DESIGN OF THE 21-METER NETWORK MONITOR AND CONTROL SYSTEM FOR DEEP SPACE NETWORK CROSS SUPPORT

A Thesis

Presented to

the Faculty of the College of Science

Morehead State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Sarah E. Wilczewski

September 26, 2019

Accepted by the faculty of the College of Science, Morehead State University, in partial fulfillment of the requirements for the Master of Science degree.

____
Dr. Benjamin K. Malphrus
Director of Thesis

Master’s Committee: ____________________________________, Chair
Dr. Benjamin K. Malphrus

____
Dr. Charles Conner

____
Kevin Z. Brown

____
Date

DESIGN OF THE 21-METER NETWORK MONITOR AND CONTROL SYSTEM FOR DEEP SPACE NETWORK CROSS SUPPORT

Sarah E. Wilczewski
Morehead State University, 2019

Director of Thesis: ____
Dr. Benjamin K. Malphrus

The Deep Space Network has three sites located around the world. The limited number of sites means that they have more satellites to track than they can handle. When NASAs new rocket, the Space Launch System, launches, it will be carrying thirteen new CubeSats for the Deep Space Network to track. To handle this increased demand, the 21-meter Space Tracking Antenna at Morehead State University is being upgraded to become an auxiliary node on the Deep Space Network. This upgrade includes creating a monitor and control system. In addition to providing data downlinked from the spacecraft, the Deep Space Network provides a host of data about the status of the Antenna. This monitor data includes information such as where the antenna is pointing, the weather at the station, signal to noise ratio, and more. This monitor information is used to ensure that the dish is working properly, and diagnose any problems, such as a loss of signal, that might arise during a satellite pass. The system is being based around a MySQL database and a GUI developed in-house. Development of the Network Monitor and Control system is still ongoing, but the requirements for it have been defined and the modular design simplifies expanding the system to incorporate additional subsystems. The network monitor and control system being developed here at Morehead State University will be used by all future users of the Deep Space Network Node at Morehead. It will also serve as a baseline for any other outside antennas that get added to the Deep Space Network in the future.

Accepted by: ______________________________, Chair
Dr Benjamin K. Malphrus

____
Dr. Charles Conner

____
Kevin Z. Brown

Acknowledgements

I'd like to thank everyone who has helped me through this journey. Specifically, my advisor Jeff Kruth for all the help, support and information he provided. Also, thanks Jay Wyatt, Tim Pham and everyone at JPL who assisted on this project. I'd like to extend my gratitude to everyone who assisted on this project, Zach Wagner, Jeremiah Lowe, Dr. Chuck Conner and Dr. Marc Beck. Finally, I'd like to thank my family for their unwavering support and encouragement over the years.

Table of Contents

Figures

Chapter I Introduction

1. MSU 21 Meter

The 21 Meter Space Tracking Antenna (21M) at Morehead State University (MSU) provides tracking, telemetry and control support for both satellites built at MSU and those built by other universities, private aerospace companies and government agencies. It is also used for radio astronomy observations. The 21M gives excellent experience in ground station operations and in radio frequency systems to the undergraduate students who are the primary operators. It was originally completed in 2006 and in 2012 it was upgraded to be compatible with the Near Earth Network. The Near Earth Network is a network of ground stations located at sites around the world that provide space communications and tracking services to missions operating in the region from Low Earth Orbit (LEO) to the Lunar Surface. The 21M currently operates at UHF, L-band, S-Band, C-Band, X-band and Ku-band[1]. Currently the 21M is able to track satellites in LEO, Geostationary Orbit (GEO), and even

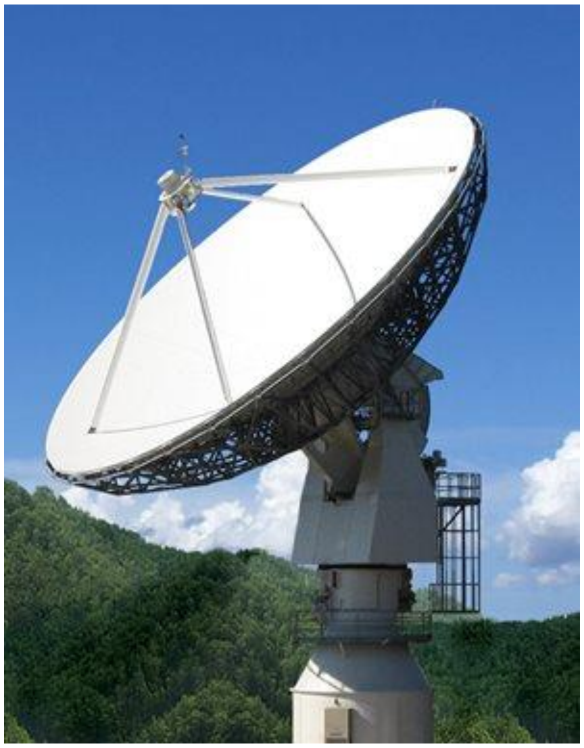


Figure 1 21 Meter Space Tracking Antenna

Source: Morehead State University. *New Antenna and Day Sky*. https://www.moreheadstate.edu /College-of-Science/Earth-and-Space-Sciences/Space-Science-Center/Satellite-Tracking -Telemetry-Control-Services

[1] "Satellite Tracking, Telemetry & Control Services." Morehead State University. https://www.moreheadstate.edu/College-of-Science/Earth-and-Space-Sciences/Space-Science-Center/Satellite-Tracking,-Telemetry-Control-Services (accessed October 15, 2018)

out to as far as the moon, with high transmission power, due to relatively low noise feed systems and a high antenna gain. It has served as the primary ground station for MSU and the University of Kentucky's Ky-Sat 2, Planet Labs Dove 1 and Dove 2 satellites, and MSU's Cosmic X-Ray Background Nanosatellite (CXBN), CXBN 2 and the primary ground station for the JPL CubeSat ASTERIA. It also assisted with the testing and calibration of the Synthetic Aperture Radar on the NASA Lunar Reconnaissance Orbiter. It has also been the secondary ground station for several other satellites in LEO[2]. The 21M is being upgraded to become a node on the DSN with the identifier of DSS-17. These upgrades include a new cryogenic X-band feed, getting DSN receive and transmit systems, a Hydrogen MASER, increasing the automation of the system and developing a Monitor and Control system, which is the focus of this project.

2. The Deep Space Network

The Deep Space Network (DSN) is a network of large radio antennas that supports interplanetary spacecraft and provides radio astronomy observations. The network is comprised of 16 antennas that provide tracking, navigation, and data transmission services among others. The locations of the DSN complexes and MSU are shown below in **Error! Reference source not found.**.

These antennas range in size from 26 meters to 70 meters in diameter. There are three Deep Space Network sites located in Goldstone California, Madrid Spain and Canberra Australia. At each complex there are a variety of antennas, including 34-m beam-waveguide, 34-m high-efficiency, and 70-m antennas.[3] In a beam-waveguide system, the feed horn and support

[2] Wyatt, Jay, and Benjamin Malphrus. "Morehead State University 21 Meter Antenna Upgrade to DSN Compatibility." In *IND CubeSat Communications Briefing and Technical Interchange*, (2015).
[3] Imbriale, William A. *Ch 7 The 34-Meter Research and Development Beam-Waveguide Antenna*. Vol. 4. Large antennas of the Deep Space Network. 2003.

equipment are placed in a stationary room below the antenna, and a system of reflecting mirrors guides the energy between the feed horn and subreflector. They are advantageous because the systems can be simplified since they do not have to tilt, they are easier to access and are sheltered from the elements.[4] The DSN sites are all about 120° apart in longitude to permit constant communication with distant spacecraft.

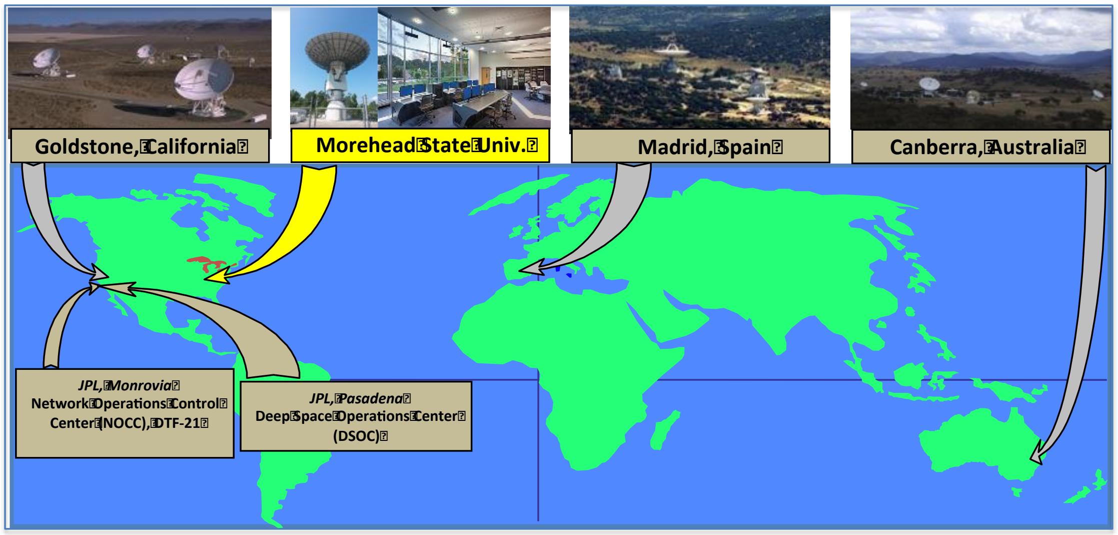


Figure 2 Locations of DSN sites around the Globe

Source: Wyatt, Jay, and Benjamin Malphrus. *"Morehead State University 21 Meter Antenna Upgrade to DSN Compatibility."* In IND CubeSat Communications Briefing and Technical Interchange (2015).

The DSN can operate at S-band with frequencies of 2110–2120 MHz uplink and 2290-2300 MHz downlink, X-band with frequencies of 7145-7190 MHz uplink and 8400-8450 MHz downlink, and Ka-band with frequencies of 34200–34700 MHz uplink and 31800–32300 MHz downlink.[5] The 34-m beam-waveguide antennas have gains of 56.8 dBi at S-band, 68.3 dBi at X-band and 79.0 dBi at Ka-band. The 34-m high-efficiency antennas have gains of 56.0 dBi at S-

[4] Pham, Tim, and Alina Bedrossian. "Deep space network services catalog." DSN No. 820, 100, (2009)
[5] Shin, Dong K. "Frequency and Channel Assignments." DSN No. 810-005, 201, Rev. C, (2014)

band and 68.3 dBi at X-band. The 70-m antennas have gains of 63.5 dBi at S-band and 74.5 dBi at X-band.[6] The current DSN ground network architecture is designed primarily to support large deep-space missions with large budgets and long mission life cycles. As Interplanetary CubeSats, which have smaller budgets and shorter life cycles, become more common the DSN and CubeSats both have to adapt to meet these challenges.[7]

The DSN was established in January 1958 and has been in continuous operation since December 1963. It has served as the primary ground station for missions such as the Surveyor missions, the Apollo missions, the Mariner missions to Venus, the Viking missions to Mars, the Cassini-Huygens mission to Saturn and many more. The 21M will become a node on the DSN with the identifier of DSS-17.[8] As the "about the DSN" website states "They provide the crucial connection for commanding our spacecraft and receiving their never before seen images and scientific information on Earth, propelling our understanding of the universe, our solar system and ultimately, our place within it."[9] Currently 49 missions, ranging in variety from interstellar missions such as Voyager, to planetary missions such as Mars Science Laboratory, and Earth satellite missions such as Spitzer Space Telescope, all use the DSN. In addition to supporting NASA missions, other space agencies around the world, such as the European Space Agency, the Japan Aerospace Exploration Agency, and the Indian Space Research Organization also use DSN services to track spacecraft. When NASAs new rocket, the Space Launch System (SLS),

[6] Pham, Tim, and Alina Bedrossian.
[7] Johnston, Mark D, et al. 2015. "NASA deep space network: automation improvements in the follow-the-Sun Era." *24th International Joint Conference on Artificial Intelligence.* Buenos Aires, Argentina.
[8] Wyatt and Malphrus
[9] NASA. About: Deep Space Network. https://deepspace.jpl.nasa.gov/about/ (accessed April 1, 2017).

launches, it will be carrying thirteen new CubeSats for the DSN to track. To keep up with the increased demand the 21M will become a node on the DSN with the identifier of DSS-17.[10]

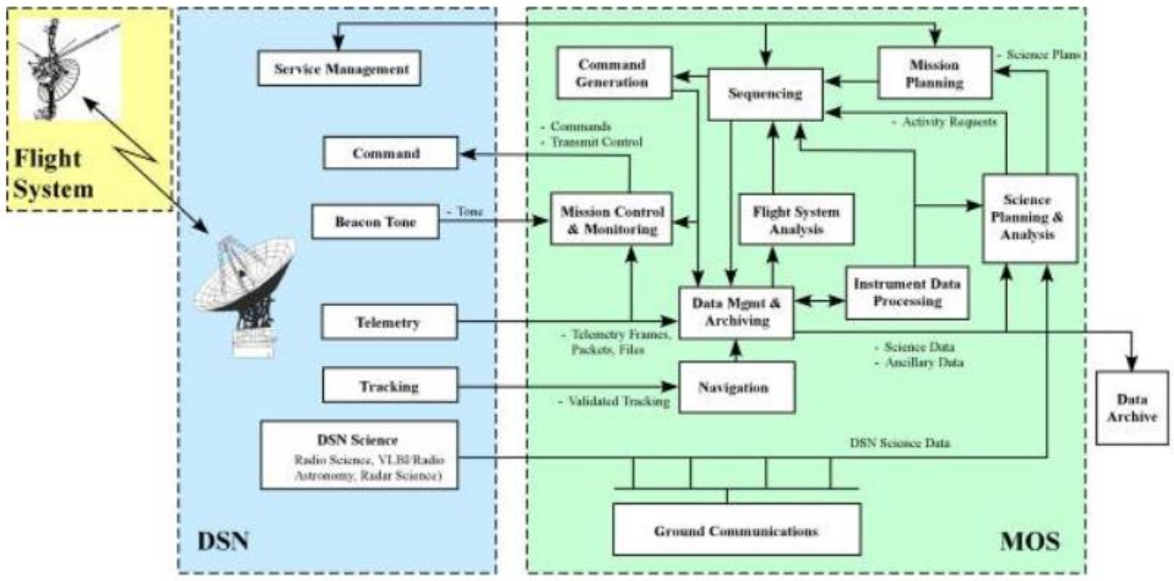


Figure 3 Data flow through the DSN

Source: Pham, Tim, and Alina Bedrossian. "Deep space network services catalog." DSN No. 820, 100 (2009)

When the DSN tracks a satellite, while the specifics of the flow of data may vary, it generally follows the same path. An example of the data flow is shown in **Error! Reference source not found.** above. Telemetry is received at the antenna from the satellite, it flows through the Space Flight Operations Facility at JPL, Data Management System, and through the Network Monitor and Control (NMC) system to the Mission Controllers. Tracking Data goes to navigation and then to the Data Management System. From the Data Management System, both the Tracking and Telemetry data gets sent to an archive and to the Missions Science Planning and Analysis team. After the Science Planning and Analysis team the data goes to Mission

[10] Wyatt and Malphrus

Planning and Sequencing and finally to Command Generation. The commands generated get sent to Data Management and through the NMC system to Mission Control. Mission Control then sends the commands to the antenna to be radiated.[11] While this may vary slightly based on mission requirements, the data flow through the DSN follows generally the same path.

The NMC system is central to the operations of the DSN Antennas, as shown in Figure 4. The NCM system acts as an antenna operator interface and connects to each subsystem in the DSN to gather monitor data and relay commands. It serves as a central repository for monitor data from every subsystem and controls the operations of each subsystem. This system is outlined further in Section 3 below.

