Text-to-Art Interpreter: Project Management Plan

By Lawrence Dickey, Wallis Muraca, Adam Carlson

Revision History:

Date	Version	Description	Authors
10/08/17	1.0	Initial version of document	Lawrence Dickey, Wallis Muraca, Adam Carlson
11/15/17	2.0	Rewritten document according to feedback	Lawrence Dickey, Wallis Muraca, Adam Carlson

Table of Contents:

List of Figures:

List of Tables:

1. Overview

1.1 Project Summary

1.1.1 Purpose, Scope, and Objectives

For this project, we will be creating software which allows users to visualize literary texts in new artistic forms. The purpose of this software is to help users, such as artists, avid readers, or the curious individual, to see influential texts take new shapes. These literary texts could range from classic novels to smaller poems written by Colby students. Once a text file is read by our program, we will create an abstract drawing of the text or a musical song representation, if time permits during project production. In terms of our potential users, both Colby arts and non-arts students and faculty have expressed interest in being able to generate these artistic interpretations of texts out of sheer fascination.

1.1.2 Assumptions and Constraints

We assume that it is possible to create a unique mapping between literary texts and abstract drawings in the allotted amount of time this semester for this course. Given this assumption, we should be able to create a unique picture for every different text file that the program reads in. Our main constraint is the processing power of the user's computer, which could adversely affect our program's runtime. However, we plan on building flexible enough software to handle various computer models and operating systems.

1.1.3 Project Deliverables

For this project, we will deliver both our working software and our various forms of documentation. This documentation will include a project management proposal, a software requirements specification, a software design document, along with a final project report which will include testing specifics as well as higher level information about the project.

1.1.4 Schedule and Budget Summary

We plan to be able to successfully build the software given 4 weeks of work. More specific details on how those weeks will be split up among various activities is described in further detail in section 5. Our software will not cost any money to create. Therefore, we have no budget to report on.

1.2 Evolution of the Plan

As our plan evolves, we will update our project documents accordingly and relay the information to relevant stakeholders such as users and members of our group. This communication will help the project stay on schedule.

2. References: IEEE Standard for Software Project Management Plans

3. Definitions
Please see our glossary of definitions in Appendix A.

4. Project Organization

4.1 External Interfaces
We will not require any external interfaces for the core of this project. All software can be run on the user's machine without the use of a server to fetch additional information.

4.2 Internal Structure
Our software will be broken up into numerous classes which will each handle one task and one task only. These various classes will keep our code modular, allowing for easier maintenance and further buildability in the future. Our software will roughly be broken up into GUI, text processing, and drawing creation classes and will follow a pipe-and-filter data processing pattern. However, our software design document will go into much further detail about our system architecture.

4.3 Roles and Responsibilities
Lawrence Dickey, Wallis Muraca, and Adam Carlson will be the main drivers of this project. Before beginning the developing process, each team member will make a thorough list of the requirements needed for the project and review with the other teammates in order to achieve clarity going further. Once done, each team member will spend a large portion of their time implementing the end product. However, we also break down higher level roles for project organization going forward.

Because of Lawrence's background in data structures, he will be the main driver of project design and architecture. This placement is to ensure that the software is created in an efficient and scalable manner. Due to Wallis' interest and experience with UX, she will work on the user experience software layout of the project. This assignment is to ensure that the end customer is getting a quality product that works intuitively. Finally, because of Adam's industry experience, he will be in charge of software testing and coding quality assurance. This testing and review will ensure the product works for all potential customers and that the code is maintainable for future development.

5.Managerial Process Plans

5.1 Start-up Plan

5.1.1 Estimation Plan

Previously, we estimated that our product production would take 4 weeks. In this section, we demonstrate where this estimation came from. We have divided our major features into smaller activities and estimated how long it would take to implement each (measured in days). We then used the Critical Path Method to create a diagram that would give us an idea of which activities could be delayed and which were critical to a timely release. The table and chart are below:

Activity	Predecessors	Completion Time (days)	Start Date	End Date
A Read in a file		1	11/20	11/21
B Store data in a useful way	A	2	11/21	11/23
C Build a UI for the draw space		4	11/20	11/25
D Build a UI to open a file	E	3	11/24	11/27
E Create user controls (buttons, etc.)		4	11/20	11/24
F Create the drawing	B, C, D	5	11/27	12/2
G Create music	D	9	11/27	12/6
		Total: 16 days (Critical Path)		

