Model Solution Prepared By Asad Ladha

Summary

Robert D. Austin's article "The Effects of Time Pressure on Quality in Software Development: An Agency Model" discusses a common problem of estimation in a development environment. In the past, managers have often systematically added slack to a software development schedule to allow for contingencies, and to allow developers time to maintain high software development standards. Austin suggests that 'systematically adding slack' does not necessarily maintain quality standards.

The article begins by describing a common scenario where developers begin taking short cuts in their work in order to meet a deadline. In this particular situation a small team of developers discover a new software requirement late in the development cycle. The software requirement can be offered in a variety of ways but and the developers are faced with two conflicting priorities: completing the project on time, and implementing a thorough and complete solution. The team reluctantly took a shortcut to complete the project on time and meet the deadline. A shortcut helped the developers individually (they met their development deadline), but it cost the company (the shortcut solution would have to eventually be replaced by a complete solution later).

To prevent short cut taking, a common policy to date is to add slack to project schedules to give developers extra time. Adding extra contingency has two major drawbacks: first it doesn't enhance software quality, second adding extraneous time costs the company money, and costs can be minimized by adopting other deadline setting policies.

Austin goes on to describe the Agency Framework, which allows him to generalize behavior in a development team. The premise is that developers have specialized skills that their managers do not. Thus, it is hard to manage and control a team of developers. The Agency Frameworks introduces two agents (developers) who are competing for rewards (pay raises, promotions etc.) and can either report that they are on schedule or behind schedule. The agency framework also introduces a set of variables: the probability of receiving a realistic deadline p, the personal gain by taking a quality-compromising shortcut Q_1, $Q2$, the penalty for claiming that you are behind schedule C. Austin describes common game theory behavior, and the mathematics involved to predict behavior. According to game theory equilibrium is reached when all developers complete high quality work or they all complete low quality work (by taking shortcuts) depending on values of C, p, Q_1, $Q2$.

Given the results from the Agency Framework, Austin describes the manager's perspective to determine optimal deadline setting policies. A manager is interested in minimizing total development costs; which is directly affected by estimating task duration and setting deadlines. It is interesting to note that providing quality software is not an immediate priority for managers. Austin suggests that planning deadlines and development deadlines should be set differently. Based on the Agency Framework, and cost functions, the optimal deadline setting policy was to create shorter development deadlines and longer planning deadlines. This forces developers to work harder to meet deadlines, and has a lesser penalty if they don't meet the more aggressive deadline. This way, developers will not take shortcuts because they don't have much to loose if they don't meet the deadline.

Austin concludes that 1. Systematically adding slack is not necessarily a cost minimizing policy; 2. Deadlines and planning estimates should be set separately; 3. Deadlines should be set aggressively to be 'stretch goals' that few developers regularly meet. Austin also mentions that although development teams do not work in the exact fashion described in his example; the results are consistent with earlier studies in 1980.

Critique

This article was very well written and provides a convincing argument. In my personal experiences with development teams, I have encountered similar problems with deadline setting and shortcut taking.

Robert Austin seems to have as much insight (or lack thereof) as some of the managers I have worked with. Austin is accredited with over 10 years of management experience, a Ph D., awards, a published book and a variety of degrees[1]; but none of which have been in computer science or related to software development. The main reason why this article is inaccurate is that Austin has not been able to bridge the gap between developers and managers.

This article is analogous to one I could have written about my ability to consistently sleep 16-18 hours a day - everyday, and that the best way for me to sleep a regular 8-10 hours is by staying awake longer according to English studies performed in the 1850s. In truth, staying awake is an obvious, but not helpful conclusion; and thankfully most people aren't as lazy as I am.

In the discussion portion of the article Austin identifies a variety of reasons why his findings are not precisely true in every situation, but that his findings are consistent with other studies and should be true in general. What Austin does not recognize is that the situation he gave as an example is rare and should not be considered normal development procedure. This fallacy is obfuscated further because most of the research he refers to was completed 4-45 years before he wrote the article. Given the pace of software development today, research completed several years ago is questionable and possibly outdated.

Austin discusses why his model is exactly that, a model and that it does not explain each and every possible behavior. However not only does the Agency model fall short of describing all scenarios, it is based on an extremely simplified employee structure. Usually a software project will have a project manager, program manager, lead developers, and developers not just a manager and developers as Austin describes. The benefit of these extra 'manager' roles is that there is less of a knowledge/skill gap between employees.

The solution of setting unrealistic deadlines, and systematically setting planning deadlines later than development deadlines is similar to systematically adding slack development deadlines except it also introduces potential HR problems such as motivation and burn-out since developers would be continuously working hard to meet unrealistic deadlines.

Future Work

The effects of time pressure and quality of work are very difficult to measure in any discipline, not only in software development. I have participated in development teams that have worked well; and none of which had the employee structure/experience that Robert Austin elicits. The problem with this study is that time pressure and quality of work are not directly related. Development teams with experienced veterans, or a thorough development process, or employee structure would produce different quality work, even if they all were subjected to the same amount of time pressure. A more useful study would be one that studies different development team dynamics (teams that vary in experience, process and structure) and the effects of time pressure on quality of work. The extra component, team dynamics, would help isolate whether the problem is with the development team or deadline setting policy.
Managers try to re-organize, or try something new when a projects fail. A manager who implements Austin's recommendation would temporarily mask problems within the development team. Future studies should focus more on solving and finding development problems.

[1] http://dor.hbs.edu/fi_redirect.jhtml?facInfo=bio&facEmId=raustin