2015 Citrus Circuits Electronic Scouting System

1678 App Programming Team

May 21, 2015

Contents

Chapter 1

System Design

1.1 Introduction

1.1.1 Goals

The goal of Citrus Circuits' scouting system is to allow for seamless data input, processing, and visualization which can be used for both match strategy and alliance selection.

1.1.2 Why Electronic?

Citrus Circuits used a paper and pencil system in 2012. However, we found that, while data was effectively collected, the difficulty of processing and visualizing it an a timely manner made the system highly impractical. The advantages of an electronic scouting system is that processing can be automated and the use of devices to visualize the data allows for a far faster, more flexible, and practical scouting system that can be used for multiple applications.

1.1.3 Why Mobile?

All of the applications in the 2015 Citrus Circuits scouting system run on mobile devices (i.e., phones and tablets). This is done for two reasons:

1. The ability to effectively transport and run the system at a variety of competitions, all with different seating, environments, etc.

2. The widespread nature of personal mobile devices allows the team to save costs, as the devices of team members can be used without needing to purchase specialized equipment.

1.1.4 A Major Problem: No Internet

Unfortunately, the competition rules specifically state that teams are forbidden from setting up WiFi hotspots in the stands[1]. As we are determined to build a completely legal system, this is a problem for data transfer. This problem can be solved through the use of wires or flash drives to transfer the data, but this makes the system less portable, more difficult to set up, and less convenient for scouts. Cellular data is another solution, but since we could not afford to purchase eight tablets with cellular data, we could not implement this solution.

[1] Rule T4: Teams may not set up their own 802.11a/b/g/n/ac $2.4GHzor5GHz$ wireless communication (e.g. access points or ad-hoc networks) in the venue.

In order to be able to transfer data without the need for flash drives or wires, we use Bluetooth to transmit data to tablets which have cellular data[2] and upload the data. This allows the system to be more portable and easier to set up.

Unfortunately, Bluetooth is inconsistent in its data transfer, so all of the data is saved before it is transmitted to ensure that we never lose any data.

1.1.5 Past Systems

Citrus Circuits created its first scouting system in 2013. Since then, the systems have become progressively more involved and complicated. For information on these past systems, see the 2013 and 2014 white papers we have published.

2013 White Paper: `http://www.chiefdelphi.com/media/papers/download/3747`

2014 White Paper: `http://www.chiefdelphi.com/media/papers/download/4122`

1.2 Overview

1.2.1 Equipment and Costs

Nexus 7 + T-Mobile SIM ($350) × 2 = $700
`https://play.google.com/store/devices/details/Nexus_7_32GB_Black_Wi_Fi_Mobile_data_Unlocked_T_Mo?id=nexus_7_32gb_2013_lte_tmo`

Nexus 7 ($165) × 6 = $990
`http://www.amazon.com/Nexus-Google-7-Inch-Black-Tablet/dp/B00DVFLJDS`

MacBook Pro ($1099, but can be bought from other sources for cheaper) × 1 = $1099
`http://store.apple.com/us/buy-mac/macbook-pro`

Approximate Total: $3000

These materials do not include the phones used for processing and viewing the data. Due to the widespread nature of mobile devices, it is highly likely that multiple team members would be willing to allow their phones to be used as part of the scouting system while at competition.

The MacBook Pro and iOS Developer Program are only necessary for iOS programming. If the system was implemented solely in Android, the cost of the system could be significantly reduced. However, a programming computer is still necessary to implement the system.

1.2.2 Necessary Capabilities

Android and iOS programming experience.

iOS Training

Follow this excellent tutorial provided by Stanford on programming iOS 8 apps: `https://itunes.apple.com/bj/course/developing-ios-8-apps-swift/id961180099`

[2] Purchased from T-Mobile.

Android Training

There are a variety of less well-organized resources for Android online. For an introduction, I would recommend Android Developer website's official tutorial: `https://developer.android.com/training/index.html`

1.2.3 Application Overview

The entire scouting system consists of seven different mobile applications. There are four input apps, one processing app, and two viewing apps.

- **Input**

  Scout
  Android: Collects easily determined data concerning robot performance, such as number of totes stacked and maximum tote height. Scouts every qualifying match.

  Super Scout
  Android: Collects more subjective data, such as driver ability and stack placing security. Generally an experienced technical team member. Scouts every qualifying match.

  Pit Scout
  Android: Collects images and robot design data by going through the pits and talking with every team. Generally a person with good social and photography skills.

  Cheesecake App
  Android: A specialized app this year, designed for the person who scouted for the ability to *cheesecake* robots. Walked through the pits and talked to every team and examined if the team is willing to cheesecake and if their robot is compatible with the cheesecake mechanism.

- **Processing**

  Server
  iOS: Received the data from the scouts, processed it, and outputted it to the viewers.

- **Viewing**

  iOS Viewer
  iOS: Offered the capability to view the match schedule, predicted match scores, actual match scores, both raw and processed data, graphs, team images, the ability to cheesecake robots, and the ability to print team cards which could be used at the alliance selection meeting.

  Android Viewer
  Android: Offered the capability to view the match schedule, predicted match scores, actual match scores, both raw and processed data, graphs, the ability to cheesecake robots, and team images.

1.2.4 Data Distribution and Storage

Requirements

The requirements for data distribution were that it must be live updating, automatic, and able to function offline. In order to accomplish these goals, two main infrastructures were used: Dropbox Sync API (`https://www.dropbox.com/developers/sync`) and Realm (`https://realm.io`).

Dropbox

Our team first began using the Dropbox Sync API in our 2014 Scouting System. Due to our generally positive experience with it in 2014, we decided to use it again. The Sync API allows for simple, automated, live syncing of files through a single folder in Dropbox.

Realm

Realm is a simple, fast relational database for iOS, Android, and OS X. It removes a significant amount of boilerplate code and allows for simple, elegant data entry and querying.