Table 1 - Project divided into major activities with time estimations

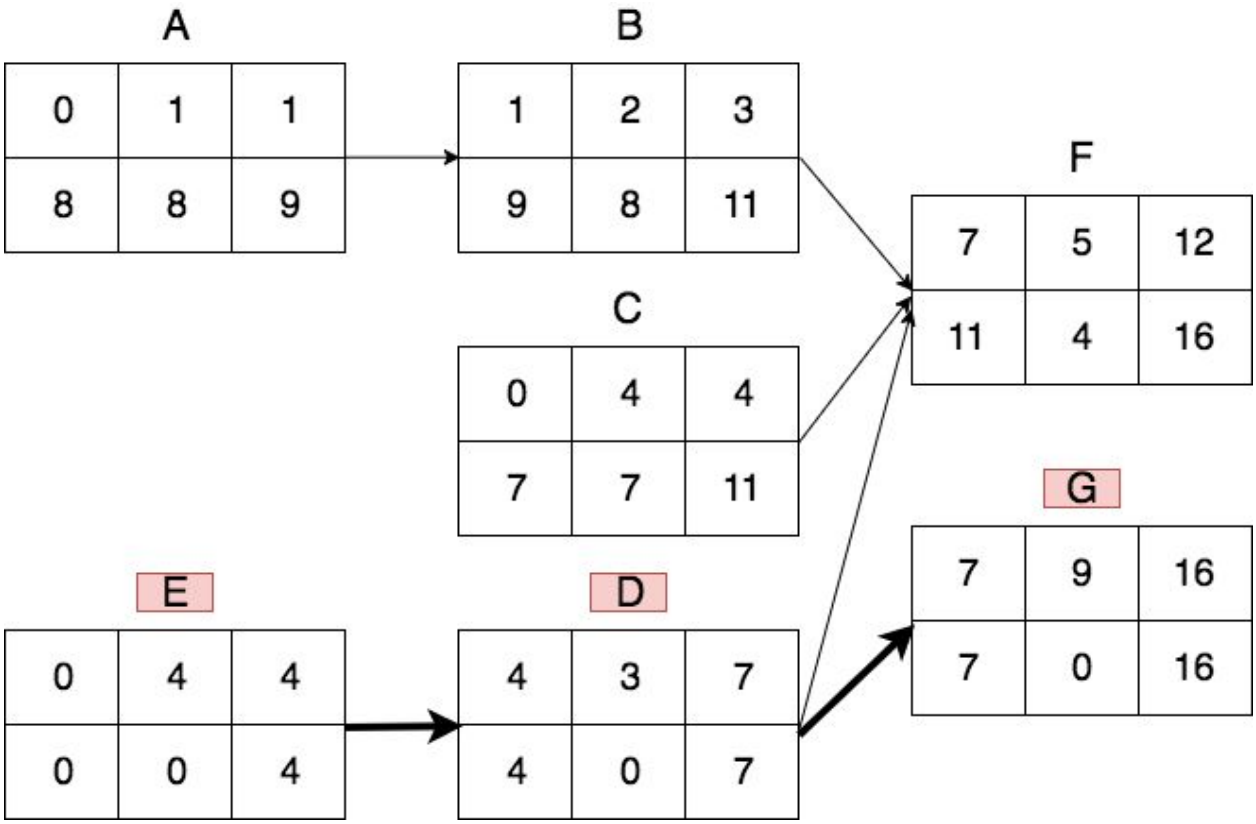


Figure 1 - Critical Path Figure for our Text-to-Art Interpreter

From the chart, we see that the critical path is to create user controls, build a UI to open files, and generate music.

5.1.2 Staffing Plan

This project should only require the three team members currently assigned to it. Particular skillsets and talent recruiting are not necessary for this particular project. As stated above, given Lawrence Dickey, Wallis Muraca, and Adam Carlson working on this project, the duration of need is estimated at 4 weeks.

5.1.3 Resource Acquisition Plan

No resource acquisition plan is needed.

5.1.4 Project Staff Training Plan

This project does not require any significant staff training.

