Working Agile in Software Development

Monica Petrica

November-2017

Agenda

Project Management – general concepts

Waterfall vs Agile – when to use each, advantages, disadvantages

What is Agile – generalities

Scrum – definitions and concepts

Agile at work – tasks breakdown

Coding agile in Distributed Teams – team work in a global organization

Agile issues / pitfalls – agile is not perfect, what we can do about that?

Agile and Continuous Delivery

Project Management?... What's that?...

planning, organizing, securing, motivating and controlling the resources to successfully complete the project

Why we do this: The better Project Management the bigger chances for project to succeed

It's about predicting!

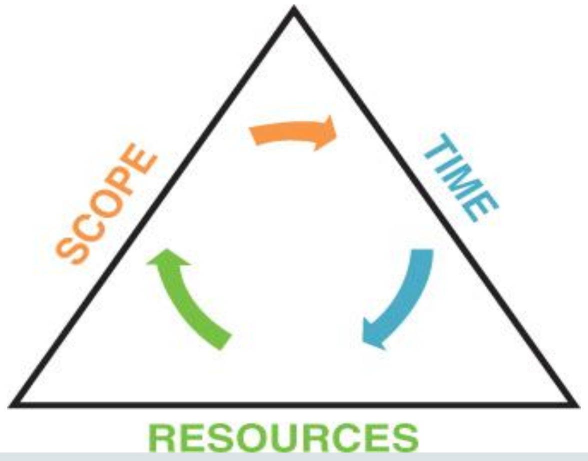
Define success!

Over budget, on time, reduced scope?

Usual Phases:

Concept > requirements gathering > design > development > testing > shipping > maintenance

Triple constraint: scope, time, resources



ORACLE

Types of PM (life cycles)

Predictive / Waterfall
Iterative

Incremental
Agile

Approach	**requirements**	**activities**	**Delivery**	**Goal**
Waterfall	Fixed	Performed once	Single delivery	Manage cost
Iterative	Dynamic	Repeat until correct	Single delivery	Correctness of solution
Incremental	Dynamic	Performed once per increment	Frequent small deliveries	Speed
Agile	Dynamic	Repeat until correct	Frequent small deliveries	Customer value through deliveries