Basic Principles

Data is communicated between the apps through Dropbox in the forms of *change packets*[3] and the database file. For a visualization of the data transfer system, see Figure 1. The scout transfers a change packet to the super scout of the same color via Bluetooth. The super scout then uploads these change packets to a folder in Dropbox. In addition, the pit scout and cheesecake app also upload their change packets to this location. Through the use of listeners/observers, the server detects when there are change packets. After waiting for file uploading to be completed, the data from the change packets is processed and inserted into the database. All of the calculated metrics[4] are then recalculated and the database is uploaded to Dropbox in a different folder. The viewing apps observe this folder and, when the database is updated, they download this database. Through the use of observers, the system is completely automated. Also, by allowing only the server to modify the database, the danger of conflicted copies of the database is avoided. The pit scout uploads pictures to a different location, where each team has its own folder. When a viewing app needs a picture, it downloads the pictures from this location.

[3] Change packets consist of small JSON files that contain the data that the server must change in the database.

[4] The calculated metrics combine the raw data into an evaluation of a robot's ability in a certain category. For example, some calculated metrics from the 2015 season included stacking ability, first pick ability, recycling container reliability, and average number of maximum height stacks.

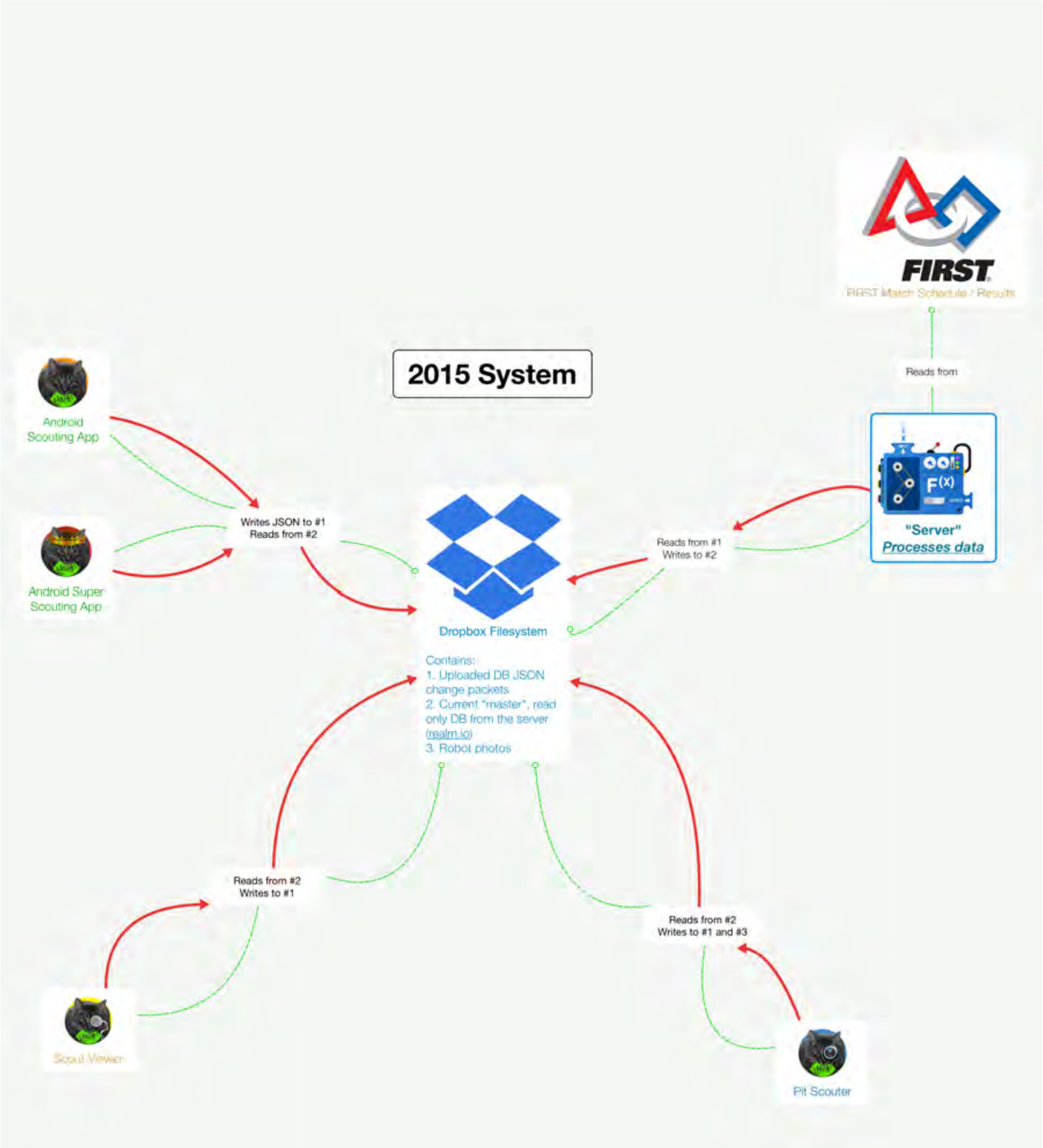
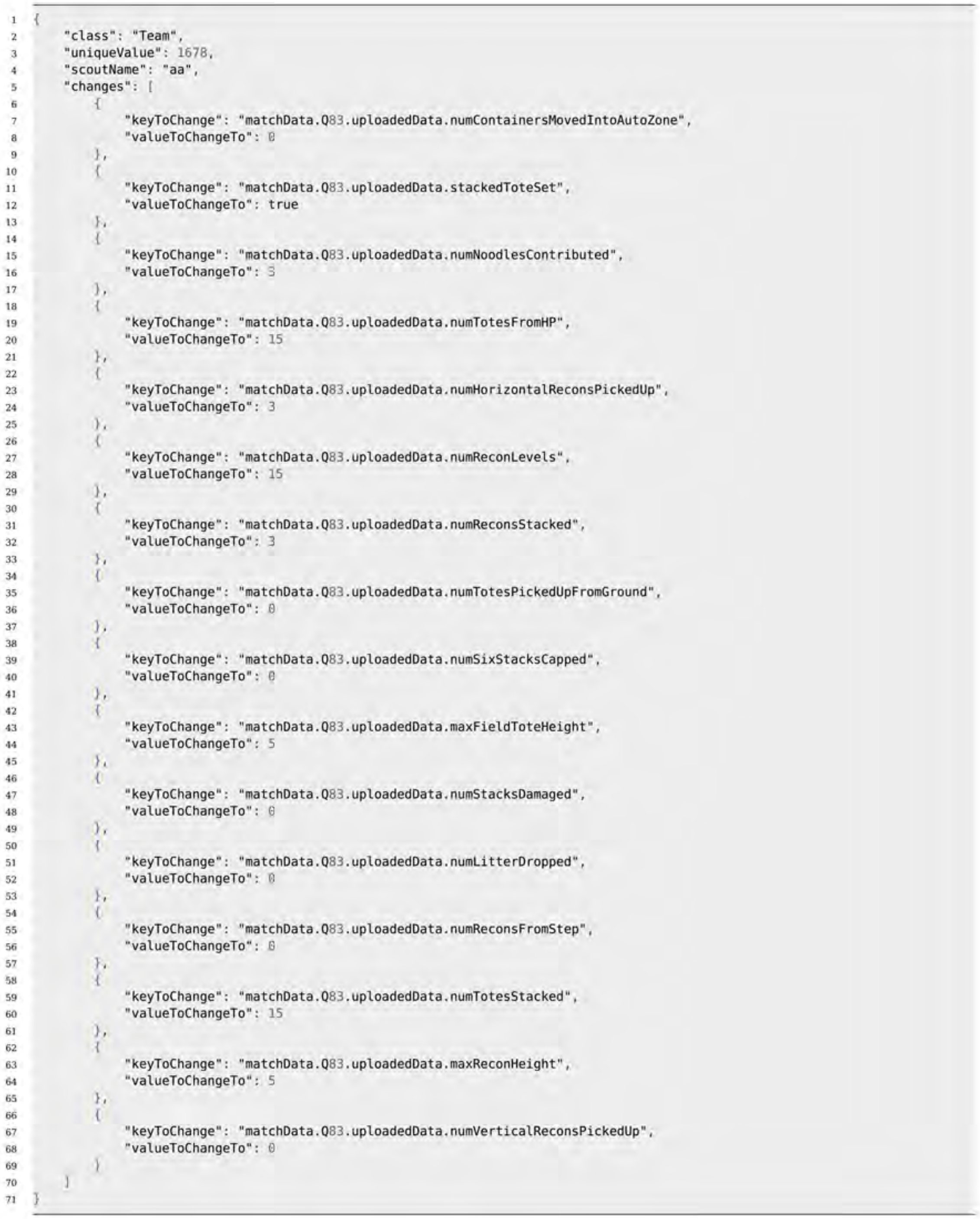


