Establishing a Definition of Done in Complex Organizations

Joseph N. Ruskiewicz, Ph.D.
Introduction

- Developing large complex software is hard
  - Ensuring quality is even harder!
- Communication and requirements play a big role in the failure of all projects
  - Over 1/3 of all failed projects attribute the failure to lack or misunderstanding of requirements*
- Good software engineering process ensures success of the project
  - Analysis paralysis can also lead to failure
- Agile is a great alternative for small to mid-size companies
  - Application to large companies requires some modification

* Ellis, Keith. The Impact of Business Requirements on the Success of Technology Projects Business Analysis Benchmark, AIG
Understanding eBay

• We have 17,536 people working within the company
  – 3,902 Engineers
  – 1,072 Quality Assurance
  – 612 Product Managers
• We have about 157 Million active users annually
  – More than 50% is now on Mobile Platforms!
• More than 800 million products listed on our marketplace
• Global presences in over 39 markets
• There will be 250 million searches performed by the end of this presentation
• This all adds up to about 80 billion a year in Gross Merchandise Volume!
  – About 2,500 USD is traded per second!
Traditional Approach

Product Managers

Clients

Designers

A specification!

Engineering

Does this reflect our world?

A product!

Quality Assurance

Product Evolution
The World we Live In.

Product Managers

Clients

Designers

Engineering

A specification!

A product!

Quality Assurance

Ahhhhhhhhhhhh!!!!

Product Evolution
Sitting down with Agile

Go Team!
Applying Agile with 17,536 people

This is just not possible!
Domainification

- Break domains down the seams of vertical functionality
  - Selling, Payments, Search, etc.
- Applying agile against a smaller problem still is not feasible
  - It takes hundreds of resources to develop the smallest of functionality
  - Scrum of scrums or any derivation falls apart as we cannot get anywhere near the recommended 10-15 person teams
- We have a historical tradition of waterfall model
  - Hard to change to a new way of thinking
- Hybrid approach merging the process of the Waterfall with the agility of Scrum
  - Allows groups to choose how they want to work
  - Creates markers for the success of a part of the product
Vertical Domains with End to End Releases

Selling
Listing
Viewing
Payments

End to End integration and validation
A Waterfall of Scrums

• We allow each group within each domain define their own process
  – The Definition of Done ensures that at each step of the process has some level of quality

• Consumption of one group’s work requires meeting definition of done
  – e.g. product is finished and signed off before experience testing
  – e.g. functional specification (Product Requirements Document) is written before engineering begins work and testing begins writing test plans
Defining the Definition of Done

• We needed to define the **Definition of Done** for each horizontal layer in the process of developing a product
  – The definition for designs differ from the definition for engineering

• A dependency of one team may require that the team satisfying the definition of done before consumption
  – Useful for ensuring markers of quality during the engineering process

• Overarching definition for all of engineering
  – Specific definitions for focused groups (e.g. mobile)
Definition of Done — General Engineering

• This is the general engineering standard that all engineering groups must use as a minimum
  – Mobile engineering
  – Services engineering
  – Web-site engineering
• Contains basic Definition of Done checks
  – Meet coding, performance, and security standards
  – Automated unit tests achieving at least 70% code coverage
  – Localized for all markets and is accessible if need be
• We only ensure that P1 and P2 issues are resolved
  – Some minimal amount of failure is acceptable
Definition of Done — Mobile Engineering

• There are some modifications for the Mobile Engineering group
  – We have a different problem than the services groups
• Code must only utilize **finalized** assets for all supported devices (iOS, Android)
• Code must pass all **Basic Acceptance Tests**
  – This is our base level suite of tests that all our mobile apps must pass
• Code must not **reference or include** any third party libraries not authorized
• New functionality must have runtime switches that enable and disable the features
  – Very useful for experimental services and A/B testing
• Testing of the code must be approved by the **Mobile Quality Assurance**
Transitioning to a new Process — DoD Light

• Applying a **checkpoint** at each step of the process can be challenging
  – Ensuring quality through metrics is a new concept for most people
• Any reasonably organization does this, it is just not documented
• Worth writing it down so that everybody is heading towards the same goals
• Example: shoot for 20% code coverage with unit testing
• Make sure all new functionality and bug fixes are tracked
  – No requirement for full specifications
• Build product then focus on Tracking, Accessibility, and Localization

• Continue working towards a better tomorrow
  – Make an organic process to our Definition of Done
Conclusions

• Agile can work for large multi-national engineering firms
  – Allow each group to define their own form of agile
  – Create flexibility in each Definition of Done
  – Have checkpoints at each part of the workflow to validate the handoff
  – A Waterfall of Scrums can be helpful for tracking progress and quality

• Hard to achieve in a large organization
  – Requires buy-in from all levels of the engineering process
  – Shifting the mindset of a large organization to one of teamwork
  – Understanding what it means to be done and what to expect from other groups

• Putting it all back together has proved to be a challenge