The Social Construction of Pattern Languages

Pascal Costanza*

June 24, 2002

"Mathematics is what mathematicians do."

Since many of my thoughts about this topics have already been articulated by other authors I begin this position statement by citing from a book that I have recently read and that I believe should help in the discussion of the focus group's subject.

The following material is taken from "The Social Construction of What?" by Ian Hacking [2]. Omissions, modifications and additional remarks by me are enclosed in square brackets.

1 The Social Construction of What? (excerpt)

1.1 Interactive Kinds

There is a big difference between quarks and [programmers].[1] [Programmers] are conscious, self-conscious, very aware of their social environment, less articulate than many adults, perhaps, but, in a word, aware. People, including [programmers], are agents, they act, as the philosophers say, under descriptions. The courses of action they choose, and indeed their ways of being, are by no means independent of the available descriptions under which they may act. Likewise, we experience ourselves in the world as being persons of various kinds. [...]

The awareness may be personal, but more commonly is an awareness shared and developed within a group of people, embedded in practices and institutions to which they are assigned in virtue of the way in which they are classified.

We are especially concerned with classifications that, when known by people or by those around them, and put to work in institutions, change the ways in which individuals experience themselves – and may even lead people to evolve their feelings and behavior in part because they are so classified. Such kinds (of people and their behavior) are interactive kinds. This ugly phrase has the merit of recalling actors, agency and action. The *inter* may suggest the way in which the classification and the individual classified may interact, the way in which the actors may become self-aware as being of a kind, if only because of being treated or institutionalized as of that kind, and so experiencing themselves in that way.

*well, at least, partially ;-)

[1] In the original text children are used as an example.

1.2 Indifferent Kinds

[...] I use [the word "kind"] to draw attention to the principle of classification, to the kind itself, which interacts with those classified. And vice-versa, of course, it is people who interact with the classification.

There can be strong interactions. What was known about people of a kind may become false because people of that kind have changed in virtue of how they have been classified, what they believe about themselves, or because of how they have been treated as so classified. There is a looping effect. [...]

There is a constant drive in the social and psychological sciences to emulate the natural sciences, and to produce true natural kinds of people. This is evidently true for basic research on pathologies such as schizophrenia and autism, but it is also, at present, equally true for some but only some investigators who study [for example] violent crime (is that an innate and heritable propensity?). There is a picture of an object to be searched out, the right kind, the kind that is true to nature, a fixed target if only we can get there. But perhaps it is a moving target, just because of the looping effect of human kinds? That is, new knowledge about "the criminal" [...] becomes known to the people classified, changes the way these individuals behave, and loops back to force changes in the classifications and knowledge about them.

The notion of an interactive kind is fuzzy but not useless. Plenty of classifications differ fundamentally from any of the human kinds just mentioned. Quarks are not aware. A few of them may be affected by what people do to them in accelerators. Our knowledge about quarks affects quarks, but not because they become aware of what we know, and act accordingly. [...] The classification "quark" is indifferent in the sense that calling a quark a quark makes no difference to the quark.

2 Computer Science: Interactive or Indifferent?

Since important fields within computer science have their roots in applied mathematics, there is a tendency in this field to treat their subjects as being of indifferent kinds. For example, it makes sense to analyse algorithms and determine their complexity in terms of runtime and memory consumption.

However, there is clearly a danger involved in keeping up this practice when in fact it is not appropriate anymore. For example, the failure of traditional software development methodologies might be due to the fact that programmers had been treated as being of indifferent kinds. The current popularity of "Extreme Programming" and "Agile Software Development" [1] can be understood as a change of classification of programmers from an indifferent kind to an interactive kind.

It is tempting to treat our artifacts as indifferent. Of course, a piece of code is not aware and does not care if it is described as an example of good object-oriented style or not. However, we may start to see things in a different light if we take the creator of this particular piece of code into account – a programmer clearly is aware and cares about whether he/she is described as being a good or bad programmer.

2.1 Pattern Languages

So how does this all relate to pattern languages? In my opinion, pattern languages should be regarded as interactive kinds because

- they are written by authors who care about what other poeple think
- they describe ways of how to apply expert knowledge – so there is a second level of interaction between them and those who apply these pattern languages
- they contain expert knowledge as such – and this expert knowledge is also drawn from interactions with "the outside world".

Here are my preliminary conclusions.

- A lot of terms in computer sciene do not have clear definitions. (Examples: What is an "object"? What is a "component"? What is a "crosscutting concern"?) This might be because computer science nowadays mainly deals with interactive kinds and therefore, sharp definitions do not make sense.
- Terms like "pattern", "pattern language", "the quality without a name" might continue to have fuzzy descriptions for a very long time because of the same reason.
- Whenever we try to fix a definition we have to take into account that people might react in totally unexpected ways, and this may change the basis for our definitions. In other words: it is not clear if we will ever reach a fixed point.
- This is not necessarily a problem, but might even be a good thing.

References

[1] Alistar Cockburn *Agile Software Development.* Addison-Wesley, 2002.

[2] Ian Hacking *The Social Construction of What?.* Harvard University Press, 1999.