University of Mary Washington

Mice Visualization Project

Project Plan Document

Cole Hodges, Evan May, Nate Nuval

CPSC-430

Dr. Karen Anewalt

3/1/17

Introduction

Purpose

This document provides a detailed description of the Mice Visualization software project explaining:

- Whom the software is for
- Why they need the software
- What functionality must the software perform
- How the project team plans to implement these requirements
- The estimated time required to complete this project

The intended audience for this project plan document is Dr. Polack (the client), Dr. Anewalt (the project manager), and the project team. After reading this document, one should have a clear understanding of what the project is, the project team's schedule and timeline, and what the client can expect of the software by May 2017.

Scope

The Biology department and the Psychology department of the University of Mary Washington conducts various studies on mice. They have requested a cross-platform application that allows them to visually analyze the data they have collected during their studies. A simple, intuitive Graphical User Interface (GUI) is necessary for this project; some users may not have advanced computer skills. The software must be a desktop application developed in Java and must be completed before the end of the Spring 2017 semester (May, 2017). Refer to the "Project Scope" section for more details.

References

Dr. Polack is the primary source for the project requirements. She is working on behalf of Dr. Parish Waters, the individual leading the studies on mice behavior. *Visualizing Mice Behavior With Heat and Vector Map Generation* by Kevin Hartnett, Colton Hall, Parker Rowland and *Mice Visualization* by Alec Carlyle, Michael Prime, Mikaela Goldrich are the two software requirements documents used, in accordance with Dr. Polack, to prepare this project plan document.

Overview of The Remainder of The Document

The rest of this document is divided into three main sections:

- Project Description
- Project Schedule
- Appendix

The Project Description section provides information on the scope of the project, the users of the software, the requirements essential to the functionality of the software, and the project's hardware and software constraints.

The Project Schedule section provides information on how we plan to approach constructing the software. A detailed breakdown of our project timeline is provided in this section. This segment of the document also includes all of the tasks and subtasks needed to complete the project, diagrams and charts displaying the estimated time it will take to complete each task and which tasks are dependent on each other, and the specific milestones and deliverables for the project.

The Appendix section provides a glossary defining any words that may be unfamiliar or uncommon. It also provides information about which specific team member contributed to each specific section, as well as explanations of all the diagrams and figures used in this document.

Project Description

Project Scope

The Mice Visualization software will be used by the UMW Biology department and the UMW Psychology department to create heat maps and vector maps based on data describing movement patterns. Currently, they are collecting data from mice. Each mouse is given a unique RFID. The floor of the mouse cage consists of pads in a grid formation. This allows the researchers to determine where a specific mouse is located and how long it stays in that location. These data elements are stored in a CSV file. The researcher can import their data to the Mice Visualization software. The system will provide the user with options for which data they want to use (which mice and for what period of time), and how they want the data to be represented (as a heat map or a vector map).

Based on the two requirements documents and the information given by Dr. Polack, there are eight functional requirements:

- Load New Dataset
- Load Previous Dataset
- Generate Heat Map
- Generate Vector Map
- Select Mice
- Start Animation
- Pause Animation
- Export Map

Use cases are provided in the “Functional Requirements” section of this document for each of these requirements.

The functional requirements were broken down into tasks and subtasks. In the “Project Schedule” section of this document we explain why we broke down tasks the way we did, which tasks are milestones, which tasks are deliverables, the amount of time we expect each task will take, which tasks are dependent on each other, and the schedule in which we plan to complete this project.

User Characteristics

The target users of this program are researchers in the fields of biology, psychology, and computer science at the University of Mary Washington. The researchers are studying the behavioral patterns of and interactions between mice. These users have achieved some level of college education and understand how to operate computers. We assume that they have had experience using basic software (Microsoft Word and PowerPoint), as well as software used to manipulate scientific data. The user’s goals for the software are:

- To input data generated by another system.
- Generate heat and vectors maps based on this data.

Functional Requirements

Every functional requirement is represented as a Use Case. The Use Case consists of (in order) the name, the description, the main flow, and the alternate flow.

Use Case 1	Load New Dataset
Description	The user loads a new dataset from a CSV file.
Main Flow	1. The user selects Load > New Dataset from the menu bar. 2. The system opens the file explorer, allowing the user to select a file to load. 3. The system checks that the file selected is properly-formatted. 4. The user is prompted to enter the grid dimensions. 5. The system saves the CSV file's location and the entered grid dimensions for easy loading in the future. 6. The system displays the new dataset.
Alternate Flow	3. The system detects that the file selected is not a properly-formatted CSV. 4. The system notifies the user that the system cannot load the file selected.