5.2 Work Plan

5.2.1 Work Activities
Please see section 5.1.1 in order to see a relevant breakdown of work activities. Further details are not needed for this project.

5.2.2 Schedule Allocation
Please see section 5.1.1 to see the relationships and constraints between various activities in this project.

5.2.3 Resource Allocation
NA

5.2.4 Budget Allocation
NA

5.3 Control Plan

5.3.1 Requirements Control Plan
All requirement changes will be communicated to relevant stakeholders, including other developers on this team. Communication will be primarily done through email, face-to-face conversations, and phone calls/text messages. In addition, we will update our software requirements specification document in order to reflect the changes. After such records have been changed, developers and stakeholders will collectively assess the impact to the schedule by re-examining the project's critical path as seen in **Figure 1**.

5.3.2 Schedule Control Plan
To make sure we stay on track, we will have weekly meetings to assess our individual progress and progress as whole developing team. The amount of time put into these weekly meetings will be recorded in a shared Google Doc. In addition, we will make use of Wunderlist in order to keep track of what each team member is working on. If we begin to fall behind, we will alert our stakeholders and discuss which features can be postponed to a future time beyond our current deadline. This decision will be made looking at the priority of each feature to the overall success of the software.

5.3.3 Budget Control Plan
NA

5.3.4 Quality Control Plan
We will ensure the quality of our work by performing and documenting tests for each module of our software. In addition, we will ask for feedback on completed portions of the projects from

potential users and stakeholders in order to determine whether we are meeting proper expectations. Finally, various members of the developer team will perform code review on other member's code in order to make sure it is free of bugs and is well-structured.

5.3.5 Reporting Plan

As previously stated, the development team will meet at least once a week in person and will also communicate via technology on a more frequent basis. As requirements are finished, progress will be communicated to other team members in order to ensure we are always working on the highest priority feature of the project. In addition, this communication will allow team members to work on separate modules of the project concurrently with one another reacting to changes or completion of requirements as necessary.

5.3.6 Metrics Collection Plan

We will record how long weekly meetings take in a Google Doc and keep track of what each team member is working on and how long that activity took through the help of an app called Wunderlist.

5.4 Risk Management Plan

This project is uniquely void of extreme monetary risks. Adam, Wallis, and Lawrence have the educational advantage of not having invested personally in the development of the software. Regardless, the time requirements to complete the project are substantial.

To ensure that time is being spent effectively, the group plans to follow a schedule as closely as possible. Planning ahead means that should the group encounter difficulties with one element of the project, there should still be enough time to make adjustments and complete the project on time. Here is a summary of possibly risks.

Risk Description:	Probability of Occurrence:	Loss Size(Days):	Risk Exposure (Days):
Parser unable to sort characters or method used to store information inefficient or faulty	40%	5	2
Inconsistencies in results will make it difficult to produce truly fascinating artistic results	35%	10	5

Midterms and other classes/tests/projects interfere with productivity	25%	3	0.75
User interface is too difficult to interpret/use and must be overhauled	25%	12	3
Testing reveals significant holes in parsers quality once alternative texts are considered	50%	3	1.5
Project completed close to deployment date causing lack of external testing by non-team members	80%	2	1.6
		Total Exposure:	13.85

5.5 Closeout Plan

We plan on presenting our project to the class at the end of the semester in order to demonstrate what we have achieved as a team. No further closeout plan is necessary.

6. Technical Process Plans

6.1 Process Model

The timing and breakdown of our work activities are specified in our estimation plan. We plan on staying on track by keeping each other informed of our progress frequently and ensuring we are targeting the project requirements effectively. The project deliverables, as previously mentioned, are the PMP, SRS, SDD, final report, and the working software itself. These deliverables are due mid December at the end of this semester.

6.2 Methods, Tools, and Techniques

We will use python as our core programming language. In order to program efficiently, we will make use of the PyCharm IDE for python. No hardware is needed for the development of this project.

6.3 Infrastructure Plan

We will all have access to the current state of our software by making heavy use of git via GitHub. No other specific development environment will be required for the project.

6.4 Product Acceptance Plan
We will present our final product at the end of the semester and explain why those in the art community and beyond might find it useful.

7. Supporting Process Plans

7.1 Configuration Management Plan
NA

7.2 Verification and Validation Plan
We will verify we built the correct project by returning to our users and ensuring we met the requirements they established. We will validate the project by taking a step back and determining whether the software we built met our individual goals since no explicit business needs were set in place.

