Jarvis

Self-Navigating Butler Robot

California University of Pennsylvania

Senior Project I

Specification Document

Dr. Weifeng Chen

11/12/2020

Project Members

Trevor Jamison – Computer Science

Elizabeth Sterling – Computer Science

Nadia Mason – Computer Science

Johnathan Bissontz – Computer Science, Electrical Engineering Technology, Computer Engineering Technology

Instructor Comments / Evaluation

Table of Contents

Abstract

Our project, Jarvis, aims to use artificial intelligence, computer engineering, and electrical engineering to produce a self-navigating robot. This project's main purpose is to create a device that provides an alternative for delivering items around a building. The robot will have the ability to carry objects on a balance plate and navigate its surroundings without dropping its objects or running into anything, in order to make deliveries to users. For example, by allowing a self-navigating robot to travel back and forth between destinations on a single hospital floor, employees will then have more time to focus on their daily tasks while also putting the safety of the patients and the individuals around them first.

The purpose of this document is to detail the requirements needed to create the initial prototype, and the final product should satisfy each detail explained in this document. This document will also specify how our product will benefit a hospital; however, there are a wide variety of places in which Jarvis may be useful. Jarvis' versatility allows him to benefit restaurants, offices, shopping centers, and schools, in addition to hospitals. The possibilities for Jarvis' use are nearly endless.

Document Description

Purpose and Use

The purpose of this document is to define the intent, scope, and outcome of the final product. Also, this document will give an overview of the software and hardware functionalities that this project must perform. Specifications may include, but are not limited to, the product's parameters and functionality. The client has full right to dispute any details set forth in this document should it not follow the intended business model or client requirements. The development team and client each reserve the right to discuss and make revisions to this document to ensure that implementation is completed in a timely manner. Should the terms and conditions in this document be accepted by both the client and development team, this document becomes a binding contract.

Intended Audience

The intended audience of this document would be any client wishing to purchase or use our product and the development team. The clients will use this document to ensure that our product is meeting any and all requirements of their business model. Any and all changes that the client wishes to pursue must be discussed with the development team and updated within this document. It is highly encouraged that clients read and acknowledge all parts of this document, seeking clarification when needed, as once the specifications are mutually agreed upon this document becomes binding. By ensuring clarity and understanding, this document will provide both the client and development team with a detailed list of features expected to be on the final

product. The details laid forth in this document will set a standard for which the development team will follow. This standard and the outlined constraints will allow the development team to provide a unique product tailored specifically to each client.

System Description

Overview

Jarvis is a self-navigating robot coupled with a remote user interface that will enable easy delivery of items on the floor of any building it has the layout of. Our robot will have three separately functioning but combined mechanisms, the first being the robot's navigation system, the second being a balance plate for the item(s) being delivered and third being a user interface that allows requests from a client and control from an owner's perspective. Once the user requests a delivery, the controller will then accept or deny the request, and if it is accepted the robot will navigate the floor plan and deliver the requested item quickly and efficiently.

Environment and Constraints

End User Profile

The first end User (The User) of this project is any staff, patient, customer, or visitor of the facility in which Jarvis is located. The User will interact with the physical robot as well as with the Web-Application. This end-user will have the ability to make requests for delivery and pick-up of items through the Web Application and confirm the completion of a transaction. The necessary tools of this specific end Users consist of:

- Internet Access

- The ability to read floor plans

The second end User is an Administrator. The Administrator will have permission to access to confirm and deny different requests made by other users. The Administrator will also have access to reports from Jarvis. This allows the Administrator to know any obstacles Jarvis may have encountered and was unable to navigate. The Administrator will also have access to Jarvis' charging station and power banks. The necessary tools of this specific end User consist of:

- Administrative sign-in credentials
- The ability to read floor plans
- The ability to understand the technical reports produced by Jarvis

User Interaction

The user will interact with Jarvis in 2 basic ways. The first interaction a user will have with Jarvis is the initial process of requesting a transaction through Jarvis' website. The user will load a webpage and have access to many different features of Jarvis. From this webpage, the user will have the ability to track Jarvis, make a request for delivery, or make a request for pick-up. Once this request is made, an administrator will have the ability to confirm or deny the requests and deploy Jarvis.

The most notable user interaction will be the physical interaction between Jarvis and a user. This interaction will happen directly after a request has been approved and Jarvis is deployed. Jarvis will navigate throughout the facility, ultimately reaching the requested destination. At this point, the user will have the ability to place or remove any item(s) from

Jarvis' balance plate. The user will then confirm pick-up or delivery and Jarvis will then travel to its next destination.

Hardware Constraints

- *Raspberry Pi 4B:* The Raspberry Pi will house the navigation software as well as the sensor software for obstacle avoidance. A separate power supply from the main system will be used to power the Raspberry Pi and attached sensors. An SD Card will be used to store navigation data. The Raspberry Pi has multiple GPIO ports for attaching sensors and other devices. The hardware constraints for the Raspberry Pi will be the memory size. We will work with the data capacity of the SD card.
    - *2x LC-217 Ultrasonic Sensors:* The Ultrasonic Sensors will be used to detect any objects in our robot's path as well as behind the robot while reversing. The constraints of the sensors will be their range of approximately 0.8" – 137".
    - *1x Servo Motor:* There will be one servo motor to allow the front facing ultrasonic sensor to sweep back and forth in order to detect any objects in the robot's path. The constraints of this servo motor are its angular range of motion, which when choosing a servo model, we will be looking to be at least 60 degrees.