Figure 1.1: Overview of 2015 Scouting System w/o Cheesecake App



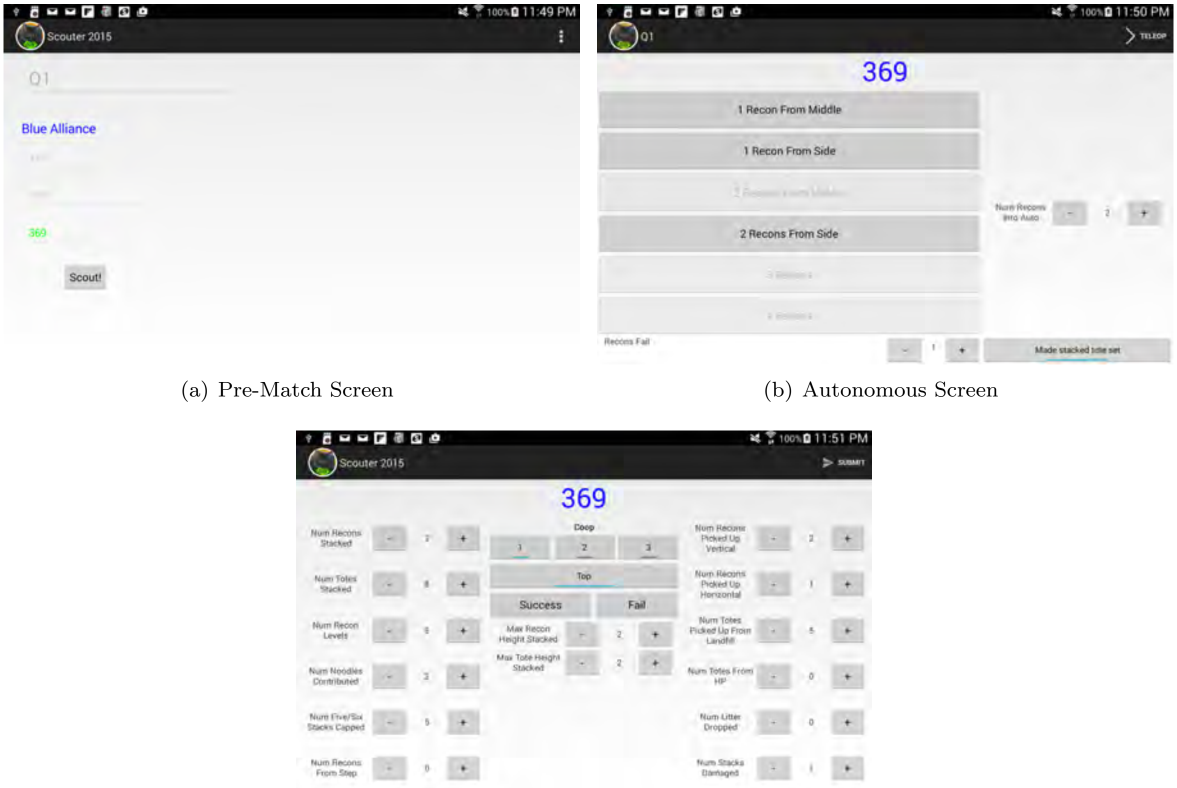
```
{
    "class": "Team",
    "uniqueValue": 1678,
    "scoutName": "aa",
    "changes": [
        {
            "keyToChange": "matchData.Q83.uploadedData.numContainersMovedIntoAutoZone",
            "valueToChangeTo": 0
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.stackedToteSet",
            "valueToChangeTo": true
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numNoodlesContributed",
            "valueToChangeTo": 3
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numTotesFromHP",
            "valueToChangeTo": 15
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numHorizontalReconsPickedUp",
            "valueToChangeTo": 3
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numReconLevels",
            "valueToChangeTo": 15
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numReconsStacked",
            "valueToChangeTo": 3
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numTotesPickedUpFromGround",
            "valueToChangeTo": 0
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numSixStacksCapped",
            "valueToChangeTo": 0
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.maxFieldToteHeight",
            "valueToChangeTo": 5
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numStacksDamaged",
            "valueToChangeTo": 0
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numLitterDropped",
            "valueToChangeTo": 0
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numReconsFromStep",
            "valueToChangeTo": 0
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numTotesStacked",
            "valueToChangeTo": 15
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.maxReconHeight",
            "valueToChangeTo": 5
        },
        {
            "keyToChange": "matchData.Q83.uploadedData.numVerticalReconsPickedUp",
            "valueToChangeTo": 0
        }
    ]
}
```

