Spaceport Command and Control System Software Development

Nicholas Chapa
NASA Kennedy Space Center
Major: Computer Science
Control System Software Development
Fall Session
Date: 31 October 2018

Command and Control System Software Development

Nicholas A. Chapa[1]

Boise State University, Boise, Idaho 83725

The control system for the Space Launch System (SLS) and Orion capsule contains a multitude of displays for use in displaying information like vehicle health, sensor information, life support, etc. Currently users in the Launch Control Center (LCC) are using a list-based selection Graphical User Interface (GUI) to open various displays. However the display names are not intuitive as to what the display looks like nor what data is presented in a display. My project involves adding additional functionality to the current display selection GUI utilizing thumbnail images of each display for easy browsing. This new GUI allows users to find their desired display much faster without needing to memorize the layout of every display based only on the display's name.

Nomenclature

KSC	= Kennedy Space Center
GUI	=Graphical User Interface
LCC	=Launch Control Center
NASA	=National Aeronautics and Space Administration

I. Introduction

The control system's Display Selection GUI uses a simple list of display names that can be selected individually or as multiple displays at once to be opened. Users can create their own displays, each with a unique name, which would then be added to the list of possible displays. My project's purpose was to add another GUI view to the existing Display Selection GUI. This would give the users two separate GUI views of available displays. They would then have the option to use the original list view or they could choose a grid view containing a grid of selectable thumbnail images of each display (with the associated display name). The ability to change between the two GUIs gives users flexibility when searching for the displays required for their particular subsystem or subsections.

II. Objectives

The end goal of my project is to create a fully functional grid-formatted selection display containing a thumbnail and name for each display that can quickly and easily be changed between the thumbnail view and the traditional list view. To accomplish this I set the following milestones to deliver an end result that exceeds expectations.

Implement a thumbnail generation feature

Generate thumbnail for each display during code build time

Create new GUI grid view

Create functionality to enable user to transition between the new grid view and original list view

Create backing model to open display when a thumbnail is selected.

[1] Software Development Intern, NE-XS, NASA KSC, Boise State University

III. Approach

A. Training and Setup

The first three weeks of my internship were spent setting up the environment that would be needed to run/modify my project. This included programs and files that needed to be downloaded, dependencies to be set up, and obtaining authorization to use all the required programs. I also attended training on how to begin using all these new programs once I was granted access. Additionally, I reviewed how past interns had used these programs and what they had learned from them. I gained the most insight, however, from meeting with the team members who wrote the original code with which I would be working. They gave me a detailed walkthrough of code functionality, dependencies, and the relationships between classes. Receiving a walkthrough of the team's entire codebase was by far the most beneficial experience in setting up my workstation and development environment.

B. Creating Thumbnails

The first goal of my project was to implement a thumbnail generation feature, capable of creating a thumbnail image for every pre-existing display as well as creating a thumbnail image for any new displays created by users. However, the feature had some restrictions. The thumbnail creation process had to be built into the current build process (i.e. there couldn't be a separate process for building each display and another for creating that display's thumbnail). Additionally, the thumbnail for a new display (created by the user) must be made automatically, without requiring the user's intervention.

In order to accomplish this, I created two different methods that would function the same but execute differently. The first method I created makes a thumbnail of any new display created by a user; this would be executed once they saved their display, with it still on-screen. That way, the user does not need to take any additional action to generate the thumbnail. Previous thumbnails will be overwritten with the newest thumbnails from the latest "save" action to prevent duplicates or loss of image data. Since a thumbnail is essentially a screenshot, having all the graphical elements of a display visible on screen made it extremely simple to generate the thumbnail. The next method I created for the purpose of generating a thumbnail of a display during its build process. This one was considerably more difficult as there would be no graphical elements visible and the build process had to happen without any additional delays. Fortunately, a fulltime engineer had previously written code that had a similar function and allowed me to use it to form the basis of my new method. With the addition of their code, I was able to draw and save images of the displays during the build process without impacting the build time. Most importantly, since both methods override any existing thumbnail and save the thumbnail to the same size, file location, and name there is no risk of duplication.

C. Creating the new Display Selection GUI

The next major goal of my project (and the goal that took up the majority of my time) was to implement a grid view that would show each display's thumbnail and name to allow for easy browsing. The only requirements I needed to meet at first were for the GUI to transition from list to grid view smoothly and to keep other graphical elements in the same place.