Use Case 2	Load Previous Dataset
Description	The user loads a previously-used dataset.
Main Flow	1. The user selects Load > Previous Dataset from the menu bar. 2. The system displays a list of previously-loaded datasets. 3. The user selects one of the datasets. 4. Using the stored file location and grid dimensions, the system loads the dataset. 5. The system displays the dataset.
Alternate Flow	4. The system cannot find the dataset using the stored file location. 5. The system notifies the user that the dataset file cannot be found.

Use Case 3	Generate Heatmap
Description	The system generates a heatmap from a loaded dataset.
Main Flow	The user presses the "Generate Heat Map" button. The system generates a heatmap from the dataset, with redder colors indicating a lot of activity and bluer colors indicating less activity. The system displays the heatmap.

Use Case 4	Generate Vector Map
Description	The system generates a vector map from a loaded dataset.
Main Flow	1. The user presses the “Generate Vector Map” button. 2. The system generates a vector map from the dataset, with each mouse’s vector being a unique colour. 3. The system displays the vector map.

Use Case 5	Select Mice
Description	The user selects which mice to display.
Main Flow	1. The user presses the “Select Mice” button. 2. The system displays a list of the mice present in the currently-loaded dataset. 3. The user selects the desired mice. 4. The system displays the selected mice on the grid.
Alternate Flow 1	4. The system displays the heatmaps for the selected mice.
Alternate Flow 2	4. The system displays the heat maps and vector maps for the selected mice.

Use Case 6	Start Animation
Description	The user starts the map animation.
Main Flow	1. The user presses the “Start Animation” button. 2. The system begins the animation.
Alternate Flow	2. The system resumes the currently-paused animation.

Use Case 7	Pause Animation
Description	The user pauses the animation.
Main Flow	1. The user presses the “Pause Animation” button. 2. The system pauses the animation.

Use Case 8	Export Map
Description	The user exports an image of the map at the current time.
Main Flow	1. The user presses the “Export Image” button. 2. The system opens the file explorer and allows the user to choose a name and location for the image to be saved. 3. The system saves an image of the current map to the specified location.
Alternate Flow	3. The system is unable to save the image to the desired location. 4. The system notifies the user that the export was unsuccessful.

Constraints

The users of the system will be using both Windows PCs and Macs. As such, the software must be cross-compatible. In order for the software to be cross-compatible, the bulk of the program must be written in Java. Other programming languages may be used, but they must be able to run using the Java Virtual Machine. As per the client’s requirements, there are no time performance requirements for the software, but the system must be able to handle up to 8 mice at once.

Project Schedule

Approach

Our approach to the project is to first learn how to use the tools/languages necessary to building the program. In this project, we plan to use NetBeans as our IDE and JavaFX to design the GUI. We plan to use Git and GitHub as our version control system, since everyone in the group has had experience working with Git and Github; we will not need to allocate time for the specific purpose of learning how to use our version control tools. Before writing any code, we will assign task to different team members and have some team members collaborate on the particularly challenging assignments.

Milestones and Deliverables

This section lists the tasks created based on the Functional Requirements. There are three types of tasks: Milestones, Deliverables, and subtasks; these tasks are represented in different ways in this table. Milestones are highlighted with dark gray and the text is in bold. Deliverables are highlighted with light blue and the text is italicized. The subtasks for each milestone are highlighted with dark grey and the text is italicized. The steps for implementing each subtask are bulleted.

Task	Name
M1	**Build a GUI**
T1	*Learn JavaFX*
	● Find online resources to help learn JavaFX
	● Study online resources
T2	*Implement GUI Skeleton*
	● Write code for GUI
	● Test code for GUI
D1	*Deliverable: GUI Skeleton*
T3	*Display Data On GUI*
	● Implement function code into the code for the GUI
	● Test code to make sure it works properly
	● Implement code to pick and choose for which mouse data is displayed
D2	*Deliverable: Working GUI*
M2	**Import Dataset**
T4	*Jython Tutorial and Data Parser*
	● Find an online tutorial for Jython
	● Work through Jython tutorial
	● Build parser to load data
	● Write code for parser
	● Test code for parser
T5	*Create Read Data Function(s)*
	● Write read data function(s)
	● Test read data function(s)