Figure 1.2: Sample Change Packet

1.3 Technical Details

1.3.1 Scout

The regular scouts collect easily determined data on one robot each match. There are six scouts, each scouting one robot. The data they collect includes the number of totes stacked, the number of recycling containers intaken, and the maximum height of the totes stacked. The regular scouting app is capable of running on any Android tablet running a recent version of Android and with Bluetooth capabilities. The app requests the schedule from the super scout via Bluetooth, preventing the scouts from needing to enter the teams every match, thereby reducing a significant number of scouting data errors. The scouts have the option of overriding the match and teams, in case of match replays or other differences from the schedule. The scout then collects the data and, when the match is over, the tablet saves the data in JSON format[5] in case there are transfer issues. It also transfers the JSON file to the super scout tablet of the same color over Bluetooth. As the Bluetooth communication is relatively inconsistent, the UI makes it very easy to re-upload data that fails to properly upload.



(a) Pre-Match Screen

(b) Autonomous Screen

(c) Teleop Screen

Figure 1.3: Scout User Interface

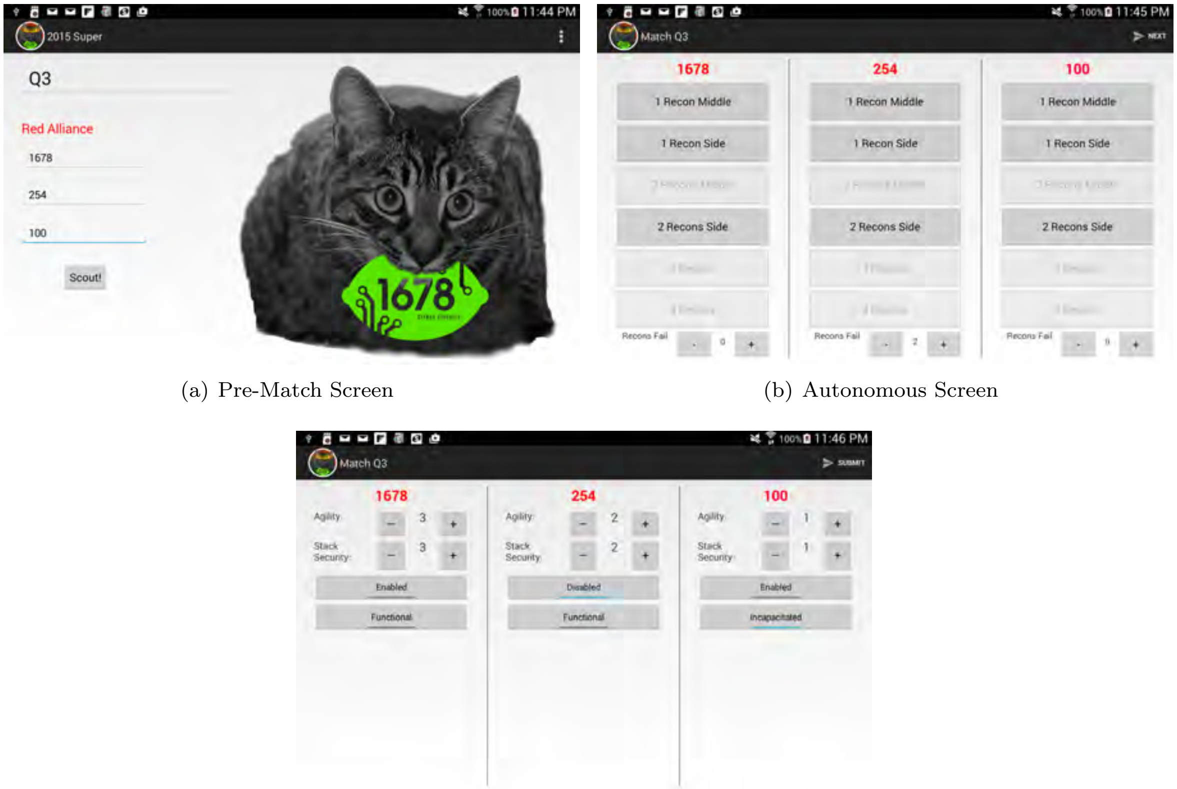
The UI consists of three screens: the pre-match screen, the autonomous screen, and the teleoperated screen (See Figure 2). The pre-match screen displays the match, the teams on the scout's alliance, and the team that the scout will be scouting. The pre-match screen also shows the matches that have been scouted and allows the data to be easily re-uploaded. When the match starts, the scout transitions to the autonomous screen, where a variety of buttons allow the scouts to enter how many recycling containers the

[5] JavaScript Object Notation. For more information, see `http://json.org/`

team collects from the step, as well as how many recycling containers and yellow totes they move into the autonomous zone and if they make a stacked tote set or not. At the end of the autonomous period, the scouts transition to the teleoperated screen, where they are able to input a variety of data points through steppers. The choice of steppers to input data allows for maximum flexibility in data input, as well being easy to implement and allow a high density of data points on the screen. At the end of the match, the scout transitions back to the pre-match screen and increments the match.