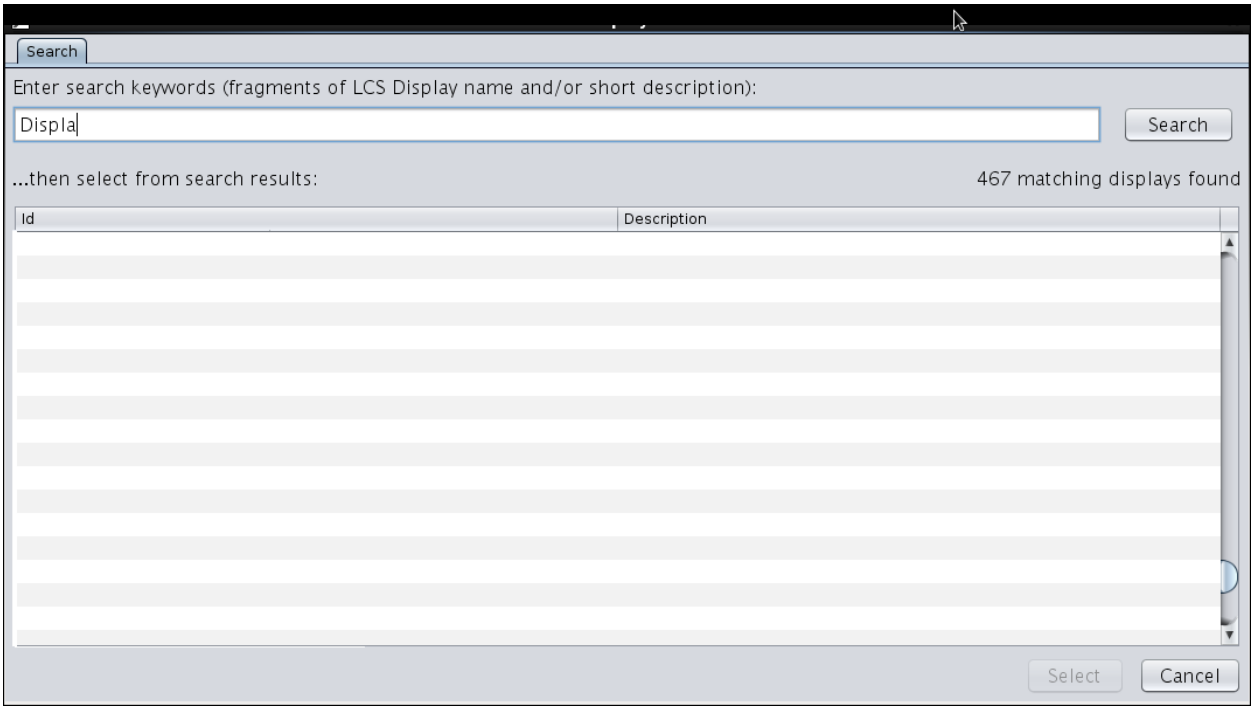

Figure 1. Original Display Selection GUI

My first idea for displaying the thumbnails with the display name was to simply add the thumbnails as an extra column in the existing list view, since that would be simpler than creating a new grid of images. I made a quick mockup of what a GUI designed like this would look like to show to my technical mentor and design lead and hear their opinions.

