CSC-436 Project Milestone 3

Matt Fernandez
Matt Ljuljic
Greggory Antoine
Jonathan Lustman

State University of New York at Oswego, Oswego, NY 13126

May 8, 2020

Contents

1 Introduction

1.1 Purpose

ICPS is an intersection control system for an autonomous vehicle. The vehicle will assess the safety of the intersection at a given point and decide a proper moment to proceed through the intersection.

1.2 Scope

The primary goal of this project is to be further implementable on more sophisticated devices.

1. ICPS shall be responsible for one and only one driving behavior which is specified in subsection 1.1.
2. ICPS will be implemented on Anki Overdrive hardware.
3. The driving track or scenario for the Anki vehicle will resemble that of a figure 8. This configuration is accomplished by:
    - connecting three turning pieces from the top part of an intersection piece to the right part of the intersection piece.
    - connecting three turning pieces from the bottom part of an intersection piece to the left part of the intersection piece.
4. A second actor is running concurrently in a separate lane on the same the track, is running independently from the main actor. This secondary will traverse the track via the use of built in software.

1.3 Definitions, Acronyms, Abbreviations

Anki Overdrive - a system encompassing Anki vehicles, Anki tracks, dedicated software, and Bluetooth transitions to simulate vehicle race and other driving activities.
Anki Track - a plastic, interconnecting track which contains magnetic strips encoded from 0 to 255 in both horizontal and vertical orientation.
Anki Vehicle - a miniature machine, modeled to look and move similarly to a vehicle, which travels along an Anki track, transmits its location-related data, and follows a path on the track as dictated by its operative software.
Autonomous - a state in which an object, machine, or system performs some set actions without non-sensory user input.
ICPS - Intersection Collision Prevention System
Obstacle - a vehicle which is not under the control of ICPS.
Vehicle - a machine, generally operable by persons, which transports people and items across land.

1.4 References

https://anki.com/en-us/overdrive.html
https://anki.github.io/drive-sdk/docs/programming-guide
https://www.youtube.com/watch?v=flU82NKrK7s
https://www.synopsys.com/automotive/what-is-autonomous-car.html
https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety
https://www.nhtsa.gov/staticfiles/rulemaking/pdf/Automated_Vehicles_Policy.pdf

1.5 Overview

Many important characterises regarding the Anki architecture in regards to the problem scope are listed below. The main points focusing on how the Anki architecture works, more specifics for how ICPS will be designed via SRS, and..

2 Overall Description

2.1 Product Perspective

Before looking at the Anki architecture, the functions of real world autonomous vehicles should be looked at. Actual autonomous vehicles, automobiles in this specific case, use an amalgamation of radar, lidar (light detection system for distance), machine learning combined with cameras to identify objects and road signs around the car. These self driving automobiles use a combination of hard coded protocols and complex algorithms for tasks such as obstacle and collision avoidance.

This product is contained within the realm of the Anki architecture. As specified, it is modeling how a autonomous car will function in ways similar to how real world scenarios may play out in an intersection using the ICPS to simulate the behaviour of an autonomous vehicle in the real world. In contrast to autonomous vehicles which have specialized hardware and software for all the functions mentioned above, the Anki cars are small scale simplified representation of how an intersection scenario would play out. The Anki cars though, unlike real world autonomous vehicles lack the the multi layered complex systems which allow the cars to work on a real world automobile. Instead, as the Anki cars run on a small closed track, the cars have awareness off the their current location on the track and send it as packages to the computer system running ICPS, which allows the ICPS constantly retrieve the location of each of the Anki cars.

2.1.1 System Interfaces

System name: Automotive CPS
System mnemonic: N/A
Specification number: N/A
Version number: N/A

Source: Bastian Tenbergen (GitHub Repository)
ICPS is the main system that ours will communicate with. ICPS is pivotal to the construction of ours because this allows us to pull a location from the other cars location. With this location, ICPS will make a decision on whether or not to cross the intersection.

2.1.2 User Interfaces

ICPS will be started and run in a command line environment. The terminal environment will start, run, accept commands, and end the program.

2.1.3 Hardware Interfaces

Since the Anki vehicle communicates directly with it's own API and that API is routed through the aforementioned system we will have no direct interface with the hardware self. The system also does not have any explicit interaction with an interface.

2.1.4 Communications Interfaces

The car will communicate through Bluetooth 4.0LE(Low Energy) signals.

2.1.5 Memory Constraints

Since the Anki Overdrive cars have no physical memory, the memory will be limited by the computer running the program.

2.1.6 Operations

ICPS has very little in terms of user input. The main functions of a user in regards to ICPS are:

- The user starts ICPS
- The user will passively observe/experience ICPS
- ICPS shall run indefinitely until input from the user to stop ICPS

The user will not have any specific character or c, instead the assumption that all people will interact with ICPS in the same way due to how limited ICPS scope is. nor will there need to be any specific environment where ICPS must be implemented.

2.2 Product Functions

- ICPS will be able to assess, identify, and react to the current state of the field.
- ICPS will stop the vehicle at an intersection if it is unsafe to proceed.

- ICPS will keep the vehicle in its lane.
- ICPS will stop the vehicle if there is an unanticipated error.

2.3 Constraints

ICPS will have to operate with in fast manner to constantly assess the intersection safety. Command issuing should be no longer than 50 ms. ICPS will have operate within the limits of the physical vehicle itself. ICPS shouldn't damage the car while in operation. ICPS will not operate if the correct configuration (figure 8 configuration) is not present. ICPS requires the additional system interface above to operate. Without that system present the ICPS will not interface with Anki API directly.

2.4 Assumptions and Dependencies

In order for ICPS to function here are some assumptions that are made:

- The Anki track is correctly configured as specified above
- The Anki track is cleaned
- The Anki vehicle is properly charged, and configured
- Second Anki vehicle or obstacle, which is not a part of ICPS, is configured to run independently at a constant speed

2.5 Apportioning of Requirements

The current iteration of ICPS is very limited in terms of its overall scope and how that relates to the overarching problem space. here are a few areas in which ICPS will grow in further iterations.

- Characteristics of this ICPS will be translated to an actually autonomous vehicle
- The track will be able to be dynamically changed
- The actor will be able to operate in more conditions

2.6 Context Entities

This is where we lay out all the previously developed concepts that ICPS is dependant on.

