# Fault Repairs

The first area we are looking at is fault repairs. We have two main systems that will be sending in faults from users. The second one is the user’s comments. Google play has a built-in system for users to leave comments and ratings. We will be watching these comments and looking for possible faults there. Our first line of bug detection is the “Report Bug” ability that we built into the application. Under certain conditions the user will be directed to a “Bug Report” form. The ways that a user could get there includes: the bottom default option of a “Try Catch” were the coding does not know what to do, unexpected shutting down of the program, the program caught in an endless loop, and a user’s option to report an issue. The bug report itself will send us several different pieces of information. These will include: the error code ( when one is generated ), the name of the map that is being played, the starting condition of the map, the current condition of the map, a record of major events in the game ( major events would be placing or destruction of buildings and a resource being depleted in a square, or growing to a square ), and the last X numbers moves the player has made ( X for example could be 25 ), and a comment left by the users describing the “Bug” / fault. With this information we should be able to recreate the conditions that caused the fault and identify its’ cause. This fault will be added to the list of changes that will be addresses in the next iteration of the software.

This is one of the reasons for, and advantages of, our choice of using an incremental method of development using agile test-driven development (TDD). The fault repairs after the application is released is essentially a continuation of our software development process. It adds things to do for the next iteration, namely repair this fault, in addition to the changes already planned like additions to the game and improving the existing code. Our initial estimate for fault repair maintenance costs is about 24%, or about ¼ of the total maintenance costs (Davidsen and Krogstie 2010). We will be able to refine our estimate for each application as time goes on. Fault repair and environmental adaption make up almost half the total maintenance. Therefore, being a 2 person company, these two maintenance responsibilities will be assigned to one person and the other task of functionally addition or modification will be the other persons maintenance responsibility.

# Environmental Adaption

Environmental adaption is the next area of maintenance. With out project being a mobile application, a game to be precise, it is in no way a critical system. It is for entertainment purposes. At some point the number of players and the income it generates will make it obsolete and no longer in need of any maintenance. While the game is still generating good income there are two things that will cause the need for environmental adaption. These would be changes in the operating system and changes the in mobile devises that is it being played on. Mobile devises and their operating systems are evolving quickly. Firstly, lets consider operating systems. For the most part they are backward compatible, but there are times when a command becomes depredated, at which point the code will need rewriting. Another possibility is the inclusion of a new command that does the same task better, this would create the need for refactoring. Looking at changes in mobile devises, size and device resolution are always changing. This can cause display issues and will be a constant cause for adaption.

The other side of the display causing the need for adaption is new methods for input to the mobile device. This could create the need for major revisions of the game and create new ideas for games that could be created. Generally environmental adaption is estimated at 19% of maintenance costs or about 1/5 of the total (Davidsen and Krogstie 2010).

.

# References

Sommerville, I. (2016). *Software engineering*. Harlow, England; New York: Addison-Wesley, 2016.

Davidsen, M.G., and Krogstie J. 2010 “A Longitudinal Study of Development and Maintenance.” *Information and software Technology* 52 (7): 707 - 719