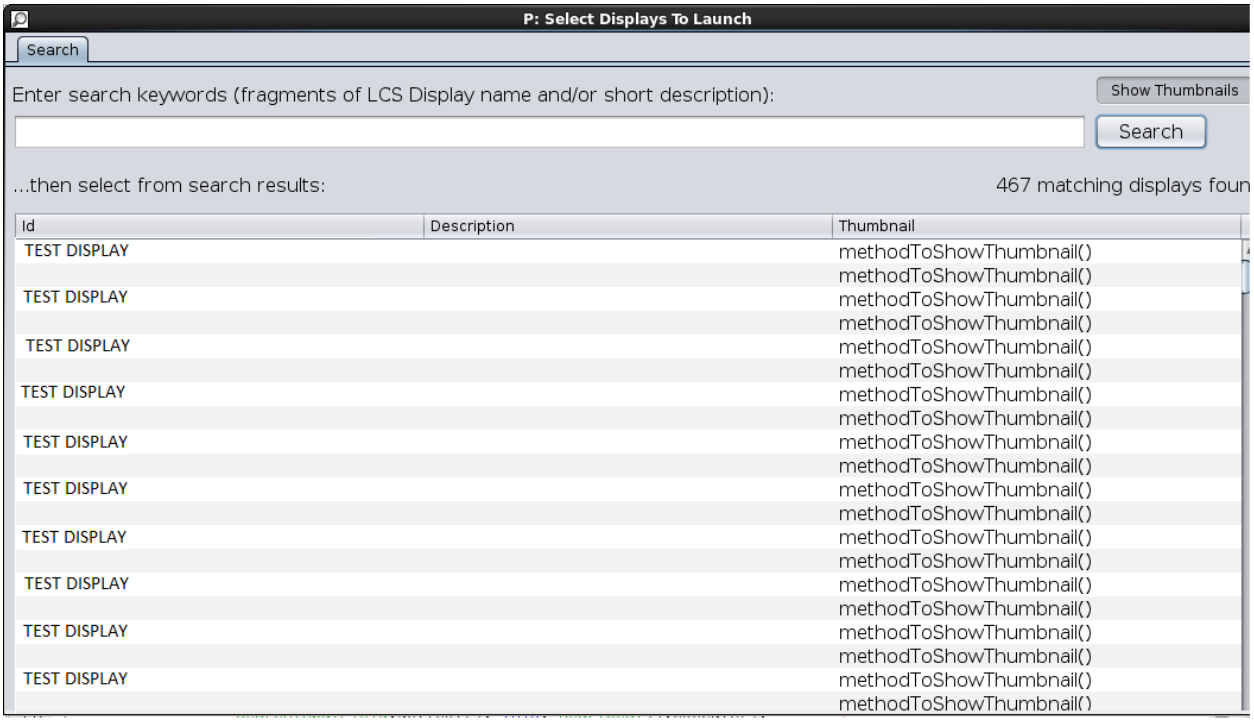

Figure 2. Original Display with extra Column for Thumbnails

They decided against it since the number of visible thumbnails would be limited in the list view and the design was not aesthetically pleasing and did not look modernized. So in my next prototype, I used a table to hold the thumbnail images and display names; this way I could control how each table cell was rendered and adjust it to hold an image. Additionally, this ensured the columns and rows were uniform in size to give a more professional look. The addition of a button made it easy to change between a grid of thumbnail images and the textual list of display names. The layout of other GUI components remained unchanged so as to preserve as much of the original flow of content as possible. Displays that did not have a thumbnail were left blank and showed only the name of the display while still being selectable. Additionally, resizing the window would cause the table to resize as well, keeping spacing between elements consistent while preventing "voids" of unused window space from occurring.

