



Engineering Disasters . . .

Norman F. Simenson
Federal Aviation Administration

Engineering disasters are always the result of bad management and never the result of bad engineering—or almost always. This article describes three lines of defense any manager can use to guard against bad engineering.

Any experienced program manager (PM) will quickly recognize that I have indulged in a little hyperbole in the following article to emphasize a number of deeply held beliefs.

I stress that the article is *not* intended to provide a shield for those PMs who go to extreme lengths to avoid any risks. By definition, good PMs are risk takers. That is their strength. Risk taking becomes a weakness only if the PM does not understand the difference between gambling and controlled risk taking. In real life, the knowledgeable risk taker will (almost) always outperform the gambler *and* the ultraconservative—by a wide margin. A word to the wise: Most insurance companies are highly profitable—by design, not by accident.

If I seem to have let engineers off easily, that is not the intent. Engineers are fully responsible for the specific task assigned to them and must answer in full for a competent job. However, only the PM and chief engineer are responsible for the full job. They must shoulder the full blame if the project fails. The point is, if the project is lost for “want of a nail! . . .,” for the failure of one—or even several—lower-level engineers, or because of a few minor glitches, the program manager and chief engineer are mightily to blame.

Bad Management

Engineering *development* disasters are *always* due to bad management and *never* to bad engineering because, for any but the smallest noncritical project, the PM must do risk management. The plea “failed due to uncontrollable forces” or “my team let me down” is almost invariably an admission of bad

management. Worse, it means not only that the PM was inept but also does not have a clue about how to improve. A manager is not merely an overpaid administrator who handles the budget and doles out the work. A manager is the principal defense against the real-life disasters that can destroy any program.

Simply put, the PM must expect and plan for *all* engineering problems. This includes workable contingency plans that will prevent problems from impacting schedule and cost. Even with the best of engineers, creativity cannot be planned for and scheduled like a train timetable; some slippage is bound to occur. The most mundane engineering projects contain requirements for a substantial degree of good engineering and creativity. Provision must be made for things not going as planned; otherwise, who needs a PM?

Contingency Plans

There are all sorts of contingency plans and safeguards against problems due to bad engineering. Some will also work for almost any unexpected technical difficulties. The different problem categories should be treated differently but have some overlap and similarities for planning purposes. Here, we will only take a superficial view of planning to prevent the impact of bad engineering.

Assessing Teammates Strengths

The first line of defense for the PM is to correctly assess the abilities of the engineer, then correctly assign the engineer to a job. The “hallway” technique of assigning people is a sure recipe for disaster. This technique assumes that everyone within the same labor category is equivalent, so it must be okay to assign the first warm body of the

appropriate category to pass the manager’s door in the hallway to the job at hand. The amazing thing is not how often the warm body fails but rather, how often they succeed.

For any given job, there are four types of people. One type has never done a similar job and may possess some or most of the necessary skills but is fundamentally an unknown quantity. The other three types have done a similar job before, but one has failed at it, one has performed acceptably, and one has performed outstandingly. Clearly, depending on the skill pool and if other priorities permit, the choice is one of the latter two types. If the task is sufficiently critical, the choice may be only someone of the last type. The PM must give special care to the selection of the chief engineer and plan to accommodate their strengths and weaknesses. (A good chief engineer will devise a similar plan to accommodate the PM.)

If people of the first two types only are available, the manager must spend considerably more effort to decide among the candidates. But if managers lack qualified people for a program, they must have the courage to tell their management and insist on a workable solution. Agreeing to take on an impossible program is managerial malpractice. It is impossible to categorize people with respect to a job if the manager has no notion of what skills the proposed job requires or what skills the candidate’s previous job required. Both are easily determined, as well as the performance of engineers on past jobs, but not without effort on the manager’s part.

In deciding to give someone “another chance,” the manager must assess whether that someone has demonstrated an unremediated lack of some

necessary technical or personality skill. To determine if someone deserves a crack at something that will probably stretch their abilities, the manager must decide whether that person has the basic technical and personality skills needed and is likely to rise to the occasion. In either case, it must be assumed that an individual that has failed at or has never performed a task will need extra support in the form of training, mentoring, and supervision. That person is also likely to take longer to perform the task than one of the successful people. Due allowance must be made.

