

By Robert Bogue

Takeaway

Project managers often use techniques that are successful at reducing cost and the development time without impacting quality. However, it is possible for them to push their techniques too far. Here is how.

Project managers

Do I have your attention yet? Most people in [software development](#) instinctively know that the project manager's drive to make sure the project is on time is at odds with the desire to have high quality software. Not that project managers don't want high quality software too, it's just they want the software and they want on-time delivery and costs that are at or less than what was estimated, in addition to quality. Their efforts are often successful at reducing cost and the development time without impacting quality. However, it is possible for them to push their techniques too far.

Although all of the following [project management techniques](#) are at least well meaning, and in some cases, they are even time honored techniques, they do have the potential for disaster.

Time boxing

Getting top honors in the list of things which can destroy software quality is the practice of time boxing. This practice is where you tell someone how long they are allowed to work on the task before it must be turned over. I say turned over and not completed because used at its extreme it often means that the code isn't complete, it's merely pushed along the process.

Time boxing works—most of the time—because it does three things:

1. It forces the developer to be creative in finding a solution that fits within their budget.
2. It eliminates unnecessary frills that are often added to software that don't necessarily add value.
3. It prevents the developer from over testing.

The intent is to get the piece working. There's a whole QA phase where detailed testing will hopefully reveal any problems that there may be with the code.

Time boxing doesn't work when the problem is unknown, the technology isn't proven, or there's no real way to predict the results. It also doesn't work when the box is made so small there's no possible way to complete the objective within the allotted time. In other words, there are some things that time boxing works well for, such as well understood, carefully estimated, execution-type tasks. And some that it doesn't work well for, like research and development, problem solving, etc.

When time boxing is used correctly it shouldn't result in poorly tested code that will cause hundreds of hours of diagnostics and rework. It should be used with moderation to ensure the lowest cost, quickest and best quality software possible.

False dates

Everyone needs something to strive for. Milestones are a respected way to motivate people towards a unified goal. That motivation can often lead to great results in short periods of time. However, everyone must accept that the milestone date won't always be met and then they must decide what to do when that happens.

Project managers must put dates out in front of the group to motivate them but when the dates aren't realistic and they are consistently missed, it's time to reevaluate the plan. The problem is that when a really important date comes up there will be little drive to hit it because an expectation has been set that the date isn't important. After all if the team misses 10 dates in a row, how important can the 11th date be? It's like the boy who cried wolf.

If timelines are being set that have no penalty behind them and people aren't meeting them it's time to put some teeth behind them or move the whole timeline.

Continuing to create a constant stressful and confusing situation does not create a good software solution in the long run. Developers need to be able to concentrate on their work. The desire to meet the date and the confusion about whether the date is real or not may lead to developers skipping critical steps in the development process and, in doing so, creating problems that will be hard to find.

Pretending nothing is wrong

When it comes to project management, ignorance is not bliss. Despite the often overwhelming political pressure to have a successful project, it's necessary to be forward in describing the risks of the project with the rest of the organization. Nearly every software development project has risks that make it possible that it will be late or over budget or both.

The problem with this is eventually there comes a time when those risks turn into a reality and then panic sets in. Everyone scrambles to put together the rest of the pieces of the project and the quality of the project will suffer from the hastiness of the final assembly.

Of course, this problem isn't fully realized until the project gets behind; however, most projects find a way to get behind a little at some point in the project. Nearly every software development project is at risk of the rush that is caused when the business learns about the true state of the project after believing that nothing was wrong for a long time.

Ignoring dependencies

In software development we have a great number of techniques for delaying dependencies. We can stub out functions, remove connecting infrastructure, or bypass extensive error handling. All of these techniques when used correctly can be helpful to moving a project along. However, when it becomes required to get the project completed and when the costs of these techniques are not factored into the planning, trouble sets in.

Often times just sequencing a software development project can be very challenging. The dependencies are often hard to see so inevitably there are dependencies that aren't in the plan to start. Scheduling around these unforeseen dependencies can drive a person mad. So the techniques of neutralizing dependencies are used. However, if too many of these techniques are used the clean up cost can often become a non-trivial part of the project's overall cost—and one that isn't discovered until the very end.

Making sure that you're doing what is necessary to manage dependencies—without picking up too much cost—is a necessary part of software development. When project managers don't balance the cost with the convenience of reducing the dependency they create hastily assembled code which exhibits quality problems.

Disaster waiting to happen?

Be on the look out for project management techniques which have gone awry and may be causing your next project to be a quality disaster.

Additional resources

- TechRepublic's [Downloads RSS Feed](#) **XML**
- Sign up for our [Downloads Weekly Update](#) newsletter
- Sign up for TechRepublic's [Project Management](#) newsletter
- Check out all of TechRepublic's [free newsletters](#)
- [Prepare for the PMP Exam with these 50 project management framework questions](#)
- [Ensure project success by first building a practical development environment](#)
- [Download our project definition template for small and medium-size projects](#)

Version history

Version: 1.0

Published: December 12, 2005

Tell us what you think

TechRepublic downloads are designed to help you get your job done as painlessly and effectively as possible. Because we're continually looking for ways to improve the usefulness of these tools, we need your feedback. Please take a minute to [drop us a line](#) and tell us how well this download worked for you and offer your suggestions for improvement.

Thanks!

—The TechRepublic Downloads Team