- *Arduino Due:* The Arduino Due microcontroller is the component that will be used for the balance plate control system and this will remain independent of the other systems. The hardware constraints for the Arduino Due microcontroller will be its bootup time.

The balance plate cannot be utilized during system bootup, and therefore an indication of when bootup is complete will need to occur.

    - *2x HiTEC HS 5485HB Servo Motors:* The two servo motors will control the two rotational degrees of freedom, that being the forward and backward tilt adjustments as well as the left and right tilt adjustments. This will ensure that the object(s) does not get dropped off of the balance plate. The main constraint in concern when choosing these servos is command response time so that we can minimize the amount of delay in balance plate reaction.

- *5-Wire Resistive Touchscreen:* This will be the balance plate itself. A touchscreen is used to be able to detect where an item is on the plate at any time. The constraints of concern here are the accuracy of the location being reported by the touchscreen and the speed in which changes in the location are sent. Both of which are being considered when choosing this component.
- *2x Motors:* The motors will be electrical machines that convert electrical energy into mechanical energy and give the robot the power needed to move its four wheels. The motor's constraints are the provided torque. Depending on which motors are chosen there will be overall system weight limitations to prevent motor burnout.

- *Power Supply:* The power supply will be a power bank that will be rechargeable. The power supply will be running the two motors, the Arduino Due microcontroller, as well as the Raspberry Pi. The power supply's constraints are that they will always need to be charged and ready for use at all times. They will also require standard maintenance

protocol and time for recharging. We plan to have a back-up power bank to be charged while the other is in use.

Software Constraints

- *Navigation:* The main constraint of the navigation is due to the ability of Jarvis to be able to fully avoid obstacles. In some cases, there may not be a way out for the bot to navigate and find. As a group, we will do our best to make sure that any obstacles in the way can and will be avoided as much as possible. In the case of an unavoidable obstacle Jarvis will send a report to the admin with the estimated obstacle location and then return to the home node.
- *Balance Plate:* The main constraint of the balance plate software is that it must run on the Arduino Due. There will be no user interaction with the software itself, but it must be able to run without a need for constant adjustments or updates. Another constraint we are considering is the speed in which the balance plate reacts to item movement, this reaction will need to be as fast as possible to account for abrupt changes.
- *User GUI:* The user interface will be web browser based, so will not require a specific operating system. Although there may be some web browsers that do not support our user interface, we will attempt to make the interaction available to a wide number of browsers. Developing an interface that is easy to understand and use is also a concern.
- *Admin GUI:* As with the user interface the administrator interface will also be web browser based. This results in similar concerns and constraints.

- *Server:* The main constraint of concern with the server is reliability, we need to ensure the server is remaining live without downtime. Scalability would also be a constraint, but given we are only creating a single Jarvis robot, this will not be too much of a concern. The development team will do its best to develop a server with a reliable infrastructure that does not require much maintenance.

Time Constraints

The time constraints of this project are determined by both the client and development team. As this project is being completed for academia, the implementation will be constrained to the length of the Spring 2021 semester. Every aspect of this project will depend on this specific constraint. Our team realizes that this constraint is crucial to the development of the project and will devote many hours toward the completion of this project.

Cost Constraints

With this project being developed by a team of four college students the cost constraint is of large concern. As a result of this, we will be developing a prototype that is mostly a proof of concept with a rough design. Our team will look to reduce cost throughout the development in order for us to be able to afford the physical product.

Other Concerns

While we plan to house our server on the Raspberry Pi to communicate data to and from the web application this may prove to be difficult or inefficient. A stable internet connection for the Raspberry Pi with proper protocol permissions is required to allow this to happen as well. In

the case of the server not being able to be developed on the Raspberry Pi, we plan to explore the option of renting a server during project development and testing.

A final concern is security. While Jarvis will not house any information about the items being delivered or information about the users being delivered to, the overall security of the software and physical security of the device itself is a concern. Due to the time constraints of the project, we will not have time to explore security strategies besides following safe networking procedures and ensuring our code is proper.

Acceptance Test Criteria

Testers

The majority of our testing will be done by the Jarvis development team. Not only will continuous testing be upheld throughout development, but also, during production each separate component will be tested by each member of the team prior to being integrated into the complete system. Once the prototype is nearly complete, Jarvis will be tested by a small group of individuals. This testing phase will provide valuable insight on the user experience without bias from the development team.

Criteria for User Acceptance

Our clients and users depend on the successful travel and delivery of items throughout their facility. In order to best achieve this, the development team has pre-determined a list of criteria that must be met in addition to the client parameters. By ensuring that every pre-determined item and client parameter is met, the development team can consider Jarvis a success.

Jarvis will be judged on the following criteria:

- Jarvis can navigate from the start location, to a delivery destination, and return without crashing into objects.
- Obstacles that cannot be navigated around are reported to the admin and Jarvis returns to the start location.
- The item(s) on the balance plate do not fall off in transit.
- User(s) can request a delivery.
- Administrators can confirm a delivery.
- User(s) and administrators can track Jarvis while in transit.
- User(s) can confirm delivery was received.
- User(s) can rate the performance of Jarvis (on a five-star scale).