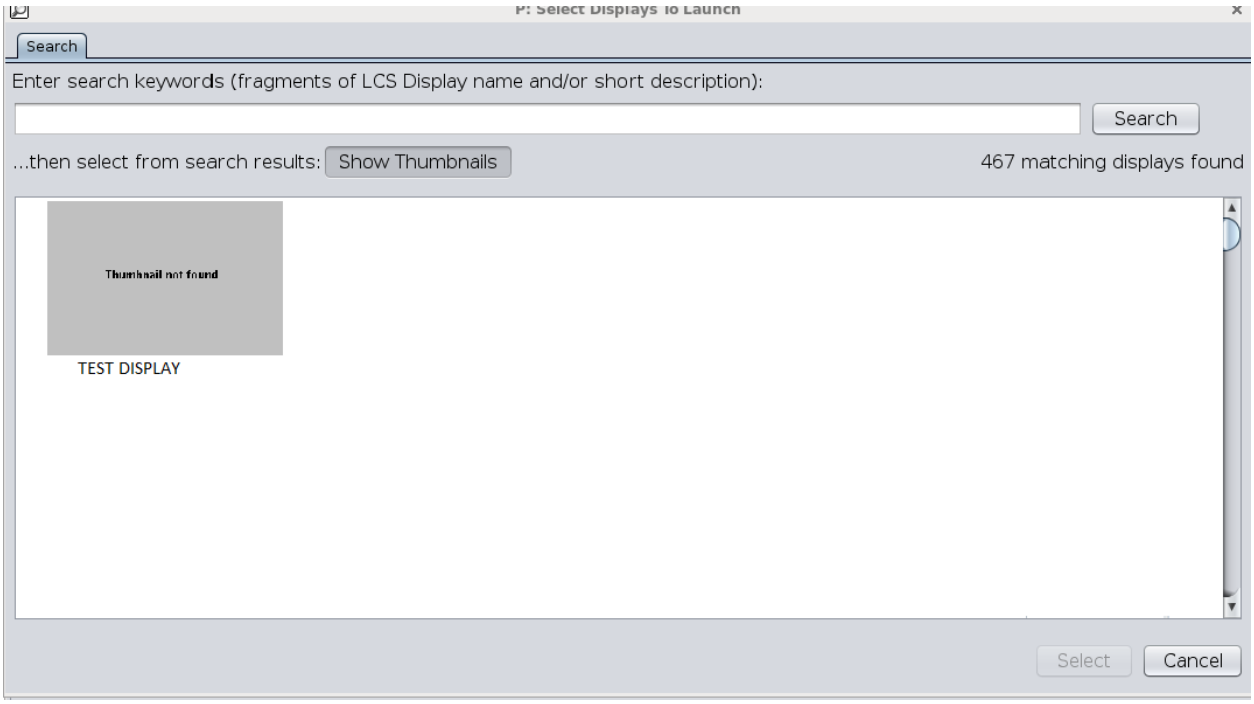

Figure 3. Grid View of Table of Thumbnails with Display Names

While this version functioned exactly as requested, there were several graphical issues that the lead requested be changed. The unused tab at the top of the page, the size and placement of the "Show Thumbnails" button, and the unnecessary "...then select from search results" text. In addition, the ops representative requested that I also implement a "preview" feature into the original list view, so that when a user selects a display in list view they can see an enlarged version of that display's thumbnail and have a good idea of what that display does. The lead also requested to replace the button with tabs to swap between grid and list view to provide a better flow of actions and more clearly establish the "list view" and "grid view" as separate entities that are not related as far as selection goes. I also added in dummy thumbnails to act as placeholders for displays that do not have a thumbnail image. To add even more functionality, I also adjusted the table to dynamically increase/decrease the number of columns to fit the size of the current window; this gives the user greater control over how many thumbnails they wish to see at one time while browsing. The last feature I added, to account for display names that may not be fully visible, was a tooltip popup for when the cursor hovers over a display's thumbnail which shows the full name of that display.

