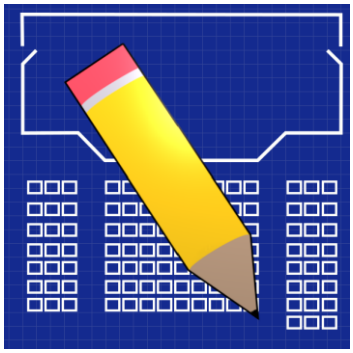


Software Engineering
Software Requirements Specification (SRS) Document

Ben Marshall, Lane Addison, Milton Moore, Jason Milloff, Ksenia Belikova
Radford University
readysetgoradford.weebly.com
10/25/16

Review & Approval

Requirements Document Approval History

Approving Party	Version Approved	Signature	Date
Project Manager	1.0.0	Ksenia Belikova	10/25/16
Dr. T. L. Lewis			

Requirements Document Review History

Reviewer	Version Reviewed	Signature	Date
Group Member	1.0.0	Lane Addison	10/25/16
Group Member	1.0.0	Benjamin Marshall	10/25/16
Group Member	1.0.0	Jason Milloff	10/25/16
Group Member	1.0.0	Milton Moore	10/25/16

Contents

1. Introduction....3
1.1 Purpose of this document....3
1.2 Scope of this document....3
1.3 Business Context....3
2. General Description....3
2.1 Product Functions....3
2.2 Similar System Information....3
2.3 User Characteristics....4
2.4 User Problem Statement....4
2.5 General Constraints....4
3. Functional Requirements....4
4. Interface Requirements....19
4.1 User Interfaces....19
4.2 Communications Interfaces....19
5. Performance Requirements....19
6. Other non-functional attributes....20
6.1 Security....20
6.2 Reliability....20
6.3 Maintainability....20
6.4 Reusability....20
7. Operational Scenarios....20
8. Preliminary Use Case Models and Sequence Diagrams....22
8.1 Use Case Model....22
8.2 Sequence Diagrams....23
9. Updated Schedule....24
10. Appendices....25
10.1 Definitions, Acronyms, Abbreviations....25
10.2 References....25

1. Introduction

1.1 Purpose of this document

The purpose of this document is to outline the requirements for the Stage Plan application, which shall be developed by Ready Set Go for our client, Ryan Gross.

1.2 Scope of this document

The requirements elicitation team consists of all members of Ready Set Go: Lane Addison, Ksenia Belikova, Milton Moore, Jason Milloff, and Ben Marshall. The main user of our system shall be our client, Ryan Gross. Stage Plan shall be written in Java, and developed and tested on Windows machines.

1.3 Business Context

Stage Plan is intended to be used by the stage manager at the Bondurant Auditorium, located in Preston Hall at Radford University. Stage Plan shall help the stage manager to create stage plots faster and easier than before, resulting in increased productivity. Stage managers at other locations may find this program useful as well, but are not our target users.

2. General Description

2.1 Product Functions

Stage Plan shall be a desktop application that allows the user to create stage plots. The GUI shall consist of a preview panel, a panel for toggleable flyrail items, and a panel for items which can be dragged and dropped. The program shall allow plots to be saved, loaded, and printed.

2.2 Similar System Information

1. **Stage Plot Designer** - Web based stage plot tool with minimal features

Pros:	**Cons:**
Easy to use	Cannot save or load plots
Free	Does not use an accurate top-down view
Runs in a web browser	

2. **Stage Plot Pro** - Desktop based stage plot tool with advanced features

Pros:	**Cons:**
Many features	Costs $40
Uses accurate top-down layout	User interface is complicated
	Documentation is poor

2. **MyStagePlan** – Web based stage plot tool with pay-per-plan pricing

Pros:	**Cons:**
Decent variety of features	Must pay 1.99 Euros per plan
Uses accurate top-down layout	Must create an account
Runs in a web browser	

2.3 User Characteristics

This program is being developed specifically for Ryan Gross, who has experience working with existing software systems within Windows. However, any stage manager is likely to have similar experience, as the job demands use of software tools.

2.4 User Problem Statement

Our client currently makes his stage plots in Microsoft Paint, as he is not satisfied with the stage plot software which currently exists. Stage Plan shall conform to his needs, allow him to replace Microsoft Paint with something faster and easier to use, and shall be provided for free.

2.5 General Constraints

The user must have Java installed. To print, the user must have a printer connected and set up.

3. Functional Requirements

1. **Minimizing the application.**

1. *Description*
   The user shall be able to minimize the application by clicking the '-' in the top right corner.
2. *Criticality* - 1
3. *Technical issues* - None
4. *Risks* - None
5. *Dependencies with other requirements* - None

2. **The user shall be able to resize the window.**

1. *Description*
   The user shall be able to select the toolbar button that looks like a square or two overlapping squares in order to resize the application windows.
2. *Criticality* - 4
3. *Technical Issues* – None
4. *Risks* - None
5. *Dependencies with other requirements*
   This feature shall be linked to closing the program and minimizing the program. Regardless of the size of the window, the user shall be able to close the program from this state if necessary.

3. Exiting the application.

1. *Description*
   The user shall be able to exit the application by clicking the 'X' in the top right corner.
2. *Criticality* - 1
3. *Technical issues* - None
4. *Risks*
   The user could exit the application by mistake.
5. *Dependencies with other requirements* - None

4. If the user tries to exit the application without saving, the user shall be prompted to save.

1. *Description*
   When the user attempts to close the application after modifying the document, a popup box shall appear asking the user to save the progress.
2. *Criticality* - 5
3. *Technical Issues*
   This feature shall be tied into the saving the project, and if any changes are made, the system shall be able to detect if any changes are being made from the previous save point.
4. *Risks*
   The condition shall not be satisfied if no changes have been made to the project or if the project was saved immediately before exiting the project.
5. *Dependencies with other requirements* - None

5. The user shall be able to access a drop-down list of file options after clicking a "File" tab.

1. *Description*
   When the user clicks the program's File Menu, a dropdown menu shall appear containing options which shall be listed in other requirements.
2. *Criticality* - 5.
3. *Technical Issues*- None
4. *Risks*
   If this feature is absent, a large portion of the program's functionality shall not be accessible. Therefore, the File menu shall be one of the first pieces of the system implemented.
5. *Dependencies with other requirements* - None

6. The user shall be able to load a stage plot from a file into the system through a dialog opened from the file drop-down menu.

