**Software Requirements Specification**

**For NASA International Space Station (ISS)**

**Version 4.0**

**Prepared by André Akinyele, Jeffery Dovan, Erika Guthrie, Tracy Hill, Pyae Kyaw, and Derek Stine**

**November 12, 2017**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| André Akinyele | 10/2/17 | Initial draft | 1.0 Draft 1 |
| Pyae Kyaw | 10/5/17 | Edited | 1.0 Draft 1 |
| André Akinyele  André Akinyele  Derek Stine  André Akinyele | 10/6/17  10/20/17  10/22/17  11/9/17 | Updated  Updated  Edited  Updated and revised context diagram | 2.0 Draft 2  3.0 Draft 3  3.0 Draft 3  4.0 Draft |
| Derek Stine | 11/12/2017 | Updated for milestone 3 with UI | 5.0 Draft |

**Table of Contents**

[**Introduction**](#_gjdgxs) **4**

[Purpose](#_30j0zll) 4

[Software Scope and Software Features](#_1fob9te) 4

[References](#_3znysh7) 4

[**Overall Description**](#_2et92p0) **4**

[Software Perspective](#_tyjcwt) 4

[User Classes and Characteristics](#_3dy6vkm) 5

[Operating Environment](#_1t3h5sf) 8

[Design and Implementation Constraints](#_4d34og8) 8

[User Documentation](#_2s8eyo1) 9

[Assumptions and Dependencies](#_3rdcrjn) 9

[**System Features**](#_26in1rg) **13**

[Astronaut Input Data](#_lnxbz9) 13

[Application Import and Display Output Data](#_35nkun2) 15

[Astronaut Trainer Output and Spacewalk Procedure Writer Input Data](#_1ksv4uv) 16

[Translation Path Output and Map Path Input Data](#_44sinio) 16

[Dijkstra Algorithm](#_2jxsxqh) 17

[**External and Internal Interface Requirements**](#_z337ya) **20**

[User Interfaces Overview](#_3j2qqm3) 20

[Hardware Interfaces](#_1y810tw) 21

[Software Interfaces](#_4i7ojhp) 21

[Communication Interfaces](#_2xcytpi) 21

[**Other Nonfunctional Requirements**](#_1ci93xb) **22**

[Performance Requirements](#_3whwml4) 22

[Safety Requirements](#_qsh70q) 22

[Software Quality Attributes](#_3as4poj) 23

[**Appendix A: Data Dictionary and Data Model**](#_1pxezwc) **23**

[**Appendix B: Analysis Models**](#_2p2csry) **23**

# Introduction

## Purpose

This software requirements specification (SRS) defines the software functional and nonfunctional requirements for release 1.0 of the NASA Path software program. As such, this document is intended for members of the project team that will implement and verify the accurate functioning of the software. However, unless otherwise noted, all requirements specified here are high priority and committed for release 1.0.

## Software Scope and Software Features

Developing a NASA Path will allow an algorithm to calculate valid paths around the exterior of the International Space Station (ISS) using a 3D model, which navigating crewmembers from their chosen start location to their chosen end location; establishing a path’s shortest distance; avoiding a path’s encounter with the least hazards; and determining a path’s validity taking into account entered arm reach. Also, the software includes a 3D rendering for the exterior of the ISS and path using handrails and structural beams, which includes arm reach entered from ranges between 4ft. to 7ft., and a list of some objects protruding from the structural surface. Finally, the software will include rendering paths in the software for users, where compatibility with “DOUG” application is of less concern.

## References

GitHub. (2017). Cytoscape.js (library that has Dijkstra Algorithm). Retrieved from https://github.com/cytoscape/cytoscape.js

GitHub. (2017). EVANav: Navigation for space walks (client repo). Retrieved from https://github.com/darenwelsh/EVANav

GitHub. (2017). Lovetostrike/nasa-path-finder/demo (team demo). Retrieved from https://lovetostrike.github.io/nasa-path-finder/demo.html

GitHub. (2017). Lovetostrike/nasa-path-finder (team repo guide). Retrieved from https://lovetostrike.github.io/nasa-path-finder

GitHub. (2017). STL file viewing. Retrieved from https://github.com/blog/1465-stl-file-viewing

# Overall Description

## Software Perspective

Developing a NASA Path software includes features that will inform the client as to which path is the shortest distance, which path encounters the least hazards, and which path requires longer arm reach when a crewmember is at a particular location and needs to get to another place on the ISS. The context diagram below in *Figure 1* illustrates the external entities and software interfaces. The software is projected to evolve over four (4) Milestones, eventually consisting of:

1. An algorithm that calculates valid paths around the exterior of the ISS using a 3D model.
   1. Determine crewmember’s particular location
   2. Navigate crewmembers
   3. Establish path(s) shortest distance
   4. Encounter the least hazards
   5. Determine path(s) requirement for within the specified arm reach
2. Maps and navigation for the exterior of the ISS using handrails and structural beams.
   1. Arm reach (from a slider that ranges from 4ft. to 7ft.)
   2. List of some objects protruding from structural surface
3. Rendering the paths in the software for user.