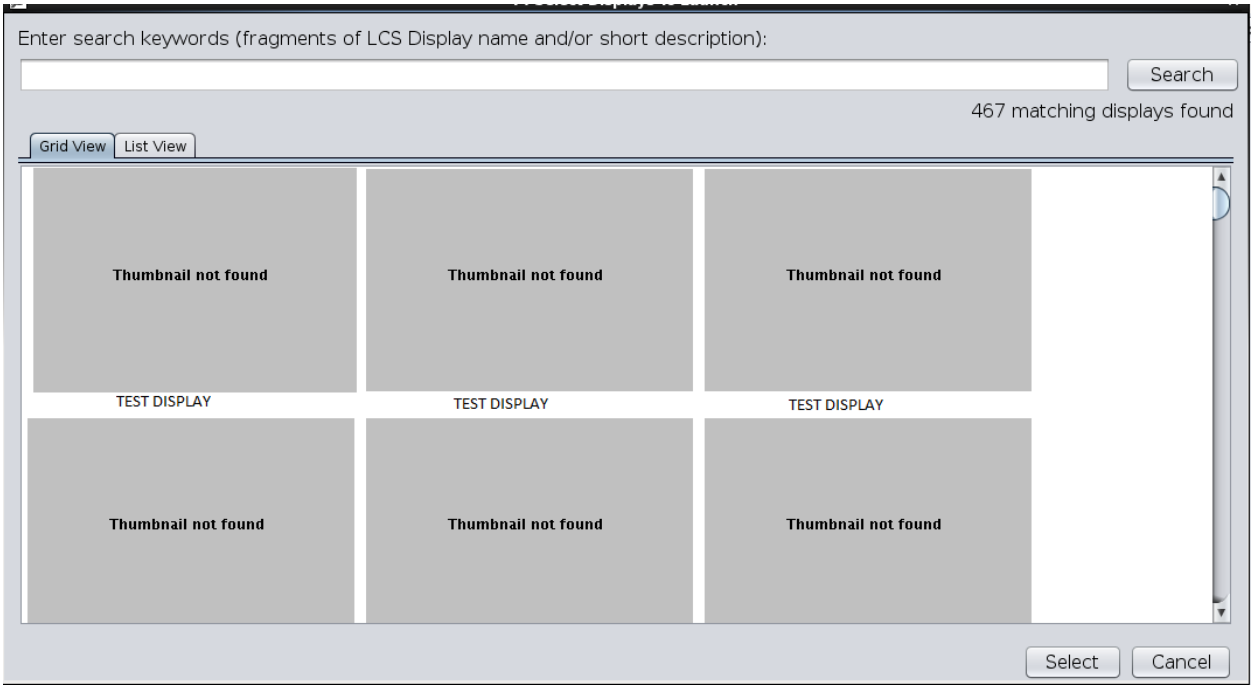

Figure 4. Final Tabbed Grid View at Default Size

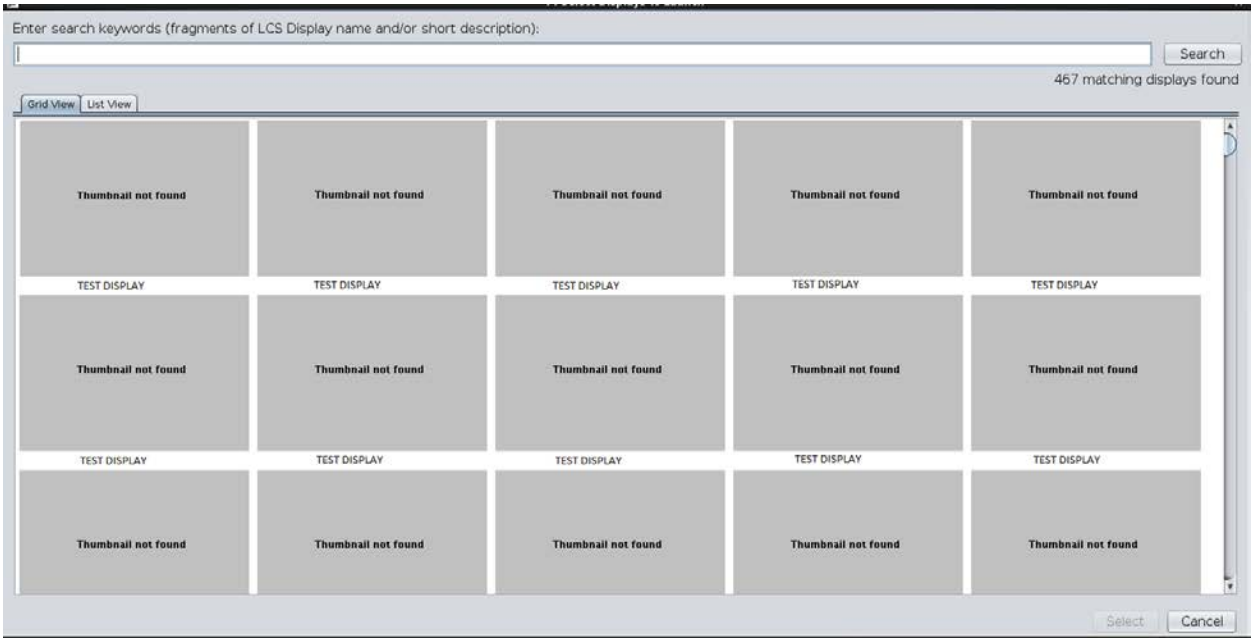

Figure 5. Final Tabbed Grid View at an Expanded Size

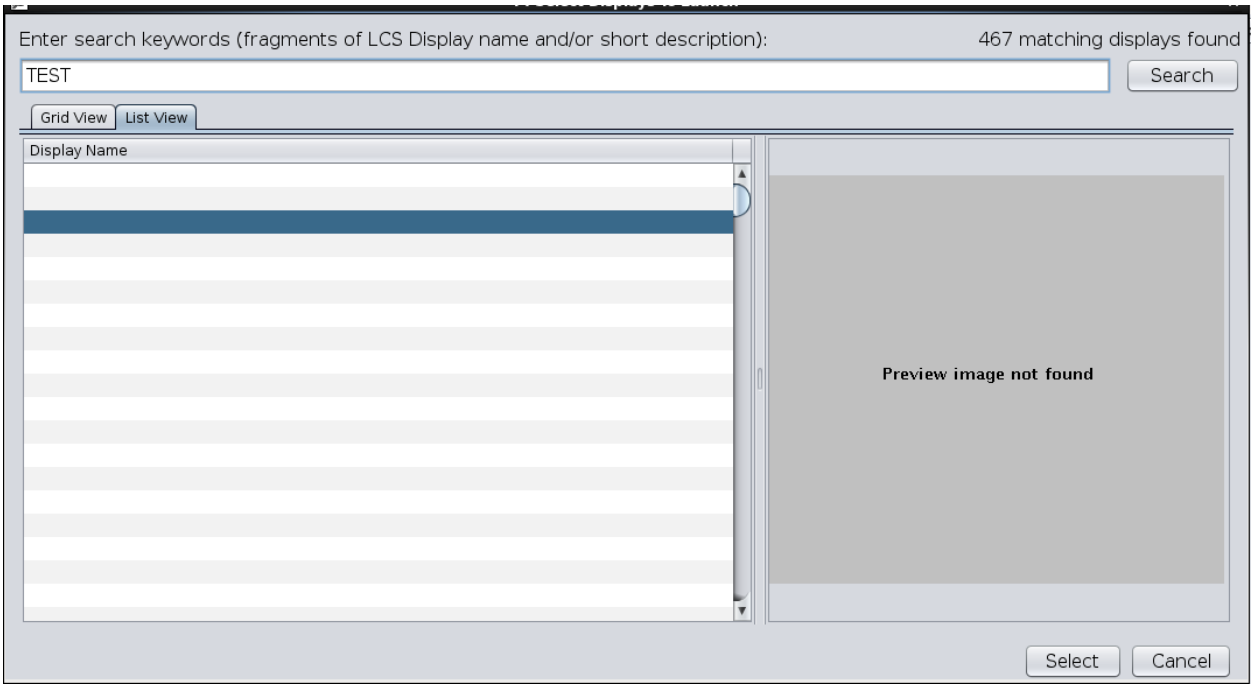

Figure 6. Final List View with Preview Image Visible

With the new Display Selection GUI finished, the design of my project was essentially complete. Before it could be fully finished, however, I had to create unit tests to ensure the functionality of my project.

D. Unit Testing

The NASA Procedural Requirement for Software Engineering requires extensive unit testing of any new software prior to integration into a class A codebase, so I spent several weeks creating unit tests and resolving issues that those tests found. I started making tests for my code to create thumbnails since that was the shortest part of my project. The first issue I ran into while writing those tests was ensuring that an invalid file path would not create a thumbnail. A bug I had not previously noticed would cause a black thumbnail image to be created if an invalid file path was referenced. This is obviously not desired behavior, but it was a fairly straightforward fix. Another issue was removing thumbnails after creating them for the unit tests (testing that a thumbnail exists after the method runs). This however, was not an issue with the function of my program but an issue of my implementation of the unit test so it was only a minor inconvenience.

Creating unit tests for the new display selection GUI went a little faster since I was able to use a lot of pre-existing unit tests as examples to base my new tests on. As such, the majority of my time on these tests was simply ensuring clicking display thumbnails would indeed select that image and that opening a selected display would open that display in a new window. The only issues I ran into during this phase was using the proper naming convention for referencing GUI controls (buttons, tabs, etc.).

After completing my unit tests and getting the approval of my technical mentor and design team members, I was able to submit it to code review, the next step in NASA's software approval process. During code review, all of the code I have written is reviewed by NASA software engineers/developers who will make comments and suggest any changes I should make to my project. Once code review is completed and the code promoted into the repository, then my project is officially finished.

