Control Room Final Report

Liam Bradley
Joshua Snyder
Robby Weaver

YCAS Radio Telescope Project
CS Capstone I, Fall 2022
Professor Donald J. Hake II
York College of Pennsylvania

Table of Contents

Abstract

The following is the technical report for the Control Room team written during the Fall Semester of the fifth year of development on the YCAS Radio Telescope Project. Over the course of the Fall 2022 semester, the Control Room team was able to make significant improvements and enhancements, finalize certain aspects, add new functionality, and make the Control Room Application more stable, testable, accessible, and functional. The team focused primarily on hardware bug fixes, but also implemented various features on the Control Room Application and the PLC, all with the intention of producing a more robust end product.

The current plan is to deploy the telescope at John C. Rudy County Park during the Spring 2023 semester. Further real-world systems-level acceptance testing will need to be done once this is completed, which future team members will be responsible for. The Control Room team has compiled a list of tasks that will need to be completed prior to installation at the park as well as a list of key tasks to complete after installation.

Introduction

York College of Pennsylvania (YCP), the York County Astronomical Society (YCAS), and the York County Park System (YCPS) collaborated in an effort to develop and design, from the ground up, a radio telescope to be installed at the York County Astronomical Society (YCAS) observatory located at John C. Rudy County Park. This telescope has the capabilities of remote access, auto location, and auto-tracking.

Some of the Control Room team's tasks this semester include continuing the development of the Control Room Application for the radio telescope, and adding any last requested features, such as:

- Implementing limit switch overrides on PLC
- Implementing a Control Room disconnection routine for the hardware
- Create Appointment Form
- Remote Desktop Setup on Control Room PC
- Handling Overdue Appointments
- Make Email & Push Notifications togglable

Other tasks of the Control Room team include improving upon and testing the work of previous teams, such as:

- Finalize existing functionality
    - Improve testing
    - Fix bugs
    - Log any unfixed bugs to the issue tracker so they do not get forgotten
    - Document how different parts fit together and interact with each other
- Verify that functionality that was instantiated from previous teams can be replicated reliably and consistently
- Resolving issues with Encryption (Mobile App communications)

The Control Room team faced many challenges and obstacles this semester including onboarding a new team member, having to learn about the hardware on the fly, and troubleshooting and accounting for new hardware issues as they arose. Perhaps the most difficult obstacle the team faced this semester, however, would be limited time. Even with the installation's delay to spring, hardware issues took far longer to diagnose and resolve than we could have anticipated. Even so, the Control Room team was able to persevere and succeed in the objectives that were defined at the beginning of the semester and are well-positioned to onboard new members and continue work into the Spring 2023 semester.

Background

Throughout the semesters, each new team has brought something new to the Control Room Application in terms of functionality and usability. The team previous to this semester, in the Spring of 2022, inherited a mostly working system, but persevered with the help of other engineering teams to deliver a mostly functional system. At the end of the 2022 Spring semester, the telescope was in an operable state with most features implemented, however the project lacked robustness which would be the overall theme for this 2022 Fall semester. In addition to robustness, because of the current lack of front-end functionality, the Control Room team would be responsible for implementing a system that would allow for the addition and management of appointments.

As the Control Room team continues to expand the functionality of the Radio Telescope, the Mobile App team must also accommodate any of the updates.

The Control Room Application software (CR) drives the telescope's hardware, communicating with a Programmable Logic Controller[2] and Motor Controller Unit (MCU)[3], and receives radio frequency (RF) data through the SpectraCyber[4] which is then saved to a MySQL[5] database. The database, using the Entity framework[6] will save relevant data that the Mobile and Front End teams can access and present for user consumption, such as:

- Appointment RF data
- Sensor statuses
- Temperatures
- Acceleration data blobs
- Weather information
- Appointment statuses

Because of this communication with the database, the YCAS members who will be administering the telescope will be able to effectively and efficiently control the telescope, maintain it, and whenever it enters an unsafe condition, return it to a safe and stable state.