ID	Entity	Facet
CE-1	Anki Overdrive System	IT
CE-2	Java Code Wrapper(Bastion's Code)	Development
CE-3	Autonomous vehicles of other companies	Subject
CE-4	The ICPS Deployment will be hosted locally	IT

3 Requirements

3.1 Goals

ID	Natural Language Requirement
G-1	Vehicle will run autonomously around the track.
G-2	ICPS must prevent vehicle collisions through the intersection.
G-3	ICPS will scan for obstacles in or approaching the intersection before it traverses through it.
G-4	ICPS must stop the vehicle if an obstacle is detected at the intersection.
G-5	ICPS will be able to safely continue operation when a reasonable amount of location queries are missed
G-6	ICPS must be able to stop the Vehicle in the case of an emergency time out

3.2 Responses

In the context of our system, metrics such as Modes, Stimuli, Functional Hierarchy, Objects, Feature, Database, and User Class did not describe the integrity of ICPS. This is where we are putting the bulk of Functional Solution Oriented Requirements.

3.2.1 Response 1: Start System

ID	Natural Language Requirement
R1-1	ICPS must provide the vehicle with the instruction to move along the track upon activation.
R2-2	ICPS will continue moving the vehicle until another response is triggered.

3.2.2 Response 2: Scan on New Section

ID	Natural Language Requirement
R2-1	ICPS will provide the vehicle with the instruction to keep moving around the track.
R2-2	ICPS will continue to operate its Stopping weight protocol.
R2-3	ICPS must be able to update both vehicles location every 150ms.
R2-4	ICPS must be able to keep a weighted moving average of location data for the Vehicle and Obstacle
R2-5	ICPS must be able stop the Vehicle if it cannot receive location data from tho obstacle or vehicle for three seconds
R2-6	ICPS shall calculate the position of the obstacle in relation to the vehicle.
R2-7	ICPS must provide instructions to the vehicle to avoid collision with the obstacle.
R2-8	ICPS shall react to the current state of the track as appropriate.
R2-9	ICPS must be able to track the position of the Vehicle in relation to the track.
R2-10	ICPS must be able to track the position of the Obstacle in relation to the track.
R2-11	ICPS must not be used on track configurations not configured into the system
R2-12	ICPS will prompt the user to select the required track configuration upon startup.

3.2.3 Response 3: Stop at Intersection

ID	Natural Language Requirement
R3-1	The ICPS will determine it unsafe to cross the intersection if it calculates the vehicle will cross the intersection concurrently with the obstacle within an error of three obstacle lengths of the obstacle and vehicle collision point.
R3-2	If the ICPS determines it unsafe to cross the intersection from the stopping weight protocol, the ICPS must provide the vehicle with the instruction to stop no closer than one fourth of the vehicle's length before the edge of the intersection.

3.2.4 Response 4: Scan Intersection

ID	Natural Language Requirement
R4-1	When the vehicle has stopped at the intersection, it will wait till the obstacle's location has been updated as passing through the intersection.
R4-2	The ICPS will provide the vehicle with the instruction to start moving once it determines it is safe from collisions.
R4-3	Once the vehicle has passed the intersection, ICPS will continue to provide the vehicle with the instruction to keep moving until another response is triggered.

3.2.5 Response 5: Stop Forever

ID	Natural Language Requirement
R5-1	Once the User decides to end the system, ICPS must stop the Vehicle from moving.
R5-2	Once the Vehicle is stopped, it will remain where it is on the track.
R5-3	Once the User decides to end the system, ICPS must stop the Obstacle from moving.
R5-4	Once the Obstacle is stopped, it will remain where it is on the track.

3.2.6 Response 6: Unforeseen Failure

ID	Natural Language Requirement
R6-1	If the vehicle and the obstacle have the same location at the same time, ICPS must order the vehicle to stop all movements.
R6-2	If ICPS cannot establish an initial connection to either the Obstacle or Vehicle, ICPS must terminate.

3.3 Quality Requirements

ID	Natural Language Requirement	Quality Property
Q-1	ICPS should instruct the vehicle to move as fast as it can go without leaving its lane.	Satisfaction

3.4 Design Constraints

ID	Natural Language Requirement
C-1	The ICPS must be run on a bluetooth enabled computer.
C-2	ICPS must provide the vehicle with instructions on how to operate.
C-3	ICPS will take no longer than 50ms to transmit a command to the vehicle.
C-4	ICPS must remain up to date as to the location of the vehicle and obstacle.

3.5 Hazard Mitigating Requirements

ID	Natural Language Requirement
M-1	ICPS must continue to query Obstacle location data through intersection
M-2	ICPS must bring Vehicle to emergency reverse if Obstacle location data overwhelms stop weight while vehicle crosses intersection
M-3	ICPS must keep Vehicle out of the Obstacle's lane at all times
M-4	If the Vehicle is in the same lane as the Obstacle, ICPS will move the Vehicle into another available lane
M-5	If Location data is not received from either Obstacle or Vehicle ICPS will start the Emergency Timer
M-6	ICPS will use stored data until a successful query returns location data or time out occurs
M-7	If Location data is not received from eater Obstacle or Vehicle while Emergency Timer Is Activated ICPS will continue the Emergency Timer
M-8	If Emergency Timer reaches 3 seconds ICPS will stop Vehicle and terminate
M-9	ICPS has an emergency function will check that the car is not stopping in the intersection
M-10	ICPS must not allow the amount of timeouts to exceed the predefined limit
M-11	If the amount of timeouts exceeds limit, ICPS will stop the Vehicle

3.6 Hazard Functional Analyses

System: ICPS Subsystem: Critical Functions			Functional Hazard Analysis				Analyst: CSC-436 Dream Team Date: 05/01/20		
Function	Hazard Index	Hazard	Effect	Cause	IMRI	Recommended Action	FMRI	Comments	Status
Vehicle stops at intersection	FuHA-1	Vehicle collides with Obstacle	Vehicle crash with potential harm to vehicle or users	Vehicle continues moving	1C	Vechicle slowly exits intersection then stops	1F	This is a temporary solution, long term solution still required	Open
Vehicle stays in lane	FuHA-2	Collision outside of intersection	Vehicle crash with potential harm to vehicle or users	Vehicle drifts out of lane	1B	Stop	1F	Vehicle much less likely to collide a second time	Open
ICPS system update	FuHA-3	System update is not received	Vehicle and obstacle drive blindly, increasing risk of crash	Bluetooth disconnection or error	1C	Start time out sequence based on previous information	1F		Closed
ICPS system location calculation	FuHA-4	Location is miscalculated	A collision may occur	Bluetooth disconnection or error	1D	When vehicle cannot query, previously stored information will be used and emergency timeout is engaged	1F		Closed

System: ICPS Subsystem: Critical Functions			Functional Hazard Analysis				Analyst: CSC-436 Dream Team Date: 05/01/20		
Function	Hazard Index	Hazard	Effect	Cause	IMRI	Recommended Action	FMRI	Comments	Status
ICPS stopping weight	FuHA-5	Only one car reports real time info, the other queries	A collision may occur	Bluetooth disconnection or error	1B	Emergency timer is arbitrarily set for a safety bubble of 3 seconds	1F	Start time out sequence based on previous information	Closed
ICPS stopping weight	FuHA-6	Vehicle stops late	A collision may occur	Delayed response from Vehicle or Obstacle	1C	Start time out sequence based on previous information	1F		Closed
ICPS system emergency timeout	FuHA-7	System timeout not functioning	A collision may occur at the intersection	Too many ACK timeouts	1B	Flooding requests until the system acknowledges the stop request	1F		Closed