Integration of Separate Parts and Installation

There is integration of many parts within Jarvis. The front-end software will be a browser-based GUI for both end users and admin. The navigation software will be located on the Raspberry Pi, as well as the sensor software for obstacle avoidance. Separately, the Arduino Due will house the balance plate control system, this will remain independent of the other systems. All of these components will be located on a single main chassis of the robot.

No installation will be required on the users end, since all interaction will be done through a web app. There will be installation of the physical components onto the robot though, and this will occur as individual components are tested to be working independently.

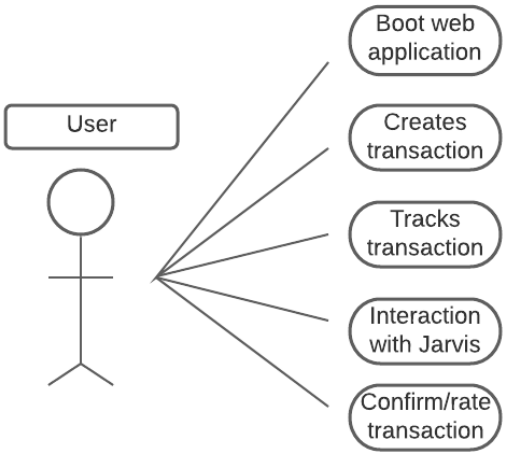
System Modeling

Functional: Use Cases and Scenarios

There are two common use cases for this specific project: The User and the Administrator. The User will use a web application to request pick-ups and deliveries from Jarvis. From this web-application, the User will also be able to track their transaction from start to finish. This will allow for the User to get a good idea of when Jarvis will be arriving to either pick-up or deliver their item(s). After the completion of their transaction, the User will have the ability to rate their experience with Jarvis.

The Administrator will use the web application to confirm a pick-up or delivery and then deploy Jarvis. Once Jarvis is deployed, the Administrator will be able to watch the transaction and ensure that Jarvis does not run into any obstacles that he is unable to navigate around. The Administrator will do this by reading different reports produced by Jarvis. After the transaction is complete, the Administrator will receive the confirmation from the User and confirm Jarvis' next transaction.

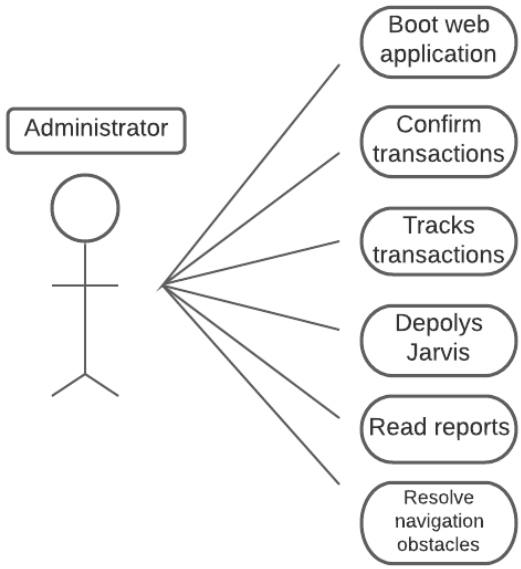
The User:



Normal User Scenario:

1. The User signs on to the Web Application
2. The User creates a transaction by requesting a pick-up or delivery
3. The transaction is approved by an Administrator – User is notified
4. Jarvis travels to User's location
5. Jarvis and User interact – Pick-up received by Jarvis or item(s) delivered to User
6. User will confirm transaction has been completed and rate transaction through Web Application
7. Jarvis travels to next location

The Administrator:



Normal Administration Scenario:

1. The Administrator will sign onto the web application
2. The Administrator will wait for a request from the User
3. Approve or deny the request
4. Deploy Jarvis
5. Track Jarvis' progress to and from destinations
6. Confirm delivery or pick-up
7. Read User satisfaction/Review
8. Confirm Jarvis' next task

