

Is It Agile or Software Anthropology?



Monday night I had the opportunity to attend an Agile Groupies meeting. It's a semi-regular gathering of folks (developers, business analysts, product managers, etc.) interested in a specific approach, **Agile**, toward software

development.

For those of you not familiar, Agile development focuses on smaller development teams, working on smaller deliverables, in highly iterative, somewhat less structured approach. Part of the thinking is that if you are delivering in smaller, more discrete 'chunks' of working product, that the overall process will be more 'agile,' more adaptive to ongoing change during a product's overall life cycle. The idea is to move away from 'heavy' less flexible disciplines, and back toward lighter, freer, approaches.

For this meeting, the discussion focused around panelists discussing how Business Analysts work with the development team. At a very high level, a Business Analysts (BA) represents customer needs, and formally crafts the requirement documents (we need the product to accept X as input, look like Y, and output Z).

The discussions were interesting. The challenges varied by size of the organization represented, including how well BA inputs truly

communicated what needed to be done, as well as where Agile methods frequently had been seen to fail.

What I've found is that Agile development meetings tend to migrate into a discussion of how organizations communicate internally. This meeting was no exception. Key issues were quality of BA deliverables; Development's keeping track of dependencies; mapping work against planned deliverables—either within a team or when working with other teams; and, anticipating or working around structural 'platform' issues.

The recurring theme though was—how individuals and teams communicate—in order to overcome their difficulties. The examples are really not significantly different than with any other software development team. The quality and completeness of BA deliverables (including acceptance specifics), development estimation and progress tracking, and including such things as user acceptance testing (or lack thereof), are the same regardless of the specific discipline(s) used within a software organization.

The key to success is clear: Smooth, rapid, delivery of a product demands effective, clear, and complete communication between all involved parties. It is better to risk over-communicating than the alternative. And, only through frequent communication can teams hope to adapt to change (they need to know about it).

Maybe we should call these Software Anthropology meetings.

(image credit: *Artiom (Artem) Chernyshevych*)