Mythical Man Month

Patience: Good cooking takes time. If you are made to wait, it is to serve you better, and to please you. Don’t rush projects at the beginning.

Man-Month Correlation (Less is more): The more people you throw at a software project increases the schedule due to the greater complexity. The only time throwing more men at a project works is it the two tasks can be partitioned perfectly with no required communication between them.

Communication Complexity: The amount of people in a project increases the communication complexity proportionately. Two works cause one link, three workers requires three, four workers require 6, five workers require ten.

Scheduling Software: 1/3 planning, 1/6 coding, ¼ component testing, ¼ system testing.

Good Programmer: Ten times better than a bad programmer. No correlation between experience and effectiveness. Only work with good programmers and don’t assume the bad ones will get better with time.

Surgical Team: One team member does the cutting and the others give him every support that will enhance his effectiveness and productivity.

Support Areas: Sounding board, finances, working space, documentation, assistants, sharpening tools, content creation, tester, lawyer.

Integrity: Each person must sacrifice some of their own ideas for the integrity of the program.

Simplicity and Straightforwardness: Simplicity is not enough, programs must be very straightforward and come from one or a few minds.

Form is Liberating: The worst projects are those whore budget was too great for the purpose to be served.

Design Phases: Architecture, Implementation, Realization, Repeat.

Manual: Describes everything that the user sees and nothing that the user does not see.

Telephone Log: Every question asked by anyone answered by the architect.

Communication: Informally with telephone and e-mail, regular project meetings, and workbook.

Thinker-Doers: Are the rarest.

Progress Interferers: Machine downtime, processing time, unrelated jobs, distractions, meetings, paperwork, financial troubles, personal issues, sickness, personal time.

Program Space: Helps dictate its cost.

Critical documents: Objectives, specifications/architecture/white papers, schedule, budget, org chart, space allocations, wireframe/storyboard.

Pilot Plant: An intermediary step that takes a big leap but before full scale is reached to test how well a program scales.

Planning for System Change: Modularization, sub routing, definitions, comments, documentation, milestones, sprints.

Program Maintenance: Change that repairs design defects.

Tools: A good workman is known by his tools. One toolmaker per team. Computer facility, operating system, language, utilities, debugging aids, test case generation.

Target v. Vehicle Machine: Target machine is machine for which software is being written. Vehicle machine is machine that provides services used in building the system.

Debugging: Hard and slow part of system.

Scaffolding: Programs for debugging but never intended to be in the final product.

Day by Day Slippage: Errors that occur daily and are hard to recognize but add up. Prevent compounding errors and Swiss cheese problem by making sure that you have redundant processes.

Imagery: What we do not understand we do not posses (Goethe).

Computer Program: Story written from a man to a machine.

Prose Description: Purpose, environment, domain and range, functions realized, input-outputs, operating instructions, user option, running time, accuracy and checking.

Realistic: Why lay a load on their backs which neither our ancestors nor we ourselves were able to carry (Acts 15: 10, TEV).

Self-Documenting: Incorporating documentation into the source programming.

Simplicity: God is not capricious nor arbitrary (Einstein). Everything has a utility. There must be a simple solution to every problem.

Accidental Complexity: The intermediary steps that you need to do to build a technology and solve a problem. As new technology evolves, these go away.

Mentorship: Provide mentors for software programmers.

Knowledge Spillover: Provide smart people the opportunity to interact with each other.

Productivity: Focus on quality and productivity will follow. Be careful setting deadlines or putting pressure on workers.

Language: People do not memorize anything but spelling. They learn syntax and semantics incrementally, in context by use. People group word composition rules by syntactic classes, not by compatible subsets of objects.

Private Feedback: Deal quietly and privately in suggestions. People are less likely to take your advice if it isn’t publicized that it cam from you. Make hidden suggestions.

Organization: Purpose is to reduce the amount of communication and coordination necessary.

Project Manager: Chief job is communicating, not decision making, only 20% of information needed should be outside their head.

Maintenance Cost: Affected by the number of users. More users more bugs. Fixing a bug has a 20-50% chance of finding another bug. Must retest whole system. Repairs destroy structure.

Murphy’s Law: Many failures concern exactly those aspects that were never quite specified.

Updates: Ship early and make updates later.

Power of Giving up Power: Creativity comes from individuals, not structures.

Meta-programming: Building programs on a program like excel add-ons.