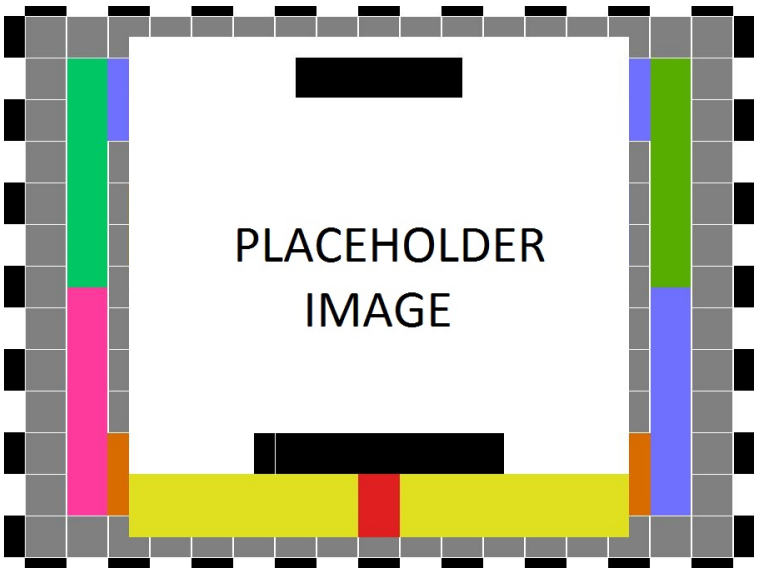
Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4317: Senior Design II
Spring 2016



Team Name
Product Name

Alan Turing
Grace Hopper
John Von Neumann
Ada Lovelace
Charles Babbage

Revision History

Revision	Date	Author(s)	Description
0.1	1.01.2016	GH	document creation
0.2	1.05.2016	AT, GH	complete draft
0.3	1.12.2016	AT, GH	release candidate 1
1.0	1.20.2016	AT, GH, CB	official release
1.1	1.31.2016	AL	added design review requests

CONTENTS

List of Figures

List of Tables

1 Introduction

Your introduction should provide a brief overview of the product concept and a reference to the requirement specification and architectural design documents in 1 or 2 paragraphs. The purpose is to provide the reader with the location of relevant background material that lead to the design details presented in this document.

2 System Overview

This section should reintroduce the full data flow diagram from the architectural specification, and discuss at a high level the purpose of each layer. You do not need to include a subsection for each layer, a 1 - 2 paragraph recap is sufficient.