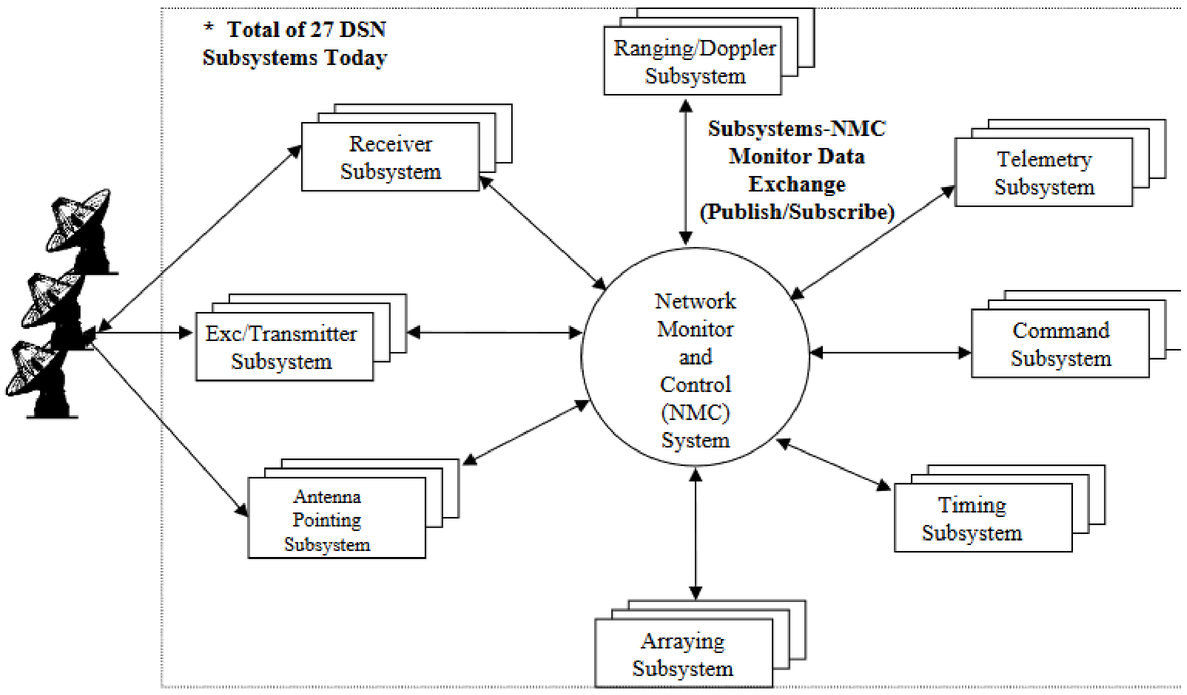

* Total of 27 DSN Subsystems Today
Ranging/Doppler Subsystem
Subsystems-NMC Monitor Data Exchange (Publish/Subscribe)
Receiver Subsystem
Telemetry Subsystem
Exc/Transmitter Subsystem
Network Monitor and Control (NMC) System
Command Subsystem
Antenna Pointing Subsystem
Timing Subsystem
Arraying Subsystem

Figure 4 DSN Monitor and Control Architecture

[11] Pham, Tim, and Alina Bedrossian.

Source: Kurtik, Susan C.. "Multimission Considerations for the NASA Deep Space Network (DSN)." (presentation, Ground System Architectures Workshop, El Segundo, Ca, March 14, 2002)

3. Network Monitor and Control

The Network Monitor and Control (NMC) system is used to observe the operational state of the network and control the subsystems. The data generated for the NMC system includes information such as where the antenna is pointing, the weather at the station, signal to noise ratio, and more. This monitor information is used by both the antenna operators and the Mission operations team to ensure that the dish is working properly, and diagnose any problems, such as a loss of signal, that might arise during a pass. The NMC system provides an interface for Antenna operators to monitor the status of the whole system during a satellite track and to send commands to the system. The monitor data can also be used by both antenna operators and mission users for post-track diagnosis of any issues with data delivery.[12]

Monitor Data is gathered from every subsystem in the network about the health and operation of the antenna and subsystems during a satellite track. The Antenna Control Assembly (ACA) and Antenna Pointing and Control Assembly (ACPA) generates data such as where the antenna is pointing, pointing error, weather conditions, scan type and tracking status. The Connection Engine (CE) generates data such as the antenna number, spacecraft number and subsystem list. The Downlink Tracking and Telemetry (DTT) System generates data such as carrier frequency, subcarrier frequency and symbol rate, actual and predicted values, lock status for each of those, decoder type and status, ranging parameters, and frames received and dropped. The Uplink

[12] Pechkam, Paul. "Monitor and Control Software for Ground Systems in the Deep Space Network." (July 22, 2002).

(UPL) system generates data such as frequency, modulation, transmitter power output, ranging parameters and Command Link Transmission Units sent, rejected, aborted.[13]

For each spacecraft being tracked by the DSN, the latest available monitor data for the relevant tracking station is periodically packaged into a Standard Format Data Unit (SFDU) and delivered to the mission users. The monitor data record is in a Compressed Header Data Object (CHDO) structured SFDU format. Each monitor parameter is formatted into a little structure similar in concept to a packet containing an identifier, known as a channel ID, and a monitor data value. Each CHDO contains one or more of these monitor packets. Each SFDU then consists of a fixed-length header and the variable-length data CHDO. These SFDUs are created at 5-second intervals then encapsulated in a Standard DSN Block. The Standard DSN Block consists of a 20-byte DSN Data Delivery header, the SFDU and a 2-byte DSN Data Delivery trailer.[14] The data provided, in its various formats, is gathered from every DSN subsystem by the Monitor Data Service. The data collected by the 21M NMC system will be used for both real-time monitoring of subsystems and for post-track diagnosis of any issues that arise by both antenna operators and mission users. Accessing the data requires use of a C language based, NASA-proprietary library. A GUI is used during the tracking session to issue directives to the NMC and subsystems. Everything is recorded in the NMC logs, which are stored in a database.[15] All subsystems in the DSN are commanded using commands (called directives) which are either manually entered by

[13] Levister, Kathrine, and Ronald Norman. *"0158-Monitor DSN Mission Monitor Data Interface*." Pasadena, CA: Jet Propulsion Laboratory (December 1, 2015).
[14] Levister, Kathrine, and Ronald Norman. *"0158-Monitor DSN Mission Monitor Data Interface*." Pasadena, CA: Jet Propulsion Laboratory (December 1, 2015).
[15] Choi, Josh, Rishi Verma, and Shan Malhotra. *"Achieving Fast Operational Intelligence in NASAs Deep Space Network Through Complex Event Processing." SpaceOps 2016 Conference* (May 13, 2016). https://doi.org/10.2514/6.2016-2375, 7

an operator or, more commonly, generated by the NMC automation software.[16] The NMC automation software is called Temporal Dependency Network (TDN) which is a directed acrylic graph containing the timing and behavioral knowledge needed to perform a specific task. The TDN is broken up into discrete events that must be performed or omitted before the graph can be further traversed, these are called TDN Blocks. These blocks are the fundamental elements of control of the DSN and perform such tasks as sending directives to ground equipment, issuing alerts for antenna movements and checking equipment status. Each block contains entry and exit conditions. The knowledge base to build the TDNs is developed from the Sequence of Events (SOE) which describes in detail all events that will occur during a spacecraft pass.[17] A NMC system had to be developed for the 21M to complete the upgrade of the antenna to Deep Space Network compatibility. As with the DSN system the data generated for the 21M NMC system will include information such as where the antenna is pointing, the weather at the station, signal to noise ratio, and more.[18] However, due to MSU's less complex system, it will only be a subset of the information normally provided by the DSN. This monitor information will be used to ensure that the antenna is working properly, and diagnose any problems, such as a loss of signal, that might arise during a pass. Connecting these subsystems will allow Mission operators to gather all the data needed to evaluate the status of the Antenna throughout a Satellite Track. It will also allow Antenna Operators to control the systems equipment from a central GUI. It will

[16] Choi, Verma, and Malhotra, 8
[17] Boker, Eva. 2001. "*Automating Operations for NASA's Deep Space Network (DSN)*." IEEE. https://trs.jpl.nasa.gov/bitstream/handle/2014/16215/00-2162.pdf?sequence=1&isAllowed=y. (Accessed February 21, 2019), 5
[18] Pechkam, Paul. "Monitor and Control Software for Ground Systems in the Deep Space Network." (July 22, 2002).

also allow both antenna operators and mission users to analyze archived data and diagnose any issues with data delivery after a Satellite track.

Chapter II Requirements

1. Functional Requirements

Before design work could begin, a set of requirements had to be defined. These requirements have been used to guide the design of the 21M NMC system. First, the NMC system must be compatible with DSN and 21M infrastructure. It must be able to operate continuously for long periods of time when necessary. The NMC system must allow the use of the DSS-17 as an experimental station on the DSN, both now and into the future. The NMC system must collect and store data from the subsystems outlined in section II.2. It must display data from the subsystems outlined in section II.2 for Antenna Operators. The NMC system must be able to control the subsystems outlined in section II.2. The system must be able to be replicated by or distributed to any antennas that may be brought into the DSN similarly to the 21M in the future. Finally, Mission users must be able to access, modify and interpret the monitor data as necessary. Meeting these requirements will allow the 21M to be compatible with the DSN both now and into the future. As the DSN improves and changes it will be important for the 21M to be able to grow and adapt with it. Having a flexible system will allow these changes to be made in the future.

2. Systems to be connected

Each of the subsystems of the 21M provide data about its health and status which are collected, displayed and archived by the 21M NMC system. They will also be controlled by the 21M NMC system. Connecting these subsystems will allow Mission operators to gather all the data needed to evaluate the status of the Antenna throughout a Satellite Track. It will also allow

Antenna Operators to control the systems equipment from a central GUI. It will also allow both antenna operators and mission users to analyze archived data and diagnose any issues with data delivery after a Satellite track. The systems that will be connected to the NMC system are briefly described below. The Low Noise Amplifier (LNA), which is a cryogenically cooled, first stage downlink amplifier, is connected via ethernet. Through a Raspberry Pi Microcontroller provided by MSU, most of the feed equipment will be connected to the system. This equipment includes the downconverter, which translates the signal from the X-band frequencies to an intermediate frequency for more efficient translation and distribution, the Waveguide Switches, which switch between the Test Translator and the full DSS-17 RF path, and the Test Translator and Noise Inject, which calibrate the system. The Klystron Power Amplifier (KPA), which is a high-powered, 2 kW, uplink power amplifier, is connected via ethernet. The Waveguide Dehydrator, which removes moisture from the waveguide to reduce losses. The Upconverter Raspberry Pi, provided by MSU, for interfacing with the upconverter that translates the uplink signal from an intermediate frequency to X-band, is connected via ethernet. Through the Op Console, the Antenna Control Unit (ACU) which controls the pointing of the 21M, is connected via RS-422. Through a MSU provided Raspberry Pi Microcontroller the MASER, which is a high accuracy timing source. The 100 MHz Level, which is a monitor point for MASER after it travels to the Lower Equipment Room (LER), is also connected via ethernet. The subsystems provided by JPL are connected through an aggregator they developed to collect, format and transfer data from the JPL subsystems. These subsystems include the Downlink Tracking and Telemetry Subsystem (DTT), which is hardware and software that take RF input through an Analog-Digital Converter (ADC), then turns the bits into telemetry, range, and doppler estimates for tracking, the Data Capture and Delivery Subsystem (DCD), which captures missions tracking and telemetry data at the antenna site and relays it to a central site at the JPL Space Flight Operations Facility, and the Uplink Tracking and Command (UPL), which handles data processing for uplink communications with spacecraft. Consolidates command, exciter, transmitter, and uplink ranging system. These subsystems are described in more detail in Chapter III below.

Chapter III Methodology

1. Overview

The primary task involved in this project is creating a monitor data system which will be gathering, processing, displaying, and archiving data related to the state of the various subsystems of the 21M, in addition to controlling the subsystems. Each subsystem will either generate CSV files or will have software to place the data into CSV files. These CSV files will be sent via Ethernet to the DSS-17 NMC computer in the Mission Operations Center. Once there, the data is displayed on the NMC GUI and archived in a MySQL database, both described in Section 8 below. Connecting these subsystems will allow Mission operators to gather all the data needed to evaluate the status of the Antenna throughout a Satellite Track. It will also allow Antenna Operators to control the systems equipment from a central GUI. It will also allow both antenna operators and mission users to analyze archived data and diagnose any issues with data delivery after a Satellite track. The subsystem equipment is spread between 4 locations, the feed of the 21M, the Azimuth platform, the Lower Equipment Room at the 21M and the Mission Operations Center. These locations are connected via Ethernet hubs. The subsystems are comprised of three relatively broad classes of equipment – commercial, MSU provided and JPL provided. In order to control and gather the data from the subsystems a new software system had to be developed. After working with and evaluating an open source software Open MCT to collect and display the data, it was determined that developing software in house would be a better option. Programing language and data storage options were reviewed against the established requirements, with a focus on the available teams' technical capabilities, the need to read various forms of data from various sources, the portability of the software and alignment with future JPL goals. In the end, an approach of using a MySQL database and a custom JAVA GUI was chosen. Each subsystem generates monitor data which is compiled into a CSV, either directly by the subsystem or by in-house software. Each CSV is delivered via ethernet to the NMC computer in the MOC. There the data will be archived in a MySQL database and displayed on a custom JAVA GUI. The MySQL database can be analyzed by operators at MSU or relevant data can be extracted from the database and sent to Mission Users to assist their analysis.

2. NMC Block Diagram

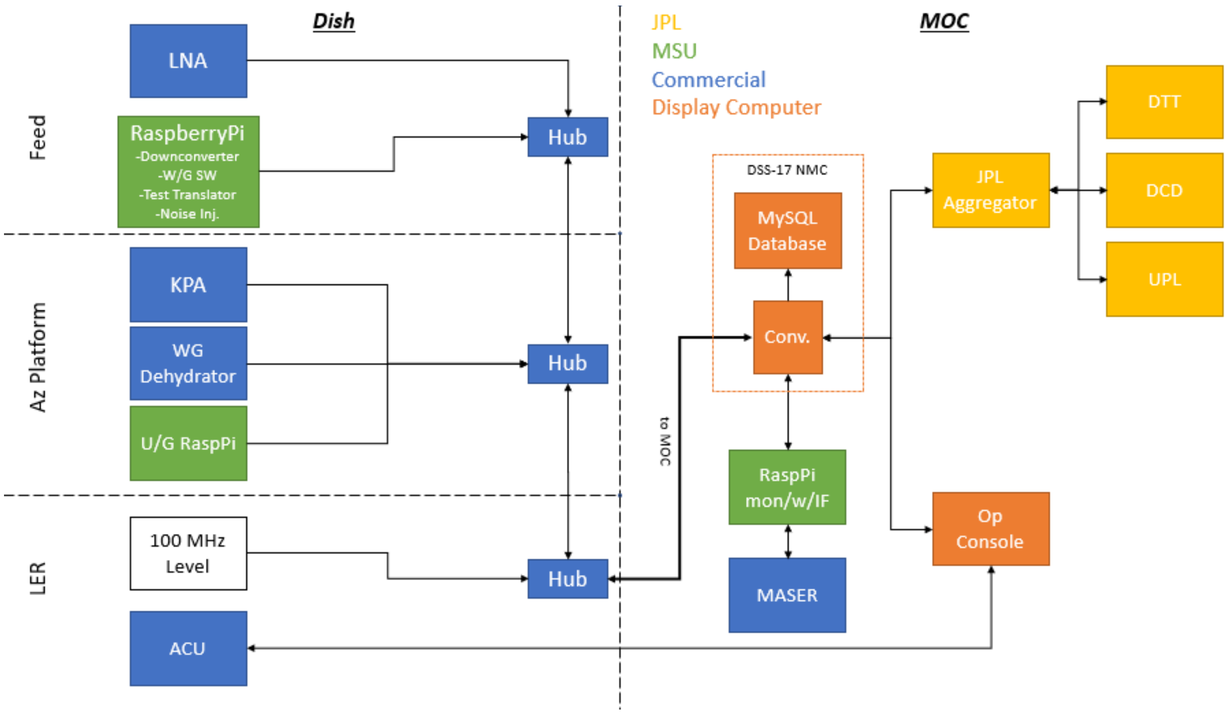


Figure 5 NMC Block Diagram

Source: Kruth, Jeffrey A. "Corrected Block Diagram of NMC." (April 5, 2019).