7.3 Documentation Plan
We will create a user manual for the software. In addition, we plan on developing a PMP, SRS, SDD, and a final report which details testing and the high level details of the project.

7.4 Quality Assurance Plan
We will assure quality in our software by performing code reviews on each other's code and by testing the software and documenting such tests. In addition, we will constantly return to the project requirements and users to ensure we stay on track and make the correct features.

7.5 Reviews and Audit Plans
NA

7.6 Problem Resolution Plan
We will document problems in our software alongside our testing documentation. Whenever a test fails, it will be documented for everyone in the group to see and respond to. We plan on submitting a final product with no failed tests.

7.7 Subcontractor Management Plans
NA

7.8 Process Improvement Plan
During our weekly in-person meetings, we will identify any blockages or slowdowns caused by our current process. If a solution exists that can be agreed upon in such meetings, we will change our process in order to expedite the delivery of quality software.

8. Additional Plans

We have no additional plans to record at this time.

9. Appendices

9.1 Appendix A: Glossary of Definitions

GUI:

Graphical User Interface

OOD:

Object-oriented design

Parser:

A parser is a module which can sift through large amounts of text, or more generally data, and return useful information for an application.

Pen Stroke:

The drawing of a single line using the turtle object

The Standard Words Per Page:

We define the standard words per page of a text document to be 300.

Turtle:

A python module providing turtle graphics used for drawing simple lines and pen strokes.

Project personnel and organization:

Lawrence Dickey, Wallis Muraca, and Adam Carlson will be the main drivers of this project. Before beginning the process, each team member will make a thorough list of the requirements needed for the project and review with the other teammates in order to achieve clarity going further. Once done, each team member will spend a large portion of their time implementing the end product. However, we also break down higher level roles for project organization going forward.

Because of Lawrence's background in data structures, he will be the main driver of project design and architecture. This placement is to ensure that the software is created in an efficient and scalable manner. Due to Wallis' interest and experience with UX, she will work on the user experience software layout of the project. This assignment is to ensure that the end customer is getting a quality product that works intuitively. Finally, because of Adam's industry experience, he will be in charge of software testing and coding quality assurance. This testing and review will ensure the product works for all potential customers and that the code is maintainable for future development.

In order to stay on track, the group will meet in person each week to discuss potential blockages and to plan for what is needed the following week. In addition, constant communication will continue via email. Lawrence will be the main project lead. Therefore, Adam and Wallis will report to him, and he will report to Professor Codabux throughout the project.

Tracking project progress:

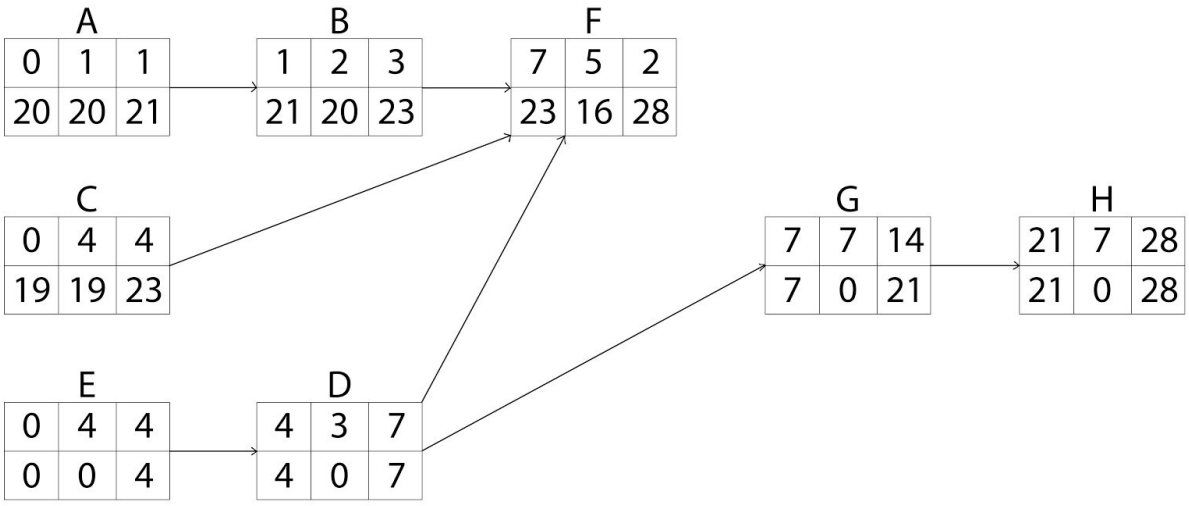
There are a number of activities that are critical and must be completed to finish the project on time. Those activities include writing the parser, storing that input, and generating the relevant image and sounds. We also have a number of extra features that, while it would vastly improve the software, are not critical to the implementation of its basic function. We would like to implement this features if we have extra time. Examples include more user customization, exporting the final image as a .PNG or animated .GIF, and sharing the image through social media.