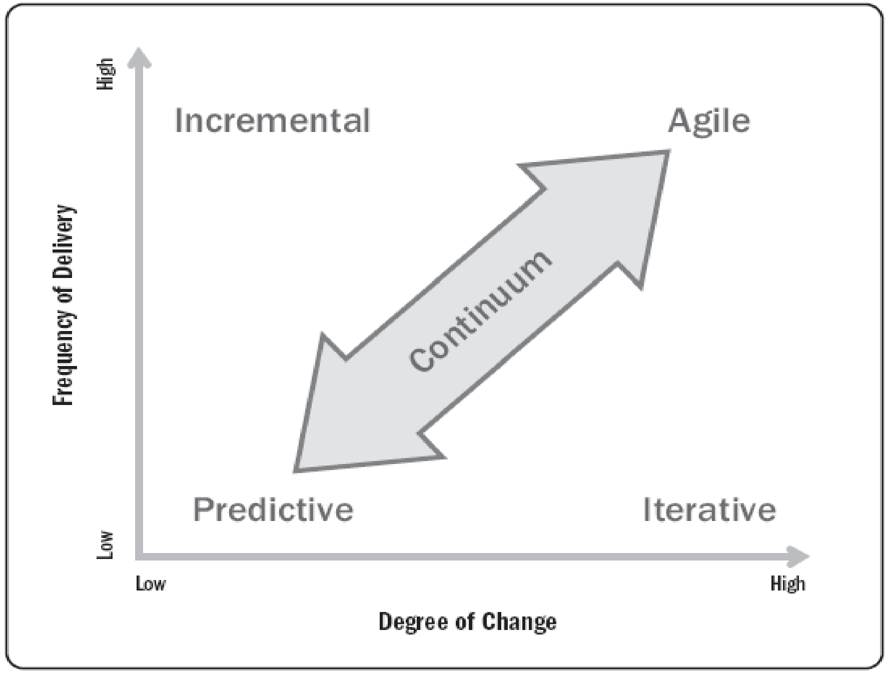
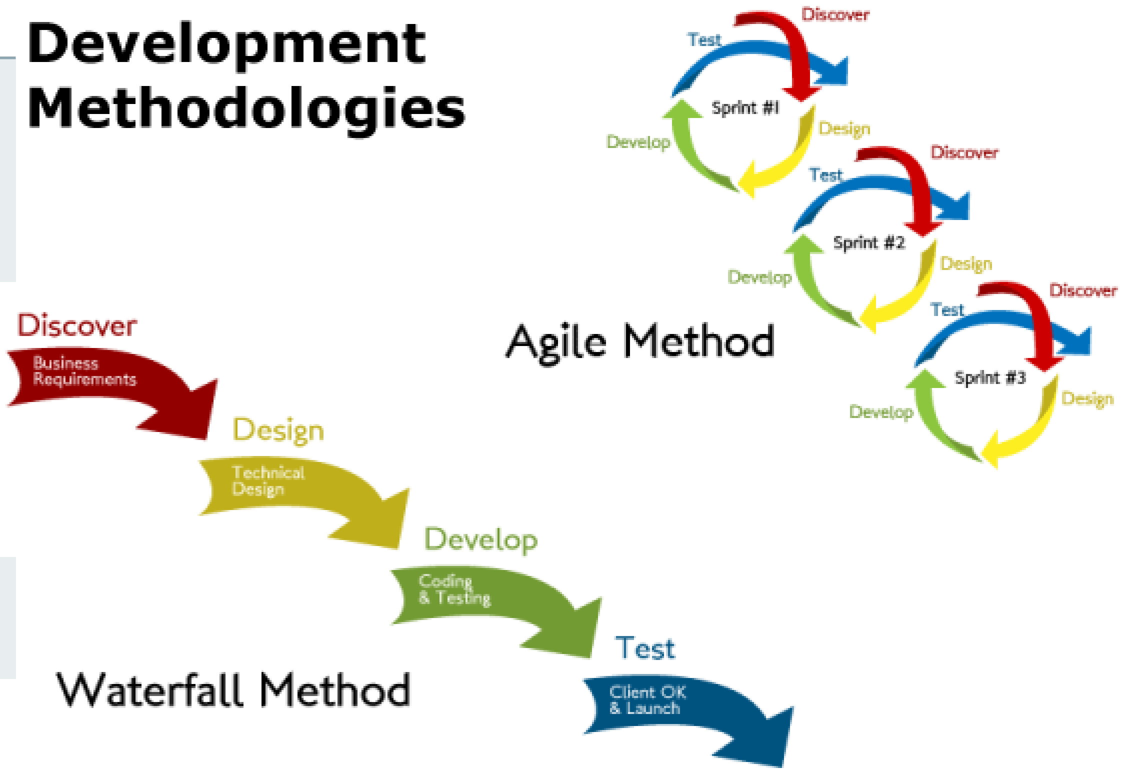


Figure 3-1. The Continuum of Life Cycles

Waterfall vs Agile

Waterfall	Agile
Sequential steps > requires one step to be fully completed before starting the next step	Iterations of waterfall steps > no need to fully define everything from the beginning
Spec/req (100% done), budget, resources > preprod > schedule > hire team > build > alpha (tests) > beta (more testers) > ship	Partial req > dev > test and… start from the beginning!
All tests are done in the end (rework now?...)	Tests are done in iterations
No interaction with end users	Users have the opportunity to see if the product is in line with their req and can provide feedback
Spend the same amount on time on all features (less or more important)	Spend more time on features people want (and less on less important features)
See everything in the end	See completed pieces on the way



Waterfall vs Agile (cont.)

What would you choose? ….

