Position Paper for the OOPSLA 2003 Workshop "Are Agile Methodologies Really Different?"

Jens Coldewey
Coldewey Consulting
Curd-Jürgens-Str. 4
D-81739 München
Germany
Tel: +49-700-26533939
Fax: +49-89-74995703
email: jens_coldewey@acm.org
http://www.coldewey.com

Approaches of a Methodology

Besides considering the processes a methodology supposes to use, another interesting topic for studies is the model that underleis the methodology. Though this is closely connected to the value system, the model bears no ethical judgement, as a value system does.

There are two popular models for projects: The control model and the systemic model. The control model assumes that there are centralized steering roles in a project, skilled individuals who plan and manage the project. These roles include the project manager, the architect, chief programmer, and so on. With this approach success and failure of the project mainly depends on the performance of these controlling roles. I will refer to these roles as the „managers", though this includes all kind of management and controlling. Methodologies using this approach usually concentrate on elements that support this structure:

- Enhancing communication between the managers and the operational team members, such as programmers.
- Providing reporting and controlling structures to enable the managers to get a good picture of the project status.
- Ensuring that the operational work is done as uniform as possible to ease planning, prediction, and controlling. Team members are assumed to play given roles and to be exchangeable „resources".
- Clear separation of responsiblities.

The ultimate extreme of this appraoch is Frederick Taylors „Scientific Management" that set the ground for the economic boost of the last century. Most traditional processes follow this model.

An alternative approach is to view a project as complex system. Complex systems have some characteristics, command-control structures don't have: Their behaviour is hard to predict and they are very flexible. To contol a complex system doesn't mean to predict and control every single action in it. Rather it means to understand the structure and interrelations of the system and set them up so the the system is able to adapt itself to the needs. Systems Thinking provides the appropriate toolset for this kind of approach.

Methodologies based on systems thinking — no matter whether consciously or not — show some typical properties:

- Few rules. Most of the rules establish feedback loops and communication to ensure all parts of the project are still heading towards the project goal
- Feedback and communication are designed to be as efficient as possible, rather than being trackable or controllable.
- Few if any centralized or hierarchical roles. Team members are assumed to be well-trained, responsible individuals who are able to organize themselves

Comparing the Approaches of Agile Methodologies

The following list checks to which approach the different agile methodologies belong:

- *Adaptive Software Development:* ASD is the only methodology that is explicitly based on the theory of complex adaptive systems. It restricts itself to establishing a feedback loop („Speculate, Collaborate, Learn“) and setting up the environment for efficient project work. A classical representative of the systems approach.
- *Chrystal Methodologies:* The major commonality of all Chrystal Methodologies is a process check workshop at least twice per increment. In addition Cockburn delivers seven principles to be observed in a project. Roles and organizations are only provided as suggestions. Hence we have another example of a systems approach.
- *Dynamic Software Development Method:* DSDM defines explicit roles and responsibilities as core of the methodology. Feedback loops are provoded for the delivery (Frequent Delivery, All changes are reversible) but not on process level. Hence DSMD uses a systems approach to let the software grow but a control model to manage the project.
- *Extreme Programming:* XP uses only a few rules to set up an environment in which software can grow. So the technical part clearly uses a systems approach. Concerning the methodology XP showed an interesting development in the last years. Though Kent Beck's metaphor of „Learning to Drive“ pointed towards a systems model, initial statements like „Either you do everything as written or you don't do XP“ were often interpreted as a sign for a control model. Figuring out that these statements drove XP into an unwanted direction, the community now agrees upon regular adaptation of the process. Therefore, XP may be considered a systems methodology today.
- *Feature Driven Development:* FDD installs the role of a Chief Programmer and uses upfront designs. Hence, FDD in my perception clearly uses a control approach rather than using a systemic view, even though it definitly is a light weight process.
- Lean Development: Most of the technical LD practices are similar to what XP found useful, so technically LD is also based on a systems model. The management is highly focussed on self-organization and feeback — indicators for a system model too.
- *Scrum:* Defining a central feedback loop as major control instrument identifies Scrum as systemic methodology. This adds to the fact that scrum defines few roles with the central role, the „Scrum Master“ having responsibility mainly for facilitating the process instead of the deliverable.

A Provoking Conclusion

I suggest to define agile development it as a methodology based on a systemic view on software development. This view leads to the agile principles as well as to many agile methodologies.

Using this definition would mean to re-evaluate the current understanding of which methodologies are agile and which aren't. According the above list this would narrow the „fully agile“ methodologies to ASD, Crystal, XP in its current form, Lean Development, and Scrum. DSDM would be considered as „technically agile“ methodology and FDD would be seen as a lightweight traditional process.