To make sure we stay on track, we will have weekly meetings as mentioned in "Project Personnel and Organization."

Effort and schedule estimation:

To gain an understanding of how long this project would take to complete, we divided our major features into smaller activities and estimated how long it would take to implement each (measured in days). We then used the Critical Path Method to create a diagram that would give us an idea of which activities could be delayed and which were critical to a timely release. The table and chart are below:

	Activity	Predecessors	Completion Time (days)
A	Read in a file		1
B	Store data in a useful way	A	2
C	Build a UI for the draw space		4
D	Build a UI to open a file	E	3
E	Create user controls (buttons, etc.)		4
F	Create the drawing	B, C, D	5
G	Produce a sound	D	7
H	Create music	G	7
			Total: 4 weeks



From the chart, we see that the critical path is to create user controls, build a UI to open files, produce sound, and generate music.

Risk management:
This project is uniquely void of extreme monetary risks. Adam, Wallis, and Lawrence have the educational advantage of not having invested personally in the development of the software. Regardless, the time requirements to complete the project are substantial.

To ensure that time is being spent effectively, the group plans to follow a schedule as closely as possible. Planning ahead means that should the group encounter difficulties with one element of the project, there should still be enough time to make adjustments and complete the project on time.

Risk Description:	Probability of Occurrence:	Loss Size(Days):	Risk Exposure (Days):
Parser unable to sort characters or method used to store information inefficient or faulty	40%	5	2
Inconsistencies in results will make it difficult to produce truly fascinating artistic results	35%	10	5
Midterms and other classes/tests/projects interfere with productivity	25%	3	0.75
User interface is too difficult to interpret/use and must be overhauled	25%	12	3
Testing reveals significant holes in parsers quality once alternative texts are considered	50%	3	1.5
Project completed close to deployment date causing lack of external testing by non-team members	80%	2	1.6
		Total Exposure:	13.85

Version 1.0

Artistic Representations of Texts

Introduction:

For this project, we will be creating software which allows users to visualize literary texts in new artistic mediums. Literary texts could range from classic novels such as *The Great Gatsby,* to smaller poems written by Colby students. Essentially, our program will have the capability to process any text file. Once a text file is read in, the software will have the capability to create either an abstract drawing of the text or a musical song representation. In terms of our potential users, both Colby arts and non-arts students have expressed interest in being able to generate these artistic interpretations of texts out of sheer fascination.

Project personnel and organization:

Lawrence Dickey, Wallis Muraca, and Adam Carlson will be the main drivers of this project. Before beginning the process, each team member will make a thorough list of the requirements needed for the project and review with the other teammates in order to achieve clarity going further. Once done, each team member will spend a large portion of their time implementing the end product. However, we also break down higher level roles for project organization going forward.

Because of Lawrence's background in data structures, he will be the main driver of project design and architecture. This placement is to ensure that the software is created in an efficient and scalable manner. Due to Wallis' interest and experience with UX, she will work on the user experience software layout of the project. This assignment is to ensure that the end customer is getting a quality product that works intuitively. Finally, because of Adam's industry experience, he will be in charge of software testing and coding quality assurance. This testing and review will ensure the product works for all potential customers and that the code is maintainable for future development.

In order to stay on track, the group will meet in person each week to discuss potential blockages and to plan for what is needed the following week. In addition, constant communication will continue via email. Lawrence will be the main project lead. Therefore, Adam and Wallis will report to him, and he will report to Professor Codabux throughout the project.