The block diagram shown in Figure 5 above describes the architecture of the NMC system and how the data flows from the different subsystems to the MySQL database. The NMC system collects monitor data from the Low Noise Amplifier, Downconverter, Waveguide Switches, Test Translator and Noise Inject via a Raspberry Pi, Klystron Power Amplifier, Waveguide Dehydrator, Upconverter, Antenna Control Unit, MASER, Downlink Tracking and Telemetry Subsystem, Data Capture and Delivery Subsystem, and Uplink Tracking and Command. During a satellite track, the NMC will collect data about the health and status of each of these systems and send commands to these systems. Each subsystem will either generate CSV files or will have software to place the data into CSV files. These CSV files will be sent via Ethernet to the DSS-17 NMC computer in the Mission Operations Center. Once there, the data is displayed on the NMC GUI and archived in a MySQL database, both described in Section 8 below. Most of the systems are connected over Ethernet. The different colors indicate the different equipment providers. There is equipment provided by JPL which is shown in yellow. Equipment provided by MSU is shown in green. Commercial equipment is shown in blue. The equipment is spread between 4 locations, the feed of the 21M, the Azimuth platform, the Lower Equipment Room at the 21M and the Mission Operations Center. Each subsystem will either generate CSV files or will have software to place the data into CSV files. These CSV files will be sent via Ethernet to the DSS-17 NMC computer in the Mission Operations Center. Once there, the data is displayed on the NMC GUI and archived in a MySQL database.

3. The 21M Feed

Interface Overview

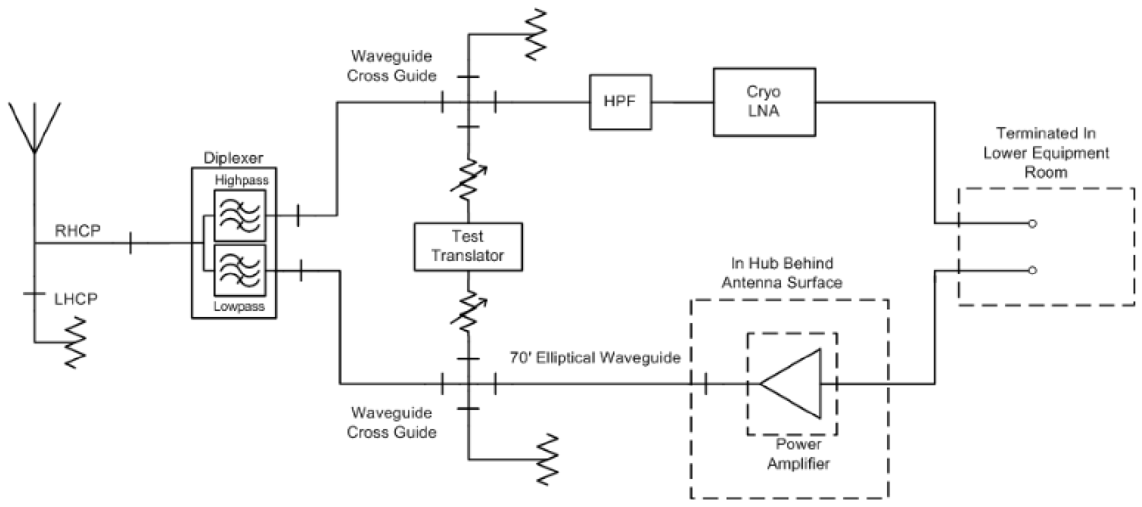


Figure 6 21M Feed Block Diagram

Source: Kruth, "Interface Control Document for Morehead State 21 M DSN X-band Upgrade Embedded Feed Electronic Monitoring & Control Subsystem (EMCS)," (April 5, 2019).

In the 21M feed is the LNA, Downconverter, Waveguide Switch, Test Translator and Noise Inject. Also, in the feed is a Raspberry Pi Microcomputer and interface board which controls the Downconverter, Waveguide Switch, Test Translator and Noise Inject. The Raspberry Pi and Interface board are connected via a 40 pin ribbon cable with two row IDC type connectors. The interface board is connected to the 40 pin I/O port on the Raspberry Pi. The port is then separated into 10 control groupings[19]. These groupings are based on function and are: control of E/M switches for RF signal routing, monitoring the position status of the RF switches, activation of the Noise Diode for System Temperature monitoring, activation of the Test

[19] Kruth, Jeffrey. "Interface Control Document for Morehead State 21 M DSN X-band Upgrade Embedded Feed Electronic Monitoring & Control Subsystem (EMCS)," (April 5, 2019).

Translator for system loopback to calibrate ground range offset, control of the Bit Attenuator, monitoring of transmit power, monitoring the status of the internal power supplies, monitoring of the Phase Lock Oscillator (PLO) status for the Down Converter Local Oscillator and test translator offset Local Oscillator, temperature monitoring of three selected physical sites within the feed, and resetting the LNA to remote, if needed[20]. The Raspberry Pi and LNA are connected via Ethernet to a hub. The hub is then connected via Ethernet to the Azimuth Platform and the Lower Equipment Room. The subsystems located at the 21M feed are described below.

Low Noise Amplifier

While a typical amplifier increases the power of both the signal and the noise present at its input, a Low Noise Amplifier amplifies a signal while minimizing additional noise. They are typically placed immediately after the signal source to reduce noise. The LNA in the 21M is a compact cryogenically cooled LNA from Callisto. It is designed to be compact, very low maintenance and low power. The LNAs noise temperature is less than 20K, meaning the overall system temperature is less than 100K.[21] A Picture of the LNA is shown in Figure 7. The Monitor Data gathered by the LNA includes alarms and various temperatures.

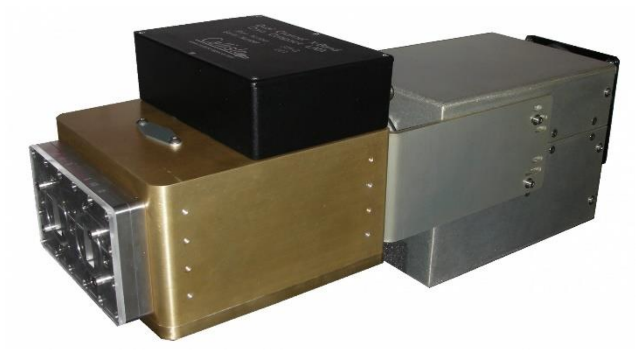


Figure 7 Image of the Callisto Cryogenically Cooled LNA

Source: "Compact Cryogenic LNA," Callisto, 2019, http://www.callisto-space.com/en/page/compact-cryogenic-lna-2.php

[20] Kruth, "Interface Control Document for Morehead State 21 M DSN X-band Upgrade Embedded Feed Electronic Monitoring & Control Subsystem (EMCS)," (April 5, 2019).
[21] "Compact Cryogenic LNA," *Callisto*, 2019, http://www.callisto-space.com/en/page/compact-cryogenic-lna-2.php

Embedded Feed Monitor and Control

Raspberry Pi Microcomputer and interface board which controls the Downconverter, Waveguide Switches, Test Translator and Noise Inject. The two boards are connected via a 40 pin ribbon cable with two row IDC type connectors. The systems connected to the Raspberry Pi are described in the paragraphs below[22]. The data read by the Raspberry Pi and sent to the NMC includes the Downconverter PLO status, Translator PLO status, Power Meter output, Power Supply Voltages, Physical Temperatures and the status of various switches in the feeds.
The Raspberry Pi is connected to the feeds test translator system using a noise injection to measure and quantify system noise temperature. It also has a test translator to help measure the round-trip time delay thru the system in order to determine the time offset for spacecraft transponder range measuring mode. An RF relay selects either the noise or translator signal to be inject into the receiver via a coupler. The "normally closed" path is set to the noise source. When noise is desired this relay must be "off" and the noise enable relay for the diode noise source must be "on". The step attenuator function is then called, and the appropriate bit pattern loaded to the attenuator to provide the proper noise level setting. This system was designed in house at Morehead State University[23]. The Noise Inject and Test Translator system, along with the downconverter and waveguide switches, is monitored and controlled by a Raspberry Pi. The block diagram of this system is shown in Figure 8.

[22] Kruth, "Interface Control Document for Morehead State 21 M DSN X-band Upgrade Embedded Feed Electronic Monitoring & Control Subsystem (EMCS)," (April 5, 2019).
[23] Kruth, "Interface Control Document for Morehead State 21 M DSN X-band Upgrade Embedded Feed Electronic Monitoring & Control Subsystem (EMCS)," (April 5, 2019).

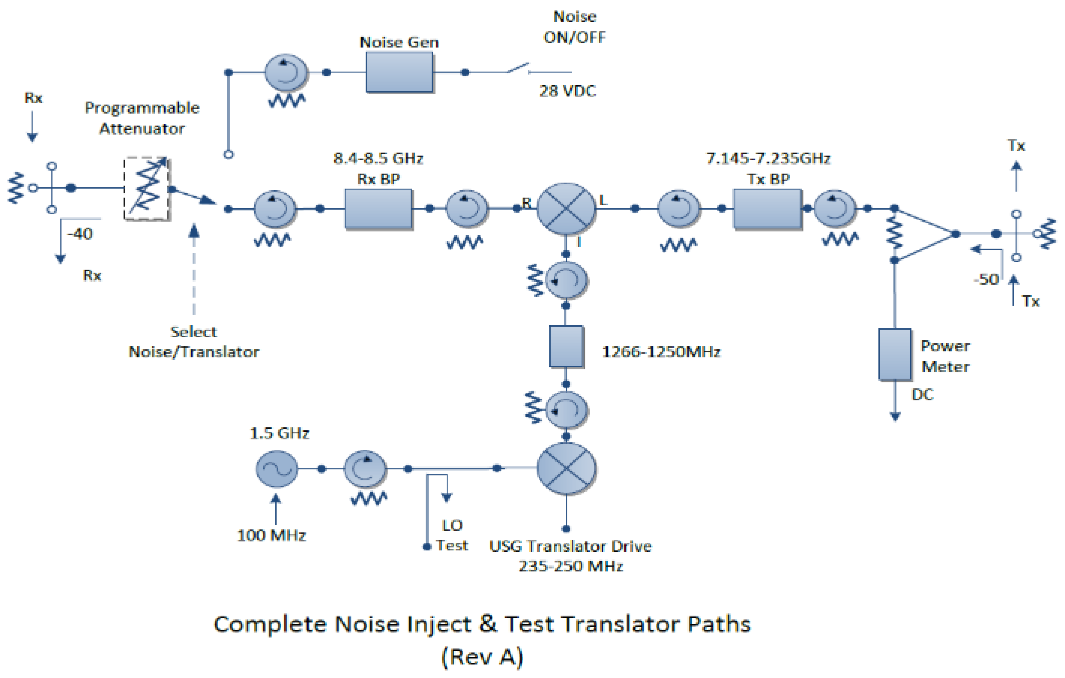


Figure 8 Block diagram of the Test Translator and Noise Inject

Kruth, "Interface Control Document for Morehead State 21 M DSN X-band Upgrade Embedded Feed Electronic Monitoring & Control Subsystem (EMCS)," (April 5, 2019)

A downconverter is a device using mixers and local oscillators that takes a high frequency signal and converts it to a lower frequency. It takes the difference between the input frequency and local oscillator frequency to create the desired lower frequency output signal. The downconverter being used in the 21M system was built in house. It uses an 8.1 GHz local oscillator for modulating the signal to create an approximately 300 MHz intermediate frequency that will be fed into the DTT[24]. The downconverter, along with the Waveguide Switches, Test Translator and Noise Inject, is monitored and controlled by a Raspberry Pi.

[24] Kruth, "Interface Control Document for Morehead State 21 M DSN X-band Upgrade Embedded Feed Electronic Monitoring & Control Subsystem (EMCS)," (April 5, 2019).

4. Azimuth Platform

Interface Overview

The Azimuth platform is the lower deck of the 21M. Located on the Azimuth platform is the Klystron Power Amplifier (KPA), Waveguide Dehydrator and Upconverter Raspberry Pi. The KPA is connected to a hub via Ethernet. The Waveguide Dehydrator is also connected to the same hub via Ethernet. It is recommended that Ethernet communications for the Dehydrator run at 10Base-T and half duplex, there is a web interface for configuration[25]. The Upconverter Raspberry Pi is connected to the same hub as the others via Ethernet. This hub is then connected to the hubs in the Feed and Lower Equipment Room. The subsystems located on the Azimuth Platform are described in detail below.

Klystron Power Amplifier

The Klystron Power Amplifier will serve to amplify the RF uplink signal. It is a high-powered uplink power amplifier. A klystron is a high-power microwave vacuum tube that uses the kinetic energy of an electron beam to amplify a high-frequency signal. They are generally high gain and have a high output power but have a limited bandwidth. The KPA being used in the 21M was built by Communications & Power Industries. It operates at 7.9 GHz and a 30 MHz bandwidth. It has an output power of 3.4 kW but will be operated at closer to 2 kW. The gain the KPA will provide is about 50 dB[26]. The monitor data generated by the KPA includes the RF output, reflected RF, beam and heater voltages, body, beam and heater currents, temperatures of the cabinet, outlet and inlet, attenuation, beacon power level, and online status.

[25] ETI. "Netcom Model ADH Automatic Dehydrator with Ethernet Quick-Reference Installation Manual." South Bend, Indiana USA. (2019)
[26] Communications & Power Industries Canada Inc. “X-Band Klystron 3.4 kW.” June 8, 2008. https://www.cpii.com/docs/datasheets/144/X-Band-3.3kw.pdf.

Waveguide Dehydrator

Unpressurized Waveguides allow the entry of moisture through leaking seals and cracks, the moisture can condense causing corrosion and reduce system performance. The Waveguide dehydrator prevents the accumulation of moisture in transmission lines by maintaining the pressure and dew point of the air inside the line[27]. The waveguide being used in the 21M system is from ETI. Air is dehydrated by passing it through a drying canister containing the drying agent, its default setting provides dry air at 34.5 mbar at 26 cubic feet per hour. The target limits that are set by default in dehydrator 21 mbar to 34.5 mbar. The waveguide dehydrator has various user configurable alarms[28]. The monitor data generated by the dehydrator includes the pressure, duty cycle, temperature, hours operating, statuses, and various alarms.

Upconverter

An upconverter is similar to the downconverter, it uses mixers and local oscillators that takes a high frequency signal and converts it to a higher frequency. It takes the difference between the input frequency and local oscillator frequency to create the desired higher frequency output signal. The upconverter being used in the 21M system was built in house and is connected to the system through a Raspberry Pi. The UPL outputs an intermediate frequency of 300 MHz that the upconverter then translates the signal to the transmit frequency of ~7.1 GHz. The upconverter is monitored and controlled through a Raspberry Pi. The monitor data generated by the upconverter includes the Upconverter PLO status.

[27] ETI. "Netcom Model ADH Automatic Dehydrator with Ethernet Quick-Reference Installation Manual." South Bend, Indiana USA. (2019)
[28] ETI. "Netcom Model ADH Automatic Dehydrator with Ethernet Quick-Reference Installation Manual." South Bend, Indiana USA. (2019)

5. Lower Equipment Room

Interface Overview

The Lower Equipment Room (LER) is located inside the base of the 21M and houses much of the 21M's digital and RF equipment. The 100 MHz reference level is located in the LER. It is ported up from the MASER over fiber. The 100 MHz reference is then connected via Ethernet to a hub and distributed to the other equipment at the 21M (on the Azimuth platform and Feed). The 100 MHz reference is described in section 6 below along with the Hydrogen MASER. Also in the LER is the Antenna Control Unit, which is connected via Ethernet to the Ops Console in the Mission Operations Center. The subsystems located in the Lower Equipment room are described below.