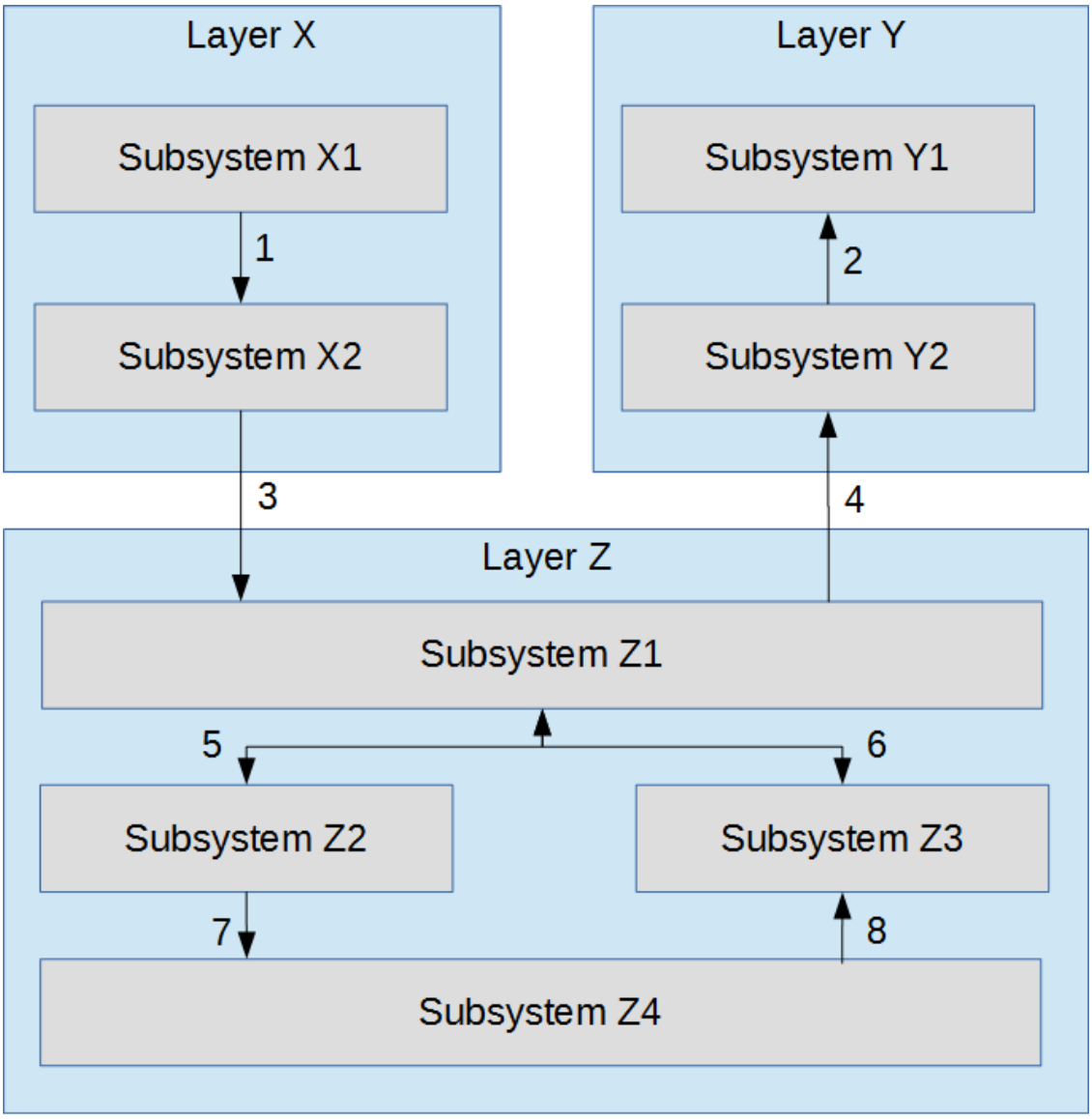


Figure 1: System architecture

3 X Layer Subsystems

In this section, the layer is described in terms of the hardware and software design. Specific implementation details, such as hardware components, programming languages, software dependencies, operating systems, etc. should be discussed. Any unnecessary items can be ommitted (for example, a pure software module without any specific hardware should not include a hardware subsection). The organization, titles, and content of the sections below can be modified as necessary for the project.

3.1 Layer Hardware

A description of any involved hardware components for the layer. For example, if each subsystem is a software process running on an embedded computer, discuss the specifics of that device here. Do not list a hardware component that only exists at the subsystem level (include it in the following sections).

3.2 Layer Operating System

A description of any operating systems required by the layer.

3.3 Layer Software Dependencies

A description of any software dependencies (libraries, frameworks, etc) required by the layer.

3.4 Subsystem 1

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.

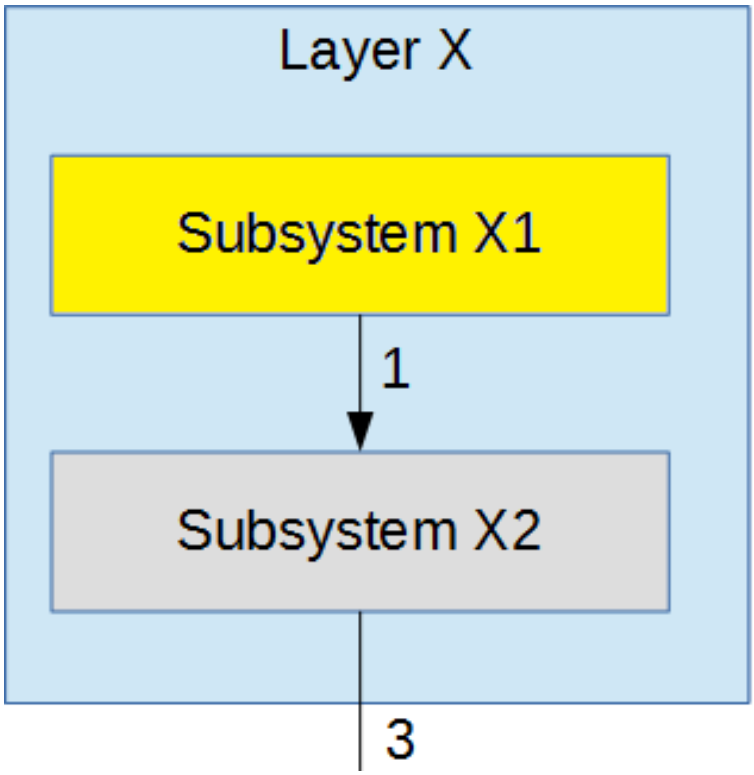


Figure 2: Example subsystem description diagram

3.4.1 Subsystem Hardware

A description of any involved hardware components for the subsystem.