1. *Description*
   The user shall be able to load a project from a file into the system through a dialog from the file drop-down menu.
2. *Criticality* - 5
3. *Technical Issues*
   The system shall be reliant on the ability to locate an external file from the system and opening it inside of the program.
4. *Risks*
   The program may be unable to load the file requested. One way to reduce the probability of this issue occurring is to make sure the program shall be able to open the necessary file type.
5. *Dependencies with other requirements* - None

7. Cancel loading stage plot.

1. *Description*
   The user shall be able to select cancel when loading a stage plot.
2. *Criticality* - 3
3. *Technical issues*
   We shall have to determine how long load a stage plot may take.
4. *Risks*

The user might cancel the load by accident.

5. *Dependencies with other requirements*
   The loading of a stage plot must be initiated.

8. The user shall be able to save the current stage plot into a file through a dialog opened from the file drop-down menu or from a macro (ctrl-s).

1. *Description*
   The user shall be able to save the progress of the ongoing project through a drop down menu or by pressing "Ctrl-S".
2. *Criticality* - 5
3. *Technical Issues*
   The program by default needs to be able to save a program.
4. *Risks*
   The system may not save the program as the correct file type.
5. *Dependencies with other requirements* – See #5

9. The user shall be able to save the file without a dialog if it has been saved in a dialog previously through the file drop-down menu or from a macro (ctrl-s).

1. *Description*
   If the current stage plan has been saved before, the user should be able to press "Ctrl-S" to save the stage plan again.
2. *Criticality* - 1
3. *Technical Issues* - None
4. *Risks*
   The user could press "Ctrl-S" by accident, saving when they do not want to.
5. *Dependencies with other requirements* – See #5

10. The user shall be able to print the stage plot from the file drop-down menu off of a printer selected via a dialog, or through a macro (ctrl-p).

1. *Description*
   The user shall be able to print a completed or incomplete project with a selection from the menu or a combination of keystrokes.
2. *Criticality* - 5
3. *Technical Issues* - None
4. *Risks*
   The user may print out the plan accidentally.
5. *Dependencies with other requirements* – See #5

11. The user shall be able to export the stage plot to a .pdf file from the file drop-down menu.

1. *Description*
   The user shall be able to save the Stage plot as a .pdf file, for later use.
2. *Criticality* - 4
3. *Technical Issues* - None
4. *Risks*
   The user could try exporting with an invalid name/file extension.
5. *Dependencies with other requirements* - #5

12. The user shall be able to access a drop-down list of tool options after clicking a "Tools" tab.

1. *Description*
   The user shall be able to choose tools from the drop-down list when clicked on “Tools” tab.
2. *Criticality* - 4
3. *Technical Issues* - None
4. *Risks*
   The user may select “Tools” tab accidentally.
5. *Dependencies with other requirements* - None

13. The user shall be able to “cut” a selected Stage Object from the tools drop-down list, or from a macro (ctrl-x).

1. *Description*
   If an object is selected and “cut”, it shall be deleted from its location and stored in the clipboard.
2. *Criticality* - 3
3. *Technical Issues*
   The user can only store one object in the clipboard.
4. *Risks*
   The user could accidentally overwrite what is currently in the clipboard.
5. *Dependencies with other requirements* – See #12

14. The user shall be able to “copy” a selected Stage Object from the tools drop-down list, or from a macro (ctrl-c).

1. *Description*
   If an object is selected and “copied,” then the object shall stay on the plan, but shall also be stored in the clipboard.
2. *Criticality* - 3
3. *Technical Issues*
   The user can only store one object on the clipboard.
4. *Risks*
   The user could overwrite what is stored on the clipboard.
5. *Dependencies with other requirements* - See #12

15. The user shall be able to paste a “cut” or “copied” object to the Stage from the tools drop-down list, or from a macro (ctrl-v).

1. *Description*
   If an object is stored in the clipboard, it shall be pasted onto the Stage view in the middle of the stage, or where the cursor is if the macro is used.
2. *Criticality* - 3
3. *Technical Issues*
   The user could try pasting to an invalid location.
4. *Risks*
   The user might not have wanted to paste
5. *Dependencies with other requirements* – See #12

16. The user shall be able to undo an action from the tools drop-down list, or from a macro (ctrl-z).

1. *Description*
   If the user applies a change that needs to be reverted, the user can select to undo the action with a drop down menu or combination of keystrokes.

2. *Criticality* - 3
3. *Technical Issues*
   The user may not have anything to undo.
4. *Risks*
   The user might have not wanted to undo a specific action.
5. *Dependencies with other requirements* – See #12

17. The user shall be able to redo an undo'd action from the tools drop-down list, or from a macro (ctrl-y).

1. *Description*
   If the user accidentally clicks undo, they can "undo" the undo.
2. *Criticality* - 2
3. *Technical Issues*
   They should not be able to redo after making a new change, so when they make a change, the redo queue needs to be cleared.
4. *Risks*
   The user could accidentally make a change, and therefore be unable to redo.
5. *Dependencies with other requirements* – See #12, #16

18. Zoom In.

1. *Description*
   The user shall be able to zoom in from the tools drop-down list.
2. *Criticality* - 1
3. *Technical issues*
   Maximum zoom in distance shall have to be determined.
4. *Risks* - None
5. *Dependencies with other requirements*
   The zoom in function works in conjunction with the zoom out function.

19. Zoom Out.

1. *Description*
   The user shall be able to zoom out from the tool drop-down list.
2. *Criticality* - 1
3. *Technical issues*
   Maximum zoom out distance shall have to be determined.
4. *Risks* - None
5. *Dependencies with other requirements*
   The zoom in function works in conjunction with the zoom out function.