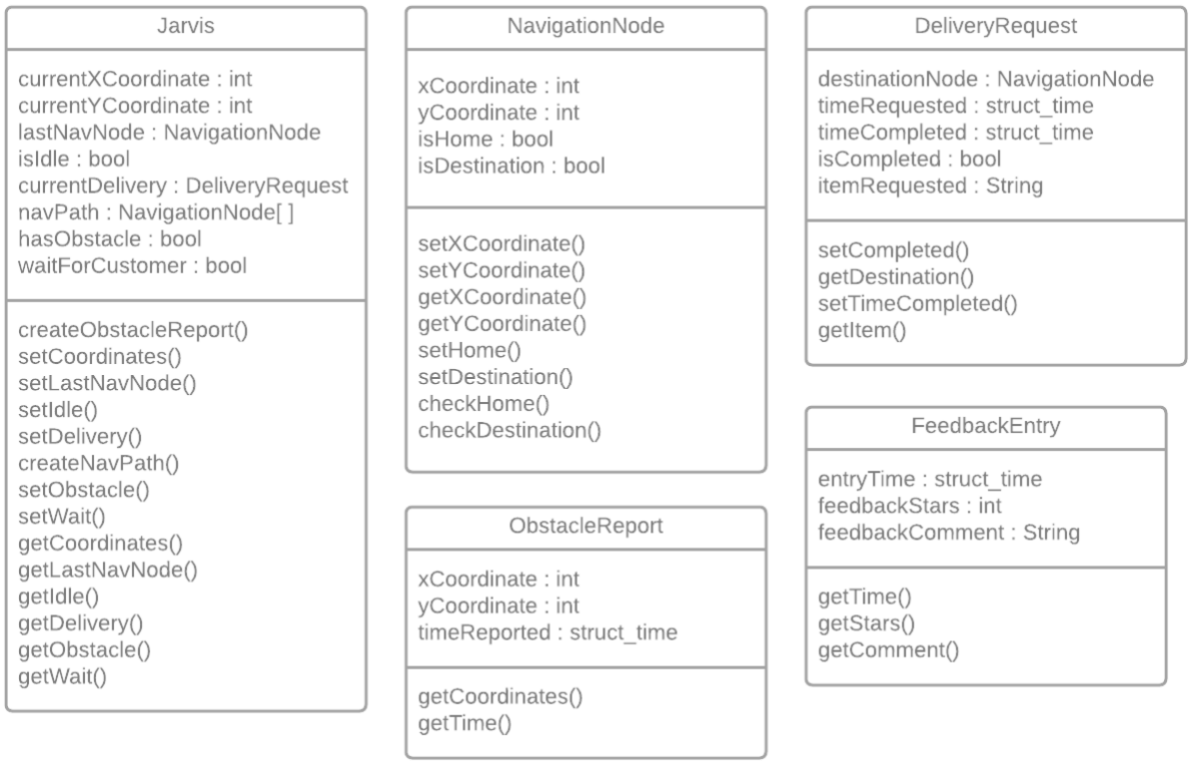
Alternative Administration Scenario:

1. The Administrator will sign onto the web application
2. The Administrator will wait for a request from the User

3. Approve or deny the request
4. Deploy Jarvis
5. Track Jarvis' progress to and from destinations
6. Jarvis returns to home state
7. Administrator reads the report as to why Jarvis returned
8. Administrator will resolve issue or remove obstacles
9. Redeploy Jarvis
10. Track Jarvis' progress to and from destinations
11. Confirm delivery or pick-up
12. Read User satisfaction/Review
13. Confirm Jarvis' next task

Entity: Class Diagram

All classes will exist in the main system or within the user interfaces. The balance plate system, as well as the sensors and motor driver will be developed in an embedded system environment and therefore will not utilize class structures.



Class Descriptions

Jarvis Class: This is the main class that will house the majority of the robot's functionality in interactions with both users and administrators. Attributes of the Jarvis class are its coordinates relative to the floorplan, its last visited node, if it is idle, its current delivery, the navigation path it is following, if it is currently navigating around an obstacle, and if it is in a wait state after arriving at a destination.

NavigationNode Class: This class is what will be used to allow Jarvis to navigate throughout a floor plan without using imagery, and instead using distance traveled relative to the past node. Attributes of the NavigationNode class are its coordinates on the floorplan, if it is the home node, and if it is a destination node.

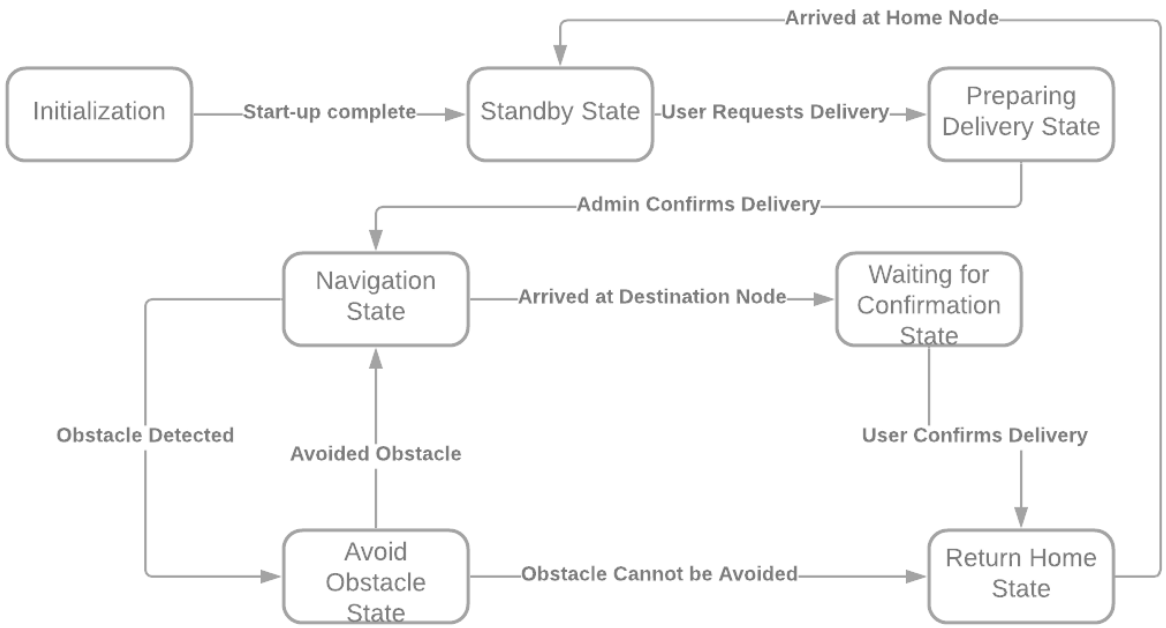
DeliveryRequest Class: This class is what will house the information about a delivery request, this is what will be created when a user fills out the Delivery Request Form. Attributes of the DeliveryRequest class are which destination node it is going to, the time it was requested, the time it was completed, if it has been completed, and what item is being requested.

