Enculturating Extreme Programmers

David M. West
New Mexico Highlands University
School of Business
Las Vegas, NM 87701 USA
+1 505 454 3173
dwest@cs.nmhu.edu

ABSTRACT
XP is a culture not a method. As a culture, XP has significant appeal to those seeking a more humane and effective way to develop software. Recognition of XP as a culture, and establishing mechanisms for expanding that culture, is essential for success. However, characterization of XP as a culture, by outsiders, is a significant threat to adoption, unless the XP community acknowledges being a culture and uses that self-awareness as a tool for promotion. This paper will be devoted to arguing the preceding assertions and suggesting enculturation as a way to address the challenges that arise from them.

Keywords
XP, culture, threats, success factors

1 XP, A CULTURE NOT A METHOD

XP (as an exemplar of light or agile methods in general) requires developers to perform certain actions, generate certain artifacts, and do so in some kind of order. Therefore it can be said that XP is a "method."

None of the actions and none of the artifacts are unique to XP. Even the order in which things are done is pretty standard for any software development project using any method.

What distinguishes XP from mere methods are the following factors:

- Worldview - development in the "Garden of Enough," a world analogous to the environment of the *Forest People* described by anthropologist Colin Trumbull as cited by Kent Beck.[1]
- Set of values. "Communication, Simplicity, Feedback, Courage."[2]
- List of principles, Kent Beck asserts five basic principles and ten that are "less central."[3]
- Styles of human-human interaction. These range from negotiation of work to story telling, to pair programming.

The worldview, values, and principles generate, in turn, norms, roles, and patterns of behavior.

All of the above are key factors in the definition of culture as an anthropological concept.[4]

Recognizing that XP is a culture is really pretty obvious. I am not the first or only person to make this kind of assertion. Even the founders of XP (this time I mean XP explicitly not the entire category of agile methods) admit that the essence of XP is cultural and philosophical in nature.

However obvious it might be, explicitly acknowledging XP as a culture yields some important benefits. Recognition of how XP relates to similar "reform" movements in the realm of software; sensitivity to what might befall XP in the future; and, insights that suggest ways to advance the XP cause in the most effective manner, are examples.

2 XP AND CULTURAL TRADITIONS

Of course XP is not the first innovation in software development that involves, at its core, a change in culture. Objects and patterns are two recent and very similar examples.

In fact, it is possible to see XP as merely the most recent example of a long-standing cultural tradition with philosophical roots that predate the advent of computing. Some examples of these antecedents:

- Robert L. Glass[5] wrote of two software development cultures: Greeks (creative, informal, artists) and Roman (formal, high ceremony, engineers).
- The field of Artificial Intelligence was the arena for conflicts between the "fuzzies" and the "neats."
- Objects had both "west coast" (Smalltalk) and "east coast" (C++) schools.
- This author has written elsewhere[6] of the conflict between Hermeneutic and Formalist cultures in both computer science and software engineering.

More recently, Michael McCormick notes that:[7]

> *"What XP uncovered (again) is an ancient, sociological San Andreas Fault that runs under the software community - programming versus software engineering (a.k.a. the scruffy hackers versus the tweedy computer scientists). XP is only the latest eruption between opposing continents."*

Not only is XP itself a culture, it has roots, acknowledged or not, in a much larger cultural tradition. XP is the latest assertion of the view that people matter. XP is the latest challenger to the dominant (and hostile) computing and software engineering culture.

If XP is the "latest eruption between opposing continents" (as both McCormick and I believe it is) is there any reason to expect XP to have any more of a significant or lasting effect on the way software is developed than RAD, objects, or patterns?

Alan Kay claims the object revolution hasn't happened yet. Christopher Alexander believes that the Patterns Movement in software "missed the point." Will Kent Beck keynote some future OOPSLA conference with a lament about how XP was never understood or given a chance to effect real change?

