Do You Know Why You're Doing That?

by Jonathan Kohl

Ambiguity can kill software projects. If you don't have a clear idea of what you are trying to achieve, you can waste time figuring it out by trial and error. The vision for a software product is the starting point for a software project. If it is vague, it leads to confusion, duplication of effort, and wasted work. If you don't know why you are doing what you are doing, you may be busy, but you may not be contributing.

Wait a minute! Isn't it someone else's job to make sure the right product gets built? Isn't that what management is for? Shouldn't someone else worry about whether our software provides value to our users? Our job is to code, test, and perform other tasks that help us develop software, right?

Sure, management has a responsibility, but so do we technical workers. Business people rely on us to help them translate their product ideas into software, and it's our job to help them be successful. After all, that's what they pay us for. Helping ensure our projects are on track may sound like extra work, but a bit of effort can save a lot of heartache in the long run.

Here's an example: A company founder has a vision for a software project. He's a "big-picture thinker" who often sees opportunities others don't. He has the connections to raise money and the influence to gather the people who can translate his vision into working software.

This visionary may say, "We're going to build best-in-class educational software for universities." That sounds focused, doesn't it? It even identifies a target market. But, look more closely. The problem is that it describes a *product category,* not a real product. It's vague. But, we have enough to go on, don't we? We don't believe in "big design up front," anyway. The design will emerge over time, right? Unfortunately, that's often not how this kind of project plays out.

Programmers are brought on board. They are given a tight deadline and start working. It's unclear from a programming perspective what the team should build. But, team members think they will figure it out over time. They can just start writing code and tests and the product will appear. Besides, it's management's responsibility to tell them what to build. If it's wrong, it's management's fault!

> "If you don't know why you are doing what you are doing, you may be busy, but you may not be contributing."

Time passes. The original vision is still vague, but at least the programmers have created working software. They demonstrate it regularly to the business folks. But, it never does quite align with the product vision. Changes are made, and the programmers bring it back again. After several iterations, the emerging product is still as vague as the original idea. Sometimes, teams even burn through their entire budget doing this, and the project gets cancelled. If this is a startup, the executives must go back to investors and explain why they spent all their money without creating a product that can be sold. It can be enough to put a startup under.

What went wrong? We had a group of people working together at great expense to the company, without anyone really knowing what he was working on. We can't just blame management—on projects like this, we are all responsible. Often, if just one or two people speak up early in the project—saying, "I don't know why we're doing this," or, "Let's get clarification before we proceed"—many problems could be averted.

A few years ago, a friend of mine showed me what she called a "do-nothing machine." It was a solid wood block with a handle on it. Turn the handle and the machine just sat there, doing nothing. You could turn the handle quickly or turn the handle slowly—and it would still do nothing. You were doing work but not accomplishing anything. This is the danger on software projects that don't have a clear vision. You work hard, you fill up your time with tasks, but you might not actually accomplish anything useful.

I've talked to many business analysts, product managers, and even company founders who can't clearly communicate what their products do and who their customers are. When this happens, it doesn't seem to matter which methodology the development team uses—whether it is a waterfall, big-design-up-front project, or a lightweight, emergent-design agile project—the vague vision produces vague software.

Teams should nail down their product visions and basic goals before starting a project. It's cheaper and easier to figure out the direction to take at that point, when there are fewer people (and costs) involved. This isn't big design up front, where we try to predict the future and which can have equally poor results. It's about getting down to basics about what we are doing and why. Early, lightweight planning can really pay off.

With a clear vision, teams experience:

- Less wasted work—Teams know what needs to get done, so there's less churn.
- Increased productivity—Individuals have a better idea of what their roles and tasks are.
- Early warning signals—Input from the people doing the work can help identify problems early on.
- More meaningful work—Tasks

feel less pointless if they are aligned to goals.
- Improved communication and collaboration—A common vision, shared across teams, inspires openness.
- Increased chances to satisfy customers—When a project's vision and goals are aligned with the customer needs and goals, we are more likely to impress them.

Here are some simple tips to help get clarity on your projects:
- Don't start projects if there isn't a clear roadmap or vision.
- Work with visionaries to distill their vague ideas into something concrete.
- Don't be afraid to stop a project and work to get clarity to get it back on track.

To help clarify vision and goals, try the following:
- Describe in one sentence what your product does.
- Describe who your customers and users are.
- Explain how your customers and users use your software to complete tasks and reach their goals.
- Map your own work to how you are creating better software that satisfies and impresses your customers.

It can be difficult to hold off on technical work while spending time getting these basics together for a project. It also takes effort and creativity to seek out customers and find out how they use our software and why they are interested in buying it. It can even be a little bit scary to face the people who use our products. However, with a little creativity and persistence, you can get that information from your customers or, barring that, from people on your team who know the customers well. There is an old saying: *If you don't know where you are going, any road will get you there.* While we may not need to have a destination in mind in some endeavors, we rarely have the time and budget on software projects to figure out by trial and error what the products are. So, know your destination, and you will choose a smoother path. {end}

Have the Last Word!

If you have a point to make or a side to take on issues and trends that affect the industry, we want to hear from you.

We are looking for insightful, thought-provoking commentary for possible use as a Last Word column.

Please send an abstract to editors@bettersoftware.com. You will be notified if you are being considered for publication.

Index to Advertisers

Better Software (USPS: 019-578, ISSN: 1553-1929) is published seven times per year. Subscription rate is US $39.99 per year. A US $35 shipping charge is incurred for all non-US addresses. Payments to Software Quality Engineering must be made in US funds drawn from a US bank. For more information, contact info@bettersoftware.com or call 800.450.7854. Back issues may be purchased for $15 per issue (plus shipping). Volume discounts available. Periodicals Postage paid in Orange Park, FL, and other mailing offices. POSTMASTER: Send address changes to Better Software, 330 Corporate Way, Suite 300, Orange Park, FL 32073, info@bettersoftware.com.