Signs Your Agile Adoption is Off Track (And How To Fix It)

Thomas Stiehm, CTO

tom.stiehm@coveros.com

About Coveros

- Coveros helps organizations accelerate the delivery of secure, reliable software

Symantec

UnitedHealth Group

IFC International Finance Corporation World Bank Group

U.S.ARMY

AKO ARMY KNOWLEDGE ONLINE

Schlumberger

RSA The Security Division of EMC

UNOS DONATE LIFE

EMPLOYERS

Forge.mil

BNP PARIBAS

FannieMae

Why do organization adopt Agile?

- What they are doing now, doesn't work
- They are no longer able to react to the marketplace
- Customer satisfaction or project success rate is lower than acceptable
- It is mandated from above

Bottom line: No one adopts Agile because things are going well

Challenges in adopting Agile?

- Culture change
- Process and practice change

The Agile Adoption Challenge is to show that Agile can provide enough value to make the cost of change worth it.

Without an incentive to change, people will naturally revert back to doing things in a way that is familiar and comfortable.

This means that only determined organizations will be able to make the change.

How should you adopt Agile?

- Pick an Agile pilot project and do it using Agile methods
- Follow an established Agile process, whether you get Agile coaches or not.
- The pilot should be
  - Visible enough to be noticed
  - Small enough to be done by a 6 to 8 person team in 3 to 6 months
  - Politically isolated enough to make success possible
- After a successful pilot – Decide if you want to continue
- If you want to continue, pick other projects to transition to Agile over time
- Start instituting software development best practices such as Continuous Integration and Unit Testing in all projects in order to smooth adoption

What you shouldn’t do

- Everyone do Agile starting … now
- Let’s create a committee that will create our own Agile process in 12 months
- Make your organizations biggest Agile detractor responsible for Agile in your organization
- Force Agile on your teams without getting their buy-in

Where will Agile work?

- Any project - you can use Agile to deliver any software project, the project and technology don't matter
- Organizations that want projects to deliver value sooner
- Organizations that want to reduce project waste
- Organizations where a collaborative, trust based process can work
- Organizations that want to change

Where Agile will not work?

- Organizations that have a culture of control
  - Agile doesn't work here because control blocks collaboration and could destroy trust
- Organizations that have a culture of power grabbing politics
  - Agile doesn't work here because power grabbing politics destroys trust and only allows for short term collaboration
  - Power grabs often involve zero sum games where someone has to lose in order for someone else to win (I want you to fail syndrome)
- Organizations that don’t want to change

Reasons Adoption Fails

- Failure to define requirements
- Scrum Master not embracing their role
- Not using Agile processes or practices
- “Doing Agile” vs. Being Agile
- Limited Management Attention

Agile Failure Patterns Format

The following slides will present an Agile failure pattern using the following format:

- Category
- Problem
- Example
- Solution

Failure to define requirements

- Failure to achieve requirements agreement before the sprint begins
- Requirements not complete or approved at the beginning of the sprint
  - In the worst case I have seen, we had a nine month project where no sprint started with a final list of stories.
- Requirements Deadline
  - Make the deadline and stick to it
  - It will be a fight, it can only be successful with a supportive project sponsor
  - Use Agile Coaches to have the tough conversations with the PO as you put a deadline in place

Failure to define requirements

- Frequent requirements change during a sprint
- After a sprint has started a large numbers of stories are removed and replaced with new stories
  - Often done with the expectation that the capacity commitment will not change
  - New stories not sized or tasked by the team
- Reduce Product Owner scope of responsibility
  - An Agile Product Owner is a different job from a legacy Product Manager and the difference in effort means that the same person can't cover as many products
  - In a traditional Product Management model a PM would manage 5-6 products, for an Agile PO this should be reduced to 1 or 2 products

Failure to define requirements

- Incomplete requirements presented as complete requirements
- A large amount of requirements refinement is required for a large number of stories every sprint
  - Some requirements refinement in a sprint is expected
  - The threshold can vary from team to team
  - The amount of time the PO can spend in the team room is a big factor
- Product Owner Proxy
  - Business Analysts (BAs) empowered to make product decisions
  - Set the expectation that the BAs should learn enough to make 95% of the product decisions correctly
  - BAs have regular planning and review meeting with the Product Owner

Scrum Master not embracing their role

- Just maintains a checklist of activities and doesn't engage in removing obstacles
- Major project activity is creating and maintaining the task clipboard
- Delegates removal of obstacles to the person that identified the issue
- Have new Scrum Master shadow an experienced Scrum Master for a period of time
  - Work with them to learn the role
  - Set explicit expectations for how they support the team
- Replace the Scrum Master
  - A painful option that needs to be available

Scrum Master not embracing their role

- Forced Velocity – Team capacity is set outside team historic results

- There are 4 sprints left and we have 160 points that have to get done so we need to do 40 points a sprint even through we have been doing 25 points a sprint so far
  - Forced velocity isn't always triggered by the Scrum Master and it often part of a negotiation process higher up in the organization