Antenna Control Unit

The Antenna Control Unit (ACU) from Vertex RSI is the primary operator interface for antenna control. It sends position commands, gathers and reports system status and faults, can create trajectory predictions, provides the operator interface, and includes the supervisory computer interface.[29] It has simultaneous positioning modes for each axis of movement. It gives an operator the ability to command operational antenna modes, set parameters, and monitor fault/status information via a windowing environment. the ACU gathers and reports system status and issues position commands for AZ/EL/POL axes. These commands are internally generated by the ACU's pointing mode algorithms. The ACU can be operated either locally or remotely, and currently is primarily operated remotely from the Mission Operations Center.[30]

[29] VertexRSI. "Antenna Control System Operation and Maintenance Manual." (February 2005).
[30] VertexRSI. "Antenna Control System Operation and Maintenance Manual." (February 2005).

While it can be gathered directly from the ACU, the monitor data will be gathered through the Ops Console, described below. The monitor data generated by the ACU includes azimuth, elevation, polarization, mode, signal and an event log.

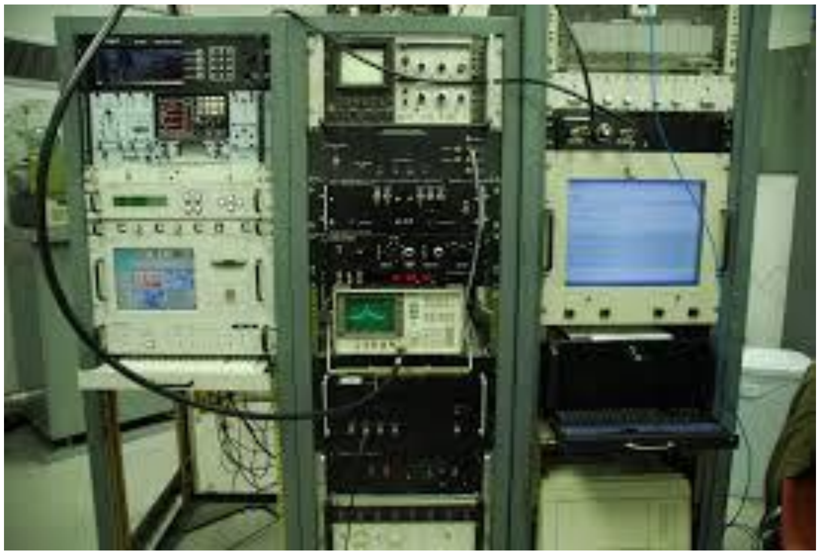


Figure 9 21M ACU

Source: Malphrus, Benjamin K. "21 M Space Tracking Antenna." January 2017. https://www.moreheadstate.edu/getattachment/College-of-Science/Earth-and-Space-Sciences/Space-Science-Center/Satellite-Tracking,-Telemetry-Control-Services/satellite-(2).pdf.aspx?lang=en-US.

6. Mission Operations Center

The hub in the LER connects via Ethernet to the Mission Operations Center. The Mission Operations Center is where the majority of the equipment is located. Most of the operation of the 21M is conducted in the MOC. The Antenna Control Unit is connected via Ethernet to the Ops Console in the Mission Operations Center. The hub in the LER is connected to the NMC computer via Ethernet. The NMC computer runs a conversion algorithm which then sends the data to a MySQL Database. The Ops console is also connected to the NMC computer conversion algorithm via Ethernet. The Hydrogen MASER is connected via serial to a Raspberry Pi. The

MASER Raspberry Pi is connected to the NMC computer conversion algorithm via Ethernet. Finally, the DTT, DCD and UPL are connected via Ethernet to a Raspberry Pi running a data aggregator for JPL. The Aggregator is then connected to the NMC computer via Ethernet.

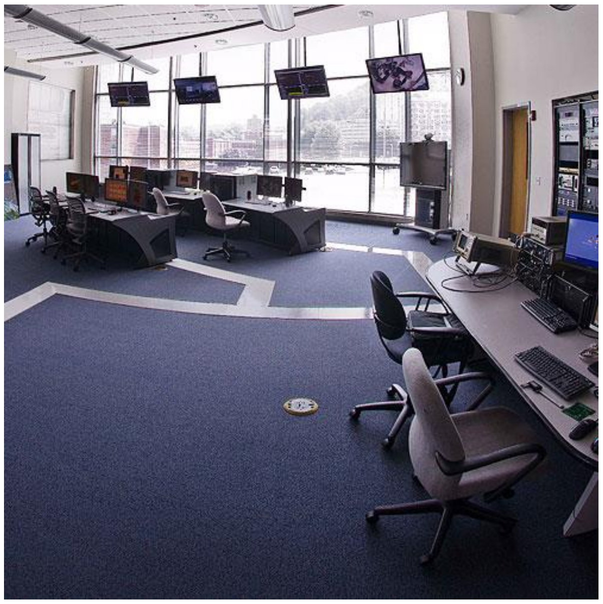


Figure 10 Morehead State Mission Operations Center

source: Morehead State University. "Space Mission Operations Center." 2019. https://www.moreheadstate.edu/College-of-Science/Earth-and-Space-Sciences/Space-Science-Center/Laboratories-Facilities/Space-Mission-Operations-Center.

Downlink Tracking and Telemetry Subsystem

The Downlink Tracking and Telemetry Subsystem (DTT) is a collection of hardware and software that processes the Radio Frequency (RF) input from the microwave subsystem to the output of telemetry frames and to tracking range and Doppler estimates. The DTT consists of receiver, telemetry processor, and downlink ranging assemblies.[31] The DTT demodulates the

[31] Levister, Kathrine, and Ronald Norman. *"0158-Monitor DSN Mission Monitor Data Interface*." Pasadena, CA: Jet Propulsion Laboratory (December 1, 2015).

carrier, the subcarrier, and the symbol stream. It converts the symbols to bits and performs frame synchronization and decoding. It also performs ranging correlation and provides downlink carrier phase data for Doppler measurements. It is responsible for the digitization of the intermediate frequency, demodulation and decoding of the signals and for downlink ranging. It can A Sun Sparc running Solaris 8 controls the DTT. The DTT is responsible for acquisition of the carrier and subcarrier frequency and phase, the acquisition of symbols, and the tracking of these values, the symbol-to-bit decoding, frame synchronization and frame formatting. It's ranging functionality includes the measurement of the carrier Doppler frequency and phase, and the ranging signal phase.[32] The DTT can support PSK, BPSK, QPSK and OQPSK modulation. It can decode Convolutional, Reed-Solomon and Turbo encoding. It typically has a frame rejection rate of 10^{-4} or 10^{-5} but the error rates depend on the SNR of the spacecraft[33]. The data generated by the DTT is sent to an aggregator created by JPL, which collects data from all the DSN provided subsystems and puts that data into CSV format before transferring it to the MySQL database. The JPL provided equipment outputs its data in its own unique format. In order to get the data in a standard CSV format JPL will provide software, which will run on a Raspberry Pi, to aggregate and translate the data to a CSV format. The DTT generates data such as carrier frequency, subcarrier frequency and symbol rate, actual and predicted values, lock status for each of those, decoder type and status, downlink ranging parameters, and frames received and dropped.

[32] O'Dea, J. Andrew. "Downlink Consolidated Software (DCS) Subsystem Operations Manual." DSN No. 837-036, Rev. X, (February 24, 2017).
[33] Pham, Tim, and Alina Bedrossian. "Deep space network services catalog." DSN No. 820, 100, (2009).

Uplink Tracking and Command

The Uplink Tracking and Command Subsystem (UPL) handles data processing for uplink communications with spacecraft, and consolidates command, exciter, transmitter, and uplink ranging system. The UPL consists of the following assemblies, which are controlled by the Uplink Control (ulc): the Command Control Processor (ccp), the Command Modulation Generator (cmg), the Exciter-Transmitter Controller (etc), and the Uplink Ranging Controller (urc).[34] The UPL receives directives and command data from the end user, modulates the uplink carrier with the command data and ranging modulation, amplifies the modulated carrier for transmission, and provides uplink carrier phase data for Doppler measurement and ranging phase data for ranging measurement. The UPL runs on a Sun X5 Ultra with a Solaris 10 operating system and a Dell R-210. The 21M system will be using the Block V Exciter instead of the Block VI Exciter which is being used in the DSN. The UPL is involved signal generation, command, tracking, ranging, and along with the upconverter is involved in frequency translation. It is broken up into three primary functions (the full DSN system has four). These functions are Command, Ranging and Exciter (S/X band) functions. The version being used at Morehead State is a slightly different than the one currently being used in the DSN. The data generated by the UPL is sent to an aggregator created by JPL, which collects data from all the DSN provided subsystems and puts that data into CSV format before transferring it to the MySQL database. The UPL supports BPSK and PSK modulation. It accepts CCSDS formatted Command Link Transmission Units (CLTU).[35] The JPL provided equipment outputs its data in its own unique

[34] Pham, Tim, and Alina Bedrossian. "Deep space network services catalog." DSN No. 820, 100, (2009).
[35] Zia, Susan. "Uplink Consolidated Software (UCS) Subsystem Operations Manual." DSN No. 837-049, Rev. Q, (September 13, 2016).

format. In order to get the data in a standard CSV format JPL will provide software, which will run on a Raspberry Pi, to aggregate and translate the data to a CSV format. The UPL generates data such as frequency, modulation, subcarrier waveform, ranging parameters and Command Link Transmission Units sent, rejected, aborted.

Data Capture and Delivery

The Data Capture and Delivery Subsystem (DCD) captures mission telemetry and tracking data, then manages the delivery of said data. It gathers all data generated by the DTT and UPL. The DCD then manages the delivery of this data based on grade of service and priority. It also provides short-term buffering of the data and maintains a catalog of the data. It is involved with the delivery of standard and beacon mode telemetry data to the DTT. It delivers tracking data from the UPL. The delivery of raw telemetry accountability data from the DTT is also managed by the DCD.[36] The latency of the data delivery is determined by the type of data. High-rate Timely data has a latency of 2 seconds, the latency is 5 seconds for Tracking Complete data, Nominal Timely data has a delivery latency of 8 seconds, 3 minutes for CFDP Complete data and Immediate Complete data has a latency of 10 minutes. [37] The data generated by the DCD is sent to an aggregator created by JPL, which collects data from all the DSN provided subsystems and puts that data into CSV format before transferring it to the MySQL database. The JPL provided equipment outputs its data in its own unique format. In order to get the data in a standard CSV format JPL will provide software, which will run on a Raspberry Pi, to aggregate and translate

[36] Pham, Tim, and Alina Bedrossian. "Deep space network services catalog." DSN No. 820, 100, (2009).
[37] Garrett, Shiela. "Data Capture and Delivery Subsystem (DCD) Operations and Maintenance Training." (Spring 2017.)

the data to a CSV format. The monitor data gathered by the DCD includes the DCD number, the number of frames transferred, rejected frames and status.

Hydrogen MASER and Frequency distribution

MASER stands for Microwave Amplification by Stimulated Emission of Radiation. By sending excited Hydrogen atoms into a tuned, resonant cavity, a microwave signal that is locked to the resonance frequency of the hydrogen atom is generated. The resonant frequency of Hydrogen is 1,420,405,752 Hz. Hydrogen MASERS have extraordinary short-term stability. They typically have a short-term stability of 1×10^{-12} over 1 second.[38] The MASER being used by the 21M, the VLG-10 (P2) Hydrogen MASER, is on permanent loan from Massachusetts Institute of Technology. The MASER is used to create a 10 Hz and 100 Hz reference signal. The 10 Hz and 100 Hz reference signals are fed into a Master Clock Assembly which distributes time code signals to the various subsystems. The 100 MHz reference signal is also piped up to the LER for use by the equipment there. Testing performed on the MASER before arrival at MSU showed it had a Allan Deviation of 2.82843×10^{-13} over 1 second, 3.53553×10^{-14} over 10 seconds, and 9.19239×10^{-15} over 100 seconds. These meet the DSN specs which are 3.29×10^{-13} over 1 second, 1.13×10^{-14} over 10 seconds and 3.29×10^{-14} over 100 seconds.[39] A Raspberry Pi serves as the interface between the MASER and the NMC system. The overall frequency and timing system is shown in Figure 11. The monitor data generated by the MASER includes various temperatures, the frequency of the maser and status.

[38] Lombardi, Michael A. "Time and Frequency from A to Z, H." *NIST*. September 26, 2016. https://www.nist.gov/time-and-frequency-services/h.
[39] Payne, Cadance, et al. "The Resurrection And Repurposing Of The Vlg-10 (P2) Hydrogen Maser." August 2016.

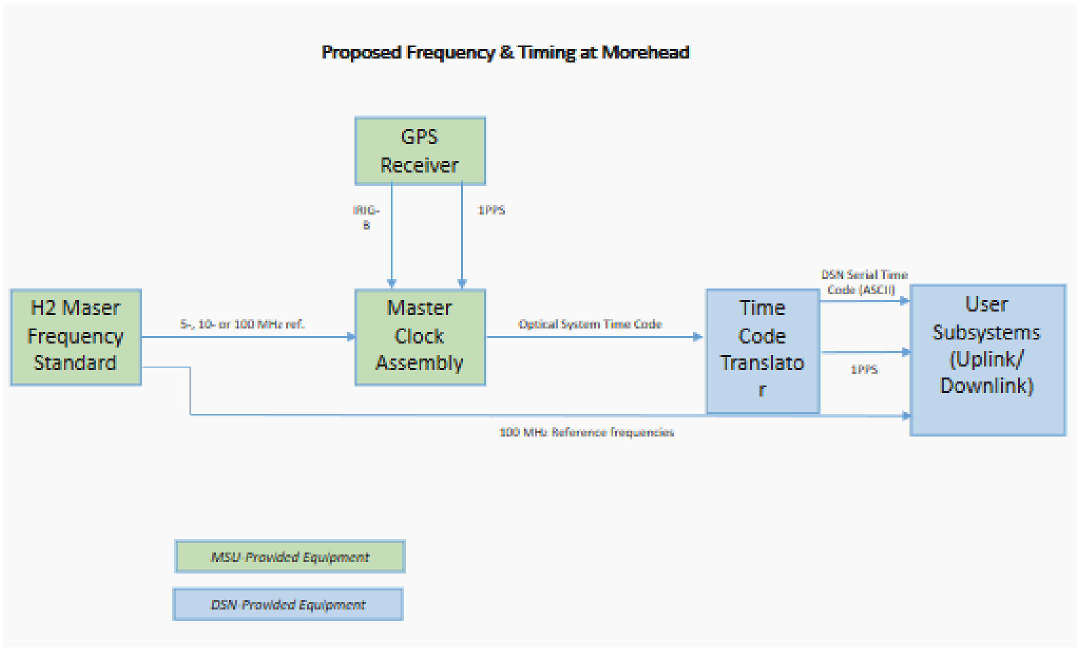


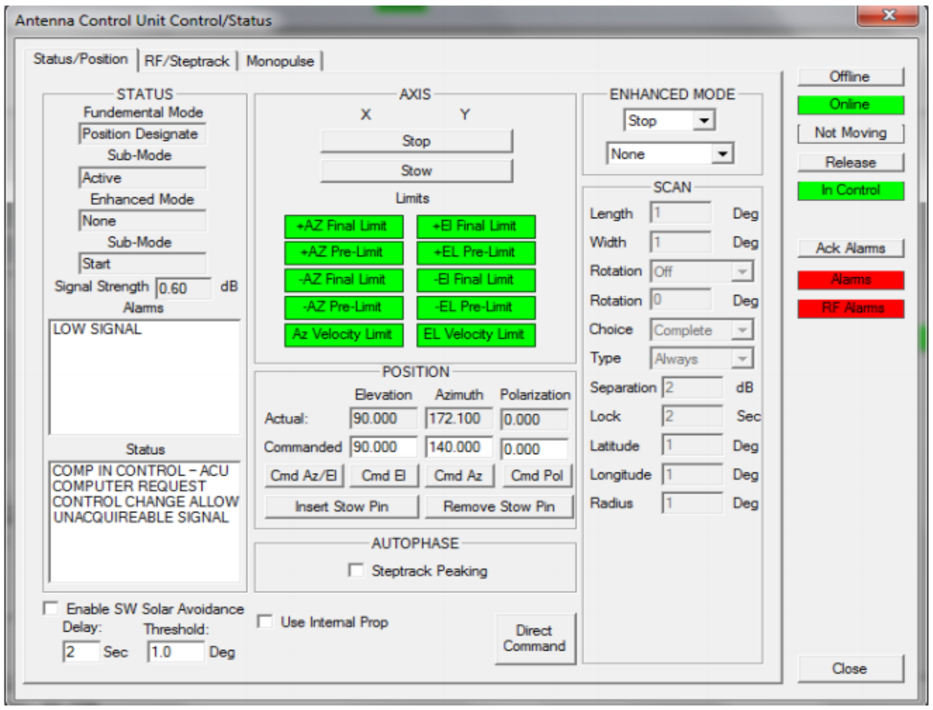
Figure 11 Frequency and Timing distribution

source: Kruth, Jeff. "Proposed Frequency & Timing at Morehead." 2017.

