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

Architecture Design Specification

Smart Grass

Project: Home Irrigation Control System

Team Members:
Belachew Haile-Mariam
Gautam Adhikari
Jeremiah O’Connor
Tung Vo

Last Updated: 1/28/2015

TABLE OF CONTENTS

Document Revision History

Revision Number	Revision Date	Description	Rationale
0.1	11/26/14	ADS First Draft	Initial draft for ADS
0.2	11/28/14	Updates based on team meeting	Layer changes
0.3	11/30/14	Changes from peer feedback	Minor updates
0.4	12/03/14	Formatting and structure changes	Preparations for submission
0.5	12/07/14	Team review changes	Preparations for ADS 1.0
1.0	12/08/14	ADS 1.0 document	Incorporated peer review changes from Patrol Crusaders
1.1	1/27/15	ADS 1.1 revisions	Incorporated presentation notes from the instructor
2.0	1/28/15	ADS 2.0 document	Added additional revisions from the review feedback and internal discussion

List of Figures

List of Tables

1. Introduction

This Architecture Design Specification (ADS) document will describe the high-level design of the Home Irrigation Control System (HICS) project. The ADS will detail the architectural vision of the team as well as provide descriptions for how data flows through each component. It will also introduce the guiding principles that we as a team feel are imperative to how HICS will be designed. In addition, the ADS will also break down each major component of HICS into layers and describe how each layer interacts with the others.

1.1 Product Concept

HICS is an intelligent home irrigation control system that utilizes soil moisture sensors to measure the amount of moisture present in the user’s lawn and use this information to water the lawn in an efficient way. Its purpose is to replace an existing sprinkler control system and allow users to control their home irrigation remotely through a web application that will scale to fit on a computer or mobile device.

HICS uses a central control unit to read soil moisture levels and set watering schedules accordingly by interfacing directly with the users existing irrigation valves. The control unit allows users to remotely schedule or monitor their lawns by communicating with the HICS web application over the internet. Through the web application, users will be granted complete access to all the features that HICS offers while providing an intuitive and easy to use interface.

1.2 Product Scope

HICS is designed to provide an intelligent home irrigation solution that conserves water and allows users to control their home’s irrigation using their own personal computer or mobile device. It is designed to integrate seamlessly into an existing irrigation system and gives users the option to expand their system to support additional valves and sensors.

The target audience for HICS are users with existing sprinkler systems and irrigation/landscaping companies who are looking for a smarter product to sell to their customers. Another major audience for HICS will be people who are concerned about conserving water in their homes. Water conservation enthusiasts will appreciate the numerous options that HICS provides to increase the efficiency of their sprinkler systems.

1.3 Key Requirements

Table 1-1 contains the key requirements for HICS as determined by SmartGrass and our sponsor Nuts and Bolts Hardware. These requirements are considered the most critical when determining how the architecture will be designed.

Req. No.	Requirement Name	Description
3.1	Central Control Unit	The primary relay between the web application, sensors, and irrigation valve(s).
3.2	Soil Moisture Sensors	In-ground sensors that monitor and report soil moisture levels.
3.3	Web Application	The interface that users interact with to monitor and control how their irrigation system operates.
3.4	Water Scheduler	The water scheduler is a remote timer setup that users can set or allow the system to determine for them. The scheduler is responsible for communicating when to start or stop watering cycles.
3.5	Soil Moisture Reports	Soil moisture reports allow users to monitor and view the current and previous soil moisture levels of each zone in their lawn.
3.8	Rain Sensor	The rain sensor detects rainfall and reports to the control unit.
3.14	Temperature Sensor	The temperature sensor monitors local temperate readings and reports to the control unit if freezing conditions are detected.
5.1	Sensor Accuracy	Since the system relies on the information provided by the sensors, accuracy is paramount in ensuring water is not wasted.
5.2	Rain Detection	In the event of rainfall the rain sensor will report to the control unit to stop all watering cycles to reduce water waste.

Table 1-1 Key Requirements Table

2. Meta Architecture

This section describes the architectural vision, guiding principles, assumptions, and tradeoffs that have been factored into designing HICS. The architectural vision will detail the high-level overview of how the system components are separated into layers and how each corresponds to a specific functionality of the overall product. The guiding principles were decided by both the team and our sponsor to ensure the product design stays focused on what we feel are the core principles of HICS. Assumptions and tradeoffs have also been established to guide the overall design process.

2.1 Architectural Vision

There are four main components that are responsible for the major functionality of HICS: sensors, the hardware I/O, an input interface, and a server. This separation allows for each component to be functionally independent and interchangeable if required. For our system, the hardware I/O and input interface are defined as each individual HICS system, and the server is defined by the web application along with the web services. The server is responsible for handling all communication from every HICS system. Each system depends on the server for information regarding the watering schedules and control commands. HICS systems also depend on the server to store all information they receive from the hardware I/O, which relies on the sensor components. Implementing this overall architecture for our design will be a critical step in ensuring all requirements are met.

2.2 Guiding Principles

2.2.1 - Scalability: The HICS product must be scalable to give users the freedom to add or remove valves or sensors as they choose within the limits of the system. This affects what hardware expansion elements are built into the system.

2.2.2 - Ease of Use: The web application interface must be intuitive and user-friendly. This means building the UI interface with simple and easy to understand components like large buttons and descriptive labels. Ease of use also means limiting the number of functions a user must go through to control their system.

2.2.3 - Maintainability: The HICS system should be designed and implemented in such a way that the system requires minimal maintenance after purchase. All rain, temperature, and soil sensors should be replaceable and reinstallation should be possible at the user level. This will mean that the central control unit will have to be built with a plug and play type hardware interface for all sensors.

2.2.4 - Compatibility: Since not all sprinkler systems are designed the same, HICS should be designed to not rely on any proprietary equipment to be installed. The means designing the central control unit to connect directly with the valve wires to ensure the majority of existing sprinkler systems are compatible.

2.3 Assumptions

There are a number of assumptions that have been established through the design process that affect the construction and implementation of HICS. Each assumption has been evaluated and will be used to guide the team’s decision process throughout the design and development phases. All assumptions have been listed below:

- The user will have an existing sprinkler system present at their house.
- The central control unit will be mounted indoors where the existing sprinkler controller was located.
- The user will have internet access at their house and have the ability to connect the central control unit to the internet via a CAT5 cable.
- The user will have a personal computer or mobile device that can access a web browser.
- The user must have the necessary tools (i.e. screws and screw drivers) to mount the central control unit to the wall.

2.4 Tradeoffs

A number of tradeoffs have been considered that influence the design of HICS. Each tradeoff is listed below along with a short description:

2.4.1 – Wireless vs. Wired Sensors: In order to ensure system maintainability HICS will be designed to only support wired sensors. While wireless sensors would be easier in terms of physical installation we feel they also bring a great deal of complexity in regards to battery replacement and connectivity setup. The tradeoff is to use wired sensors to ensure any sensor replacements or additions are easy to install.

2.4.2 - Performance vs. Complexity: Due to the limited amount of time that we have for implementing our prototype, one tradeoff is allowing some performance delay in order to avoid a complex and time extensive approach to the communication between the server and the individual HICS systems. We have decided to have all communication between the HICS systems and server be initiated by each individual systems. This means that the central server will not be able to directly communicate with each HICS system. The complexity of enabling two-way communication between the two would result in a substantial increase in implementation time. The performance tradeoff for this reduced complexity is that now the individual HICS systems must ping the server every two minutes to check for commands or schedule updates. This two minute downtime in between communication means that the system could potentially have up to a two minute delay from when the web service receives a command request to when it is relayed to back to the device.