1.3.2 Super Scout

The super scouts collect data that is more subjective for all the robots on an alliance. There are two super scouts, one for each alliance. The super scout runs on tablets that have Bluetooth capabilities and cellular data access. The super scouts download the Realm database from Dropbox, giving them the schedule. When a scouting tablet scouting the same alliance requests the schedule over Bluetooth, the super scout transfers the schedule back in JSON format. The super scout has a similar pre-match and autonomous screen to the regular scouting app, but the teleoperated screen displays steppers that allow input of different data points. The super scout also handles the uploading of the data to Dropbox. All of the change packets are transferred via Bluetooth to the super scouts, and this data, along with that collected by the super scout, is uploaded to Dropbox.



Q3

Red Alliance

1678

254

100

(a) Pre-Match Screen

1678: Recons Fail 0

254: Recons Fail 2

100: Recons Fail 0

(b) Autonomous Screen

1678: Agility: 3; Stack Security: 3; Enabled; Functional

254: Agility: 2; Stack Security: 2; Disabled; Functional

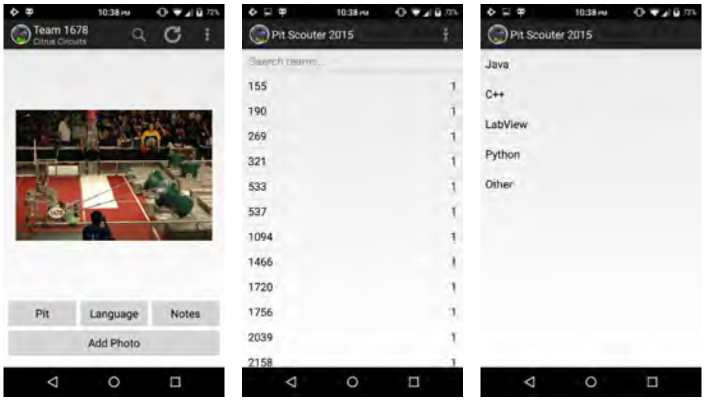
100: Agility: 1; Stack Security: 1; Enabled; Incapacitated

(c) Teleop Screen

Figure 1.4: Super Scout User Interface

1.3.3 Pit Scout

The pit scout goes through the pits and collects images and design details on every robot, such as programming language. The app consists of a list of teams which displays the number of photos taken of each robot, so that the the pit scout can ensure that every team is scouted. The teams can be selected, opening a separate view which has capabilities for data uploading. When the data is entered, the local version of the Realm database is modified, in addition to a change packet being saved to the device. When the data is uploaded, the app uploads all of the change packets and deletes them. If the data upload fails, the files are preserved and can be uploaded again, ensuring that no data is lost. Files are uploaded to Dropbox directly. In order to ensure that the best photo is used in the viewing apps, the pit scout has the ability to choose a *selected image*, which is shown in the viewing apps if it exists.

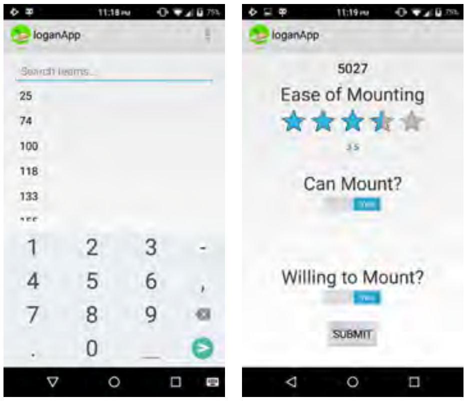


(a) Team Screen (b) Teams List (c) Data Entry

Figure 1.5: Pit Scout User Interface

1.3.4 Cheesecake App

The Cheesecake App[6] collects data on the feasibility of cheesecaking other robots at the competition. A similar method of storing and uploading the data to pit scout is used in the Cheesecake App.



