ECE 321: Software Requirements Engineering Assignment 1

Arun Woosaree

XXXXXXX

September 18, 2018

1 "Passing the word" Review

1.1 Main Contributions

The author mainly goes over his recommendations for best practice in the industry, based on his past experiences. He begins with describing the characteristics of an ideal manual in his eyes. The article also gives various tips for writing manuals, such as making the manual's language more consistent. The author also describes the importance of the formal definition. He also mentions the usefulness of having informal documentation, and how it might be created. Many other tips are outlined in the article as well. The author explains methods used, and the structure of meetings that he found to work most effectively in his experience. The author also outlines the importance of having multiple implementations in a product's development cylce, and how it relates to the manual. Finally, the author goes over the importance of product testing, which brings everything mentioned in the article together and serves as a final checkpoint before the product is ready for the customer.

1.2 Criticisms

The author makes mostly good points in this article. Naturally, there are some points I must disagree with, the first being the author's opinion that the manual be written by only one or two men. This approach is simply not

scalable, and is impractical for large projects. Furthermore, having multiple people collaborate on the manual provides the opportunity for people with multiple perspectives to review and edit the document, so that inaccuracies can be found and fixed quicker. Having multple perspectives also makes it easier to identify portions of the document which may need to be reworded to make it easier or more interesting to read. Additionally, the style in which one person writes one week might be different in the next week, based on that person's emotions. With multiple writers, each writer can keep a lookout on each other to make sure the language is consistent. Next, I must disagree with the author's suggestions regarding meetings. While his suggestions may have been effective years ago, nowadays, the agile workflow seems more attractive. Meetings are held every day between the people developing the product, but they are short. This way, problems can be found quickly and it allows the team to prioritize issues effectively, in addition to making the right people aware of issues that may concern them. My suggestion for a second type of meeting would be before any major product deployment, and the frequency and length of these meetings should depend on the scale of the project. If deployments are happening frequently, this might be a monthly affair, or an annual event like the author suggested. This meeting should consist of the people developing the product as before, and also any managers, representatives, product owners, stakeholders, etc.

1.3 Answers to questions

1.3.1 Good manual style

In the author's view, a well-written manual must use precise, consistent language, have quantized changes, and it should also describe every detail the user sees, omitting the details that the user does not see. The author mentions that this can be done in a formal and/or inform al style. In my opinion, an informal style is more desireable, since quick ammendments and changes can be made to the document as necessary, and using informal language makes it easier and less daunting to read overall for the average person.

1.3.2 Effective meetings

The first type of meeting the author claims is effective is a weekly half-day conference led by the chief architect, as well as all the architects, official

representatives of the hardware and software implementers, and market planners. The second type of meeting is what the author refers to as an annual "supreme court session", which lasts two weeks. In this type of meeting, the project manager leads, with members from the architecture group, the programmers' and implementers' architectural representatives, and the managers of programming, marketing, and im plementation efforts joining.

1.3.3 Independent product-testing organization

In the author's view, an independent product-testing organization acts as a "surrogate customer", which allows for early detection of bugs and departure from the design. I mostly agree with this viewpoint, since programmers are human, and are prone to making mistakes like everyone else. Although it might slow development time, having a dedicated team for finding these flaws before deployment is important, so that the customer has a better overall experience.

2 "Requirements Engineering: The State of the Practice Review"

2.1 Main Contributions

The purpose of this paper is to bring to light the most common practices and techniques used in the software development industry. According to the authors, before this paper, most people including themselves were referring to sources without credible facts or statistics to back their claims. This paper aims to fix that. The authors have taken the time to survey a group of people who they felt would be representative of the software development industry, and to compile the results to make a few graphs. The paper seems to accomplish its goal of providing statistics and facts, as well as analysis of the data in a format that is credible and can be cited by others who need a source to back up their claims about common practices in the software industry.

2.2 Criticisms

wtf this article is mostly fact you can't really criticize facts

2.3 Answers to questions

2.3.1 Most frequently used lifecycle model

According to the survey, the Incremental lifecycle model is most popular for projects which last over two years, while the Waterfall model is more widely used for shorter projects. The Incremental model is likey most popular for longer projects, because the project may evolve and necessitate changes as time passes, or as the stakeholders may change preferences in that time. On the other hand, shorter projects are less likely to undergo much change due to their shorter lifecycle, so in this case, the waterfall model is ideal because the end result is agreed upon before the project begins, which can be more efficient compared to other models.

2.3.2 Prototyping

Of the respondents in the survey, about 60?% did some sort of prototyping. The most frequent type of prototyping is with the user interface. This is likely because the user interface is what's visible in the end product, and is what the stakeholders are most likely concerned with, as opposed to the technical implementation.

2.3.3 Elicitation of users' requirements

According to the survey, the top three methods for elicitation of users' requirements are: Scenarios & use cases, focus groups, and informal modelling. Given the option, I would probably choose to the use cases and scenarios approach for requirements elicitation. Use cases and scenarios are an attractive option, because the focus is on the end goal, which does not need to be technical. This lack of technicality can simplify communications with stakeholders, since the stakeholder does not need to know the technical details of the project, just how it should work. This can help mitigate problems where the end product is drastically different from what the stakeholder(s) imagined, which explains why it's the most widely used technique.

2.3.4 Is the informal modelling of the requirements more popular than the formal representations

According to the survey, formal models are rarely used, and informal models are much more popular, which is likely because informal models make it

easier and more natural to work with. This may end up in a higher quality end product, as less time is spent on obsessing over unessecary details which don't affect the end product.

2.3.5 According to the survey, were longer projects less often finished on time and within the budget?

Yes.