2.4.3 – Scalable Web Interface vs. Standalone Mobile Application: A major design consideration was to have a scalable web interface instead of a standalone Android or iOS mobile application. In considering our guiding principle for ease of use the team decided that the web application will be built to scale properly for usage on a mobile device in order to maintain the same look and feel across platforms. The web application interface should be consistent so that the user can easily transition from web to mobile usage seamlessly. The UI will be designed to scale in a way that the user can access the same features from a mobile device and have all the UI elements scale properly to fit within the limited screen space.

3. Layer Definitions

This section details the layers that compose HICS, their specifications, and the dataflow between their sub-layers. HICS consists of four layers: The Interface Layer, Server Layer, Hardware I/O Layer, and Sensor Layer. We have two types of input: Hardware input and Software input. The installed sensors provide hardware input whereas the user, through the web application, provides the software input. The output consists of the display from the web application to the user, and output to the irrigation valves that control the water flow to the sprinklers. The features and functions of each layer are described in detail below.

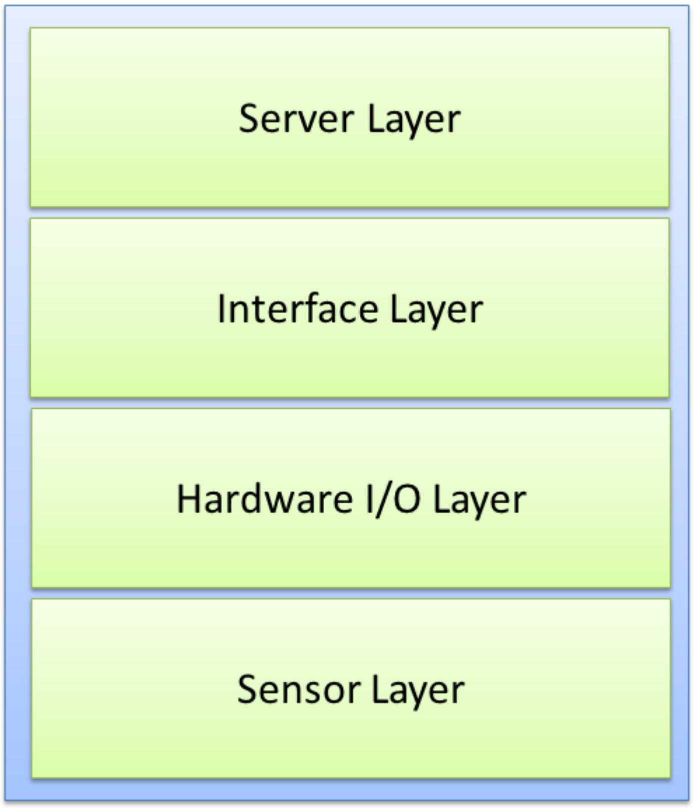

Figure 3-1 Layer Diagram

3.1 Sensor Layer

The Sensor Layer has three sub systems: Rain Sensor, Temperature Sensor, and Soil Sensor(s). The Sensor Layer is responsible sending all the different environment condition readings to the Hardware I/O Layer. Each Sensor Layer subsystem communicates independently of the other subsystems.

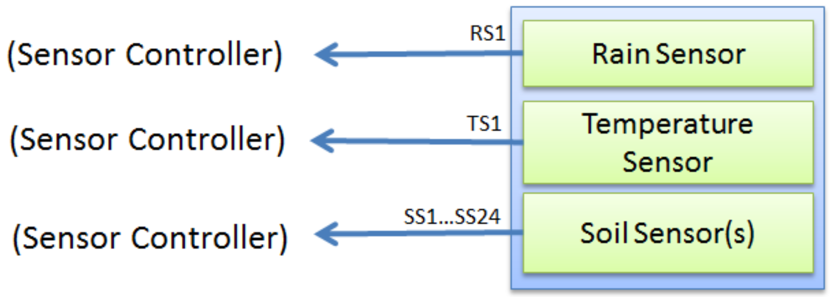

Figure 3-2 Sensor Layer Diagram

3.2 Hardware I/O Layer

Hardware I/O Layer consists of two sub systems: Sensor Controller and the Valve Controller. This layer is responsible for exchanging data between the Interface Layer and the Sensor Layer. The Sensor Controller takes software input from the sensors and sends it to the Data Processing sub-system on the Interface Layer. The Valve Controller takes the input command from the Data Processing to control (switch on/off) the irrigation valves.

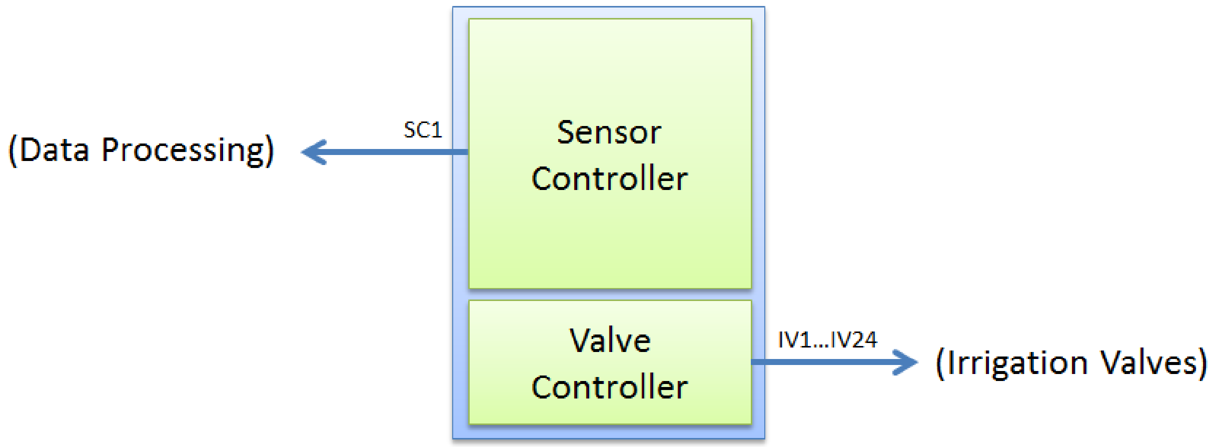

Figure 3-3 HW I/O Layer Diagram

3.3 Interface Layer

The Interface Layer consists of two sub-systems: the Service Caller and Data Processor. The Interface Layer is responsible for processing all communication between the Hardware I/O and the Server Layer. The communication between the two layers consists of input readings from the Hardware I/O Layer as well as control commands from the Server Layer.

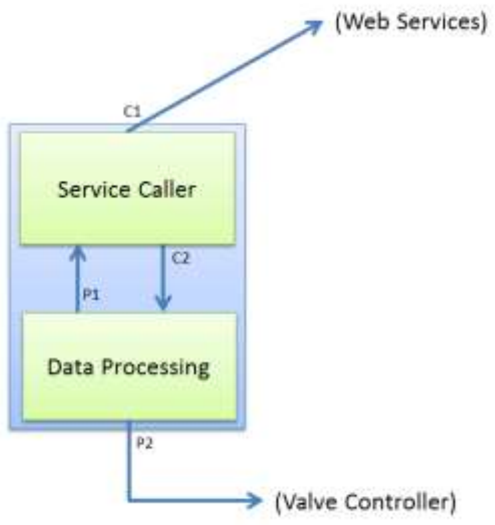

Figure 3-4 Interface Layer Diagram

3.4 Server Layer

The Server Layer has three sub systems: Web Application, Web Services, and Database. The Server Layer handles communication and control between the web application/web services and the Interface Layer. The Server Layer has access to the database and uses the internet to exchange data between all interfaces.