Waterfall when:

- Clear, known, unchangeable requirements
- Technology is understood
- Have required expertise

Agile when:

- ***Developing software products!***
- Ambiguous, changeable requirements
- Rapid deployment – time to market
- Flexibility in budget and timeframe
- But…
  - In Agile you don't write code faster – instead you write the code in the **right direction**
  - Agile is good but not perfect > adapt the recipe / **adjust the process**
  - Agile assumes resources are interchangeable > hm… not really! Are skills interchangeable?...
  - Communication challenges: people from different disciplines communicate differently (diff terminology)

mixture of both

Agile world

Agile manifesto

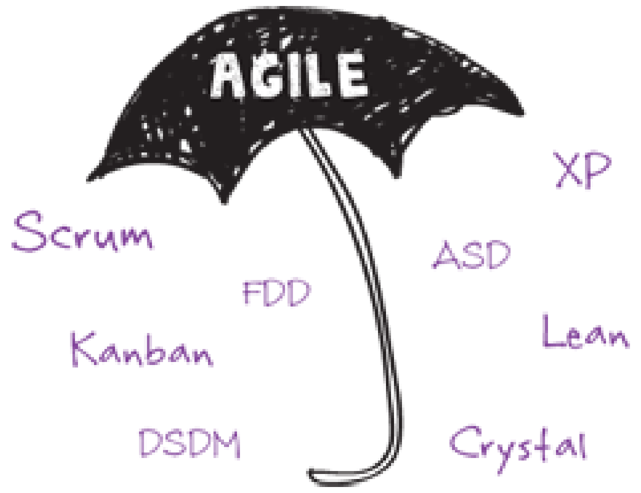
Individuals and interactions over processes and tools

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.



Multiple flavours of Agile – most popular is Scrum

From Waterfall to Agile

Organizational **culture**: people and processes => results

Culture change is the primary barrier to further Agile adoption at companies

Why is hard to change the culture:

- it is the accumulation of years of interactions and experiences

First step: shift in culture

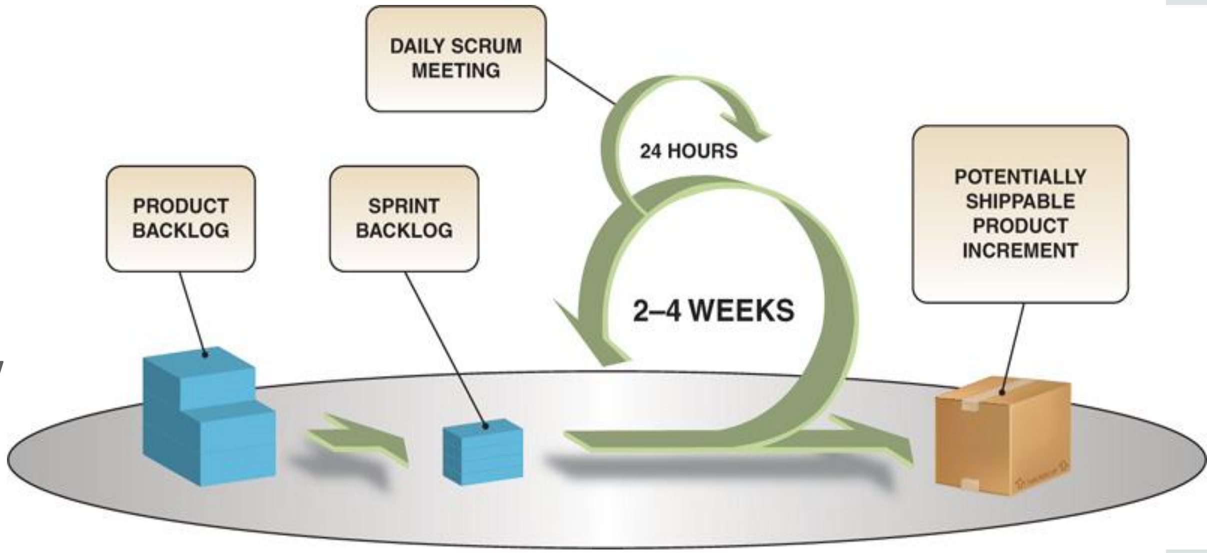
People:

Why would team members want to embrace Agile?