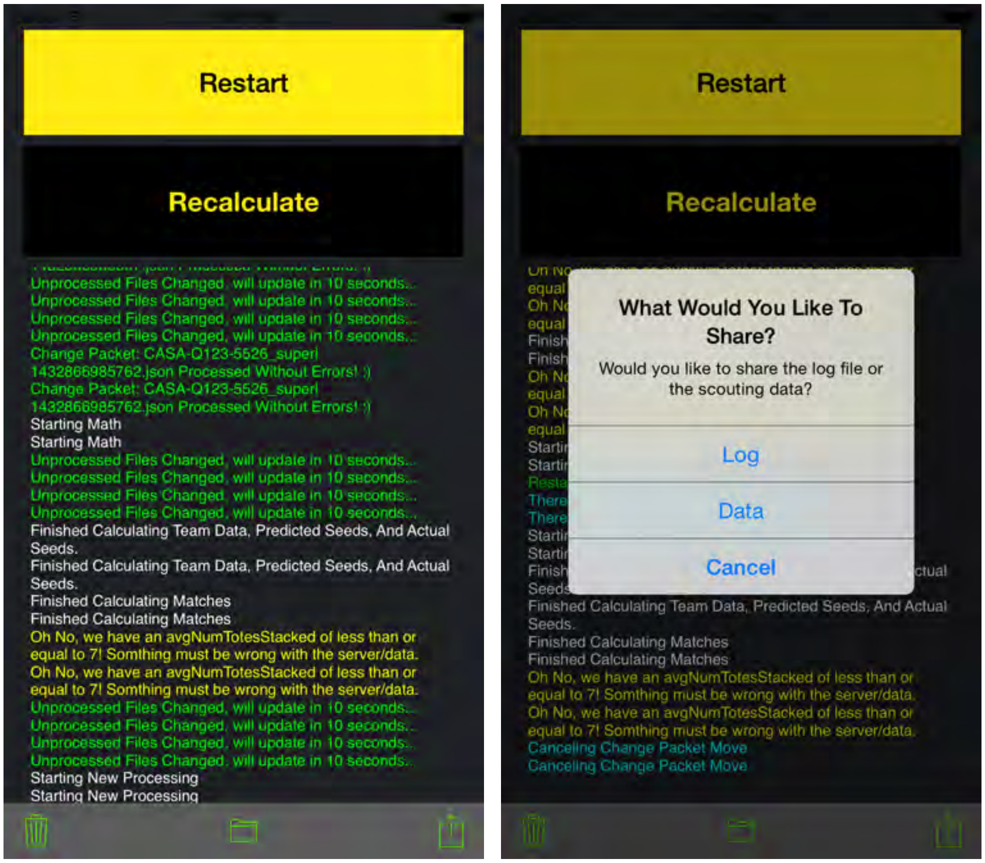
(a) Teams List (b) Data Entry

Figure 1.6: Cheesecake App User Interface

[6] Also known as the Logan App

1.3.5 Server

The server app runs on any device running iOS 7 or greater. The server is the only app to directly modify the central Realm database, preventing conflicting copies of the database from being created. In addition, the server calculates the calculated metrics, enters them into the database, and uploads the database to Dropbox. The server detects whenever new change packets are uploaded. It then waits for a certain period of time to ensure that all of the change packets are uploaded before it begins processing. The server parses all of the change packets, enters the data, and runs the calculations.



```
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Change Packet: CASA-Q123-5526_superl
1432866985762.json Processed Without Errors! :)
Change Packet: CASA-Q123-5526_superl
1432866985762.json Processed Without Errors! :)
Starting Math
Starting Math
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Finished Calculating Team Data, Predicted Seeds, And Actual
Seeds.
Finished Calculating Team Data, Predicted Seeds, And Actual
Seeds.
Finished Calculating Matches
Finished Calculating Matches
Oh No, we have an avgNumTotesStacked of less than or
equal to 7! Somthing must be wrong with the server/data.
Oh No, we have an avgNumTotesStacked of less than or
equal to 7! Somthing must be wrong with the server/data.
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Unprocessed Files Changed, will update in 10 seconds...
Starting New Processing
Starting New Processing
```

(a) Main Screen

What Would You Like To Share?

Would you like to share the log file or the scouting data?

```
Oh No
equal
Oh No
equal
Finish
Finish
Oh No
equal
Oh No
equal
Startin
Startin
Resta
There
There
Startin
Startin
Finish                                     ctual
Seeds
Finished Calculating Team Data, Predicted Seeds, And Actual
Seeds.
Finished Calculating Matches
Finished Calculating Matches
Oh No, we have an avgNumTotesStacked of less than or
equal to 7! Somthing must be wrong with the server/data.
Oh No, we have an avgNumTotesStacked of less than or
equal to 7! Somthing must be wrong with the server/data.
Canceling Change Packet Move
Canceling Change Packet Move
```

(b) Export Server Log

Figure 1.7: iOS Server User Interface

Why a mobile device for a server?

We decided to use a mobile device as a server this year because:

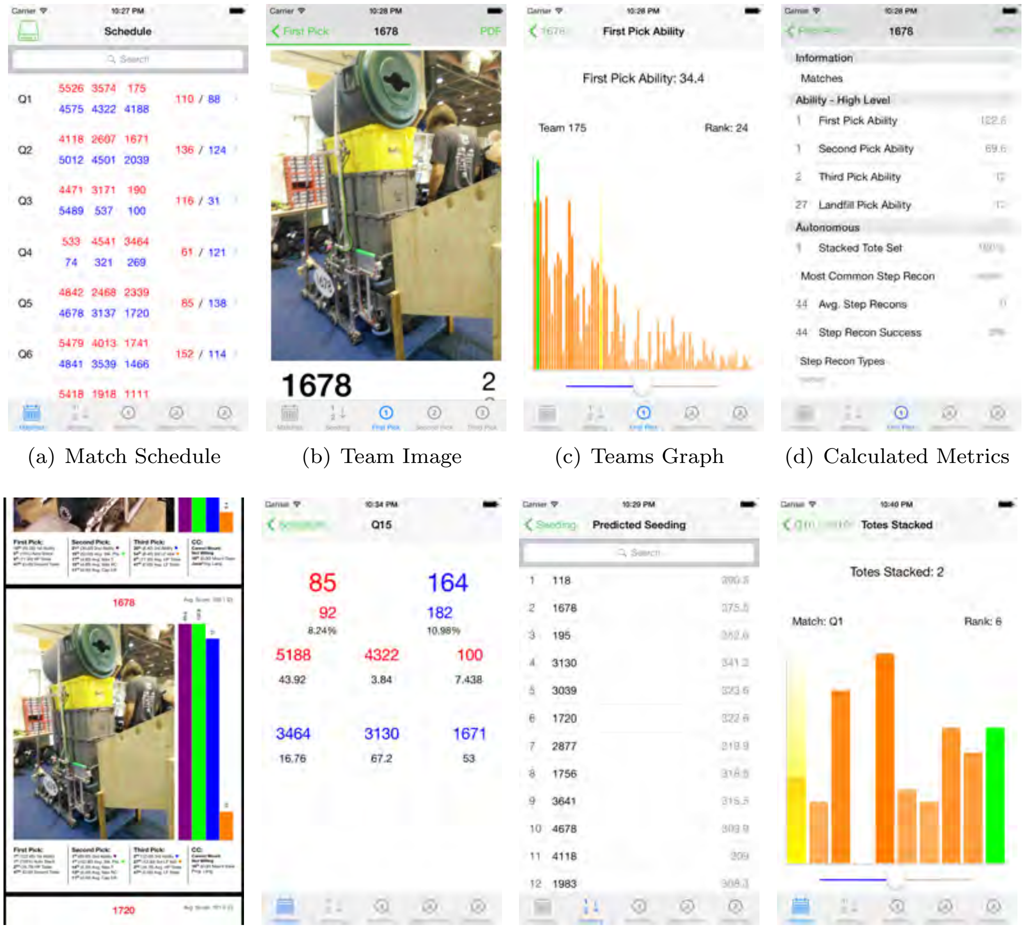
1. The abundance of mobile devices allows us to not have to worry about our server going down during competition.