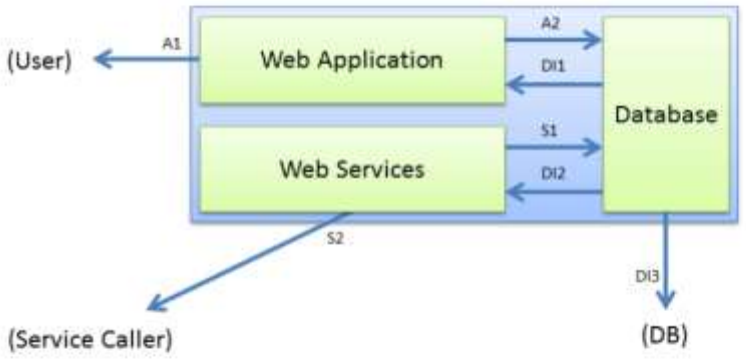

Figure 3-5 Server Layer Diagram

4. Inter-Subsystem Data Flow

4.1 Data Flow Diagram

Figure 4.1 shows the diagram for the overall architecture design and data flow of HICS. Each data flow has been named according to the subsystems from which it flows. Irrigation valves and the user are also represented in the diagram to display how the subsystems communicate with external interfaces.

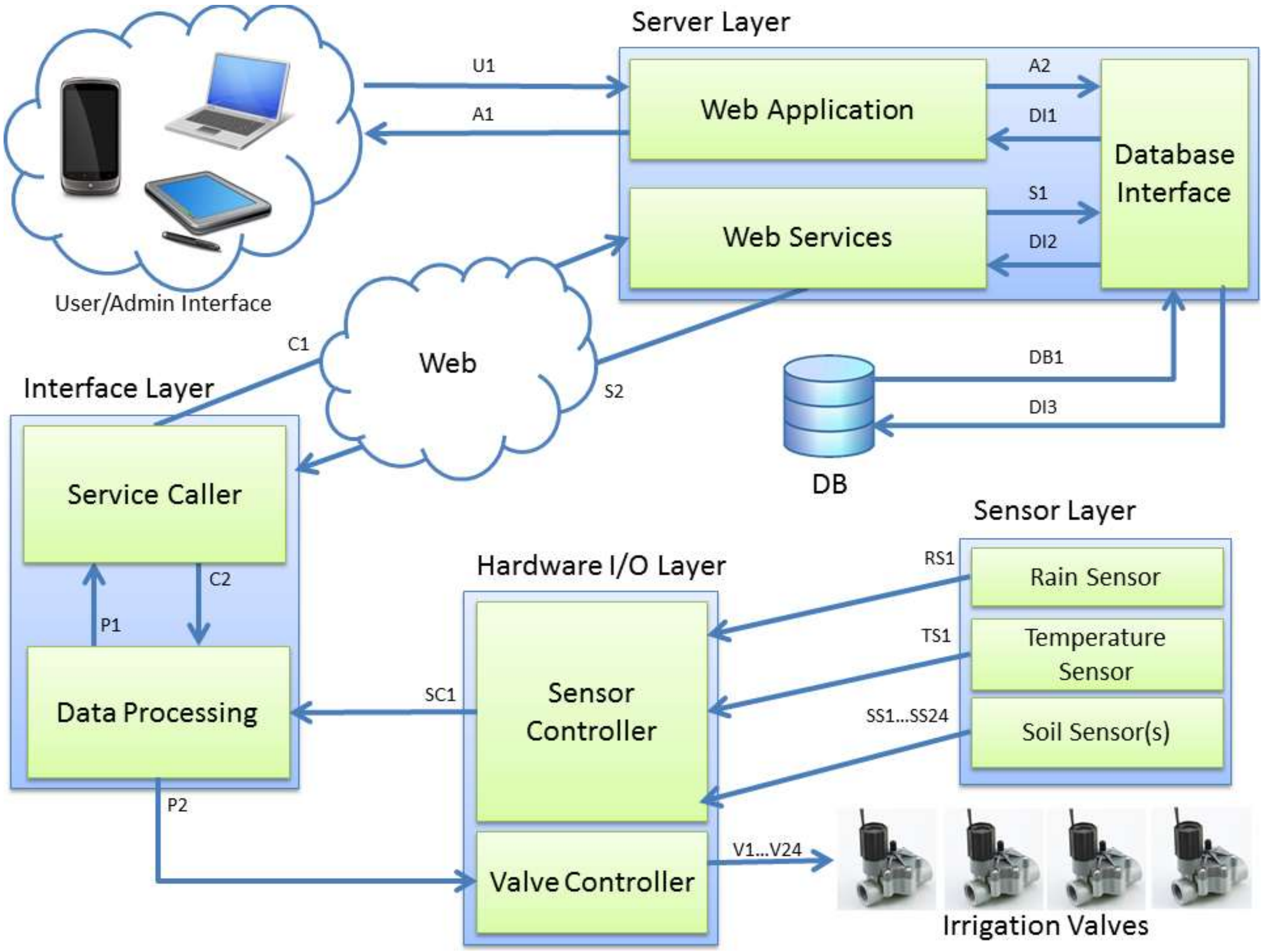

Figure 4-1 Data Flow Diagram

4.2 Data Flow Definitions

Table 4.1 breaks down each labeled data flow and describe how data is communicated to and from each subsystem.

Data Element ID	Description	Source	Sink
U1	User enters input into the web application or interacts with the applications GUI.	User or Admin	Web Application
A1	User requested data is presented through the application GUI on the user's web or mobile browser.	Web Application	User or Admin's web or mobile browser.
A2	Data is received from the user to either update or request information about their system.	Web Application	Database
DB1	Data is queried from the database and passed along to the Database Interface subsystem.	DB	Database Interface
DI1	Requested data from the web application query is located and returned.	Database Interface	Web Application
DI2	Requested data from the web service query is located and returned.	Database Interface	Web Services
DI3	The Database Interface stores data into the DB.	Database Interface	DB
S1	The data requested from the interface is queried or data is sent to the database for storing.	Web Services	Database
S2	The service caller receives a response from a web service with requested information or a response status code.	Web Services	Service Caller
C1	A request sent to the web service for either a valve switch command (on/off), data update response, or the caller sends new sensor readings for storage.	Service Caller	Web Services
C2	Data returned by the web service is relayed to the data processor to check for valve switch commands to turn the water valves on or off.	Service Caller	Data Processing

P1	Data from the sensors is processed and formatted then passed to the service caller.	Data Processing	Service Caller
P2	Response data from the service caller is processed and an operation command is sent to the valve controller.	Data Processing	Valve Controller
SC1	Parsed data from the sensors is sent to the data processing subsystem.	Sensor Controller	Data Processing
IV1+	Control signals are parsed and sent to the irrigation valves to turn them on or off.	Valve Controller	Irrigation Valve(s)
RS1	An alert signal is sent from the rain sensor if a specified amount of rain has been detected.	Rain Sensor	Sensor Controller
TS1	A temperature reading from the temperature sensor is sent to the sensor controller for parsing.	Temperature Sensor	Sensor Controller
SS1+	Soil moisture readings are sent from the soil sensors to the sensor controller for parsing.	Soil Sensor(s)	Sensor Controller

Table 4-1 Data Flow Definitions Table

4.3 Producer-Consumer Relationships

Table 4.2 demonstrates the relationship between producer and consumer by illustrating how data flows between the two. From the matrix we see that each subsystem is relatively balanced in terms of the number of input and output data flows, with the exception of the sensor subsystems. The design of these relationships is to maintain internal data flows within the overall system and only interface externally with the user, database, and irrigation valves.