Team members	**Managers**
Self-Organizing Teams – select their own tasks Continuous improvement Frequent delivery Removing us vs them (dev vs testing, teams vs PO) Physical workspace	Questions not solutions Clearing roadblocks They should focus on technology architecture and true employee development Trust the team Staff the team for success

Get organized in Scrum

emphasizes the importance of organizing a project into specific durations



Sprint: iteration of couple of weeks: 1 to 4 weeks

- development sprints (designing, coding, testing, etc.)
- hardening sprints (shape down, stabilize code, no new features)

Scrum players

- **Product Owner** – product vision, prioritize features
- **Scrum Master**: facilitator, mediator, run calls, team coach, keeper of the documents, keep communication fluid; vision holder
- **Team member**: make things happen! share information and stay informed
  - roles

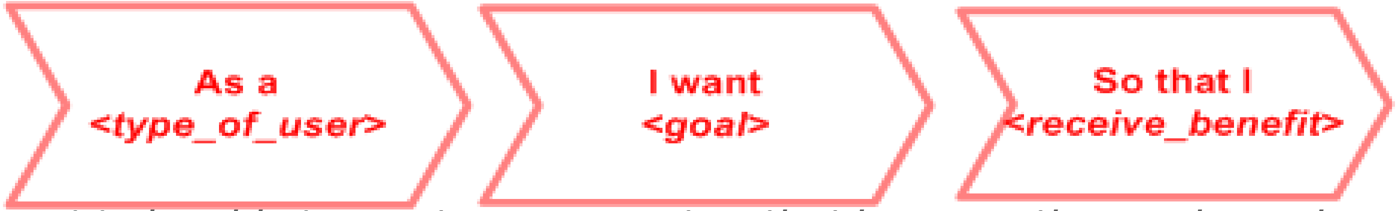
As a team: find a problem?... Adapt the process!

Scrum board

- physical board (cards) or Software tools (Jira)

Requirements in Scrum

User story: smallest unit of work delivering value to customer; small enough that it can be designed, coded, and tested within the sprint or iteration.



As a customer, I want to be able to create an account so that I can see the purchases I made in the last year to help me budget for next year.

Stories split in implementation items – tasks (and subtasks)

Epic: a large body of work that cannot fit into an iteration – multiple stories
Feature: multiple epics
Release: feature pack
Acceptance criteria: tests that the product owner will use to grade the successful development of the user story.
Definition of Done: feature completion; common understanding of various stakeholders

Requirements in Scrum (cont.)

Example:

Feature: Interface for creating and maintaining service requests (incidents)

- Epic: Create an incident
  - Story: as a customer I want to submit an incident so that it will be resolved by a support agent
  - Story: as a customer I want to be able to add an attachment to an incident so that support agent have enough information

- Epic: Update an incident

- Epic: integrate knowledge management with incidents

Grooming and Planning

Product backlog: Feature list for the whole project; the highest-priority stories reside at the top of the backlog and are in the lowest level of detail.

- Prioritization based on value (business value + ROI) – from customer prospective
- Prioritization and estimation often take place in a session called product backlog grooming – clarify and improve users stories, break down requirements, add acceptance criteria

Sprint backlog: List of stories planned in a certain sprint

Estimation

- LOE, T-shirt sizing
- Ideal time
- story points – Fibonacci sequence

How teams estimate - Planning Poker, etc.