T6	*Create Save Data Function(s)*
	• Write save data function(s)
	• Test save data function(s)
M3	**Implement Heat Maps**
T7	*Generate Heat Map*
	• Brainstorm and create idea for algorithm as a team
	• Create plan for implementing algorithm in code and do research as necessary
	• Write code for heat map
	• Test code for heat map
D3	*Deliverable: Heat Map Generator*
T8	*Export Heat Map*
	• Write code to create and export image of map at a specific point in time
	• Test export map code
D4	*Deliverable: Heat Map Exporter*
T9	*Animate Heat Map*
	• Implement code to play animation
	• Implement code to stop animation
	• Implement code to pause animation
	• Implement code to allow user to choose specific time
D5	*Deliverable: Heat Map Animation*
D6	*Deliverable: Fully-Implemented Heat Maps*
M4	**Implement Vector Maps**
T10	*Generate Vector Map*
	• Examine code for generating heat map and brainstorm ideas for adjusting it to generate a vector map
	• Brainstorm and create idea for algorithm as a team
	• Create plan for implementing algorithm in code and do research as necessary
	• Write code for vector map
	• Test code for vector map
D7	*Deliverable: Vector Map Generator*
T11	*Export Vector Map*

	● Write code to create and export image of map at a specific point in time
	● Test export map code
D8	*Deliverable: Vector Map Exporter*
T12	*Animate Vector Map*
	● Examine code for animating heat map and brainstorm ideas for adjusting it to animate a vector map
	● Implement code for vector map animation based on the completed parts of heat map animation
D9	*Deliverable: Vector Map Animation*
D10	*Deliverable: Fully-Implemented Vector Maps*

Work Breakdown Structure

This section details the amount of time each milestone/deliverable will take, the justification for our estimates, and how many people we plan to assign to each task.

Build a GUI (48hrs)

At the end of this milestone, we expect to have a working GUI as a deliverable.

Learn JavaFX (18hrs)

Eighteen hours of this milestone has been allotted to learning how to work with JavaFX, because our team has little experience with building GUI's for java applications and with using JavaFX. The time it will take to watch video tutorials, read tutorials, and practice using NetBeans and JavaFX are all contributing factors to our estimated time of 18 hours. As we learn the basics of GUI building with JavaFX, we can implement each step we learn to our actually project. This subtask will be divided among two people in our team.

Implement GUI (12hrs)

We estimate that it will take another 12 hours to actually build a skeleton for our GUI. This is because it will be a slower step by step process based on the time it will take to learn the basics and when we reach sections of tutorials that can be applied specifically to this project. At the end of this subtask, we expect to have a GUI skeleton as a deliverable. This subtask will be divided among two people in our team.

Display Data On GUI (18hrs)

A final 18 hours is included in the time estimate for this subtask for implementing our functions into the GUI—linking the backend code and the front end user interface. We estimate 18 hours for this subtask, since we have very little experience building Java GUIs, and this subtask involves a fair amount of research, learning, and practice. This is unfamiliar territory, and we expect plenty of research and trial and error testing. This subtask will be divided among two people in our team.

Import Dataset (30hrs)

Jython Tutorial and Data Parser (18hrs)

Python's libraries that involve parsing data is simpler and more intuitive than the libraries in Java. Because the project is required to be built using Java, we decided to use Jython. This will allow the code for the data parser to be written with the simplicity of Python while still adhering to the client's requirements. We allotted 18 hours for this milestone; Finding tutorials, and learning is time consuming. The time it will take to build the parser along side with learning is why our time estimate is 18 hours. This subtask will be assigned to one person on our team.

Create Read Data Function(s) (6hrs) & Create Save Data Function(s) (6hrs)

We gave both of these subtasks 6 hours for the same reason. We kept them as lower estimated times because we all have experience writing programs that read and store data. But the

last time any of us worked with Java was over a year ago. Because we need to refresh ourselves on Java programming, these subtasks are estimated to take longer than 3 hours. We believe that 6 hours is an accurate estimate for both of these milestone due to the factors above. Each of these subtasks will be assigned to one person.

Implement Heat Maps (62hrs)

At the end of this milestone, we expect to have fully functional heat map capabilities as a deliverable.