	Consumer	Irrigation Valve(s)	User/Admin Interface	Rain Sensor	Temperature Sensor	Soil Sensor(s)	Sensor Controller	Valve Controller	Data Processing	Service Caller	Web Services	Database Interface	DB	Web Application
Producer														
Irrigation Valve(s)														
User/Admin Interface														U1
Rain Sensor							RS1							
Temperature Sensor							TS1							
Soil Sensor(s)							SS1							
Sensor Controller									SC1					
Valve Controller		V1												
Data Processing								P2		P1				
Service Caller									C2		C1			
Web Services										S2		S1		
Database Interface											DI2		DI3	DI1
DB												DB1		
Web Application			A1									A2		

Table 4-2 Producer-Consumer Matrix

5. Sensor Layer

The purpose of the Sensor Layer is to gather data from the sensors to monitor different environment conditions. The Sensor Layer consists of a Rain Sensor, a Temperature Sensor, and Soil Moisture Sensors. For each kind of sensor there will be a subsystem responsible for collecting the different environment conditions.

5.1 Rain Sensor

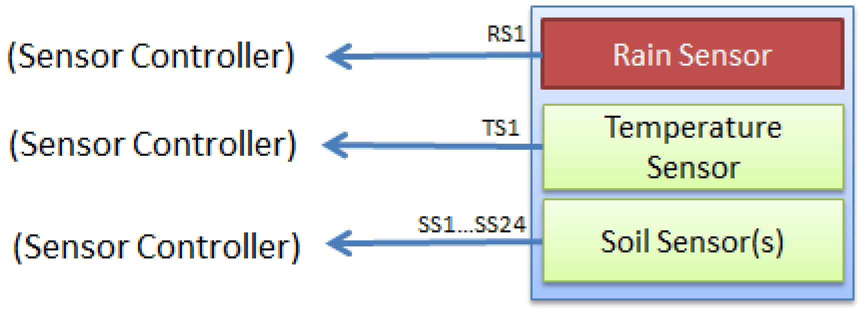

Figure 5-1 Rain Sensor Diagram

5.1.1 - General: The Rain Sensor will detect precipitation amounts by collecting rainfall. This sensor will interface with the Sensor Controller and will send an alert signal to stop all watering cycles if it detects rain.

5.1.2 - Assumptions: The Rain Sensor will only send an output data when it detects rain.

5.1.3 - Responsibilities: The Rain Sensor will send an alert signal when it detects enough rain fall. It is responsible for interfacing with the Sensor Controller to process the alert.

5.1.4 - Inter-Layer Interfaces:

Method	Description	Info Required	Info Returned
sendAlert	This method sends an alert as a signal upon rain detection.	None	None

Table 5-1 Rain Sensor Subsystem Table

5.1.5 - Public Interfaces: This subsystem does not have any external interfaces.

5.2 Temperature Sensor

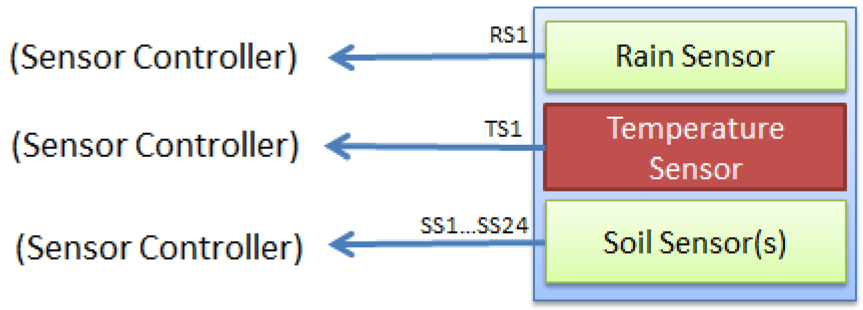

Figure 5-2 Temperature Sensor Diagram

5.2.1 - General: The Temperature Sensor will measure the current temperature of the environment. This sensor will interface with the Sensor Controller to send data about the current temperature in the area.

5.2.2 - Assumptions: The Temperature Sensor will correctly measure the temperature of the environment in Fahrenheit.

5.2.3 - Responsibilities: The Temperature Sensor will measure the temperature of the environment at all times. It is responsible for communicating with the Hardware I/O Layer to monitor temperature readings for freezing conditions.

5.2.4 - Inter-Layer Interfaces:

Method	Description	Info Required	Info Returned
sendInput	This method sends data of the current temperature.	None	None

Table 5-2 Temperature Sensor Subsystem Table

5.2.5 - Public Interfaces: This subsystem does not have any external interfaces.

5.3 Soil Moisture Sensors

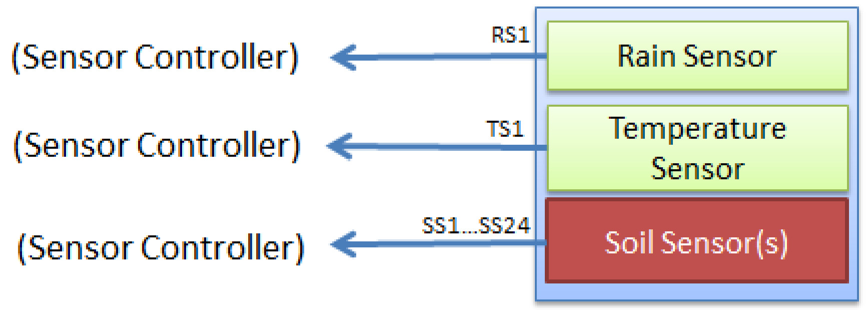

Figure 5-3 Soil Moisture Sensor Diagram

5.3.1 - General: The Soil Moisture Sensor(s) will measure the current soil moisture levels at its corresponding irrigating zone. This sensor will interface with the Sensor Controller and send data consisting of the current soil moisture levels. Each Soil Moisture Sensor is associated with its own irrigation valve.

5.3.2 - Assumptions: The Soil Moisture Sensor(s) will correctly measure the current moisture levels of their corresponding irrigation zones.

5.3.3 - Responsibilities: The Soil Moisture Sensor is responsible for reporting soil moisture levels to the Hardware I/O Layer for processing.

5.3.4 - Inter-Layer Interfaces:

Method	Description	Info Required	Info Returned
sendInput	This method sends data of the current soil moisture levels.	None	None

Table 5-3 Soil Moisture Sensor Subsystem Table

5.3.5 - Public Interfaces: This subsystem does not have any external interfaces.

6. Hardware I/O Layer

The Hardware I/O Layer collects information from the environment sensors and also controls the activities of the irrigation valve(s). The Hardware I/O Layer includes a Sensor Controller subsystem and a Valve Controller subsystem.