The Control Room team's main tasks this semester included:

- Implement Software Stops Toggle and Backup Redundancy - H
- Finish implementing limit switch overrides - H
- Control Room exceptions out when the connection gets ended - H
- Fix Shutdown RT occasionally freezing the application - H
- Fix Slow Encoder Updates on Diagnostics Form - H
- Implement Azimuth Encoder Averaging routine - H
- Resolve issue where unplugging the MCU freezes up the CR UI - M
- Fix Hardware Disconnection Exceptions - H

- If sensor data goes offline, make sure we handle that scenario - H
- Handle Bad Accelerometer Data - H
- Create an executable project file for CR - H
- Check other Appointment types - H
- Handle Overdue Appointments - M
- Make email and push notifications togglable - M
- Spectral Scan export to CSV - M
- Appointment management from the Control Room and interval appointments - M
- TCP does not close forms - L

Key:

- H - High Level Importance
- M - Medium Level Importance
- L - Low Level Importance

Implementation

Sensor Network

The Sensor Network[1] gathers sensor data from various sensors and sends that data, at regular intervals, to the Control Room software. This data is received in a packet that must be decoded on the Sensor Network Server end and then parsed into a representation that can be displayed on the Control Room software's user interface. The Control Room software and/or the user can then make decisions based on that decoded data, if necessary. Requests to turn the fan on and off are also sent from the Control Room to the Sensor Network. A separate Sensor Network Testing Suite application that facilitates troubleshooting and generating test CSVs and packets has also been developed alongside the Control Room software.

Here are the various types of sensor data being retrieved from the Sensor Network:

- Azimuth Motor Temperature
- Elevation Motor Temperature
- Azimuth Motor Acceleration
- Elevation Motor Acceleration
- Counterbalance Acceleration
- Elevation Absolute Position
- Azimuth Absolute Position
- Elevation Motor Position
- Azimuth Motor Position
- Internal Temperature and Humidity

Simplified Encoder Data / Fix Encoders Too Far Apart Error

In order to increase the speed that data can be processed and sent to Control Room, the data for both the elevation and azimuth absolute encoders has been reduced. Originally, 200 samples were being collected, processed and sent. We changed this to only 10 samples, and calculated the minimum, maximum, and average values of both data sets. These new pieces of data are now being sent to Control Room for data processing. The Control Room will now save a running average of 10 packets, and perform a comparison when processing a new packet to ensure that the packet is within an acceptable range. This range is calculated based on the speed of the telescope. In addition to this, there is a certain level of error allowed in the event of inconsistent readings (perhaps from a bump) before canceling the movement.

Manual Control Disabled on Control Form Close

Originally, the state of the Activate/Deactivate Manual Control button would be saved to the local database, thus, when reopening the RT Control Form, if one had already activated manual control, it would open the form with manual controls activated.

To address this issue, we had the values of the manual control buttons set to their deactivated state upon closing the form. These values were set within the Form_Closing method, which is called whenever a WinForms window is closing.

Spectra Cyber Reconnection Routine / Checking If Online

The issue with the Spectra Cyber's online status originated with what we thought was only an issue where a table value was not being updated on the Diagnostics Form. However, we would discover that there were various issues with the Spectra Cyber connection. Namely, we were using the incorrect COM port number and the means of updating the Diagnostic Form's table of hardware statuses was not functional. The process of updating to the table was incorrect, as was the method which checked if the Spectra Cyber was connected or not.

To check if the Spectra Cyber is online or not, we complete a single scan to the SC. If the scan times out, we know that the Spectra Cyber is not connected. This method, along with checks to see if the Sensor Network and MCU are connected, are executed in a separate thread.

This method worked well until we considered appointments. Sending an additional scan while an appointment is running can interrupt that process. So, we ended up reverting to a formerly implemented status-checking method that checks if the Spectra Cyber is mid-appointment before sending a scan request.

However, this method would lead to an error as well when appointments were running. Sometimes during appointments, the Mode Type Enum of the Spectra Cyber would be set to Unknown, which would lead to an error being thrown in the process of writing a command to send to the Spectra Cyber. Defaulting to a continuum scan fixed this issue.