Generate Heat Map (18hrs)

We estimate that this subtask will take 18 hours to complete. Similarly to previous subtasks, this subtask involves a fair amount of research; none of us has experience with data maps and visualization. We assume that there will be plenty of trial and error within this portion of the project. In order to ensure that we produce at least one working map generating function for the program, we will work on finishing the heat map capability of the program before we begin working on the vector map generating capability. At the end of this subtask, we expect to have heat map generation as a deliverable. This subtask will be assigned to all three members of the group.

Export Heat Map (8hrs)

This subtask is dependent on the completion of several previously stated subtasks. The most difficult part is completed within the previous subtasks. We estimate that this will take 8 hours, because even though the harder parts of this subtask have already been completed, a fair amount of time is still needed to learn how to export our visualizations. At the end of this subtask, we expect to have the ability of our program to export heat maps as a deliverable. This subtask will be assigned to two or three members of the group.

Implement Heat Map Animation (36hrs)

We decided that this deliverable will take the longest amount of time because of all the different parts that need to interact with each other. The 36 hours includes the time it will take to learn how to create animations (which neither of us have done before) and to

successfully implement this function. We allotted a large amount of time because this subtask involves unique parsing and reading of the data, interaction between the back-end code and the front-end visualization of the data, as well as the many different functions of the animation (start, stop, pause, step). At the end of this subtask, we expect to have heat map animation as a deliverable. This subtask will be assigned to all three members of the group.

Implement Vector Maps (62hrs)

At the end of this milestone, we expect to have fully functional vector map capabilities as a deliverable.

Generate Vector Map (18hrs)

Just like the “Generate Heat Map” subtask, this is on the longer end of our estimating because of the unfamiliarity of the implementation. With all the research and trial and error involved, we estimate that this subtask will take 18 hours to complete. In order to ensure that we produce at least one working map generating function for the program, we will only work on this capability of the program if we still have time after implementing the heat map generating capability. At the end of this subtask, we expect to have vector map generation as a deliverable. This subtask will be assigned to all three members of the group.

Export Vector Map (8hrs)

We estimate that this subtask will take 8 hours to complete. We will have already implemented the ability of our program to export heat maps at this point, and exporting vector maps will hopefully not be too different. But, since this is new to us, we want to make sure we leave enough time to complete this subtask. At the end of this subtask, we expect to have the ability of our program to export vector maps as a deliverable. This task will be assigned to two members of the group.

