Thomas O’Brien

CMSI 402

1/29/18

Homework Assignment #1

**1.1.** The basic tasks that all software engineering projects must handle are the following: requirements gathering, high-level design, low-level design, development, testing, deployment, maintenance, wrap-up.

**1.2.** Requirements gathering: Doing research to figure out exactly what the customer both wants and needs.

High-level design: Making decisions on which platforms to use, what data design to use, interfaces with other systems, and how these are all used together to make the application

Low-level design: Determining how to actually build all of the needed pieces of the application so that other software engineers can begin to program.

Development: Begin to write the required code for the application.

Testing: Write tests that use the application under various conditions to maximize what the tests actually cover.

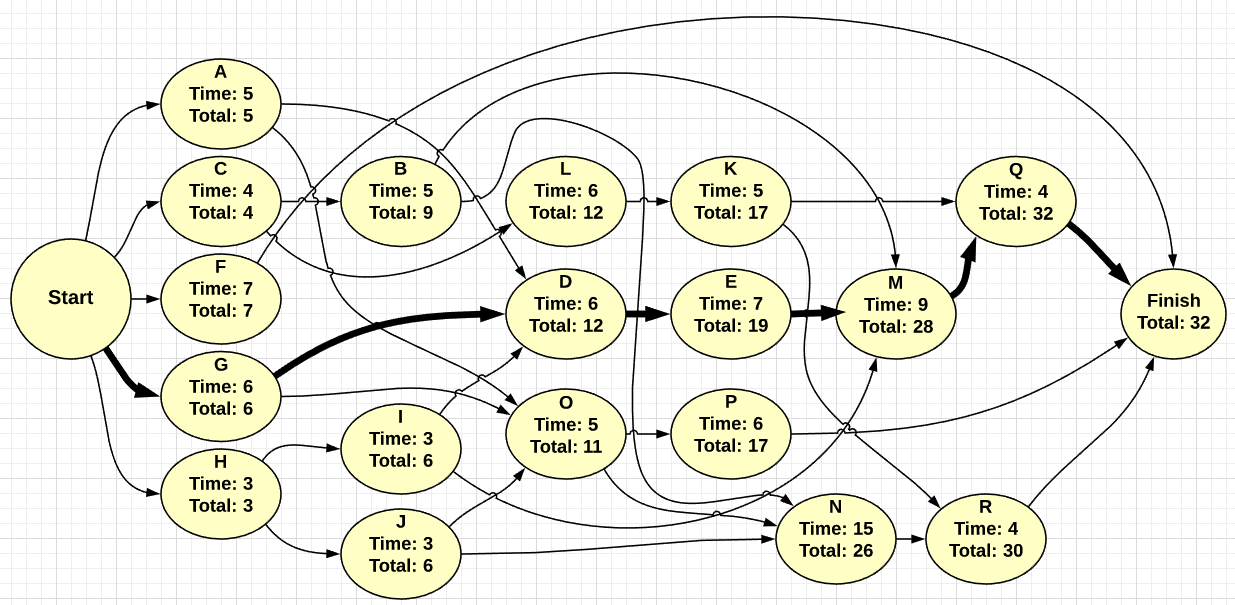
Deployment: Push the application to production and open access to users.