- If it is the Scrum Master then coaching them on how to establish a velocity
- If higher up then re-negotiate scope focusing on a Minimum Viable Release

Scrum Master not embracing their role

- Allows team members to be pulled from the team during a sprint
- When emergencies come up, doesn't shield team members from the chaos of the situation
  - Actively volunteers team members to fix problems outside the project
  - Expects team velocity to remain the same
- One on one coaching to highlight the need for the team to be focused on the project in order to maintain velocity
  - Expect push back, especially from stretched thin organizations
  - Work on velocity trade-offs and other ways to show the impact

Not using Agile processes or practices

- CI that isn't CI - More like Continuous Build without unit testing
- The team invests in implementing a CI process but then doesn't create unit tests
  - A limited number of unit tests can be worse then none
- Create a practice period for unit testing
  - Conduct team training sessions to introduce the team to unit testing and the value unit testing can bring to the project
  - Work one on one with team members showing them how to write unit tests

Not using Agile processes or practices

- Refusal to use or try Agile engineering practices
- Whenever an engineering practice is discussed team members throws out reasons why they shouldn't do it, when you bring it to a head they flat out refuse
- One on one coaching
  - Work with individual team members to give them a feel for the practice
  - Focus on the benefits and value of the practice
- Trial periods
  - Most people will give it a try if the trial is time-boxed and non-binding

Not using Agile processes or practices

- Not completing committed work in a Sprint
- In a recent project, the team we were working with habitually failed to complete the sprint commitment for all sprints in a year long project
- Adjust the velocity expectation on and within the team until they can hit the sprint commitment
  - Work on ways to increase velocity once the team starts hitting their commitments
  - Realize that sometimes missing a commitment by a story or two (i.e. a relatively small amount of variance) is expected and happens

Not using Agile processes or practices

- Refusal to work in the team room
- Segregated QA
  - The QA group for one of our clients refused to spend time in the team room, much less sit in the team room full time
- Find a way to get the team into a team room
  - Team rooms make communication and collaboration easier, especially as teams are forming, being in one place for some period of time is critical to establishing and renewing relationships
  - If a room large enough for the team isn't possible, find ways to work in the same room together for part of each day
  - Team rooms can be hard to come by in some organizations, the work space and the politics make it hard
  - Do everything you can to make it happen

Not using Agile processes or practices

- I can overrule the team syndrome
- A senior developer on the team decides he knows more about the problem then the rest of the team and unilaterally changes the design
- Rather than the team estimating each story for a sprint, senior members of the team estimate the stories and ignore team input
- Insist that team decisions stick and that individuals can't override the team
  - If a change can be sold to the team then fine it can change, if it can't be sold to the team, there is almost always a good reason why

“Doing Agile” vs. Being Agile

- Blind adherence to agile practices can be as bad as not following agile practices
- One team I worked with had a number of passive aggressive employees that would only do their jobs to the minimum possible, such as:
  - Limited participation during retrospectives and other team meetings
  - Refused to work on certain areas of the application
- Make sure everyone participates in meetings, calling them out if necessary, in the meeting discuss the purpose and value of Agile practices
- One on one meeting with influential team members to enlist them in convincing the other team members

“Doing Agile” vs. Being Agile

- Using Agile terms without understanding them
- One of our customers is required to “use” Agile by the terms of their contract. In order to win the business they said they *knew* agile so they sprinkled agile terms into their non-agile process such as:
  - Design Sprint
  - Requirements Sprint
  - Testing Sprint
- Use Training and Coaching to teach them what Agile means
- Executive Awareness Briefings – Agile didn’t invent any specific practice but it does put them together in specific ways, get decision makers to see what makes it work

Limited Management Attention

- Not paying attention can obviate the teams efforts to be transparent
- During a recent assessment, the management of the company complained that they couldn't see how the team was doing; when we checked it out, the team had good information radiators, management just didn't look at them
- Promote your information radiators, if it is the team room wall then take pictures and send them out, if it is online then email out links and go to the executives desk and bookmark the link for them. If they don't see it, it is as if it doesn't exist

Limited Management Attention

- No feedback on the teams efforts or feedback too late in the cycle to do anything about it
- Ambush at the end of a release – On one team I worked with, the executive management had a reoccurring practice of giving us no feedback until just a day or two before a release and then giving us damning feedback
- Demand feedback early in the cycle, schedule time to sit with the executives and show them the system, ask questions to get their feedback, make them comfortable so they don't derail the project
- Find a new job at a more rational company

Limited Management Attention

- Management avoids accountability for their decisions
- Trade-offs are a large part of Agile and making certain scoping trade-offs can mean the difference between hitting a release date and not. One manager I worked for, liked to make the trade-offs to hit the date then pretend he didn't agree to the trade-off after the fact and held the team accountable for the other half of the trade-off
- Remind them of the trade-off and the teams progress to date, create a plan to get "back on track" and work with the team to get there
- Quit and go live on the beach