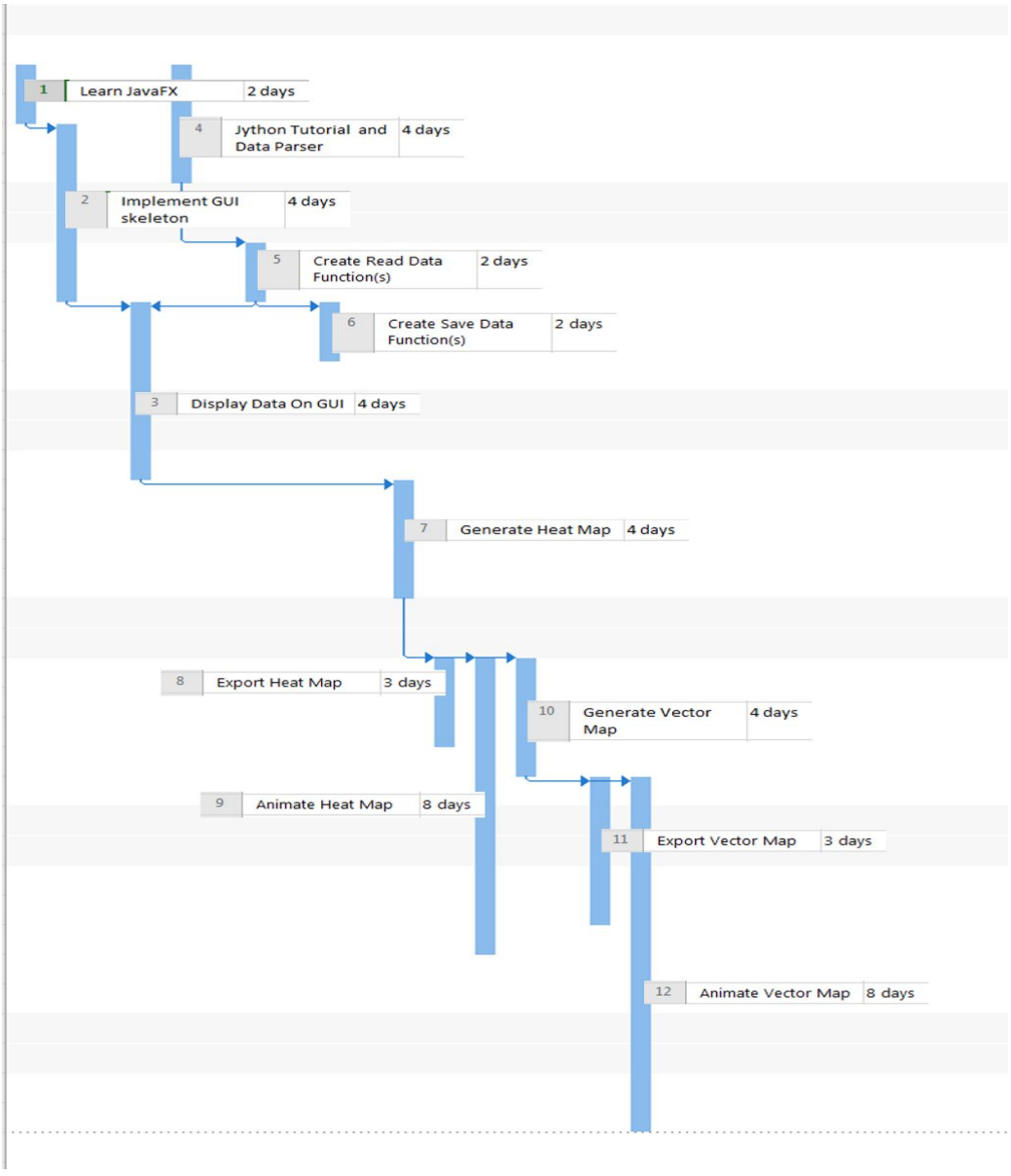
Implement Vector Map Animation (36hrs)

We estimate that this subtask will take 36 hours to complete. We will have already implemented animated heat maps into our program at this point, and at least some of that knowledge should be transferrable. But, there may be some inherent differences in

displaying vector maps that can are animated over time. We want to leave enough time to complete this subtask, as we are not really sure what it will entail. At the end of this subtask, we expect to have vector map animation as a deliverable. This task will be assigned to all three members of the group.

Gantt Chart

The Gantt Chart is a visual version of the tasks listed in the “Milestones and Deliverables” section and the days and duration data from the Task Dependency table. Each blue bar has the corresponding task number, task name, and duration (in days). This chart shows the tasks that are dependent on each other as well as the tasks that will be completed simultaneously.



Task Dependency Diagram

The full names and step-by-step plans for these tasks can be found in the “Milestones and Deliverables” section earlier in this document.