3 CULTURES IN CONFLICT

XP is the latest attempt to make software development more humane, creative, a-formal, and effective. This is the source of its greatest appeal to working developers. As such it is in direct conflict with the "official" and "management sanctioned" approach to software development.

XP is already evoking the same hostile, demeaning, and emotional reactions from computer scientists, software engineers, "pragmatists," and academics that were used to attack all previous challenges to the dominance of the formalist, scientific, software development culture.

Once again the gauntlet has been laid down. What might we expect from the "religious wars" this time?

Defeat is always possible. Some highly influential person or group might put forward an argument that effectively destroys the XP approach the way that Marvin Minsky derailed research in neural networks.

Retreat is also possible, and more likely. Unable to sway the majority, proponents of XP, "take their ball and go home;" becoming a kind of guerilla cult. (Some think this is what has happened to the Smalltalk community.)

The most likely outcome, however, is some sort of cooption. The mainstream culture takes on the form of XP, while ignoring its essence. This is exactly what happened to LISP based RAD, Smalltalk and behavior based objects, and design patterns. It is also evident in the efforts to incorporate XP in RUP (Rational Unified Process) or to show that XP is a degenerate [*note the pejorative term*] form of RUP.

Another form of cooption, and one that is more insidious because it appears to be more accepting, is found in the arguments advanced by McCormick in the article cited earlier.

McCormick suggests that XP should be added to the developer's toolkit along with all the other processes and formalisms. XP could then be used when appropriate and ignored when not. XP is ceded status as a tool, but nothing more than a tool. Lip service is paid to the ideas and ideals of XP without any danger that those ideas and ideals will have any serious effect. Certainly there is no danger of using those ideas and ideals to change the way "pragmatists" actually think or feel. It is like telling rowdy children to "go do

whatever you want, just stop yelling at each other."

History suggests that one or both forms of cooption comprise the likely future of XP. The culture represented by XP will return to the background until another eruption in another form takes place.

Unless … unless, of course, the proponents of XP can find a way to bring about some real cultural change.

4 CULTURAL CHANGE

Cultures do change. The introduction of new technologies as well as new belief systems can be an instrument of such change.

Lauriston Sharp eloquently describes[8] how the introduction of simple steel axes destroy an aboriginal culture and a continent spanning economic and trade system. Introduction of the automobile resulted in profound changes including the "sexual revolution" and a plethora of Los Angeles-like cityscapes.

XP is not a discrete technology like a steel axe, automobile, or computer. It is more akin to a belief system - like a religion. To effect true culture change, to succeed to any significant extent, XP needs to adopt and execute some of the same factors that have allowed religions to flourish.

Be bold - go ahead and admit that XP is a "religion" of sorts. In doing so XP advocates are usurping the negative power of a characterization made of them by their opponents. Following the example of blacks appropriating the N-word for their own use, or gays the queer label.

Saying that XP is a 'religion' is asserting the claim that XP is primarily a philosophy, an ethos, and a community (a culture) that values humanity - that of software developers and software users - above all else. In direct contrast to 'idolaters' bowing before the altars of process, machine efficiency, or mathematical formalism.

This leads, in turn, to the claim that adherents of the XP religion/culture can practice their craft, software development, and generate superior results. Not because XP is a better method, technique, or tool, but because XP is founded on better ideas, ideals, and principles.

Promise immediate and tangible benefits - then deliver something concrete and of real value to potential adherents - something other than "The Truth." Buddhism (and later Islam) offered Hindus a means of escaping the rigid caste system. Christianity succeeded in large part because women converts significantly increased their social status (from mere property to actual human person; although, unfortunately, still a second class person).

Actually, XP already offers this kind of benefit to developers - the promise of a working environment quite different from the "cubicle hell" of Dilbert comics and just as different from the hyperactive (but adolescent and self-indulgent) extremes of the "dot com" playrooms. XP offers *koyaanisqatsi* (Hopi for life-in-balance) for software developers, managers, and users.

