SIEMENS
Ingenuity for life

Siemens Digital Industries Software

Seven lessons learned in Agile

Moving to Enterprise Agile

Executive summary

The key mistakes made in Agile software development are tied to the inability to improve the process continuously, mainly because most purportedly Agile organizations are not able to make a shift in their mindsets. Even those that do a better job in revolutionizing their behavior are opposed by the inertia of the tools they are using, or by the reluctance to change or the inability to transfer the improvements achieved in the process into the tool chain. Through the introduction of Enterprise Agile, corporations and other large organizations see the opportunity to finally have the full benefit of Agile adoption.

Hard lessons learned in Agile

Many development organizations are considering moving to Enterprise Agile, hoping that the common issues encountered with Agile at the team level can be solved through moving to Enterprise level. Probably they will not – they will just be amplified.

The following are some of the typical lessons learned from having made the common mistakes in Agile adoption and usage with development teams.

Lesson 1: Great (inflated) expectations

"Today we go Agile." This simple statement makes people mutate their DNA, from the development team, to managers, to customers. Immediately the team forgets about deadlines and throws away any specification. Managers cut investments and feel safe that their project will be on time and on budget. Customers start changing their wishes and requirements every second minute.

Here are some of the expectations that Agile injects into the development team:

- "With Agile you can forget specifications and documentation."
- "With Agile we have less work to do."
- "With Agile we're not accountable."

Here some examples of over-expectations created by Agile in the management team or steering committee:

- "With Agile we can cut budgets in development, QA and tools."
- "With Agile we will deliver on (short) time and on (reduced) budget."
- "Agile will make dummies into geniuses."
- Finally, here are some expectations from customers and end users:
- "At the end of every iteration we will have a usable build."
- "The iteration will last for two days."
- "We can change anything, anytime."

Lesson 2: Agility is mostly between the ears

"We've been developing code for many years and we've seen hundreds of methods. Agile is just another one."

Agile is a mindset, normally called Agility. Someone argues that if you are not Agile you cannot become Agile. Others say that there is not so much to change in order to be Agile.

After years of experience, the consensus seem to go in the direction that becoming Agile is possible, but it takes time, commitment, engagement and coaching. Some of the misconceptions or misleading beliefs related to Agility include:

- Agile is easy.
- Developers must be Agile; others don't need to be.
- If you work in a company using Agile methods, you are Agile-minded.
- By putting any Agile tool in the process, you will increase your Agility.
- Agile training will make everyone Agile.
- I'm a strict Agilist. Whatever you do that does not comply with my definition of Agile is not Agile (that is, it's garbage).

Lesson 3: Agile through role colored glasses

The third lesson is about the definition of Agile and its possible misinterpretations by different stakeholders who often see Agile through a lens colored by their own role:

The manager's (distorted) view:

- "Agile is chaos."
- "Agile means no control."
- "Agile means no commitment."
- "Agile means no interference."
- "Agile is low-budget."
- "Every change needs a plan."
- "Every change needs a contract."

The developer's (distorted) view:

- "Agile is for developers; you are not a developer so you can't understand."
- "Agile means no commitments."
- "Agile means no milestones."
- "I'm an Agile developer, so I know what I should do."
- "We are an Agile team, so let's go on vacation."
- The Agile master's (distorted) view:
- "I'm an Agile master, so I decide what you do."
- "This is not what Agile says it should be."

Lesson 4: Bill of (wrong) rights licenses

Once a team or a department embraces Agile, there are some sacrifices that people think they must make in its name. Erroneously, Agile developers think they have the right to:

- Use "Well, we are Agile" as the answer to any question (usually meaning "no")
- Use Agile as a license not to document anything
- Use Agile as a license to forget about contracts
- Use any tool to be Agile, or even no tool

On the other hand, a very dangerous set of "rights" arises among project stakeholders:

- The "right" to give fuzzy (more than ever) descriptions of their needs
- The "right" to keep changing their minds
- The "right" to say "Stop discussing and give me a solution."

Lesson 5: Process, tools and change

One of the key hard lessons learned in Agile is about process improvement. Improving a process is not easy, as it represents a change and people don't like change. A continuous process improvement is a continuous change.

Typical arguments against process improvement include:

- "I don't think that this practice will be good for us, so we are going to skip it."
- "Our process is stable."
- "Our process works now (more or less). Why talk of changing it?"

On the other hand, one of the key enemies of process improvement is the tool that is in use:

- "We don't need a tool."
- "Excel is the best tool for Agile."
- "We are Agile because we have an Agile tool."
- "We cannot change the process because the tool does not support the change we wish to make."

The last statement is central to many discussions.

A process is good when either:

- It becomes a matter of habit: people don't need to think about process, or constantly refer to a manual that explains to them how to move forward in the process.

or

- It is automated: the process knowledge, the "how to move forward in the process," is embedded into

the tool where it is not subject to misinterpretation, as opposed to relying on people to first learn, then remember all the details.