We also implemented a routine that will attempt to reconnect to the Spectra Cyber in the event that we lose connection to the device. Previously, the Control Room application wouldn't attempt to reconnect. We used the BringUp and BringDown methods as part of the routine to try and connect to the device.

Downloading Code to the PLC / Using PC Worx Express

Implementing the Limit Switch Overrides and CR Disconnect → Stow routine required us to learn how to download code to the PLC as well as write this code in the first place. Our Phoenix Contact PLC uses PC Worx Express as its means of programming. PC Worx Express uses function blocks and ladder logic and does not have wonderful documentation. However, the Control Room team has taken the time to write documents that explain how to download code to the PLC as well as how to use PC Worx Express in general.

CR Disconnect → Stow Routine

When the Control Room disconnects from the Radio Telescope, the telescope is not necessarily left in a safe state. In fact, it could be in an unideal state where the dish is horizontal to the ground. This is a dangerous position for the telescope when there are high wind speeds, as the dish and supports have to withstand these gusts.

In order to address this, we implemented a routine that will move the telescope to Stow upon disconnection from the Control Room. To create this routine, we first needed to interface the PLC so that it was capable of writing movement commands to the MCU. The Control Room, PLC, and MCU all can communicate with one another through the Modbus TCP/IP protocol. These communications were set up for synchronous communications between the PLC and MCU, such as for checking if MCU errors were hit or an E-Stop button was pressed. However, the PLC was not set up for sending specific movement commands to the MCU when certain events occur (such as the Control Room disconnecting). This required us to install a Modbus TCP/IP function block library for PC Worx Express.

Once this function block library was installed, we used the function blocks to create a Modbus Client connection with the MCU. Once connected, the client connection would send a WriteMultipleRegisters command that would have the MCU jog the telescope upwards until it hit the upper limit switch. We had to manually type the movement command into the PLC code and determine the starting register address in the MCU for elevation movements.

This process was not ideal. Once the telescope hit the upper limit switch, the motors were stopped. Hence, we weren't able to send another movement to have the telescope jog off of the limit switch and properly stow.

To address this issue, we implemented a better routine. On Control Room disconnect, the telescope waits ten seconds (in case CR only disconnects for a couple seconds and then reconnects). The PLC then creates a Modbus TCP/IP Client connection with the MCU. After this is created, the PLC sends a movement command to home the elevation. After 45 seconds, which allows enough time for the telescope to reach the elevation home, the PLC then sends a movement command for a relative move. We calculated the number of degrees of movement we needed to reach stow from elevation home. From here, we had to convert degrees to motor steps, also accounting for the 50:1 gear box on the elevation motors. With this change, the telescope is able to go to stow properly on CR Disconnect.

Note: The number of steps may need to be adjusted on the PLC if the homing bracket is moved.

Limit Switch Overrides

Limit switch overrides are in place to prevent a limit switch from preventing movements. We have two specific use cases for limit switch overrides: to back off of a limit switch if we are stuck on the limit switch, and to move to a safe position when we have a malfunctioning LS that

is preventing movements all together. In both cases, we only allow manual controls and block movements that go towards the disabled limit switch.

Initially, we did not have this understanding of what was wanted. Originally, we thought that if a limit switch was disabled, we would be able to move in any direction regardless of if we hit a limit switch or even went past it. This was what we first implemented until receiving further clarification from Professor Hake, who advised us to always prevent movements in the direction of the disabled limit switch.

In order to prevent the PLC from stopping motors when hitting a limit switch, we had to designate a PLC Modbus register as the value used to record which limit switches were overridden. We were able to create this register and its address in the Bus Configuration screen on PC Worx Express.

On the PLC side, the value stored in the register is an integer value (word). The value can be written by the Control Room to be either 0, 1, 256, or 257. These values map to both limit switches enabled, lower disabled, upper disabled, and both disabled. Initially, the register value would be read from the PLC and updated via addition/subtraction based on the configuration on the Control Room. If the register value is equal to one of these integer values, the PLC disables/enables the respective limit switches. We had to alter the current STOP_MOTORS routine to account for disabled limit switches. With this implementation, we were able to move past the limit switches when the limit switch is disabled.

