Sangey Lama

CS 370

Homework 1

1. The fundamental software engineering principles of process, dependability, requirements management, and reuse are relevant to all software systems types. The process within software engineering are all the important steps taken to produce an ideal software system for the users. The software process models are what makes up all the processes. A software product with a better process approach results in lesser software failures and can live to up to user requirements. The general process models outline the arrangement of software processes. The types of general models that can make a proper software system are Waterfall model, Incremental development, and integration and configuration. These process models are an important tool for the community to work and develop software together which help with results of software that are quality controlled and dependable. Also, a good process approach create communication with users about their system needs, enhance software quality, and avoid poor approach on requirements and design. Dependability in software system is capability to provide the services and stay stable over time. System dependability is significant because of unstable software systems that can lead to financial loss, loss of confidentiality, and harm to the users. The dependability of software system can also alter the user’s trust in it, and it is important that it is reliable, safe, and secure. Safety and security have been such a bigger part of dependability since we have a competitive environment that compete to serve the best product but may forget to follow the standards to make sure their software systems are safe and secure. All the errors in systems all comes down to human mistakes so a repeatable process can be necessary to lower the bugs. And performing a confirmation after each repeatable stage can make a software system very dependable. Another dependable attribute is diversity and redundancy which is essential that tolerate the faults by providing the same functionality in different ways so that are no system failures. Requirement management is the task of documenting all the important information that is required to exhibit the system’s purpose and effectively provide the services. Also, it gives a clear understanding to the user about what is required and helps to not waste time, assets and effort. The two requirements that optimize the services are functional and non-functional. Functional requirements state that system must lay out the services it will provide for the software to be functional. Non-functional requirements do not directly regard the services towards users, but it concerns with efficiency in software system. Performance, scalability and usability are the attributes of non-function requirement that shapes the system to be user friendly, straightforward and acceptable. Requirement management is a continuous procedure and constant communication between the developer team and users occur to adjust to the changes of requirement. Reuse in software is the strategy of reusing existing software within the software development process. The assets that are reused can be classes, libraries, software components, and other complete application systems. Reusability is relevant to software system due to the output of increasing productivity, lowering risk of starting new projects and lower costs. Since reusing software components helps with reducing costs and time, reusability is getting popular with a lot of software company. Although there are some things that hinder the approach of reuse, especially with flexibility and complexity of software components, the disadvantages have been somewhat tackled with well secured techniques to get past the downsides. Software reusability has really changed the way the software is developed, and it is still evolving with help of innovation to stay more applicable to all software systems types.

2.A The most appropriate generic software process model for a system to control antilock braking in a car is the waterfall model. In the waterfall model approach, the project goes through linear sequential phases and it is appropriate for critical system. Antilock braking system belong in the safety critical system where the system requires full safety and reliability analysis of the software specification and design. The antilock braking is designed to save the driver from accidents so it’s important that the system is well tested throughout the production so there are no faults in the system. The analysis and the design requirements must progress without any failures before the actual system is implemented for the correct performance of the anti-lock braking. Therefore, safety related issue in the specification and design can be very costly at the execution phase if not properly investigated in the beginning phase. It is definite that this system is a plan-driven process as it must go through long term planning due to safety purposes.

2.B The waterfall model is the most appropriate generic process model for a system for a university accounting system that replaces an existing system. In a plan driven process, the success of the development is dependent on executing the prerequisite closely and complete requirements are clearly documented in the beginning of the phase. The waterfall model can go well for this circumstance because the prerequisite of this accounting system can be noticeably marked from the existing system beforehand and can lead to better development from recognizing flaws from previous one. The requirements are very well investigated, understandable and stable. Also, there are communications occurring back and forth between the developer and the user for the requirements at early stage so the developer can utilize information on the software principles applied on the existing system.

2.C Agile incremental development is what I believe is the most appropriate generic software process model for an interactive travel planning system that helps user plans journeys with lowest environmental impact. In this system, the software evolves through increments, which is a cyclic process that improves the system. Since it’s an incremental progress, the developers and the user understand exactly what is thorough or not. Agile incremental development favor systems that regularly have requirement changes from the user. The users also expect quick delivery from the developers after the changes have been implemented. Customers also take part in the process of the development and they help with all the important requirements to better the software. The customers who travel come with numerous travel experiences and their preferences on the software features can change regularly. Therefore, agile method incorporates requirements elicitation with development. Developers collect customers travel experience stories so they can figure out the best features that can satisfy user’s requirements with beneficial functionality. Also, the direct contact between the team and client can make the team very motivated and self-organized, there is a possibility of higher productivity which can force a better outcome on the projects. Early feedbacks from the user keep coming so there are less risks and it can keep the user’s satisfied because of continuous changes they can provide with lower costs.