ObstacleReport Class: This class is what will house the reports from Jarvis to the administrator about any obstacle that could not be navigated around. Attributes of the ObstacleReport class are its coordinates relative to the floorplan and the time it was created.

FeedbackEntry Class: This class is what will house the entries into the user feedback page that will be sent to the admin. Attributes of the FeedbackEntry class are the time it was created, a score out of 5 stars, and any comments that the user left.

Dynamic: State chart

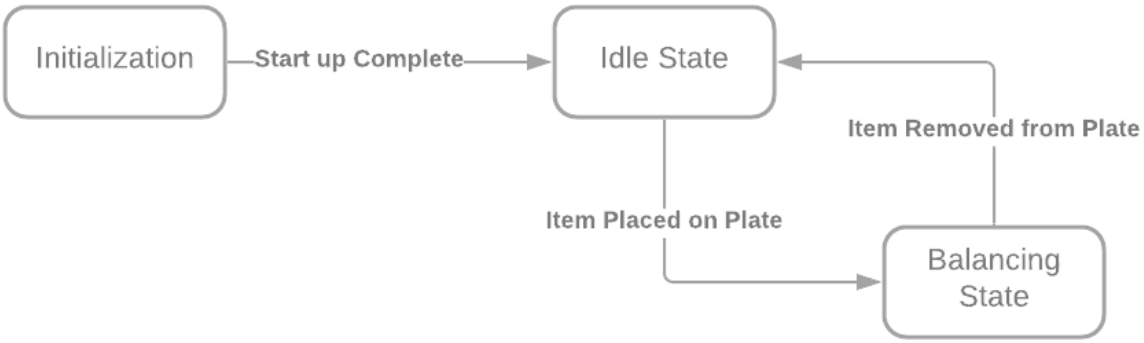
Main Navigation System



- States
  - Initialization: Jarvis is turned on and is initializing the navigation software.
  - Standby: Jarvis is standing by waiting for delivery requests.
  - Preparing Delivery: Jarvis is waiting for an admin to confirm the delivery before moving.
  - Navigation: Jarvis makes its way along a route toward its destination node.
  - Avoid Obstacle: Jarvis attempts to navigate around the obstacle.
  - Waiting for Confirmation: Jarvis is waiting for recipient to confirm delivery completion.
  - Return Home: Jarvis navigates back to home node.

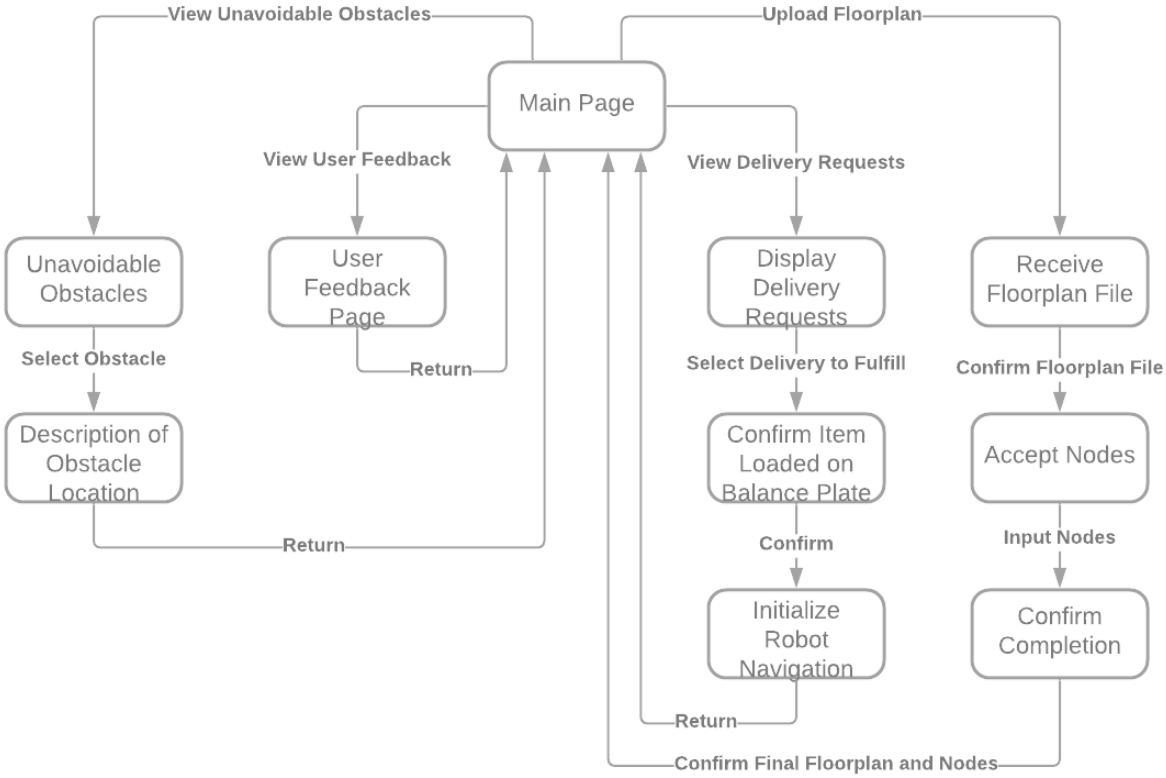
- Events
    - Events are all the scenarios that cause Jarvis to transition into a different state. The events that Jarvis will encounter are user requests, admin confirmations, obstacle encounters, arrival of destinations, confirmation of deliveries and lastly arrival back to the home node.
- Transitions
    - Transitions occur in several different scenarios. The first transition being when Jarvis goes from the “standby” state to the “preparing delivery” state and this occurs when the user requests a delivery. Once the delivery request has been accepted and approved by the admin that is when the next transition occurs, and Jarvis goes into the “navigation state”. From the “navigation state” Jarvis has two possible states that could be transitioned to. If Jarvis encounters an obstacle, then transitions into the “avoid obstacle” state and then back into the “navigation state”. If no obstacle is detected and the delivery is made Jarvis transitions into the “waiting for conformation” state. After the user confirms the delivery, Jarvis enters the “return home” state and from there finally back into the “standby” state.