20. The user shall be able to rotate a selected Stage Object from the tools drop-down list, or from a macro (ctrl-r).

1. *Description*
   The object can be rotated up to 360 degrees on the Stage plot.
2. *Criticality* - 4
3. *Technical Issues*
   The rotation should be limited to 360, to avoid overflow if they happen to rotate a lot.
4. *Risks*
   The user could rotate on accident, and need to get back to previous rotation.
5. *Dependencies with other requirements* - See #12, #24

21. The user shall be able to cancel a rotation by clicking "Esc".

1. *Description*
   If the user is in the process of rotating a drag and drop prop, they shall be able to cancel the process by pressing the escape key, leaving the prop unchanged.
2. *Criticality* - 4
3. *Technical Issues*
   There should be another way to cancel, in case the user's escape key is broken.
4. *Risks*
   The user could accidentally press the escape key and cancel their rotation.
5. *Dependencies with other requirements* - See #20

22. The user shall be notified by the system to select an object, if they attempt to rotate without selecting an object first.

1. *Description*
   If the user has not selected a specific object and tries to rotate the object, the system shall return a message saying that an item needs to be selected first.
2. *Criticality* - 4
3. *Technical Issues*
   An attempted rotate of a non-existent object is impossible.
4. *Risks*
   No message shall appear if an object from the toolbar is already selected.
5. *Dependencies with other requirements* – See #20, #21

23. Adding a Text-Box.

1. *Description*
   The user shall be able to add a text box from the tool drop-down list.
2. *Criticality* - 3
3. *Technical issues*
   The max size and max character input shall have to be determined to conserve space.
4. *Risks*
   If the text box fails to work, written input for the user shall not be available.
5. *Dependencies with other requirements*
   The text box feature is tied in with the drop down menu.

24. Object Selection.

1. *Description*
   The user shall be able to select an object from the right panel by left clicking.
2. *Criticality* - 4
3. *Technical issues*
   The objects available shall have to include everything that the user needs.
4. *Risks*
   If the drop down menu fails, the user would not have any objects to place on the stage.
5. *Dependencies with other requirements*
   The object selection is directly dependent on the right panel.

25. If the user drags an object from the right panel to the Stage view, it shall be scaled to the right dimensions.

1. *Description*
   When an object is dragged onto the view, it should show up as the right size, even though it hasn't been placed yet.
2. *Criticality* - 4
3. *Technical Issues*
   The scale should change depending on the zoom level of the stage.
4. *Risks*
   The user could be dragging the wrong object over.
5. *Dependencies with other requirements* - See #24

26. Drag and Drop.

1. *Description*
   The user shall be able to place an object in the Stage panel by dragging and dropping.
2. *Criticality* - 5
3. *Technical issues*
   The drag and drop function shall need to be flawless to ensure the full functional capabilities of the application.
4. *Risks*
   If the drag and drop function does not work properly, the user shall not be able to place items onto the stage area.
5. *Dependencies with other requirements* – See #24

27. If the user drags an object back from the Stage view to the right-panel, it shall be un-scaled.

1. *Description*
   If the user drags an object from the stage into the right panel, the scale of the image should change from the stage scale to the right-panel scale until the user releases the mouse button.
2. *Criticality* - 1
3. *Technical Issues*
   The change of scale should depend upon the zoom level of the stage.
4. *Risks*
   If this feature is not complete, the user experience shall be worsened, but nothing crucial should break.
5. *Dependencies with other requirements* - See #25

28. If the user drags an object to an invalid location, the cursor shall be replaced with an "X".

1. *Description*
   System should replace cursor with an "X" if user drags object from the right panel bypassing stage to the left panel.
2. *Criticality* - 1
3. *Technical Issues*
   Dragged object would be scaled back to the normal size.
4. *Risks*
   The user could accidentally drag an object to the invalid location, so we need to ensure that they would know if it is invalid.
5. *Dependencies with other requirements* - See #24, 25

29. If the user drags an object back to a valid location, the cursor shall be set back to normal.

1. *Description*
   After an object is dragged and dropped to a valid location, the cursor shall reset to the default cursor.
2. *Criticality* - 1
3. *Technical Issues* - None
4. *Risks*
   The user may incorrectly place the object.
5. *Dependencies with other requirements* – See #24, #25

30. Invalid Location.

1. *Description*
   If the user drops an object on an invalid location, the object shall not be placed, and shall be deselected.
2. *Criticality* - 4
3. *Technical issues*
   Invalid locations must involve all areas not within the staging area.
4. *Risks*
   The user accidentally clips an invalid area with the object they are placing, resulting in the object not being placed.
5. *Dependencies with other requirements*
   The object selection is directly dependent on the right panel.

31. The user shall be able to deselect an object by clicking on the right panel, without clicking an object.

1. *Description*
   If the user clicks off of the object, then it shall be deselected.
2. *Criticality* - 4
3. *Technical Issues*
   There needs to be a space not occupied by objects for the user to click on
4. *Risks*
   The user could lose track of what they were altering by deselecting it
5. *Dependencies with other requirements* – See #24

32. The user shall be able to choose from a menu of options by right-clicking on an object in the right panel.

1. *Description*
   When the user right clicks an object in the right panel, a drop down menu shall appear with various different options.
2. *Criticality* - 4
3. *Technical Issues*
   The dropdown menu could appear to be cut off by the screen
4. *Risks*
   The menu may be hidden by a toolbar or bottom of the window
5. *Dependencies with other requirements* – See #24

33. The user shall be able to view the height, width, and image of an object by clicking "properties" from the menu of options.

1. *Description*

By selecting “properties”, a dialog shall appear stating various properties of the object.
2. *Criticality* - 2
3. *Technical Issues* - None
4. *Risks* - None
5. *Dependencies with other requirements* – See #32

34. The user shall be able to create a new object from a dialog that starts with the same characteristics as the object selected by clicking “clone” from the menu of options.

1. *Description*
   If the user has selected a drag and drop prop, they shall be able to use a clone dialog to create an identical copy of that prop.
2. *Criticality* - 2
3. *Technical Issues*
   We must be sure that we make a proper copy rather than copying the reference of the original.
4. *Risks*
   If we do not properly implement this feature, the user experience shall be worsened, but the core functionality should remain intact.
5. *Dependencies with other requirements* – See #32

35. The user shall be able to remove the object from the list of available objects by selecting “delete” from the menu of options.

