The Development of the Agricultural Sensor System has, despite a number of changes, gone smoothly. The original designs supplied to us were initially confusing due to many inconsistencies between the class diagram and sequence diagram. However as we planned and developed different parts of the system we were able to fill in the blanks using the UI designs. As the UI designs acted as an end goal we were able to develop around the functionality it displayed.

The initial planning of the project was undertaken when first presented with the designs. It was clear from the class diagram that there were a number of larger more important classes that would need to be implemented first before much of the system could take shape. In order to manage time for efficiently early on in development we first split the different classes between ourselves. The basic structure of each class was then coded and setup so that as a team we could tackle the key classes and method that effect the most functionality.

We continued to develop the software as a group following an Agile development based sprint approach. Each meeting involved working through a section of the UI storyboard designs, and the functionality needed for it to function whilst deciding on its perceived importance and development man hours. This development could then be continued as a group, formulating the important classes first. Potentially this may have slowed down our production as development outside of the team meetings may have helped speed up the process. However while focusing on the key underlying functionality we thought it would best be to develop as a team in order to ensure each member understood the processes in the program.

We used a set of units tests along with UI test automation, which helped us find problems that occurred during development. The tests would fail, when previously they passed, alerting us of recent code changes which caused the problems. Using this Test-Driven Agile methodology it was much easier for us to notice and solve bugs early in development.

The designs supplied for the development of the agricultural sensor system required a number of different changes in order for the system to work effectively. Initially when developing the class diagrams functions it may have been easier to start developing the sequence diagrams and the UI. The class diagram lacked descriptions as to the purpose of the methods and we found ourselves trying to find a place where they could be used. However the system itself allowed us to properly implement different design patterns, such as singleton, something that would have been useful to bear in mind at the start of development.

In most cases the changes made were obvious and minor. Such as the addition of a Vector<SensorData> to the Historical Data class. Without the vector the historical data would have had no data to return. Many inconsistencies were missing parameters in methods differing between the class and sequence diagrams. AddSensor in the FieldStation took 3 Strings used in the Sensor constructor however the initial constructor only took 2 strings, missing one for the units. These changes would have gone unnoticed had we worked following the sequence diagrams, however these small inconsistencies delayed production as we analysed the documents looking for a way the functions could perform without the missing attributes. Though in analysing the documents in further detail we have had to make very few major changes, instead the system matches the sequence diagrams and UI almost exactly.

However larger changes included the addition of an extra Map<> to the servers data. This changed ‘Map<String, HistoricalData> data’ to ‘Map<String, Map<String, HistoricalData>> data’. Originally this data was to contain the historical data relating to the FieldStation Id however as we found from implementing the UI there was a requirement for HistoricalData relating to individual sensors as well. This alteration changes the behaviour of different methods in the Report Class as now data could be returned from either individual Sensors or FieldStations as a whole.

Further changes made to the original design documents were mainly additions to the features presented. For example the original documents lacked a description of how the users register. In order for the system to work effectively we had to design a new register page and add a button to the original login page. Equally when adding a new Sensor a number of fields to add important sensor attributes, such as threshold and upper limit, were not included without as such the system would not work correctly.

Further to this the original UI displayed ‘Detail’ buttons for both sensors and field stations. Without any knowledge of what to present to the user we had to design our own dialogs deciding what information to display. Potentially before starting development it would have been beneficial to first plan out each aspect of the UI, and how it relates to the back end classes. This would have allowed us to develop the UI first using a front to back development approach. However that said, our development sprints focused on individual storyboard screens allowing us to ensure the functionality for those screens worked effectively before moving on.

The evaluation part of the project could have been made easier with a larger cohort of volunteers. Due to this lack of Users we had to scale our method accordingly while taking focusing in greater detail on the little data we could collect. Were it to be repeated, longer time should be spent on evaluation ensuring more volunteers and more tasks presented to them. This increase may have allowed for further suggestions to be made on user interface changes covering deeper areas in the system. Further to this it would have been beneficial to alter our original evaluation method to collect more in depth user interface evaluation metrics. Were a method such as eye tracking heat maps used we may have been able to see more clearly the UI components that slowed the users process down.