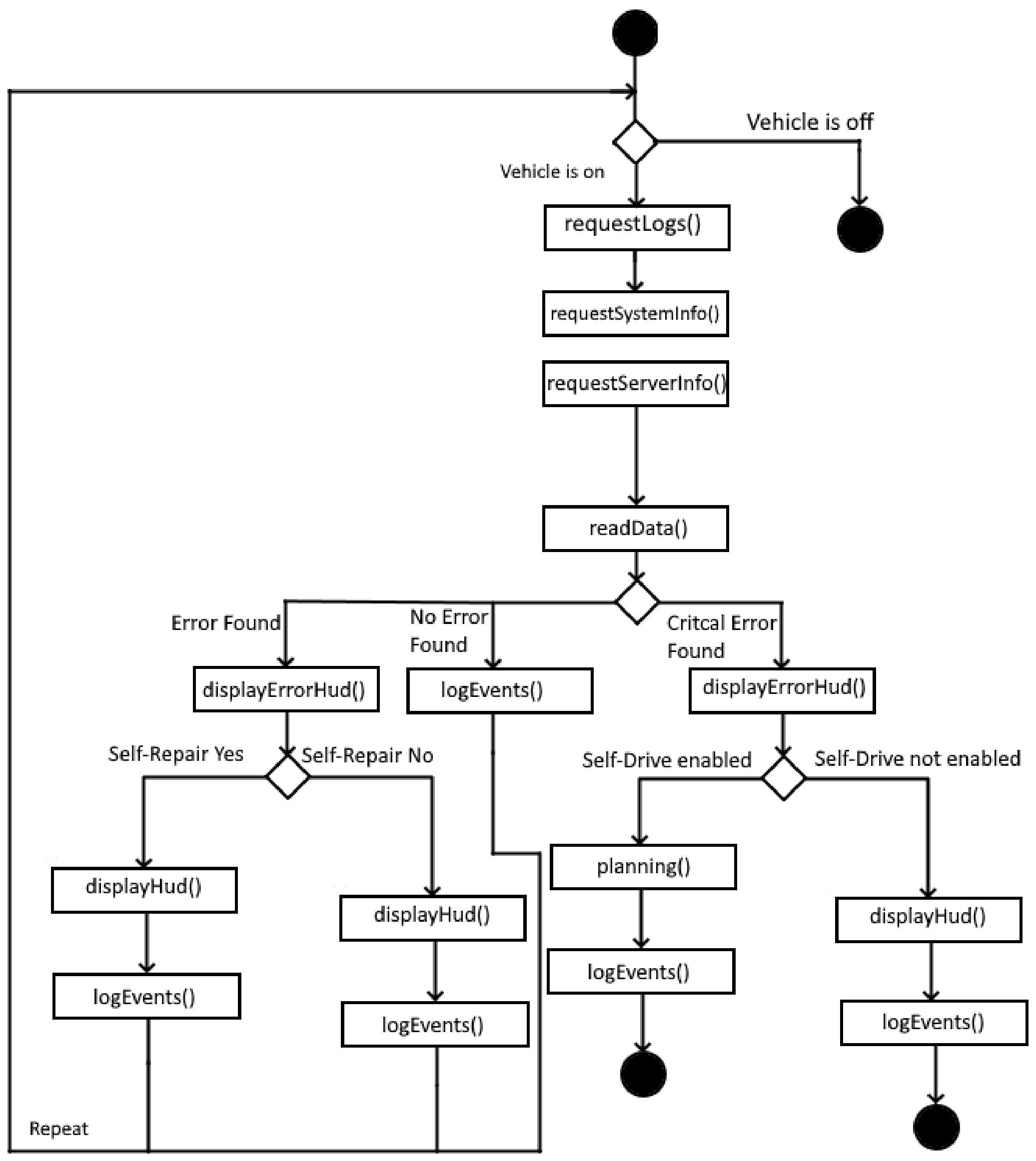
4.3.1 Use Case

- Pre-Condition: Vehicle is on.
- Post-Condition: Self-Diagnosis has read logs and data from other systems. Has attempted self-repair if the user selected yes. Display message to user on HUD.