1. *Description*
   The user can delete an object from the list of available objects. (**PERMANENTLY**)
2. *Criticality* - 3
3. *Technical Issues*
   The object needs to be deleted both from immediate GUI and long-term storage.
4. *Risks*
   The object could be deleted accidentally.
5. *Dependencies with other requirements* – See #32

36. The user shall be prompted for confirmation before permanently deleting an object from the list of available objects.

1. *Description*
   Upon selecting to delete an object from the right panel, a pop-up message shall appear to notify the user of the change that’s about to be made.
2. *Criticality* - 4
3. *Technical Issues*
   The object shall be deleted from the local system and the program system’s memory.
4. *Risks*
   The user cancels the deleting process.
5. *Dependencies with other requirements* - See #35

37. The user shall be able to see a menu of options by right-clicking on the right panel.

1. *Description*
   A menu of options should appear when user right-clicks on the right panel.
2. *Criticality* - 4
3. *Technical Issues*
   Menu appears in the different location.
4. *Risks*
   User accidentally right-clicks.
5. *Dependencies with other requirements* - None

38. The user shall be able to create a new Stage Object by clicking "New Object" from the menu of options.

1. *Description*
   The user shall be able to create a new object by filling out a dialog, this object shall be added to the list, and also long-term storage.
2. *Criticality* - 4
3. *Technical Issues*
   The user may want to add a picture, or an unusual shape
4. *Risks*
   The user may not have actually wanted to make a new object
5. *Dependencies with other requirements* – See #37

39. The user shall not be able to submit a new object without filling out the dialog completely.

1. *Description*
   The user shall not be able to press the Save button when adding a new stage prop, until all forms are filled out.
2. *Criticality* - 4
3. *Technical Issues*
   We must be sure to notify the user why they cannot submit/save, or else they may be confused.
4. *Risks*
   If we do not prevent the user from saving an incomplete prop, then the incomplete data may cause unexpected issues while using the prop.
5. *Dependencies with other requirements* – None

40. The user shall be able to scroll through the list of objects in the right panel.

1. *Description*
   System should have a scrollbar to allow user to scroll through the list of the objects in the right panel.
2. *Criticality* - 4
3. *Technical Issues*
   Scrolling doesn't update the list of objects.
4. *Risks*
   User doesn't have a mouse with the scroll wheel.
5. *Dependencies with other requirements* - None

41. The user shall be able to pan the Stage view by dragging the view.

1. *Description*
   The user can pan left and right on the view by clicking and dragging.
2. *Criticality* - 4

3. *Technical Issues*
   The stage should only be pan-able to an extent.
4. *Risks*
   The user could pan too far and lose track of the stage plot.
5. *Dependencies with other requirements* - None

42. Deselect Object.

1. *Description*
   The user shall be able to deselect an object by left-clicking the stage view.
2. *Criticality* - 3
3. *Technical issues*
   The user shall need space to click on the stage view for deselection.
4. *Risks*
   The user might accidentally deselect an object when click on the stage view.
5. *Dependencies with other requirements*
   This requirement is dependent on object selection requirement.

43. View Menu Options.

1. *Description*
   The user shall be able to view a menu of option by right clicking the stage view.
2. *Criticality* - 4
3. *Technical issues*
   When right clicking the stage view the request must ignore any objects that may be in the stage view area.
4. *Risks*
   The user might select an object by left clicking instead of right clicking when attempting to access the menu options.
5. *Dependencies with other requirements* - None

44. The user shall be able to paste an object from the clipboard by selecting "paste" from the menu of options.

1. *Description*
   The user shall be paste the object from the clipboard onto the stage, without removing the object from the clipboard memory
2. *Criticality* - 4
3. *Technical Issues*
   The stage could run out of room for objects to be placed
4. *Risks*
   The user could paste again on accident, not realizing it is still in the clipboard
5. *Dependencies with other requirements* – See #14

45. The user shall be able to zoom in on the Stage view by selecting "Zoom in" from the menu of options.

1. *Description*
   The user shall be able to enlarge the view of the project.
2. *Criticality* - 4
3. *Technical Issues*
   The view of the object may be zoomed all the way in.

4. *Risks*
   The user might zoom out by mistake.
5. *Dependencies with other requirements* - None

46. The user shall be able to zoom out on the Stage view by selecting "Zoom out" from the menu of options.

1. *Description*
   The user can shrink the view of the stage plot.
2. *Criticality* - 3
3. *Technical Issues*
   The stage should only be able to be zoomed out a certain amount.
4. *Risks*
   The user could lose track of the stage plot by zooming out too far.
5. *Dependencies with other requirements* - None

47. The user shall be able to select an object on the Stage by left-clicking it in the Stage view.

1. *Description*
   The user can have an object selected, ready for an action by single-clicking it.
2. *Criticality* - 5
3. *Technical Issues* - None
4. *Risks*
   The user loses track of the object they previously had selected by selecting another one.
5. *Dependencies with other requirements* – See #26

48. The user shall be able to change an object's location on the Stage by dragging it in the Stage view.

1. *Description*
   The user shall be able to select an object on the stage and move it to a new location
2. *Criticality* - 5
3. *Technical Issues*
   The object may be moved to an illegal location on the design.
4. *Risks*
   The user may make an accidental position move of an object
5. *Dependencies with other requirements* – See #47

49. Object Removal.

1. *Description*
   The user shall be able to remove an object from the Stage by dragging and dropping it to an invalid location.
2. *Criticality* - 4
3. *Technical issues*
   The object removal area shall have to include all areas except the stage view.
4. *Risks*
   The user might remove the wrong object.
5. *Dependencies with other requirements – See* #30

50. If the user drags an object from the Stage to an invalid location, the cursor shall be replaced with an "X".

1. *Description*
   System should replace cursor with an "X" if user drags object from the stage to the left panel or to non-drop-off location.
2. *Criticality* - 1
3. *Technical Issues*
   Dragged object would be scaled back to the normal size.
4. *Risks*
   User accidentally drags object.
5. *Dependencies with other requirements* – See #24, 25

51. If the user drags an object from the Stage to an invalid location, and then back to a valid location, the cursor shall be resumed to normal.

