Q1 Software Development Models

12 Points

Your company needs to install a well-known payment system at a big financial company within a week. And your company has done many such installs on other big financial companies before so basically your company is an expert in installing this particular payment system in big financial companies. One thing to notice is that you probably won't have enough time for testing because of limited time. Another thing is that customers are too busy to discuss all requirements so your team is still unclear about some requirements.

In this case, what development process would you choose? Make sure to describe where it lies in terms of *predictive vs. adaptive* as well as *incremental vs. iterative vs. Both vs. None*. Motivate your answer and explain why that particular process is a good choice for this problem.

Like in all reality, no one development process will be ready to use as is. Some modification will be necessary!

Select a base development model to start with. From there, explain what modifications you would make in order to fit the problem statement.

The base development model I would choose is Sashimi model. The basic mechanism of Sashimi is that a phase can start before the previous phase ends. Similar to waterfall, the team has experience building similar software.
In terms of predictive or adaptive, Sashimi is almost completely predictive but maybe a little bit adaptive. Regarding incremental vs. iterative vs. Both vs. None, Sashimi is incremental.

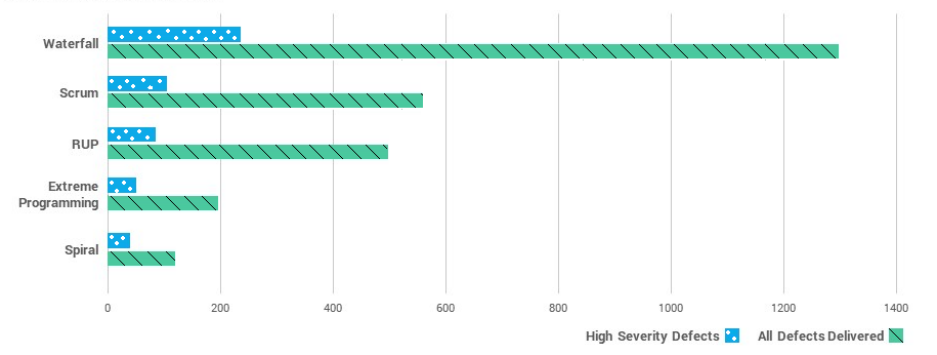
The reason why I choose Sashimi as the basic development model are:
1. Due to the limited time, there might not be enough time for testing. And Sashimi can shorten the development time and people can start working without waiting for the previous phases to finish, which is efficient and time-saving.
2. The development model we choose should be predictive as well as a little bit adaptive. The reasons why it needs to be predictive is that the company is an expert in this kind of project and it has done many such installs before, so it can predict what the outcome should be like and how to implement the project.
The reasons why it also needs to be a little bit adaptive is that the customers doesn't provide a detailed requirements and our team is still unclear about some requirements. So we need to do some adjustment based on the feedbacks during the development process.
3. It should be incremental, because (1) we have implemented similar projects before, so we have a clear idea of how the project should work; (2) since we do not have detailed requirements in the beginning of the process, some adjustments are needed, it would be much less work building the program based on the top of the existing system instead of total replacement.

The modifications I would make include:
1. We will start testing early in the development process to reduce the possibility that we have utterly no time to do the testing.
2. The development model will not only be predictive, but also adaptive (not just "a little bit", might be a lot), based on the analysis above. Because it is very important to change corresponding to the feedbacks and changes of requirements since we do not have a clear requirement in the beginning.

Q2 Defects in Software Design Lifecycles

12 Points

The following is a chart that depicts the number of defects delivered for a product using different software design lifecycles:



For each of the software design lifecycles, explain why and what parts of the lifecycle lead to the results seen in the chart.

Note: There should be an explanation for Waterfall, Scrum, Rational Unified Process (RUP), Extreme Programming, and Spiral.

Waterfall:
Waterfall has the largest number of defects delivered due to it high cost of change. This high cost of change happens at every phrase of the development process. And it is getting higher and higher as time elapses. For example, during verification, pieces don't work together and it leads to an expensive fix. One of the assumptions of Waterfall method is that the requirement engineering is perfect, but it is actually extremely difficult to accomplish. So it is highly possible that we have to go back to the previous phase and work again. Another example is that, after the deployment phase or on the maintenance phase, the users of clients argue that the system doesn't meet the initial requirements, so the team has to go all the way back to the first phase which is to change and decide the requirements again, and then work on the design, implementation, testing, and deployment again, which costs a huge number of time and effort and money. Additionally, potential market might also shift which also leads t o an expensive fix.

Scrum:
Scrum is the incremental and iterative framework for developing, delivering, and sustaining products. It assume that customers will change what is wanted. Scrum has a lower defect rate than Waterfall because it has feedbacks and can do adjustments in the middle of the development process. However, Scrum is still somehow with high defect rates because quality is hard to ensure until the team goes through the testing process.

Rational Unified Process (RUP):
Rational Unified Process (RUP) is of iterative development. The business value is delivered incrementally in time-boxed cross-discipline iterations. RUP has a relatively low defect rate because it is used by people who work in IBM and they are good developers who are familiar with RUP.

Extreme Programming:
Extreme Programming is a type of agile software development. Developers work in pairs and they do unit tests of all code and acceptance testing of all requirements. Because it is test-driven development, it has low defect rates.

Spiral:
Spiral has low defect rates because of its risk analysis.

Q3 Favorite Ice Cream Flavor

1 Point

What's your favorite flavor of ice cream?

Coffee

Quiz 04

GRADED

STUDENT
Ruijun Ni

TOTAL POINTS
20.5 / 25 pts

QUESTION 1
Software Development Models
12 / 12 pts

QUESTION 2
Defects in Software Design Lifecycles
7.5 / 12 pts

+ 2.4 pts Waterfall explanation includes why and what parts of the methodology leads to the results shown in the chart

+ 2.4 pts Scrum explanation includes why and what parts of the methodology leads to the results shown in the chart

+ 2.4 pts RUP explanation includes why and what parts of the methodology leads to the results shown in the chart

✔ + 2.4 pts Extreme Programming explanation includes why and what parts of the methodology leads to the results shown in the chart

+ 2.4 pts Spiral explanation includes why and what parts of the methodology leads to the results shown in the chart

💬 + 5.1 pts Waterfall -1.0: does not address the method's style of testing and why that leads to the results show in the chart.

Scrum -1.5: does not address the method's ability to dynamically change requirements, style of testing, and why these leads to the results show in the chart.

RUP -1.0: does not address the method's style of testing and why that leads to the results show in the chart.

Spiral -1.0: does not address the method's style of testing and why that leads to the results show in the chart.

QUESTION 3
Favorite Ice Cream Flavor
1 / 1 pt