Op Console and NMC Computer

The Op console is the computer running the software that remotely controls the ACU. The software that controls the ACU from the Op Console is called HWCTRL. It's a Microsoft Windows based application. The user interface consists of the main window which optionally displays instrument status information, pending scheduled events, and recent system messages.[40] The features of HWCTRL include electronic entry of satellite ephemeris from a variety of sources such as Two-Line Elements and Az-El Tables, scheduled satellite tracks based on user selected ephemeris sets, the ability to specify the configuration of connected hardware for satellite tracks.[41]

[40] DeWitt, Henry "HWCNTRL User's Reference." (November 2009).
[41] DeWitt, Henry. "HWCNTRL User's Guide." (November 2009).


Antenna Control Unit Control/Status

STATUS
Fundemental Mode: Position Designate
Sub-Mode: Active
Enhanced Mode: None
Sub-Mode: Start
Signal Strength: 0.60 dB
Alarms: LOW SIGNAL
Status:
COMP IN CONTROL – ACU
COMPUTER REQUEST
CONTROL CHANGE ALLOW
UNACQUIREABLE SIGNAL

☐ Enable SW Solar Avoidance
Delay: 2 Sec Threshold: 1.0 Deg

AXIS X Y

POSITION

	Elevation	Azimuth	Polarization
Actual:	90.000	172.100	0.000
Commanded	90.000	140.000	0.000

AUTOPHASE
☐ Steptrack Peaking

☐ Use Internal Prop

ENHANCED MODE
Stop
None

SCAN

Length	1	Deg
Width	1	Deg
Rotation	Off	
Rotation	0	Deg
Choice	Complete	
Type	Always	
Separation	2	dB
Lock	2	Sec
Latitude	1	Deg
Longitude	1	Deg
Radius	1	Deg

Figure 12 Screenshot of HWCTRL

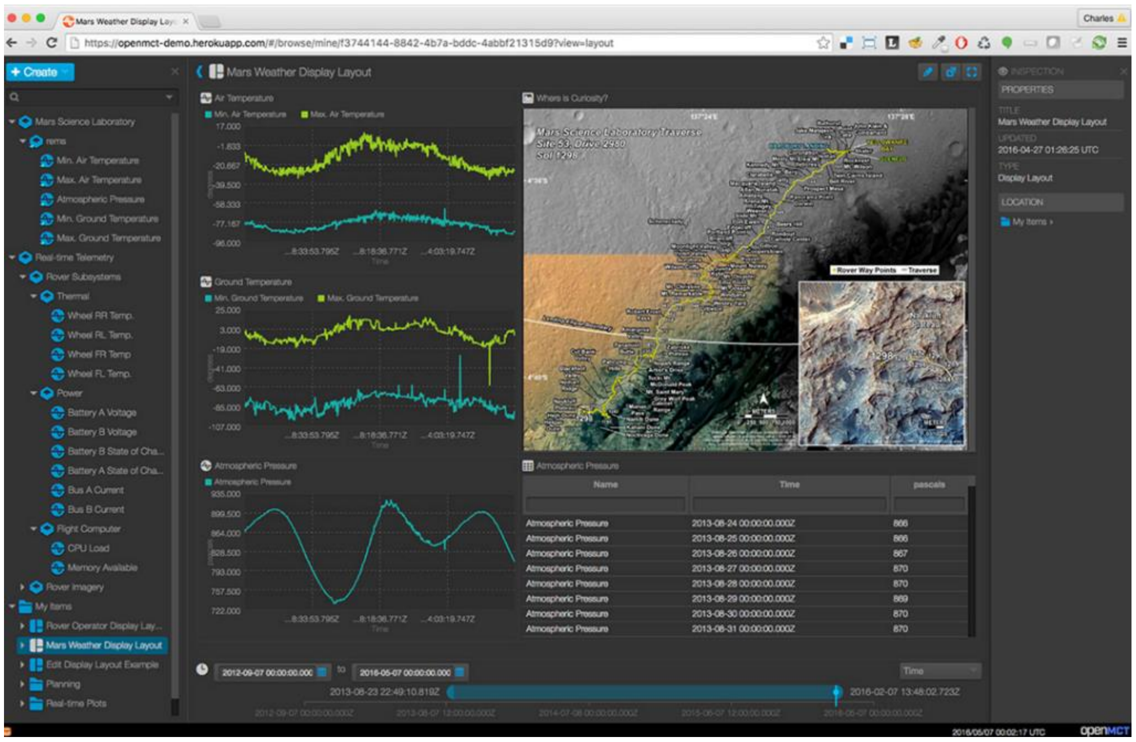
The NMC computer is the computer that runs the whole NMC system. All the Monitor data will be transferred there. It will hold the MySQL database, the GUI and all the code necessary to convert the data into useful formats. The software for the NMC system is described in more detail in Section 8. Most of the code for the NMC system is shown in Appendix B and Appendix C. This computer will be central to the whole operation of the NMC system and will provide an interface that can operate the 21M system.

7. Software Selection

Open Source Exploration

Open Source software solutions were explored at the beginning of this project. Building an entirely new monitor system was not an easy task. Adapting existing software would, in theory, considerably reduce the amount of work required to establish the system. Several software solutions were evaluated. One option considered was the COSMOS software by Ball

Aerospace. COSMOS offered real-time commanding, real-time and historical telemetry display and limits monitoring. It can also export telemetry as CSV files and can create documentation of available commands and telemetry packets[42]. The other option considered was Open MCT which was developed by NASA Ames in collaboration with JPL.



Atmospheric Pressure

Name	Time	pascals
Atmospheric Pressure	2013-08-24 00:00:00.000Z	866
Atmospheric Pressure	2013-08-25 00:00:00.000Z	866
Atmospheric Pressure	2013-08-26 00:00:00.000Z	867
Atmospheric Pressure	2013-08-27 00:00:00.000Z	870
Atmospheric Pressure	2013-08-28 00:00:00.000Z	870
Atmospheric Pressure	2013-08-29 00:00:00.000Z	869
Atmospheric Pressure	2013-08-30 00:00:00.000Z	870
Atmospheric Pressure	2013-08-31 00:00:00.000Z	870

Figure 13 Open MCT Display

Source: NASA Ames Researh Center. *About Open MCT.* https://nasa.github.io/openmct/about-open-mct/. (Accessed April 10, 2017.)

Open MCT is a flexible, web based, open-source software. It's used for building applications for planning, operation and analysis of systems that produce telemetry data and can display streaming and historical data, imagery, timelines and procedures, among other things. It can also

[42] Ball Aerospace. 2018. *COSMOS: Welcome*. https://cosmosrb.com/docs/home/ (accessed September 9, 2018)

export telemetry as a CSV and has customizable views. It operates by reading telemetry from a WebSocket and using definitions supplied by the operator to read and sort the data being received and displaying it through a webserver. The webserver can be connected to a database to store the data in CSV format[43]. Open MCT was chosen due to the customizable views, the nice GUI and the fact that it is web based, allowing users easier access. An example screenshot of Open MCT is shown in Figure 13. This path was followed for some time but in the end due to limited technical knowledge in JavaScript and Open MCTs lack of built in control structure, it was determined that developing software in house would be a better option.

In House System

After determining that an in-house developed software solution was a better option the next step was to design the system. Programing language and data storage options were reviewed against the established requirements, with a focus on the available teams' technical capabilities, the need to read various forms of data from various sources, the portability of the software and alignment with future JPL goals. In the end, an approach of using a MySQL database and a custom JAVA GUI was chosen. The primary factor driving this decision was the similarity to planned DSN NMC upgrades, described in Chapter V.1 below, and the recommendation of JPL engineers. The GUI will be able to be packaged up and distributed if necessary, it can also be sent to antennas being brought into the DSN. However, it will primarily be used by the 21M operators, with the archived data from the database being used by mission users. This approach will

[43] NASA Ames Researh Center. *About Open MCT*. https://nasa.github.io/openmct/about-open-mct/. (accessed April 10, 2017)

8. Software Specifications

Data Collection

The NMC system will be collecting data from three relatively broad classes of equipment. The first is commercially provided equipment, which includes the LNA, KPA, ACU, MASER and Dehydrator. Next is MSU fabricated equipment, which comprises 3 Raspberry Pi interfaces connected to Feed subsystems, the Upconverter and MASER monitoring. Finally, JPL provided equipment comprising of the DCD, DTT & USG which will have a JPL provided data aggregator software to compile the data. These subsystems provide data in a variety of formats, to make the data compatible, easier to manipulate and store, it will all be converted to CSV files if CSVs are not already provided. Some of the commercial subsystems, such as the LNA and ACU, already provide CSVs, others such as the KPA and Dehydrator will need a program to convert the data to CSV. The MSU provided equipment has been designed to output CSVs. The output of the JPL aggregator at this moment in time is unknown. An example of the CSV is shown in Appendix A.

The commercially provided equipment consists of the LNA, KPA, ACU, MASER and the Waveguide Dehydrator. The LNA outputs a CSV already so it doesn't need to be converted. The ACU outputs a file that needs to be modified to fit the CSV format. The KPA and Waveguide Dehydrator need to be modified to output files which need to be in CSV format. The MASER output is directed through the MSU fabricated Raspberry Pi described below. MSU fabricated equipment, comprises 3 Raspberry Pi interfaces connected to Feed subsystems, the Upconverter and MASER monitoring. These Raspberry Pi's have been programed to output CSVs. The JPL provided equipment outputs its data in its own unique format. In order to get the data in a standard CSV format JPL will provide software, which will run on a Raspberry Pi, to aggregate and translate the data to a CSV format.

Database

Data gathered from the subsystems will be housed in a MySQL database. MySQL was chosen because it is open source, commonly used, modular and intuitive. Some of the subsystems do not generate data in a format that is immediately ingestible into the database. MySQL is based on a client-server model. The core of MySQL is MySQL server, which handles all of the database instructions. It can handle large databases quickly and typically installed on only one machine, it is able to send the database to multiple locations, as users are able to access it via different MySQL client interfaces. In order to get the data from the subsystems into the MySQL database, two bash scripts are required to reformat the data that is not automatically ingestible and to automate the process of creating the database tables. To create the database two bash scripts have been developed comprising mostly of simple SQL commands which are shown in Appendix B. The first creates tables in the database, using the 'create table' command along with a list of data points and data types. Each subsystem will need to have a table created. removes first row of each csv file, stores that data in a temporary file called 'temp' that is used to ingest the data into each table. Using a MySQL database allows for easy accessibility, modification and distribution. The first row of the CSV needs to be removed before it is ingested into the MySQL database because SQL doesn't read headers properly

User Interface

The GUI will be used by 21M operators to monitor the status of the antenna systems during tracks. The GUI has been designed to be modular and adjustable to allow for long- and short-term flexibility. An early concept of the GUI design is shown in Figure 14.

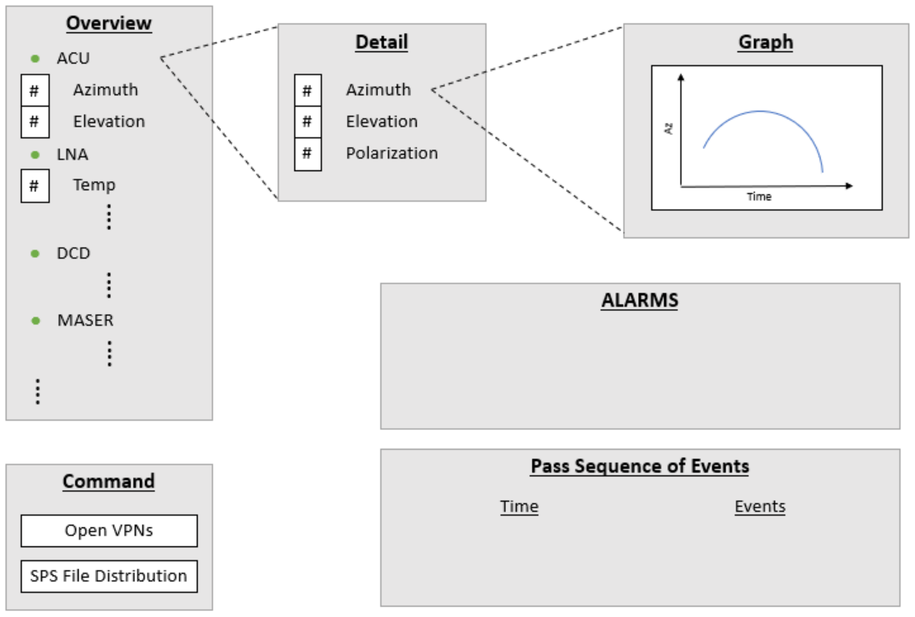
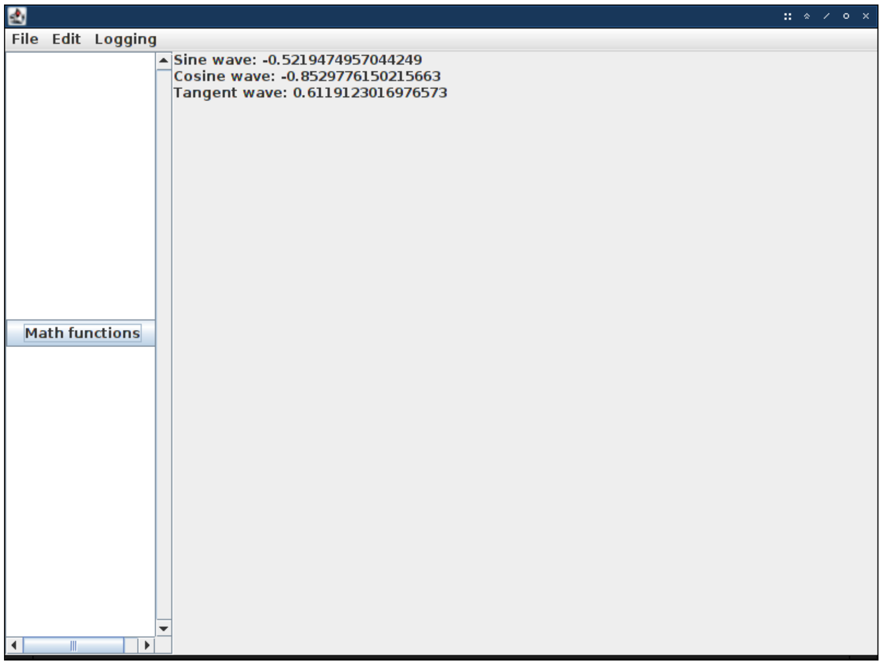


Figure 14 Proposed GUI design

The GUI of the NMC system will be written in JAVA with a Custom Library. Figure 15 shows the current state of the GUI. There is still some work left to do on the GUI. It needs to be integrated with the code that connect it to the database shown in Appendix C. Graphing, Commanding and Scheduling functionalities need to be added. The GUI will be able to be packaged up and distributed if necessary, it can also be sent to antennas being brought into the DSN. However, it will primarily be used by the 21M operators, with the archived data from the database being used by mission users.



```
Sine wave: -0.5219474957044249
Cosine wave: -0.8529776150215663
Tangent wave: 0.6119123016976573
```

Figure 15 Current GUI design

Source: Lowe, Jeremiah. "NCM Current Display." (April 2019)

Command

Adding control elements to the NMC system posed a bit of a challenge. Each subsystem has its own unique method of commanding. Due to the variety of communication protocols, adding in full command capabilities for each subsystem would be a monumental task. This is currently simplified by creating a script to access GUIs of each subsystem to allow direct commanding of the subsystems. Due to the variety of control methods of the different subsystems, to more fully integrate control elements, it is necessary to create unique software to access each subsystem and send commands. The other major component of commanding is to pull configuration information from the JPL Service Preparation System (SPS) and send the information to the DSN subsystems at MSU. A script has been developed to automate this which is shown in Appendix D.