3.7 Failure Mode and Effects Analysis

System: ICPS Subsystem: Critical Functions			Failure Mode and Effects Analysis				Analyst: CSC-436 Dream Team Date: 05/08/20			
Item	Failure Mode	Failure Rate	Casual Factors	Immediate Effect	System Effect	Method of Detection	Current Controls	Hazard	Risk	Recommended Action
ICPS controller	Vehicle fails to stop at intersection	D	Inaccurate readings; miscalculated stopping weight	Stopping weight probability is low at intersection, when it should be high	Vehicle does not stop at intersection when it is about to collide with Obstacle	Continuous stream of location data from ICPS	Vehicle can stop and turn around	Vehicle collides with Obstacle	1F	Continue querying locations in intersection to implement an emergency direction change out of intersection
ICPS controller	Failure to receive ACKs	C	Vehicle or Obstacle unresponsive or out of range	ICPS controller waits for a limited amount time	Connection unsuccessful, ICPS terminates	Connection timer	ICPS fails to activate	ICPS unable to prevent collisions	1C	Resend connection query
ICPS controller	Failure to switch lanes	B	Inaccurate readings	Wrong or neglected lane data for Obstacle and Vehicle	Vehicle and Obstacle are in the same lane	Continuous stream of location data from ICPS	Vehicle can change lanes	Collision outside of intersection	1F	Move vehicle out of Obstacles lane at all cost

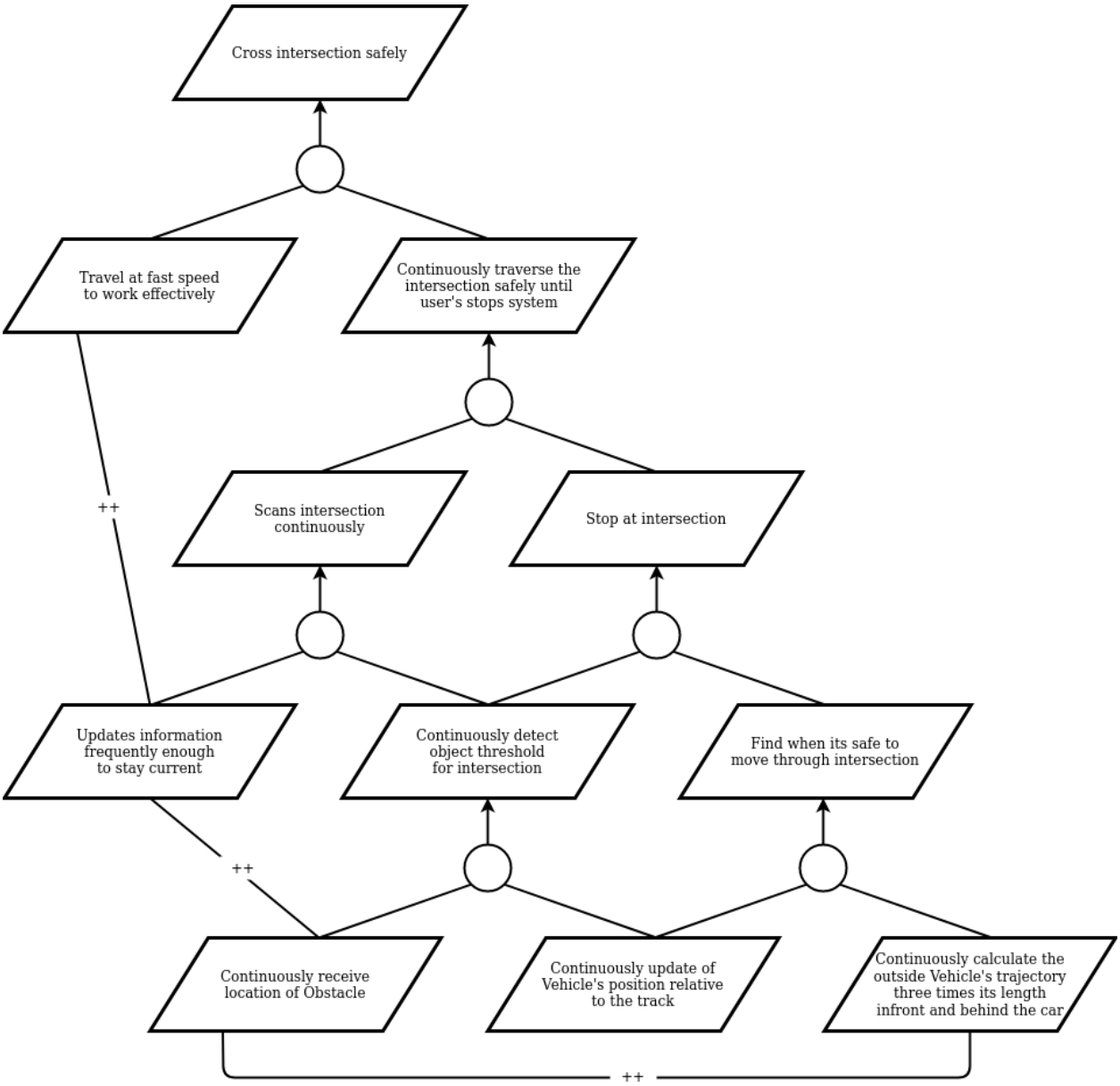
Item	Failure Mode	Failure Rate	Casual Factors	Immediate Effect	System Effect	Method of Detection	Current Controls	Hazard	Risk	Recommended Action
ICPS Controller	Bluetooth fails to report location	C	Bluetooth disconnects from one car during operation	Car will not report real time information	ICPS will lose track of where one car is and only be able to see one other car	The loop will fail, causing the system to retrieve stored data on problem car	Store previous location data, start emergency timer, and create an average and use last know location	Only one car reports real-time info,the other queries	1F	Emergency Timer reaches set time and preforms a system shutdown
ICPS controller	System timeout not functioning	D	Too many ACK timeouts	Vehicle continues around the track	Obstacle or Vehicle locations become increasingly inaccurate	None	Timer has static length	Locations unknown, ICPS cannot prevent collision	2E	Limit the amount of timeouts allowed
ICPS controller	ICPS fails to send or receive stop signal on time	C	Delayed response from vehicle or obstacle	Failure to send or receive stopping signal	Car could stop in the intersection	None	None	Vehicle stops late	1D	Emergency function that prevents the car from stopping in the intersection

