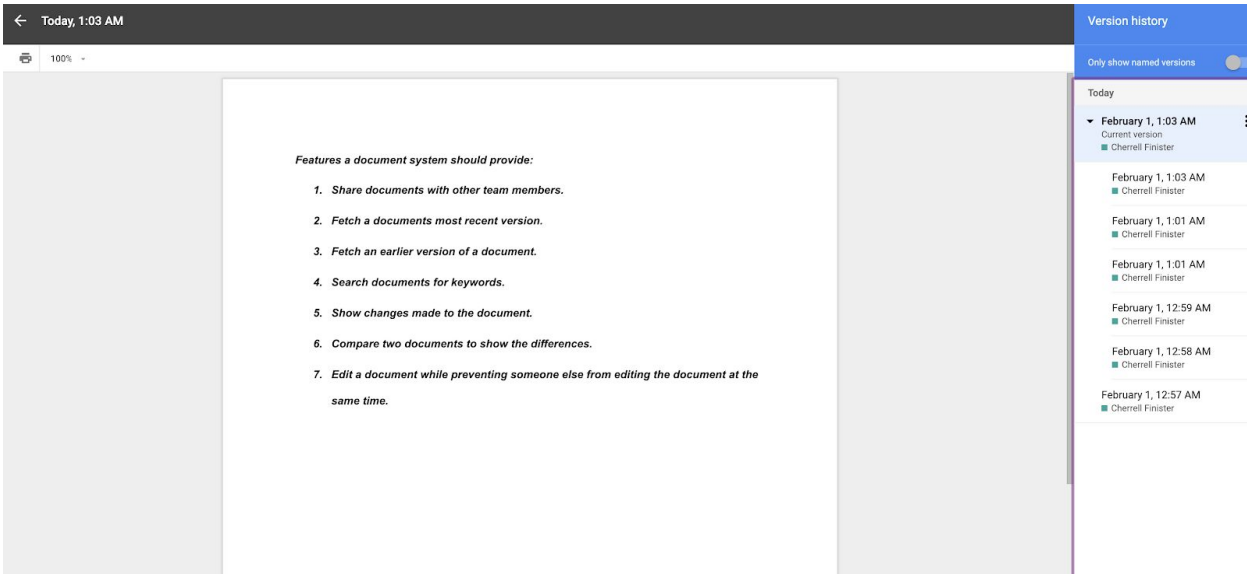
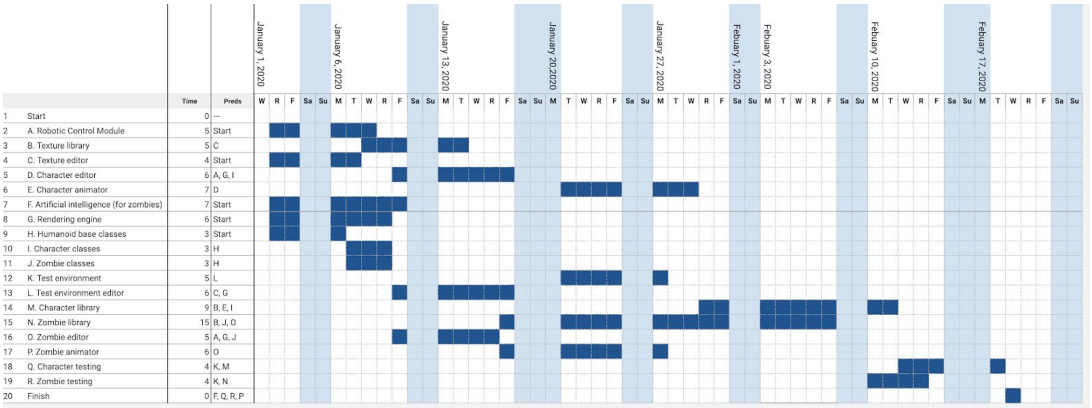
Cherrell Finister
Professor BJ
CMSI 402: Senior Project Lab
6 February 2019

