Commonalities of Agile Methodologies

Pete McBreen's position paper

Market Positioning - starting with a smile

A commonality I note between all of the agile methods is that they are trying to position themselves into a different space than both the Unified Process and Extreme Programming. This has been a massive challenge for the agile methods because of the massive amount of coverage that the Unified Method has been getting from the UML world and the grassroots awareness of Extreme Programming.

In effect, the label "agile method" is an attempt to create a distinction between these methods and UP/XP, when both UP and XP are both applicable in roughly the same methodology space as the agile methods. Having to position themselves as Not-UP and Not-XP has made it harder for these agile methods to achieve very widespread usage.

DSDM is an exception to this generalization, one possible explanation is that because DSDM predates XP, it achieved popularity by occupying the not-heavyweight, not-overkill process space that XP now dominates.

Radically Incremental Development

All of the Agile approaches practice incremental requirements capture. This means that a subset of application functionality can be released before the details of all of the requirements are known. Although the Agile approaches share this practice with the Unified Process, it is practiced more aggressively in the Agile methods.

The Agile methods expect the design of the application to evolve as new feature are identified, included into the requirement set and eventually implemented. This makes organizations that are more used to a waterfall style of development very uncomfortable.

Some developers have a hard time with this type of incremental development because they want to know exactly what it is they are supposed to be building before they start doing any design work.

Dancing on the edge of uncertainty

The agile methods seem to be taking the position that "even if it were possible to fully define an application up front, it would still be better to work as if the requirements were emergent." The idea being that some aspects of software development are unknown and unknowable early in a project and that this is a good thing.

Embracing this uncertainty is a common theme in all of the agile methods.

Finding room for the generalist

Agile methods have fewer roles than the more traditional processes. This means that the agile methods support the idea of a generalist that has a greater involvement in the complete project compared to the specialist roles defined in traditional processes.

Although the different agile methods use different names for this "just a programmer" and "minimizing handoffs" the intention is to improve overall involvement in, and commitment to, the complete project.

Human scale software development

All of the agile methods seem to accept the idea that it is much better to have small, social teams that interact on a daily basis. In effect, they all attempt to build a close knit community around the project.

Ideally this community involves both the developers and the key business users. Scrum even goes as far as telling stores about those who are truly committed as opposed to those who are merely involved “a pig and a chicken agree to set up a restaurant...”

An interesting side effect of this community building is that it is hard to do a low involvement agile process.

Making life hell for the Quality Assurance specialist

Agile methods by their very nature require a different form of quality assurance. Traditional quality assurance procedures are much less relevant because an agile process leaves much less of a paper trail than a traditional process. After all, how can the QA specialist compare the application to the documented requirements, when the written requirements are “just a promise for a future conversation”?

Agile methods all make the key business users a very strong partner in assuring quality. Rather than leaving quality to the professionals, agile projects make these key users responsible for ensuring that the application is fit for purpose.