3.8 Additional Comments

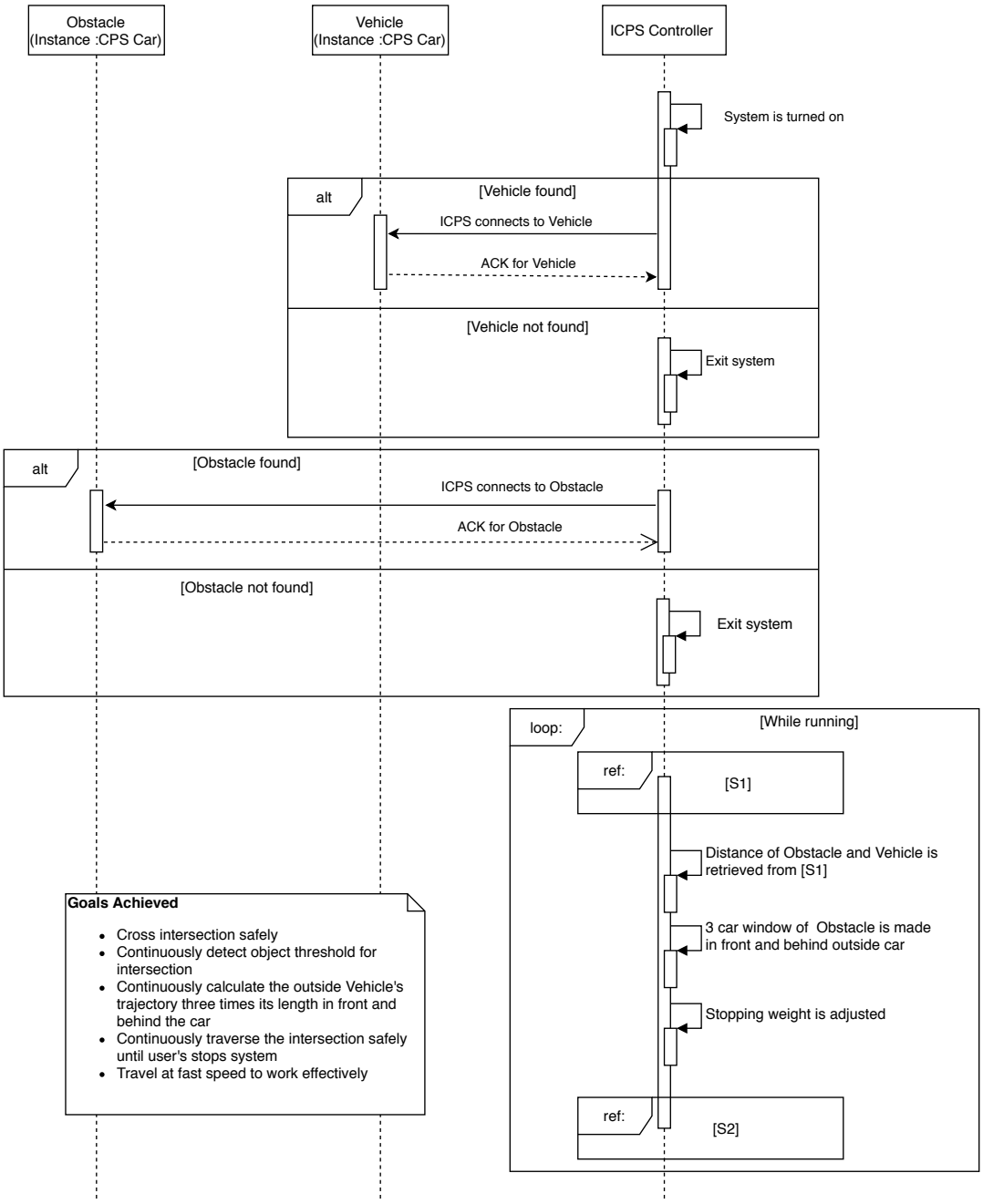
Standards Compliance, and its subsections have been cut because because there are no assigned standards for autonomous vehicles.