Provide Support and Resources

The second line of defense is to ensure that the engineer assigned to the task has the necessary support and resources. If these are not supplied in sufficient quality and quantity, the correlation between past and future success is bound to be poor. No matter how skillful the carpenter, a good job of nailing one board to another is unlikely without a hammer or nails or if one board is at another site two miles away. Unfortunately, it is common to overload an outstanding performer. It is easy to assume that a fraction of one good worker is better than all of a weak one or that a strong performer can make do with significantly less, i.e., inadequate support and significantly fewer resources. Almost equally disastrous is the practice of rationing resources “impartially” with little or no regard to the difficulty of the job or the ability of the performer. It does not work for parents, and it will not work for managers.

Plan Redundancy

A third line of defense is planned redundancy. It always amazes me how top managers will skimp on a program when sufficient planning and use of resources will ensure delivery, then spend resources, seemingly without limit, once failure is imminent and generally unavoidable. If you plan for failure upfront, it is avoidable. Suppose the only candidates for a job are of the first two types of employee described earlier. In this case, it is best to start two or even three on different aspects of the same job

in parallel. When one or two of your candidates fail, go with the successful candidate. Never make the mistake of shifting resources from a succeeding candidate to a failing candidate. This is a formula to ensure that everyone will fail, which has been proven repeatedly in military combat. Cut your losses quickly—this includes getting rid of bad engineers quickly. (Bad engineers are those who are not only incompetent but also take no responsibility for, and therefore do not learn from, their mistakes.)

If you wind up with all failing candidates, make sure you have the machinery in place to detect this early and, as each candidate fails, switch to an alternative strategy. This may involve supporting or replacing them with a more successful performer. This is one reason why you want to plan to support the weak candidate with more time and resources and with more supervision at the outset. It also is a reason never to start a program without some engineering reserve (which may be no more than potential overtime). Programs that start with everyone already overcommitted always fail. For critical tasks, it is a good idea to backup even the best engineers. Consider this “bus” insurance. (You must always worry about and plan for your key engineers being hit by a bus or other catastrophe.)

Using a low-risk, redundant approach assures against the failure of any high-risk approach. The resources expended on the redundant alternative(s) should be considered an insurance premium. Where a parallel approach strategy is used, multiple successes will speed the result, so not much is lost if properly planned.

There are many other strategies a competent manager can use to ensure against bad engineering or other disasters. Enough strategies should be used to reduce the risk of failure to a tolerable level at an acceptable cost.

Conclusion

In closing, I make three points. First, nothing in this article applies particularly to software managers. Second, through the software process improvement method and related techniques

initiated by the Software Engineering Institute and others, software has done more than its fair share to raise “scientific” management from a simple collection of heuristics toward the status of a science. Third, despite the general wailing, breast-beating, and gnashing of teeth, software must be doing something right. There has been nothing like the headlong rush to software since the similar rush to electronics after World War I. The average automobile of today has more software in it than the first Apollo spacecraft to arrive at the moon less than 30 years ago! ♦

About the Author



Norman F. Simenson is vying for the title of “grand old man” of hard real-time software. (He has an IBM Engineering School diploma dated March 1955 and signed by T.J. Watson—Sr. and Jr.) He also has various advanced degrees from the University of Michigan.

Currently, he is a member of a Software Engineering Group at the Federal Aviation Administration (FAA). This group is charged with the responsibility to improve the FAA’s software engineering processes and practices. He participates in most Software Capability Evaluations performed at the FAA and also is editor (and resident curmudgeon) of the *FAA Software Engineering Process Group Interface* newsletter: www.faa.gov/ait/sep/sep/news1.htm.

Simenson has been a programmer, program manager, chief engineer, chief architect, and chief scientist. He has worked with real-time systems for about eight years.

DOT/FAA/AIT-5
800 Independence Avenue, S.W.
Washington, D.C. 20591
Voice: 202-267-7431
Fax: 202-267-5080
E-mail: norm_simenson@faa.dot.gov

Note

1. “For want of a nail, a shoe was lost. For want of a shoe, a horse was lost. For want of a horse, a message was lost. For want of a message, a battle was lost. For want of a battle, a war was lost. For want of a war, a kingdom was lost. And all for the want of a nail!”