Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4317: Senior Design II
Spring 2025

Heals on Wheels
HMETV

Nicholas Doerfler
Rency Kansagra
Tristan Moses
Dyana Rahhal
Christian Brown

Revision History

Revision	Date	Author(s)	Description
0.1	2.23.2025	all	document creation
0.2	2.28.25	CB	complete draft

CONTENTS

List of Figures

List of Tables

1 Introduction

The HMETV (Hospital Medical Equipment Transport Vehicle) is a system of autonomous carts that deliver medical supplies, tests, or any other needed resources, to where they need to go in an efficient and reliable manner, taking the burden of menial tasks off of critical hospital staff. The system takes orders through a mobile application, and then summons a supply cart to the desired destination with the requested supplies.Upon receiving a request,the system identifies an available cart and directs it to the designated pickup and drop-off locations using advanced pathfinding algorithms and real-time obstacle detection.The carts seamlessly navigate hospital environments,ensuring the timely and secure transportation of medical resources while reducing the physical workload on healthcare professionals. This document presents the detailed design of the HMETV system,outlining its key components,functionalities,and implementation strategies.The design specification are based on the foundational requirements outlined in the System Requirements Specification (SRS) document, which details the system's objectives,user interactions,and functional expectations. Additionally,the Architectural Design Specification (ADS) document provides an in-depth overview of the system's structural components,including the mobile application, application backend, onboard computer, power supply, chassis, sensors, and drive mechanisms. By referencing these document,the following sections provide comprehensive view of the system's architecture,ensuring alignment with both functional and technical requirements.

2 System Overview

The overall system is designed to facilitate communication between the physical cart (HMETV), the onboard computing, and computing not done on the cart (Off site computing). At an overall point, the offsite computing is for processing requests, such as sending a cart to a location. The onboard computing manages the physical cart to obtain environmental data and process it to determine what the cart should do. The HMETV is the physical cart and is what physically moves

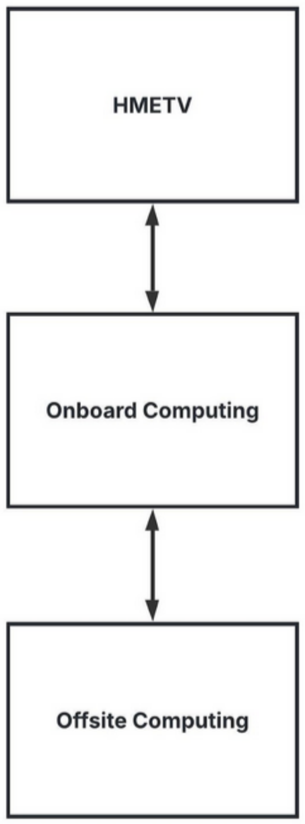

Figure 1: System architecture

2.1 HMETV

The HMETV subsystem is the physical cart. It is responsible for facilitating movement as well as power for the entire cart, along with having an emergency stop. This contains a subsystem that has the motors and motor control boards. It also contains a subsystem for power, which has the power supply and battery. The battery provides power to the power supply and the power supply safely provides power to the motors and raspberry pi.

2.2 Onboard Computing

The Onboard computing subsystem is responsible for data acquisition, processing and determining what the cart should do based on said information. This has the subsystem for the raspberry pi, which is where the central computing is done. It receives data from the lidar as well as sending information to the arduino to determine movement. The arduino is responsible for getting the motor control boards to power the motors.

2.3 Offsite Computing

The offsite computing is responsible for communicating to carts. It has both a subsystem for a front end UI and a backend to facilitate fleet management. The frontend UI allows for wireless control of the cart. While the overall backend facilitates the communication between it the frontend and the raspberry pi (onboard computing), as well as the database.

3 HMETV Layer Subsystems

The HMETV Layer consists of the pieces of the physical product that don't require programming or logic. This layer is entirely hardware, and includes the chassis and other physical/mechanical components of the product.