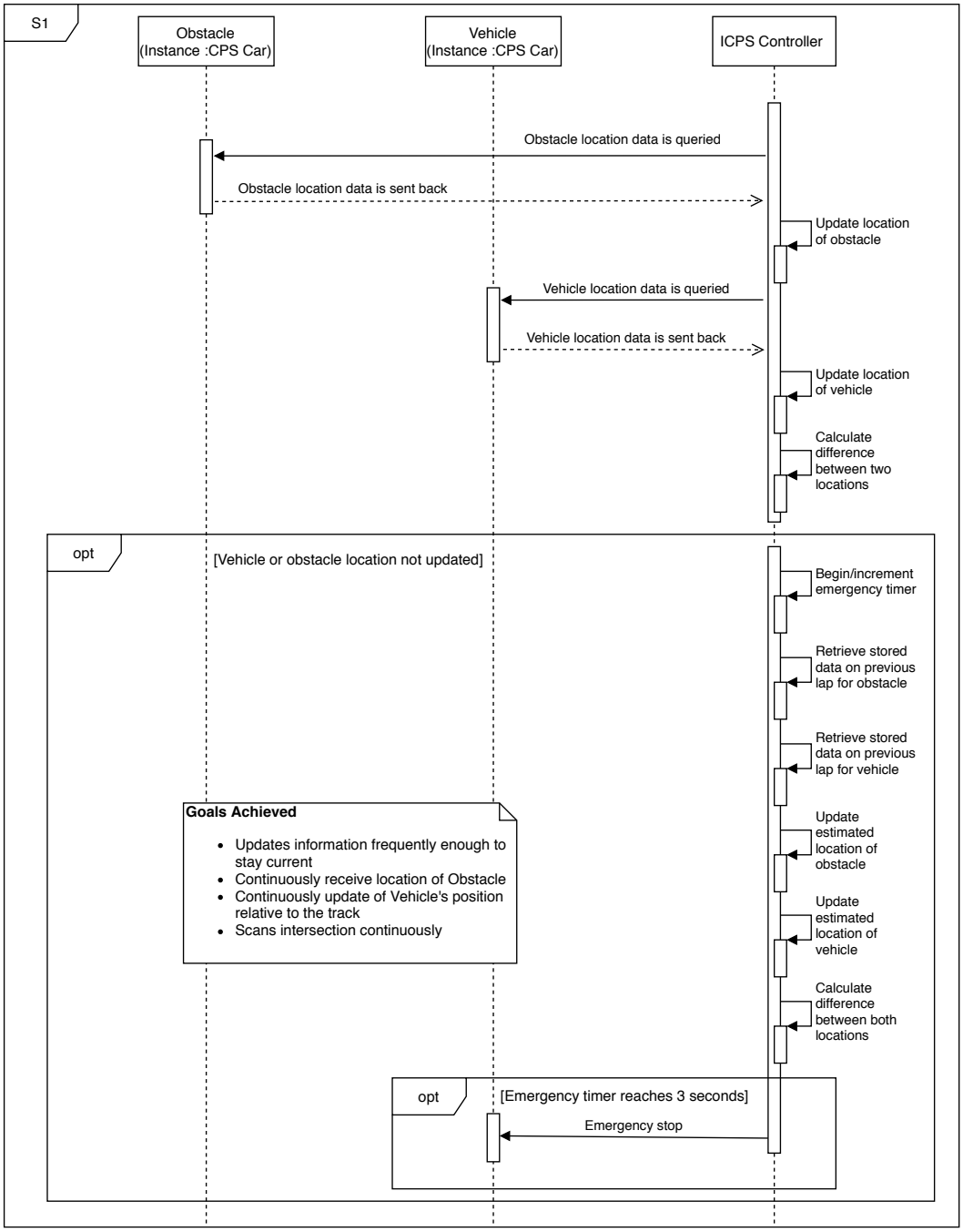
4 Structure

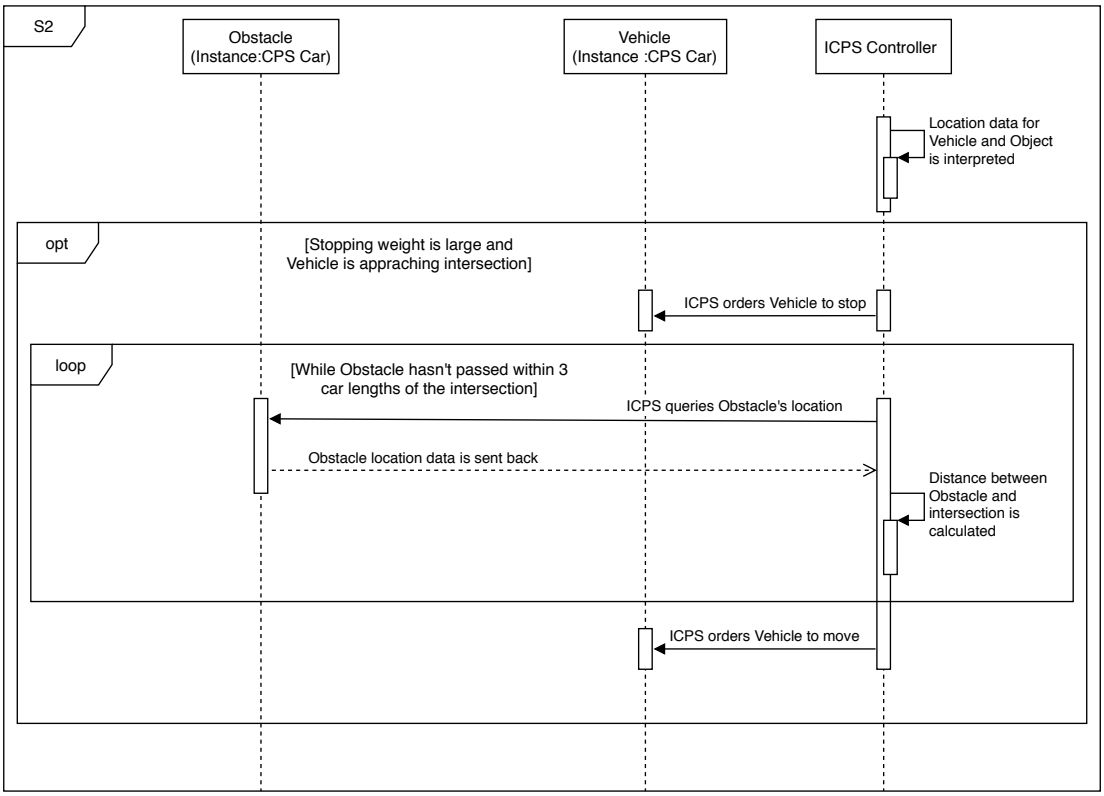
4.1 KAOS Diagram



4.2 Sequence Diagram



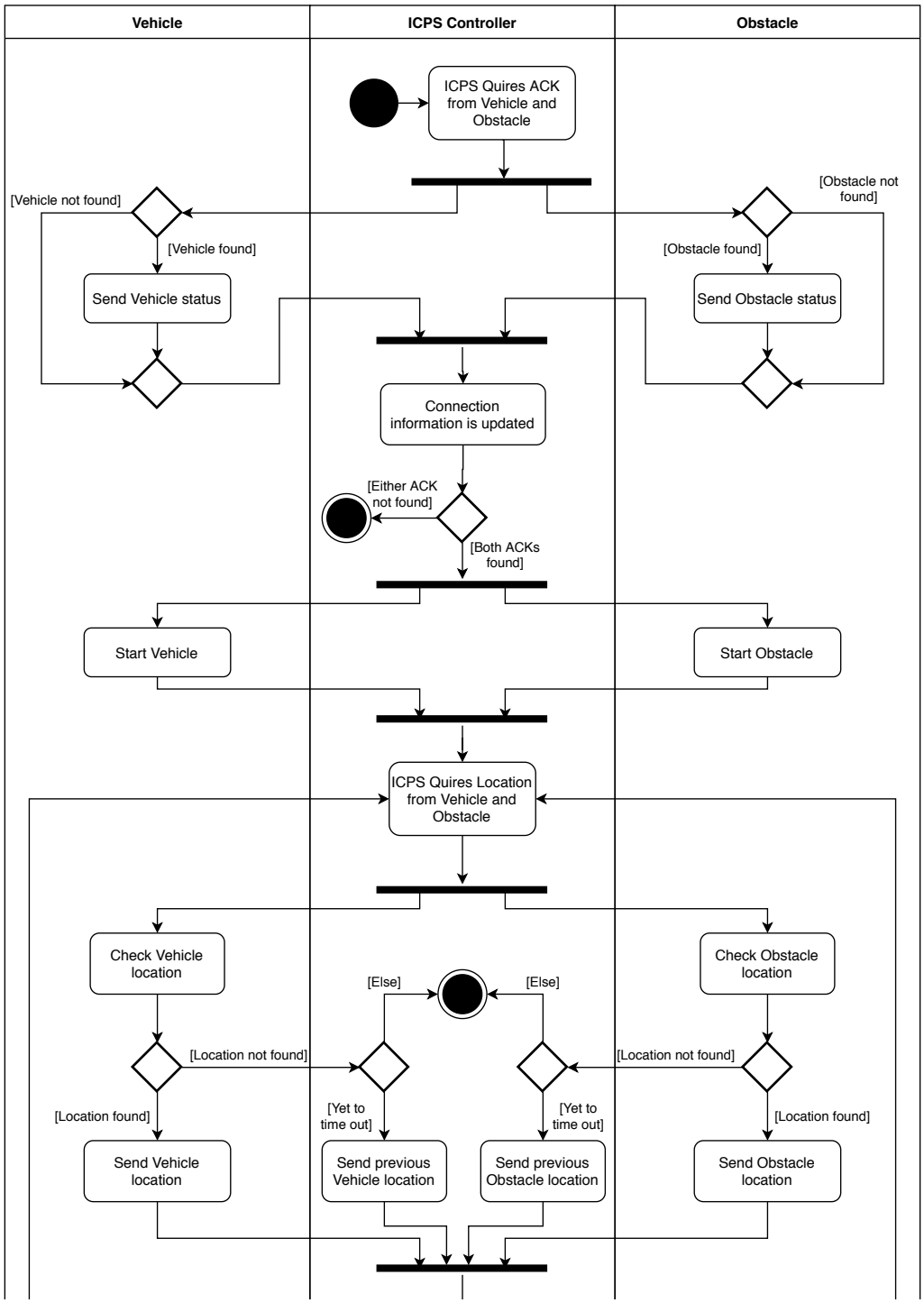

S1
Obstacle
(Instance :CPS Car)
Vehicle
(Instance :CPS Car)
ICPS Controller
Obstacle location data is queried
Obstacle location data is sent back
Update location
of obstacle
Vehicle location data is queried
Vehicle location data is sent back
Update location
of vehicle
Calculate
difference
between two
locations
opt
[Vehicle or obstacle location not updated]
Begin/increment
emergency timer
Retrieve stored
data on previous
lap for obstacle
Retrieve stored
data on previous
lap for vehicle
Goals Achieved
Updates information frequently enough to stay current
Continuously receive location of Obstacle
Continuously update of Vehicle's position relative to the track
Scans intersection continuously
Update
estimated
location of
obstacle
Update
estimated
location of
vehicle
Calculate
difference
between both
locations
opt
[Emergency timer reaches 3 seconds]
Emergency stop