Tracking project progress:

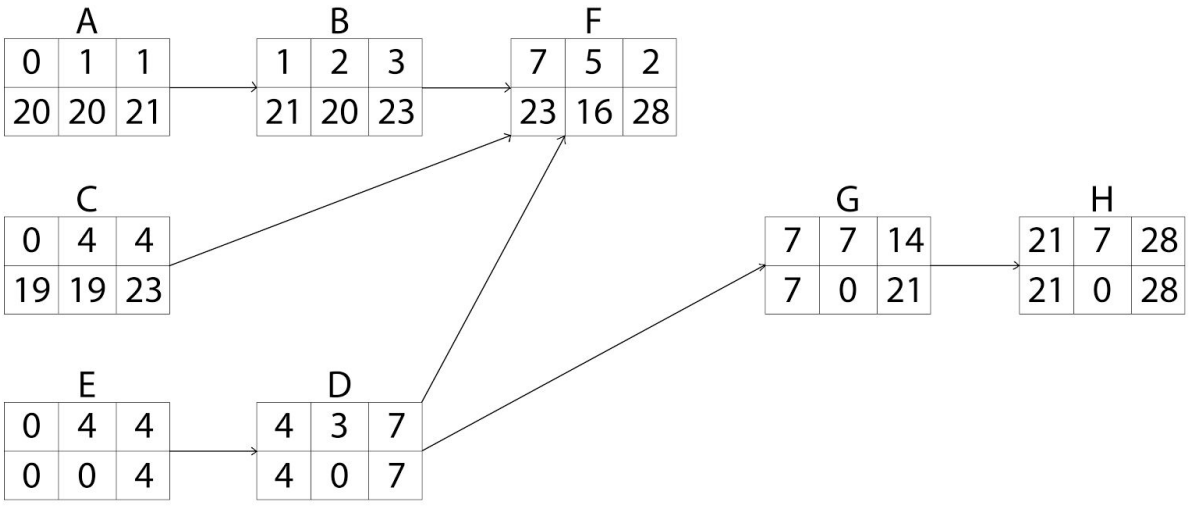
There are a number of activities that are critical and must be completed to finish the project on time. Those activities include writing the parser, storing that input, and generating the relevant image and sounds. We also have a number of extra features that, while it would vastly improve the software, are not critical to the implementation of its basic function. We would like to implement this features if we have extra time. Examples include more user customization, exporting the final image as a .PNG or animated .GIF, and sharing the image through social media.

To make sure we stay on track, we will have weekly meetings as mentioned in "Project Personnel and Organization."

Effort and schedule estimation:

To gain an understanding of how long this project would take to complete, we divided our major features into smaller activities and estimated how long it would take to implement each (measured in days). We then used the Critical Path Method to create a diagram that would give us an idea of which activities could be delayed and which were critical to a timely release. The table and chart are below:

	Activity	Predecessors	Completion Time (days)
A	Read in a file		1
B	Store data in a useful way	A	2
C	Build a UI for the draw space		4
D	Build a UI to open a file	E	3
E	Create user controls (buttons, etc.)		4
F	Create the drawing	B, C, D	5
G	Produce a sound	D	7
H	Create music	G	7
			Total: 4 weeks



From the chart, we see that the critical path is to create user controls, build a UI to open files, produce sound, and generate music.

Risk management:
This project is uniquely void of extreme monetary risks. Adam, Wallis, and Lawrence have the educational advantage of not having invested personally in the development of the software. Regardless, the time requirements to complete the project are substantial.

To ensure that time is being spent effectively, the group plans to follow a schedule as closely as possible. Planning ahead means that should the group encounter difficulties with one element of the project, there should still be enough time to make adjustments and complete the project on time.

Risk Description:	Probability of Occurrence:	Loss Size(Days):	Risk Exposure (Days):
Parser unable to sort characters or method used to store information inefficient or faulty	40%	5	2
Inconsistencies in results will make it difficult to produce truly fascinating artistic results	35%	10	5
Midterms and other classes/tests/projects interfere with productivity	25%	3	0.75
User interface is too difficult to interpret/use and must be overhauled	25%	12	3
Testing reveals significant holes in parsers quality once alternative texts are considered	50%	3	1.5
Project completed close to deployment date causing lack of external testing by non-team members	80%	2	1.6
		Total Exposure:	13.85