3.1 Chassis Subsystem

The Chassis acts as the housing and structural support for all the other layers. It also contains an emergency stop button as an additional safety measure for the device

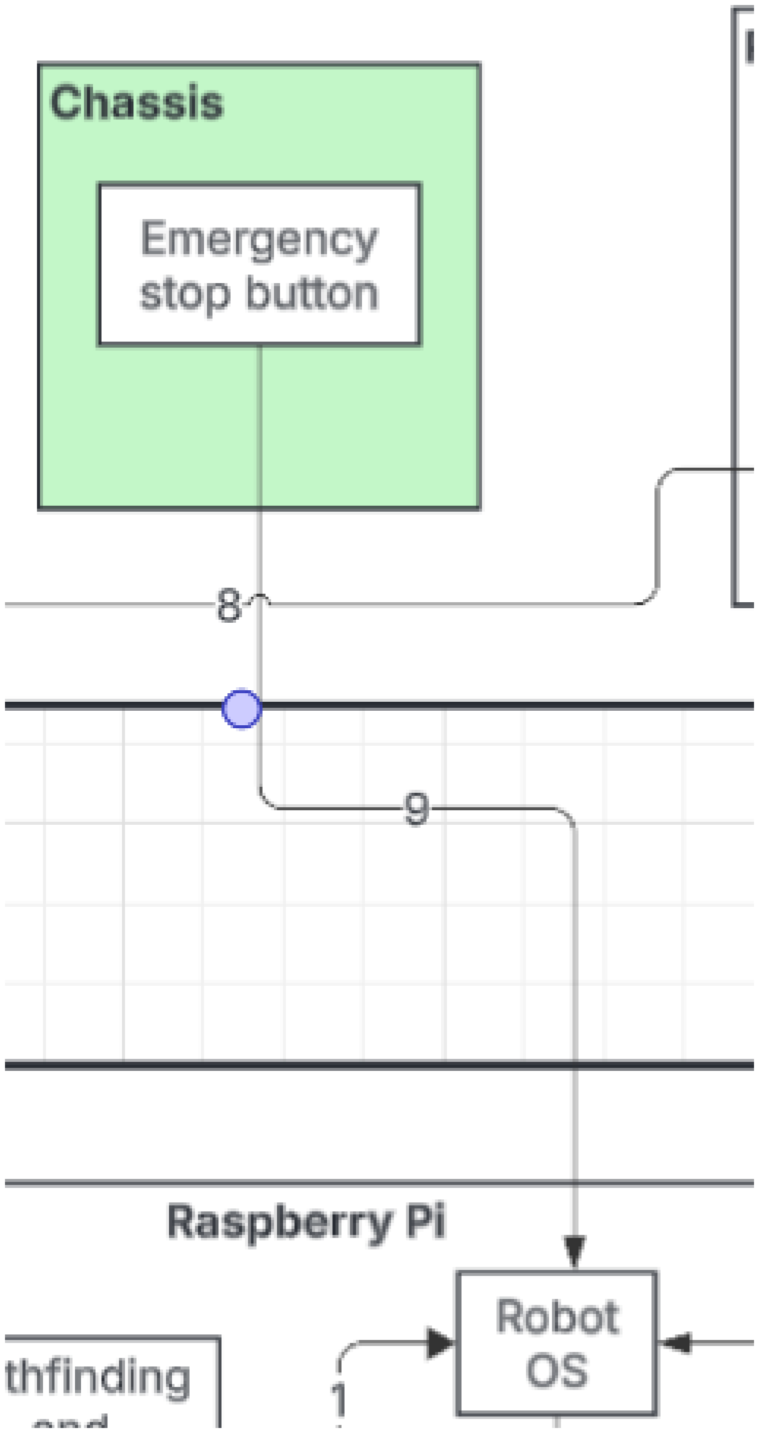

Figure 2: Chassis

3.1.1 Subsystem Hardware

ELAFROS Heavy Duty Plastic Utility Cart 34 x 17 inch

3.2 Drive System (subsystem)

This subsystem is responsible for motor control and locomotion at the hardware level. It consists of the systems motor control boards and their connected motors. The control boards receive commands from the Arduino in the Onboard Computing layer.