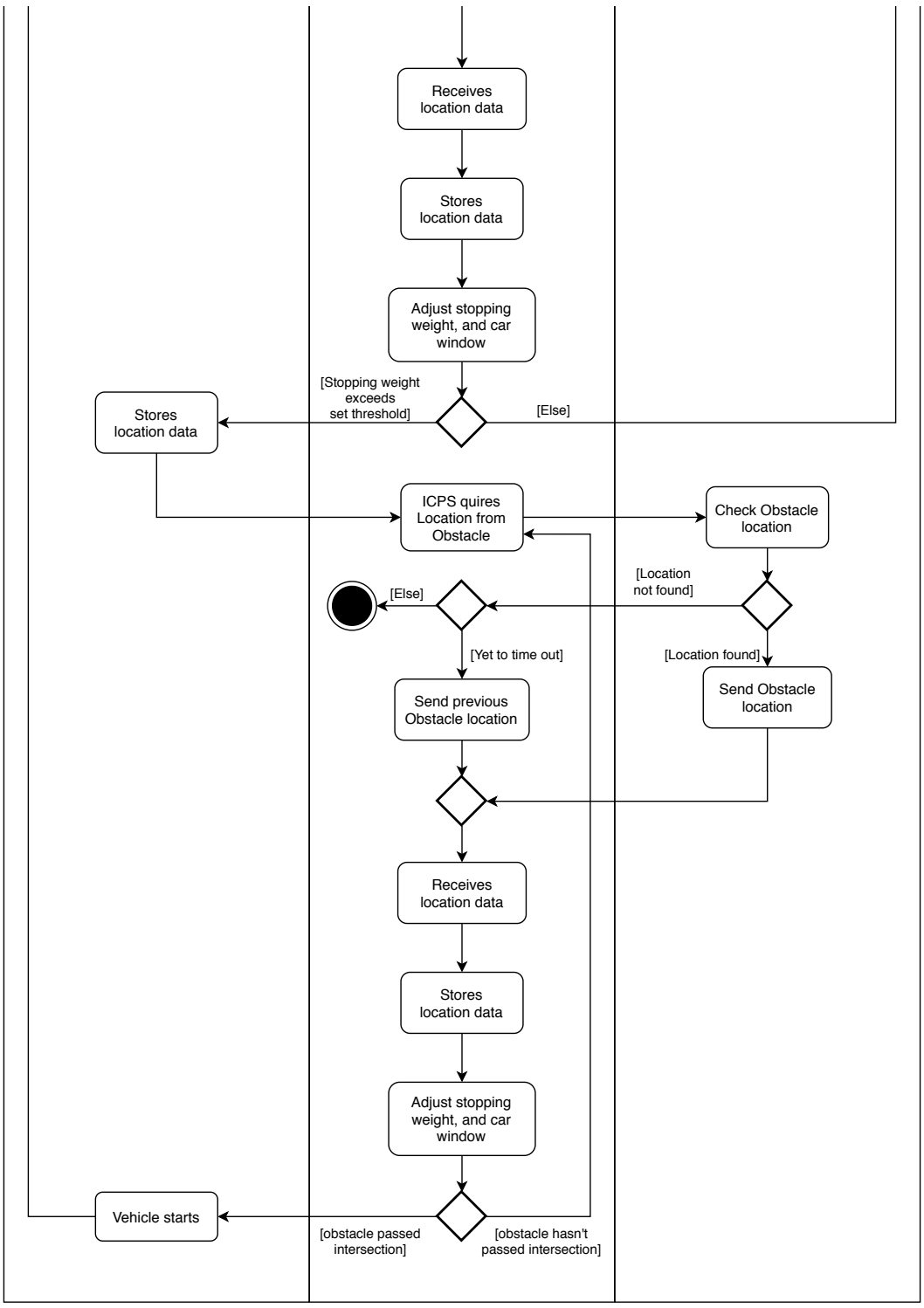


Goals Achieved

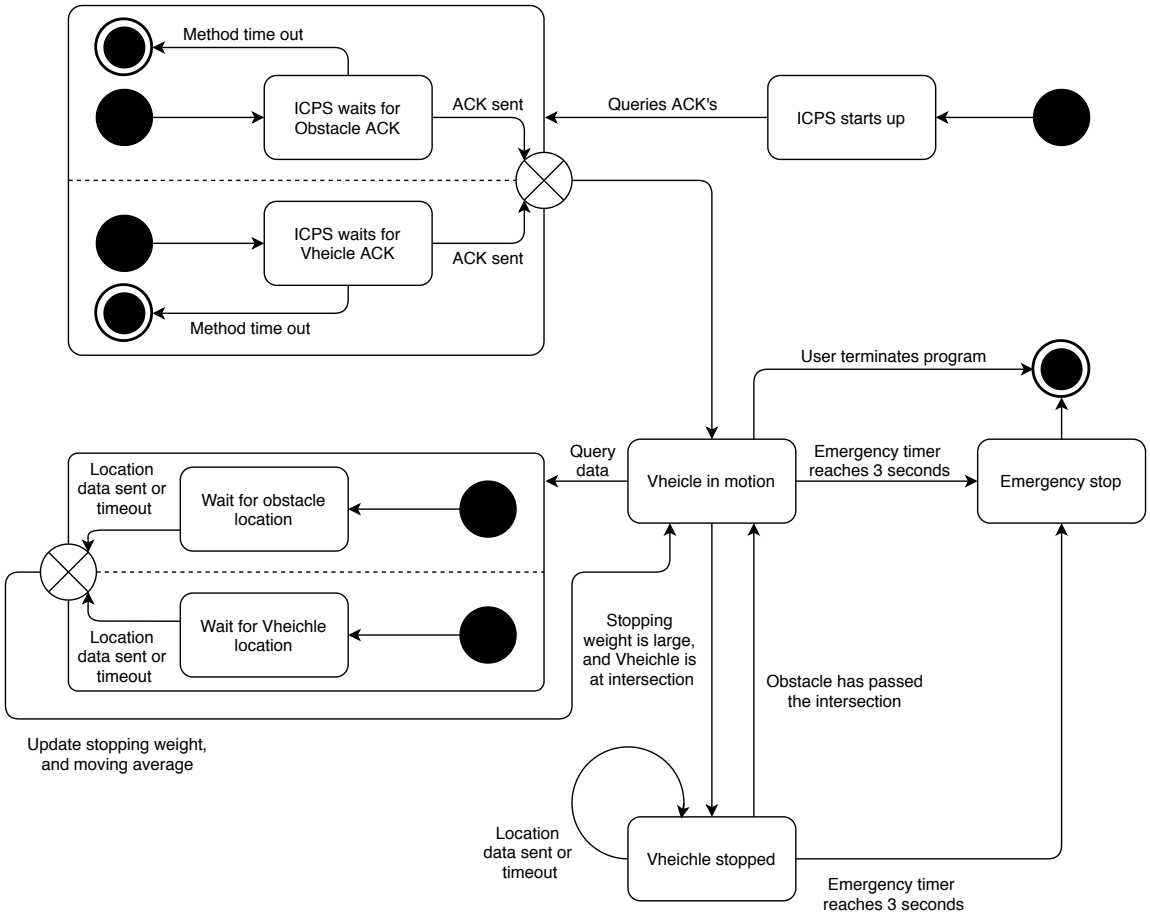
- Find when its safe to move through intersection
- Stop at intersection

4.3 Activity Diagram

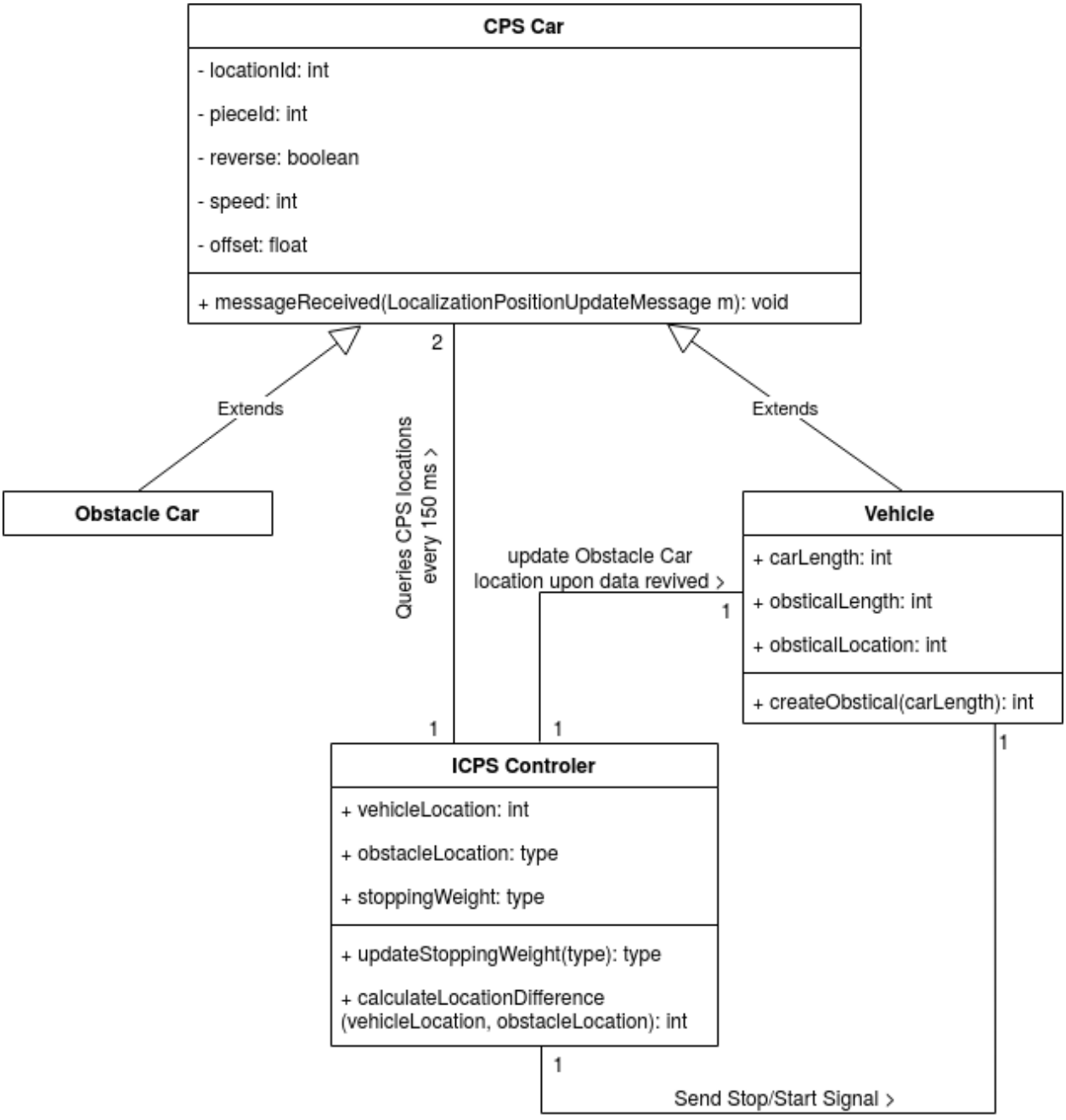



Receives location data
Stores location data
Adjust stopping weight, and car window
[Stopping weight exceeds set threshold]
[Else]
Stores location data
ICPS quires Location from Obstacle
Check Obstacle location
[Location not found]
[Else]
[Location found]
[Yet to time out]
Send previous Obstacle location
Send Obstacle location
Receives location data
Stores location data
Adjust stopping weight, and car window
Vehicle starts
[obstacle passed intersection]
[obstacle hasn't passed intersection]

