Defect Tracking Programs

It's harder than you might think to squander millions of dollars, but a flawed software development process is a tool well suited to the job.

—Alan Cooper, The Inmates Are Running the Asylum

Every development team needs a formal system for tracking bugs and reminding the team what's left to do. Maybe your system is as simple as a bunch of Post-it Notes stuck on the side of your monitor, or maybe you use a professional-strength defect tracking program. But any successful project must have some system where all the known bugs are stored in a single place. If you try to keep everything in your head, bugs will slip through the cracks. You'll discover five bugs, and you'll spend so much time fixing four of them that you'll completely forget about the fifth. Then your product ships with a bug that could have been avoided.

Very small teams might still be able to succeed with ad-hoc systems like the Post-it Note method, but I wouldn't recommend it. Spend some money to buy or build a professional defect tracking program. You don't have to spend huge amounts of money. Defect tracking programs are such that the difference between a good one and a bad one is negligible compared to the difference between a bad one and nothing at all. If you have to get your tracking program from the lowest bidder, then so be it; but make sure you get something.

In this chapter, we'll go over the reasons why defect tracking software is so essential for keeping projects on course, and we'll examine what sorts of data should be added to your system. If you're already familiar with defect tracking, then this section may already be familiar to you. Then we'll discuss some of the advanced tricks a project leader can do with a good tracking program. Much of this chapter is focused on using a defect tracking system to make sure bugs get fixed, rather than on actually fixing the bug. But since this is a book about debugging, we'll also look at some debugging techniques for using a defect tracking system to debug your programs faster, as well.
NOTE Since no defect tracking program ships with Visual Studio .NET, I debated whether to pick some popular tracking program and talk about it or whether to merely talk about tracking programs in general. I decided to go with the later approach, but it shouldn't matter—most defect tracking programs are very similar, anyway.

Why Do You Need Defect Tracking Software?

The most important reason to have a defect tracking system is to enforce a process workflow. Anyone who wants a bug fixed must enter a bug report (usually called a PR for Problem Report) into a central repository, and developers are held accountable for fixing only those bugs in the repository. One benefit of this is that all bugs are listed in a single place that is viewable by the entire team—bugs can no longer slip through the cracks. But an equally important benefit is that each PR can be tracked, can be assigned to exactly one person, can be prioritized with respect to other bugs, and can be included in project status reports. Each PR can serve as a historical record of when the bug was found, what the problem was, whether the bug has been fixed yet, and if so, how long the fix took.

A development workflow doesn't end with the developer writing code to fix a bug. Next, the tester needs to verify the fix works, the project manager needs to decide whether the fix will be immediately released as a hotfix or deferred until the next version, the tech support representative may need to notify a customer that the fix is now ready, and the documentation writer may need to post a knowledge base article about the bug on the company's web site. A defect tracking system can enforce a workflow so that each person is reminded of his or her role at the appropriate time.

Large development teams have too much going on at a time for any one person to keep track of it all. Let's look at some of the issues that may be brewing underneath the surface of your project.