3.4.2 Subsystem Operating System

A description of any operating systems required by the subsystem.

3.4.3 Subsystem Software Dependencies

A description of any software dependencies (libraries, frameworks, design software for mechanical parts or circuits, etc) required by the subsystem.

3.4.4 Subsystem Programming Languages

A description of any programming languages used by the subsystem.

3.4.5 Subsystem Data Structures

A description of any classes or other data structures that are worth discussing for the subsystem. For example, data being transmitted from a microcontroller to a PC via USB should be first be assembled into packets. What is the structure of the packets?

3.4.6 Subsystem Data Processing

A description of any algorithms or processing strategies that are worth discussing for the subsystem. If you are implementing a well-known algorithm, list it. If it is something unique to this project, discuss it in greater detail.

4 Y Layer Subsystems

In this section, the layer is described in terms of the hardware and software design. Specific implementation details, such as hardware components, programming languages, software dependencies, operating systems, etc. should be discussed. Any unnecessary items can be ommitted (for example, a pure software module without any specific hardware should not include a hardware subsection). The organization, titles, and content of the sections below can be modified as necessary for the project.

4.1 Layer Hardware

A description of any involved hardware components for the layer. For example, if each subsystem is a software process running on an embedded computer, discuss the specifics of that device here. Do not list a hardware component that only exists at the subsystem level (include it in the following sections).

4.2 Layer Operating System

A description of any operating systems required by the layer.

4.3 Layer Software Dependencies

A description of any software dependencies (libraries, frameworks, etc) required by the layer.

4.4 Subsystem 1

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.

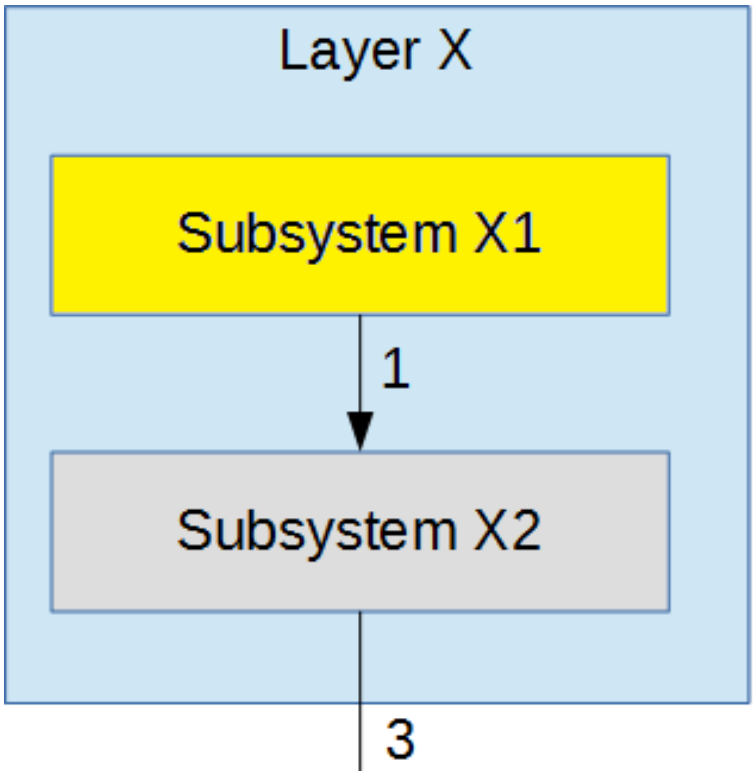


Figure 3: Example subsystem description diagram

4.4.1 Subsystem Hardware

A description of any involved hardware components for the subsystem.

4.4.2 Subsystem Operating System

A description of any operating systems required by the subsystem.

4.4.3 Subsystem Software Dependencies

A description of any software dependencies (libraries, frameworks, design software for mechanical parts or circuits, etc) required by the subsystem.

4.4.4 Subsystem Programming Languages

A description of any programming languages used by the subsystem.

4.4.5 Subsystem Data Structures

A description of any classes or other data structures that are worth discussing for the subsystem. For example, data being transmitted from a microcontroller to a PC via USB should be first be assembled into packets. What is the structure of the packets?

4.4.6 Subsystem Data Processing

A description of any algorithms or processing strategies that are worth discussing for the subsystem. If you are implementing a well-known algorithm, list it. If it is something unique to this project, discuss it in greater detail.