Balance Plate



- States
    - Initialization: the balance plate is initializing software.
    - Idle: the balance plate is ready for an item to be placed on it.
    - Balancing: the plate is continuously attempting to keep the object centered.
- Events
    - Events are what cause the transitions to occur. The events of the balance plate are whenever an object gets placed onto the balance plate and when the object gets removed from the balance plate.
- Transitions
    - Transitions occur whenever an item is placed or removed from the balance plate. When placing an item, the balance plate goes from the “idle” state to the “balancing” state. Similarly, when an item is removed from the balance plate, it transitions from the “balancing” state to the “idle” state.

Admin UI

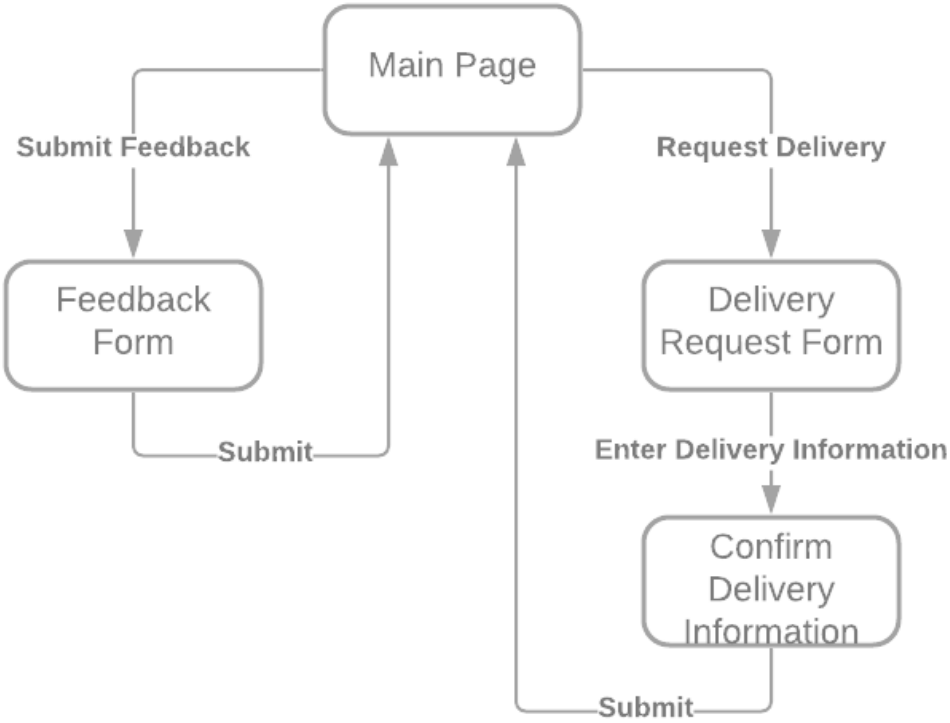


- States
    - Main: home state in which Jarvis' current location will be displayed.
    - User Feedback: will display a list of submitted user feedback.
    - Unavoidable Obstacles: will display a list of unavoidable obstacles reported by the robot.
    - Description of Obstacle Location: an approximate location and time the selected obstacle was reported.
    - Display Delivery Requests: will display a list of current requested deliveries.
    - Confirm Item Loaded on Balance Plate: requesting the admin confirm that the item has been loaded on the balance plate.

  - Initialize Robot Navigation: starting the navigation system.
  - Receive Floorplan File: allowing the upload of a floorplan file.
  - Accept Nodes: requesting home node, delivery nodes, and intersection nodes be added to the floor plan.
  - Confirm Completion: requesting admin to confirm the floorplan and node combination.

- Events/Transitions
  - Events are all the scenarios that cause Jarvis to transition into a different state. For this state chart there are four main branches of states with scenarios. The first set of scenarios being uploading a floor plan, confirming a floor plan, inputting nodes, and then confirming final floor plans and nodes. The second set of events would be viewing delivery requests, selecting which delivery to complete and then confirming and returning. The third set of events pertains to obstacles and they are viewing the unavoidable obstacle, selecting the obstacle and returning. The last event that could occur on the Admin UI would be the view user feedback selection.

