Brady Field

The Mythical Man-Month

**Chunks of Wisdom**

*Chapter 1*

Wisdom – Programming is more rewarding than it is frustrating

Meaning – As outlined in the chapter, software development has its ups and downs, but the benefits far outweigh the drawbacks.

Application – I remember an experience of when I was in CS124. I was working on the budget program for the first unit. I remember feeling elated when I compiled and ran the finished product. I have experienced many frustrations since then in development, but I always get that same great feeling every time I compile and run my code.

*Chapter 2*

Wisdom – Scheduling issues are the main causes of problems.

Meaning – Various scheduling estimation factors combine to give stress to developers and delay delivery of the product. This is by far the biggest culprit in software development problems.

Application – Beginning CS students many times underestimate how long a given project will take. This leads to procrastination, which leads to headaches, unforeseen problems, and late assignments.

*Chapter 3*

Wisdom – Small teams don’t work on big projects

Meaning – While small teams are many times very productive in their tasks, there is too much ground to cover for a small team to realistically compete with a larger team on large projects. There are several methods outlined in this chapter equipped to deal with this dilemma.

Application – When it comes to designing an interface, one must cover a lot of ground. Even a small team of 2 to 4 people may not be able to do a proper breadth first search to come up with an optimal design. A large team is needed, but should be managed or broken up so as not to counteract the productivity of small teams.

*Chapter 11*

Wisdom – The first attempt will never be the best.

Meaning – Plan to throwaway the system. The first run-through will always be barely functional compared to later revisions.

Application – When I start a new project, I first come up with a simple design and implement it. I then go over it again, redesigning individual components, because I know that my first idea is never my best.

*Chapter 12*

Wisdom – Problems with tools slows down development time.

Meaning – When a team fails to achieve unity of tools on a project, or tools are poorly chosen, the productivity for a project suffers greatly.

Application – When working on a team web project, my team was plagued by compilation and execution issues for the first two weeks of the project. These issues arose due to a single member using a different IDE than the other team members. This had unforeseen side effects.

*Chapter 13*

Wisdom – Failing to plan is planning to fail.

Meaning – There is a common phrase among developers that states “Days of coding will save you hours of planning.” Designing beforehand will result in better designs and higher productivity.

Application – As an assistant in the Linux lab, if enjoy seeing CS students when they finally realize that the good old code and fix method isn’t going to cut it anymore.

*Chapter 14*

Wisdom – It’s the little things that will get you.

Meaning – Big fires are easy to put out because they are easily noticed and require a big effort to deal with. Little delays are harder to notice, and therefore deal with, but they all add up to cause really big issues that are difficult to fix.

Application – This happened to me in CS124 a lot. I would estimate how long it would take me to complete a project, and then I would end up taking much more time than initially anticipated. This was due to my lack of experience in predicting what little things could happen that I would have to deal with. With failure to plan beforehand, my projects often took much longer than initially predicted.

*Chapter 15*

Wisdom – Documentation is the lifeblood of a project’s code.

Meaning – Code is most often not meant to be exclusively used or maintained solely by the creator. Without good documentation, everyone suffers. The product will be harder for users to use, and developers will find the code harder to maintain, change, or update.

Application – I was a student in CS246 where I was t take over a project as part of a team. The previous team had plenty of code, and no documentation to go with it. Before my team could even start writing our own code, we lost a month in reading through and documenting existing code so that we could make changes and actually get what had already been written in code to work on a basic level. Had there been documentation, this time very well could not have been lost.

**Book as a Whole**

The name of the book takes itself from the second chapter to illustrate that the problems encountered therein, and thus in the workplace, are not the same as other work problems from other fields of study such as painting or manufacturing. Therefore, the solutions cannot be the same either.

The problems discussed are not technology dependent, or rather; they are not inherent in, nor solvable by, advances in technology. They also cannot be solved by adding more people to the project. They are abstract problems that require creative solutions. The problems are mostly the same today as they were when this book was first written. Therefore, not only are the problems contained therein still applicable today, but many of the suggested approaches contained in the book that attempt to address these problems are also still applicable.

The book revisits the theme that many of the roadblocks encountered are human based. These are things that we bring upon ourselves. Bad estimations, in time or budget, serve as prime examples to these. These are the same problems we encounter in software today despite our advances in technology. We still suffer from inaccurate estimation. We still have difficulty in budgeting. No amount of throwing more people at the project, or other perceived “bandage solutions” will make up for these shortcomings either. On the contrary, many times such shallow approaches will in fact hurt, rather than help, the problem.

Many of the suggested techniques, if not used directly, still serve as the foundation of what we use today for common mitigation practices. The problems encountered in software development are unique to this field of study. Nowhere else does one find a product with properties quite like software. “The magic of myth and legend has come true in our time. One types the correct incantation on a keyboard, and a display comes to life, showing things that never were nor could be.” [7 - 8]