Timothy Rine

CSC 3380

2/3/17

Mission-Critical Project Development

The first step in determining what type of software process best suits a mission-critical project is identifying and categorizing the project itself. There are shortcomings and advantages to each type of software process model which means that choosing the right process model can be very important. Several factors need to be preemptively analyzed to make the best choice: the scope of the project, the size of the development team, any deadlines the project may have (which includes prototype builds), and all potential risk involved in the project.

A safe choice for a mission-critical software process model is the Spiral Model. The Spiral Model provides for a lot of planning time which is beneficial when determining risks and requirements. Part of the first quadrant of the Spiral Model is gathering information for risk assessment. This can be very beneficial to large projects because quantifying potential risk factors before they become problems can save the project and company involved a substantial amount of time and money. User surveys are another key facet of the Spiral Model. With the collected data from user surveys the development team can easily alter their code to implement ideas the direct consumers will prefer. This code change is a built-in advantage of the Spiral Model- rework is easily done at almost any point in the project. Due to the extensive analysis and planning, there is a relatively low chance of making mistakes. The waterfall model is effective in providing the project team with a specific and detailed outline for software development. Another advantage of the Spiral Model is the flexibility of maintenance. Maintenance is an extremely important step is maintaining a software product and this software process allows the development team the functionality of being able to do so.

A potential disadvantage of choosing the Spiral Model, however, is that the process itself can be tedious and slow compared to an agile software process model. Although agile software process models are not plan-driven like traditional models, they adapt very well to changing project requirements. Code also tends to be clean and easy to modify. Through continuous testing the project team can verify the integrity of the software being developed. Due to the nature of agile processes, there is a substantial amount of uncertainty involved when using this type of process model. For that reason, it is not advisable to use an agile process model for a mission-critical project.

If given the choice of what type of software process model to use for a mission-critical project, I would choose the Spiral Model. Although a lengthy software process model, the extensive time spent on risk analysis and determining requirements is valuable over the course of the projects lifetime. For a mission-critical project it is of the utmost importance that the software is developed correctly and works functionally. To achieve the optimal product for a mission-critical software project, I think that the Spiral Model would yield the best results.