**What are some of the main observations he makes about how we should design?**

The author discusses a software design approach known as Extreme Programming, or XP design, that addresses issues with predecessor software designs. XP is a hybrid of Planned Design and Evolutionary Design with a heavier emphasis on the latter. The principles of this design approach include a non-detailed sketch of how the program will be designed with the purpose that it gives a direction of how the program should be written out. Another principle of XP is simplicity of code so that it’s readable and easy to maintain, and only features that are needed at that moment should be implemented ie. (YAGNI) there shouldn’t be any attempt to implement flexibility in the structure of code until it’s absolutely necessary to do so.

Along with those main principles, XP consists of the core practices of Testing, Continuous Integration, and refactoring. Thorough testing ensures that the program is working safely, and Continuous Integration ensures that the entire team is working efficiently on the present code. The enabling of these two practices flattens the ‘change curve’, or the understanding that a change in analysis further down the line will exponentially rise, and this makes Evolutionary Design work. As for refactoring, the author says that it’s so effective that he was motivated to “write a whole book about it.”

**Relate these recommendations to your personal experiences, where it may have**

**worked or may not have worked for you.**

I found that when it comes to coding and designing a solution, I think in a similar fashion of the author’s. When starting my design, I create a high level view of classes and their relationships, and that’s what I start coding off of, but remain courageous enough to amend my original design if need be. I find that a high level diagram is makes solution of tedious code much more easier to comprehend act upon.

I completely agree that the will to design has a direct relationship with the design quality. Without a strong will, every piece of code becomes ad-hoc.