Alesandra Roger
Rashad Kayed
Ian Hamilton
Alex Mitchell
Yugraj Singh
Evan Hughes

Sprint 1 Report - Smart Irrigation - Team iSlug

What things should we stop doing?

During our first sprint, some key issues caused a slower production of work. The first issue that slowed the team down was that we got bogged down with trying to coordinate tasks and get organized with team meetings. Since our group is doing an IDEASS project, organizing a time to meet with both our sponsor, Kevin, and all of the group members has been difficult. Another key issue that caused some delay is that we didn't have lab access where the team can coordinate and do work while keeping track of the tasks. Ultimately, these issues have been resolved, but initially getting on the same page was a slow process. However, these issues that arose can not be simply washed away by NOT doing particular actions. Each of us as individuals have other obligations as well little control over the authoritative power that allows us lab access. That being said, our resolutions have been to stay highly involved at every opportunity that arises, even when outside influences work against us.

What should we start doing?

During the next sprint we hope to delegate roles ahead of time so the workflow runs more smoothly. Our communication is on target for the most part, but could always be more succinct. We need to clearly delegate roles for each member of the team because we don't want to end up with half of the team doing all the work while the other half are along for the ride. To do this, we simply have to develop a stronger schedule for each member during our SCRUM meetings. As previously stated, getting this project off of the ground has been the most challenging aspect in that we have had a lack of access to essential resources. But now that we have what we need, progressing in our development should be much easier. For the next sprint, we will have much more time for our hands on development. This also means that we will be spending more time together programming and assembling our product, which will ultimately get us more in touch with each other's schedules, strengths, weaknesses, and need for new resources.

What is working well that we should continue to do?

Above we stated that we should role play a bit better and let others know what roles are going to whom ahead of time, but in terms of communicating about our hardware from the IDEASS lab, and working on our project ideas, our team has been on the same page. No one seems to be slacking too much or is unaware of what is going on. We will continue to keep this up by ALWAYS working together when programming/assembling our product. Our rule is that when we work, we have at least two members working so that we can have a stronger perspective. This also helps when informing the other members who had other obligations during a work session during a different scheduled work session or SCRUM meeting about what is currently happening.

What work was completed, and not completed?

Here are the tasks and User Story Listings we wanted to achieve during our first sprint.

1. As a hardware developer of the system, I need to know a system architecture so that I can identify hardware restrictions and required functionality! **(5)**
    1.1. Decide what type of microcontroller we will use (Arduino or BeagleBoard). Consider: cost, flexibility, capabilities, power consumption, documentation/support community. (3 hours)
    1.2. Decide what kind of moisture sensor to use (resistive vs capacitive vs other?) Consider: cost, durability, expected lifetime, power consumption, (3 hours)
    1.3. Estimate what kind of power budget we should stick to allow solar power later. Consider: cost, power output, likely power requirements of the other hardware units (4 hours?)
    1.4. Decide whether the system will be distributed or centralized. (many small isolated probes, or a network of sensors connected to a central controller) (3 hours)
2. As a software developer of the system, I need to know what hardware we are using so that I can write appropriate code. **(1)**
    2.1. Research best programming language to use coupled with the most appropriate programming environment. (Some languages and environments are better depending on type of hardware being used). (1 hour)
    2.2. Break down system into modules or smaller parts (Soil moisture sensors, Solar, Switch, Backend Data Manipulation). (2 hours)
    2.3. Understand how the hardware and software truly interact. (2 hours)
3. As a developer of the system, I need other developers to document their work thoroughly (using comments in code, a manual, a website, or other methods) so that I know what their stuff does. **(1)**
    3.1. Document work while developing code for each fragment that has significant value (Software). (1 hour)
    3.2. Work with at least one other teammate at all times to help build perspectives and strong learning environment (Hardware). (2 hours)
    3.3. Maintain appropriate git(version control system) practices. (< 1 hour)
4. As a farmer, I need the system to be able to detect the moisture level in the soil so that the system knows plants need water. **(8)**
    4.1. Build the initial prototype consisting of the microcontroller and a moisture sensor (2 hours)
    4.2. Write the program which accepts data from the sensor and does any processing necessary to make the data usable for the rest of the system. (2 hours)
    4.3. Test moisture sensors in dry soil and record results. (< 1 hour)
    4.4. Test moisture sensors in moderately wet soil and record results. (< 1 hour)
    4.5. Test moisture sensors in heavily wet soil and record results. (< 1 hour)
    4.6. Compare results to understand when the system needs to water plants. (3 hour)
5. As a tester of the system, I need the system to log data so I can run tests that take a long time. **(5)**
    5.1. Make sure that the hardware extracts data correctly. (1-2 hours)
    5.2. Make sure that the data transferred to software is readable and precise. (1-2 hours)
    5.3. Seek edge cases with experimental/unorthodox behavior with hardware and if necessary, solve issues in efficient manner. (2 hours)
    5.4. Make sure the data are being stored! (< 1hr)

We were able to accomplish a majority of these tasks and took into consideration each user story along the way during Sprint 1. We did not fully accomplish User Stories 4 and 5, due to the fact that we don’t physically have access to the moisture sensors yet, but prototype design and code have been generated. We did not accomplish the User Story regarding the tester of the system fully because we did not have access to any hardware until late in the sprint.

What is our rate of completing work?

User story 1 = Story points 5
User story 2 = Story points 1
User story 3 = Story points 1
User story 4 = Story points 8
User story 5 = Story points 5

Total story point count = 20. The estimates on our story points were pretty accurate for sprint one in terms of difficulty level and how much effort and time it took to accomplish certain tasks.

We put a total of 24 hours into sprint one, each of us doing 4 hours of work. We stayed on target for the most part during sprint one, but we were laying the groundwork for our next sprints which involve more development, design implementation, and testing. These next sprints will be more challenging for this is the first time we have worked with the arduino in this context and a coupled with the sensors, we are dealing with hardware that is variable to take us in varied directions.