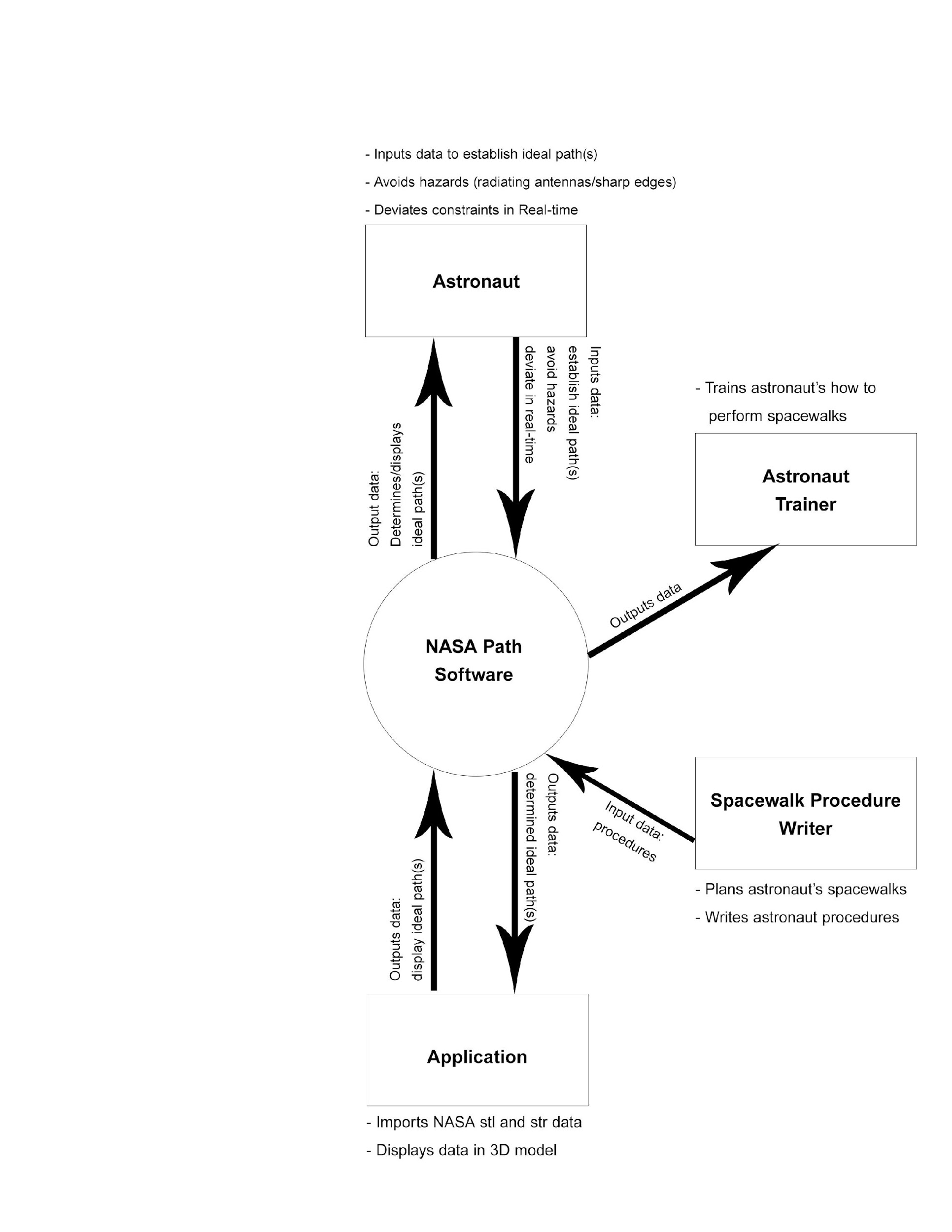
## User Classes and Characteristics

Astronaut (favored) An Astronaut is a person who is trained to travel in a spacecraft. There are several potential astronauts who are expected to use the NASA Path software. The software data input takes input from the astronauts to establish ideal path(s) for the astronaut. The software data output displays output data of the determined ideal path(s) for the astronaut. Astronauts will also use the software to avoid hazards, such as radiating antennas and sharp edges. Finally, astronauts will use the software in real-time when deviations to the timeline must be made, whether due to contingencies or timeline constraints.

Astronaut Trainer The Astronaut Trainer will train astronauts on how to perform spacewalks.

Spacewalk Procedure Writer The Spacewalk Procedure Writer will plan astronaut’s spacewalks and write astronaut procedures.

Application The application will import NASA stl and str data and display that data in a 3D model. After the Astronaut input coordinates and the software establishes the ideal path(s) that avoid hazards and deviates constraints in real-time, the software outputs the determined ideal path(s) to the application. The application then displays the ideal path(s) to the software, where the software determines/displays the ideal path(s) to the Astronaut.

*Figure 1*. Context diagram of the NASA Path software.

## Operating Environment

OE-1: The NASA Path software features shall operate with the following Mac or PC systems: Mac OS X (version 10.7.x or higher), Windows 7 or higher, and Ubuntu 14 or higher with 3.4 GHz CPU or faster.

OE-2: The NASA Path software features shall operate on 8 GB or more of RAM.

OE-3: The NASA Path software features shall operate with at least 1 GB or more of GPU for smooth operation of 3D software.

OE-4: The NASA Path software features shall operate with hard disks at least 10 GB or more of free disk space.

OE-5: The NASA Path software features shall operate with a monitor, keyboard, and mouse.

OE-6: The NASA Path software features shall operate with a TCP/IP network Internet connection or data connection to upload/download files.

OE-7: The NASA Path software features shall operate with Chrome, Firefox, and Internet Explorer (IE) Web browsers.

## Design and Implementation Constraints

DC-1: The NASA Path software model objects include:

* Handrails (all the yellow/gold or red handrails are to be used)
* Arm reach (user enters number from ranges between 4 ft. to 7ft.)
* Hazards (made by the development team as a list of some objects that protrude from the structural surface to showcase the functionality)

DC-2: The NASA Path software features shall use a Web-based 3D model application.

DC-3: The NASA Path software performance features shall accommodate a 3D model that:

* Starts from a crewmember’s entered location
* Navigates crewmembers to entered destination
* Establishes a shortest path and 2 alternate paths
* Avoids a path’s encounter with the least hazards
* Determines a path’s requirement based on entered arm reach