5 Digital Components Layer Subsystems

The digital components layer will consist of 4 devices. The first is a Raspberry PI. This will act as the main computer of the system. It will the majority of the data both from the server, the UI, and the microcontrollers. The other 3 components will all be ESP32 microcontrollers, but they will each serve a different purpose. An ESP32 will receive information from a thermometer sensor and relay it to the heat control ESP32. Based on what the temperature is, the heat control ESP32 will trigger the relay to activate or deactivate the heating element. The final ESP32 will be used to trigger the relay to turn on the pump based on what the message broker instructs it to do.

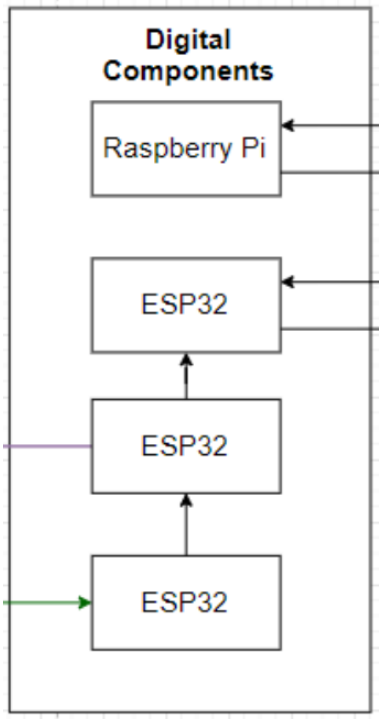


Figure 4: Digital Components Subsystem

5.1 Digital Components Layer Hardware

The hardware used in the digital components subsystem will made up of two different types of electronics. The computer that acts as the arbiter of tasks will be a Raspberry Pi 3b+. The microcontrollers used are ESP32s.

5.2 Digital Components Layer Operating System

The Raspberry Pi 3b+ will run Raspbian OS. There will not be an operating system on the microcontrollers.

5.3 Raspberry Pi Subsystem

The Raspberry PI will host the web server and the MQTT client. The RBP will receive information from the web server about what the user inputed via the UI. It will then take that information and relay it to the microcontrollers using the MQTT protocol. The RBP will then receive sensor data from the microcontrollers through MQTT messages and send store the information in the web server.

5.3.1 Subsystem Hardware

The hardware involved in this layer will consist only of a Raspberry Pi 3b+.

5.3.2 Subsystem Operating System

The operating system used will be Raspbian OS.

5.3.3 Subsystem Software Dependencies

This subsystem will use the Mosquitto MQTT Message Broker.

5.3.4 Subsystem Programming Languages

This subsystem will at the most use a scripting language such as BASH.

5.4 Temperature Sensor Subsystem

The microcontroller will await messages from the message broker. The message broker will recieve information from the thermometer ESP32. If the temperature is too low, the message broker will send a message to this microcontroller to trigger the relay and allow power to flow to the heating element. If the temperature gets too high on the heating element, the message broker will tell this microcontroller to deactivate the heating element.

5.4.1 Subsystem Hardware

The hardware involved in this layer will consist of a microcontroller ESP32 and a temperature sensor MAX6675.

5.4.2 Subsystem Operating System

None will be used for this subsystem.

5.4.3 Subsystem Software Dependencies

Arduino libraries will be used for the temperature sensor.

5.4.4 Subsystem Programming Languages

C++ will be used to program the ESP32.

5.4.5 Subsystem Data Structures

The ESP32 will read data from the temperature sensor, and send it to the Raspberry Pi for a decision to be made. The data will be passed via the MQTT protocol.

5.5 Heat Control Subsystem

The microcontroller will await messages from the message broker. The message broker will recieve information from the thermometer ESP32. If the temperature is too low, the message broker will send a message to this microcontroller to trigger the relay and allow power to flow to the heating element. If the temperature gets too high on the heating element, the message broker will tell this microcontroller to deactivate the heating element.

5.5.1 Subsystem Hardware

The hardware involved will be an ESP32 and a 120V heating element.

5.5.2 Subsystem Operating System

None will be used for this subsystem.

5.5.3 Subsystem Software Dependencies

Arduino MQTT libraries for the ESP32.

5.5.4 Subsystem Programming Languages

C++ will be used to program the ESP32.

5.5.5 Subsystem Data Structures

Commands will be sent to the ESP32 via MQTT messages.

5.6 Pump Control Subsystem

This microcontroller will await messages from the message broker. When the message broker sends it the message to turn on the pump, the microcontroller will trigger the relay to allow power to go to the pump. When the message broker sends the message to turn off the pump, the microcontroller will cease sending power to the relay, causing it to close, and power will not be supplied to the pump.