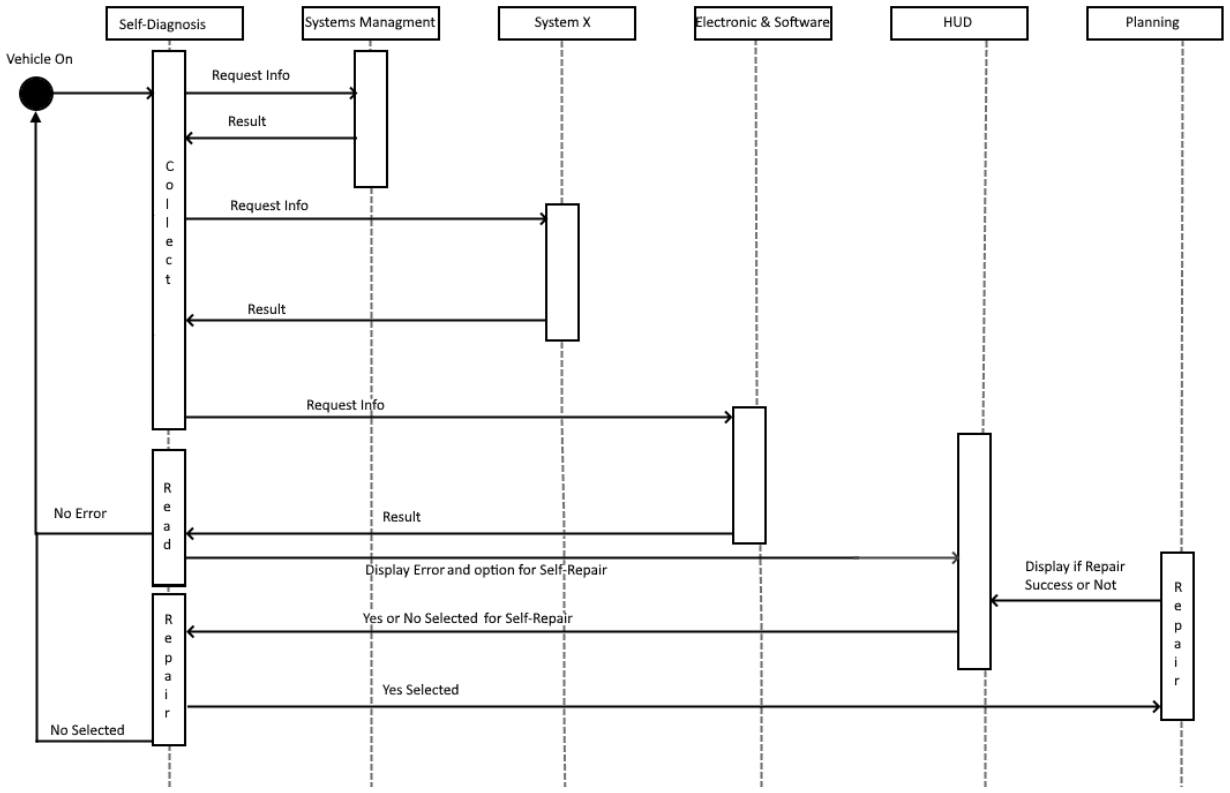
1. Self-Diagnosis requests logs from Systems Management for system x.
2. Self-Diagnosis requests information from system x.
3. Request technical information about system x from Electronics and Software.

4. Read the data or logs and detect any abnormalities in conjunction with server information.
5. If critical failure, send a request to HUD to display to the driver to stop the vehicle in a safe location as soon as possible.
    a. Display Error message and inform Driver to ask for assistance once vehicle has stopped.
    b. If the vehicle is self driving, send a request to HUD to display to driver an Error Message and attempt to stop in a safe location (Ex:shoulder of a highway).
    c. Log events
6. If an error is detected, send a request to HUD to display an Error Message that informs the Driver of a possible error and depending on server information, the vehicle may be able to self-repair. Display a choice of yes or no for the vehicle to self repair.
    a. If yes, send a request to planning and any necessary server information to Planning.
        i. Planning will attempt to follow server instructions and attempt self-repair. Send a request to HUD displaying information if repair was successful or not.
        ii. Repeat the process for every other system.
    b. If no, display an error onto the HUD. Display to driver that vehicle should be looked at by a mechanic.
    c. Log events and repeat the process for every other system.
7. If no error is detected, log events and repeat the process for every other system.