User UI

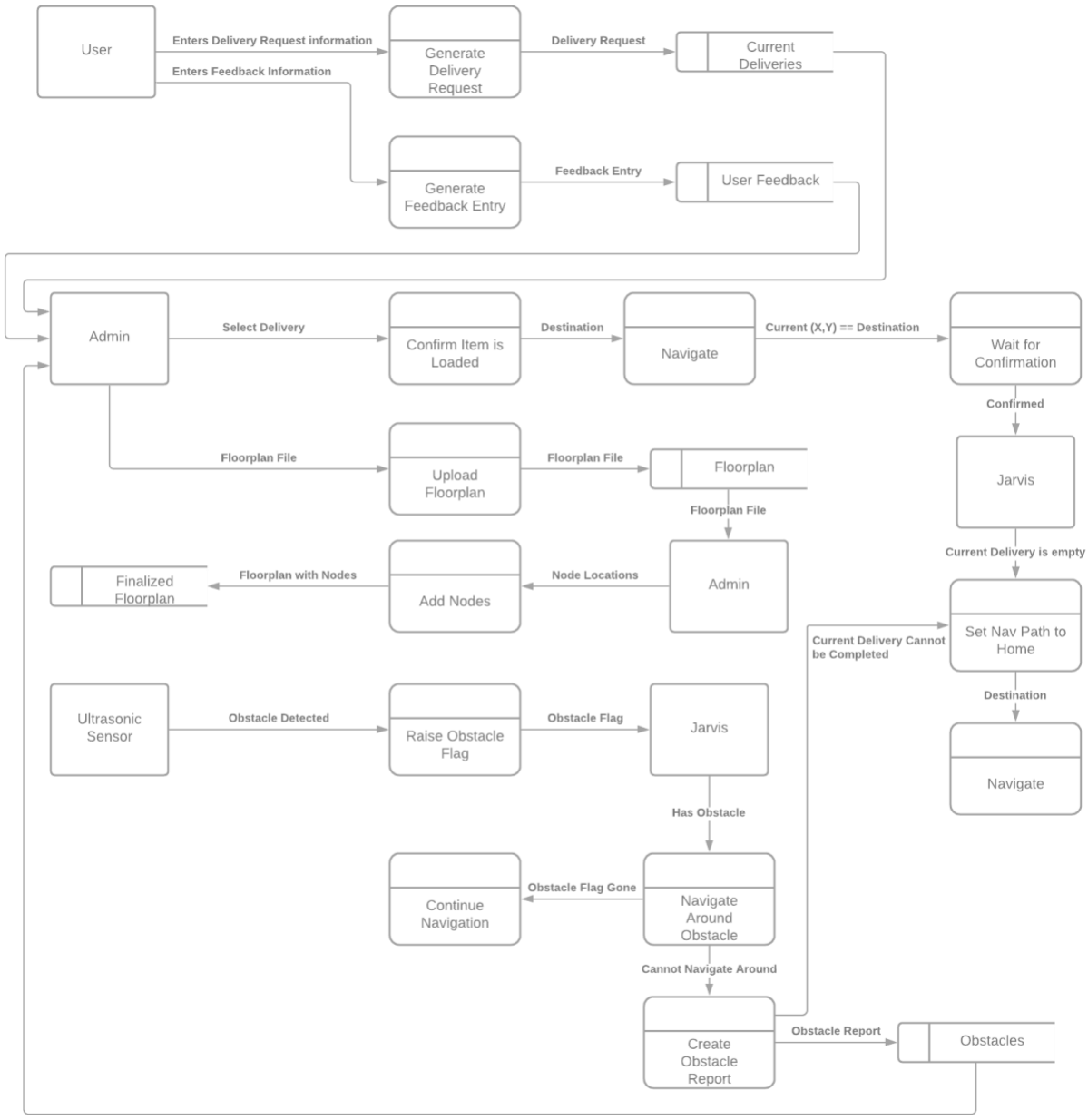


- States
    - Main: home state in which Jarvis' current location will be displayed.
    - Feedback Form: will display a form to submit feedback to the admin.
    - Delivery Request Form: will display a form to submit a delivery request to the admin.
    - Confirm Delivery Information: will request the user to confirm their delivery information prior to sending the request.
- Events
    - Events are what causes the transitions to occur. The events of the User UI are whenever a delivery gets requested, they enter the desired delivery items

information and then they submit that request. Also, another event that causes transitions to occur in the User UI is when the user is attempting to submit feedback.

- Transitions
    - The user has two different options whenever they are in the initial “main page” state. The first type of transition that can occur is when the user requests a delivery; this transitions the user from the “main page” state into the “delivery request form” state. From there the user fills out the information. Once submitted that will trigger a transition into the “confirm delivery information” state. Finally, from there the user will submit and that sets them back to the “main page” state. The other type of transition the user has the ability to do is submit feedback from the main page. This transitions them from the “main page” state to the “feedback form” state and once submitted they go back into the “main page” state.

Dataflow Diagram



Components / Tools Needed

Hardware

Main System:

- Raspberry Pi 4B
    - Power Cable
    - SD Card
    - Pi Case with Heatsink
- 2x LC-217 Ultrasonic Sensors
- 1x Servo
- 4x Wheels
- 4x Motors
- 2x Power Banks

Balance Plate:

- Arduino Due
    - Power Cable
- 2x Servo HiTEC HS 5485HB
- 5-Wire Resistive Touchscreen
- 2x Servo Arm
- 1x Support Arm

Development Tools:

- Monitor/Display

- Keyboard
- Mouse

Software

- Raspbian OS
- Python 3
- Windows 10
- Arduino IDE

Appendix A: Technical Glossary

Administrator (Admin) – in the context of Jarvis, an admin is anyone who has access to the admin UI and can approve, deny, and view delivery requests, as well as view user feedback and obstacle reports.

