Agile Software Development

JOHN YORK

EECS 441 WINTER 2018

Who am I?

John York

Director of Engineering at ProQuest Dialog

Chief Technologist SpellBound AR

A Computer Engineer from the University of Michigan!

An agile development expert with over 12 years of experience deploying and managing agile development at several local companies
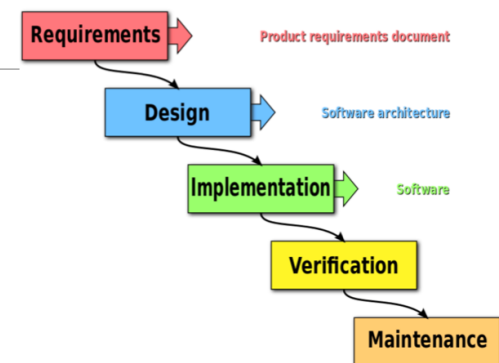
Certified Scrum Master
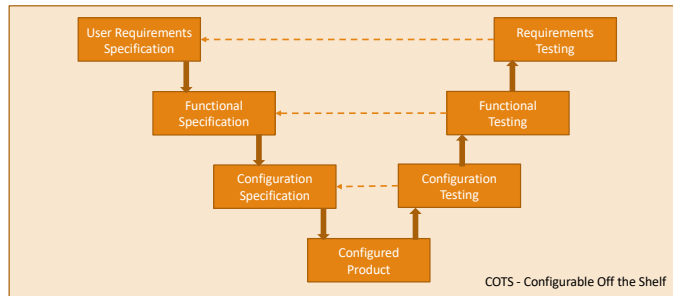
https://www.linkedin.com/in/johnryork

Software Development Methodologies

A BRIEF LOOK

Waterfall

Requirements → Product requirements document

Design → Software architecture

Implementation → Software

Verification

Maintenance

Iterative/GAMP 5 – V Model

User Requirements Specification → Requirements Testing

Functional Specification → Functional Testing

Configuration Specification → Configuration Testing

Configured Product
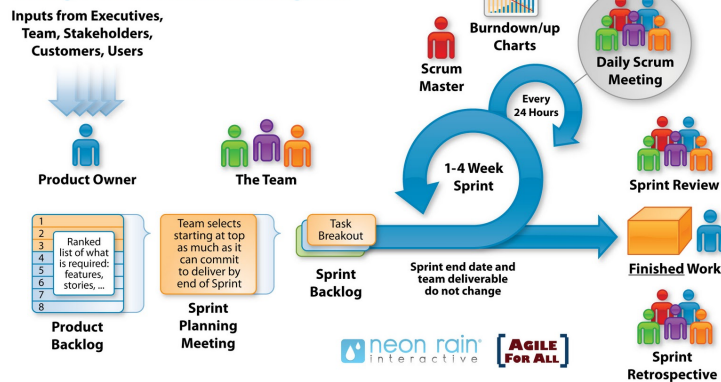
COTS - Configurable Off the Shelf

Agile

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
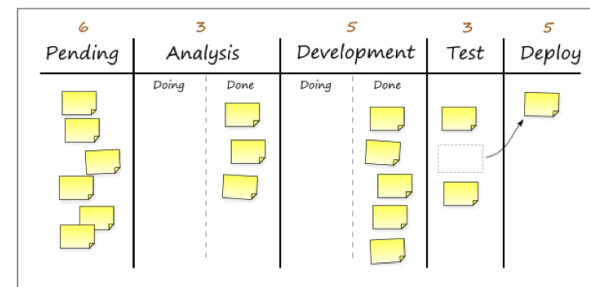
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

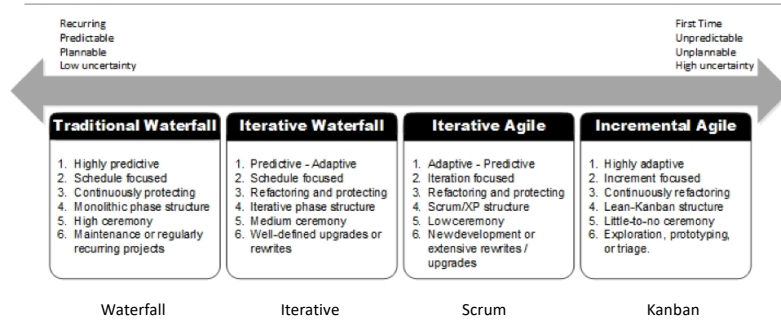That is, while there is value in the items on the right, we value the items on the left more.