In order to prevent movement in the direction of a disabled limit switch, we implemented a software stop specifically to check if a limit switch is disabled. With this, all movement commands which move in the direction of the limit switch are canceled before they can execute.

We did end up rewriting the routine in which we write to the PLC register for limit switch overrides. Originally, we read the PLC register and updated based on which limit switch was hit by adding/subtracting to the register value. We realized later that this was less reliable and adjusted the routine so that PLC was never read. Instead, the Control Room determines what to write entirely based on the desired LS override configuration in the Control Room.

Email / Push Notification Enable/Disable Button on Diagnostics Form

At times, it is desirable to disable push/email notifications when running tests on the Control Room. This avoids sending a plethora of error messages to the mobile app users and email accounts associated with RT.

To address this, we implemented a checkbox that changes a boolean value for whether or not to enable/disable push and email notifications. This checkbox is located on the Diagnostics Form. Because the Email and Push notifications are static methods in their respective classes, we had to change the methods slightly. We added a parameter that is set to true when notifications are enabled. This required us to add a notification boolean to each class that used the notification classes, as this allows us to set the value of the parameter in the static notification methods. These values in each class are updated in the Main Form upon clicking the checkbox. It wasn’t

the cleanest of solutions, but it was the best way to avoid having to rewrite the email and push notifications entirely.

Remote Desktop Connection to Control Room PC

When the Control Room PC and Radio Telescope are stationed in the park, we would have to drive to the park in order to access the PC and Control Room Application itself. However, with a Remote Desktop Connection, an authorized user will be able to connect to the PC and use the PC remotely.

We installed Windows 10 Pro onto the CR PC, which allows for the CR PC to be accessed remotely. Any Windows 10 PC (Home and Pro licenses) will be able to access the CR PC remotely. The remote desktop connection was verified to work as expected when using the dplink router in the Project Workspace. In order to use it over the Internet, some IP configuration will need to be completed. We could not complete this yet, as we do not yet have the PC stationed in the park with an Internet connection separate from eduroam.

Handling Overdue Appointments

At the start of the semester, the Control Room Application would start an appointment with the earliest start time automatically so long as the appointment was not yet canceled or completed. However, it did not consider if the appointment was overdue. We consider an overdue appointment to be an appointment whose start and/or end time has passed. This might occur if the Control Room goes offline for a significant amount of time. Currently, we allow for an appointment to start up to one minute late. Thus, an appointment that starts at 1:30:30 but is scheduled for 1:30:00 is not considered overdue.

There were 3 main cases that we needed to consider:

- Case 1: Appointment is on time and starts on time / within the overdue threshold.
- Case 2: Appointment is not on time, but the end time has not been reached.
- Case 3: Appointment is not on time and the end time has already passed.

Each of these cases requires a different course of action. For case 1, we can simply continue the appointment as it is on schedule.

For case 2, we can still start the appointment. However, it is possible that the appointment will not finish on time due to the delay to its start. Hence, we will start the appointment and notify the User via Email that their appointment started late. The Control Room will end this appointment at its end_time.

For case 3, the appointment was missed entirely. Since we do not want to potentially interrupt other appointments, this appointment is canceled and the User is notified of such via Email.

Graph Skipping in Diagnostics Form

On the Sensor Data tab in the Diagnostics Form, three graphs display the acceleration data from the three motor accelerometers (azimuth, elevation, counterbalance). There was a small bug where the graphs would, on occasion, skip over received data instead of displaying data in real time.

We discovered that the graph skipping was caused by a small logic error in the Diagnostics Form code. In the previous version of the Control Room Application, the graphs would be updated if the incoming data was not null and if the incoming data matched the previous data. However, we want the updates to occur when the incoming data does NOT match the previous data. Hence, we simply had to update the logic with NOT operators.

