David Lambertson

**A Software Crisis Opinion**

1. **A Brief Explanation of Software Engineering**Software Engineering,defined by Merriam-Webster is “a branch of computer science that deals with the design, implementation, and maintenance of complex computer programs.” Through software engineering software engineers are able to choose and know the model in which they should design and develop software. The different models of waterfall, evolutionary, and the others which the software development profession has, affects the methodologies of what software engineers do in developing software.
2. **A Brief Explanation of the Software Crisis**Software crises are common even though people try their hardest to avoid them. A software crisis is when an error, mostly critical to success or failure, is missed or looked over. Due to this, the system is ultimately set up for failure from the beginning. We see this happen in most of the large to huge software projects people and companies tackle. According to Dinesh Thakur “the major causes of software crisis are the problems associated with poor quality software such as malfunctioning of software systems, inefficient development of software, and the most important, dissatisfaction amongst the users of the software.” When we think of software crises in these terms, we are able to come up with different ways to diffuse the probability of a software crisis happening.
3. **Yes or No Can software engineering reduce the software crisis to insignificance?**No.
4. **Justification of my response**I chose no because while software engineering can provide cushions and make it less likely, if a software crisis happens, it will still be significant. Even if it doesn’t happen, the likely of it happening cannot be reduced to insignificance. I say this because stakeholders, software engineers and project managers should all always have risk in mind. They should plan for cases that could come up. They should also not pass over any risk that an engineer or developer bring forward. Many of the software crises that we hear of today could have been avoided if only an engineer or developer had been listened to. Looking at sections one and two, we see that software engineering can improve the devestation or likeliness of a software crisis to be low, but it still would be significant because of the damage it can cause.

Sources:

<http://www.merriam-webster.com/dictionary/software%20engineering>

<http://ecomputernotes.com/software-engineering/software-crisis>