## User Documentation

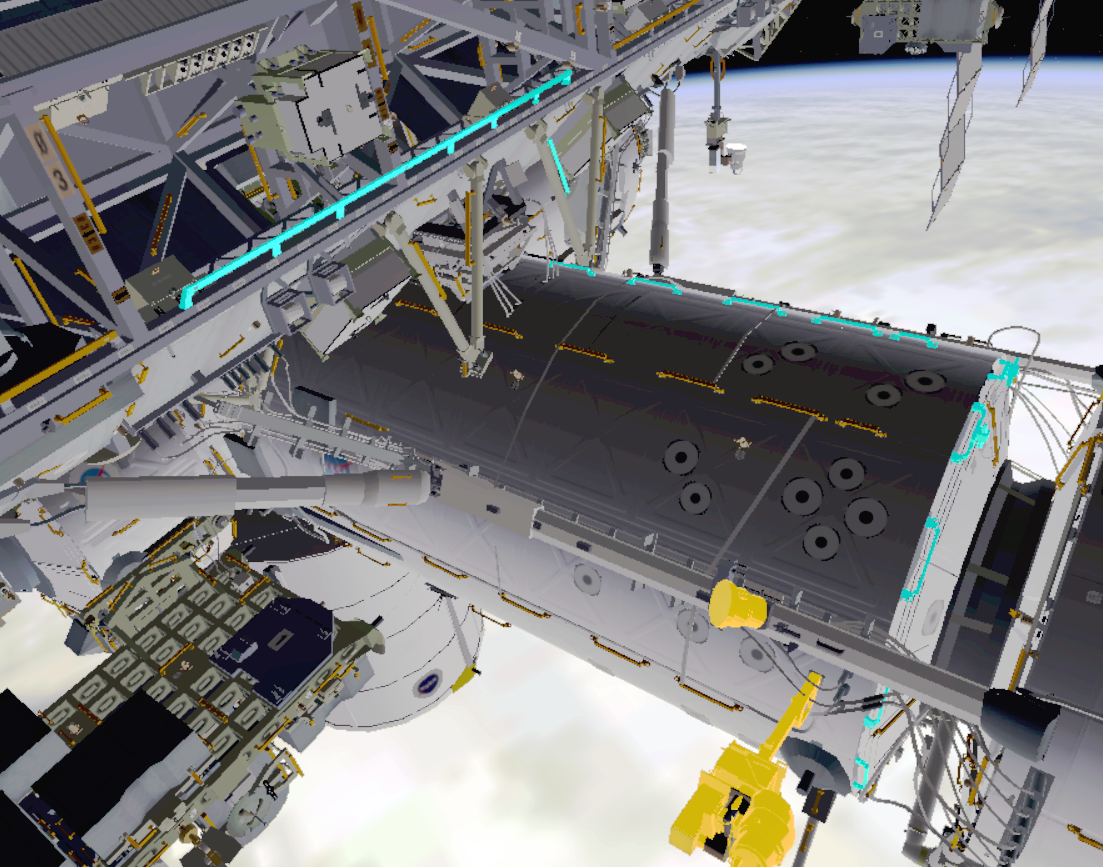
UD-1: The NASA Path software documentation shall be a README file.

## Assumptions and Dependencies

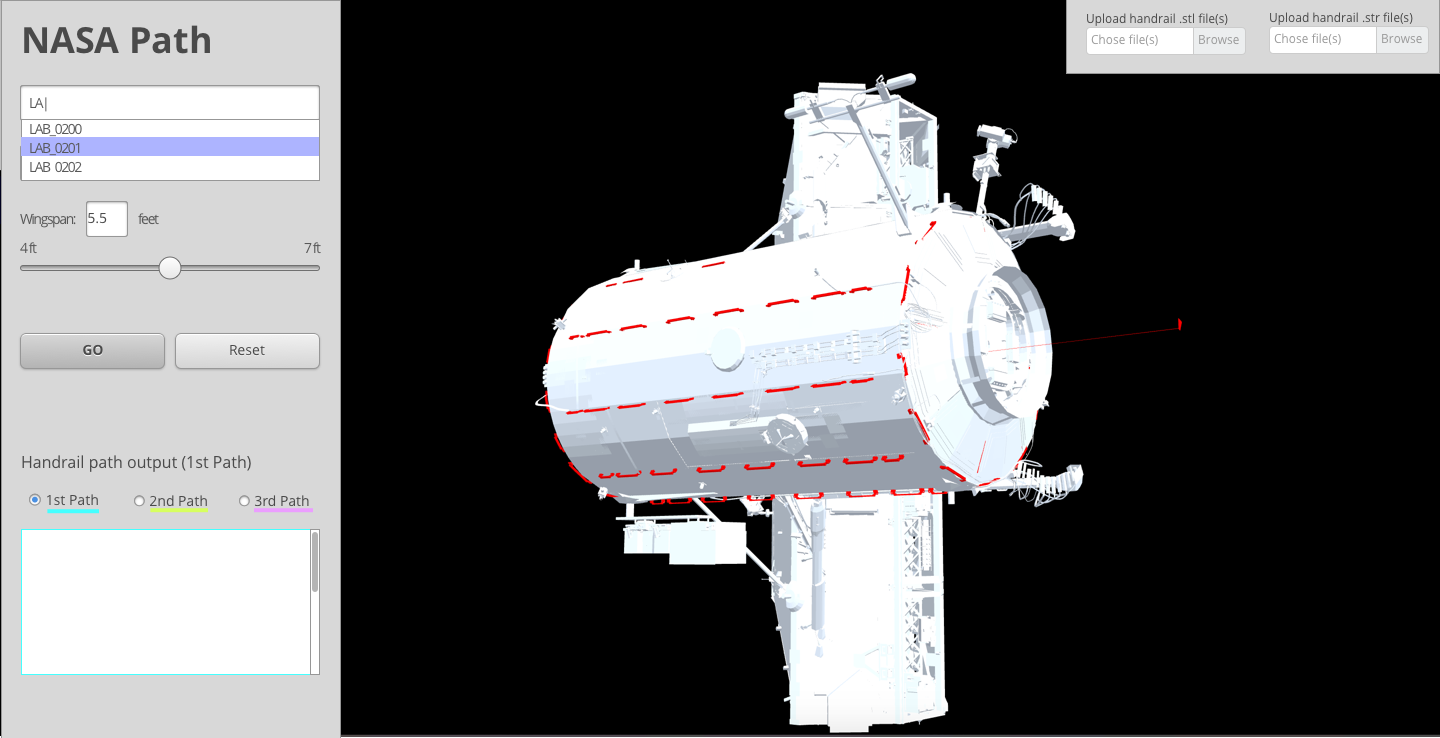
AS-1: The development can be web-based.

AS-2: There are no software language or technology preferences.

