Books
Articles
Interviews
Presentations
About Me
Contact Me

Rapid Development: Taming Wild Software Schedules. Redmond, Wa.: Microsoft Press, 1996. 660 pages. Retail price: $35. ISBN: 1-55615-900-5. Available from Microsoft Press 1-800-MS-PRESS (1-800-677-7377).

Buy Rapid Development from Amazon.com.

A Case Study in Classic Mistakes

Take the Classic Mistakes Survey!
My company is collecting data on "the new" classic mistakes. Here's my blog entry on the background. Here's where you can take the survey!

Mike was eating lunch in his office and looking out his window on a bright April morning. He was a technical lead for Giga Safe, a medical insurance company.

"Mike, you got the funding for the Giga-Quote program! Congratulations!" It was Bill, Mike's boss. "The executive committee loved the idea of automating our medical insurance quotes. It also loved the idea of uploading the day's quotes to the head office every night so that we always have the latest sales leads online. I've got a meeting now, but we can discuss the details later. Good job on that proposal!"

Mike had written the proposal for the Giga-Quote program months earlier, but his proposal had been for a standalone PC program without any ability to communicate with the head office. Oh well. This would give him a chance to lead a client-server project in a modern GUI environment, and that's what he wanted. They had almost a year to do the project, and that should give them plenty of time to add a new feature. Mike picked up the phone and dialed his wife's number. "Honey, let's go out to dinner tonight to celebrate ..."

The next morning, Bill met with Mike to discuss the project. "OK, Bill. What's up? This doesn't sound like quite the same proposal I worked on."

Bill felt uneasy. Mike hadn't participated in the revisions to the proposal, but there hadn't been time to involve him. Once the executive committee heard about the Giga-Quote program, they'd taken over. "The executive committee loves the idea of building software to automate medical insurance quotes. But they want to be able to transfer the field quotes into the mainframe computer automatically. And they want to have the system done before our new rates take effect January 1. They moved the software-complete date you proposed up from March 1 to November 1, which shrinks your schedule to 6 months."

Mike had estimated the job would take 12 months. He didn't think they had much chance of finishing in 6 months, and he told Bill so. "Let me get this straight," Mike said. "It sounds like you're saying that the committee added a big communications requirement and chopped the schedule from 12 months to 6?"

Bill shrugged. "I know it will be a challenge, but you're creative, and I think you can pull it off. They approved the budget you wanted, and adding the communications link can't be that hard. You asked for 36 staff-months, and you got it. You can recruit anyone you like to work on the project and increase the team size too." Bill told him to go talk with some other developers and figure out a way to deliver the software on time.

Mike got together with Carl, another technical lead, and they looked for ways to shorten the schedule. "Why don't you use C++ and object-oriented design?" Carl asked. "You'll be more productive than with C, and that should shave a month or two off the schedule." Mike thought that sounded good. Carl also knew of a report-building tool that was supposed to cut development time in half. The project had a lot of reports, so those two changes would get them down to about nine months. They were due for newer, faster hardware, too, and that could shave off a couple of weeks. If he could recruit really top-notch developers, that might bring them down to about seven months. That should be close enough. Mike took his findings back to Bill.

"Look," Bill said. "Getting the schedule down to seven months is good, but it's not good enough. The committee was very clear about the six-month deadline. They didn't give me a choice. I can get you the new hardware you want, but you and your team are going to have to find some way to get the schedule down to six months or work some overtime to make up the difference."

Mike considered the fact that his initial estimate had just been a ballpark guess and thought maybe he could pull it off in six months. "OK, Bill. I'll hire a couple of sharp contractors for the project. Maybe we can find some people with communications experience to help with uploading data from the PC to the mainframe."

By May 1, Mike had put a team together. Jill, Sue, and Tomas were solid, in-house developers, and they happened to be unassigned. He rounded out the team with Keiko and Chip, two contractors. Keiko had experience both on PCs and the kind of mainframe they would interface with. Jill and Tomas interviewed Chip and recommended against hiring him, but Mike was impressed. He had communications experience and was available immediately, so Mike hired him anyway.

At the first team meeting, Bill told the team that the Giga-Quote program was strategically important to the Giga Safe corporation. Some of the top people in the company would be watching them. If they succeeded, there would be rewards all around. He said he was sure that they could pull it off.