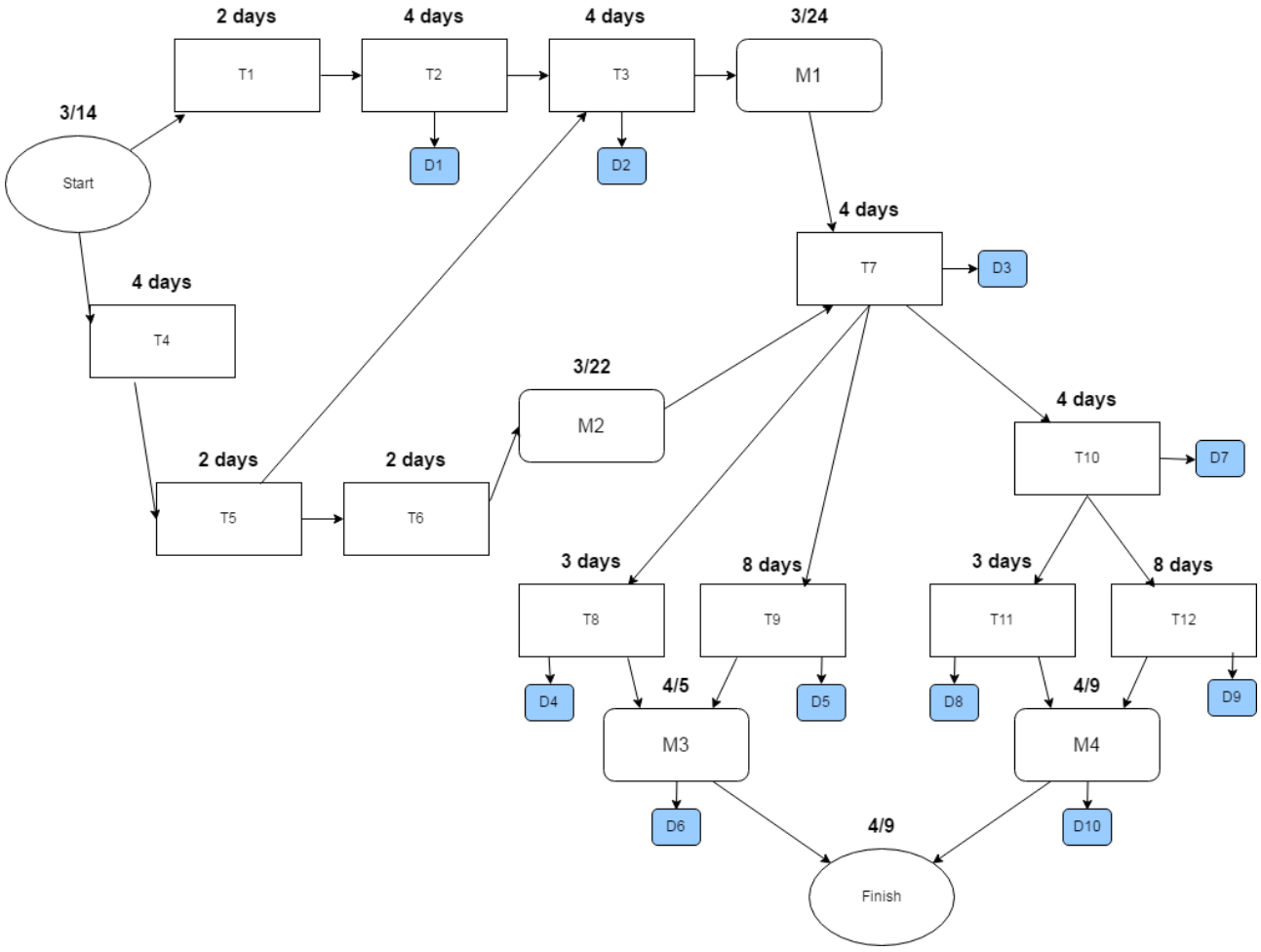
- T2 is dependent on T1, because the team needs to know how to use JavaFX in order to make a GUI with it.
- T3 is dependent on T2 and T5, as displaying a dataset on the GUI requires the GUI to have been built, and requires that a dataset can be read and parsed by the software.
- T5 is dependent on T4, as the software must be able to parse a dataset before it can read one in to be used.
- T6 is dependent on T5, as software must be loaded before it can be saved.
- T7 is dependent on M1 and M2, because datasets have to be able to be read and displayed before they can be used to generate heat maps to be displayed.
- T8 and T9 are dependent on T7, as exporting and animating heat maps obviously require the ability to generate heat maps in the first place.
- T10 is also dependent on T7, as the heat map has higher priority than the vector map. Due to this, the team will focus on getting heat maps working before beginning on vector maps. In addition, the code used to generate heat maps will likely be used as a reference when building vector maps.
- T11 and T12 are dependent on T10, as exporting and animating vector maps obviously require the ability to generate vector maps in the first place.

In table form, these dependencies are:

Task	Effort (person-hours)	Duration (days)	Dependencies
T1	6	2	
T2	12	4	T1
T3	18	4	T2,T5
T4	18	4	
T5	6	2	T4

T6	6	2	T5
T7	18	4	M1, M2
T8	8	3	T7
T9	36	8	T7
T10	18	4	T7
T11	8	3	T10
T12	36	8	T10

These dependencies, together with the times allotted in “Work Breakdown Structure” section of the document, can be used to create the following activity network diagram:



Appendix

Glossary

Cross-compatible - Software that isn't restricted to a single platform. In this context, the software must be able to run on Windows PCs and Macs.

CSV - "Comma-separated values", a simple tabular file format store in plaintext. CSV files have the file extension .csv.

Deliverable - Items delivered to a client of stakeholder intended to show progress.

GUI - "Graphical user interface", allows users to interact with the system using buttons and icons instead of through text-based commands.

