Software Requirements Specification for Flight Plan Editor

Cedrick Cooke
Ian Littke
Owen Roth-Lerner
Sander Scherman Garzon

CSCI 492: Spring 2017

Revision History

Name	Date	Reason for Change	Version
Ian Littke	2017-02-21	Initial document blocking	0.1
Cedrick Cooke, Owen Roth-Lerner, Sander Scherman Garzon	2017-02-23	Expanded sections 3 and 4	0.2
Ian Litke	2017-05-16	Reformat Document	0.3

Contents

1 Introduction

1.1 Purpose

This SRS (Software Requirements Specification) describes the software functional and non-functional requirements for release 1.0 of the Flight Plan Editor. This document is intended to be used by the members of the project team that will implement and verify the correct functioning of the system. Unless otherwise noted, all requirements specified here are high priority and committed for release 1.0.

1.2 Document Conventions

This document features some terminology which readers may be unfamiliar with. See section 1.5 for a list of these terms and their definitions. The format of this SRS is simple. Bold face and indentation is used on general topics and or specific points of interest. Font style is Computer Modern and the document is typeset using LaTeX;

1.3 Intended Audience and Reading Suggestions

This document is intended for all individuals participating in the initial release of Flight Plan Editor. Including, but not limited to, all project team members; our team mentor; and our CAP contact point.

1.4 Project Scope

The Flight Planning Editor will allow flight planners in the Civilian Air Patrol (CAP) to create their flight plans in a graphical interface and store them on a Secure Digital (SD) card. A detailed project description is available in the *Flight Plan Editor Vision and Scope Document*. The section in that document titled "Scope of Initial and Subsequent Releases" lists the features that are scheduled for full or partial implementation in this release.

1.5 Definitions, acronyms, and abbreviations

CAP

Civilian Air Patrol

FAA

Federal Aviation Administration

Garmin G1000

Integrated flight instrumentation system used in planes

GeoTIFF

Metadata standard which allows georeferenceing information to be embedded within a TIFF file

GPS

Global Positioning System

Sectional

Sectioned chart covering a section of an area, usually by state

SRS

Software Requirements Specification

SD Card

Secure Digital Card

TIFF

Tagged Image File Format

VFR

Visual Flight Rules

XML

eXtensible Markup Language

1.6 References

Vision and Scope	`https://github.com/CIOS-Digital/vision-scope`
FAA VFR Raster Charts	`https://www.faa.gov/air_traffic/flight_info/aeronav/digital_products/vfr/`

2 Overall Description

This section gives an overview of the proposed functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next section.

2.1 Product Perspective

This flight planning editor addresses a CAP emergency flight planning need. Specifically, this new software will allow a user to input latitude and longitude coordinates and view them on a map to develop flight plans. These flight plans can then be used in the Garmin G1000 navigation system.

2.2 Product Features

The most important feature of this product is the ability to read, edit, and write XML containing flight plans. This is accomplished through a user interface in which allow the user will enter coordinates which are then displayed on a map for reviewing and modification. Once the flight plan is finished, the user can save their plan to the SD card to be used in the G1000.

2.3 User Classes and Characteristics

There is only one type of user that will interact with the system: users that have the authority to create flight-plans. In the case of the Bellingham CAP, this is limited to two people. The user is expected to have knowledge of flying area and can read charts. The user shall also be expected to have latitude-longitude coordinates of waypoints of interest to include in the flight plan.

2.4 Operating Environment

The system will be able to run on Microsoft Windows products, starting with Windows 7. The system only focuses on Windows due to the availability of Windows machines utilized by the CAP.

2.5 Design and Implementation Constraints

The system's design and code shall be written in C# The User interface shall be written in XAML for Windows Presentation Foundation. All flight plans shall be recorded in XML.

2.6 User Documentation

The system shall provide hierarchical and cross-linked help system in HTML that describes and illustrates all system functions.

The system shall also offer tool-tips to assist in user functions

2.7 Assumptions and Dependencies

2.7.1 Assumptions

Flight operators can map around restricted areas and or altitude. The system will not alert the user if a flight path passes through restricted airspace or through a mountain. It remains up to the user to be able to through these issues.

It is a dependency that the SD card used to save the flight plans on are of the same make and model that is able to be read by the G1000 system.

3 System Features

3.1 Input Destination Coordinates

3.1.1 Description and Priority

In the top left corner of the application, there are two input windows for latitude-longitude coordinate input. Once the user enters the data, points on the sectional map will appear as well as a text list for each coordinate.