Velocity: the amount of work that the team can usually deliver within the time frame of a sprint and can be used as a predictor for future iterations

ORACLE

Meetings in Agile

Meetings - time boxed; fixed time interval, agenda, involved participants

Sprint planning

- team goal: what are deliverables . 1-2h
- Add stories from product backlog in sprint backlog, break the features in tasks, add estimates

Daily scrums / standups

- 10-15 min – just updates from team members, do not solve problems in this meeting

Sprint Review

- demo deliverables
- Is this what we want? => adjust product backlog

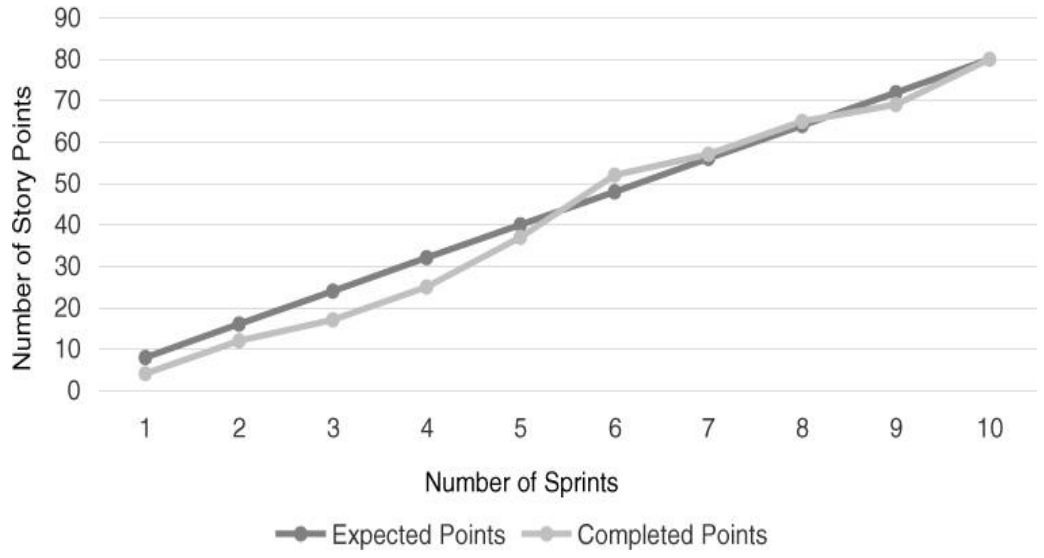
Retrospective

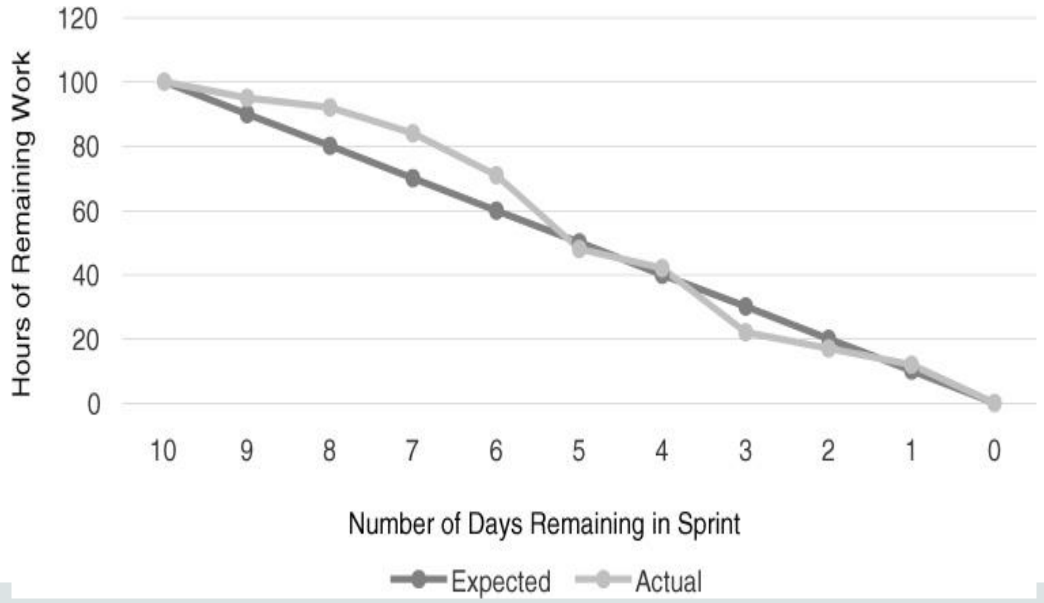
- improve things for the team and about team
- review the process
- things to do differently – short list

