Team Name: Honey Badgers

Date: 12/3/2016

Team member name and ID:

John Walter

Thomas Dye

Target accomplishment by the report date:

Basic functionality of the prototype is to be established, including:

- Radio communication between controller and command station
- Sensor readings collected
- Water valve will open and close by command

User interface prototype finished, final polish and testing scheduled

Actual accomplishment by the report date:

- Command Station: Scheduling algorithm and web server were integrated into command station. Command station can now handle incoming data from the controller, run it through the scheduling algorithm, and if determined, pack a watering schedule and send it back to the controller. The web server can now display limited information being produced by the command station.
- Controller: Created wrapper classes for all sensors, controller logic for sending and receiving packets, controller logic for scheduling irrigation, and controller logic for the remote command interpreter.

Analysis of the progress:

We've met the goal of getting basic functionality of the prototype with the controller completely finished, and the command station logic is nearing completion. We are not finished with the user interface, though the document object model exists and most web forms are hooked up. We may be able to polish the interface by writing CSS for the page design, but that is uncertain as of right now because of time constraints. Testing is ongoing.

User interaction with the irrigation controller is handled through a webpage which interfaces with the command station, running on the Raspberry Pi. We first evaluated what we wanted the user experience to be by figuring out what could (or should be controlled). This was further refined by sketching out a user interface, which was then wire framed.

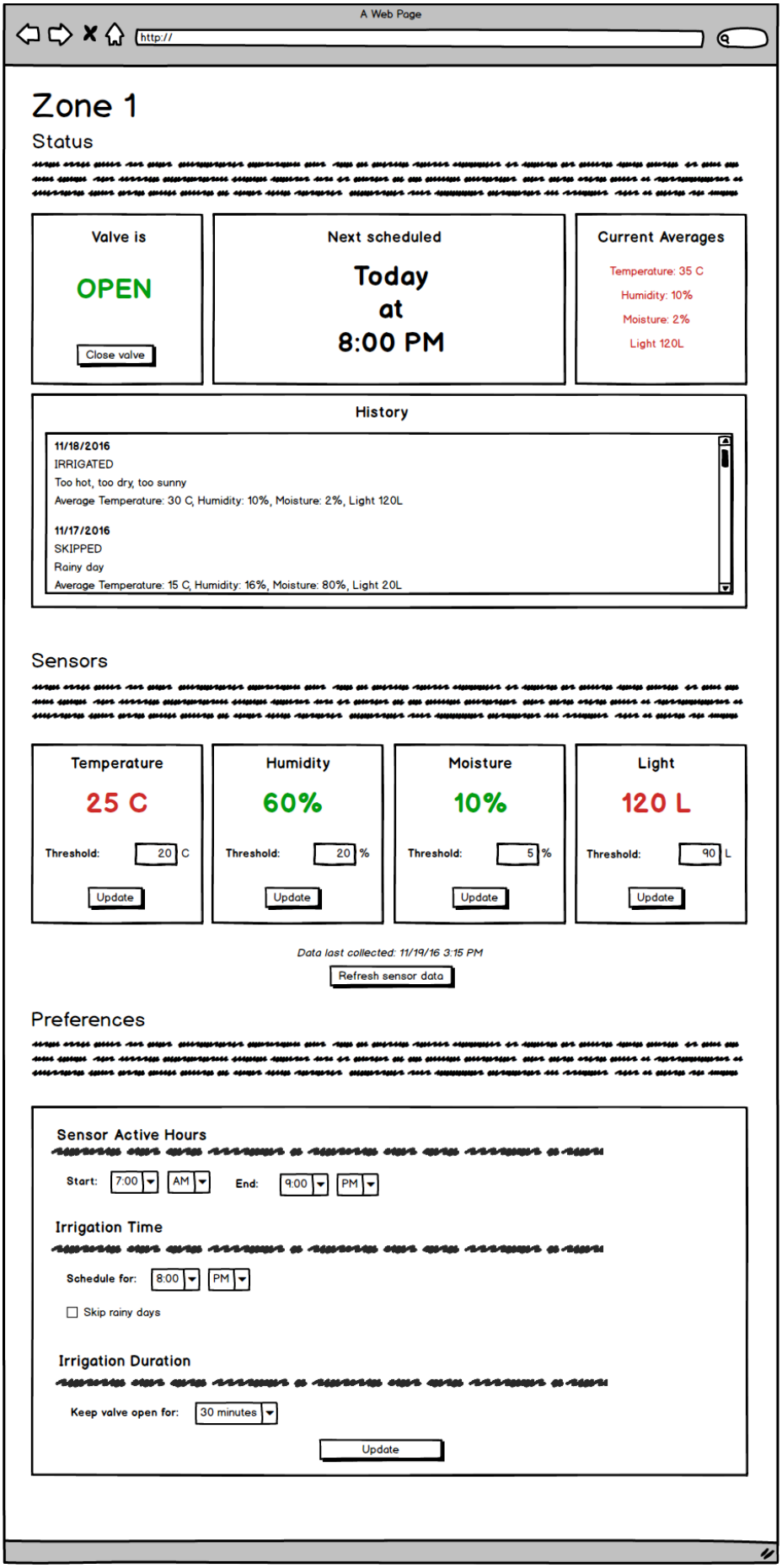

Figure 1: Sprinkler Page Wireframe Mockup

By getting a low fidelity mockup, this allowed us to identify all points of control on the command station that would require coding.

There are three web forms that are responsible for accepting user input valve control, sensor threshold settings, and user preferences, respectively. A user will be able to submit to any of these forms and the command station will be able to use the submissions to influence the behavior of the controller. Currently only displaying of command station data in the webserver has been implemented. However code exists that only needs to be integrated for the command station to be able to gather the form data and change its behavior accordingly.