Implement Software Stops Toggle and Backup Redundancy

The Control Room needed a way to continue to read elevation data in the event that one or more of the sensors reading the value fails. One of the first tasks this semester was getting this implemented. The telescope now had three methods of reading elevation data; the elevation absolute encoder, the counterbalance accelerometer, and the motor encoder. A change was implemented to have the Control Room automatically switch between these sensors if the current sensor in use fails. It should be noted that the motor encoders will only be used if both the elevation absolute encoder and the counterbalance accelerometer are both in a failure state. Users will also be unable to select any sensor that is currently in a failure state. Once the sensor is no longer failing, users will be able to select and use it again.

Appointment Management via Control Room

Users now have the ability to perform basic appointment management from the Control Room. They have the ability to add and view both appointments and users. To access these functions, users will need to connect the Control Room to the telescope. After this, they will be able to click into the Appointment Control form and access all of the appointment management features. The forms were made to be easy to navigate and understand, and some items are inputted as drop down lists to ensure the user enters the correct input for these said items. Anytime a user or appointment is added with the form, it is immediately injected into the database, hence any kind of appointment added can be run without needing to perform any additional actions. The same goes with users. A new user added using the Control Room can then be used to run a new appointment. This removes the need of using the front end, which is currently in an unstable state, to create and manage appointments.

Documentation

General Information

This manual provides general information for setup and use of the radio telescope.

Telescope Startup Manual

This manual provides information on how to start the telescope from a cold state, as well as how to configure the CR Software for use with the simulation and production settings.

Control Room Manual

This manual provides information on how to use all features associated with the Control Room application.

ESS Manual (OTA Updates/USB Upload)

This manual provides information on how to modify the embedded sensor network code, and how to upload it to the ESS via Bluetooth or from a USB cable.

PLC / PC Worx Express Manual

This manual provides information on how to use PC Worx Express and how to download code from PC Worx Express onto the PLC.

Remote Desktop Manual

This manual provides instructions on how to use the remote desktop functionality.

Future Work

There will be various tasks that need to be completed in the spring semester. We have sorted tasks based on whether they need to be completed prior to installation or after installation.

Before Installation at John C. Rudy Park

Verify Overall Functionality with Mobile App, Appointments, Calibrations, etc.

Test every feature that can be tested. If you run into an error, document the error and resolve the bug.

Updating the Number of Steps in Disconnect → Stow Routine

Depending on the degree orientation of the elevation's home, the Spring Semester team will need to update the number of steps for the relative move in the CR Disconnect routine on the PLC. This should be a fairly straightforward task, but will be much easier if they reach out to Robby or Liam before attempting to update this value.

It might be beneficial to store this number of steps in a Modbus register on the PLC that can be updated from the Control Room Application. That way, we can update the number of steps without having to redownload code to the PLC.

Email Notification for Overdue Appointments

The email and push notification for overdue appointments is currently commented out. Implement this notification and ensure it can properly notify the user.

ESS Watchdog and Ethernet Reconnection issues

The Watchdog is functioning properly. However, on a software reset, we are unable to reestablish the ESS' ethernet connection. As of now, the Ethernet.begin method gets stuck on a while loop that waits for the link_status to become true.

One way to address this issue would be to find a way to power off the ESS and reboot it, as it functions properly on the initial boot from a powered-off state. This could perhaps be accomplished via a routine in the PLC, if its power source could be controlled this way.

Spectral Scan to CSV

This task was not completed by the end of the semester. We will want to be able to record this data to a CSV so that the users can more easily utilize the data collected during appointments.

Create Appointment Form on Control Room

The Create Appointment Form is mostly finished on the Control Room Application. However, some issues arise whenever the Control Form is opened with the Appointment Control functionality implemented. It does not allow the Control Form to open due to an error trying to add the Free Control appointment to the form. This is because the database will not allow the appointment to save based on its fields. Investigate this issue and ensure that the AddAppointment Database method is being called properly and adds the correct data to the database.

After Installation at John C. Rudy Park

Remote Desktop Setup

The Control Room PC is updated to Windows 10 Pro. However, we cannot truly test the remote desktop connection until the telescope is installed at the park and has a separate Internet connection from the school's wireless connection. This may require some network configuration on the Control Room PC and the devices that are connecting to the CR PC, but should be a matter of determining IP addresses and recording those configurations.

