



Workshop position paper:  
Agile values meet different value systems  
- Acknowledge the people dimension

---



Lars Arne Skår,  
Miles,  
Norway



The agile values are important for us who preaches about agile practices and methodologies. Still, we see that not all organizations buy in to them immediately and in particular we get challenged when we get into established IT departments who have been developing systems for some time. An attitude we often meet is – “It may be OK for small tactical web-applications, but certainly not for our real business critical systems, where we do real work.”

However, even such organization face challenges with increased expectations on their ability to deliver timely, with quality and as productive as possible with the additional challenge of increased complexity in terms of the functionality that is expected. Thus, the need for change might be clearly understood, and although the agile values and practices in theory should respond well to these challenges, it is not always easy to change in that direction. The resistance is high, and the theory of different value systems is an interesting way of approaching this.

I have been working as an agile coach since about 2001, and been conducting workshops on XP200x and Agile200x, this year on challenges with distributed organizations and being true to the agile values (exposing the “devils within”). In the early days, when working on smaller projects delivering tactical web-applications, working agile was not that controversial. The last 4 years I have been working with larger organizations, and the resistance (and probably the value mismatch) has been much bigger. Even as a line manager (CTO) in such an organization I faced huge resistance, in particular from “veterans” and “architects”. There was less resistance from the offshoring centers which may come as a surprise. The motivation



for agile practices that I experienced was the need for increasing quality and meeting customer demands in a better way.

People are more difficult to change than systems, so be patient and spend the time needed to understand what is driving them to be able to change them. Also do the changes incrementally – don't expect people to change dramatically. And do acknowledge that change is needed, it is generally expected that IT people and departments do need to change to be able to meet the increased expectations in terms of quality, productivity and delivery capability.

Changing from a controlling project manager used to a command-and-control style to a SCRUM-based self-organized team directly can often lead to an almost paralyzed team not being able to drive itself. In particular if there is little outside pressure on the end result. What we often see, in particular with an external scrum master, is that the team looks to the scrum master to delegate tasks and make prioritization on user stories or tasks. Thus, do expect a transition along the way, and that the scrum master in this case or a regular project manager do need to facilitate this transition. However, the self-organizing element is important to get the full power out of an agile team and the highest quality of decisions, so it is important that the team gets steadily increased empowerment. I would be happy to share my thoughts on this in the workshop. I believe this is critical and there is some scientific proof that distributed cognition of a problem not only is possible, but gets higher quality. This agile value (i.e. self-organizing teams) challenges both project managers and team members who have gotten used to the command-and-control style which is an established value in many traditional IT shops. However, there is a general management development style to increase empowerment and it is fair to say that it definitely speeds up the decision making and in my mind also improves the quality of decisions being made as they are being made closer to the matter at hand.

Another popular challenge is the claimed science of estimation and the granting authorities' need of a precise estimate to grant a budget or approval for developing a given functionality. Often times, such authorities will claim that not being able to provide such a precise estimate is a sign of inadequate skills in that area. In such a case, in reality it boils down to gaining trust, and getting enough faith to get started. A practical way around is to get the best estimate possible, and prove through valuable deliverables, that it is the result that counts, and in particular that the result adds more value than it costs – i.e. you could use traditional cost/benefit analysis to sell in the fact that it pays off to get valuable results early with –some- accuracy in the estimate than spending a lot of time planning to hopefully get some more accuracy. This value (i.e. collaborating towards creating value as opposed to regulating this through contracts) challenges value systems that are used to bureaucratic procedures, where typically economical decisions are made by people outside of the actual domain that needs a problem solved or a solution



made. In this case, you might want to ensure that you somehow get the problem owner in the loop so that they can influence the economical decision (i.e. expect a balanced risk in order to maximize the value of the result).

In addition you should work on those values that are in common or are easy to agree with, such as focus on quality, reducing risks and delivering value. At a high level, these are difficult to argue against. Still, you might meet some resistance when it is experienced that test-driven development for instance do require more investment upfront in terms of planning and developing tests – at least I have which came as a surprise. Some environments may perceive it is cheaper to do tests in the end, than to do them all the time based on tests that are planned and designed first. In these cases, often times you can point to the current level of quality and demonstrate a clear improvement in quality. If for instance your company has done a CMMI assessment recently with a low score (which would normally be the case if you do tests at the end, with the consequence of a huge backlog of change requests, issues and defects to deal with, and possibly even worse, complaints from your customers). Make sure this issue is understood, and you might actually have a fair chance of getting acceptance for introducing more proactive tests. In this case also do work in steps; i.e. start with introducing the concepts of tests, and start with unit testing for instance if you haven't already done that. This is hardly controversial any more. Another trick could be to expose the quality levels through measuring and reporting it in as transparent way as possible. This can even lead to your project manager starting to take an interest in quality, as well as your most reluctant programmers who haven't seen the need for increased quality earlier. I.e. although this agile value (focus on working software over comprehensive documentation) should be apparent you might have to work on getting the intention and realities understood in your organization. I believe it is possible if you manage to show the consequence of the state you are in. And it actually helps if you have a problem with quality.

There are other interesting value crashes as well that I have been exposed to in my line of work:

- Sometimes you may get the architects against you, as most practices around technology architecture tend to be somewhat formal and waterfall-ish;
- A claim that business critical systems needs careful planning before starting to make the system – in a time where it is clearly expected a shorter turn-around this is also important to face. Clearly business critical systems with lots of dependencies are challenging. However, dealing with it through a long planning and analysis phase is not the only way to deal with it – you could also break it down to more manageable pieces which seems to be a way many organizations are considering these days



- The assumed need for specialized skills to be able to deliver our complex systems creates a difficult bottleneck to pass, and definitely disrupts the steady rhythm we value in agile development projects. Do try the path of generalizing your specialists, so that your team is more empowered throughout and you get less dependent on your specialists.

To summarize, I would recommend that you do try to understand what makes your organization tick, and through that motivate the organization to change to be able to improve how they deliver. They would certainly be interested in that, and by acknowledging the need for change, there are ways of getting through the necessary changes. And it is likely that at least some agile values and practices will make them improve.