Chapter IV Findings and Analysis

The project has evolved over time as specifications and requirements were refined. It was originally planned to use Open MCT, a mission control framework developed by NASA, to collect and display the data. The use of Open MCT was abandoned after it became clear that based on available expertise and major changes being made to Open MCT, creating software in house is a better option. It was decided to replace Open MCT’s functionality as a GUI and as a monitoring system by software that will be developed in house using shell scripts as well as code written in Python and in Java. After determining that an in-house developed software solution was a better option, options for programing language and data storage were reviewed against the established requirements, with a focus on the available teams’ technical capabilities, the need to read various forms of data from various sources, the portability of the software and alignment with future JPL goals. These future JPL goals include updating the NMC system to be based around Complex Event Processing, Apache Kafka and ElasticSearch. As the DSN improves and changes it will be important for the 21M to be able to grow and adapt with it. This is important because it will allow the 21M to continue to serve as a DSN asset, relieve some of the DSN overscheduling and serve as a low-risk, experimental asset for testing new DSN features. Having a flexible system will allow these changes to be made in the future. It was determined an approach of using a MySQL database and a custom JAVA GUI was chosen. The data being gathered is being standardized and formatted into a CSV file to be transferred and stored in a MySQL database. Most of the subsystems are being communicated with via Ethernet. This standardization is to make transferring and storing the data simpler. Adding control functionality to the NMC system posed a bit of a challenge. It would be necessary to create unique software to access each subsystem and send commands. Currently, a script to access GUIs of each subsystem to allow direct commanding of the subsystems is being created.

Chapter V Conclusions and Future Work

1. Future Work

Connecting Additional Subsystems

There are still several subsystems that need to be connected to the NMC System, these include:

- Feed Raspberry Pi-
    - Downconverter
    - Waveguide Switch
    - Test Translator
    - Noise Inject
- Upconverter Raspberry Pi
- JPL Aggregator
    - Downlink Tracking and Telemetry Subsystem (DTT)
    - Data Capture and Delivery Subsystem (DCD)
    - Uplink Tracking and Command (UPL)

The data provided from the KPA and Waveguide dehydrator needs to be confirmed.

When connecting a new subsystem to the NMC, there are a few scripts that need to be created or modified. The first thing to do is to add it to the create database script shown in Appendix B. Add a ‘create_table’ command with the applicable data and datatype is necessary. When all the subsystems are added to the create database script, run the ingest script. Then, create the specific MySQL to GUI software, an example of which is shown in Appendix C.

DSN NMC upgrades

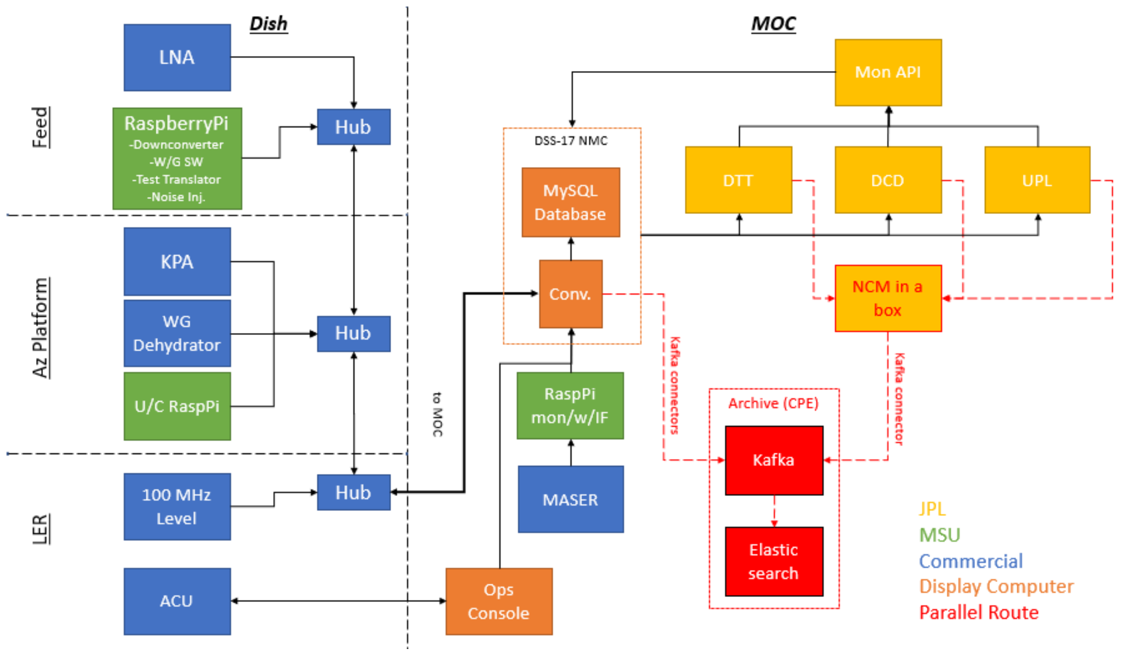


Figure 16 Expanded NMC block Diagram

Source: Kruth, Jeffrey A. "Corrected Block Diagram of NMC." (April 5, 2019).

The DSN is also working on upgrading its own NMC architecture in the near future. The upgrades will make use of Apache Kafka and ElasticSearch, both of which run on top of MySQL. Kafka is a fast, scalable streaming platform written in Java and Scala. It is open source and was originally developed by LinkedIn. Elastic Search is a flexible and highly scalable search engine that allows the user to perform and combine many types of searches. Integrating ElasticSearch and Kafka should be seamless. The NMC block diagram with Kafka and ElasticSearch added is shown in Figure 16. Apache Kafka and ElasticSearch can be easily incorporated into the planned 21M NMC system without too much modification because they are also based around MySQL.

Complex Event Processing, Apache Kafka and ElasticSearch

Complex Event Processing (CEP) is a method of taking as inputs a variety of real-time data from different sources, combining them, and analyzing them to detect relationships, trends and patterns. As DSN operations become more complex CEP will allow for rapid and complex analysis of the real time data sources generating data throughout the DSN. It can detect and predict meaningful events as well as analyze errors that may occur to match incidents to known discrepancies. The CEP infrastructure provides a mechanism to augment DSN's existing capabilities. Data from the NMC, configuration tables, and Sequences of Events are fed into a CEP engine. Then algorithmic rules are run on the data streams and events are output through streams of messages, logs, and TCP streams. Client applications can be connected to these output streams. Operators can then be alerted to conditions that can be watched over by the CEP system. [44]

Apache Kafka is a fast, scalable streaming platform written in Java and Scala, it functions as a distributed publish-subscribe messaging system. Some advantageous features of Kafka are using topics as the primary abstraction for publishing and subscribing to related data and treating each topic partition as a log. Kafka allows for easy recovery of input streams in the event of an interruption and allows the isolation of data consumers. **Error! Reference source not found.** shows an example Kafka data flow structure. It is open source and was originally developed by LinkedIn[45].

[44] Choi, Verma, and Malhotra, 14
[45] Apache Software Foundation . *Apache Kafka*. https://kafka.apache.org/. (2017)

ElasticSearch is a flexible and highly scalable search engine that allows the user to perform and combine many types of searches. It's an easily accessible, scalable, database search engine that can use SQL queries[46]. It is fast, scalable, flexible, and has a web interface, allowing for easy distribution of the data. One of the planned next steps in developing the monitor and control system will be to integrate CEP, Kafka and ElasticSearch into the MSU NMC architecture.

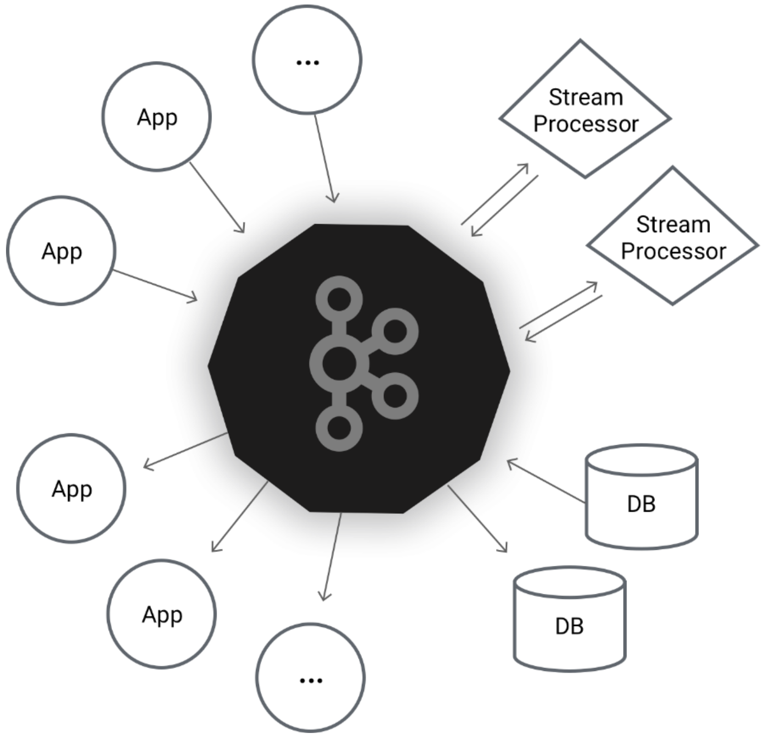


Figure 17 Apache Kafka Structure Diagram

Source: Apache Software Foundation . *Apache Kafka*. https://kafka.apache.org/. (2017)

ElasticSearch and Apache Kafka are designed work with SQL databases such as MySQL. This means that CEP, Kafka and ElasticSearch should smoothly integrate into the 21M NMC

[46] Elasticsearch. 2019. *Elasticsearch*. https://www.elastic.co/products/elasticsearch. (accessed March 28, 2019)

system. This will allow the 21M to be compatible with the DSN both now and into the future. As the DSN improves and changes it will be important for the 21M to be able to grow and adapt with it. Having a flexible system will allow these changes to be made in the future. This is important because it will allow the 21M to continue to serve as a DSN asset, relieve some of the DSN overscheduling and serve as a low-risk, experimental asset for testing new DSN features.

2. Conclusions

The 21M is being upgraded to become a node on the DSN with the identifier of DSS-17. These upgrades include a new cryogenic X-band feed, getting DSN receive and transmit systems, a Hydrogen MASER, increasing the automation of the system and developing a Monitor and Control system, which is the focus of this project. The development of the NMC system ended up being a challenging project. The primary tasks of the monitor data system are processing, displaying, and archiving data related to the state of the various subsystems. The project has evolved over time as specifications and requirements were refined. At first, open source software, Open MCT, was evaluated, but due to limited technical knowledge in JavaScript and Open MCT's lack of control structure, it was determined that developing software in house would be a better option. An approach of using a MySQL database and a custom JAVA GUI was chosen.

To make this decision several requirements were evaluated. The NMC system must be compatible with DSN and 21M infrastructure. It must be able to operate continuously for long periods of time when necessary. The NMC system must allow the use of the DSS-17 as an experimental station on the DSN, both now and into the future. The NMC system must be able to collect monitor data from and control all the systems in the 21M. The system must be able to be replicated by or distributed to any antennas that may be brought into the DSN similarly to the

21M in the future. Finally, Mission users must be able to access, modify and interpret the monitor data as necessary.

The different types of subsystems having different communications protocols made collecting the data difficult. To combat this, software is being developed to standardize the data being gathered and format it into a CSV file to be transferred and stored in a MySQL database. Most of the subsystems are being communicated with via Ethernet. Adding control functionality to the NMC system posed a bit of a challenge. It would be necessary to create unique software to access each subsystem and send commands. Currently, a script to access GUIs of each subsystem to allow direct commanding of the subsystems is being created. The decision to change course late in the project compounded the challenges. The use of Open MCT was abandoned after it became clear that based on available expertise and major changes being made to Open MCT, creating software in house is a better option. Despite these challenges in the end a plan that satisfies most of the outlined requirements and a baseline for most of the required software has been developed. Also, a path forward for the completion of the system has been planned, which is outlined in the Future Work section above.

During every DSN track monitor information is gathered by the NMC and analyzed to ensure that the dish is working properly, and diagnose any problems, such as a loss of signal, that might arise during a pass. The outlined approach will allow the 21M to be compatible with and meet the standards of the DSN both now and into the future. It will allow the 21M, in addition to assisting the DSN with everyday operations, to be a testbed for the DSN and be a model of how they can bring additional outside antennas into the network. At first, open source software, Open MCT, was evaluated, but due to limited technical knowledge in JavaScript and Open MCT's lack of control structure, it was determined that developing software in house would be a better option. Using a MySQL database and a custom JAVA GUI, a NMC system that is functional, low cost, modular and can easily be distributed is being designed and will allow the NMC to be compatible with DSN infrastructure.