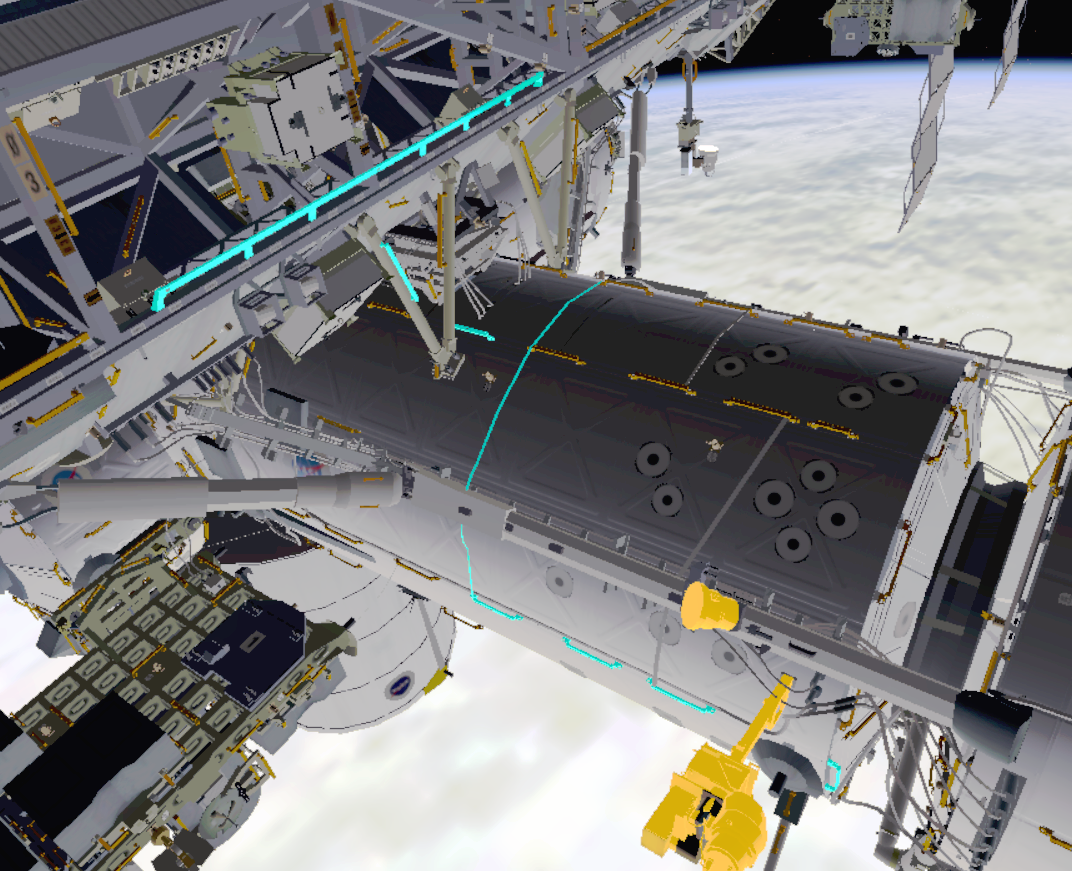
AS-3: Application provides client with a 3D model, which highlights handrail paths and hardware, similar to the illustrated below in *Figs. 2–3*.



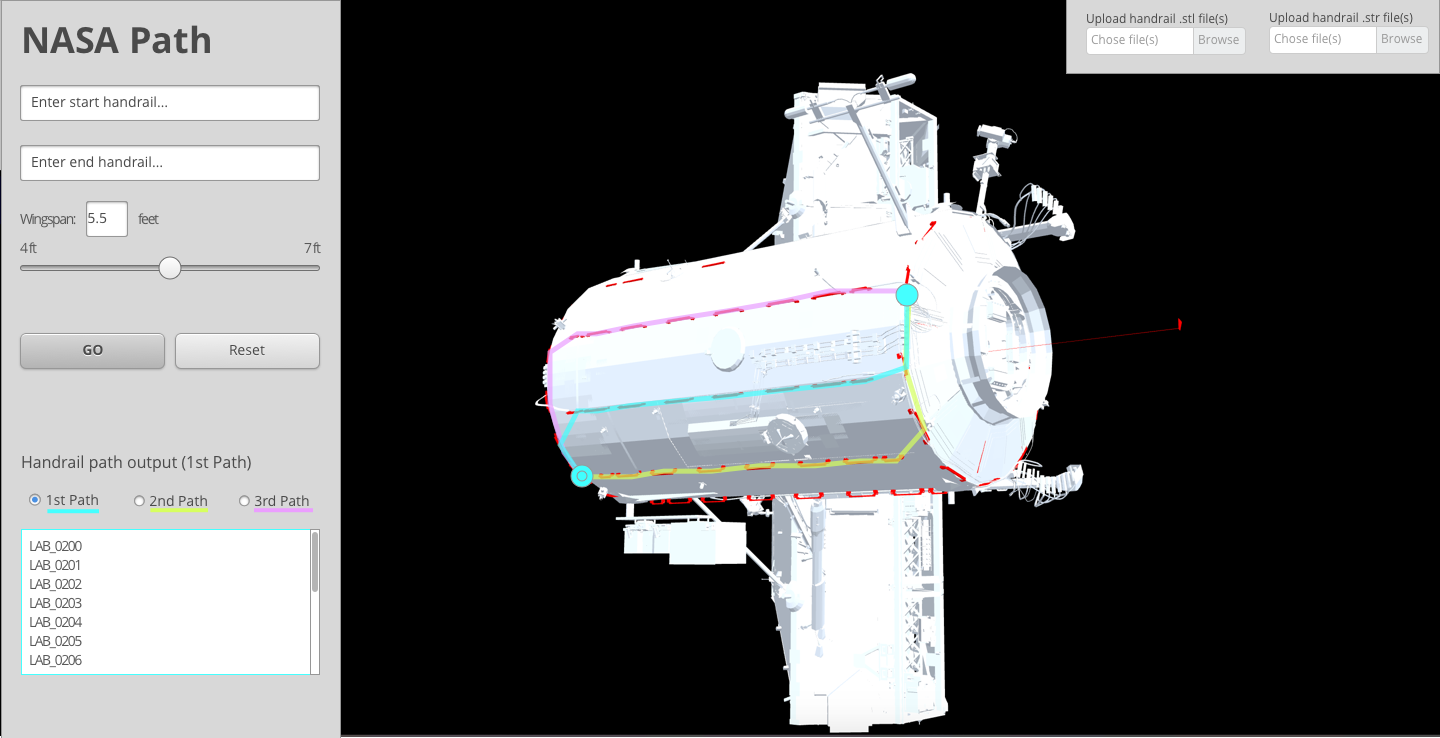
*Fig. 2*: 3D model 1. Crewmember must maneuver around hardware highlighted in yellow  
(paths highlighted in cyan).



