Teddy Chu, Justin Kyle Torres, Alejandro Zapata

Homework #1

Problem 1.1: What are the basic tasks that all software engineering projects must handle?

The tasks are: gathering, high-level design, low-level design, development, testing, development, testing, deployment, maintenance, and wrap-up.

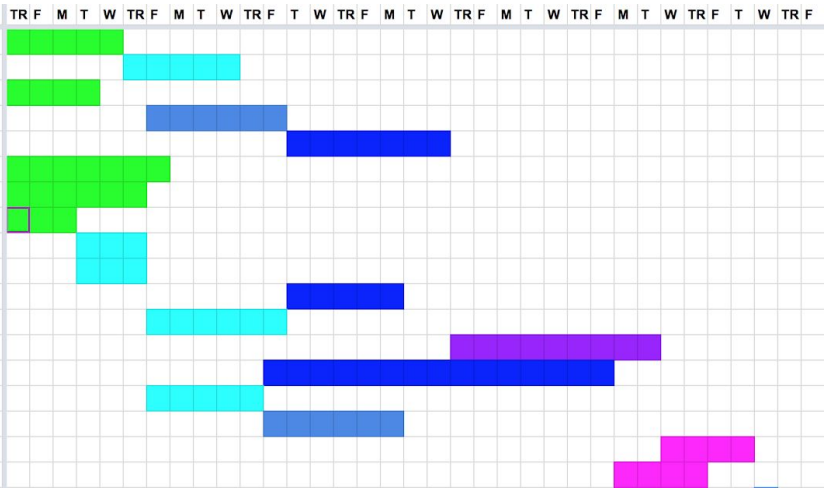
Problem 1.2: Give a one sentence description of each of the tasks you listed in Exercise 1.
Requirements gathering: You need to understand what your customer is, what team members will be doing, and what the goals of the project are.
High-level design: These describe the project's design without diving into the implementation.
Low-level design: This is a design of how each piece of the design should work.
Development: Programmers do their jobs of writing codes and iterating on code to improve where feasible.
Testing: Perform a multitude of checks or "tests" to verify intended functionality.
Deployment: Getting your software out to the users and dealing with any necessary logistics
Maintenance: Bugs will be found by users or developers, and work will need to be done to address those bugs post-deployment.
Wrap-up: Getting a post-mortem done so you can have take-away's to learn from the future.
Problem 2.5: What does JBGE stand for and what does it mean?
"Just barely good enough". It stands for the philosophy that you should have bare minimum documentation, as providing too much is a waste of time. However, the author notes having too many comments has never detracted from a project's success.

Problem 3.2:

R -> N -> O -> J -> H for 30 days.

Problem 3.4:

Task	Time	Predecessors
A. Robotic control module	5	—
B. Texture library	5	C
C. Texture editor	4	—
D. Character editor	6	A, G, I
E. Character animator	7	D
F. Artificial intelligence (for zombies)	7	—
G. Rendering engine	6	—
H. Humanoid base classes	3	—
I. Character classes	3	H
J. Zombie classes	3	H
K. Test environment	5	L
L. Test environment editor	6	C, G
M. Character library	9	B, E, I
N. Zombie library	15	B, J, O
O. Zombie editor	5	A, G, J
P. Zombie animator	6	O
Q. Character testing	4	K, M
R. Zombie testing	4	K, N



Problem 3.6: How can you handle these sorts of problems?

One important way to handle this problem is to expect the unexpected. You can factor in some estimate of time that might by lost into each task by cushioning the number higher to factor for things like this. Planning carefully, and also representing time off as a task can also help you schedule properly.

Problem 3.8:
The biggest mistake you can make is to ignore the problem and hope you can make up the time later. The second biggest mistake you can make is to pile extra developers on the task and assume they can reduce the time needed to finish it.

Problem 4.1: List five characteristics of good requirements.

The five requirements are: clear, unambiguous, consistent, prioritized, and verifiable.

Problem 4.3: List the audience-oriented categories for each requirement. Are there requirements in each category?

- a. Allow users to monitor uploads/downloads while away from the office.
    - nonfunctional
- b. Let the user specify website log-in parameters such as an Internet address, a port, a username, and a password.
    - User
    - functional
- c. Let the user specify upload/download parameters such a number of retries if there's a problem.
    - User
    - functional
- d. Let the user select an Internet location, a local file, and a time to perform the upload/download.
    - User
    - functional
- e. Let the user schedule uploads/downloads at any time.
    - User
    - functional
- f. Allow uploads/downloads to run at any time.
    - Nonfunctional requirements
- g. Make uploads/downloads transfer at least 8 Mbps.
    - Nonfunctional requirement
- h. Run uploads/downloads sequentially. Two cannot run at the same time.
    - Nonfunctional requirements
- i. If an upload/download is scheduled for a time when another is in progress, it waits until the other one finishes.
    - nonfunctional
- j. Perform schedule uploads/downloads.
    - User
- k. Keep a log of all attempted uploads/downloads and whether the succeeded.
    - nonfunctional
- l. Let the user empty the log.
    - functional

- m. Display reports of upload/download attempts.
    - functional
- n. Let the user view the log reports on a remote device such as a phone.
    - User requirement
- o. Send an e-mail to an administrator if an upload/download fails more than its maximum retry number of times.
    - nonfunctional
- p. Send a text message to an administrator if an upload/download fails more than its maximum retry number of times.
    - Nonfunctional

Implementation requirements were not here. This is because there were no external temporary features needed to get the software working. There were also no business goals, as there were no high-level descriptions that described any goals.

Problem 4.9:

Must:

- visually penalize users for selecting incorrect letters and also reward them for selecting right letters.
- Gray out letters that have already been used.
- All words in dictionary must be valid and correct.

Should:

- move the new game button to only appear when you win or lose.
- Have large enough keyboard

Could:

- Change shape/indicator for guessed words.
- Not use stock photo of skeleton.

Won't:

- Won't include a picture of Mr. Bones because he is taking up valuable space that the keyboard would have. This makes it much easier for the user to use the application.