



White paper

Triple your schedule

Why projects exceed their schedules

www.stdtime.com
Scoutwest, Inc.

Triple your schedule

One of my old bosses used to say, “Take whatever time estimate the programmers give you and triple it. Then you’ll know when the product will be ready.” He used to talk about “programmer time” as if it were an obscure interplanetary time system. Even though it was said in jest, I don’t think he was exaggerating. At that company, we never used any detailed project management techniques, or even any QA. The programmers all wrote their own stuff, and tested it a little bit before giving it directly to customers. I’m guessing that my boss arrived at this 3X rule of thumb over time, having been a programmer himself, and employing several programmers at his company. He was pretty informal about the whole thing, never worrying when a release would be available or how his resources were allocated.

We never dug too deeply into the reasons for the 3X factor. We just joked about it and went on. I suppose, in all fairness, it didn’t really matter much to our customers. Programmers at that company were always asked informally to predict how long things would take. There was never an extended period of thought leading up to the predictions. The boss would just come over to the desk and ask, and we’d give the best answer that came to us. Then he’d triple it. We never knew the factors that caused our development cycles to be three times longer than our predictions. We never gave it that much thought. Regardless of whether it mattered or not; it was a bad practice.

That mode of operation may be okay for some companies, but not all. In fact I think it may be okay only for small companies with vertical customers. Larger companies, consultancies, and companies in horizontal markets probably need to watch their resource allocation and scheduling a little more closely. If you know what to look for, you don’t need to apply sweeping 3X multipliers to your schedule. The items below will give you some factors to consider when making schedules.

Forgotten tasks can account for 25% of a schedule

Nobody can remember every task they need to perform for a product release. The more detailed your requirements and design documents are, the fewer tasks you'll forget to account for. Even with detailed requirements and design documents, your project schedule should account for unknown tasks in some way.

Normal people only work about 6 hours per day

Normal people go to the bathroom, take breaks, surf the web, talk with friends, plan lunch, come in late, leave early, run errands, get snacks, and call their wives – perhaps not all in the same day, but these are normal functions. Don't schedule people for 8-hour days. It's not going to happen.

Informal meetings can occupy 2 hours in a day

Have you scheduled time for telephone, email, and office meetings? Granted, these are usually unscheduled and unpredictable, but in real life, they happen. A simplistic project schedule that doesn't include them will come up short.

Long release cycles make predicting hard

The rate of error in task scheduling multiplies as the size of the project increases. How big is your project? Can you break it up into smaller releases that are easier to estimate?

Context switches take time

Switching context from one task to the next takes 30 minutes. The granularity of any given task should be no less than 1 hour. Since people are not machines, it is sometimes hard for them to mentally switch from task to task without any downtime.

Thrash is your enemy

Every project experiences redesign because of things you didn't think of. The key is to reduce this time with a small amount of process, policy, and thorough research. If you are frequently redesigning, you are wasting time. This usually results from insufficiently researched tasks. If you don't dig deeply into the details, then you'll likely have to do it later. Sometimes you'll end up doing the research after you've already implemented something the wrong way. You'll have to rip that out and redesign. That's thrash.

Unusually tough debugging happens

Don't count on clear sailing through QA and debugging cycles. It rarely happens, but it's rarely accounted for. What happens if you hit a tough problem? Will it cost you any extra time?

Unfamiliar technologies take time to learn

If your product includes new or unfamiliar technologies that some of the developers aren't proficient at, then make sure you schedule some time to come up to speed on them. You may also need some time to build proof of concept prototypes that include the new technologies.

Politics and gossip

Hopefully, your company doesn't have any of this, but if it does, it's a huge time sink and morale buster. If you are on a man-hauling project, you can't avoid some of this. That's another reason to keep your release cycles short. Discontentment can grow as projects wear on.

Integration with other products

If your product integrates with other products in any way, leave extra time to come up to speed on those technologies. There may be some unknowns there to trip you up, and extra detailed research may be required.

Optimizations

Some products need optimizations that are not obvious and identified during initial development. It's common for software products to be big, fat, and slow. If you release a product with performance problems you'll wish you had spent the time to fix it in the first place.

Thorough testing can occupy 50% of the cycle

It doesn't happen all the time, but sometimes QA time can dominate half the development cycle. Leave time for at least 25 - 35%. This is another good reason to track your time closely. You'll get a better sense for the amount of time each part of the cycle takes. Beta testing is likely part of this time, and it can be pretty time consuming just to line up testers and get them to look at the product. They have other work to do, and don't always jump when you ask them to test your product. It takes time to coordinate this effort. Bottom line, if you completely ignore the testing cycle in your schedule, you'll be way off base.

The installer and user documentation

Windows 2000 installer makes things easy for software products these days, but make sure you schedule some time to produce and test this. Installers and user help systems are often an afterthought because they are not part of the core product. Make sure your project schedule includes tasks for these items.

Final release considerations

Did you schedule time for packaging, manufacturing, special shipping needs, distribution, web page support, web download postings, tech support training, sales training, marketing materials, and press releases? They all take time, especially for the very first release of the product.

About Us

Scoutwest, Inc. develops and publishes project management and time tracking products for consulting, manufacturing, government, and general business applications.

Thousands of small to large businesses, in dozens of countries worldwide, trust their mission critical business processes to Scoutwest products. Standard Time® and Standard Issue® work together to offer well-rounded project management solutions.

We specialize in packaged software for timesheets, project management, time tracking, defect tracking, and issue tracking. Standard Time is a web-based timesheet that also runs on Windows, Palm OS, and Pocket PC. It can be used for client billing and task management. Standard Issue is used for bug tracking and general issue tracking.

Please visit these web sites for more information.

www.stdtime.com
www.sdtissue.com