CS 471/571 Assignment #2

Full Name: Milson Munakami
Student ID: 114012811

1. (Chapters 1 and 2) Instead of implementing component c9 of a software product, the developers decide to buy a COTS (Commercial Off-The-Shelf) component with the same specification as c9.

(a) What are the advantages and disadvantages of this approach if the COTS component is closed-source? **(10 points)**

Answer:

Some of the advantages and disadvantages of using COTS component which is closed-source are as follows:

Advantages

1. It provides a good documentation with an expectation that it'll work in the way promised. If it doesn't work in that way, you can take legal action against the provider.
2. Closed source software usually equates to better security and support.
3. Unlike open source systems, developers don't have to spend as much time securing code.
4. More secure than open source Commercial off the shelf software products whose code and internal development architecture is never disclosed.
5. Support is readily available from the software provider with nominal cost. This is a convenience, because it cuts down on the development time and cost.
6. Can be cheaper and more reliable than custom software because it's requirements are determined largely by market research.
7. Often used by the business world instead of custom software (cheaper and more reliable)
8. Sold on the open market so readily available.

Disadvantages

1. A smaller community means less experience and collective knowledge.
2. Requirements determined largely by market research so may not meet the exact requirements of a particular user.
3. Software providers carry the never-ending burden of upgrading their products. This means they need to invest heavily in keeping up with, or ahead of, the industry. Naturally, these costs are passed onto the user.
4. While premature software distribution can mean speedy testing, and therefore fast bug-fixing, which give a way to vulnerability risks to the software.

5. The most common method of fixing bugs is for the software producer to supply a patch. Even though, system remain vulnerable even after a bug is found, users may be unaware and unable to apply those released patches.
6. Since closed source companies are the only organizations with the rights to build and develop their products so it has risk of vulnerability due to limited versioning of software that can be easy target of hacking and security exploitations.
7. It may include hidden cost in terms of Service/ Support terms.

(b) What are the advantages and disadvantages of this approach if the COTS is open-source? **(10 points)**

Answer:

Some of the pros and cons of using COTS which is open source are as described below:

<u>Pros:</u>

1. Open source means it is community based development and there are a lot of people working on the same software. Plenty of individuals are making sure the code is solid and bug-free and that the software is easy to use and understand.
2. Documentation is usually easy to find, and there are plenty of people out there writting about "how-to use".
3. It provides continuous and regular updates that are continually improving the product in real-time.
4. Cost effective because it is open source and freely available to use and modify.
5. It is easy to manage as we can download and install any numbers of it and in any system no need to track the count and monitor the use.
6. Company Independence software development process

<u>Cons:</u>

1. Open source code, which creates a higher risk for hacking and vulnerability.
2. Even though it is initially free, it's Not Entirely Free. Open source software can involve unanticipated implementation, administration, training, service and support costs.
3. Individual change or customization needs the development time and cost.
4. Learning Curve of such software is high. So you may need to hire an expert having domain knowledge of that software to train how to use.
5. Due to regular updates users may get confuse to which version to use because parallel development efforts are ongoing all the time.
6. Use on Your Own risk. Unlike with a commercial customized product, no one in the open source community is obligated to help you or answer any of your questions.

2. (Chapter 2) What is the connection between the waterfall model and the iterative and incremental model? **(20 points)**

Answer:

The waterfall model is the classical model of software engineering paradigm. This model emphasizes planning in early stages, it ensures design flaws if any before they develop. In addition, it is documentation-driven so its intensive document and planning make it work well for projects in which quality control is a major concern. But in real life we cannot always guarantee the initial analysis phase will give all features and needs of a software so instead the analysis and iteration is spread out over the whole software life cycle in case of the iterative and incremental model. SO according to the new change or requirements the software is adapted and modified to meet the final requirements. Unlike the waterfall model, requirements and design phase are not single phase instead there are multiple instances of these phases. Thus it removes the sequential phases of waterfall model. It involves five core workflows or activities that are repeated over the entire life cycle. Whole project can be considered as a set of many mini projects i.e. increments. Each mini project can have its own iteration and incrementation. These part of iteration can be viewed as a small but complete waterfall life-cycle model which are completely tested and corrected. Thus using the iterative and incremental model, the risk can be mitigated early stage rather like at the end in waterfall model using bottom-up approach.

3. (Chapter 2) Compare Scrum with the iterative and incremental model. **(20 points)**

Answer:

Scrum is a part of Agile Software development process to develop any complex systems and products. Scrum is about work management and team dynamics which gives the artifacts as the product and sprint backlogs to accomplish the development. Scrum model uses Analysis, Design, Code, Testing and delivery of a small functional pieces in iterations with short duration which involves the Sprint cycles.

The iterative and incremental approach is less result oriented than Scrum model. Scrum is more dynamic and full of action as it involves feedback and changes.

In Scrum the feedback involves at the end of every Sprint whereas in the iterative and incremental model it occurs at the end of every iteration.

In case of Scrum process, there involves daily standup meeting and a dedicated Scrum master to conduct and facilitate the planning and estimation whereas in case of iterative and incremental approach all such decisions are made by project manager.

4. (Chapter 2) List three risks involved in developing the team project of this class and discuss how you would attempt to mitigate each risk. **(20 points)**

Answer:

Risk	*Mitigation*
Team member is less experienced in Java programming, IDE and Tools use – Technical Risks	Proper training and providing adequate learning resources to be familiar and learn fast and easily
Team member can leave the project before it is completed	Need to motivate and encourage each members to keep up the high spirit and focus to complete the tasks on given time.
Risk of not getting requirements correctly	Proper planning and analysis of requirements and making a proper design before starting development – architectural design and functional design of the software

5. (Chapter 3) Consider the requirements workflow and the analysis workflow. Would it make more sense to combine these two activities into one workflow than to treat them separately? **(20 points)**

Answer:

Since in both of these workflow the similar kind of activities are involved like collecting the all possible client's needs, concept, requirements and specifications and refining them. I think it would make more sense if these similar kind of activities are combined together as a core activities rather than as separate activities. As they produce similar kind of result (artifacts). The outcome of requirements workflow is client based imprecise user requirements usually written in natural language. On the other hand, the outcome of Analysis workflow is more refinement of the collected requirements from previous workflow to make them more precise and complete so that they can be started over next workflow. Only for refinement I think to dedicated workflow is not appropriate which can be easily done during earlier requirement activities so that it can save time as well as money. Hence, there is no need to separate these two similar core workflows which can only create confusion among stakeholder to divide the work.