4.3.2 Activity Diagram



4.3.3 Sequence Diagram

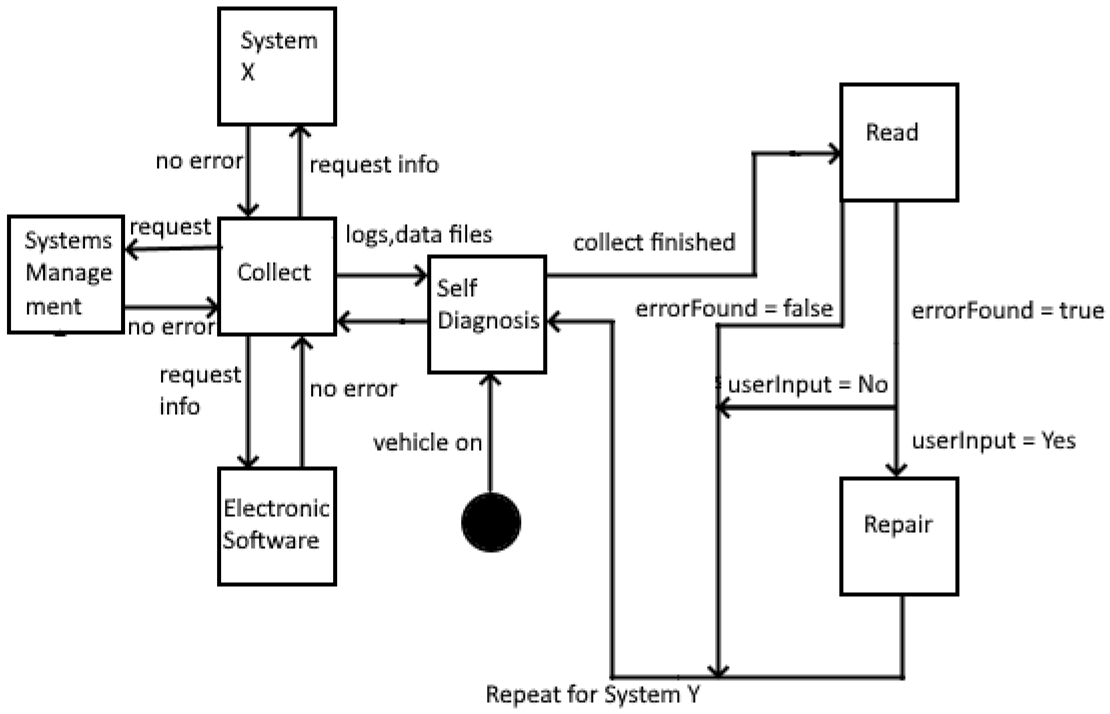


-

4.3.4 Classes

- Collect
  - Retrieves information on system X through logs from Systems Management and directly from system x.
  - Retrieves technical information from our servers about system X.
  - Proceed with Read unless an Error was thrown out somewhere.
- Read
  - Takes logs and information from System X and reads data.
  - Detects and flags if an error has been found.
  - Display error message and prompts User.
  - Proceed with Repair.
- Repair
  - Depending on user input, will send requests to Planning to continue with the repair request.

4.3.5 State Diagrams



4.4 Flexible Routing

4.4.1 Use Case

- Pre-Condition:
- Post-Condition:

1.
2.
3.
4.
5.
6.
7.
8.

4.4.2 Activity Diagram

4.4.3 Sequence Diagram

4.4.4 Classes

-

4.4.5 State Diagrams

4.5 AI Sensor

4.5.1 Use Case

- Pre-Condition: The car is running and Self-Diagnostics detect no fault in any of the sensors.
- Post-Condition: Retrieves information accumulated by sensors and relays them to the appropriate modules.

1. Sensor Fusion requests a self diagnosis of each sensor in both localization and perception.
2. Sensor Fusion requests data from localization sensors and perception sensors
3. All sensors record their information and send it back to the localization module or perception module which in turn send it to the Sensor Fusion
4. Sensor Fusion combines the information given into a single format to send off to the planner
5. Sensor Fusion sends event information to the Systems Management for it to be logged for later use.
6. The process is repeated until the car is no longer in use and is turned off.
7.
8.

4.5.2 Activity Diagram

4.5.3 Sequence Diagram

4.5.4 Classes

-

4.5.5 State Diagrams

WINE
Car should have wine in its fuel reserve
Wheels are kegs for maximum comfort

(filled with reza’s aging wine)