Position paper: "Are Agile Methodologies Really Different?"

Andy Schneider
as@bjss.co.uk

Despite significant commonalities between agile methods there are also many real differences. Intuitively there have to be significant commonalities between agile methods because, to be considered "agile", the methods satisfy a common set of criteria.

This paper is concerned with differences and not commonalities. There are many significant differences between agile methods and this position paper seeks to enumerate some of the more obvious ones.

Agile methods differ in terms of the requirements they are trying to satisfy, the scope of the solution proposed, the underlying practices described, the degree in which the methods are considered atomic units and the explicit degrees of freedom supported.

Agile methods differ in the requirements they satisfy:

- XP small, co-located teams.
- SCRUM, not limited to small, co-located teams.
- Crystal Clear, amongst other things, clearly targets "Up to 6 developers, typically 3-4"
- Crystal Orange targets projects up to 40 developers.

Agile methods differ in their scope:

- XP prescribes a set of engineering practices within a very lightweight project management framework. It is hard to imagine embedding another method *within* XP.
- SCRUM focuses on providing an agile management framework rather then specifying particular engineering practices. You can embed XP practices within SCRUM (XP@SCRUM) or, for example, use Function Point analysis as the estimation model.

Agile methods differ in their practices:

- XP uses a metaphor to define the architecture, giving the team a common language around which to structure the system. XP does not require a common object model.
- Crystal Clear requires a common object model. This model can be in the form of drawings and/or words. However, the emphasis is on a tangible artefact rather than design stored in the code and oral tradition of the project.

Agile methods differ in the degree in belief that the whole is greater than the sum of its parts:

- Kent Beck advises that newcomers to XP use the whole method, since it is the combination of all the practices that add most value.
- Alistair Cockburn encourages uses of Crystal Clear to select the practices and level of detail according to the requirements of the project. There is less emphasis on the whole creating some synergistic entity that delivers maximum value.

Agile methods also differ in the degree of scaling they explicitly support.

- Most literature on XP describes how XP practices facilitates rapid development of the software and provides advantages to developers within the standard XP model. Very little time is spent discussing how XP projects interface into the rest of the organisation and how formality can be increased as the team scales or organisational requires interfere with "vanilla XP".
- Crystal Clear has a well defined set of work products and techniques and explicit discussions on increasing or decreasing formality depending on project circumstances. It could be argued that whilst this approach offers less promise of a step change in productivity through application of the whole, it provides greater guidance to users of the method than XP when faced with a change in scale or organisation requirements.

The above examples show that the despite the commonalities between methods there are significant differences. These differences express themselves in the different degrees of freedom explicitly supported within the methods. Degrees of freedom help when applying an existing method to a project within an organisation because of the many factors that need to be considered when doing so. Examples of factors to be considered when utilising a development method are:

- Existing programme management, project management, control and governance structures and processes.
- Mandatory requirements placed on the project by the organisation the project is embedded in.
- Size of the project team.
- Skill of the project members.
- Complexity of the project.
- Size (e.g. in function points) of the project.
- Number of organisations the project interfaces to.
- Complexity of deployment environment.
- Non-functional requirements.
- Cost of system failure.
- Politics within the organisation.
- Amount of change the organisation can tolerate.
- History of the project (if already established) or its predecessors.
- Etc.

Given the problem of selecting a method is characterised by so many dimensions it is hard to see how one “theory of development” can satisfy all the constraints. Methods do not have to be unique to have differences. Therefore, whilst “It is easy to argue that the differences between the methods come from different metaphors or backgrounds rather than from inherently unique solutions.”, understanding the many differences between the different agile methods improves the chances that the team may choose an appropriate method for the job in hand and tailor it in an appropriate way.