Security Camera Installation

This might not fall into the Control Room's hands, but security cameras need to be installed at the park.

References

[1] (n.d.) SensorNetwork embedded software repository
- https://github.com/YCPRadioTelescope/YCP_RT_SensorNetwork

[2] (n.d.) Programmable Logic Controller quick start guide
- https://drive.google.com/file/d/1Ehr0wrP6aTUBFwSFX0TD-wXjXachdyYj/view?usp=sharing

[3] (n.d.) Motor Controller Unit manual
- https://drive.google.com/file/d/1FZIEM_U2YgjXn-HmcZCzW56SdwLZef_1/view?usp=sharing

[4] (n.d.) SpectraCyber documentation
- http://www.ncra.tifr.res.in/rpl/facilities/4m-srt/sci_iimanual.pdf

[5] (n.d.) Entity Framework Overview
- https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/ef/overview

[6] (n.d.) MySQL Official Site
- https://www.mysql.com/

[7] (n.d.) Amazon Web Services documentation
- https://docs.aws.amazon.com/index.html?nc2=h_ql_doc_do

[8] (n.d.) RemoteListener Documentation
- https://docs.google.com/document/d/16_HVjoNUsJ0Viq6EgDgaGzdX-fo2Y_0RmtDoi1egdQU/edit

[9] (n.d.) TCP Protocol Overview
- https://drive.google.com/file/d/1cV8tHFrGfRm__4U1mG9G2fSVQmTVpjK5/view?usp=sharing

[10] (n.d.) TCP Protocol In-depth documentation
- https://docs.google.com/document/d/1xSQuwcbpDC_5UnIyre336b_8t7jicmA4HsSlqfWuxt4/edit?usp=sharing

[11] (n.d.) Fall 2021 Control Room Team Final Tech Report
- https://docs.google.com/document/d/10RPAiB_wCfEqlzePWlvaZsp6BjeSuPRP_TsW9eVkqEU/edit?usp=sharing

[12] (n.d.) ADXL345 3-axis Accelerometer Data Sheet
- https://drive.google.com/file/d/1wJvjnoiX1-KQEXDOtdXxK6EDPpy6VVqw/view?usp=sharing

[13] (n.d.) Honeywell SMART Position Sensor 100 deg Data Sheet
- https://drive.google.com/file/d/1n9GoiZGoPCNxCLIzkwQu4hh-jnUnLp0B/view?usp=sharing

[14] (n.d.) Teensy 4.1 Microcontroller documentation
- https://www.pjrc.com/store/teensy41.html

[15] (n.d.) User Manual

- https://docs.google.com/document/d/1mzbJ0J7MzUnVulLPzXmTxSjKzZEhG1Y5/edit?usp=sharing

[16] (n.d.) Inertial Testing Procedure
- https://docs.google.com/document/d/1kJurZy5Rmlt9pgg3f9qcnR7MGIRgR46dzYnf-0yaomc/edit?usp=sharing

[17] (n.d.) Team Viewer Remote Desktop
- https://docs.google.com/document/d/1kJurZy5Rmlt9pgg3f9qcnR7MGIRgR46dzYnf-0yaomc/edit?usp=sharing

[18] (n.d.) Python Testing Tools Setup
- https://docs.google.com/document/d/1bwCuV5lJX-5D1dGGSGY11Q_mC1pPDzClWTKUtc605HU/edit?usp=sharing

[19] (n.d.) Remaining Tasks
- https://docs.google.com/document/d/1ecdeRKI8cJe0zdkzhhsniZOlR7Ki8IYFYQ0V4hHgrxc/edit?usp=sharing

[20] (n.d.) Workflow Walkthrough
- https://docs.google.com/document/d/1C_X7ITpnPVH8c_-ZnJ_Rbwotm3vXP_cXGYUhj2JmJFs/edit?usp=sharing

[21] (n.d.) DHT22 Datasheet
- https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf

[22] (n.d.) Midi Range Inductive Angle Encoder
- https://drive.google.com/file/d/13ax3rYI_46NK4k0j9LQoruXowl3f3kFk/view?usp=sharing