After Bill's pep talk, Mike sat down with the team and laid out the schedule. The executive committee had more or less handed them a specification, and they would spend the next two weeks filling in the gaps. Then they'd spend six weeks on design, which would leave them four months for construction and testing. His seat-of-the-pants estimate was that the final product would consist of about 30,000 lines of code in C++. Everyone around the table nodded agreement. It was ambitious, but they'd know that when they signed up for the project.

The next week, Mike met with Stacy, the testing lead. She explained that they should begin handing product

builds over to testing no later than September 1, and should aim to hand over a feature-complete build by October 1. Mike agreed.

The team finished the requirements specification quickly, and dove into design. They came up with a design that seemed to make a good use of C++'s features.

They finished the design by June 15, ahead of schedule, and began coding like crazy to meet their goal of a first-release-to-testing by September 1. The project hadn't been entirely smooth. Neither Jill nor Tomas liked Chip, and Sue had also complained that he wouldn't let anyone near his code. Mike attributed the personality clashes to the long hours everyone was working. Nevertheless, by early August, they reported that they were between 85 and 90-percent done.

In mid-August, the actuarial department released the rates for the next year, and the team discovered that they had to accommodate an entirely new rate structure. The new rating method required them to ask questions about exercise habits, drinking habits, smoking habits, recreational activities, and other factors that hadn't been included in the rating formulas before. C++, they thought, was supposed to shield them from the effects of such changes. They had been counting on just plugging some new numbers into a ratings table. But they had to change the input dialogs, database design, database access, and communications objects to accommodate the new structure. As the team scrambled to retrofit their design, Mike told Stacy that they might be a few days late releasing the first build to testing.

As expected, the team didn't have a build ready by September 1, and Mike continued to assure Stacy that the build was only a day or two away.

Days turned to weeks, and the October 1 deadline for handing over the feature-complete build to testing came and went. Development still hadn't handed over the first build to testing. Stacy called a meeting with Bill to discuss the schedule. "We haven't gotten a build from development yet," she said. "We were supposed to get our first build on September 1, and since we haven't gotten one yet, they've got to be at least a full month behind schedule. I think they're in trouble."

"They're in trouble, all right," Bill said. "Let me talk to the team. I've promised 600 agents that they would have this program by November 1. We have to get that program out in time for the rate change."

Bill called a team meeting. "This is a fantastic team, and you should be meeting your commitments," he told them. "I don't know what's gone wrong here, but I expect everyone to work hard and deliver this software on time. You can still earn your bonuses, but now you're going to have to work for them. As of now, I'm putting all of you on a 6-day-per-week, 10-hour-per-day schedule until this software is done." After the meeting, Jill and Tomas grumbled to Mike about not needing to be treated like children, but they agreed to work the hours Bill wanted.

The team slipped the schedule two weeks, promising a feature-complete build by November 15. That allowed for six weeks of testing before the new rates went into effect in January.

The team released its first build to testing four weeks later on November 1 and met to discuss a few remaining problems areas.

Tomas was working on report generation and had run into a roadblock. "The quote summary page includes a simple bar chart. I'm using a report generator that's supposed to generate bar charts, but the only way it will generate them is on pages by themselves. We have a requirement from the sales group to put the text and bar charts on the same page. I've figured out that I can hack up a report with a bar chart by passing in the report text as a legend to the bar-chart object. It's definitely a hack, but I can always go back and reimplement it more cleanly after the first release."

Mike responded, "I don't see where the issue is. We have to get the product out, and we don't have time to make the code perfect. Bill has made it crystal clear that there can't be any more slips. Do the hack."

Chip reported that his communications code was 95-percent done and that it worked, but he still had a few more tests to run. Mike caught Jill and Tomas rolling their eyes, but he decided to ignore it.

The team worked hard through November 15, including working almost all the way through the nights of the 14th and 15th, but they still didn't make their November 15 release date. The team was exhausted, but on the morning of the 16th, it was Bill who felt sick. Stacy had called to tell him that development hadn't released its feature-complete build the day before. Last week he had told the executive committee that the project was on track. Another project manager, Claire, had probed into the team's progress, saying that she had heard that they weren't making their scheduled releases to testing. Bill thought Claire was uptight, and he didn't like her. He had assured her that his team was definitely on track to make their scheduled releases.