1. *Description*
   The cursor should stay consistent as to what is invalid and what is valid placement.
2. *Criticality* - 2
3. *Technical Issues*
   The system needs to keep track of valid and invalid locations through a variety of situations, so the cursor is never wrong.
4. *Risks*
   The user could not notice that the cursor is telling them the location is invalid.
5. *Dependencies with other requirements* – See #48, #49, #50

52. If the user double-clicks on a text box on the stage, they shall be able to edit the text.

1. *Description*
   When the user double-clicks a text box, the text box should switch into an edit mode, where the text can be edited.
2. *Criticality* - 5
3. *Technical Issues*
   We must be sure to include a visual indicator that the text can be edited or the user could be confused.
4. *Risks*
   If we do not include this feature, our program shall not be fully functional.
5. *Dependencies with other requirements* – See #23

53. The user shall be able to stop editing the text in a text box by clicking outside of it.

1. *Description*
   When they click outside the text box, the editing mode should end.
2. *Criticality* - 3
3. *Technical Issues*
   The system needs to also save the text when the editing is done.
4. *Risks*
   The user may have not wanted to stop editing.
5. *Dependencies with other requirements* – See #52

54. The user shall be able to "cut" an object from the Stage by right clicking it and selecting "cut".

1. *Description*
   The user shall right click on the object, and select to 'cut' from the stage.
2. *Criticality* - 4
   Technical Issues
   System cuts the wrong object.
3. *Risks*
   The user may accidentally cut an object off of the stage.
4. *Dependencies with other requirements* – See #13

55. The user shall be able to "copy" an object from the Stage by right clicking it and selecting "copy".

1. *Description*
   System should allow user to copy an object from the Stage by clicking it and selecting "copy" option.
2. *Criticality* - 4
3. *Technical Issues*
   System copies the wrong object.
4. *Risks*
   User copies object by the accident
5. *Dependencies with other requirements* – See #32

56. The user shall be able to remove an object from the Stage by right clicking it and selecting "delete".

1. *Description*
   To delete an object from the plot, they can right click it and select it.
2. *Criticality* - 5
3. *Technical Issues*
   The object needs to be deleted just from the plot, not clipboard memory or anywhere else.
4. *Risks*
   The user may delete the wrong object
5. *Dependencies with other requirements* - None

57. Object Information.

1. *Description*
   The user shall be able to see information about an object from the Stage by right clicking it and selecting "properties".
2. *Criticality* - 3
3. *Technical issues*
   Object information needs to be accurate and correct.
4. *Risks*
   The user may left-click instead of right clicking the object resulting in the object being picked up.
5. *Dependencies with other requirements* - None

58. Mouse Wheel Zoom In.

1. *Description*

The user shall be able to zoom in on the stage view by scrolling up on the mouse.

2. *Criticality* - 2
3. *Technical issues*
   Create a macro to work with the wheel.
4. *Risks*
   Without the macro the user would have accidental cases of zooming in and out on the stage area.
5. *Dependencies with other requirements* - See #59

59. Mouse Wheel Zoom Out.

1. *Description*
   The user shall be able to zoom in on the stage view by scrolling down on the mouse.
2. *Criticality* - 2
3. *Technical issues*
   Create a macro to work with the wheel.
4. *Risks*
   Without the macro the user would have accidental cases of zooming in and out on the stage area.
5. *Dependencies with other requirements* – See #58

60. The user shall be able to fly a rail in by selecting "Fly in" next to that rail in the left panel.

1. *Description*
   If a flyrail item is flown out, the user shall be able to press a 'Fly in' button to bring it onto the stage
2. *Criticality* - 5
3. *Technical Issues*
   This button must be very obvious, as it is a core feature that shall be often used.
4. *Risks*
   If we do not include this feature our program would not be functional.
5. *Dependencies with other requirements* - None

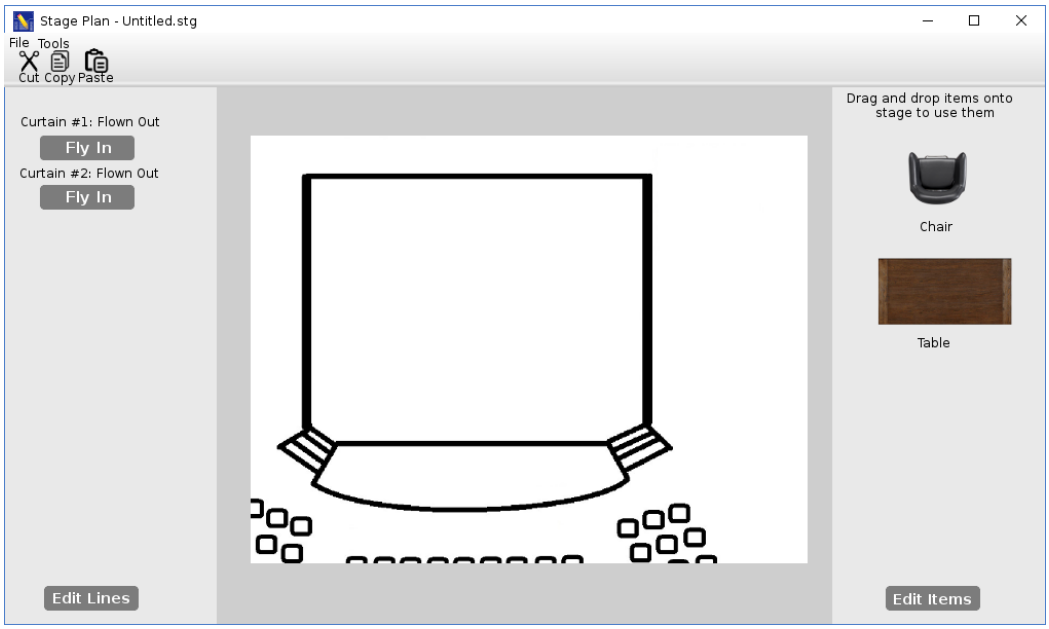
61. The user shall be able to fly a rail out by selecting "Fly out" next to that rail in the left panel.