6.1 Sensor Controller

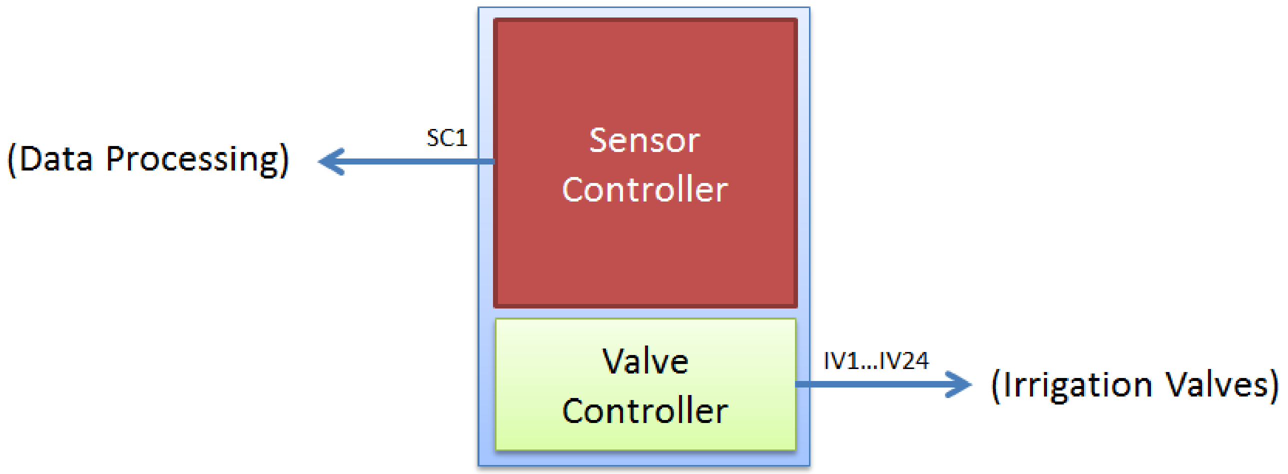

Figure 6-1 Sensor Controller Subsystem Diagram

6.1.1 - General: The Sensor Controller subsystem will interface directly to all sensors, including the rain sensor, soil moisture sensor(s), and temperature sensor. The main component of the Sensor Controller is a microcontroller that reads the signal transmitted from the sensors.

6.1.2 - Assumptions: Each sensor will interface with the Sensor Controller subsystem independently, so that each sensor will generate a unique input that can be differentiated from the inputs of the other sensors.

6.1.3 - Responsibilities: The Sensor Controller is responsible for parsing all input from the soil, temperature, and rain sensors, differentiating the input, and formatting the data to be sent to the Data Processing subsystem.

6.1.4 - Inter-Layer Interfaces:

Method	Description	Info Required	Info Returned
parseInput	This method parses the signal input from the sensors to be formatted and sent to the Data Processing subsystem.	Sensor readings as input data.	None
sendSensorData	This method sends the formatted sensor data to the Data Processing subsystem.	JSON formatted sensor readings as data.	None

Table 6-1 Sensor Controller Subsystem Table

6.1.5 - Public Interfaces: This subsystem does not have any external interfaces.

6.2 Valve Controller

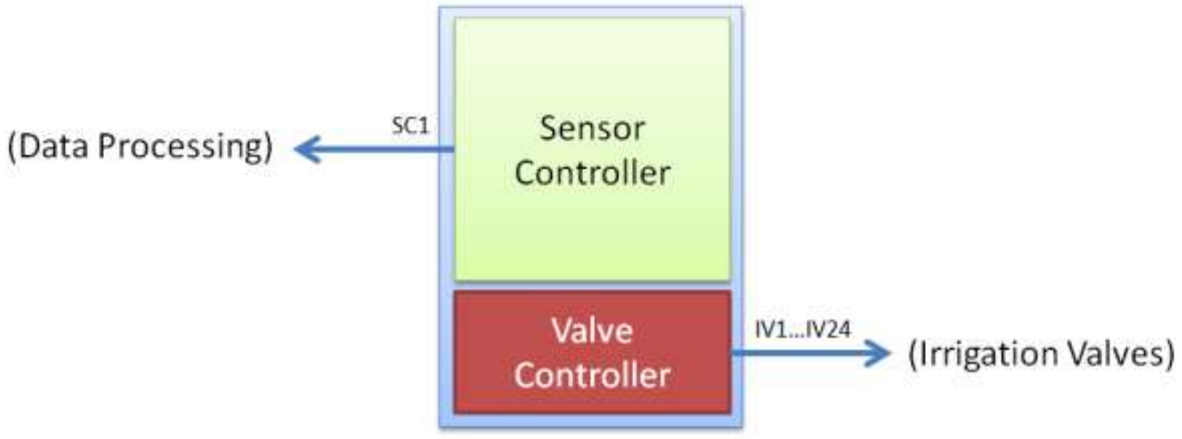

Figure 6-2 Valve Controller Subsystem Diagram

6.2.1 - General: The Valve Controller subsystem will interface directly with the irrigation valve(s). It will also communicate with the Data Processor subsystem to receive control command to turn the valves on or off. The main component of the valve controller is a microcontroller, which will receive the commands from the Data Processor subsystem and use them to control the operation of irrigation valves.

6.2.2 - Assumptions: The Valve Controller system will be able to control every single valve independently.

6.2.3 - Responsibilities: This subsystem will be responsible for receiving commands and using the data to control the irrigation valves. The outputs of this system are signals to turn the valve(s) on or turn off. The input responsibilities of this subsystem are to take the commands sent from the Data Processor subsystem and control the valves accordingly.

6.2.4 - Inter-Layer Interfaces:

Method	Description	Info Required	Info Returned
processCommand	This method processes the control command from the data processing subsystem and determines whether to turn the irrigation valves on or off.	Control command data.	None
turnValveOn	This method sends a control signal to an irrigation valve to turn on.	The valve that is to be turned on.	None
turnValveOff	This method sends a control signal to an irrigation valve to turn off.	The valve that is to be turned off.	None

Table 6-2 Valve Controller Subsystem Table

6.2.5 - Public Interfaces: This subsystem does not have any external interfaces.

7. Interface Layer

The purpose of the Interface Layer is to process and relay all communication between the Hardware I/O and Server Layer to report sensor readings and issue control commands. It consists of two main subsystems—the Service Caller subsystem and the Data Processing subsystem. Each of the subsystems is explained in detail below.

7.1 Service Caller

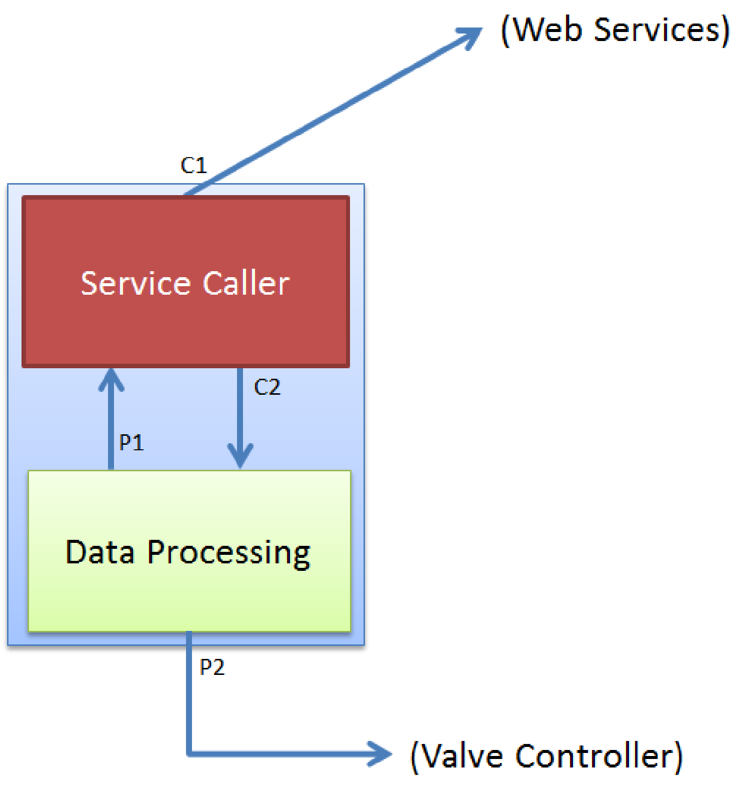

Figure 7-1 Service Caller Subsystem Diagram

7.1.1 - General: The Service Caller subsystem will be a mechanism within the Interface Layer to make web service calls to the Web Services subsystem. It will take the response from the web services and relay it to the Data Processing subsystem.