Bill told Mike to get the team together, and when he did, they looked defeated. A month and a half of 60-hour weeks had taken their toll. Mike asked what time today they would have the build ready, but the only response he got was silence. "What are you telling me?" he said. "We are going to have the feature-complete build today, aren't we?"

"Look, Mike," Tomas said. "I can hand off my code today and call it 'feature complete', but I've probably got three weeks of cleanup work to do once I hand it off." Mike asked what Tomas meant by "cleanup." "I haven't gotten the company logo to show up on every page, and I haven't gotten the agent's name and phone number to print on the bottom of every page. It's little stuff like that. All of the important stuff works fine. I'm 99-percent done."

"I'm not exactly 100-percent done either," Jill admitted. "My old group has been calling me for technical support a lot, and I've been spending a couple hours a day working for them. Plus, I had forgotten until just now that we were supposed to give the agents the ability to put their names and phone numbers on the reports. I haven't implemented the dialogs to input that data yet, and I still have to do some of the other housekeeping dialogs too. I didn't think we needed them to make our 'feature complete' milestone."

Now Mike started to feel sick too. "If I'm hearing what I think I'm hearing, you all are telling me that we're three weeks away from having feature complete software. Is that right?"

"Three weeks *at least*," Jill said. The rest of the developers agreed. Mike went around the table one by one and asked the developers if they could completely finish their assignments in three weeks. One by one, the developers said that if they worked hard, they thought they could make it.

Later that day, after a long, uncomfortable discussion, Mike and Bill agreed to slip the schedule three weeks to December 5 as long as the team promised to work 12 hour days instead of 10. Bill said he needed to show his boss that he was holding the development team's feet to the fire. The revised schedule meant that they would have to test the code and train the field agents concurrently, but that was the only way they could hope to release the software by January 1. Stacy complained that that wouldn't give QA enough time to test the software, but Bill overruled her.

On December 5, the Giga-Quote team handed off the feature-complete Giga-Quote program to testing before noon and left work early to take a long-awaited break. They had worked almost constantly since September 1.

Two days later, Stacy released the first bug list, and all hell broke loose. In two days, the testing group had identified more than 200 defects in the Giga-Quote program including 23 that were classified as 'Severity 1'-'Must Fix'-errors. "I don't see any way the software will be ready to release to the field agents by January 1," she said. "It will probably take the test group that long just to write the regression test cases for the defects we've already discovered, and we're finding new defects every hour."

Mike called a staff meeting for eight o'clock the next morning. The developers were touchy. They said that although there were a few serious problems, a lot of the reported bugs weren't really bugs at all but were misinterpretations of how the program was supposed to operate. Tomas pointed to bug #143 as an example. "The test report for bug #143 says that on the quote summary page, the bar chart is required to be on the right side of the page rather than the left. That's hardly a Sev-1 error. This is typical of the way that testing overreacts to problems."

Mike distributed copies of the bug reports. He tasked the developers to review the bugs that testing had assigned to them and to estimate how much time it would take to fix each one.

When the team met again that afternoon, the news wasn't good. "Realistically, I would estimate that I have two weeks' worth of work just to fix the bugs that have already been reported," Sue said. "Plus I still have to finish the referential integrity checks in the database. I've got four weeks of work right, total."

Tomas had assigned bug #143 back to testing, changing its priority from Sev-1 to Sev-3-"Cosmetic Change." Testing had responded that Giga-Quote's summary reports had to match similar reports generated by the mainframe policy-renewal program, which were also similar to pre-printed marketing materials that the company had used for many years. The company's 600 agents were accustomed to giving their sales pitches with the bar chart on the right, and it had to stay on the right. The bug stayed at Sev-1, and that created a problem.

"Remember the hack I used to get the bar chart and the report to print on the same page in the first place?" Tomas asked. "To put the bar chart on the right, I will have to rewrite this particular report from scratch, which means that I will have to write my own low-level code to do the report formatting and graphics." Mike cringed, and asked for a ballpark estimate of how long all that would take. Tomas said it would take at least 10 days, but he would have to look into it more before he would know for sure.

Before he went home for the day, Mike told Stacy and Bill that the team would work through the holidays and have all of the reported defects fixed by January 7. Bill said he had almost been expecting this one and approved a four-week schedule slip before leaving for the four-week Caribbean cruise he had been planning since the previous summer.