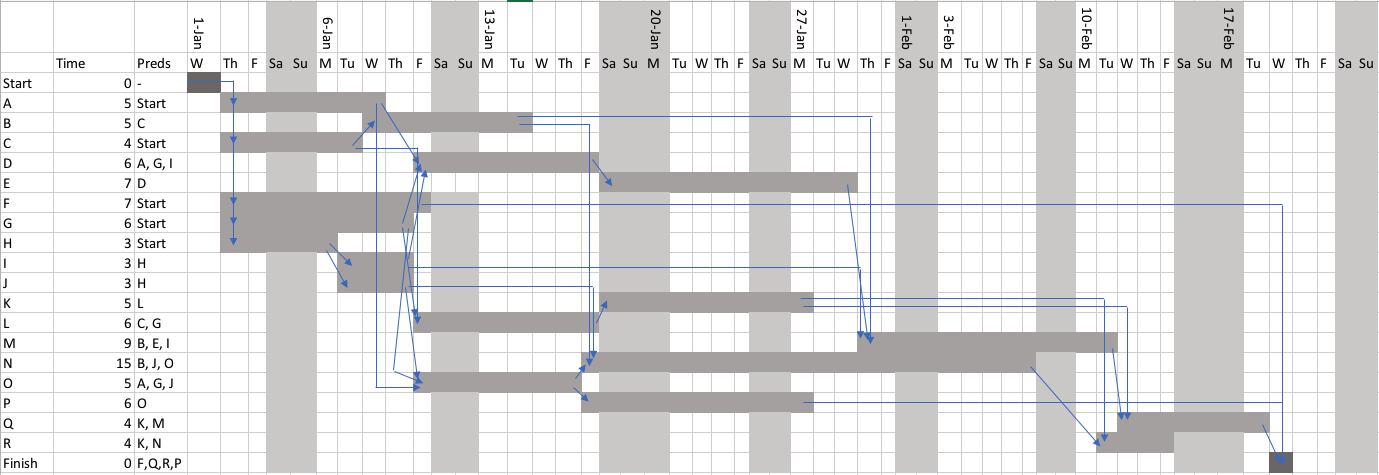
Maintenance: Fix bugs that appear and continue to develop the application through new features, improvements, and other kinds of updates.

Wrap-up: Look over the application’s development history to determine what went right and wrong and use this to learn from experience.

**2.4.** Exercise was conducted.

**2.5.** JBGE stands for “just barely good enough”. The idea here is to not write too many comments in one’s code. Too many comments will be required to be changed as the code is modified over time. However, this is not a reason to neglect writing comments. It is important to write clear, concise comments to get the message across.

**3.2.**  The critical path takes 32 days. The tasks done are in bold (G, D, E, M, and Q).

**3.4**

**3.6** The best way to deal with these sorts of completely unpredictable problems is to treat them like tasks that take up a certain number of days. Insert these tasks into your schedule in order to ensure that they are considered for. By doing this, the team is able to understand why a project might be delayed by a certain number of days and still understands the task at hand.

**3.8** One mistake that is commonly made is not taking action when a task begins to slip a bit. This does not mean one should panic if someone is a bit behind, but it is important to take action in case they begin to fall more and more behind. The second common mistake would be piling extra developers on a task and assume that they can reduce the time needed to finish it. Sometimes, a new member will require time to get them caught up to speed, which ultimately will not shorter the total time.

**4.1** The five characteristics of good requirements are clear, unambiguous, consistent, prioritized, and verifiable.

**4.3** a. Business

b. User/Functional

c. User/Functional

d. User/Functional

e. Nonfunctional

f. Nonfunctional

g. Nonfunctional

h. Nonfunctional

i. Nonfunctional

j. Functional

k. Functional

l. User/Functional

m. User/Functional

n. User/Functional

o. User/Functional

p. User/Functional

Chances are that the user will have all of the needed tools and hardware to build this application. The main concern would be ensuring that the machine running this application has plenty of hardware space and a strong bandwidth connection running around the clock.

**4.9** Referring to the MOSCOW method, one thing that *must* be changed is the placement of the keyboard to be in the center of the screen. The current location is rather awkward and is not intuitive. Another must would be to make it so the game could be played both horizontally and vertically. One *should* would be to change the grayed-out keys to be either a slightly grayed green or a slightly grayed red. This gives a clearer notion of not only letters that have been attempted, but which ones were right or wrong without having to check the top of the screen. One c*ould* change is to include a button to restarted the game with an entirely new word. Another possible could is to have a button for a hint regarding the answer. A *won’t* could be something like adding a social aspect to the game. This social aspect would allow the user to add friends on the game and see what their high scores are. This adds a competitive edge to the game.