Organize, Evangelize, and Enculturate - implement the mechanisms necessary to bring people into the culture and instill the culture in those you attract. Anthropologists use the term 'enculturation' to describe the mechanisms and process by which people become full members of a culture.

5 ENCULTURATION

Enculturation is the social process, facilitated by a set of artifacts, used to teach and learn the culture of a group. Three aspects of enculturation need to be made explicit.

Establish a Canon. Given that we are a literate society the Canon should exist primarily in written form while acknowledging the value and importance of an accompanying oral tradition.

The Addison-Wesley XP Series is not a sufficient Canon even when coupled with the all the other publications of XP proponents. At best these writing constitute a kind of "New Testament" for XP. The Canon must also include predecessor works, like the writings of Robert L. Glass on software creativity and the work of James Coplein on organizational patterns. Acknowledgment must be made of philosophical roots, the work of Gadamer, Heidegger, and Husserl, for example.

Think visually and artistically. Develop a body of iconographic imagery and art reflective of the XP culture. The purpose of these artifacts is to create a visual environment that reminds members of the culture about key ideas and ideals.

One of the ways that any culture maintains itself and maintains relative equilibrium is via the establishment of tangible iconographic environment. This environment embodies a variant of the "publish-subscribe" pattern.

Publish-subscribe is a way to notify a collection of interested parties of changes in an object of mutual concern. Implicit in the pattern is the idea of keeping that same collection of parties in-synch with the state of the common object of concern.

Human cultures employ this pattern in many forms.

For example, in my office I have a collection of statuary representing deities from many different cultures. Isis shares shelf space with Lilith, Pan, assorted Kachinas, Hannuman, Ganesh, various images of the Buddha, and many others. On my wall I have a Thangka painting that graphically depicts the entire cosmology/philosophy of Tibetan Buddhism. The purpose of all of these artifacts - to those whose culture they reflect - is to create a common reference point - and a kind of constant reminder - of who they are and what they believe.

Secular examples are also available - the ubiquitous THINK sign at IBM or Steve Jobs' blue jeans and sandals.

Be Social. The Canon and the iconographic environment are artifacts useful in the transmission and maintenance of a culture. But artifacts alone are insufficient.

> "… culture is socially learned. To say that culture is socially learned is to say that individuals acquire it from others … To say that culture is socially learned is to say that people do not learn culture primarily by trial and error learning. The main way (they) learn is by observation, imitation, communication, and inference …"[9]

Culture is learned only via participatory interactions with other people already living that culture. Key word here is 'living.' An anthropologist who wants to learn about another culture must spend an extended period of time engaged in participant-observation in that culture. Participant-observation retains some element of detachment (after all, there are Ph. D. theses to be written) but stresses the participation aspect. The anthropologist lives with, eats with, works with, worships with, and parties with the people she is trying to understand.

For XP, this implies that the only way you can become an XP developer is by working with other XP'ers.

Conferences, like *XP Universe*, are akin to tent revival meetings, useful for spreading the word and generating enthusiasm but not as effective means for cultural transmission. (Even if they were coupled with a solid Canon and a well-developed iconography.)

Enculturation is a slow process. It takes a child twenty to thirty years to become proficient in the culture into which he was born. For an adult to learn a culture other than the one into which he or she was born takes even longer. Ask any expatriate. Few will admit to knowing their adopted culture, "like a native," despite the fact that they may have spent decades living in that new environment.

Ignoring the need to allow sufficient time for enculturation can lead to disaster. At the risk of sounding cynical - the lure of notoriety, lucrative consulting contracts, and remunerative training programs tempts too many into promising results in timeframes they know to be impossible. The more profound the innovation or the more extensive the cultural change, the greater the amount of time required.

Even if there is no hyperbole in terms of the degree of innovation in a new culture or technology, exaggerating the ease and speed of enculturation alone is sufficient to invite a backlash. When overselling is detected (usually moments after the sales pitch has been made) the innovation is dismissed and the culture devalued.