2. The lower cost, since we do not have to pay for our own server.

3. The ease of debugging, as cellular data connections at competition can be very slow, causing SSHing to modify files on the server slow and difficult to do.

1.3.6 iOS Viewer

The iOS viewer allows anyone with an iOS device running iOS 8 or later to easily view all of the data collected by the system. The app displays both the raw data and the calculated metrics in an easily readable and analyzed form. The iOS viewing app downloads the Realm database so that it is able to function offline. The app downloads all of the images from Dropbox. The app allows the user to instantly view low resolution team images through Dropbox, as well as download higher resolution images on demand. The app possesses the ability to filter robots by their willingness and ability to cheesecake, as well as rank teams based on the highest level calculated metrics. The app allows the user graph and view data over both the team's matches and all of the robots in the competition. In addition to these capabilities, the iOS viewing creates PDFs of scouting cards that can be used in strategy meetings. These cards show high-resolution images of the teams, their calculated metrics, and displays graphs of the most critical data points for easy and fast understanding of the team's abilities. The iOS app displays the schedule and official scores, in addition to using algorithms to predict the scores and final seeding based on the collected scouting data. The app is also live-updating, allowing members of the drive and strategy teams to use the most recent data to plan match strategies.



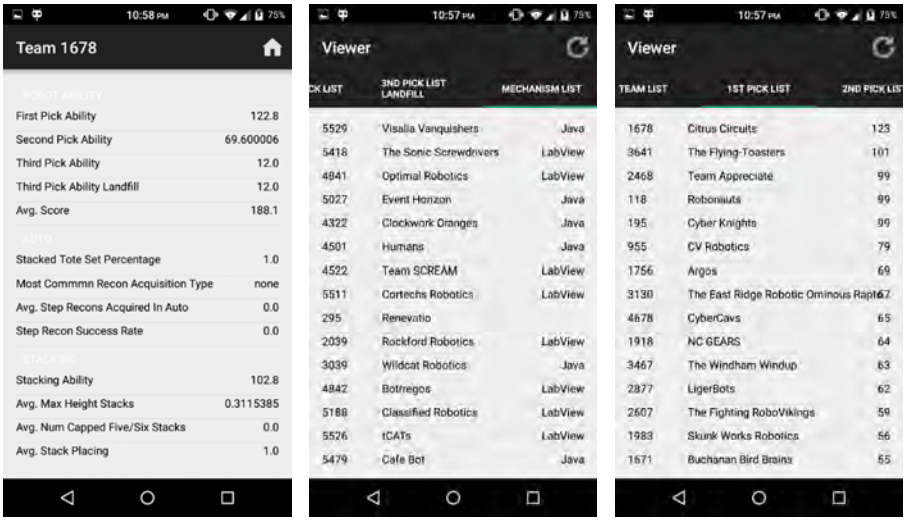
(a) Match Schedule (b) Team Image (c) Teams Graph (d) Calculated Metrics

(e) Team PDFs (f) Match Details (g) Predicted Seeding (h) Matches Graph

Figure 1.8: iOS Viewer User Interface and Capabilites

1.3.7 Android Viewer

The Android viewer allows anyone with an Android device running a recent version of Android to view the data. Most of the capabilities of the Android Viewer are the same a those of the iOS viewer, but it lacks the ability to generate the team card PDFs.



First Pick Ability	122.8
Second Pick Ability	69.600006
Third Pick Ability	12.0
Third Pick Ability Landfill	12.0
Avg. Score	188.1
Stacked Tote Set Percentage	1.0
Most Commmn Recon Acquisition Type	none
Avg. Step Recons Acquired In Auto	0.0
Step Recon Success Rate	0.0
Stacking Ability	102.8
Avg. Max Height Stacks	0.3115385
Avg. Num Capped Five/Six Stacks	0.0
Avg. Stack Placing	1.0

(a) Calculated Metrics

5529	Visalia Vanquishers	Java
5418	The Sonic Screwdrivers	LabView
4841	Optimal Robotics	LabView
5027	Event Horizon	Java
4322	Clockwork Oranges	Java
4501	Humans	Java
4522	Team SCREAM	LabView
5511	Cortechs Robotics	LabView
295	Renevatio	
2039	Rockford Robotics	LabView
3039	Wildcat Robotics	Java
4842	Botrregos	LabView
5188	Classified Robotics	LabView
5526	tCATs	LabView
5479	Cafe Bot	Java

(b) Cheesecake Filter

1678	Citrus Circuits	123
3641	The Flying-Toasters	101
2468	Team Appreciate	99
118	Robonauts	99
195	Cyber Knights	99
955	CV Robotics	79
1756	Argos	69
3130	The East Ridge Robotic Ominous Rapt	67
4678	CyberCavs	65
1918	NC GEARS	64
3467	The Windham Windup	63
2877	LigerBots	62
2607	The Fighting RoboVikings	59
1983	Skunk Works Robotics	56
1671	Buchanan Bird Brains	55

(c) First Pick List

Figure 1.9: Android Viewer User Interface and Capabilities

Chapter 2

System Logistics

2.1 The Scouting Team

2.1.1 Overview

The Citrus Circuits scouting team consists of 8 regular scouts, 2 super scouts, 1 pit scout, and 2 cheesecake scouts. The extra regular scouts allow the regular scouts to take breaks from scouting, which helps to increase data accuracy, as a combination of sleep deprivation and scouting without a break can cause scouts to lose focus and begin to accidentally enter incorrect data, especially considering the large number of data points our scouting system required entering this year. In addition, we bring two members of the app programming team to competition to maintain the system. However, the system could function with 6 regular scouts, 2 super scouts, 1 pit scout, and 1 cheesecake scout.