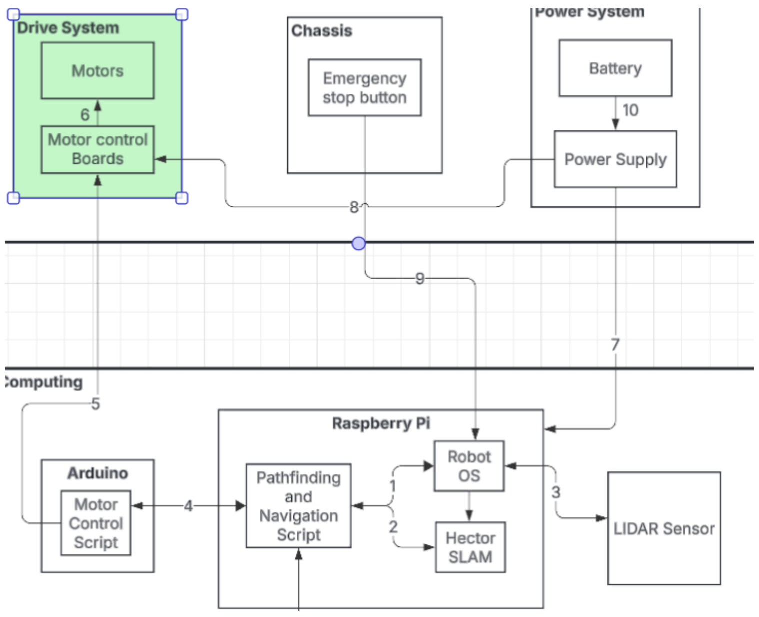

Figure 3: Drive System

3.2.1 Subsystem Hardware

- Sabertooth Motor Control Boards x2
- D63R-2480-160R Motors x4
- Custom printed mounts, x4
- Omnidirectional Wheels x4

3.3 Power Subsystem

The Power System is responsible for supplying electricity to all components of the HMETV and Onboard Computing Layers. It consists of a battery connected to a power distributor

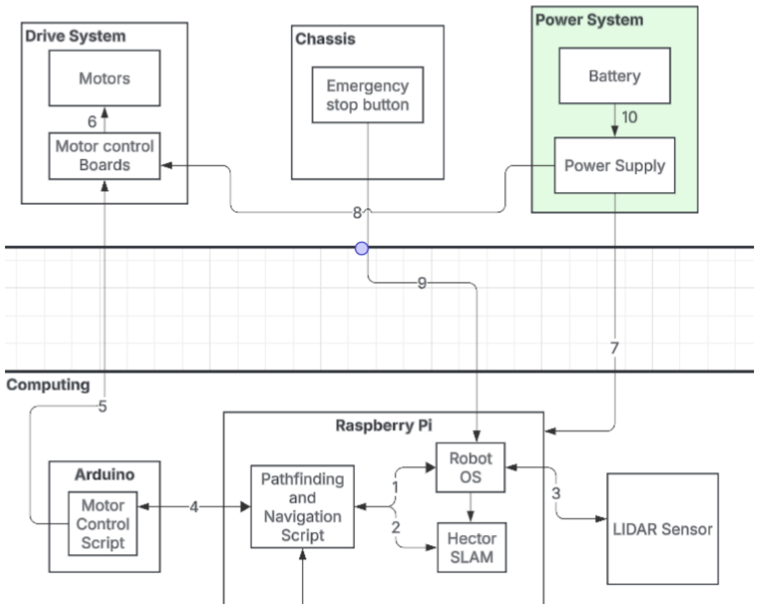

Figure 4: Power Subsystem

3.3.1 Subsystem Hardware

To Be Determined

4 Onboard Computing Layer Subsystems

This layer handles the pathing, sensory data gathering, and remote control aspects of the HMETV. It is essentially the device's "brain". The Onboard Computing Layer consists of 3 devices working together in tandem, the Raspberry Pi takes care of wireless communication and navigation, the Arduino relays commands from the Raspberry Pi to the Motor Control Boards, and the LIDAR Sensor gathers sensory data to send to the Raspberry Pi

4.1 Raspberry Pi Subsystem

This is the core subsystem of this layer. It uses ROS as a central interface to communicate with the LIDAR Sensor, the Arduino, and the Application Backend. Data gathered by the LIDAR sensor is sent to ROS and then to Hector SLAM, which generates a map that will be used by the Pathfinding and Navigation Script. Pathing decisions are output to the Arduino, which will control the motors.