3.1.2 Use-Case Sequences

New waypoint

Step 1 User will click on the latitude text box to input the latitude, and complete the same for longitude.

Step 2 The map shall display the point respective to the latitude-longitude input.

User can then request to edit an existing waypoint.

The coordinate is displayed, and user will have the option to either save or discard the changes.

3.1.3 Functional Requirements

r1.1

There will be a text input box for both latitude and longitude coordinate input

r1.2

Upon entering coordinates, a waypoint will be drawn on the sectional map to represent its location

r1.3

Selecting an already created waypoint allows the user to modify the coordinates, or discard the waypoint

3.2 Display Sectional Map

3.2.1 Description and Priority

The application will display the sectional map as the main interface. The map will respond to input from latitude-longitude coordinate input as well as having the ability to pan around the map.

3.2.2 Requirements

Click map to add map waypoint

Step 1 A user can click on the map to input a waypoint at that coordinate.

Step 2 A new waypoint is displayed on the map.

Pan map

Step 1 A user can click-hold on the map and drag their mouse across the screen.

Step 2 The program will adjust the view of the map vertically and horizontally, in the opposite direction of the mouse pointers movement.

Zoom map

Step 1 A user may also scroll with the mouse-wheel.

Step 2 The map will either zoom in or zoom out depending on the mouse scroll.

3.2.3 Functional Requirements

r2.1

Upon starting the program, the sectional map will be displayed

r2.2

This map will have the ability to maneuver and pan based upon user mouse movement

r2.3

Map will also respond to changes in waypoints, or selecting different flight plans

3.3 Distinguish waypoint Coordinates

3.3.1 Description and Priority

As a user inputs more latitude-longitude coordinates, the program will draw a new waypoint onto the sectional map for viewing as well as having a small subtext containing the latitude-longitude coordinates for viewing to distinguish each waypoint.

3.3.2 Use-Case Sequences

Step 1 A user adds or removes a waypoint

Step 2 The map is updated to show the change in waypoints

3.3.3 Functional Requirements

r3.1

Each waypoint drawn on the map shall display the latitude and longitude coordinates

r3.2

Each waypoint will also be displayed as a list between other points for connections

3.4 Draw Flight Path on Section Map Between Coordinates

3.4.1 Description and Priority

The user should have the ability to not only add or delete points, but change in which order he connects one point to another. This allows the user to actually route between points rather than have to manually navigate between them.

3.4.2 Use-Case Sequences

New waypoint

Step 1 User adds a new waypoint

Step 2 Map updates to create a path from the previous waypoint to new waypoint

Remove waypoint

Step 1 User removes a waypoint

Step 2 Map updates to remove the path from the waypoints

3.4.3 Functional Requirements

r4.1

The point list shall have ability to edit ordering of latitude and longitude coordinates

r4.2

If adding new coordinates, the previous coordinate will connect to new one

r4.3

If coordinate is last on list, the path will end there

r4.4

The map shall update to show new path as the list is edited or extended

3.5 Open and Save Flight Plan in XML to SD Card

3.5.1 Description and Priority

The Garmin G1000 uses SD cards to store flight plans, and does so in the XML format. For this program to be of any use, it *must* support reading and writing these files.

3.5.2 Use-Case Sequences

Opening a file:

Step 1 User initiates an open action.

Step 1.1 If the user has an unsaved, open flight plan, prompt the user if they want to save it. If the user chooses to save the plan, complete the ‘saving a file’ section before continuing.

Step 2 User is prompted to select a file to open.

Step 3 The file is deserialized and displayed.

Step 3.1 If the file is invalid, corrupt, or otherwise fails to open, display a warning to the user.

Saving a file:

Step 1 User initiates a save action or a save-as action.

Step 2 If the user selected a save-as action, or if the current plan is not already associated with a file, the user selects the file location and file name they would like to use.

Step 3 The current flight plan is serialized to XML, and written to the file.

Step 4 After writing the file completes, a message is displayed to alert the user.

3.5.3 Functional Requirements

r5.1.1 The program shall let a user open files.

r5.1.2 The program shall prompt the user to select a file to open.

r5.1.3 After a plan is selected to open, if it fails, the user shall be alerted to the failure.

r5.1.4 After a plan is selected to open, if it succeeds, the user interface should automatically update to reflect the plan.

r5.2.1 The program shall let a user save files.

r5.2.2 If the program cannot detect where the file was opened from, or if the save-as option was used to initiate the save, the program shall prompt a user for a location to save the file.

r5.2.3 If an attempt to save a file fails, the user shall be alerted to the failure.

r5.2.4 If an attempt to save a file succeeds, the program shall display a message indicating the success.