*Fig. 2.5*: ISS path 1. Handrails in NASA Path application model highlighted in red.



*Fig. 3*: 3D model 2. Crewmember routed away from the hardware highlighted in yellow  
(paths highlighted in cyan).



*Fig. 3.5*: ISS path 2. Top three paths calculated highlighted in 1) cyan, 2) magenta and 3) yellow.

AS-4: If team formats data as shown below in Fig. 4, “DOUG” application developers can import it into the application. Format uses nodes on a graph (a scene-graph), where each node in the graph contains a transformation of the frame that its model and child nodes would be relative to. Each node is presented with five (5) consecutive lines of text. Compatibility with the “DOUG” application is not covered in the scope of this project but may be added in future versions. The software will use its own 3D rendering interface.

|  |  |
| --- | --- |
| <unique\_node\_name>  <geometry\_file\_name> or "SYSTEM"  <x> <y> <z>  <pitch> <yaw> <roll>  <parent\_node\_name> or "NULL" |  |

*Fig. 4*: “DOUG” application data format for importing into application.

AS-5: Team can estimate any path that requires a “plane change” by providing more time.

AS-6: “DOUG” application is shareable upon request but has proven too slow to obtain within the project timeline in order to facilitate development during this project (future projects could add compatibility).

AS-7: “DOUG” application developers may not be willing to open-source the application for extensions.

AS-8: No data to determine which handrail paths are faster than others.

AS-9: There are no depictions of hazards to avoid.

DE-1: The operation of the NASA Path software features depends on supporting more than one browser (Chrome, Firefox, and IE), if Web-based.

DE-2: The operation of the NASA Path software features depends on software and/or technology fostering long-term support from collaborative development.

DE-3: The operation of the NASA Path software features depends on a “white box” delivery, rather than a “black box” delivery.

DE-4: The operation of the NASA Path software features depends on a clear and simple user interface (UI).

DE-5: The operation of the NASA Path software features depends on an input UI that includes:

* Start and end points
* Optional waypoints in the middle of the route
* Options for route determination from top 3 paths

DE-6: The operation of the NASA Path software features depends on the following:

* Avoiding crew hazards (things that hurt crew or their suit)
* Avoiding hardware hazards (things that could be hurt by crew)
* Minimizing rotations and plane changes
* Minimizing fields to enter crewmember’s wingspan

DE-7: The operation of the NASA Path software features depends on an output interface (OI) that includes being/having:

* Transparent
* Sequential lists of handrail numbers
* Distance between each pair of handrails
* Format using nodes on a graph that represents handrail and hazard locations, similar to the format illustrated in *Fig. 4*.

DE-8: The operation of the NASA Path software features depends on measuring distances between points in compliance with arm span length.

DE-9: The operation of the NASA Path software features depends on using a right-handed coordinate system, where rotation is about the “Y-axis”, “Z-axis”, typically applied in Pitch-Yaw-Roll order, similar to the “DOUG” application that uses a right-handed coordinate system.

DE-10: The operation of the NASA Path software hazard features depends on the team making a list of some objects that protrude from the structural surface to showcase the functionality.

DE-11: The operation of the NASA Path software features depends on designing the software with highlighted paths similar to Google Maps with Navigation using a 3D model.

DE-12: The operation of the NASA Path software features depends on estimating any path that requires a “plane change” (i.e., that will require more time).

DE-13: The operation of the NASA Path software features depends on adding about 1 foot volume around any model with the names that include the word “antenna” and designating those models as a “Keep Out Zone.”

# System Features

## Astronaut Input Data

An Astronaut inputs data to establish ideal path(s), avoid hazards such as radiating antennas/sharp edges, and deviate constraints in real-time. When the Astronaut inputs Start Handrail and End Handrail values, the software establishes ideal path(s) that avoid hazards and deviates constraints in real-time. Priority = High.

**3.1.2 Stimulus/Response Sequences**

Stimulus: Astronaut inputs Start Handrail and End Handrail values.

Response: Software establishes/determines ideal path(s) that avoid hazards and deviates constraints in real-time.

Stimulus: Astronaut clicks drop-down list and selects the optional Middle Waypoint Handrails

Response: Software selects and displays the optional Middle Waypoint Handrails. Software selects and displays Crew Hazards to avoid.

Stimulus: Astronaut inputs integer values to determine the astronaut’s wingspan.

Response: Software displays astronaut’s wingspan (the distance this crewmember can reach between handrails).

**3.1.3 Functional Requirements**

|  |  |
| --- | --- |
| Input.StartHandrail.EndHandrail:  Input.StartHandrail.EndHandrail.Enter: | The software shall let an Astronaut input start handrail and end handrail values.  The software shall establish ideal path(s), avoid hazards, and deviate constraints in real-time. |
| Click.DropdownList.MiddleWaypointHandrail.Select:  Click.DropdownList.MiddleWaypointHandrail.Select. Display: | The software shall let an Astronaut click on a drop-down list to select the optional middle waypoint handrails.  The software shall select and display the optional middle waypoint handrails. |
| Input.IntegerValue.Enter:  Input.IntegerValue.Enter.Display: | The software shall let an Astronaut input an integer value to determine the astronaut’s wingspan (distance this particular crewmember can reach between handrails).  The software shall display astronaut’s wingspan (the distance this crewmember can reach between handrails). |

## Application Import and Display Output Data

The NASA Path application can import NASA stl and str data and display that data in a 3D model. After the Astronaut input coordinates and the software establishes ideal path(s) that avoid hazards and deviate constraints in real-time, the software then outputs the determined ideal path(s) to the application. The application then displays the ideal path(s) to the software, where the software determines/displays the ideal path(s) to the Astronaut. Priority = High.