Appendix A: Sample CSV

```
Date:Time       Com Alarm       LNA Tip Temp            Comp Temp       LNA Amb Temp            Cooler
Power   PCB PSU         LNA Ch1 V       Cooler PSU      LNA Ch1 I       Fan Speed       AlarmsStatus
        Comp Status     S/N Dewar       S/N PSU
03:10:00:01:00          0       84.6    40.8    36.3    59.9    23.9    14.9    30.0    327     6127
        0       13      1       0.67    1B021C9
03:10:00:02:00          0       84.5    40.8    36.4    59.9    23.9    15.0    30.0    327     6066
        0       13      1       0.67    1B021C9
03:10:00:03:00          0       84.5    40.8    36.3    59.9    23.9    15.0    30.0    327     6126
        0       13      1       0.67    1B021C9
03:10:00:04:00          0       84.6    40.8    36.2    59.8    23.9    15.0    30.0    327     6125
        0       13      1       0.67    1B021C9
03:10:00:05:00          0       84.5    40.7    36.2    59.5    23.9    15.0    30.0    327     6062
        0       13      1       0.67    1B021C9
03:10:00:06:00          0       84.5    40.8    36.2    59.5    23.9    15.0    30.0    328     6126
        0       13      1       0.67    1B021C9
03:10:00:07:00          0       84.6    40.8    36.2    59.5    23.9    15.0    30.0    327     6124
        0       13      1       0.67    1B021C9
03:10:00:08:00          0       84.5    40.7    36.2    59.5    23.9    15.0    30.0    327     6064
        0       13      1       0.67    1B021C9
03:10:00:09:00          0       84.6    40.7    36.2    59.5    23.9    15.0    30.0    327     6125
        0       13      1       0.67    1B021C9
```

```
03:10:00:10:00        0       84.6   40.7   36.1   59.5   23.9   14.9   30.0   327    6065
       0       13     1       0.67   1B021C9
03:10:00:11:00        0       84.6   40.7   36.1   59.5   23.9   15.0   30.0   327    6124
       0       13     1       0.67   1B021C9
03:10:00:12:00        0       84.6   40.7   36.1   59.5   23.9   15.0   30.0   327    6126
       0       13     1       0.67   1B021C9
03:10:00:13:00        0       84.6   40.7   36.1   59.5   23.9   15.0   30.0   327    6125
       0       13     1       0.67   1B021C9
03:10:00:14:00        0       84.6   40.6   36.1   59.5   23.9   15.0   30.0   327    6065
       0       13     1       0.67   1B021C9
03:10:00:15:00        0       84.6   40.6   36.1   59.6   23.9   15.0   30.0   327    6066
       0       13     1       0.67   1B021C9
03:10:00:16:00        0       84.6   40.6   36.1   59.5   23.9   14.9   30.0   327    6124
       0       13     1       0.67   1B021C9
03:10:00:17:00        0       84.5   40.5   36.2   59.2   23.9   14.9   30.0   327    6127
       0       13     1       0.67   1B021C9
03:10:00:18:00        0       84.5   40.6   35.9   59.5   23.9   15.0   30.0   327    6125
       0       13     1       0.67   1B021C9
03:10:00:19:00        0       84.5   40.5   36.0   59.5   23.9   15.0   30.0   327    6123
       0       13     1       0.67   1B021C9
03:10:00:20:00        0       84.5   40.5   36.0   59.5   23.9   15.0   30.0   327    6123
       0       13     1       0.67   1B021C9
```

```
03:10:00:21:00      0      84.5   40.5   36.0   59.2   23.9   15.0   30.0   328    6126

         0      13     1      0.67   1B021C9

03:10:00:22:00      0      84.6   40.4   36.2   59.2   23.9   15.0   30.0   327    6066

         0      13     1      0.67   1B021C9

03:10:00:23:00      0      84.6   40.4   36.0   59.2   23.9   14.9   30.0   327    6123

         0      13     1      0.67   1B021C9

03:10:00:24:00      0      84.6   40.5   35.9   59.1   23.9   15.0   30.0   327    6122

         0      13     1      0.67   1B021C9

03:10:00:25:00      0      84.6   40.5   35.9   59.2   23.9   14.9   30.0   327    6063

         0      13     1      0.67   1B021C9

03:10:00:26:00      0      84.6   40.4   35.9   59.2   23.9   15.0   30.0   327    6062

         0      13     1      0.67   1B021C9

03:10:00:27:00      0      84.6   40.3   35.9   59.2   23.9   15.0   30.0   327    6123

         0      13     1      0.67   1B021C9

03:10:00:28:00      0      84.6   40.3   35.9   59.1   23.9   15.0   30.0   327    6124

         0      13     1      0.67   1B021C9

03:10:00:29:00      0      84.6   40.3   35.8   59.2   23.9   15.0   30.0   327    6063

         0      13     1      0.67   1B021C9

03:10:00:30:00      0      84.6   40.3   35.9   59.2   23.9   14.9   30.0   327    6123

         0      13     1      0.67   1B021C9

03:10:00:31:00      0      84.6   40.2   35.8   59.1   23.9   15.0   30.0   327    6124

         0      13     1      0.67   1B021C9
```

```
03:10:00:32:00      0     84.6   40.2   35.9   59.2   23.9   15.0   30.0   327    6062

          0     13    1     0.67   1B021C9

03:10:00:33:00      0     84.6   40.2   35.8   59.1   23.9   15.0   30.0   327    6064

          0     13    1     0.67   1B021C9

03:10:00:34:00      0     84.6   40.2   35.8   59.2   23.9   15.0   30.0   327    6121

          0     13    1     0.67   1B021C9

03:10:00:35:00      0     84.6   40.1   35.8   59.2   23.9   14.9   30.0   327    6063

          0     13    1     0.67   1B021C9

03:10:00:36:00      0     84.6   40.1   35.8   59.2   23.9   15.0   30.0   328    6062

          0     13    1     0.67   1B021C9

03:10:00:37:00      0     84.6   40.0   35.7   59.1   23.9   15.0   30.0   327    6061

          0     13    1     0.67   1B021C9

03:10:00:38:00      0     84.6   40.1   35.7   59.2   23.9   15.0   30.0   327    6122

          0     13    1     0.67   1B021C9

03:10:00:39:00      0     84.6   40.0   35.7   59.1   23.9   15.0   30.0   327    6061

          0     13    1     0.67   1B021C9

03:10:00:40:00      0     84.6   40.0   35.7   59.2   23.9   15.0   30.0   327    6123

          0     13    1     0.67   1B021C9

03:10:00:41:00      0     84.6   39.9   35.7   59.1   23.9   15.0   30.0   327    6125

          0     13    1     0.67   1B021C9

03:10:00:42:00      0     84.6   39.9   35.8   59.2   23.9   15.0   30.0   327    6062

          0     13    1     0.67   1B021C9
```

```
03:10:00:43:00          0        84.6    39.9    35.7    59.1    23.9    15.0    30.0    327     6120

        0       13      1        0.67    1B021C9

03:10:00:44:00          0        84.6    39.9    35.6    59.2    23.9    14.9    30.0    327     6123

        0       13      1        0.67    1B021C9

03:10:00:45:00          0        84.6    39.9    35.6    59.1    23.9    14.9    30.0    327     6059

        0       13      1        0.67    1B021C9

03:10:00:46:00          0        84.6    39.8    35.5    59.2    23.9    15.0    30.0    327     6058

        0       13      1        0.67    1B021C9

03:10:00:47:00          0        84.5    39.9    35.6    59.2    23.9    15.0    30.0    327     6060

        0       13      1        0.67    1B021C9

03:10:00:48:00          0        84.6    39.8    35.5    59.1    23.9    15.0    30.0    327     6059

        0       13      1        0.67    1B021C9

03:10:00:49:00          0        84.5    39.8    35.5    58.5    23.9    14.9    30.0    327     6059

        0       13      1        0.67    1B021C9

03:10:00:50:00          0        84.6    39.8    35.5    58.5    23.9    15.0    30.0    327     6061

        0       13      1        0.67    1B021C9

03:10:00:51:00          0        84.6    39.8    35.4    58.5    23.9    14.9    30.0    327     6120

        0       13      1        0.67    1B021C9

03:10:00:52:00          0        84.6    39.9    35.5    58.5    23.9    15.0    30.0    327     6123

        0       13      1        0.67    1B021C9

03:10:00:53:00          0        84.6    39.8    35.4    58.5    23.9    15.0    30.0    327     6122

        0       13      1        0.67    1B021C9
```

```
03:10:00:54:00      0       84.6   39.7   35.4   58.4   23.9   15.0   30.0   327    6060

         0       13       1       0.67   1B021C9

03:10:00:55:00      0       84.6   39.8   35.4   58.5   23.9   15.0   30.0   327    6121

         0       13       1       0.67   1B021C9

03:10:00:56:00      0       84.6   39.7   35.3   58.4   23.9   14.9   30.0   327    6060

         0       13       1       0.67   1B021C9

03:10:00:57:00      0       84.6   39.7   35.4   58.5   23.9   14.9   30.0   327    6061

         0       13       1       0.67   1B021C9

03:10:00:58:00      0       84.7   39.7   35.4   58.5   23.9   14.9   30.0   327    6120

         0       13       1       0.67   1B021C9

03:10:00:59:00      0       84.7   39.8   35.3   58.5   23.9   15.0   30.0   327    6120

         0       13       1       0.67   1B021C9

03:10:01:00:00      0       84.7   39.6   35.3   58.5   23.9   14.9   30.0   327    6121

         0       13       1       0.67   1B021C9

03:10:01:01:00      0       84.7   39.7   35.3   58.5   23.9   14.9   30.0   327    6060

         0       13       1       0.67   1B021C9

03:10:01:02:00      0       84.7   39.8   35.2   58.5   23.9   14.9   30.0   327    6120

         0       13       1       0.67   1B021C9

03:10:01:03:00      0       84.7   39.7   35.3   58.5   23.9   15.0   30.0   327    6121

         0       13       1       0.67   1B021C9

03:10:01:04:00      0       84.7   39.7   35.3   58.7   23.9   15.0   30.0   327    6120

         0       13       1       0.67   1B021C9
```

```
03:10:01:05:00      0     84.7  39.7  35.2  58.4  23.9  14.9  30.0  327   6063

          0     13    1     0.67  1B021C9

03:10:01:06:00      0     84.7  39.7  35.2  58.5  23.9  15.0  30.0  327   6121

          0     13    1     0.67  1B021C9

03:10:01:07:00      0     84.6  39.7  35.3  58.5  23.9  15.0  30.0  327   6123

          0     13    1     0.67  1B021C9

03:10:01:08:00      0     84.6  39.7  35.2  58.5  23.9  14.9  30.0  327   6124

          0     13    1     0.67  1B021C9

03:10:01:09:00      0     84.6  39.7  35.2  58.5  23.9  15.0  30.0  327   6060

          0     13    1     0.67  1B021C9

03:10:01:10:00      0     84.7  39.7  35.2  58.5  23.9  14.9  30.0  327   6121

          0     13    1     0.67  1B021C9

03:10:01:11:00      0     84.7  39.7  35.2  58.5  23.9  15.0  30.0  327   6121

          0     13    1     0.67  1B021C9

03:10:01:12:00      0     84.6  39.6  35.1  58.5  23.9  15.0  30.0  327   6121

          0     13    1     0.67  1B021C9

03:10:01:13:00      0     84.6  39.7  35.1  58.4  23.9  14.9  30.0  327   6061

          0     13    1     0.67  1B021C9

03:10:01:14:00      0     84.6  39.6  35.2  58.5  23.9  15.0  30.0  327   6120

          0     13    1     0.67  1B021C9

03:10:01:15:00      0     84.6  39.6  35.2  58.5  23.9  15.0  30.0  327   6123

          0     13    1     0.67  1B021C9
```

```
03:10:01:16:00      0      84.6   39.6   35.2   58.5   23.9   15.0   30.0   327    6121

          0      13     1      0.67   1B021C9

03:10:01:17:00      0      84.6   39.6   35.1   58.5   23.9   15.0   30.0   327    6057

          0      13     1      0.67   1B021C9

03:10:01:18:00      0      84.6   39.6   35.1   58.5   23.9   15.0   30.0   327    6121

          0      13     1      0.67   1B021C9

03:10:01:19:00      0      84.6   39.6   35.1   58.5   23.9   15.0   30.0   327    6121

          0      13     1      0.67   1B021C9

03:10:01:20:00      0      84.6   39.6   35.1   58.5   23.9   14.9   30.0   327    6121

          0      13     1      0.67   1B021C9

03:10:01:21:00      0      84.6   39.6   35.1   58.4   23.9   15.0   30.0   327    6121

          0      13     1      0.67   1B021C9

03:10:01:22:00      0      84.6   39.6   35.1   58.5   23.9   15.0   30.0   327    6061

          0      13     1      0.67   1B021C9

03:10:01:23:00      0      84.6   39.6   35.1   58.4   23.9   14.9   30.0   327    6119

          0      13     1      0.67   1B021C9

03:10:01:24:00      0      84.6   39.6   35.1   58.5   23.9   15.0   30.0   326    6121

          0      13     1      0.67   1B021C9

03:10:01:25:00      0      84.6   39.6   35.1   58.4   23.9   15.0   30.0   327    6120

          0      13     1      0.67   1B021C9

03:10:01:26:00      0      84.6   39.6   35.0   58.5   23.9   14.9   30.0   327    6061

          0      13     1      0.67   1B021C9
```

```
03:10:01:27:00     0    84.6  39.7  35.1  58.4  23.9  15.0  30.0  327   6061

         0    13    1    0.67  1B021C9

03:10:01:28:00     0    84.6  39.6  35.0  58.5  23.9  15.0  30.0  327   6060

         0    13    1    0.67  1B021C9

03:10:01:29:00     0    84.6  39.5  35.0  58.4  23.9  15.0  30.0  327   6120

         0    13    1    0.67  1B021C9

03:10:01:30:00     0    84.6  39.6  35.0  58.5  23.9  14.9  30.0  327   6121

         0    13    1    0.67  1B021C9

03:10:01:31:00     0    84.6  39.5  35.0  58.5  23.9  15.0  30.0  326   6119

         0    13    1    0.67  1B021C9

03:10:01:32:00     0    84.6  39.5  35.0  58.5  23.9  15.0  30.0  327   6117

         0    13    1    0.67  1B021C9

03:10:01:33:00     0    84.6  39.5  35.0  58.5  23.9  15.0  30.0  326   6121

         0    13    1    0.67  1B021C9

03:10:01:34:00     0    84.6  39.5  35.0  58.5  23.9  15.0  30.0  327   6120

         0    13    1    0.67  1B021C9

03:10:01:35:00     0    84.6  39.5  35.0  58.5  23.9  15.0  30.0  327   6118

         0    13    1    0.67  1B021C9

03:10:01:36:00     0    84.6  39.5  35.0  58.5  23.9  15.0  30.0  327   6117

         0    13    1    0.67  1B021C9

03:10:01:37:00     0    84.6  39.4  35.0  58.5  23.9  15.0  30.0  327   6117

         0    13    1    0.67  1B021C9
```

```
03:10:01:38:00      0     84.6   39.4   35.0   58.5   23.9   15.0   30.0   327    6059

        0     13    1     0.67   1B021C9

03:10:01:39:00      0     84.6   39.5   34.9   58.5   23.9   15.0   30.0   327    6059

        0     13    1     0.67   1B021C9

03:10:01:40:00      0     84.6   39.5   34.9   58.5   23.9   14.9   30.0   328    6119

        0     13    1     0.67   1B021C9

03:10:01:41:00      0     84.5   39.5   35.0   58.5   23.9   15.0   30.0   326    6120

        0     13    1     0.67   1B021C9

03:10:01:42:00      0     84.5   39.5   34.9   58.5   23.9   14.9   30.0   326    6120

        0     13    1     0.67   1B021C9

03:10:01:43:00      0     84.5   39.5   34.9   58.5   23.9   15.0   30.0   327    6118

        0     13    1     0.67   1B021C9
```

Appendix B: Database Scripts

Script to create the database:

```
create database db

use db

create table LNA (Date_Time varchar(20), Com_Alarm int, LNA_Tip_Temp double(4,1),
Comp_Temp double(3,1), LNA_Amb_Temp double(3,1), Cooler_Power int, PCB_PSU
double(3,1), LNA_Ch1_V double(3,1), Cooler_PSU double(3,1), LNA_Ch1_I int, Fan_Speed
int, Alarms int, Status int, Comp_Status int, SN_Dewar double(3,2), SN_PSU varchar(7));


create table KPA (RF_output varchar(5), Beam_Voltage varchar(7), Reflected_RF varchar(5),
Body_current varchar(5), Beam_Current varchar(6), Cabinent_Temperature varchar(5),
Beacon_1 varchar(10), Attenuation varchar(6), Heater_Voltage varchar(6), Inlet_Temperature
varchar(5), Beacon varchar(10), Heater_Current varchar(6), Outlet_Temperature varchar(5),
Online_Status tinyint(1));


create table ACU (Azimuth varchar(8), Elevation varchar(7), Polarization varchar(7), ACUmode
varchar(30), ACUsignal varchar(6));


create table Maser (MaserDate varchar(20), Time varchar(8), Temp1 varchar(8), Temp2
varchar(8), Temp3 varchar(8));


create table FeedControl(DC_PLOstatus int,Trans_PLO int, PowerMeter int, PowerSupply int,
PhysicalTemp int,BBswitch int, TRNAS_noise_SCL int,Noise varchar(3));
```

Ingest Script:

```
cat $1 | sed -e '1d' > $2.txt

mysqlimport --local ssc $2.txt
```