Mike spent the next four weeks holding the troops together. For four months, they had been working as hard as it was possible to work, and he didn't think he could push them any harder. They were at the office 12 hours a day, but they were spending a lot of time reading magazines, paying bills, and talking on the phone. They seemed to make a point of getting irritable whenever he asked how long it would take to get the bug count down. For every bug they fixed testing discovered two new ones. Bugs that should have taken minutes to fix had project-wide implications and took days instead. They soon realized there was no way they could fix all of the defects by January 7.

On January 7, Bill returned from his vacation, and Mike told him that the development team would need another four weeks. "Get serious," Bill said. "I've got 600 field agents who are tired of getting jerked around by a bunch of computer guys. The executive committee is talking about canceling the project. You have to find a way to deliver the software within the next two weeks no matter what."

Mike called a team meeting to discuss their options. He told them about Bill's ultimatum and asked for a ballpark estimate of when they could release the product, first just in weeks, then in months. The team was silent. No one would hazard a guess about when they might finally release the product. Mike didn't know what to tell Bill.

After the meeting, Chip told Mike that he had accepted a contract with a different company that started February 3. Mike began to feel that it would be a relief if the project were canceled.

Mike got Kip, the programmer who had been responsible for the mainframe side of the PC-to-mainframe communications, reassigned to help out on the project and assigned him to fix bugs in the PC communications code. After struggling with Chip's code for a week, Kip realized that it contained some deep conceptual flaws that meant it could never work. Kip was forced to redesign and reimplement the PC side of the PC-to-mainframe communications link.

As Bill rambled on at an executive meeting in the middle of February, Claire finally decided that she had heard enough and called a "stop work" on the Giga-Quote program. She met with Mike on Friday. "This project is out of control," she said. "I haven't gotten a reliable schedule estimate from Bill for months. This was a six-month project, and it's now more than three months late with no end in sight. I've looked over the bug statistics, and the team isn't closing the gap. You're all working such long hours that you're not even making progress anymore. I want you all to take the weekend off; then I want you to develop a detailed,

step-by-step report that includes everything-and I do mean *everything*-that remains to be done on that project. I don't want you to force fit the project into an artificial schedule. If it's going to take another nine months, I want to know that. I want that report by end-of-work Wednesday. It doesn't have to be fancy, but it does have to be complete."

The development team was glad to have the weekend off, and they attacked the detailed report with renewed energy the next week. It was on Claire's desk Wednesday. She had the report reviewed by Charles, a software engineering consultant who also reviewed the project's bug statistics. Charles recommended that the team focus its efforts on a handful of error-prone modules, that it immediately institute design and code reviews for all bug fixes, and that the team start working regular hours so that they could get an accurate measure of how much effort was being expended on the project and how much would be needed to finish.

Three weeks later, in the first week in March, the open-bug count had ticked down a notch for the first time. Team morale had ticked up a notch, and based on the steady progress being made, the consultant projected that the software could be delivered-fully tested and reliable-by May 15. Since Giga Safe's semi-annual rate increase would go into effect July 1, Claire set the official launch date for June 1.

Post Mortem

The Giga-Quote program was released to the field agents according to plan on June 1. Giga Safe's field agents greeted it with a warm if somewhat skeptical reception.

The Giga Safe corporation showed its appreciation for the development team's hard work by presenting each of the developers with a $250 bonus. A few weeks later, Tomas asked for an extended leave of absence, and Jill went to work for another company.

The final Giga-Quote product was delivered in 13 months rather than 6, a schedule overrun of more than 100 percent. The developer effort including overtime consisted of 98 staff-months, which was a 170-percent overrun of the planned 36 staff-months.

The final product was determined to consist of about 40,000 nonblank, noncomment lines of code in C++, which was about 33 percent more than Mike's seat-of-the-pants guess. As a product that was distributed to 600 in-house sites, Giga-Quote was a hybrid between a business product and a shrink-wrap product. A product of its size and type should normally have been completed in 11.5 months with 71 staff-months of effort. The project had overshot both of those nominals.

Take the Classic Mistakes Survey!

My company is collecting data on "the new" classic mistakes. Here's my blog entry on the background. Here's where you can take the survey!

Email me at stevemcc@construx.com.