Tracking and Reporting

Burn charts

- Burn-up: measure overall progress

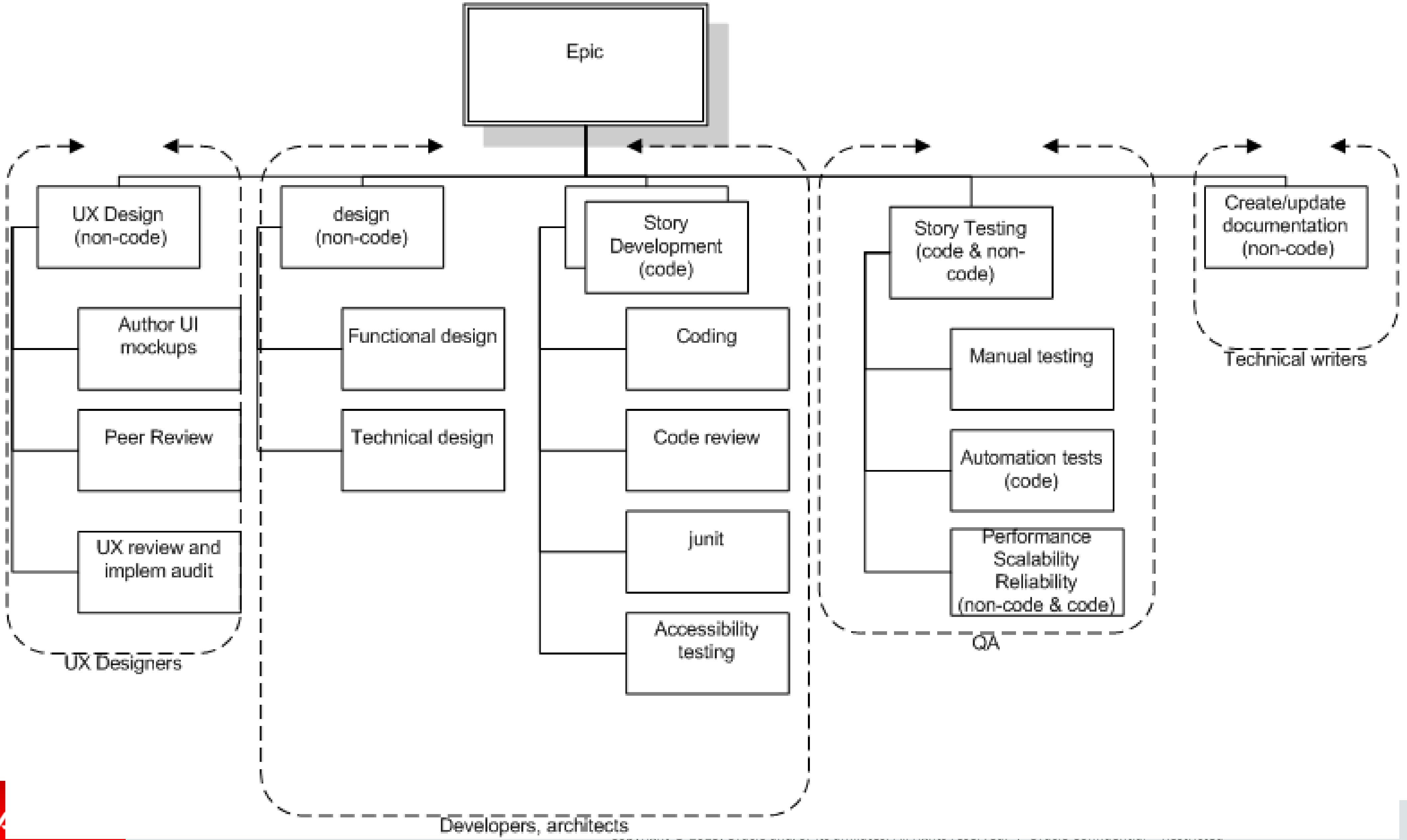




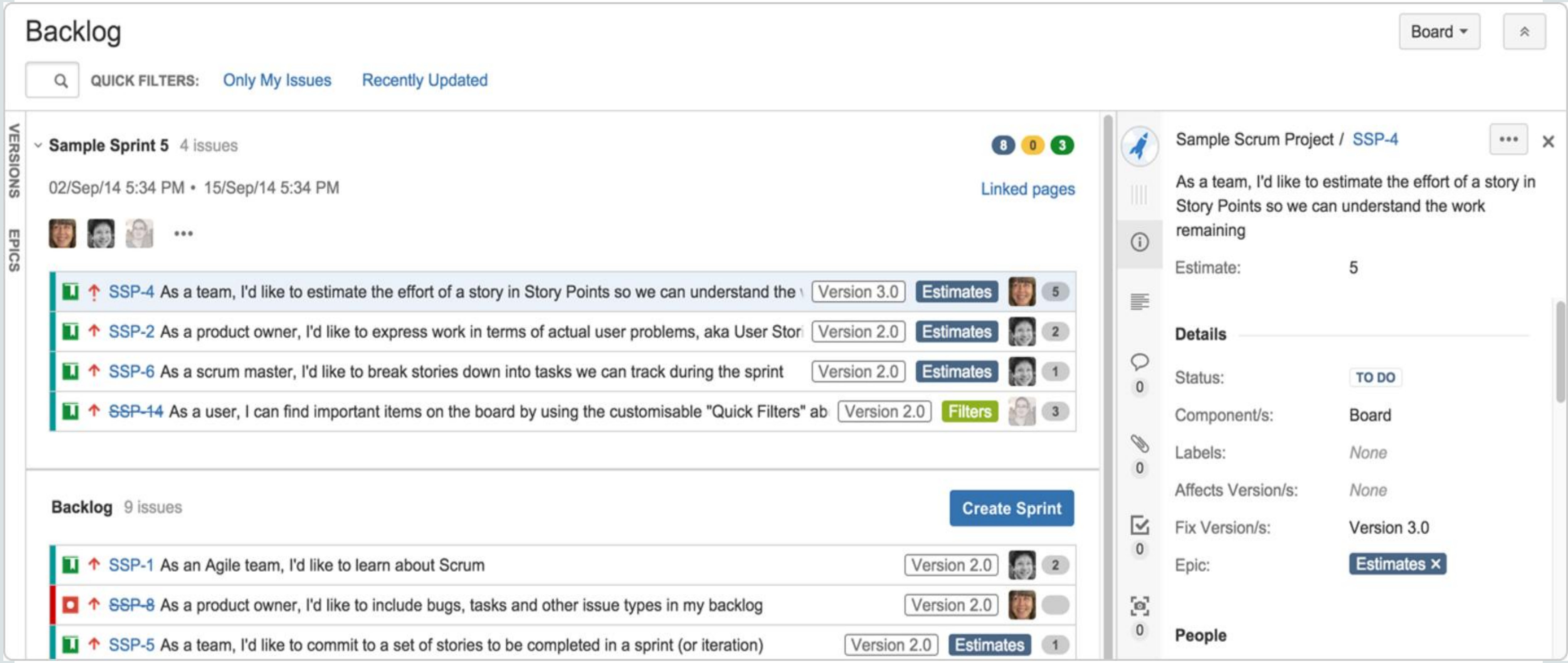
- Burn down charts These are the daily status checks for the team relative to where they expected to be at a particular point in time.

