Senior Software Engineering Project
Small Shell: Vision & Scope

Ryan Stachura, Ezra Taylor, Haoxiang Dai

Table of Contents

1 Background 3
2 Vision Statement 3
3 Success Measures 3
4 Prioritized Project Constraints 3
5 Stakeholders 3
6 Risk 4
7 Scope 4

1 Background

The programming language C has been around since 1972 and is used across the Earth to date. Understanding one of the most fundamental languages inside and out can only help. On top of this, Unix based systems are also an extremely important part of today's society. Therefore, we will be combining them in such a way that we can show our understanding of each.

2 Vision Statement

This is a very time sensitive project with specific goals. Due to the time constraints, we will not be adding extra features unless all goals have been met. There is no extra expense needed nor any additional constraints that could hold us back. Each member of the team will do their job in order to complete this project on time.

3 Success Measures

We will be measuring our success on not only time spent each week on the project but also goals achieved in a timely manner. Each person on the team will be assigned their own tasks in order for us as a team to complete a goal. Each team member will be measured by how much time they spent in one week specifically on writing code. These time measurements should come out to around at least eight hours each week. With these measurements of everyone on the team and the complete progress of the project, we can use this information to confirm that we will be able to reach our final goal on time.

4 Prioritized Project Constraints

Our main constraint with this project is time, since we are completing a 3-month project. Because of this, we will prioritize the essential features of our project such as changing directories, using foreground and background processes, interrupt signal processing, input/output redirection, etc. In order to release by the end of this term, our resources have been limited to C and Linux-based systems/servers.

5 Stakeholders

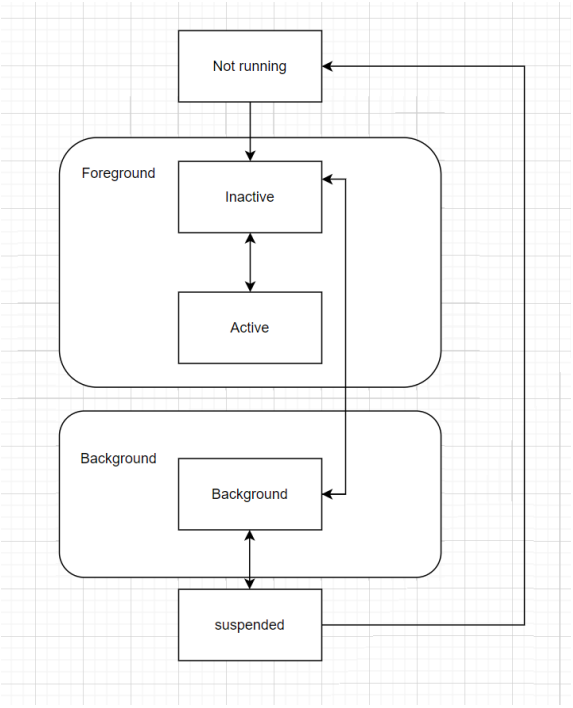
- **Operating system learner**
  This project provides the most basic features of an operating system. The os learner can easily pick up those features and designs.

6 Risk

Risk	Likelihood	Impact	Mitigation Strategy	Early Detection	Consequence
All the paperwork and documentation needed for the project not getting done. Other teams have had three terms to complete, as we only have one.	Unlikely	Mediu m	To mitigate this, we have a plan of action to complete this project as fast as possible. When this project is complete in this way, it will give us some extra time in the end to complete all necessary documentation needed for project and expo demonstration.	Weekly plan updates result in an iteration plan that goes beyond the deadline.	Should the mitigation strategy fail to prevent/avoid the risk, then our project will not look as nice without documentation

7 Scope

1.1 Process Flows



1.2 User Stories (Epics and Features)

As a user, I want the system to have a cd command that allows me to change current position to anywhere I want.

As a user, I want to be able check the status of my current programs by using a status command.

As a user, I want the system to have the most basic commands: ls, pwd, kill.

As a user, I want to be able to have a foreground only mode.

As a user, I want to be able to modify the command.

As a user, I want to be able to see a different color of text in my system.

1.2.1 Epic (e.g. Manage People)

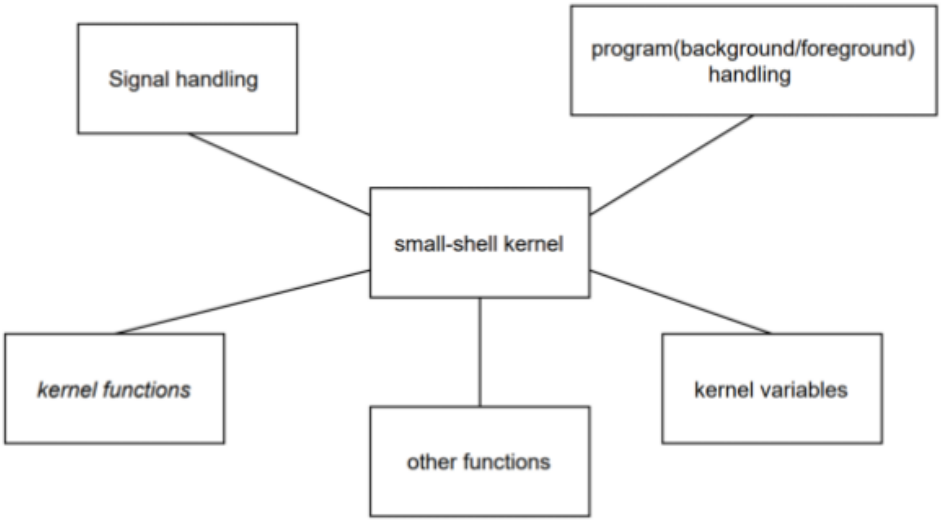
As an operating system learner, I want this system to clearly show the information with a good format.
As a designer, I want to implement the features by using least API
As a programmer, I want all of the code be well-commented
As a programmer, I want the code can be run on multiplatform

1.3 Iteration Plan and Estimate

Each week we will be having a sprint meeting to talk about what needs to be done this week and if every has put in their time from the previous week. Each person has their own goal for the end result of the project. As long as everyone puts in their time each week, there should be no problem finishing this project in time for the expo.

1.4 Solution Architecture

C was chosen as the language to implement this project in because of how unrestricted it is. Unlike other common languages, C will do just about anything you tell it to do. Of course this can be a bad thing too, but part of this project is to show our understanding of C -- to show what we have learned. Linux is the operating system of choice because of how well it interacts with C. On top of this, one of the goals is to create entire processes, whether it be in the foreground or background. C provides us with the *fork()* function which allows us to manage these processes systematically.


Signal handling
program(background/foreground) handling
small-shell kernel
kernel functions
other functions
kernel variables