If you want to change a process every second week, the first option won't be applicable: the process won't become a matter of habit. You won't have time to get used to it. So a continuous or frequent process improvement won't happen if you do not use a tool that is able to drive people through the process, and that is open to easily embrace any change to the process. Many vendors claim they can do this. Only one tool that demonstrates this ability in the field, and was conceived and developed since day one for this purpose, is the Polarion® application lifecycle management (ALM) solution from Siemens Digital Industries Software.

Lesson 6: Best laid plans

One of the key revolutions that Agile methodologies have been introducing is related to planning. If well applied, iteration planning gives high flexibility, allows positive interactions between developers and project stakeholders, and reveals continuous and tangible progress – the tricky part of course being "if well applied."

Here are some of the misinterpretations related to Agile planning that normally add risk to Agile adoption and/or success.

- Agile is "no plan."
- Agile is no milestones.
- Agile is flexible milestones.
- An iteration is a time frame in which to do something.
- An iteration is a time frame between two review meetings.
- An iteration starts two days before it ends.
- An iteration can't be cancelled.
- Agile helps planning.

Lesson 7: Indictments

Any combination of the misuses and misconceptions from Lessons 1 to 6 really amount to indictments against Agile itself. No one ever says "We are no good at using Agile." Rather, they say "Agile is no good."

Here are some of the most typical statements;

- "Agile can't be used in big groups/projects/enterprises."
- "Agile can't be used if you have to commit to a deadline."
- "Agile can't be used because we have bureaucracy."
- "Agile can't be used if you want to control and monitor your development activities."
- "Agile can't be used with our customers/developers/managers."

Moving to Enterprise Agile

This paper occasionally overstates the misinterpretations and the distorted practices that have been affecting attempts to use Agile over the past 15 years. Agile has been mainstream for quite some time now, and development teams have been learning and improving year after year, project after project. It can be tempting to dismiss anything stated as "not for me," or just something belonging to the past.

The risk could be that in the Agile development environment for Siemens'Polarion product development we no longer see (or we are used to doing, or we have just come to accept) some of the conceptions and behaviors discussed in the hard lessons above. And admittedly, it may be because they are not affecting our success in deploying Agile methods all that much just now.

Five recommendations about Enterprise Agile

1. Challenge your decision to move to Enterprise Agile. Do it only if you are truly happy with your Agile adoption and you need something more pervasive in your organization. Don't do it just because you are not happy with the Agile implementation you have now, or just because you think that Enterprise Agile will fix the issues you have encountered in Agile in the team/project scope.
2. If you think that your organization is not suited for team/project Agile, then it won't be suited for Enterprise Agile. Ideas like the following will surely kill your Enterprise Agile attempt:

- "Using some Agile practices will make our Enterprise waterfall become Enterprise Agile."
- "We cannot use (teamscale) Agile because we are too big, so let's go with Enterprise Agile."

3. Don't think that Enterprise Agile is bureaucracy to let teams remain Agile, or is something that allows non-Agile organizations to control better, or to get nice documentation and firm milestones out of their Agile teams. Agility is a mindset. And Enterprise Agility is a corporate mindset.
4. Use an incremental approach. There are a couple of different approaches to embrace Enterprise Agile: the "one shot" approach, or planned improvements over time. If you investigate, it quickly becomes apparent that the "one shot" approach doesn't work. So it was in the development of Polarion software: the team embraced Agile development in 2008, and is still adopting new practices and steering the process after eight years – and is gradually moving to Enterprise Agile. In the InfoQ article "Seven Sins of Scrum and other Agile Antipatterns" (http://www.infoq.com/news/2016/03/agileindia-7sins-scrum) the concept of incrementality is clearly stated as final advice for Agilists:

- Use retrospectives
- Improve incrementally
- Pick one or two items at a time
- Get coaching as needed

5. Last but not least: use the right tool. If using the wrong tool is an issue in Agile, it will be a nightmare in Enterprise Agile. Enterprise Agile processes are far more complex than Agile processes. That's obvious: they involve many more people in the organization and they touch many more activities than "just" software development.

Using Enterprise Agile, we must follow an incremental approach in adoption. Success will be in frequent or continuous process improvement.

When adopting and using Enterprise Agile you need a toolset that:

- covers many disciplines
- incrementally supports practices and processes
- embeds the process knowledge and drives users through the process
- helps you measure the quality of the process
- is open to easily embrace process improvements

Siemens Digital Industries Software

Headquarters
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 972 987 3000

Americas
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 314 264 8499

Europe
Stephenson House
Sir William Siemens Square
Frimley, Camberley
Surrey, GU16 8QD
+44 (0) 1276 413200

Asia-Pacific
Unit 901-902, 9/F
Tower B, Manulife Financial Centre
223-231 Wai Yip Street, Kwun Tong
Kowloon, Hong Kong
+852 2230 3333

About Siemens Digital Industries Software

Siemens Digital Industries Software is driving transformation to enable a digital enterprise where engineering, manufacturing and electronics design meet tomorrow. Our solutions help companies of all sizes create and leverage digital twins that provide organizations with new insights, opportunities and levels of automation to drive innovation. For more information on Siemens Digital Industries Software products and services, visit siemens.com/software or follow us on LinkedIn, Twitter, Facebook and Instagram. Siemens Digital Industries Software – Where today meets tomorrow.

siemens.com/software

58112-C9 4/17 N