7.1.2 - Assumptions: When making service calls to the different web services, data will be transferred in JSON format. The web services will expect a JSON request and will reply to the service calls with a JSON response. Also, the Web Services subsystem will not be able to initiate contact with the Interface Layer. The Interface Layer, by means of the

Service Caller, will be able to contact the Web Services subsystem to establish communication.

7.1.3 - Responsibilities: The Service Caller will be responsible for receiving information from the Data Processing subsystem and sending it to the Web Services subsystem in the Server Layer. It will be responsible for initiating contact with the Web Services subsystem, which it will do every two minutes. This subsystem will also be responsible for sending the response received from the web service to the Data Processing subsystem.

7.1.4 - Inter-Layer Interfaces:

Method	Description	Info Required	Info Returned
sendJsonData	This method sends a structure of JSON data to the address of the appropriate online web service.	JSON data to be sent to the web service.	JSON data returned by the web service either with the requested data or a response status code.
relayResponse	This method sends the response data to the Data Processing subsystem.	JSON data from the HTTP request.	None

Table 7-1 Service Caller Subsystem Table

7.1.5 - Public Interfaces: This subsystem does not have any external interfaces.

7.2 Data Processing

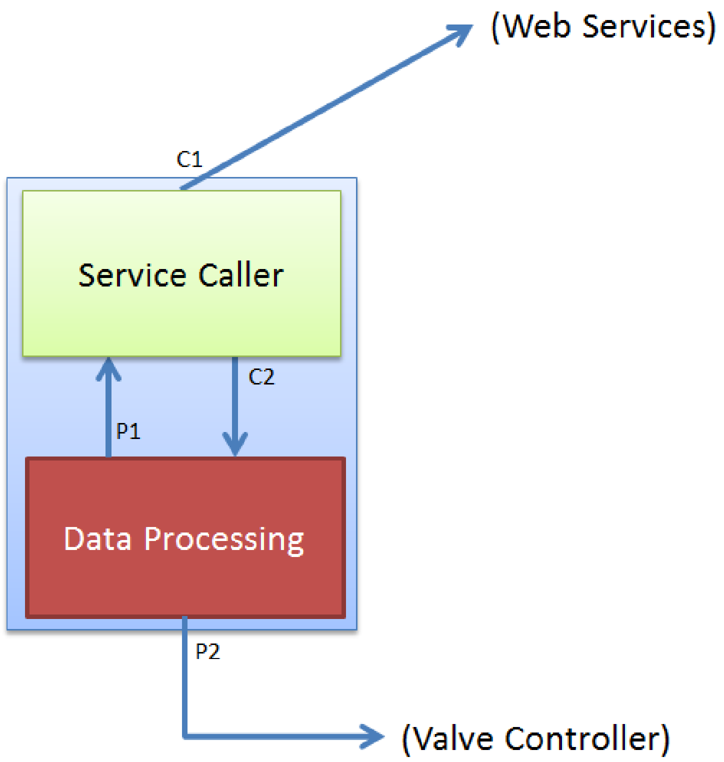

Figure 7-2 Data Processing Subsystem Diagram

7.2.1 - General: The Data Processing subsystem will be used to accept incoming data from the Sensor Controller subsystem, format it into a JSON message, and pass it along to the Service Caller subsystem. It will also be used to do the inverse of this process—accept JSON data from the Service Caller subsystem, parse it into individual valve control signals, and send these signals to the Valve Controller subsystem.

7.2.2 - Assumptions: There will be enough inputs on the Raspberry Pi module to accommodate the Sensor Controller subsystem. There will be enough outputs on the Raspberry Pi module to accommodate the Valve Controller subsystem.

7.2.3 - Responsibilities: The Data Processing subsystem will be responsible for turning sensor input into JSON data and sending it to the Service Caller. It will also be responsible for taking JSON data from the Service Caller subsystem, formatting it, and relaying it to the Valve Controller subsystem.

7.2.4 - Inter-Layer Interfaces:

Method	Description	Info Required	Info Returned
sendValveData	This method processes and outputs valve control data to the Valve Controller subsystem.	Valve control data that has been parsed from a JSON object.	None
processSensorReadings	This method takes sensor reading data from the Sensor Controller and processes/formats the data to send to the Service Caller.	Sensor input as data.	None

Table 7-2 Data Processing Inter-Layer Interfaces Table

7.2.5 - Public Interfaces: This subsystem does not have any external interfaces.

8. Server Layer

The Server Layer is responsible for interfacing with the user to collect input as watering schedules, user settings, and control commands. This information is used to update the database which all interfacing HICS systems rely on for their functionality. The Server Layer is also responsible for collecting data about the different environment conditions from the sensors to store and determine watering schedules. These sensor readings are also displayed to the user via the web application GUI.

8.1 Web Application

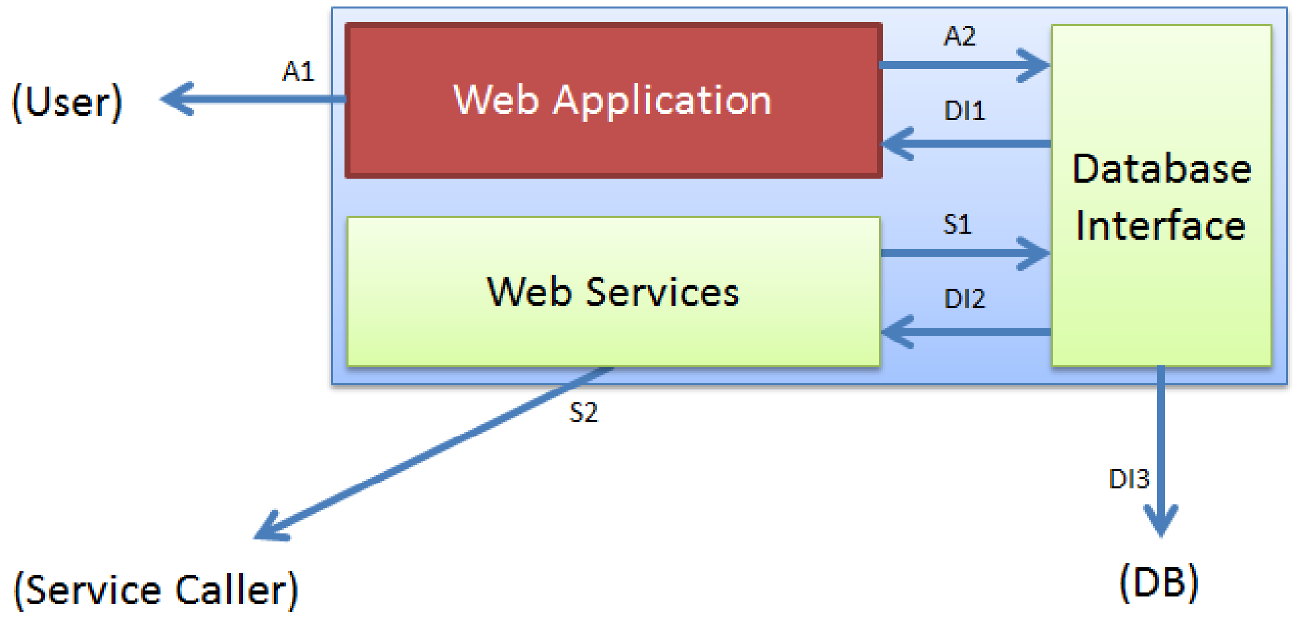

Figure 8-1 Web Application Subsystem Diagram

8.1.1 - General: The Web Application subsystem is the primary interface with which the user communicates with the rest of the system. This subsystem will accept input from the user via a web browser that will be stored. The Web Application subsystem also provides the GUI which is used to present information to the user.