The Agile: Scrum Framework at a glance

Inputs from Executives, Team, Stakeholders, Customers, Users

Product Owner

The Team

Ranked list of what is required: features, stories, …

Product Backlog

Team selects starting at top as much as it can commit to deliver by end of Sprint

Sprint Planning Meeting

Sprint Backlog

Task Breakout

1-4 Week Sprint

Scrum Master

Burndown/up Charts

Daily Scrum Meeting

Every 24 Hours

Sprint end date and team deliverable do not change

neon rain interactive   [AGILE FOR ALL]

Sprint Review

Finished Work

Sprint Retrospective

Kanban

6 Pending	3 Analysis		5 Development		3 Test	5 Deploy
	Doing	Done	Doing	Done		

Which method is right for me?

Recurring
Predictable
Plannable
Low uncertainty

First Time
Unpredictable
Unplannable
High uncertainty

Traditional Waterfall	Iterative Waterfall	Iterative Agile	Incremental Agile
1. Highly predictive 2. Schedule focused 3. Continuously protecting 4. Monolithic phase structure 5. High ceremony 6. Maintenance or regularly recurring projects	1. Predictive - Adaptive 2. Schedule focused 3. Refactoring and protecting 4. Iterative phase structure 5. Medium ceremony 6. Well-defined upgrades or rewrites	1. Adaptive - Predictive 2. Iteration focused 3. Refactoring and protecting 4. Scrum/XP structure 5. Low ceremony 6. New development or extensive rewrites / upgrades	1. Highly adaptive 2. Increment focused 3. Continuously refactoring 4. Lean-Kanban structure 5. Little-to-no ceremony 6. Exploration, prototyping, or triage.
Waterfall	Iterative	Scrum	Kanban

Scrum in Detail

Components of Scrum

3 Roles

3 Artifacts

4 Ceremonies

Three Roles

Product Owner   The Team   Scrum Master

Responsibilities

Three Main Artifacts

User Stories

As a <type of user>,
I want <some goal>
so that <some reason>.
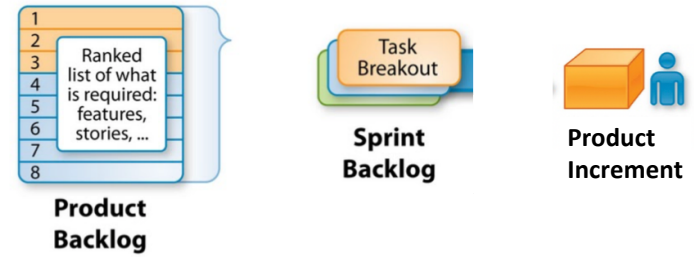
Acceptance Criteria

Acceptance criteria are the conditions that a software product must satisfy to be accepted by a user, customer, or in the case of system level functionality, the consuming system

Estimating and Story Points

Story points are a unit of measure for expressing an estimate of the overall effort that will be required to fully implement a product backlog item or any other piece of work.