5.6.1 Subsystem Hardware

The hardware for this subsystem will consist of a single ESP32.

5.6.2 Subsystem Operating System

None will be used for this subsystem.

5.6.3 Subsystem Software Dependencies

Arduino MQTT libraries for the ESP32.

5.6.4 Subsystem Programming Languages

C++ will be used to program the ESP32.

5.6.5 Subsystem Data Structures

Commands will be sent to the ESP32 via MQTT messages.

6 Web Service Layer Subsystems

Since web service API and Database are hosted on Cloud service, no hardware is required. This section will discuss API tool platforms and databases.

6.1 Web Service API

API is an application interface that integrates two applications or systems to support the transfer of data to each other. Mulesoft is a platform where provides many convenient functions to build and deploy API. Using Mulesoft make it easier to connect MySQL.

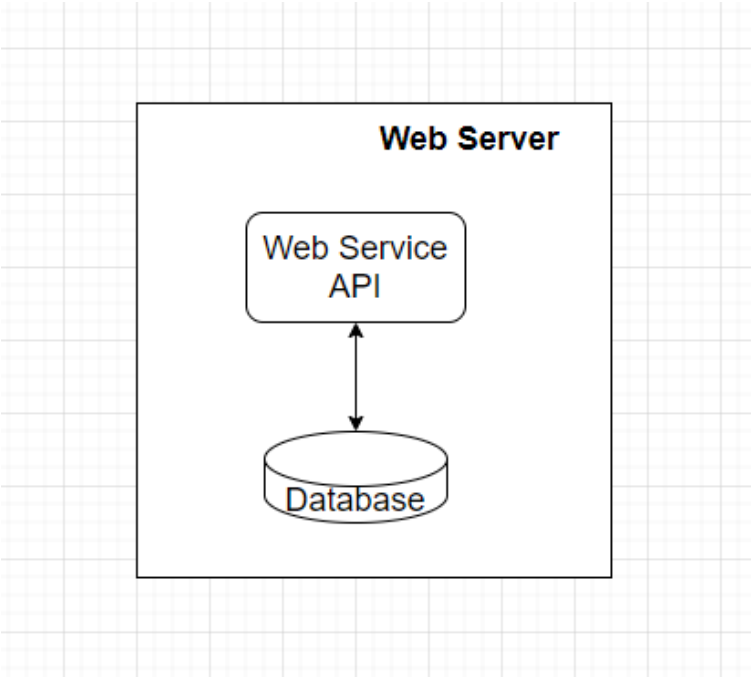


Figure 5: Example subsystem description diagram

6.1.1 Web Service API Software Dependencies

MuleSoft Exchange provides many dependencies and other APIs that are needed to build our API through a connector that prebuilds connectivity to an endpoint.

6.1.2 Web Service API Programming Languages

DataWeave is the MuleSoft expression language for accessing and transforming data that travels through a Mule app.

6.1.3 Web Service API Data Structures

MuleSoft Design Center has a function called API specification which helps to define incoming and output data format and creates automated documentation for the APIs. Through API specification, the general structure of API will be provided.

6.1.4 Web Service API Data Processing

API will listen to HTTP requests and based on a certain path, it will retrieve data from the database and return it to the user.

6.2 Database

The online database allows you to store sensor readings from the ESP32 and read from anywhere in the world by accessing your server domain.

6.2.1 Database Programming Languages

A query is used to communicate with the database.

6.2.2 Database Data Structures

There are some required tables that the database has to have such as temperature for each component and recipe.

6.2.3 Database Data Processing

The database will Run the query for requests and return the data to API.

7 User Interface Layer Subsystems

7.1 User Interface

The website used to control the system will be compatible with major web browsers and will be connected to a database and with the brewing system. Website is verified to work on Chrome for Windows 10, version 91.0.4472.124, and Chrome for MacOS Big Sur, version 91.0.4472.124

7.2 Layer Operating System

Website available on Windows 10 version 21H1 and MacOS Big Sur version 11.4

7.3 Layer Software Dependencies

The user interface components will depend on Mulesoft's Mule software, version 4.3.0

7.4 Website

The website provided to the user will have multiple responsabilities. First, it will display real-time information relevant to the user about the current brew. The website will also display a controls section for the user to interact with the brewing system. The information send to and from the website will be sent to the servers and then to the brewing system utilizing the above mentioned Mulesoft software.

7.4.1 Subsystem Operating System

No specific operating system required.

7.4.2 Subsystem Software Dependencies

The website and UI will utilize the React JS library

7.4.3 Subsystem Programming Languages

JavaScript will be used for creating the website.

8 APPENDIX A

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

References