Appendix C: MySQL to GUI software

```
import java.sql.*;   // Use 'Connection', 'Statement' and 'ResultSet' classes in java.sql package

import java.util.List;

import java.util.Scanner;

import java.util.Arrays;

import java.util.stream.Collectors;


public class LNA_data {

    public static void main(String[] args) throws ClassNotFoundException {

        // load and register JDBC driver for MySQL

        Class.forName("com.mysql.cj.jdbc.Driver");

        try {


            // Create Connection

            Connection conn = DriverManager.getConnection(


"jdbc:mysql://localhost:3306/NMC?useUnicode=true&useJDBCCompliantTimezoneShift=true

&useLegacyDatetimeCode=false&serverTimezone=UTC", "root", "shopps22");


            // Create an SQL statement

            Statement stmt = conn.createStatement();


            String time[] = time_range();
            String start = time[0];
            String end = time[1];

            int choices[] = selections();

            // Execute statement, this needs to have an interface with the UI, so we can create the
proper statements
            String strSelect = "select Date_Time" + sql_string(choices) + " from LNA where
Date_Time between '" + start + "' and '" + end + "';";
            System.out.println("The SQL query is: " + strSelect); // Echo For debugging
            System.out.println();

            ResultSet rset = stmt.executeQuery(strSelect);

            dataSet(rset, choices);

        } catch (SQLException ex) {
            ex.printStackTrace();
        }
    }
    // Find the specific collection of data for specific times.  We may want to put this in epoch
time, and do it for all SQL Tables
    public static String[] time_range() {
```

```
    Scanner reader = new Scanner(System.in);

    System.out.println("Enter start time (DD:MO:HH:MM:SS):");

    String start_time = reader.next();

    System.out.println("Enter an end time (DD:MO:HH:MM:SS):");

    String end_time = reader.next();


    //String start_time = "15:11:21:38:00";

    //String end_time = "15:11:21:58:00";

    String ar[] = new String[2];

    ar[0] = start_time;

    ar[1] = end_time;


    return ar;


}

// Get specific data points that want to be graphed.

public static int[] selections() {

    int counter = 0;

    int sels[] = new int[15];


    do {

        System.out.println("Select the data points you desire to be displayed:");
        System.out.println(" [1] Com Alarm \n [2] LNA Tip Temperature \n [3] Computer
Temperature \n [4] LNA Ambient Temperature \n [5] Cooler Power \n [6] PCB PSU \n [7] LNA
Chanel 1 Voltage(?) \n [8] Cooler PSU \n [9] LNA Chanel 1 Current (?) \n [10] Fan Speed \n
[11] Alarms \n [12] Status \n [13] Computer Status \n [14] S/N Dewar \n [15] S/N PSU \n [0]
Done.");
        Scanner reader = new Scanner(System.in);
        int sel = reader.nextInt();

        if (sel == 0) {
            break;
        }

        //Insert stream magic here
        List<Integer> list = Arrays.stream(sels).boxed().collect(Collectors.toList());

        boolean result = list.contains(sel);

        if (result){
            System.out.println("Already in your selections.  So far, you have selected: ");
            System.out.println(Arrays.toString(sels));
        }
        else if (!result){
            sels[counter] = sel;
            counter++;
        }
    } while (counter < 16);

    return sels;
}
// Create The SQL string for data pull
public static String sql_string(int[] choices) {
    String output = "";
    for (int current:choices) {
        if (current == 1) {
            output = output + ", Com_Alarm";
        } else if (current == 2) {
            output = output + ", LNA_Tip_Temp";
        } else if (current == 3) {
            output = output + ", Comp_Temp";
        } else if (current == 4) {
            output = output + ", LNA_Amb_Temp";
        } else if (current == 5) {
            output = output + ", Cooler_Power";
        } else if (current == 6) {
            output = output + ", PCB_PSU";
        } else if (current == 7) {
            output = output + ", LNA_Ch1_V";
        } else if (current == 8) {
            output = output + ", Cooler_PSU";
        } else if (current == 9) {
            output = output + ", LNA_Ch1_I";
        } else if (current == 10) {
            output = output + ", Fan_Speed";
        } else if (current == 11) {
            output = output + ", Alarms";
        } else if (current == 12) {
            output = output + ", Status";
        } else if (current == 13) {
            output = output + ", Comp_Status";
        } else if (current == 14) {
            output = output + ", SN_Dewar";
        } else if (current == 15) {
            output = output + ", SN_PSU";
        } else if (current == 0) {
            break;
        }

    }
    return output;
```

```
}
//Sort through Result Set
public static void dataSet(ResultSet rset, int[] choices) throws SQLException {
    String result = "";
    while (rset.next()) {
        String Date_Time = rset.getString("Date_Time");
        result = result + Date_Time;
        for (int current : choices) {
            if (current == 1) {
                int Com_Alarm = rset.getInt("Com_Alarm");
                result = result + ", " + Com_Alarm;
            } else if (current == 2) {
                double LNA_Tip_Temp = rset.getDouble("LNA_Tip_Temp");
                result = result + ", " + LNA_Tip_Temp;
            } else if (current == 3) {
                double Comp_Temp = rset.getDouble("Comp_Temp");
                result = result + ", " + Comp_Temp;
            } else if (current == 4) {
                double LNA_Amb_Temp = rset.getDouble("LNA_Amb_Temp");
                result = result + ", " + LNA_Amb_Temp;
            } else if (current == 5) {
                double Cooler_Power = rset.getDouble("Cooler_Power");
                result = result + ", " + Cooler_Power;
```

```
} else if (current == 6) {
    double PCB_PSU = rset.getDouble("PCB_PSU");
    result = result + ", " + PCB_PSU;
} else if (current == 7) {
    double LNA_Ch1_V = rset.getDouble("LNA_CH1_V");
    result = result + ", " + LNA_Ch1_V;
} else if (current == 8) {
    double Cooler_PSU = rset.getDouble("Cooler_PSU");
    result = result + ", " + Cooler_PSU;
} else if (current == 9) {
    int LNA_Ch1_I = rset.getInt("LNA_Ch1_I");
    result = result + ", " + LNA_Ch1_I;
} else if (current == 10) {
    int Fan_Speed = rset.getInt("Fan_Speed");
    result = result + ", " + Fan_Speed;
} else if (current == 11) {
    int Alarms = rset.getInt("Alarms");
    result = result + ", " + Alarms;
} else if (current == 12) {
    int Status = rset.getInt("Status");
    result = result + ", " + Status;
} else if (current == 13) {
    int Comp_Status = rset.getInt("Comp_Status");
```

```
                result = result + ", " + Comp_Status;
            } else if (current == 14) {
                double SN_Dewar = rset.getDouble("SN_Dewar");
                result = result + ", " + SN_Dewar;
            } else if (current == 15) {
                String SN_PSU = rset.getString("SN_PSU");
                result = result + ", " + SN_PSU;
            } else if (current == 0) {
                break;
            }
        }
        System.out.println(result);
        System.out.println();
        result = "";
    }
  }
}
```

Appendix D: SPS Portal Predict Distribution

```
from bs4 import BeautifulSoup, SoupStrainer

from requests.compat import urljoin

from os import path

import os

import csv

import re

import requests

import urllib.request

import pandas as pd

import pexpect


########## PULL FILES FROM SPS PORTAL ##########


## Ask User Input for URL

Mission = input('Mission Name: ')

Antenna = input('Antenna: ')

Start = input('Start Time (formated as yyyy-doyThh:mm:ss ex. 2017-178T00:00:00): ')

End = input('End Time (formated same as Start Time): ')

##Mission = 'MRO'

##antenna = '26'

##Start = '2017-178T00:00:00'

##End = '2017-180T00:00:00'
```

```
MissionInfo = {'mission': Mission, 'antenna': Antenna, 'startTime': Start, 'endTime': End}


## Prepare URL

r = requests.get()

data = r.text


## Create beautifulSoup object and grab table

soup = BeautifulSoup(data, "html.parser")

table = soup.find_all('table')[0]

rows = table.find_all('tr')[1:]

data = {

    'VERSION' : [],

    'WEEK' : [],

    'YEAR' : [],

    'STARTTIME' : [],

    'BOT' : [],

    'EOT' : [],

    'ENDTIME' : [],

    'FACILITY' : [],

    'PROJUSER' : [],

    'ACTIVITY' : [],

    'CONFIGCODE' : [],

    'EQUIPMENTLIST' : [],
    'WRKCAT' : [],
    'SCHEDULEITEMID' : [],
    'SOECODE' : [],
    'ACTIVITYID' : [],
    'ACTIVITYTYPE' : [],
    'NIB' : [],
    }
for row in rows:
    cols = row.find_all('td')
    data['VERSION'].append( cols[0].get_text() )
    data['WEEK'].append( cols[1].get_text() )
    data['YEAR'].append( cols[2].get_text() )
    data['STARTTIME'].append( cols[3].get_text() )
    data['BOT'].append( cols[4].get_text() )
    data['EOT'].append( cols[5].get_text() )
    data['ENDTIME'].append( cols[6].get_text() )
    data['FACILITY'].append( cols[7].get_text() )
    data['PROJUSER'].append( cols[8].get_text() )
    data['ACTIVITY'].append( cols[9].get_text() )
    data['CONFIGCODE'].append( cols[10].get_text() )
    data['EQUIPMENTLIST'].append( cols[11].get_text() )
    data['WRKCAT'].append( cols[12].get_text() )
    data['SCHEDULEITEMID'].append( cols[13].get_text() )
    data['SOECODE'].append( cols[14].get_text() )

    data['ACTIVITYID'].append( cols[15].get_text() )

    data['ACTIVITYTYPE'].append( cols[16].get_text() )

    data['NIB'].append( cols[17].get_text() )


activityData = pd.DataFrame( data )

activityData.to_csv("activity.csv")

df = pd.DataFrame( data['ACTIVITYID'] )

df.to_csv("activityid.csv", index=False, header=False, line_terminator=' ')


## Append Activity ID to URL then scrape site for files

with open('activityid.csv') as f:

    for activity in map(str.strip,f):

        url = "".format(activity)

        print(url)

        s = requests.get(url)

        soup = BeautifulSoup(s.content, "html.parser")

        for a in soup.find_all('a', href=True):

            link = a.get('href')

            text = a.text

            subdir = path.join('/Documents', activity)

            if not os.path.exists(subdir):

                os.mkdir(subdir)
            filepath = path.join('/Documents', activity, text)
            if os.path.exists(filepath):
                print(filepath, 'already exists')
            w = requests.get(link)
            open(filepath, 'wb').write(w.content)

## Rename .dlf file if necessary
ans = input('Does the .dlf file need to be renamed? (Y/N)')
if ans = 'Y':
    dlf_old = input('Enter original name of .dlf file: ')
    src = path.join('/Documents', activity, dlf_old)
    dlf_new = input('Enter new name of .dlf flile: ')
    dst = path.join('/Documents', activity, dlf_new)
    if os.path.isfile(src):
        os.rename(src, dst)

############ TRANSFER FILES TO DTT ################


pwd = input('Enter Password: ')
activity = input('Enter activity id: ')
print('Enter paths of files to be transfered (if no file press ENTER)')
cfg = input('cfg file path: ')
scap = input('scap file path: ')
```

```
dlf = input('dlf file path: ')


if cfg:
    try:
        cfg_command = ‘'.format(cfg, activity)


        child = pexpect.spawn(cfg_command)
        child.expect('Password:')
        child.sendline(pwd)
    except:
        print('something went wrong transfering the cfg file?')


if scap:
    try:
        scap_command = ''.format(scap, activity)


        child = pexpect.spawn(scap_command)
        child.expect('Password:')
        child.sendline(pwd)
    except:
        print('something went wrong transfering the scap file?')


if dlf:
    try:
        dlf_command = ‘'.format(dlf, activity)


        child = pexpect.spawn(dlf_command)
        child.expect('Password:')
        child.sendline(pwd)
    except:
        print('something went wrong transfering the dlf file?')


print('Done!')
```

Bibliography

Apache Software Foundation. "Apache Kafka." 2017. https://kafka.apache.org/.

Ball Aerospace. *COSMOS: Welcome.* 2019. https://cosmosrb.com/docs/home/ (accessed September 9, 2018).

Boker, Eva. "Automating Operations for NASA's Deep Space Network (DSN)." IEEE, 2001.

Cheung, Kar-Ming, Charles Lee, and Stefan Waldherr. "Architecture and concept of operation of next-generation ground network for communications and tracking of interplanetary smallsats." *14th International Conference on Space Operations.* 2016. 2401.

Choi, Joshua S, Rishi Verma, and Shan Malhotra. "Achieving Fast Operational Intelligence in NASA's Deep Space Network Through Complex Event Processing." *SpaceOps 2016 Conference.* 2016.

Communications & Power Industries Canada Inc. "X-Band Klystron 3.4 kW." June 8, 2008. https://www.cpii.com/docs/datasheets/144/X-Band-3.3kw.pdf.

"Compact Cryogenic LNA." *Callisto.* 2019. http://www.callisto-space.com/en/page/compact-cryogenic-lna-2.php.

DeWitt, Henry. "HWCNTRL User's Guide." November 2009.

DeWitt, Henry. "HWCNTRL User's Reference." November 2009.

Elasticsearch. "Elasticsearch." 2019. https://www.elastic.co/products/elasticsearch (accessed March 28, 2019).

ETI. "Netcom Model ADH Automatic Dehydrator with Ethernet Quick-Reference Installation Manual." South Bend, Indiana USA , 2019.

Garrett, Shiela. "Data Capture and Delivery Subsystem (DCD) Operations and Maintenance Training." Spring 2017.

Imbriale, William A. *The 34-Meter Research and Development Beam-Waveguide Antenna.* Vol. 4. Large Antennas of the Deep Space Network, in *Deep Space Communications and Navigation Series*, 167-225. 2003.

Johnston, Mark D, Michael Levesque, Shan Malhotra, Daniel Tran, Rishi Verma, and Silvino Zendejas. "NASA deep space network: automation improvements in the follow-the-Sun Era." *24th International Joint Conference on Artificial Intelligence.* Buenos Aires, Argentina, 2015.

Kruth, Jeff. "Interface Control Document for Morehead State 21 M DSN X-band Upgrade Embedded Feed Electronic Monitoring & Control Subsystem (EMCS) ." April 5, 2019.

Kruth, Jeff. "Proposed Frequency & Timing at Morehead." 2017.

Kruth, Jeffrey A. "Corrected Block Diagram of NMC." April 5, 2019.

Kurtik, Susan C. "Multimission Considerations for the NASA Deep Space Network (DSN)." Paper presented at Ground System Architectures Workshop, El Segundo, Ca, March 14, 2002.

Levister, Kathrine, and Ronald Norman. "0158-Monitor DSN Mission Monitor Data Interface." Pasadena, CA: Jet Propulsion Laboratory, Dec 01 , 2015.

Lombardi, Michael A. "Time and Frequency from A to Z, H." *NIST.* September 26, 2016. https://www.nist.gov/time-and-frequency-services/h.

Lowe, Jeremiah. "NCM Current Display." April 2019.

Malphrus, Benjamin K. "21 M Space Tracking Antenna." January 2017. https://www.moreheadstate.edu/getattachment/College-of-Science/Earth-and-Space-

Sciences/Space-Science-Center/Satellite-Tracking,-Telemetry-Control-Services/satellite-(2).pdf.aspx?lang=en-US.

Morehead State University. "New Antenna and Day Sky." n.d.

Morehead State University. "Space Mission Operations Center." 2019. https://www.moreheadstate.edu/College-of-Science/Earth-and-Space-Sciences/Space-Science-Center/Laboratories-Facilities/Space-Mission-Operations-Center.

Morehead State University Space Science Center. "Satellite Tracking, Telemetry & Control Services." n.d. https://www.moreheadstate.edu/College-of-Science/Earth-and-Space-Sciences/Space-Science-Center/Satellite-Tracking,-Telemetry-Control-Services (accessed Oct. 14, 2018).

NASA. *About: Deep Space Network.* n.d. https://deepspace.jpl.nasa.gov/about/ (accessed April 1, 2017).

NASA Ames Research Center. *About Open MCT.* n.d. https://nasa.github.io/openmct/about-open-mct/ (accessed April 10, 2017).

O'Dea, J. Andrew. "Downlink Consolidated Software (DCS) Subsystem Operations Manual." DSN No. 837-036, Rev. X, February 24, 2017.

Payne, Cadance, et al. "The Resurrection And Repurposing Of The Vlg-10 (P2) Hydrogen Maser." August 2016.

Pechkam, Paul. "Monitor and Control Software for Ground Systems in the Deep Space Network." July 22, 2002.

Pham, Tim, and Alina Bedrossian. "Deep space network services catalog." DSN No. 820, 100, 2009.

Shin, Dong K. "Frequency and Channel Assignments." DSN No. 810-005, 201, Rev. C , 2014.

VertexRSI. "Antenna Control System Operation and Maintenance Manual." February 2005.

Wyatt, Jay, and Benjamin Malprus. "Morehead State University 21 Meter Antenna Upgrade to DSN Compatibility." *IND CubeSat Communications Briefing and Technical Interchange.* 2015.

Zia, Susan. "Uplink Consolidated Software (UCS) Subsystem Operations Manual." DSN No. 837-049, Rev. Q, September 13, 2016.