A Social Science of Design

Brian Marick (marick@testing.com, www.testing.com)

My position is that one Science of Design should be a science of *people doing design*. In the beginning of a design project, relevant people agree there's no design. Eventually (we hope), relevant people agree there is. What happens in between? What activities do people characteristically perform? What resources do they use, and how? How do their actions depend on their context? How do they marshal support, negotiate agreement, or bury disagreement? How do people become "relevant people"? How do the answers to all these questions contribute to successful vs. unsuccessful vs. unfinished designs?

In short, at least one Science of Design should be a *social* science, more akin to anthropology or social studies of scientific practice than to physics.

Examples of research projects.

Let me be concrete. Here are two examples of the sort of research that should be done.

In vitro. Some people are, by common consent, great designers. Put them together in small groups and have them work on something important to them. Give them time to do a serious amount of work. Observe them carefully and record much data for later analysis (videotape their work, archive intermediate copies, have ethnographers talk to them about what they're doing, even record keystrokes).

In vivo. Thousands of groups are doing design right now. Send anthropologists (or people using anthropologically-inspired methods) to selected groups. They will not be able to gather as much data as in the *in vitro* case, but they will have at least two advantages. First, the less sheltered environment will reveal new sources of design influence. Second, the visits can be periodic, allowing observation of how decisions ripple forward through time.

Two risks to relevance - and how to cope. What would count as success.

In 1934, Karl Popper published *The Logic of Scientific Discovery*. It's an elegant description of what it means for a scientific process to be rational. Scientists propose testable hypotheses. Other scientists test them through experiment. When an experiment refutes a hypothesis, the rational scientist discards it and searches for a new one.

A small problem: real scientists don't behave like that. Kuhn (1970) shows how anomalies (refuted hypotheses) are ignored until *something* (precisely what, he doesn't specify) forces a science into a time of crisis. Lakatos gives further examples (Lakatos 1987 and Motterlini 1999). Feyerabend (1993) is the most exuberant in claiming that great scientists will, under necessity, ignore all rules and, indeed, embrace logically contradictory situations. According to these authors, Popper wrote a recipe for failure.

Here's the first risk. Software development attracts people fond of tidy logical structures. We see that in the many proposals for software processes that feel *so close* to being automatable, that try to wall off the human element. I fear the NSF is much more likely to fund people who will write *The Logic of Design Discovery* than (following Feyerabend) *Against Design Method*. If the above authors - and others like them - are correct, and if design shares characteristics with science, that funding will not improve the state of actual software design.

The solution is to make use of the techniques - and, if possible, the researchers - who already study scientific practice. (Pickering 1992 and Latour 1987 are good examples of this thread of research.) It's my belief that what these authors have accomplished is applicable to software

design. For example, Pickering (1995) tells the story of Hamilton's discovery of quaternions (a mathematical abstraction). Last summer, I spoke with Ward Cunningham, surely one of today's great designers. He told the story of how he and his team invented "advancers"[1], an abstraction in a bond trading application. I was struck by the parallels between the two stories - both can be described in Pickering's terminology of resistance, accommodation, bridging, and extension. If designers were familiar with those ideas, might they not design better? I think so.

But… note my qualification: "*if* designers were familiar with those ideas…" It appears that social studies of science have had little effect on the actual practice of science. There are a variety of reasons. One is that they've gotten tangled up in "the science wars"[2], specifically the issue of whether the social processes of science call into question its claims about objective reality. That shouldn't be a problem in software: it's hard to see many people getting upset at the notion that a Smalltalk class named Advancer is a social invention, not a reflection of reality.

But another reason the studies haven't affected practice, it seems to me, is that scientists aren't their intended audience, any more than sociologists are the intended audience for an article in *Physics of Plasmas*. So the literature is fairly inaccessible to software people (except for oddballs like me). And here's the second risk: the same thing could happen to a social science of design.

The NSF should make it clear that funding a social science of design is a bad use of the public's money unless *one* of the results is a publication with the effect on practice of Gamma et. al.'s *Design Patterns* (1995). Funding should include money for popularizing the science of design. A successful research program would win the Jolt software productivity award as well as help someone gain tenure.

My background.

I am an independent software consultant, best known in the testing community. My main current interest is in how checked examples (tests) can be used to provoke programmers to write the right program. I am the author of *The Craft of Software Testing* (1995), one of the authors of the "Manifesto for Agile Software Development" (www.agilemanifesto.org), a newly-elected board member for the Agile Alliance nonprofit (www.agilealliance.org), program chair for the 10th Pattern Languages of Programs conference, an editor for Software Testing and Quality Engineering magazine (www.stqemagazine.com), and long-time member of Ralph Johnson's software architecture reading group.

References.

Feyerabend, Paul. 1993. *Against Method* (3rd edition).

Gamma, Helm, Johnson, and Vlissades. 1995. *Design Patterns*

Kuhn, Thomas. 1970. *The Structure of Scientific Revolutions* (2nd edition).

Lakatos, Imre. 1978. *The Methodology of Scientific Research Programmes*.

Latour, Bruno. 1987. *Science in Action*.

Motterlini, Matteo (ed). 1999. *For and Against Method*. See specifically "Lectures on Scientific Method".

Pickering, Andrew (ed). 1992. *Science as Practice and Culture*.

Pickering, Andrew. 1995. *The Mangle of Practice: Time, Agency, and Science*.

Popper, Karl. 1934. *The Logic of Scientific Discovery*.

---

[1] c2.com/cgi/wiki?WhatIsAnAdvancer

[2] members.tripod.com/ScienceWars