- The amount of work to do
- The complexity of the work
- Any risk or uncertainty in doing the work

Use Fibonacci, not time!
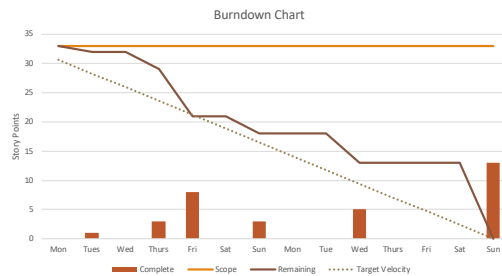
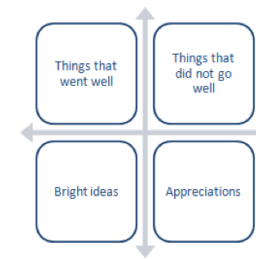(0, 1) 1 2 3 5 8 13 21 34 55

Four Ceremonies

Team selects starting at top as much as it can commit to deliver by end of Sprint

Sprint Planning Meeting

Daily Scrum Meeting

Sprint Review

Sprint Retrospective

Daily Scrum

Three Questions

What did you do yesterday?

What will you do today?

What is blocking your progress?

Burndown Chart

Retrospective

Things that went well
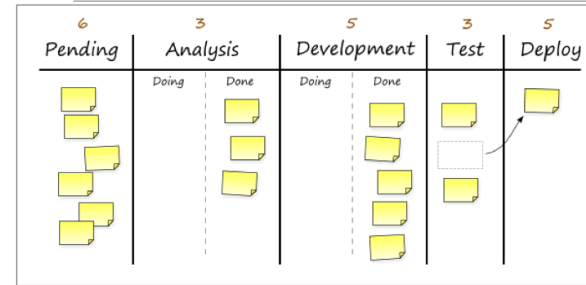
Things that did not go well

Bright ideas

Appreciations

Why Kanban?

Kanban

- Great fit for small teams/startups
- Great for prototyping
- Great for learning
- All team members help complete each step/no formal roles

A Scrum Example

CLASS REGISTRATION APP

Steps for a Scrum Project

1. Define the Problem/Goal
2. Assign Roles
   - Scrum Master
   - Product Owner
   - Team
3. Break it Down
   - Story Map
   - Product Backlog
4. Determine Minimum Viable Product
5. Write User Stories
6. Groom/Estimate User Stories
7. Sprint Planning
8. Sprint (1-4 weeks)
   - Daily Scrum
   - Sprint Review
   - Sprint Retrospective
9. Iterate 5-8 until MVP
10. Release?
11. Repeat 1-11 while $$$

1. Problem Statement and Goal

PROBLEM STATEMENT

Create a mobile app to allow students to register for classes

GOAL

Find a class and register a student for it

2. Assign Roles

Ideally Self-Assigned

Scrum Master - ?

Product Owner - ?

Team - ?

3. Break it Down

Product Owner

Product Owner Driven

The Team

User Activities
- Login
- Find class
- Register for the class
- View class schedule

Scrum Master

3. Story Map/Product Backlog

Product Owner

User Activities
- Login
  - Create home page
  - Create navigation
  - Register
  - Reset password

- Find class
  - Display list of classes
  - View a single class

- Register for the class
  - Add register link to class view page
  - Register page
  - Confirmation page

- View class schedule
  - Add link to navigation page
  - View class schedule

The Team

Scrum Master

4. MVP

Product Owner

User Activities
- Login
  - Create home page
  - Create navigation
  - ~~Register~~
  - ~~Reset password~~

- Find class
  - Display list of classes
  - View a single class

- Register for the class
  - Add register link to class view page
  - Register page
  - Confirmation page

- View class schedule
  - ~~Add link to navigation page~~
  - ~~View class schedule~~

Epics

User Stories

The Team

Scrum Master

5. Write User Stories

Product Owner

User Activities
- Login
  - Create home page
  - Create navigation
  - ~~Register~~
  - ~~Reset password~~

- Find class
  - Display list of classes
  - View a single class

The Team

Scrum Master

Create Home Page

As a student

I want to start the app

so that I can register for classes

Acceptance Criteria

AC1 – Create an app icon

AC2 – Show the home page at launch

6. Groom/Estimate/Prioritize Stories

Product Owner

Create Home Page   3

As a student

I want to start the app

so that I can register for classes

Acceptance Criteria

AC1 – Create an app icon

AC2 – Show the home page at launch

Create Navigation   5

As a student

I want buttons

so that I can access main functions of the app

Acceptance Criteria

AC1 – Navigation at top of app

AC2 – Show buttons for Classes and Schedule

The Team

Scrum Master

7. Sprint Planning

Product Owner

Scrum Master Driven

Review the top priority stories from the product backlog and decide how many stories can be completed in the sprint.

