Five Traps that Lead to Project Failure (and How to Avoid Them)

No one starts a software project thinking that it is doomed to fail, but, as readers of this publication surely know, many projects end up falling far short of expectations. And while a recent report by PMI suggests that success rates are improving, there remains cause for concern. A significant number of companies are still underperforming expectations - failing to deliver software that functions as intended and drives positive business results.

PMI's report breaks out project development teams into two distinct camps: "overachievers" and "underachievers." The former are delivering projects on time and on budget and also meet their original intent or business goals. The latter are seeing the opposite – projects that are running over budget and off schedule, and, worse, failing to deliver the intended benefits.

What's the difference between the two groups? Primarily, overachievers understand and practice mature project management processes. As the report states, "when proven project, program, and portfolio management practices are implemented, projects are more successful."

But companies that have found greater project development success are doing more than just adhering to sound management strategies. They are also managing to avoid the traps that often lead to failure in the first place. They have become adept at forgoing bad (often traditional) practices in favor of employing more flexible, data-driven, and accurate tactics that are more likely to lead to projects being completed cost effectively and within realistic timeframes. Most importantly, they are using these tactics to deliver solutions that, at the end of the day, deliver the expected functionality and true value to shareholders.

Here are five traps that these organizations are successfully avoiding, and better strategies that can be used in their place.

Mistaking "agile" for "rushing"

The phrase "fools rush in" exists for a reason. Too often, teams will enter into a development project without having to do the necessary amount of due diligence. Under immense pressure to deliver, their first instinct is to begin work without a clear roadmap backed up by data. They guess and make assumptions. They certainly do not estimate. After all, who has time for that? They have coding they need to get to!

Organizations tend to fall into this trap all the time. They feel the need for greater agility exempts them from detailed planning, when in fact planning can actually help improve agility further down the line. Plans help determine the number of resources and amount of functionality required for a project. They lead to the creation of realistic timelines and expectations. They provide an invaluable roadmap that can lead to success.

And yes, plans can and probably will change. But that does not mean they should not happen in the first place. Plans will just need to be reforecasted as the project progresses. The overachievers understand this.

Taking a DIY attitude

Some people do indeed create plans, but often those plans are created in a vacuum. These are the teams that think that project estimation is pure overhead. They do not see the value in leveraging a data-driven estimation

process to gauge the time, work, and human resources hours it will take to complete their projects.

As a result, they attempt to create a plan based off of “gut feel.” I can tell you in my decades of experience working in project estimation, this hardly ever goes well. The problem is that internal project managers are unlikely to have all of the data necessary to accurately plan the scope of their projects. Instead of referring to historical, real-world data pulled from similar projects that have been successfully completed by other companies, they solicit input from their own past experiences.

But, as we all know, every project is different. A project that a team did two years ago may not map accurately to the one they are about to start working on. It helps to have an outside perspective and data from similar projects of size and scope. Only then can a true “apples to apples” comparison be made.

Estimating from the bottom up

Those that take on a DIY approach to planning also tend to make the mistake of estimating from the bottom up. In essence, the project manager creates a scope based on input from the development team. They provide estimated numbers of hours it will take to complete each individual task of the project. The manager then inputs their recommendations into a spreadsheet and figures out a schedule and budget.

These numbers are not based on hard data. Rather, they are the result of best guesses. As such, the project is built on a shaky foundation before it even begins.

Overachieving organizations tend to plan from the top down. Top down scope-based estimation looks first at the scope of the project and factors in the efficiency of the team and the complexity of the work.. Teams use historical data to determine how long it will take to complete their own work, and how much money it will cost.

While it’s true the data can be pulled from internal sources, it also helps to look at overall industry trends, which can be found in external project databases. This information can help project managers create accurate, comprehensive, and realistic plans that take into consideration virtually every aspect of the project before a single line of code is written.

Think estimation is anti-agile

Agile has become the preferred development method for most businesses, which makes sense. Agile allows developers to move, shift, and course correct as necessary. It gives them the freedom to quickly respond to changing requirements and shareholder demands.

Many of these development teams are anti-estimation. They feel that scope-based planning restricts the agile process. They do not wish to be tied down by an upfront plan that shows the path ahead.

But nothing is further from the truth. Indeed, scope-based estimation and agile development complement each other. While the planning process does provide an accurate representation of a project’s scope, planning does not prohibit changes as development commences. In fact, an agile team armed with an upfront estimate can more easily react to changes as they become necessary, because they can see how those changes might impact the rest of the process. They can also use the initial plan to get a better understanding of the project’s requirements before work is begun, allowing them to make agile adjustments from the start.

Far from discouraging recalibration, scope-based estimation supports the process. It allows project managers to see and evaluate different scenarios and make better choices during the course of development.

Panic and add more people

Despite best efforts, projects may end up falling behind schedule. The natural reaction to this problem is to bring more developers into the mix. The wisdom is that more people can take on more work. This will accelerate development and get things back on track.

History shows that this is not the case. I have seen this firsthand. Back in the 1980s I worked directly with a photo development company that spent years working on a project that fell woefully behind schedule. They brought in more people, spent more money – and still came up short. Companies are still making the same mistakes today, ignoring what noted computer scientist and author Fred Brooks once wrote: “adding manpower to a late software project makes it later.”

Adding additional resources to a flailing software project creates a myriad of problems. More people create more lines of communication to manage, which can lead to even more confusion and opportunities for misunderstandings. Adding resources can also take developers away from other projects that they may be more suited for. All of this can open up the potential for mistakes to be made and software defects to be created, and possibly missed entirely or, worse, ignored.

An organization that I worked with a few years ago found this out the hard way. Their project fell behind schedule and, despite adding a massive amount of personnel to fix the problem, many defects went unnoticed. The company decided to proceed with its release anyway. Guess how that turned out?

The solution is hardly ever adding more people. Rather, it is having the right amount of people to begin with, and allowing them to work smarter. This goes back to the initial planning phase. Careful upfront planning can provide project managers with a good idea of how many team members should be assigned to their projects. In some cases that may require a larger team, but in many situations the team can be smaller. Smaller teams provide greater opportunities for success because they can be more efficient, communicative, productive – and, yes, agile.

Perhaps it is not so much what the champions do that sets themselves apart from the rest of the pack, but what they do *not* do. Specifically, they do not fall into the same traps that many organizations tend to find themselves in. They avoid them and take different paths. Those paths have led these organizations to better success in delivering software that meets the expectations of key stakeholders.