**3.2.1 Stimulus/Response Sequences**

Stimulus: Astronaut inputs Start Handrail and End Handrail values to establish ideal path(s), avoiding hazards and deviating constraints in real-time, to NASA Path software.

Response: Software establishes ideal path(s), avoiding hazards and deviating in real-time, outputting the determined ideal path(s).

Stimulus: The application determines the ideal path(s) and displays the ideal path(s) in the NASA Path software interface.

Response: Software outputs data determining/displaying the ideal path(s) to Astronaut.

**3.2.2 Functional Requirements**

|  |  |
| --- | --- |
| Input.StartHandrail.EndHandrail:  Input.StartHandrail.EndHandrail.Enter: | The software shall let an Astronaut input start handrail and end handrail values.  The software shall establish ideal path(s), avoid hazards, and deviate constraints in real-time. |
| Import.Path(s):  Import.Path(s).Display: | The software shall import stl and str data to model and determine ideal path(s).  The software shall display the determined ideal path(s). |
| Display.Path(s):  Display.Path(s).Output: | The software shall let an Astronaut view the determined ideal path(s).  The software shall display the determined ideal path(s). |

## Astronaut Trainer Output and Spacewalk Procedure Writer Input Data

An Astronaut Trainer can use the software to train astronauts how to perform spacewalks using written astronaut procedures from the Spacewalk Procedure Writer. Priority = Low.

**3.3.1 Stimulus/Response Sequences**

Stimulus: Spacewalk Procedure Writer inputs astronaut procedures data to NASA Path software for Astronaut Trainer.

Response: Software outputs astronaut procedures data to Astronaut Trainer.

**3.3.2 Functional Requirements**

|  |  |
| --- | --- |
| Input.Procedures:  Input.Procedures.Enter: | The software shall let a Spacewalk Procedure Writer input astronaut procedures similar to the Astronaut.  The software shall output astronaut procedures data to Astronaut Trainer. |
| Display.Procedures:  Display.Procedures.Output: | The software shall let an Astronaut Trainer view astronaut procedures.  The software shall display the astronaut procedures. |

## Translation Path Output and Map Path Input Data

A Translation Path can use the software to map out translation paths for Astronauts. Priority = High.

**3.4.1 Stimulus/Response Sequences**

Stimulus: Software inputs map path(s) data to Translation Path.

Response: Translation Path maps out path(s).

Stimulus: Translation Path translates the path(s) to NASA Path software.

Response: Software displays path(s) to Astronaut.

