Differences in Agile Methods: A Matter of Conscious Practices or Not?

Rebecca Wirfs-Brock
rebecca@wirfs-brock.com
www.wirfs-brock.com

Agile methods are infused with practices geared towards producing value while eliminating wasteful acts and artifacts. Key to most agile practices are customer involvement, short development iterations, and delivery of working code. Yet there is a profound difference among agile methods (and agile thought leaders) in how much thinking and reflection they encourage.

Christopher Alexander in Notes on the Synthesis of Form contrasts unselfconscious and conscious design processes. Alexander characterizes unselfconscious design cultures as places where:

- Building skills are learned informally
- Builders live in the systems they create
- The unspoken rules are of great complexity and are rigidly maintained
- There is a way to do things and a way not to do things
- There is firmly set tradition which is accepted without question
- Changes are made locally whenever something needs improvement

Unselfconscious building processes result in well-fitting forms that persist in equilibrium with the system they exist within. Structures produced in such a system are, according to Alexander, "not the work of individuals, and their success does not depend on any one man's artistry, but only on the artist's place within the process" The major downside is that when presented with complicated (new choices) unselfconscious processes aren't readily adaptable. Unselfconscious builders solve problems in a narrow, well-known context.

In contrast, in a selfconscious design culture things are done very differently:

- Principles and practices are distilled into systems of concepts. There are schools of thought, teachers who teach them, and students who debate them
- Builders think in terms of abstractions, categorization, and explicit principles
- Buildings are more permanent and frequent repair and readjustment is less common
- Construction isn't in the hands of the inhabitants

As a consequence, Alexander emphasizes that reaction to failure is less direct. Failures are reported and described before they are fixed—but only by specialists. With a selfconscious culture, concepts and principles and practices create blinders that can lead to failure: "concepts control…perception of fit and misfit—until..he sees nothing but deviations from his conceptual dogma, and loses.. the mental opportunity to frame his problems more appropriately."

Of course, Alexander is referring to builders of buildings while we build complex software. There are parallels, but we shouldn't blindly look for them. As software developers we don't have a long culture or tradition of building things repeatedly to draw upon. Our systems are not rebuilt again and again and again. Our materials and techniques are rapidly changing. And so are our practices.

So how do different agile methods deal with selfconsciousness? No agile method is purely unselfconscious. All agile methods have written practices. Fortunately most agile authors write slim, approachable books. But some agile methods clearly provide frameworks for thinking while others offer an integrated set of techniques that are learned largely practiced unselfconsciously. Agile methods can be characterized along several dimensions:

- Being more or less open—where change and adaptation by practicing builders is actively encouraged
- Consisting of practices accompanied by explicit principles and values or not
- Being integrated systems adopted "whole cloth" or collections of practices

- Providing conceptual frameworks for decision-making and thinking or explicit practices and techniques

Some agilists would like practitioners to be mostly selfconscious. Yet many builders of software will never be happy with closed world views:

> One of my issues with the leaders of movements like SCRUM and XP is that they believe that you should be using their techniques verbatim— I once heard one of them say "You're either doing XP exactly, or you're NOT agile!"—a reviewer of a book on Amazon.com

Others become upset that when authors wax philosophical:

> …[title purposefully deleted] is one of the few software books that I would return. It is full of abstract fluff (which is repeated in several different ways). This reminds me of an academic textbook. Just take a simple concept and invent some terminology to wrap around it in several different ways. —another reviewer on Amazon.com

When I first read Alexander's words, I got angry. He was clearly biased towards unselfconscious practices and against forming conscious practices. I tend favor agile methods that encourage critical thinking and exercising judgment within a value framework. But I'm too much of a thinker to blindly do anything without knowing the "why behind it" so I can adapt when necessary! I'm the proverbial two-year old who questions "why, why, why?" I resonate with authors that encourage critical thinking and exercising judgment as well as distillation of their principles and practices. I want to have the best of both worlds.

I caution you (and myself) against placing black and white value judgments on selfconscious or unselfconscious building processes. There is a time and place for both. I don't want to think when I slam on the brakes to avoid a collision. Sometimes I want to think about what I have for dinner and how tastes and textures of food complement each other. Sometimes I don't. Not every act or practice needs to be questioned. Software isn't always like driving a car in rush hour traffic or preparing a gourmet meal. In the software cultures I have been part of, developers freely quibble with various methods and practices. Most always take every practice and fit it into their own context.

I encourage agile thought leaders to be explicit about the contexts and cultures they are intimate with. Say what your method doesn't address as well as what practices are important and vital. In our hyper-connected age, it is hard for someone to follow ideas "correctly" after skimming a book, reading an article, or listening to an entertaining speech. You won't be there to guide practitioners. Yet your ideas are out there making an impact. Think long and hard about whether you want to encourage unselfconsciousness (and what you want to be codified into a self-conscious practice). Think about whether you want to be open and adaptable and explain why. Work hard at explaining both your core values and core practices. When you learn more, share these learnings with others.

To practitioners of agile methods, I offer some questions for sorting out various methods: Do you want a system of practices or a framework for making decisions? How much do you want to add to or adopt a set of practices? How much reflection on your practices can or will you tolerate? Are you looking for answers or approaches?

References

Christopher Alexander, Notes on the Synthesis of Form, Harvard University Press, 1964