HW1

Nikhil Ram (nikhilram@vt.edu)

1. From the software engineering processes discussed in class Tuesday, select which one would be the best in the scenarios below. To get full credit, you must only answer one process for each scenario and provide a reasonable justification for why you selected the specific process:

a) You are developing a software for a space mission to Pluto, the largest dwarf planet in the solar system.
Answer. Waterfall model (Plan-driven Model). For this critical and one-time mission software I would like to do it right the first time, all the requirements must be carefully figured out and there should not be any need to go back after implementation or at any other stage in the software life cycle.

b) As a programmer in the 1990s, you are developing a web application.
Answer. Code-and-Fix (Plan-driven model). If I am developing a full-stack application on my own I would prefer code and fix approach because all the requirements would be known and I can keep fixing the bugs as I implement the tasks. Especially in the 1990s when there was just HTML and mainly static websites were made.

c) You are working with other graduate students in CS-5704 on the team project for this class.
Answer. V-model or Waterfall model (Plan driven model). Working on a close project where all requirements can be well stated before the design and implementation is perfect for Plan driven models.

d) Your client keeps requesting to remove several high-priority features added to the product.
Answer. Agile development process. This model allows making rapid and flexible changes in the software easily.

2. In your own words, describe the difference between plan-driven and iterative process models.
Answer. Plan-driven models are generally adapted when software doesn't require any rapid changes and can be completed under a deadline. Iterative process models allow multiple iterations for the development there is a scope for changes in the subsequent iterations.

3. Compare and contrast the incremental and prototyping software process models.
Answer. Incremental needs clear requirement specification at the initial level but in the prototyping model, later on, new requirements could be accommodated.
Incremental models do not take user feedback while prototype models take user feedback into consideration.

4. The Chaos software development model, created by someone only known as L. B. S. Racoon, is based on strategies for the two-person games Chess and Go and defined in it's simplest form as "always resolve the most important issue first". Briefly describe the

advantages and disadvantages of this process based what you know about software processes.
Answer. The advantages of this strategy are high priority issues get resolved faster, however, any old non-priority issue might not get resolved, ideally if issues are resolved in first-in first-serve manner resolve time could be lessened.
Product issues may be reported by multiple customers since this strategy might result in more waiting time for some issues it may not satisfy customers.

5. While things have improved over the years, there are still many problems in software engineering today. What are some of the main challenges in SE that you have experienced or have knowledge of? Briefly describe at least two challenges and why they are important.

Answer. Though there are models for satisfying different software development needs, in practicality no software models truly help in meeting all objectives ofter times they are used in combinations and/or adding a few unconventional processes. An example though as a software developer I practiced agile methodology but while working on a sub-feature I adopted plan-driven models.
I feel software engineering still has a long way to go in evaluating the potential of a programmer for his ability to resolve issues in reality programmers often have to learn and use google or stack overflow to resolve the issues, this increases the resolution time overall however I don't think quality code is compromised this way.
The use of kanban board and such software is sometimes though useful but takes a lot of time we should optimize the time taken for those practices.

6. Do you or have you ever (i.e. while working in industry) considered yourself a to be a "software engineer"? Why or why not?
Answer. I worked in the startup for 3 years starting as a Software engineer working on bugs and features to being the software lead managing 17 developers handling to them their tasks taking scrum, and designing the system architecture of Software. Yes, consider myself a software developer.