Heat map - Graph using different colors to show which spaces on the grid were most commonly- visited. In this case, the maps show where the mice spent the most time.

IDE - "Integrated development environment", software that provides a user with the tools and space to build software.

Java Virtual Machine - Software that allows a computer to run Java programs.

JavaFX - A combination of graphics and media packages that help a programmer design, create, test, debug, and deploy software that function across different platforms.

Jython - Implementation of Python designed to run in Java. Allows Python programs to run using the Java Virtual Machine, thus allowing cross-platform compatibility.

Milestone - A recognized end of a project phase.

NetBeans - An Integrated Development Environment (IDE) used to program in Java. It has built-in GUI-creation tools.

Python - A programming language with better data parsing abilities than Java.

RFID - "Radio-frequency identification", a small device used to track animals.

Vector map - Graph using arrows to show path of movement through the grid. In this case, it shows where the mice moved.

Contributions

Cole Hodges - User Characteristics, Approach, Milestones and Deliverables, Work Breakdown Structure, Gantt Chart.
Evan May - Functional Requirements, Constraints, Milestones and Deliverables, Task Dependency Diagram, Glossary.
Nate Nuval - Introduction, Project Scope, Work Breakdown Structure, Gantt Chart.

References

Reference 1: Test Data

This is one of the test datasets provided by Dr. Polack as an example of what the datasets used by the program will look like.

```
DateTime;IdRFID;IdLabel;unitLabel;eventDuration;sense1duration;sense1Events;sen
seRFIDrecords;outFuncLabel;outLabel;SystemMsg;MsgValue1;MsgValue2;MsgValue3
#ID-Device;RFID1;0;0;0;SAM
#ID-Device;RFID2;0;155;0;SAM
#ID-Device;RFID3;45;775;0;SAM
#ID-Device;RFID4;45;2325;0;SAM
#ID-Device;RFID5;90;0;0;SAM
#ID-Device;RFID6;90;155;0;SAM
#ID-Device;RFID7;135;775;0;SAM
#ID-Device;RFID8;135;2325;0;SAM
#ID-Device;RFID9;180;0;0;SAM
#ID-Device;RFID10;180;155;0;SAM
#ID-Device;RFID11;225;775;0;SAM
#ID-Device;RFID12;225;2325;0;SAM
#ID-Device;RFID13;270;0;0;SAM
#ID-Device;RFID14;270;155;0;SAM
#ID-Device;RFID15;315;775;0;SAM
#ID-Device;RFID16;315;2325;0;SAM
#ID-Device;RFID17;360;0;0;SAM
#ID-Device;RFID18;360;155;0;SAM
#ID-Device;RFID19;405;775;0;SAM
#ID-Device;RFID20;405;2325;0;SAM
#ID-Device;RFID21;450;0;0;SAM
#ID-Device;RFID22;450;155;0;SAM
#ID-Device;RFID23;495;775;0;SAM
#ID-Device;RFID24;495;2325;0;SAM
#ID-Device;RFID25;0;0;0;SAM2
42669.7918760417;;;Control;;;;;;;start;SAM25_ind.xlsx;;
42669.7919392361;041940A1C3;unknown;RFID20;200;;;2;;;;;;
42669.7919422338;041940A1C3;unknown;RFID22;716;;;3;;;;;;
```

```
42669.7919687963;041940A1C3;unknown;RFID20;176;;;1;;;;;;
42669.7919659028;041940A1C3;unknown;RFID22;1729;;;3;;;;;;
42669.7920011227;041940A1C3;unknown;RFID22;202;;;2;;;;;;
42669.7920161227;041940A1C3;unknown;RFID20;201;;;1;;;;;;
42669.7920206019;041940A1C3;unknown;RFID22;1720;;;4;;;;;;
42669.7920603472;041940A1C3;unknown;RFID22;713;;;3;;;;;;
42669.7920940046;0419408444;unknown;RFID20;228;;;1;;;;;;
42669.7920913889;041940A1C3;unknown;RFID22;603;;;2;;;;;;
42669.7921115162;0419408444;unknown;RFID20;200;;;1;;;;;;
42669.7921158912;041940A1C3;unknown;RFID22;201;;;1;;;;;;
42669.7921214931;041940A1C3;unknown;RFID18;200;;;1;;;;;;
42669.7921407755;;;Control;;;;;;;end;;;
```