8.1.2 - Assumptions: The Web Application subsystem will be able to handle multiple user requests and inputs simultaneously. The user is also assumed to be accessing the web application from a compatible browser.

8.1.3 - Responsibilities: The Web Application subsystem will be responsible for receiving input from the user and passing that information to the Database subsystem for storage. The Web Application provides a GUI that enables the application to gather input from mouse and keyboard events by the user. The GUI also provides a visual display of data received from the Database subsystem for the user to see.

8.1.4 - Inter-Layer Interfaces: The Web Application subsystem does not interface with another layer.

8.1.5 - Public Interfaces:

Method	Description	Info Required	Info Returned
eventListener	The event listener method waits until an event is triggered from the user in the form of a mouse or keyboard event.	User input data (commands, routing elements, etc.)	None
updateDisplay	The update display method will update the user's page with the corresponding information from the data it received.	Web page location and corresponding model data if required.	Web page with all corresponding scripts, images, and information.

Table 8-1 Public Interfaces for Web Application Subsystem

8.2 Database

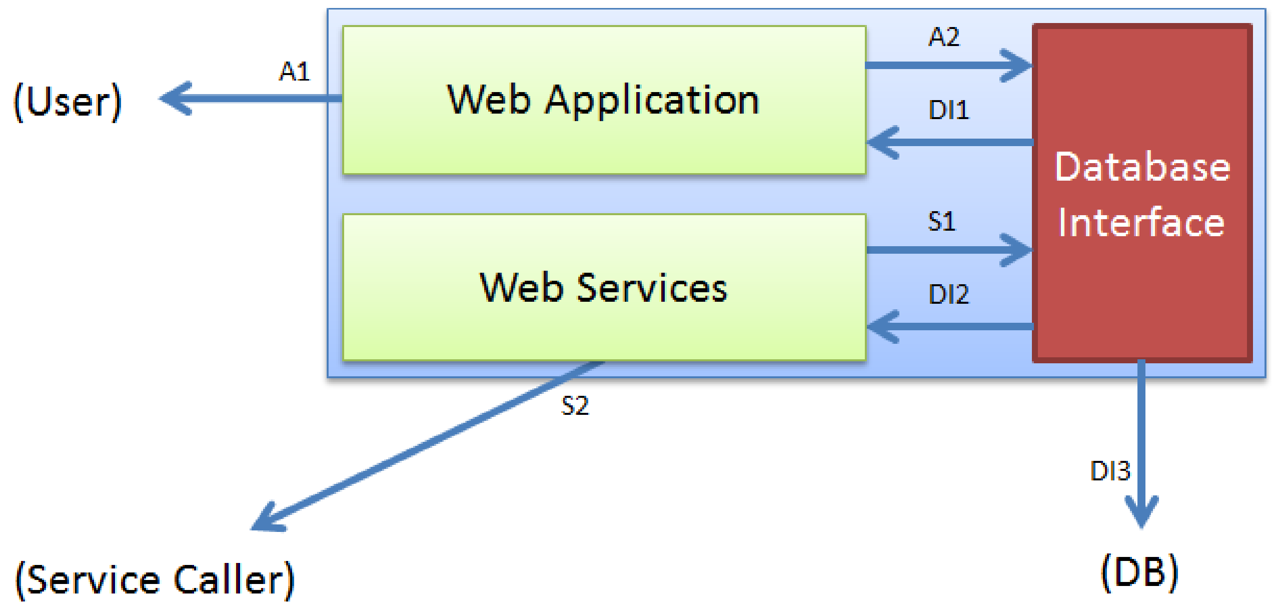

Figure 8-2 Database Subsystem Diagram

8.2.1 - General: The Database subsystem is the central hub for data communication between the server and interface layers. The behavior of each interfacing layer is dictated by the information it sends and receives from the Database subsystem. Data that is communicated to the Database subsystem must update and save correctly to ensure the system behaves as expected.

8.2.2 - Assumptions: All data that is being provided to the database subsystem is verified and validated to avoid data corruption. Also all transactions are serially executed to preserve database consistency.

8.2.3 - Responsibilities: The Database subsystem is responsible for handling all incoming and outgoing data communications from the Web Application and Web Services subsystem. The subsystem ensures that all the data it receives is stored in the database properly. All data requests to the Database subsystem must be provided in its entirety and in a timely manner.

8.2.4 - Inter-Layer Interfaces: The Database subsystem does not interface with another layer.

8.2.5 - Public Interfaces: This subsystem does not have any external interfaces.

8.3 Web Services

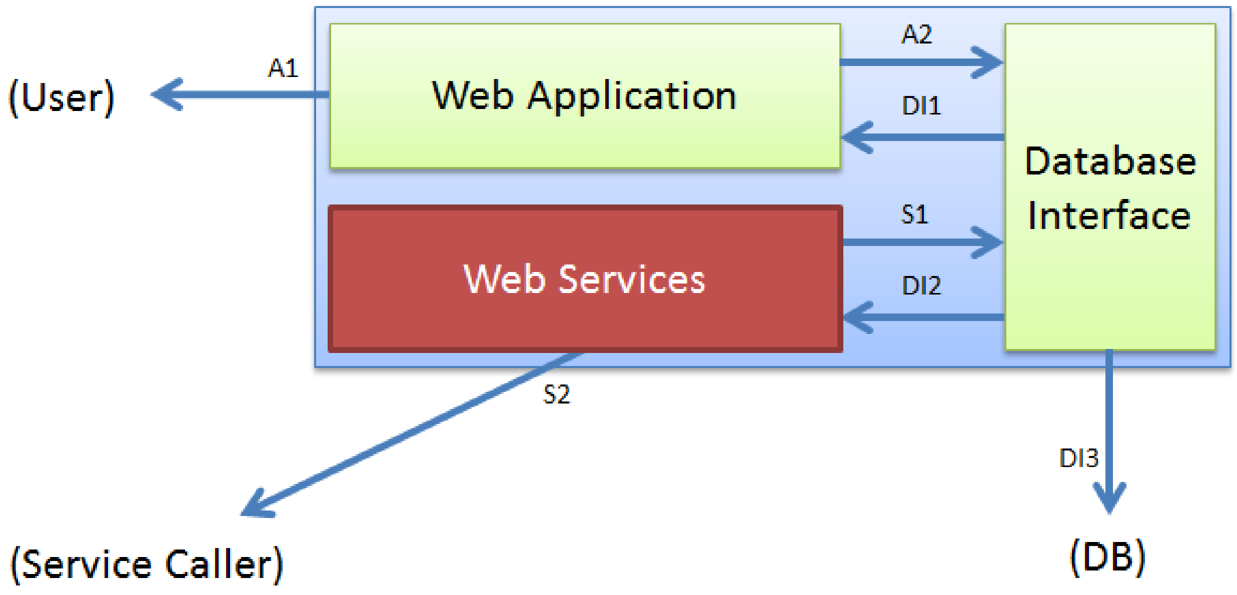

Figure 8-3 Web Services Subsystem Diagram

8.3.1 - General: The Web Services subsystem serves as the communication link between the interfacing systems and the database. The subsystem takes requests from the Service Caller subsystem and either stores or retrieves data depending on the request type.