The Team

Pull as many stories as the team thinks can be achieved into the sprint backlog.

Break stories down into tasks doable in a few hours. If story is not doable in a single sprint, consider breaking it down into smaller stories.

Scrum Master

8. Sprint

Scrum Master Drives Daily Meeting, Sprint Review at end of Sprint and Sprint Retrospective

Team raises any obstacles, Scrum Master works to help resolve them through Product Owner, other team members, outside resources, etc.

Scrum master can create a burn down chart for stories in the sprint and track daily progress against it.

9. Iterate 5-8 until MVP

If product meets MVP, release it! If not, keep iterating until releasable.

10. Release

Congrats!

11. Repeat 1-11 while $$$

Define new goals, update product backlog, define new MVP and iterate

Appendix

Product Owner Challenges

Product Owner

Get a backlog

Maintaining the backlog

Managing scope

Writing user stories

Reviewing completed work

Customer proxy

Team Challenges

The Team

Cross functional

Co-located vs. disperse

Existing siloes such as UX, QA, etc

Over/under committing

Sizing vs. estimating

Technical debt
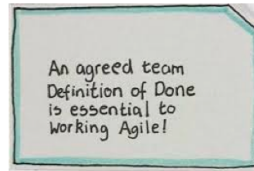
Team size (3-9 optimal)

Scrum Master Challenges

Scrum Master

Removing obstacles/impediments

Coaching vs. controlling

Managing tools (JIRA, Trello, Excel, etc)

"Recovering" project manager

Gaining team commitments

Optional Artifacts

Burndown/up Charts

An agreed team Definition of Done is essential to Working Agile!
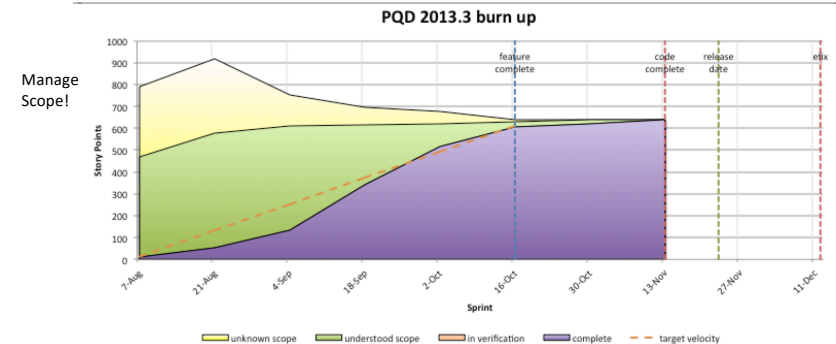
Definition of Done

Definition of Ready

User Story - Definition of Ready

- User Story must be clearly understood by all team members
- User Story must have Acceptance Criteria
- User Story must be estimated
- User Story must be sized appropriately
- User Story must be free of external dependencies

User Story - Definition of Done Example

- QA Manual regression Test scripts written and updated
- Automated tests (JUNIT or Selenium or etc.) with documented exceptions
- All Acceptance Criteria tested and passed
- Code peer-review with component owner
- Adhere to code conventions
- Product owner review/sign off

Burn Up Chart

Manage Scope!

PQD 2013.3 burn up

Minimum Viable Product (MVP)

PRIMARY GOAL: Allow users to receive an individual, customised pair of shoes

USER FLOW

Customize shoe | Buy shoe | Manage order | Deliver order

WALKING SKELETON

MVP LINE

HIGH NECESSITY

LOW NECESSITY