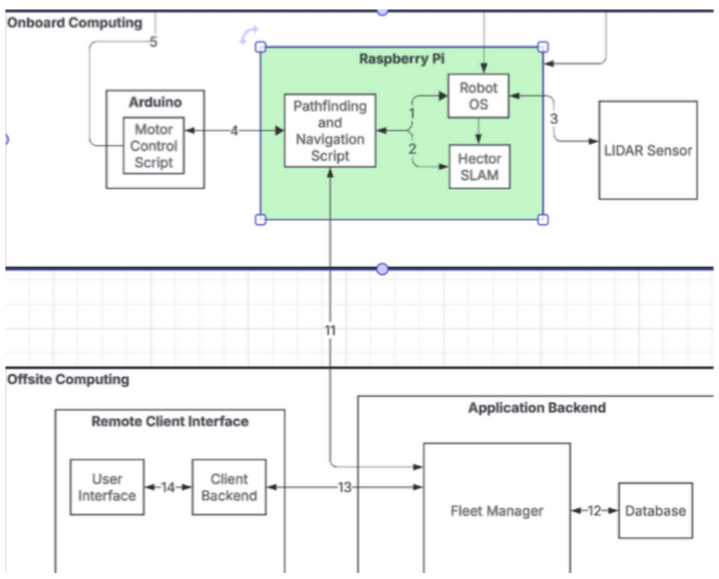

Figure 5: Raspberry Pi Subsystem

4.1.1 Subsystem Hardware

Raspberry Pi 4b, 64-bit, 4GB

4.1.2 Subsystem Operating System

Ubuntu 20.0.4 LTS arm64

4.1.3 Subsystem Software Dependencies

- ROS Noetic Ninjemys
- Hector SLAM: hector_mapping node
- RPLIDAR: rplidarNode
- Python 3.8 or above

4.1.4 Subsystem Programming Languages

- Python 3

- Arduino Script

4.2 Arduino Subsystem

This is a pure hardware subsystem, it does not have an operating system. The Motor Control Script runs as long as the Arduino has power, it receives commands from the Raspberry Pi and translate them into instructions to give to the Motor Control Boards

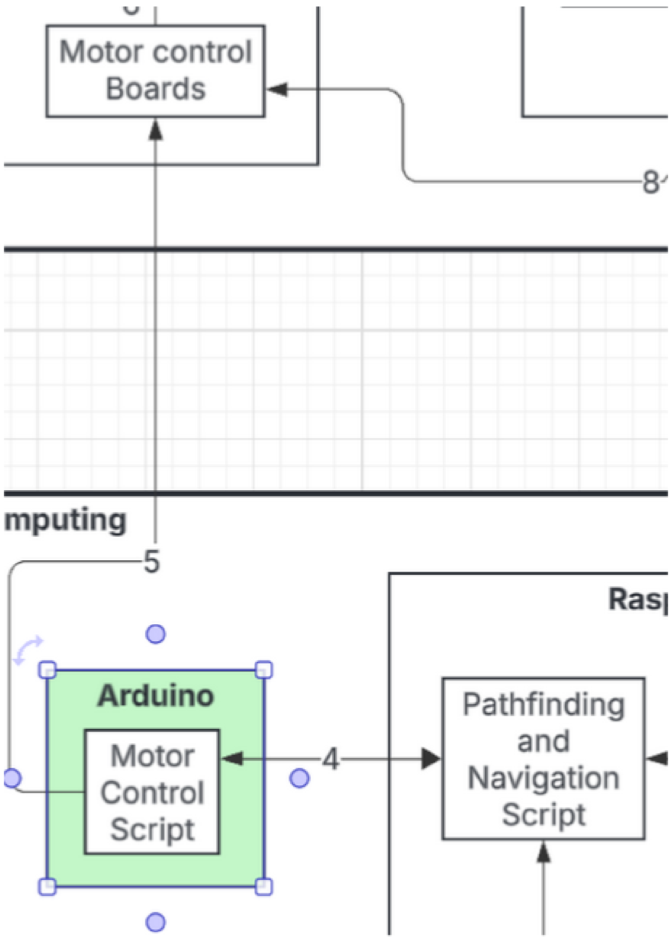

Figure 6: Arduino Subsystem

4.2.1 Subsystem Hardware

Arduino Mega 2560

4.2.2 Subsystem Programming Languages

Arduino Script

4.3 LIDAR Subsystem

This is a pure hardware subsystem, all data gathered is sent back to the Raspberry Pi via USB connection and processed by the Pi.

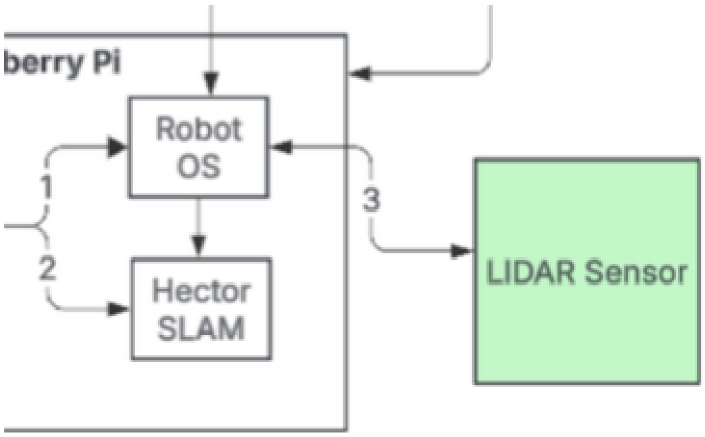

Figure 7: LIDAR Subsystem

4.3.1 Subsystem Hardware

Slamtec A1M8 Lidar Sensor

5 Offsite Computing Layer Subsystems

This section describes the hardware and software components of the offsite computing system layer.