1. *Description*
   If a flyrail is already flown in, there shall be a button to have it fly back out.
2. *Criticality* - 5
3. *Technical Issues*
   The button must have the right state, it should say fly in or fly out, not the wrong one, or it would get confusing.
4. *Risks*
   The user needs this function to create full stage plans, the function must work for all occasions.
5. *Dependencies with other requirements* - None

4. Interface Requirements

4.1 User Interfaces

- **4.1.1 GUI**



The user interface shall always consist of one window with three panels. The center panel shall be a preview of the stage. The left panel allows the user to toggle items on the flyrail. The right panel allows the user to drag and drop items onto the stage. Cut, copy, and paste commands shall be available in the toolbar.

- **4.1.2 CLI**
  None

4.2 Communications Interfaces

For printing, a printer must be connected and its relevant driver software installed.

5. Performance Requirements

• Response Time – Almost all user-interactions should have program response times of less than 150ms. The exceptions are saving, loading, and printing. These operations should provide a visual indicator of progress for the user.

• Workload – The system should be able to support a few hundred added props without a performance decline.
• Scalability – See Workload requirements above.
• Platform – The system should be developed to run on the latest version of Windows, with the latest Java distribution.

6. Other non-functional attributes

6.1 Security

Being a desktop application and a stage-specialized editor, no security measures need to be taken.

6.2 Reliability

Stage Plan should not use deprecated libraries or language features, to minimize risks of bugs due to a later distribution of Java.

6.3 Maintainability

The flyrail system, drag and drop items, and stage background shall be user customizable, so that the user shall not need to alter the software itself to reflect any upgrades/additions to the stage.

6.4 Reusability

Our code for exporting to images and printing in particular should lend well to being ported to other projects.

7. Operational Scenarios

Scenario 1: User creates and prints a plot.
Description: User creates and prints a stage plot, exiting without saving.
Event Flow:

1. User opens Stage Plan
2. User selects their desired flyrail items from the left panel.
3. User drags and drops their desired items from the right panel into the center panel.
4. User navigates to the File Menu -> Print
5. User selects their printer in the Print Dialog.
6. User presses the Print Button
7. Printer prints their plan.
8. User retrieves their printout.
9. User clicks the red X in the upper right hand corner of the application.
10. A dialog appears asking if the user wishes to save their work.
11. User selects ‘No.’
12. Program terminates

Possible Errors:

- User does not have a printer installed.

Concurrent Activities:

- None

Finished State:

- Stage Plan is terminated.
- User has printout of their plan.

Scenario 2: User creates and saves a plot.
Description: User creates a stage plot, saves it, and exits.
Event Flow:

1. User opens Stage Plan.
2. User selects their desired flyrail items from the left panel.
3. User drags and drops stage props from the right panel.
4. User goes to File Menu -> Save.
5. User selects their desired save location.
6. User clicks the save button.
7. User clicks the red X to close the application.
8. Programs terminates.

Possible Errors:

- When the user attempts to save their stage plan, the user declines instead of accepting the save.

Concurrent Activities:

- None

Finished State:

- Stage Plan is terminated.
- User has a saved file containing their plan.

Scenario 3: The user loads and prints a selected project.
Description: User loads a stage plot, prints it, and exits.
Event Flow:

1. User opens Stage Plan.
2. User navigates to File -> Open.
3. In the Open File Dialog, the user selects an .stg file.
4. In the Open File Dialog, the user clicks ‘Open.’
5. User navigates to the File Menu -> Print.
6. User selects their printer in the Print Dialog.
7. User presses the Print Button.
8. Printer prints their plan.
9. User retrieves their printout.
10. User clicks the red X in the upper right hand corner of the application.
11. Program terminates.

Possible Errors:

- User does not have a printer installed.

Concurrent Activities:

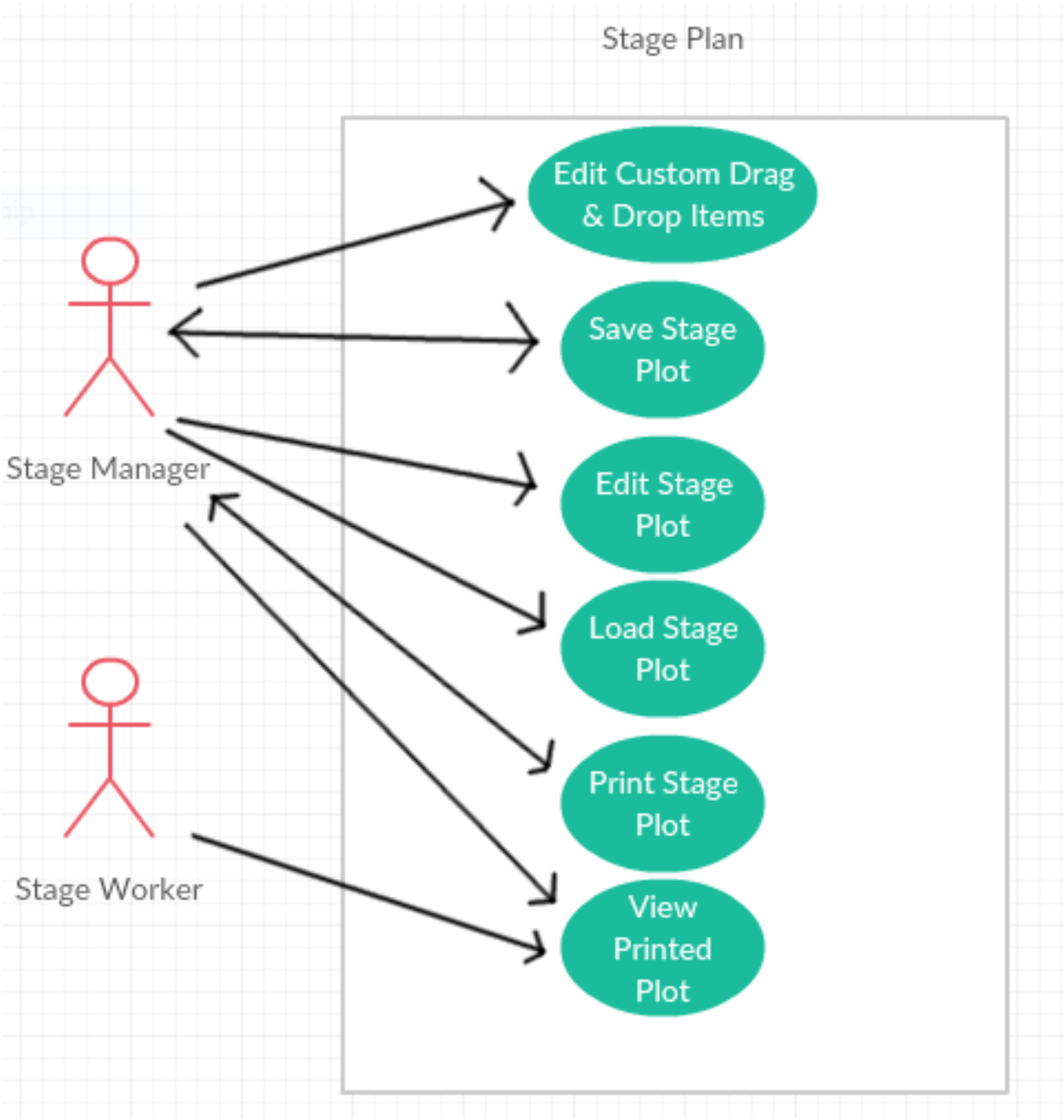
- None

Finished State:

- Stage Plan is terminated.
- User has printout of their plan.

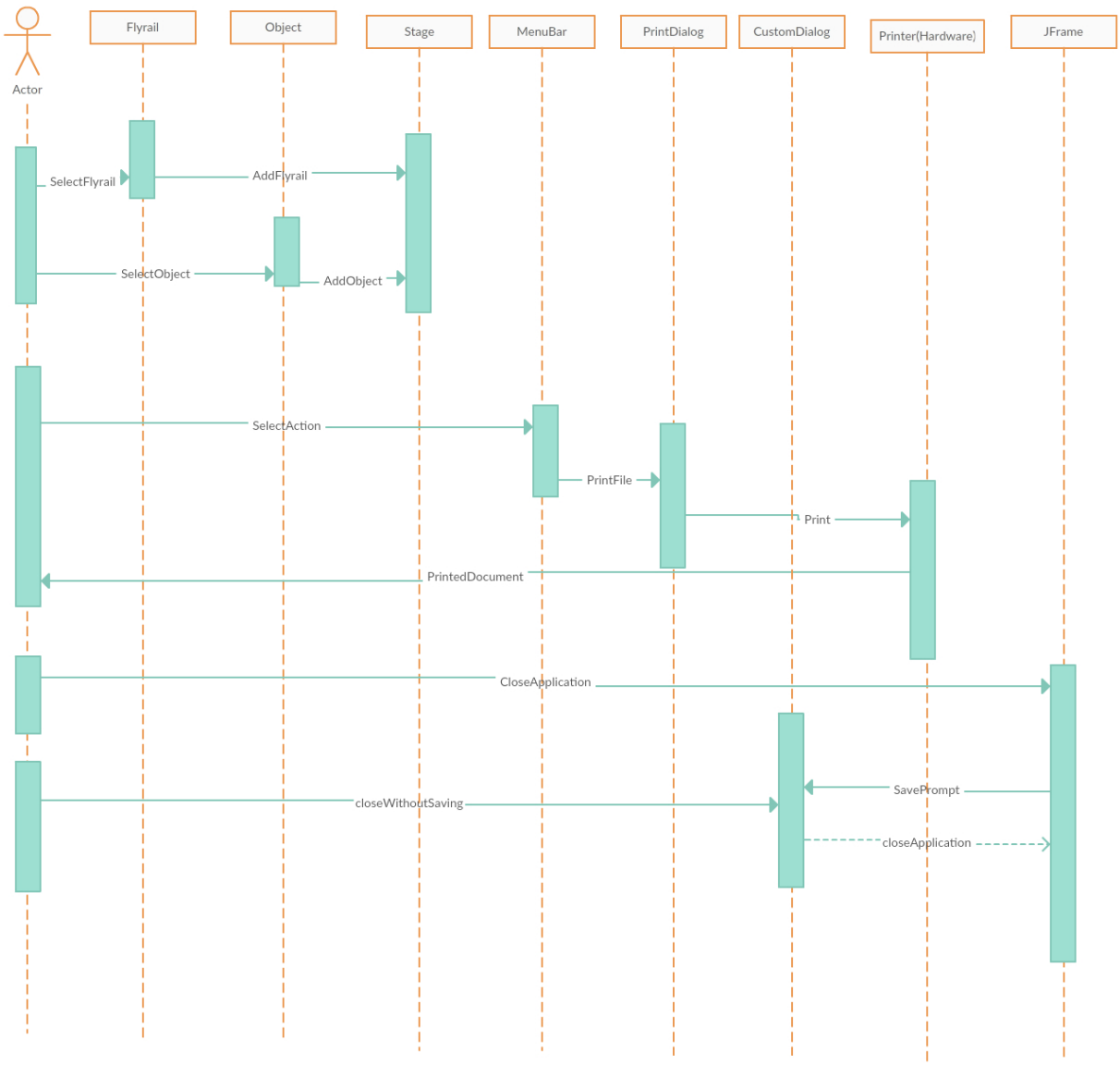
8. Preliminary Use Case Models and Sequence Diagrams