The Arduino irrigation controller is completely finished, supporting the following radio commands:

Sends:

- Acknowledgement of unrecognized commands
- Acknowledgement of remote command success or failure.
- Status of a zone's valve (open or closed)
- Sensor data for a zone

Receives:

- Command to evaluate and report on a zone's valve state (open or closed)
- Open, close, or toggle the valve state for a zone
- Command to poll sensor data on demand for a zone
- Command to poll all sensors on demand
- Command to set irrigation schedule

To accomplish this, it was first necessary to define a data format and specification guide for radio communication. All payload information is transmitted as character arrays as literal values, i.e., the int 1023 is four bytes long, sent as characters '1', '0', '2', and '3'. Each data element in the payload is delimited with a comma. This method allowed for easy communication between the Arduino controller and the python based command station, and in most cases, makes for less data transmitted than if raw integers were packed into the payload buffer.

Code	**Code, HEX**	**Code, DEC**	**Additional Payload**	**Notes**
C_ACK	0x00	0		Response sent when a command is received but not recognized.
C_SUCCESS	0x01	1		Response sent when a command is received, understood, and is executed.
C_FAILURE	0x02	2		Response sent when a command is received, understood, but does not have valid parameters.
C_GET_VALVE_STATE	0x10	16	Z	Request the valve state for a zone Z. Generates a C_VALVE_DATA response.
C_SET_OPEN_VALVE	0x11	17	Z	Instruct the valve open for a zone Z. Generates a C_SUCCESS response.
C_SET_CLOSE_VALVE	0x12	18	Z	Instruct the valve close for a zone Z. Generates a C_SUCCESS response.
C_SET_TOGGLE_VALVE	0x13	19	Z	Instruct the valve position (open/close) to toggle for a zone Z. Generates a C_SUCCESS response.

Code	Code, HEX	Code, DEC	Additional Payload	Notes
C_SET_TIME	0x14	20	S,M,H,W,D,M,Y	Instruct the RTC to update system time. Parameters are S:seconds (0-59), M:minutes (0-59), H:hours (0-23), W:weekday (1-7), D:day (1-31), M:month (1-12), Y:year (0-99). Generates a C_SUCCESS response.
C_GET_TIME	0x15	21		Request the time from a device. Generates a C_TIME_DATA response.
C_GET_ZONE_SENSORS	0x16	22	Z	Request sensor data for a zone Z. Generates a C_SENSOR_DATA response.
C_GET_ALL_SENSORS	0x17	23		Request sensor data for all zones. Generates a C_SENSOR_DATA response for each defined zone.
C_GET_SCHEDULE	0x18	24	Z	Request the irrigation schedule for a zone Z. Generates a C_SET_SCHEDULE response.
C_SET_SCHEDULE	0x19	25	Z,H,M,D	Instruct the device to begin irrigation at a specified time and duration for a zone Z. Parameters are H:hour (0-23), M:minute (0-59), D:duration(in minutes). Generates a C_SUCCESS response.
C_VALVE_DATA	0x30	48	Z,S	Data for valve state for a zone Z. State S = 0 is closed. State S = 1 is open.
C_TIME_DATA	0x31	49	S,M,H,W,D,M,Y	Data for the time on a device. Parameters are S:seconds (0-59), M:minutes (0-59), H:hours (0-23), W:weekday (1-7), D:day (1-31), M:month (1-12), Y:year (0-99).
C_SENSOR_DATA	0x32	50	Z,L,T,H,M	Data for sensor readings for a zone Z. Parameters are L:light (lux), T:temp (celcius), H:humidity (percentage), M:moisture (level)

Table1: Data Format and Specification

In all cases, the first element in the payload is the identifier code. Additional payload elements are necessary when the code format requires them. For example, C_ACK is delivered by the controller when a command is received by the controller, but is not recognized. There is no additional payload associated with this command. In comparison, C_SENSOR_DATA has the expected format: code, zone, lightSensor, temperature, humidity, and moistureSensor. Over the radio, raw data may be sent such that the command station receives the string: 50,1,138,22,45,1023 which implies it is sensor data from zone 1, where the light brightness is 138 lux, temperature is 22 C, humidity is 45%, and the moisture conductivity level is none (1023 for no water conductivity in dry soil). Table 1 contains the full data format and specifications for radio commands.

Communication reliability was also addressed during this period. While XBee radios have the ability to detect if the local radio failed to transmit or if the remote radio is offline, this does not take into account data integrity and retransmission. The controller will check the quality of transmission for messages sent and received. Sent messages will retransmit a total of five times before failure, with a maximum timeout period of five seconds for each attempt. Received messages automatically generate acknowledgements sent for unknown codes, or whether or not messages were successfully executed. See payload notes from Table 1 for more details.

Our algorithm for determining if irrigation is supposed to occur for a zone is still in place, however for the purpose of a short classroom demo, the scheduling logic has been adjusted so an entire day’s worth of readings are not required.

The system we will be using during the presentation will use a 30 second polling from the controller. Of that, the last four readings will be averaged and then evaluated if it is necessary to begin irrigation. If so, irrigation will be scheduled one minute in the future for a duration of one minute. As we do not have control over temperature or humidity in the room, our hands off demonstration will use soil moisture and light intensity to trigger the algorithm. To evaluate temperature and humidity, it is possible to use the web interface to change thresholds of those two sensor readings to redefine what is considered hot and humid.

System picture:

There are no changes to the physical structure of our setup since the last report.