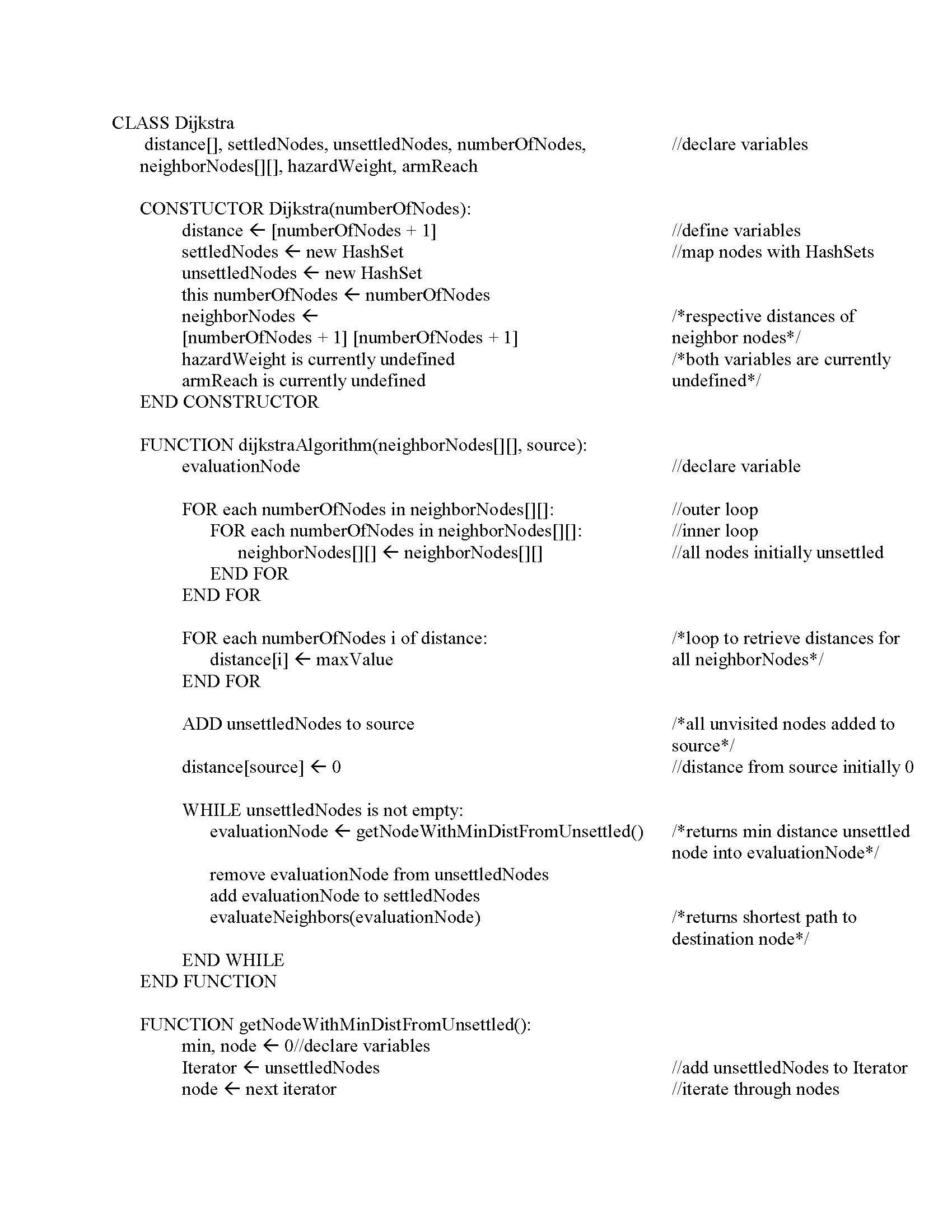
**3.4.2 Functional Requirements**

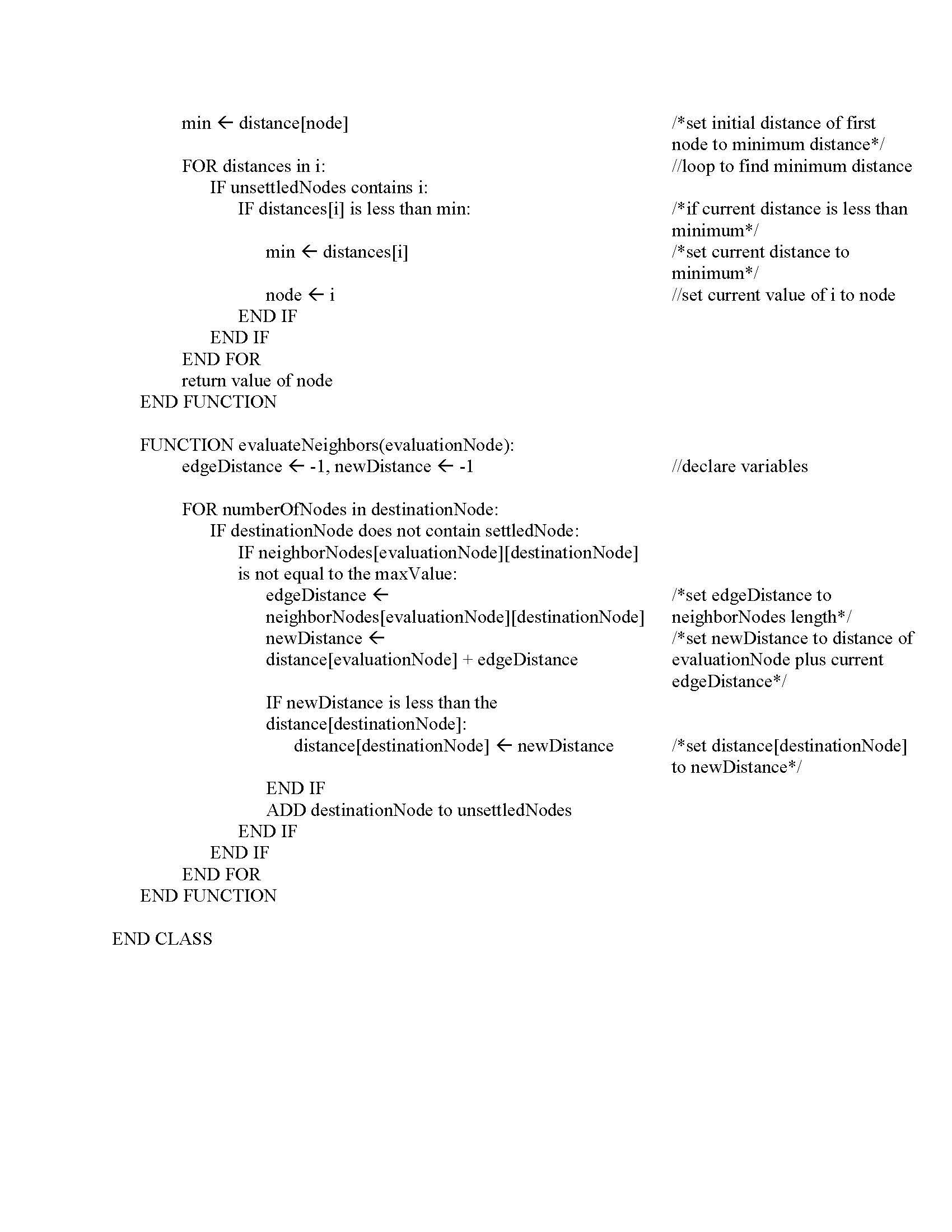
|  |  |
| --- | --- |
| Input.MapPath(s):  Input.MapPath(s).Enter: | The software shall let input data map path(s).  The software shall transmit map path(s) data to translation path. |
| TranslationPath.MapPath(s):  TranslationPath.MapPath(s).Translate: | The software shall let Translation Path map out path(s).  The software shall let Translation Path translate path(s). |
| Display.TranslationPath:  Display.TranslationPath.Output: | The software shall let an Astronaut view the translated path(s).  The software shall display the translated path(s). |

## Dijkstra Algorithm

The Dijkstra Algorithm is used to find the shortest path between two points in a graph. Priority = High.

