Favor the Simple Over the Complex

Scott Davis
Broomfield, Colorado, U.S.

AS FAR AS I'M CONCERNED, my microwave oven only has one button: "add a minute." To boil a cup of water for my coffee, I press the button three times. To melt cheese on my crackers, one click. To warm up a flour tortilla, I press "add a minute" and then open the door after 15 seconds.

Would a one-button microwave oven ever make it out of the planning committee? Probably not. I can tell by the (never used) features on my microwave that the committee favored complexity over simplicity. Of course, they probably cloaked "complexity" in the euphemism "feature-rich." No one ever starts out with the goal of making a product that is unnecessarily complex. The complexity comes along accidentally.

Suppose that I have a slice of cold pizza that I want to warm up. According to the manufacturer's directions, I should press the "menu" button. I am now faced with the options "speedcook" or "reheat." (Um, "reheat," I guess, although I'm kind of hungry. I wonder if speedcook will be any faster than reheat?)

"Beverage," "pasta," "pizza," "plate of food," "sauce," or "soup"? (I choose "pizza," although it does have sauce on it, and it is on a plate.) "Deli/Fresh" or "Frozen"? (Neither, actually—it's leftover delivery pizza. I'll choose "Deli/Fresh," I guess.) "1 slice," "2 slices," "3 slices," or "4 slices"? I have no idea how much longer this interrogation will last, so I press Cancel and then the "add a minute" button.

What does this have to do with software development? As far as I'm concerned, Amazon.com only has one button: "one-click purchase." Oh, sure, I had to type in my address and my credit card number the first time I visited, but now I am one click away from my impulse buy.

Software generally solves complex problems. The question is how much of that inherent complexity are you forcing onto the end-user? Is your software a complexity amplifier? Great software is generally a complexity sink—it bears the brunt of the problem on behalf of the user instead of passing it along.

As a software project manager, are you a complexity sink or a complexity amplifier? The best ones absorb complexity from all sides—from the programmers, from the end-users, from management—and never amplify it. As the end-users generate seemingly contradictory requirements, your job is to help clean them up, rather than passing them blindly on to the developers. As the developers cite arcane technical reasons for not being able to fulfill a requirement, your job is to translate (absorb) that complexity and present the end-users with enough information to help them choose a different path.

How easy is it to use your application? How easy is it to add a new feature to your application? How easy is it to request a new feature? Report a bug? Deploy a new version? Roll back a bad version?

Simplicity doesn't happen accidentally. It needs to be actively cultivated. Complexity is what happens when you aren't paying attention.