5.1 Layer Software Dependencies

- TCP Networking Capability
- AES Encryption Capability

5.2 Remote Client Interface Subsystem

Descibe at a high level the purpose and basic design of this subsystem. Is it a piece of hardware, a class, a web service, or something else? Note that each of the subsystem items below are meant to be specific to that subystem and not a repeat of anything discussed above for the overall layer.

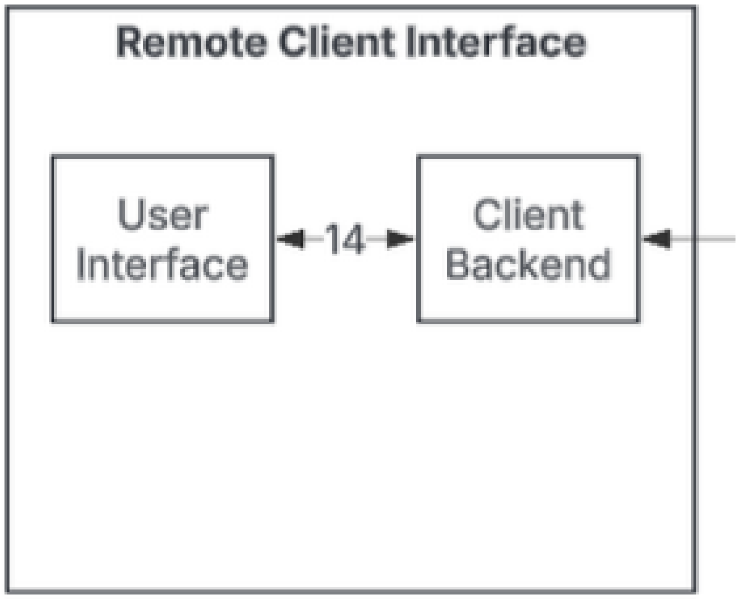

Figure 8: Remote Client Interface Subsystem

5.2.1 Subsystem Hardware

Any device capable of running a web browser

5.2.2 Subsystem Operating System

Any mainstream operating system (Windows, Mac, Linux, Android, IOS, etc.) with a web browser to access the client website

5.2.3 Subsystem Software Dependencies

- Python Flask (for hosting the webpage)
- GUI and backend interface scripts

5.2.4 Subsystem Programming Languages

- HTML
- Javascript

5.2.5 Subsystem Data Structures

- Outgoing packet structure to Application Backend (client id, destination, ordered items, priority, order date and time)
- Incoming packet structure from Application Backend (acknowledgement message, estimated delivery time, cart id)

5.3 Application Backend Subsystem

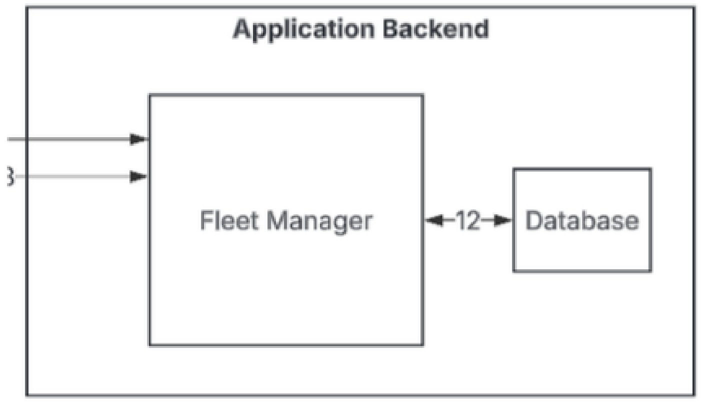

Figure 9: Application Backend Subsystem

5.3.1 Subsystem Hardware

Local Server Computer

5.3.2 Subsystem Operating System

Ubuntu Server 20.0.4 or a similar Linux Distribution

5.3.3 Subsystem Software Dependencies

- MySQL Database
- Python 3.8 or higher

5.3.4 Subsystem Programming Languages

- SQL
- Python 3

5.3.5 Subsystem Data Structures

- Outgoing packet structure to cart (cart id, order id, destination, ordered items)
- Incoming packet structure from client (client id, destination, ordered items, priority, order date and time)
- Outgoing packet structure to client (acknowledgement message, estimated delivery time, cart id)

5.3.6 Subsystem Data Processing

Priority queue for scheduling orders

6 Appendix A

Include any additional documents (CAD design, circuit schematics, etc) as an appendix as necessary.

References