8.1 Use Case Model

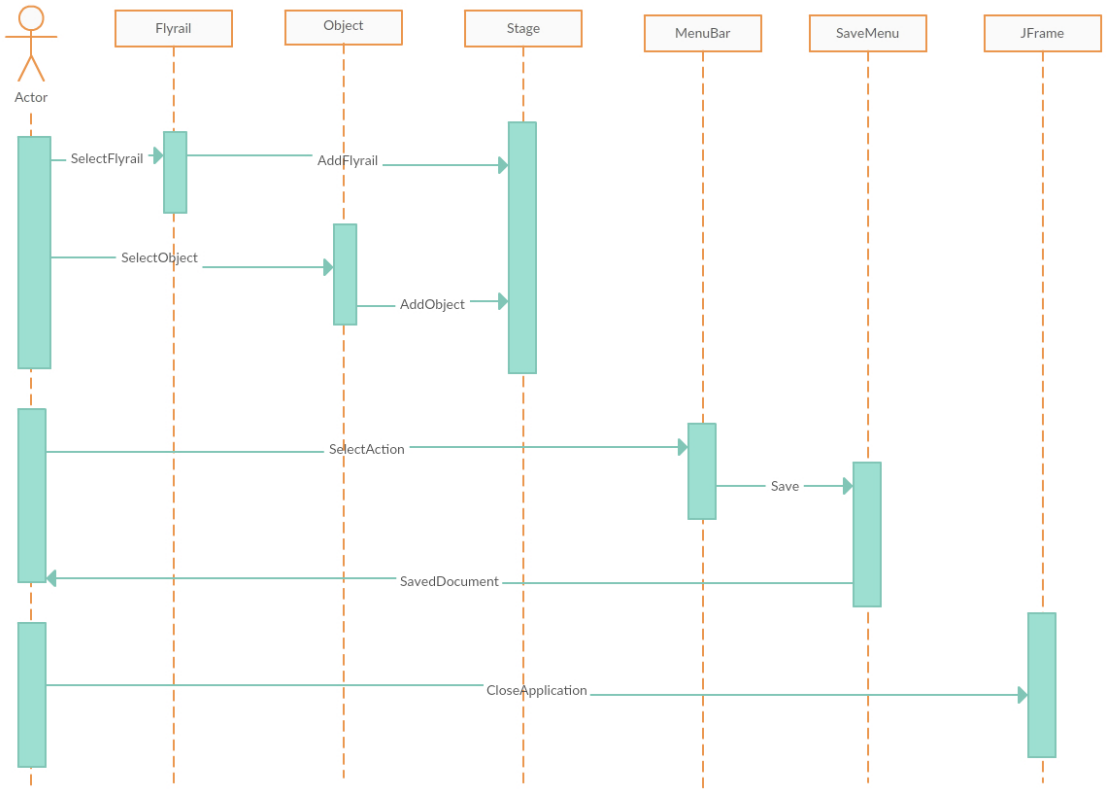


8.2 Sequence Diagrams

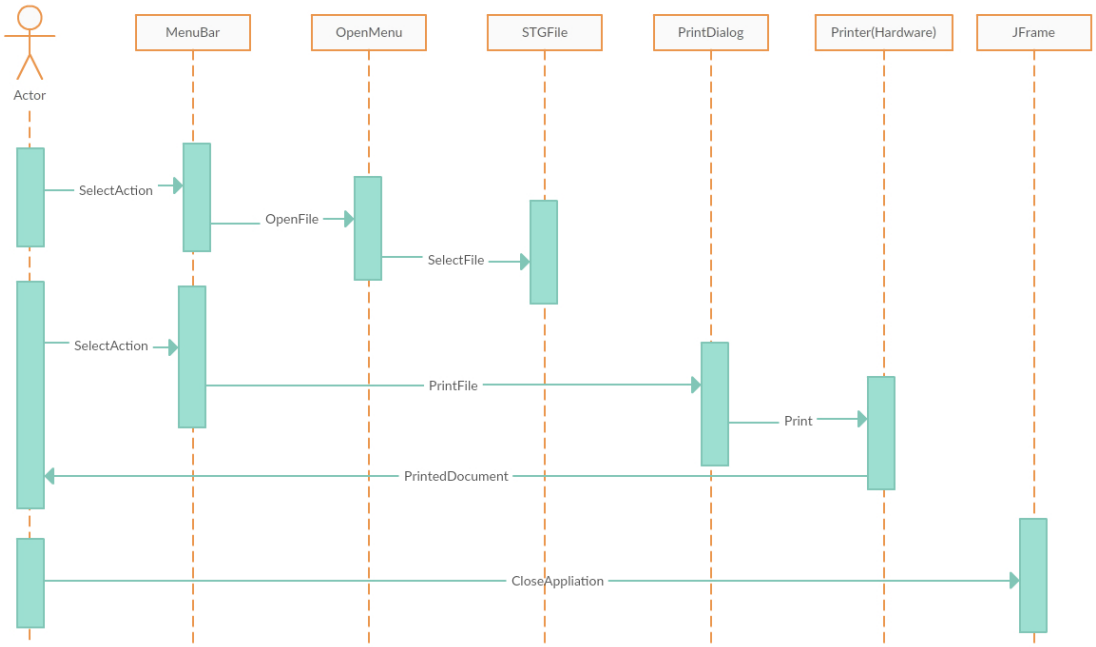
Scenario 1: User creates and prints a plot.



Scenario 2: User creates and saves a plot.



Scenario 3: The user loads and prints a selected project.



9. Updated Schedule

See attached Gantter document.

10. Appendices

10.1 Definitions, Acronyms, Abbreviations

Flyrail – Stage system for raising and lowering things on and off the stage

Flown in – Refers to a thing on the flyrail system being present on the stage

Flown out – Refers to a thing on the flyrail system not being present on the stage

D&D – Drag and Drop aka Stage Object. Refers to items from the right panel which can be positioned on the stage.

PNG – Portable Network Graphics image format

PDF – Portable Document Format. A document format.

JPEG – Joint Photographic Experts Group image format

GUI – Graphical user interface

PM – Project Manager

AM – Assistant Project Manager

10.2 References

Stage plot designer - http://www.bosstweedbackline.com/stage-designer/

Stage Plot Pro - http://www.stageplot.com/

MyStagePlan - www.mystageplan.com