As slow as full enculturation might be - there are ways to propagate the innovative culture a bit faster and more efficiently than forcing everyone

to spend a year working on a real world project with one of the XP founders.

[Although, a bit of cultural iconography might be developed, based on personal association. Mathematicians speak of an "Erdos Number" reflective of their association with the noted mathematician, Paul Erdos. An Erdos Number of '1' meant you had co-authored a paper with Paul, a '2' meant you had co-authored a paper with someone having an Erdos Number of '1', and so forth. Kent Beck claims that XP (specifically XP and not all agile methods) is an expression of Ward Cunningham's natural way of working. So perhaps a Cunningham Number might be a fun, cultural, thing to establish.]

Even though most enculturation takes place implicitly and at-large in the culture itself, most cultures establish more explicit mechanisms for cultural transmission. Schools for example.

This suggests at least two mechanisms for the propagation of the XP culture. One is an apprenticeship program; another, the establishment of *gompas* (Tibetan Buddhist communities similar to seminaries or working monasteries).

Just south of our conference site, Ken Auer (http://rolemodelsoft.com) is implementing his ideas about apprenticeships for software developers. His ideas are based on the example of medieval guilds. He brings young people into his business as apprentices, teaches them a highly values-based (cultural) approach to development along with the technical skills required to develop software.

A basis for the *gompa* approach might be found in the attempt by some members of the Hillside Group to establish a Master of Fine Arts in Software degree at a major university.

An MFA degree would recognize that software development is as much an art as it is a science. More importantly it would incorporate the "studio model" of teaching and learning. Students would work closely with teachers and more advanced students in the creation of their "art." They would learn by doing. They would learn by *observation, imitation, communication, and inference.* They would learn the culture as well as skills, tools, and techniques.

Not being a member of the semi-closed Hillside Group, I cannot speak with authority on the exact nature of their program. However, if they have not considered a residency component (similar to a seminary, an honors house, or an athletic dorm) for their program I would suggest that such a component be given serious consideration. And if the XP community wishes to establish a 'school program' to propagate XP I would suggest they do the same thing.

The extent of cultural change required by XP is significant. The forces that will mitigate and contravene any classroom or even studio-based experience are enormous. The additional cultural reinforcement provided by being a member of a live-in community will likely be required for success.

Enculturation is a very real need if XP is to succeed where other, often very similar, efforts to reform software development have failed.

6 CONCLUSIONS

XP is a culture, not a method, not a tool, not a set of techniques.

XP will succeed in overcoming the forces of rejection and cooption to the extent that it discovers ways propagate itself that are solidly grounded in its essence as a culture.

There are concrete and feasible ways to bring about the enculturation of a large community of XP developers, even though most of those means are slower and more labor intensive than might be wished.

7 INFORMATION AND QUESTIONS

For more information, contact: dwest@cs.nmhu.edu

REFERENCES

[1] Beck, Kent. *Extreme Programming Explained" Embrace Change*. Boston: Addison-Wesley. 2000. xviii-xix.
[2] *ibid, 29*
[3] *ibid*, 37-39

[4] Peoples, James and Garrick Bailey. *Humanity. 5th Edition*. Belmont, CA: Wadsworth. 2000. 16-24.
[5] Glass, Robert L. *Software Creativity*. Englewood Cliffs, NJ: Prentice Hall. 1995.
[6] West, David. Hermeneutic Computer Science. *Communications of the ACM*, 40(4) April 1997, 115-116.
[7] McCormick, Michael. Programming Extremism. *Communications of the ACM* 44(6), June 2001, 109-110.
[8] Sharp, L. 1980. Steel Axes for Stone-Age Australians. In, *Conformity and Conflict*, eds. J. P. Spradley and D. W. McCurdy, 345-359. Boston: Little, Brown, & Company. [Reprinted from *Human Organization*, 1952(11) 17-22.]
[9] Peoples. *op. cit*., 17-18.