Agile Software Management Methodology

Lessons learned during implementation within a growing team creating complex solutions

Agenda

- Problem Space
- Basic Tenets
- Real-World Benefits
- Potential Pitfalls and Solutions
- Keyfactor’s Team History
- Q&A and Feedback

NOTE: the presentation is focused on Agile, but feel free to ask any questions i.e. resumes, interviews, expectations, etc...

Problem Space

- The traditional approach – ‘Waterfall’ – involved detailed design prior to any implementation
  - Does not account for changes in requirements, personnel, or technologies
- In some cases, predominant use of this methodology may be the correct approach
  - Building something in which a strict outcome overrides the need to adapt (space shuttle)
- The problems with this approach, in general, increase according to the complexity and length of development lifecycle
  - Complexity = more difficult to design in a ‘vacuum’
  - Long lifecycle = more opportunities for unanticipated changes

Basic Tenets

- People collaborating in a structured but flexible manner
  - Constant communication is the anchor of a successful outcome
- Usage over detailed documentation and specifications
  - Focus on outcomes instead of outputs
- Continuous feedback
  - Quickly create working components that can be reviewed and tested
- Rapid response to change
  - Limit rework through quick iterations building on previous work

Real-World Benefits

- Greater ability to adapt to ever-changing requirements
  - Adapt to changes in requirements, technologies, and personnel
- Rapid and improved feedback
  - Much easier to have a productive review and discussion of a working component vs a static specification
  - Countless times opinions change when seeing something ‘in action’
- Increased team member collaboration
  - Fosters increased ownership, accountability, and performance
  - Everyone has a stake in the final product

Potential Pitfalls and Solutions

PITFALL: Creation of unnecessary process

- The unstructured nature of Agile can lead to confusion over implementation
- In order to create structure, some approaches involve processes that are more of a hindrance than help
- Potential Solutions:
  - Continuous process evaluation: is value added? (we are iterating our process now)
  - Goals vs rules: focus on goals instead of details of how to achieve them
  - Management tools: these can assist with ‘suggestions’ on how to implement
  - Trust and listen to the team: the team will help dictate cost vs benefit

Potential Pitfalls and Solutions

PITFALL: Lack of any structure

- Unable to predict outcomes
  - Cannot forecast or communicate timelines
- Team members don’t know what is expected of them
  - Leads to frustration and decreased productivity
- Potential Solutions:
  - Plan for change: create goals according to what is currently known, while identifying potential points of uncertainty
  - Refine estimations: time estimations must be created and continuously updated if necessary
  - Trust and listen to the team: the team will help determine what they need to be successful

Potential Pitfalls and Solutions

PITFALL: Existential bug crisis

- Agile does not account for bugs and so they tend to not be included in forecasting or process
- Leads to incorrect timelines and unpredictable results
- Potential Solutions:
  - Treat bugs as formal work items: estimate them and account for the time necessary to implement them
  - Set aside time each sprint or at least release to work on technical debt and bugs
  - Trust and listen to the team: the team will help indicate what bugs should be fixed at what stage

Potential Pitfalls and Solutions

PITFALL: Lack of sufficient Quality Assurance

- Rapid development can sometimes come at the expense of quality
- Can be difficult to predict the amount of time required
- Can be hard to find qualified analysts, depending upon the complexity of the software
- Potential Solutions:
  - Involve QA into the process as early as possible
  - Foster the mindset that everyone is responsible for quality
  - Trust and listen to the team: the team will help guide on what works and what doesn’t

Keyfactor’s Team History

- First developer hired
- No real process up to about 3 or 4 engineers
- Began Agile implementation with the help of a knowledgeable consultant
- Created multiple scrum teams at around 8 or 9 engineers
- Utilize Microsoft Azure DevOps for all aspects
- Currently 3 teams

Questions?

- Gary.Galehouse@keyfactor.com