Arduino – an open-source electronics platform based on easy-to-use hardware and software.

Arduino board – a microcontroller capable of reading inputs and sending outputs along GPIO.

Boot – to start a program on a computer.

GPIO (General Purpose Input Output) – a pin or port on a board used for interfacing with peripheral devices.

GUI (Graphical User Interface) – a user interface that includes graphical elements and allows users to interact with an application or electronic device.

PID (Proportional Integral Derivative) **control** – a control system that utilizes system feedback for constant correction and control.

Raspberry Pi – a low cost, credit-card sized computer that is capable of doing nearly everything a desktop computer can do. Also has multiple GPIO ports for attaching sensors and other devices.

Resistive Touchscreen – a touchscreen that uses two flexible sheets coated with a resistive material and separated by a small air gap. When the touchscreen is pressed the two sheets touch together altering the voltage read and this voltage difference is translated into a position on the screen.

SD Card – a small, non-volatile memory card format commonly used in digital cameras, phones, and other portable devices.

Server – a computer equipped with software and/or hardware that allows it to provide functionalities to other computers over a network.

Servo – a small device similar to a motor with an output shaft that can have the angle of the shaft adjusted with control signals.

Torque – the rotational force applied to a shaft.

Ultrasonic Sensor – a sensor that measures distance using ultrasonic wave "chirps" by measuring the time it takes for the waves to bounce back to the sensor.

User – in the context of Jarvis, anyone who is receiving a delivery is referred to as a user.

Web Application –also known as a "web app", a software program that uses web browsers to complete tasks.

Appendix B: Team Details

The requirement document was developed by the following individuals, all of whom contributed the following:

Trevor Jamison – Trevor Jamison was responsible for the document's overview, abstract, hardware constraints, and software constraints. He contributed to the events and transitions for the main navigation system, balance plate, user UI and admin UI. He also worked with team members by reviewing and editing as necessary and contributed to keeping a uniform format throughout the progression of this document.

Johnathan Bissontz – Johnathan Bissontz worked on the constraints, acceptance test criteria, integration of separate parts, class diagrams, class descriptions, components/tools, state charts, dataflow diagram, and glossary. He also worked with the team to help organize completion of different sections and to proofread and edit throughout.

Elizabeth Sterling – Elizabeth Sterling worked on the completion of the abstract, overview, time constraints, cost constraints, purpose and use, intended audience, use cases, and use case diagrams. She also worked in multiple other sections to help the team organize and complete every required piece of the document. She also worked with the writing center to proofread and get a new perspective on the paper.

Nadia Mason – Nadia Mason… table of contents, purpose and use, user criteria, some software constraints, other concerns, glossary, grammar check (before writing center).

Appendix C: Report from Writing Center

Due to the length of this assignment our group was required to attend 2 sessions from the Writing Center. Here are screen shots of the feedback we received from each appointment.

First Appointment

Cal U Vulcan Learning Commons Report

Client: Elizabeth Sterling

Staff or Resource: Sarah C.

Date: November 12, 2020, 3:00pm - 4:00pm

What course was serviced by this visit?: CSC490

Did the student request that the instructor receive a visit report?: Yes

Please provide any additional comments relevant to this session.:

How did the process of this consulting session address the established goals?:

Because there were only ten pages left to be reviewed, it did not take me very long. I agree with what Megan said in her report--the paper is well-written. I just made comments on some minor grammar and wording elements. One thing I will advise is making sure puncuation and capitalization in Appendix A is consistent.

Second Appointment

Cal U Vulcan Learning Commons Report

Client: Elizabeth Sterling

Staff or Resource: Megan R.

Date: November 12, 2020, 11:00am - 12:00pm

What course was serviced by this visit?: CSC490

Did the student request that the instructor receive a visit report?: Yes

Please provide any additional comments relevant to this session.:

How did the process of this consulting session address the established goals?:

I reviewed most of the project, and left a note to mark where I left off. The paper has a few grammar and punctuation errors, and I made some wording suggestions, but overall it is well-written.

Appendix D: References

Arduino.cc. (n.d.). *Arduino Store*. Retrieved from Arduino: https://store.arduino.cc/usa/due

HiTec Multiplex. (n.d.). *HiTec Products*. Retrieved from https://hitecrcd.com/products/servos/sport-servos/digital-sport-servos/hs-5485hb-standard-karbonite-digital-servo/product

Microsoft. (n.d.). *Windows 10*. Retrieved from Microsoft.com: https://www.microsoft.com/en-us/windows/get-windows-10

Progressive Automations. (n.d.). *High Accuracy Ultrasonic Sensor*. Retrieved from Progressive Automations : https://www.progressiveautomations.com/products/lc-217

Python. (n.d.). *Python 3 Information and Download*. Retrieved from Python.org: https://www.python.org/download/releases/3.0/

RaspberyPi. (n.d.). *RaspberryPi Store*. Retrieved from https://www.raspberrypi.org/products/raspberry-pi-4-model-b/?resellerType=home