4 External Interface Requirements

This section provides a description of all inputs into and outputs from the system. It also gives a description of the software and communication interfaces and provides a basic prototype of the user interface.

4.1 User Interfaces

The user interface is divided into two main sections: the map view on the right and waypoint editing on the left. The user may load up previously saved flight plans or create a new plan as well as modify the current selected plan. See figure 1 for a sample mockup interface.

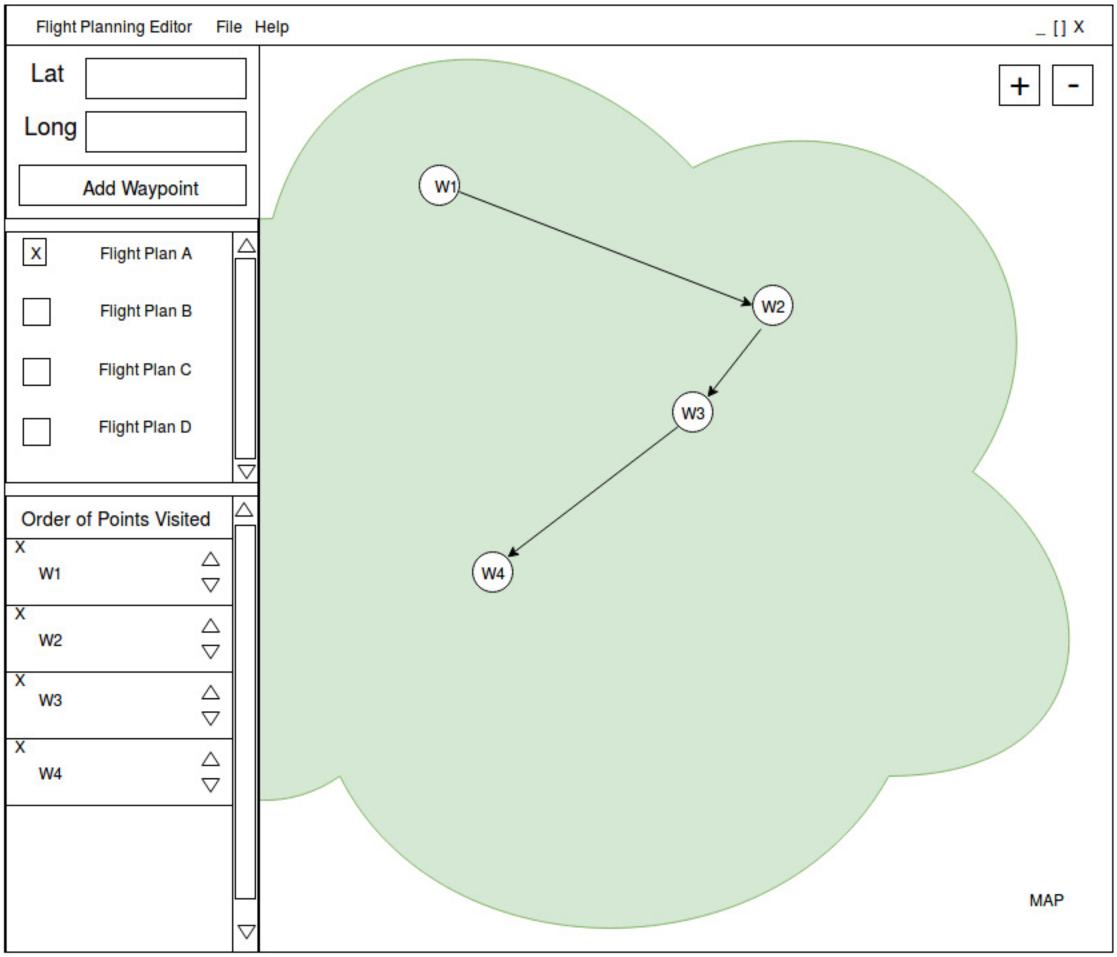

Figure 1: Example of an interface - final version may not be represented

4.2 Hardware Interfaces

There are no direct hardware interfaces, however the flight plans that are saved must be able to written to SD card.

4.3 Software Interfaces

No software interface requirements have been identified at this time.

4.4 Communications Interfaces

No communications interfaces exist.

4.5 Performance Requirements

The user interface should be able to render the maps within 1 minute of request from the user.

4.6 Safety Requirements

No safety requirements have been identified.

4.7 Security Requirements

No security requirements have been identified.

4.8 Software Quality Attributes

No additional software quality attributes at this time.