Homework #1

1. What are the basic tasks that all software engineering projects must handle?
    a. All software engineering projects must be able to the handle the tasks of requirements gathering, high level design, low level design, development, testing, deployment, maintenance, and wrap up.
2. Give a one sentence description of each of the tasks you listed in number 1.
    a. Gathering together what the users need and want within the application; coming up with a plan for what is going to be required of you to develop.
    b. High level design is information about what platforms will be used, data design, interfaces, and etc, as well as explaining the project's architecture at a high level; breaking the application down into pieces.
    c. Low level design is narrowing down those high level requirements into groups and information about how it should work.
    d. Development is the implementation of all requirements set up throughout the requirements document and both the high/low level plans to the point of programmers being able to implement those codes/designs.
    e. Testing is using the code in different situations in order to fix bugs or flaws within the project.
    f. Deployment is rolling out the application to users.
    g. Maintenance is keeping the application up to date for all users; as soon as they start using it sometimes they find bugs and it is up to the developers to continually be on the lookout to fix them.
    h. Wrap up is doing an overview of the project and determine what went right and what went wrong with the project in order to learn from what was done.
3. Like Microsoft Word, Google Docs provides some simple change tracking tools. Then create a document, save it, close it, reopen it, and make changes to it as you did in Exercise 1.

a. 

4. What does JBGE stand for and what does it mean?
    a. JBGE stands for “just barely good enough” and it means that you should only include the comments in your code that is the absolute necessity to understand the code of the project.
5. Use critical path methods to find the total expected time from the project's start for each task completion. Find the critical path. What are the tasks on the critical path? What is the total expected duration of the project in working days?
    a. The amount of working days is 32 days
    b. The cpm is G → D → E → M → Q and the tasks in that path are tasks G, D, E, M, Q.
6. Build a Gantt chart for the network you drew in Exercise 3. Start on Wednesday, January 1, 2020, and don't work on weekends or the following holidays.

a. 

    i. *Couldn’t figure out how to add arrows in google sheets*

7. In addition to losing time from vacation and sick leave, projects can suffer from problems that just strike out of nowhere. Sort of a bad version of *deus ex machina*. For example, senior management could decide to switch your target platform from Windows desktop

PSs to the latest smartwatch technology. Or a strike in the Far East could delay the shipment of your new servers. Or one of your developers might move to Iceland. How can you handle these sorts of completely unpredictable problems?

   a. You treat these problems all the same where you add these lost times at the end of schedule. When something does go wrong, you would add it into the tasks.

8. What are the two biggest mistakes you can make while tracking tasks?
   a. The biggest mistake you can make you can make is not taking action when tasks are behind and to always make sure you are up to date with tasks and watching for deadlines. The second biggest mistake is adding too many people to a task to cut time, as it might seem like a good idea to get things done quicker but to split a task into multiple roles could get confusing.
9. List five characteristics of good requirements.
   a. Easy to understand
   b. Unambiguous
   c. Consistent
   d. Prioritized
   e. Verifiable
10. For this exercise, list the audience-oriented categories for each requirement. Are there requirements in each category? [If not, state why not…]
   a. Business
   b. User and functional
   c. User and functional
   d. User and functional
   e. Nonfunctional
   f. Nonfunctional
   g. Nonfunctional
   h. Nonfunctional
   i. Nonfunctional
   j. Functional
   k. Functional
   l. User and functional
   m. User and functional
   n. User and functional
   o. User and functional
   p. User and functional
      i. Implement category has no requirements because that is only needed when an application is transition or updating a system and this project is not changing databases or anything like that.

11. Brainstorm this application and see if you can think of ways you might change it. Use the MOSCOW method to prioritize your changes.
    a. Quick Guess (could) : Could be added so that users can guess all at once a word if they think they might know what it is
    b. Multiple levels (Could) : Could add multiple levels for users to choose from so that they could be either beginner, normal, or hard if they are good at the game.
    c. Being able to drag the letters (won't) : allow users to drag and drop letters instead of clicking on them to place them
    d. Changing the colors (won't) : allow the users to change the colors of the game from the previous background and letter colors