IV. Conclusion

With the completion of my project, users within the LCC will have an improved method of browsing for displays by being able to see a thumbnail of each display. My final GUI has a much different look from what I envisioned at the beginning of my semester, but I am very satisfied with how it has turned out. My technical mentor, Jason, and the ops representative, Kevin, were also pleased with my end result which has given me an additional source of pride developing this GUI for NASA. I am told it could be implemented into the release for users to begin testing as early as the next software build. Since this particular project was initially requested by multiple system users, I believe they will be extremely satisfied with the results of my internship at KSC.

Acknowledgments

I never would have been able to accomplish my work and have my programs run as smoothly as they do without the help of so many others here at NASA. My mentor Jamie and co-Intern Lead Jill helped get me settled in and explained how the NE-XS branch operates and how we interns would fit into the work being done for Exploration Ground Systems. I'd like to thank Jason for walking me and the other interns through how all the programs this project uses work and how I should be using them (as well as giving feedback on all my work). Mario and Thong, members of the displays team, left KSC before my internship ended but they explained so much of the code to me that I needed to use; without their help I would have been incredibly lost. Sam on the system applications team also spent a lot of time helping me understand the graphical code and answered all my questions. Jonathan, a full time engineer from another team, who was a great guy and would stop by often to see how the other interns and I were doing, helped with a lot of simple issues I ran into. Working here has been an incredible experience and I'm thankful to everyone else in the organization who put up with the intern antics; this place has the best work environment I've ever been in and I hope to return someday.

References

N/A