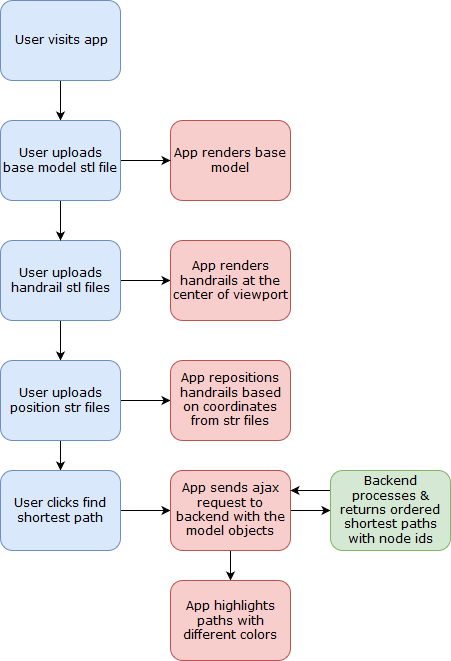
**3.5.1 Dijkstra Algorithm Pseudocode**





# External and Internal Interface Requirements

## User Interfaces Overview



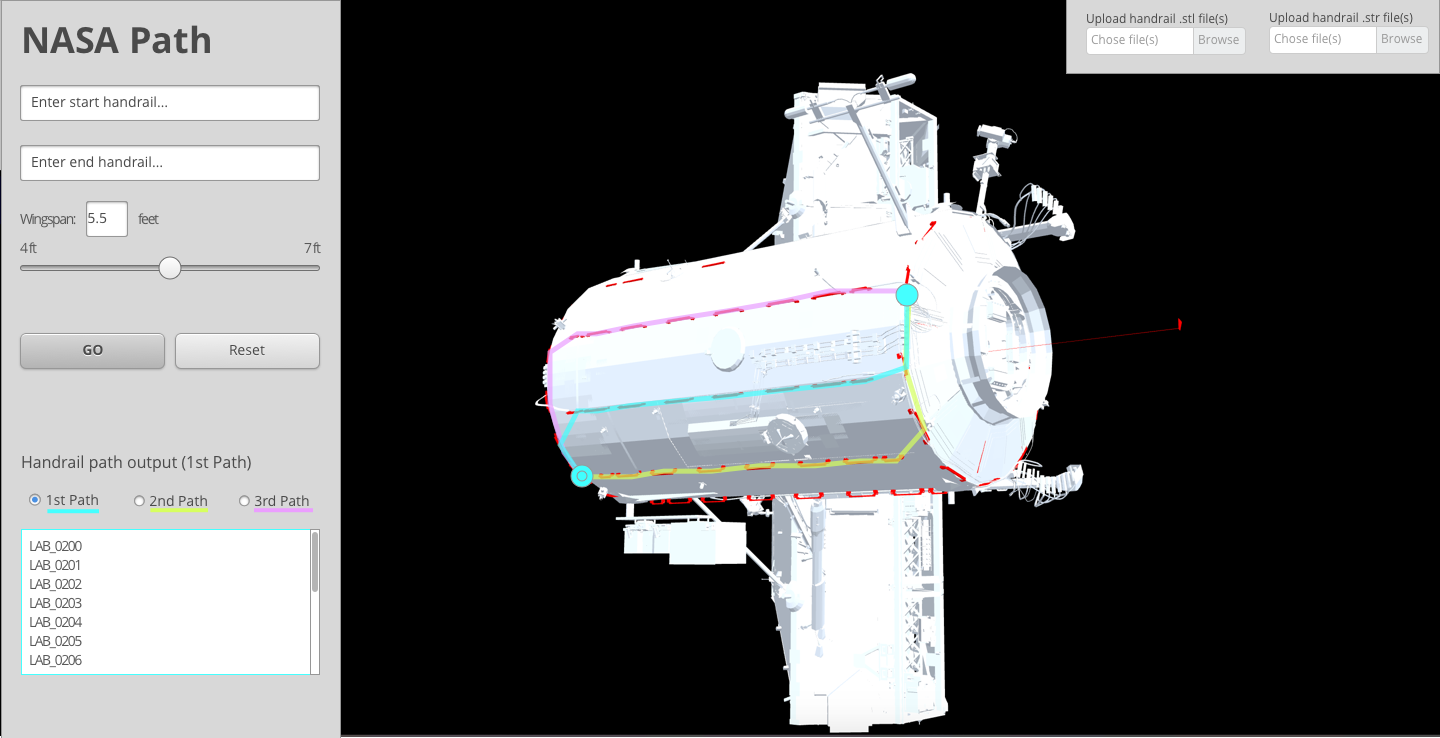
*Fig. 5*: User Interface Data Flow.

UI-1: The NASA Path software UI features shall be simple, using:

* A form with some pull down (drop down lists)
* A form with some slider input fields
* A modal if needed to provide additional information or confirmation on click

UI-2: The NASA Path software UI features shall include variables starting with:

1. Start handrail
2. End handrail
3. Optional middle waypoint handrails
4. Option to avoid crew hazards and sensitive hardware
5. Integer value for astronaut’s wingspan (the distance this particular person can reach between handrails)



*Fig. 5.5*: User Interface mockup with paths highlighted.

## Hardware Interfaces

HI-1: Monitor, keyboard, and mouse

HI-2: Internet connection or data connection

## Software Interfaces

SI-1: Internet

SI-1.1: The NASA Path software features shall utilize the Chrome, Firefox, and IE Web-browsers.

SI-1.2: The NASA Path software features shall operate on Windows 7 or higher, Mac OS X or higher, and Ubuntu 14 or higher.

## Communication Interfaces

CI-1: The NASA Path software features shall use the network TCP/IP.

# Other Nonfunctional Requirements

## Performance Requirements

PR-1: The NASA Path software shall accommodate a 3D model written in .stl format either in binary or text format.

PR-1.1: The 3D model shall inform the client where a crewmember is at a particular location written in .str format.

PR-1.2: The 3D model shall inform the client how to navigate a crewmember when that member needs to get to another place on the other end of ISS.

PR-1.3: The 3D model shall inform the client which path is the shortest distance.

PR-1.4: The 3D model shall inform the client which path encounters the least hazards.

PR-1.5: The 3D model shall inform the client which path requires longer arm reach.

PR-1.6: The NASA Path software calculates valid path(s) around the exterior of the ISS.

PR-1.7: The NASA Path software renders path(s) for user.

## Safety Requirements

SR-1: The NASA Path software must determine which path encounters the least hazards, such as radiating antennas and sharp edges.

SR-2: The NASA Path software must avoid crew hazards, things that hurt the crew or their suit.

SR-2.1: The NASA Path software UI must have an option to avoid crew hazards.

SR-3: The NASA Path software must avoid hardware hazards, things that could be hurt by the crew.

SR-3.1: The NASA Path software UI must have an option to avoid sensitive hardware.

SR-3.2: The NASA Path software hazards are to be made by the development team, which should consist of a list of some objects that protrude from the structural surface to showcase the functionality.

SR-3.3: If any of the model names include the word “antenna”, it has some, about one (1) foot, volume around it and should be designated as a “Keep Out Zone”.

## Software Quality Attributes

Availability-1: The NASA Path software features shall be available to Astronauts, Astronaut Trainers, Spacewalk Procedure Writers, and DOUG Application developers housed on GitHub (publicly).

Robustness-1: NASA Path software on GitHub is a passing build and allowing access to the software and its included features, visit lovetostrike.github.io/nasa-path-finder.

# Appendix A: Data Dictionary and Data Model

Data dictionary and data models are not required.

# Appendix B: Analysis Models

Analysis models are not required.