8.3.2 - Assumptions: All incoming data for storage is JSON formatted. The subsystem also will only be receiving HTTP requests. The Web Services subsystem will also be able to handle multiple requests simultaneously. All service calls must be initiated by the interface devices.

8.3.3 - Responsibilities: The Web Services subsystem is primarily responsible for relaying data to and from the database to the Interface Layer. The Web Services subsystem is required to service all incoming request calls from the interfaces and either store or retrieve data depending on the request type. The service does not contact interfaces directly. It is only responsible for responding to request calls and returning any data if specified.

8.3.4 - Inter-Layer Interfaces:

Method	Description	Info Required	Info Returned
processPost	The services will receive data from a HTTP POST request that must be processed and sent to the Database subsystem.	Data that needs to be stored in JSON format.	Response code (404, 500, etc.)
processGet	The services will process HTTP GET requests from interfaces and must process those requests to query the Database subsystem for data retrieval.	Schedule or command request data in JSON format.	Request data (watering schedules, control commands, etc.)

Table 8-2 Inter-Layer Interfaces for Web Services Subsystem

8.3.5 - Public Interfaces: This subsystem does not have any external interfaces.

9. Requirements Traceability

This section outlines where each key requirement is integrated with respect to the Server, Interface, Hardware I/O, and Sensor layers. Table 9.1 lists each key requirement and the corresponding layer/subsystems that they are mapped to.

9.1 Requirements Traceability Matrix

Requirements No.	Requirements Name	Sensor Layer			Hardware I/O Layer		Interface Layer		Server Layer		
		Rain Sensor	Temperature Sensor	Soil Sensor(s)	Sensor Controller	Valve Controller	Data Processing	Service Caller	Web Services	Database	Web Application
3.1	Central Control Unit					✔	✔	✔			
3.2	Soil Moisture Sensors			✔	✔						
3.3	Web Application				✔				✔		✔
3.4	Water Scheduler	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
3.5	Soil Moisture Reports			✔			✔	✔	✔	✔	✔
3.8	Rain Sensor	✔			✔						
3.14	Temperature Sensor		✔		✔						
5.1	Sensor Accuracy	✔	✔	✔	✔		✔				
5.2	Rain Detection	✔			✔	✔	✔				

Table 9-1 Requirements Traceability Matrix

9.2 Requirements Traceability Analysis

Table 9-1 shows how there can be multiple subsystems from many different layers involved in a single requirement. The Sensor Controller subsystem is by far the most utilized in regards to the key requirements. Since HICS utilizes sensor readings to optimize and control watering schedules it’s apparent that the Hardware I/O Layer and its sub systems play an important role. Many key requirements span across multiple subsystems but the water scheduler is the only one that requires every subsystem for its functionality. The water scheduler is the most critical feature of the HICS system and to take full advantage of it all subsystems must be involved from the Soil Sensors all the way through the Web Application. To uphold all key requirement standards each subsystem and its data flows must be carefully examined to ensure all functions that HICS offers are accounted for before prototyping.

10. Operating System Dependencies

This section considers the operating system dependencies for each layer in the architectural design. Dependencies include external class libraries and files. The system is designed so that each layer performs a well-defined function.

10.1 Hardware Layer

The Hardware I/O Layer includes two controller subsystems: Sensor Controller and Valve Controller. Both subsystems will use microcontrollers for their functionality. The microcontroller functions independently from an operating system.

10.2 Interface Layer

The main component of the Interface Layer is a Raspberry Pi microcomputer, which will run Linux as its operating system. This operating system will support Python and C++ so that programs running on the microcontroller can make service calls to Web Services subsystem and output commands to Arduino microcontrollers in the Hardware I/O Layer.

10.3 Server Layer

There are three subsystems at the Server Layer that each have different operation system dependencies.

10.3.1 - Web Application subsystem: At the Web Application subsystem, a user will use a web browser to interactive with system. For the web browser, the application will work on all major web browsers, including Internet Explorer 9.0+, Mozilla Firefox, and Google Chrome. Web page creation will depend on web technologies such as HTML, CSS, and JavaScript. The subsystem requires the operating system to support .NET Framework 4+.

10.3.2 - Web Services subsystems: The Web Services subsystem will be dependent on the operation system of the hosting service, which will allow API requests and run server side coding languages.

10.3.3 - Database subsystem: The Database subsystem will depend on the hosting service of provider, which will allow the JSON or XML data to be processed or stored correctly. This layer will also be dependent on SQL server and a database driver like JDBC or OLEDB. The operating system must also support Java and .NET standard libraries.

10.4 Sensor Layer

The Sensor Layer includes an electronic rain sensor, temperature sensor, and moisture sensors that are independent from an operating system.

11. Testing Considerations

This section explains the testing considerations relevant to each layer of HICS' architectural design. Each layer in the system can be tested to ensure that it functions properly and safely. Testing considerations also ensures that we are following our guiding principles.

11.1 Overall Considerations

11.1.1 - Ease of Use: HICS should be simple and very easy use. No extra knowledge and tutorial should be required to use it.

11.1.2 - Modularity: Layers of HICS should be stand-alone and modular. Subsystems of a layer should not be dependent on any internal mechanisms of another layer.

11.1.3 - Scalability: HICS should be scalable. The number of hardware components like sensors and valves should be scalable.

11.1.4 - Internal-System Interaction: The layers and subsystems should send/receive data only from a specific layer.

11.1.5 - External-System Interaction: The layers should be interactive to the user. The user should be able to customize the settings of their HICS.

11.2 Hardware I/O Layer

11.2.1 - Modularity: Hardware I/O Layer should be stand-alone and modular. Subsystems of the Hardware I/O Layer should not be dependent on internal mechanism of another layer.

11.2.2 - Internal-System Interaction: The Hardware Layer and its subsystems should send/receive data only from the Sensor and Interface Layers. This layer should not communicate directly with the Server Layer.

11.2.3 - External-System Interaction: The layers should be interactive to the user. This layer will be receiving data sent by the Sensor Layer in the form of rain sensor readings, soil moisture readings, and current temperature readings.

11.3 Interface Layer

11.3.1 - Modularity: The Interface Layer should be stand-alone and modular. Subsystems of this layer should not be dependent on internal mechanism of another layer.

11.3.2 - Internal-System Interaction: The Interface Layer and its subsystems should send/receive data only from the Hardware I/O and Server Layers. This layer should not communicate with the Sensor Layer directly.

11.3.3 - External-System Interaction: The Interface Layer should be interactive. This layer will be receiving data sent by the Hardware I/O Layer, which contains the different sensor readings.

11.4 Server Layer

11.4.1 - Modularity: The Server Layer should be stand-alone and modular. Subsystems of this layer should not be dependent on internal mechanism of another layer.

11.4.2 - Internal-System Interaction: The Server Layer and its subsystems should send/receive data from the Interface Layer only when service caller makes a HTTP request. This layer should not communicate with the Sensor or Hardware I/O Layers.

11.4.3 - External-System Interaction: The Server Layer should be interactive. This layer will be receiving data sent by the Interface Layer through the internet via a HTTP request.

11.5 Sensor Layer

11.5.1 - Modularity: The Sensor Layer should be stand-alone and modular. Subsystem of the Sensor Layer should not be dependent on internal mechanisms of another layer.

11.5.2 - Scalability: Soil moisture sensors and valves should be scalable as desired by the user.

11.5.3 - Internal-System Interaction: The Sensor Layer subsystems should send data only to the Hardware I/O Layer.