2.1.2 Training

One of the most crucial elements of a good scouting system is those entering the data. Incorrect data can be worse than no data at all, as it can lead to incorrect decisions being made. Therefore, it is imperative that the scouting team is able to consistently enter accurate data throughout an entire day of scouting. Therefore, there are high expectations and a rigorous training process for Citrus Circuits scouts.

Scout Training

Members of the travel team who are not part of the pit crew, drive team, or strategy team are expected to scout. The travel team is decided before the first competition and remains consistent throughout all of competition season. Scouts are first walked through the app by the programmers who wrote it. After this, they undergo a test that tests their understanding of the importance of scouting and how the app works. They must retake the test until they receive a 95%, ensuring that the data which is entered will be accurate. After this, the scouts practice with actual match footage[1] in order to give the scouts the difficulty of scouting a real time match with difficult viewing angles. In order to easily measure the accuracy of the data, all of the scouts scout a single robot and are not allowed to move on to another match until all of their data is the same. Training was considered complete this year when the scouts were able to input the same data five different matches in a row. The use of such a rigorous training process, while highly time consuming, ensures that the data is entered as accurately as possible, which is absolutely critical to the success of any scouting system. In addition, it helps to emphasize the expectations for scouting, ensuring that they concentrate and work well at competition.

[1] Match footage can be easily found through the Blue Alliance and YouTube.

Super Scout Training

While super scouts receive training, the training is usually less rigorous than that of the scouts due to the nature of the data they input. As the data is more of an overall, subjective metric, the ability to accurately input data in real time is less critical. Instead, the most important part of the super scout training process is selection. Super scouts should be experienced, have a thorough understanding of the game, and be highly dedicated, as they are not given breaks from scouting.

Pit Scout Training

Pit scouts must be competent at taking photos and talking with teams. Pit scouts are generally members of the business and media team who already possess the photography skills to succeed. They are also chosen based on their ability to interact well with other teams. The pit scout is then given a walkthrough of the app and time to test it while they have the ability to question the app programmers if they are confused. Due to the more simple nature of this job, there is less training needed than for scout and super scout.

Cheesecake Scout Training

Cheesecake scouts must be competent both technically and interpersonally. Cheesecake scouts are generally members of the mechanical team who already possess the technical skills to determine the possibility of successfully cheesecaking a team. They are also chosen based on their ability to interact well with other teams. The cheesecake scout is then given a walkthrough of the app and time to test it while they have the ability to question the app programmers if they are confused. Due to the more simple nature of this job, there is less training needed than for scout and super scout.

Full Competition Simulation

Before competition, to ensure that the system will be able to function completely, a full systems test is held. The scouting team gathers with all of the equipment that wull be used at competition and scouts a livestreamed competition as if they were present. In addition, the processing and viewing elements are run as they would be at competition. This helps to ensure that any bugs are discovered and questions dealt with in a less high-pressure, more controlled environment. It also allows examination of the calculated metrics and their accuracy in measuring robot capabilities.

2.2 Designing a Scouting System

The design of a scouting system is integral to its success. A poorly designed system can waste valuable time and effort, and only hinder a team. Therefore, in order to ensure that the system is properly designed, the Citrus Circuits strategy and app programming teams have a multi-day meeting immediately after the initial strategy and brainstorming session which continues until the system and all of the calculated metrics are completely designed and agreed upon. Through this process, the quality of the scouting system is ensured, reducing the amount of time later necessary to implement and optimize the system.r

Chapter 3

Review and Discussion

3.1 Conclusions

3.1.1 Overall

The Citrus Ciruits' scouting system successfully enabled us to process and visualize data in real time, and played a significant and successful role in alliance selection and match strategy.

3.1.2 Infrastructure

Dropbox

Dropbox did not perform its role well this year, and we strongly recommend not using it for this application. We abused the Sync API, leading it to sometimes fail completely. For example, at St. Louis, for an hour the API completely ceased working, causing the entire system to fail irreparably for that brief period of time. It also sporadically failed to correctly sync the local and remote storage and made many tasks more difficult than they needed to be, such as implementing live syncing. Because of this, it was also very difficult to successfully implement some features. Overall, Dropbox performed its intended purpose well, but our abuse of the API led to failures and poor performance. We will not be using it next year.

Realm

Realm performed its role well, but we did encounter some issues. It was inconvenient that it was only mobile and OS X, and it was very difficult to remove it from a device. However, overall, while it was in some ways inconvenient, it performed well, and many parts of it were very convenient to work with. However, due to its mobile-only nature, we will not be using it next year.

3.1.3 Lessons Learned

Do Not Abuse Technologies

The main issues in the system occurred when technologies were not used for their intended purpose. For example, the Dropbox Sync API is designed to allow simple storage of files in Dropbox. However, we used it as a hub for our entire system, with complex file transfers, reads, and writes. This abuse of the API led it to both fail and be very difficult to work with. We had a similar experience when we attempted to stretch the capabilities of Realm. **When selecting infrastructures for a scouting system, ensure that infrastructures fully cover and are intended to fully cover the intended usage.**

Communicate with Users

Many of the issues we encountered could have been avoided by ensuring that the features we were implementing were desired by the strategy team. While the strategy team was often to busy to talk, it is imperative to force them to communicate and to have everything prepared so they can have a concise meeting. **When designing a scouting system, ensure that the users and the designers agree on the planned usage of the system.**

3.1.4 Final Words

The Citrus Circuits scouting system effectively helped to develop match and alliance selection strategies. Through the use of the data provided, the strategy team was able to make better-educated decisions, contributing to our success. An electronic scouting system is a powerful tool in the belt of an FRC team in helping to optimize robot performance on the field.

3.2 Contact Us

We are highly interested in helping to develop the FRC scouting community and opening the power of an electronic scouting system up to many more teams. If you have any questions, please contact us at frc1678@gmail.com. In addition, all of the code can be found on Github at `https://github.com/frc1678`.