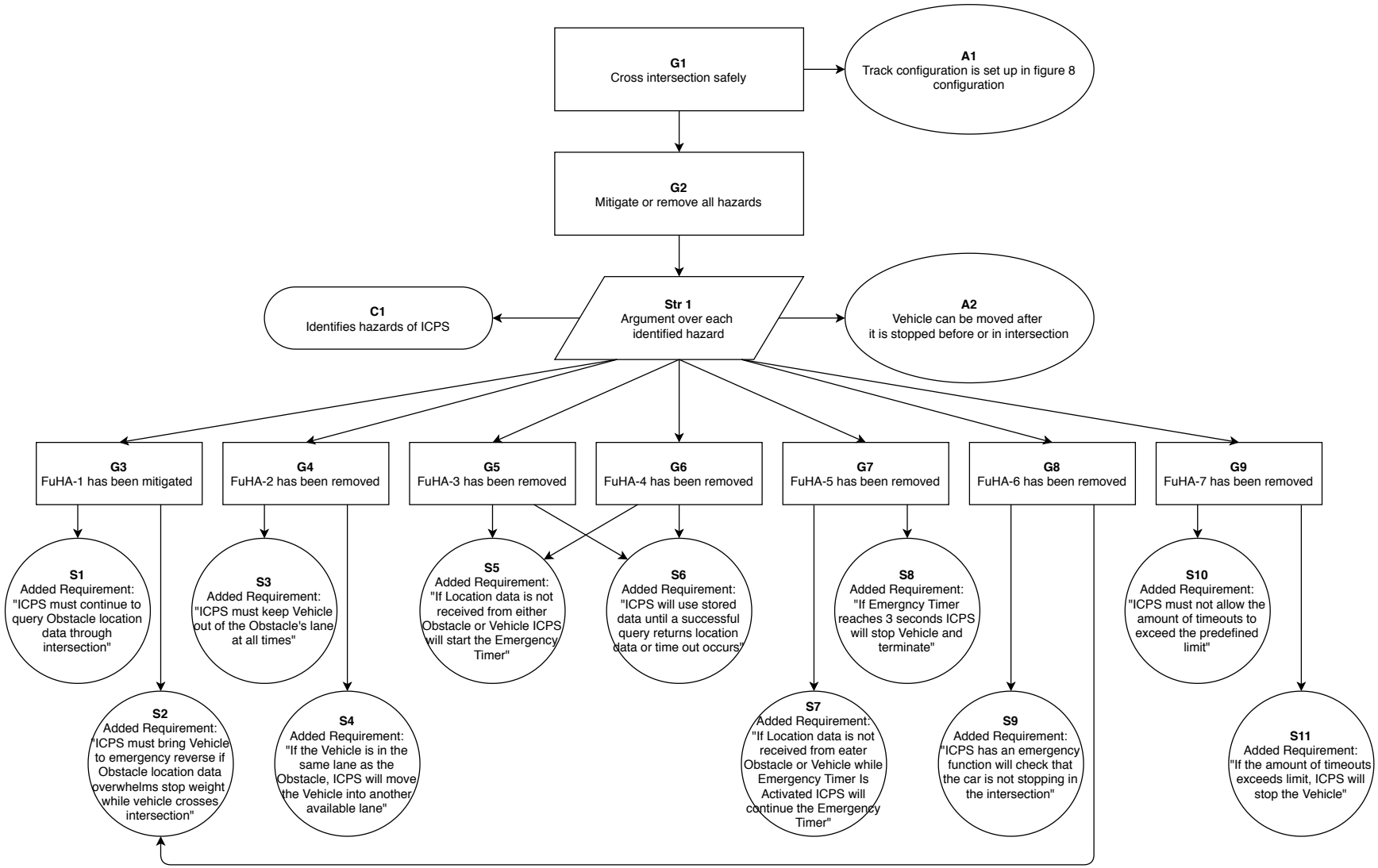
4.4 State Diagram



4.5 Class Diagram (Data)



4.6 GSN Safety Case Diagram



5 Team Reflection

At the beginning the team was split on the exact way to determine how our car would detect the obstacle car was in the intersection. Because of this we had a hard time coming up with what we thought were concrete, reasonable high level requirements. What we settled on, were high level goals that we knew were an absolute necessity for our system to even be possible. Examples include: assumptions that we made, safety, and the removal of human interaction. The further on the project when the more research our team and the more consensus we reached in terms of what we wanted our project to be. We sat in a room in the library initially, drawing out the early ideas of what we wanted the system to be on various whiteboards. These ideas would be pictures at first, but eventually became less abstract until we started writing down what would become our first milestones requirements. We went over what was written down and passed it by each member to ensure the requirements were air-tight. Once our class became affected by the whole COVID-19 ordeal with social distance learning, we were grateful that all the initial heavy-lifting was out of the way. There was also the relief that Overleaf and Draw.io were very good tools for distanced collaboration. Especially since our team has been utilizing this software since CSC 380. Part of what made our workflow so effective was our group's ability to constantly question one-another and revise each other's work without fear of being "wrong" or "not knowing". Our criticism was purely constructional and was always meant to further the project and not to bring others down.

In terms of what we found to be the most beneficial representations of our system the state diagram.This diagram seems to be the most human-readable. Considering how the system has many of the same states, there wasn't very much reiteration within this diagram, which leads to a very intuitive orientation to all of these states. The activity diagram in its early stages was very difficult to read, but It definitely shows much more of the nitty-gritty, and the current iteration definitely has much more of a concise flow about it. Our Kaos diagram was definitely the most infamous, because it was up to constant revision. Part of why it changed daily was our inability to secure a proper top level goal. Our sequence diagrams say much about our system in a similar way to the activity diagram, neither seem to be better or worse at describing ICPS. We didn't think either of the class diagrams did much in terms of describing our system. The main reason being that much of the computations are held within one class. We spent more time on how to make it bigger, than we did actually making contributions to the diagrams.

When we originally set out to do this, we all had a very clear idea on what we want the system to be. For the most part, a good chunk of what we had thought up was brought into the final design. We can say with utmost certainty that making all of the diagrams opened us up to many things that we didn't foresee, and gave us new insights on how to innovate our original design. Things like the conditions to handle timeouts, and weighted moving average were made a reality from this process. In the end, this allowed us to make more robust requirements, and our system blueprint stronger.