Team can see immediately if something wrong

Agile at work: Tasks breakdown



Agile at work: Scrum Board



Backlog

QUICK FILTERS: Only My Issues Recently Updated

Sample Sprint 5 4 issues

02/Sep/14 5:34 PM • 15/Sep/14 5:34 PM

Linked pages

SSP-4	As a team, I'd like to estimate the effort of a story in Story Points so we can understand the	Version 3.0	Estimates	5
SSP-2	As a product owner, I'd like to express work in terms of actual user problems, aka User Stor	Version 2.0	Estimates	2
SSP-6	As a scrum master, I'd like to break stories down into tasks we can track during the sprint	Version 2.0	Estimates	1
~~SSP-14~~	As a user, I can find important items on the board by using the customisable "Quick Filters" ab	Version 2.0	Filters	3

Backlog 9 issues

Create Sprint

SSP-1	As an Agile team, I'd like to learn about Scrum	Version 2.0		2
~~SSP-8~~	As a product owner, I'd like to include bugs, tasks and other issue types in my backlog	Version 2.0		
SSP-5	As a team, I'd like to commit to a set of stories to be completed in a sprint (or iteration)	Version 2.0	Estimates	1

Sample Scrum Project / SSP-4

As a team, I'd like to estimate the effort of a story in Story Points so we can understand the work remaining

Estimate: 5

Details

Status:	TO DO
Component/s:	Board
Labels:	None
Affects Version/s:	None
Fix Version/s:	Version 3.0
Epic:	Estimates

People

Coding Agile in Distributed Teams

Why: large project, budget constraints, etc.

Biggest challenge: **communication** and coordination

Distributed team vs **dispersed** team

- Within the same company or between different companies

Communication *between customers and developers* is as important—and as challenging—as it is *among developers.*

Differences: cultural, linguistic, physical distance

Centrally Coordinated or Globally Integrated

Team(s): stay focused on the **SAME goal**

real challenge is not the organization of the work but the **integration** of the distributed development effort into one working system

Coding Agile in Distributed Teams

Approaches – Divide project team in

- a “waterfall” model: analysts, programmers, testers etc.
- architectural layers of the software: backend, UI, etc.
- **Feature teams**

“Trust is developed incrementally, broken precipitously.”

It’s about **people**:

- face to face **meetings**; meetings location – rotate location, team members to participate (if not all team can be there)

- **Vocabulary** – for ex nightly builds – night where?

- **Cultural** differences

Keep sites in touch: via leader involvement, off-line interaction, infrastructure quality.

Distributed teams benefit from following agile practices.

Agile is not perfect

Productivity myth: with agile productivity will increase

- In Agile you don’t write code faster – instead you write the code in the right direction

Can be seen as an undisciplined approach

Requirements emerge and evolve throughout the development => **scope creep** => risk of ever-lasting projects.

Requirements are clarified just in time for development and can be documented in much less detail => **less information / documentation** available to new comers.

Continuous testing – high costs

Agile development is rather **intense** for developers.

Agile assumes resources are **interchangeable**

Org culture should support this + changes in org behaviour; agile transformations are only likely to succeed if heavily supported by the top management

Agile pitfalls

- Completing Sprints While Leaving Behind Technical Debt
- Problem Solving in the Daily Scrum.
- Assuming That Agile = Faster
- Planning Every Iteration for the Project In Advance
- Assuming That Velocity is Equal Across Teams
- Forgetting Artifacts (documentation, training materials etc.)
- Adding Stretch Goals to Sprints
  These are the “nice to haves” that are not planned but at the same time are planned.
- Product Owner Specifies Solutions
- Using Metrics to Measure Productivity
  Metrics such as velocity or number of story points completed in a given sprint are not a good measure of productivity

ORACLE

Agile and Continuous Delivery

Continuous Delivery: frequent releases of new software through the use of automated testing and continuous integration.

Continuous integration=> continuous delivery => continuous deployment

code > label > branch(es) > production

accelerating the entire build-test-deploy cycle

Quiz

1. What is the triple constraint in project management?
2. When shall I use waterfall as a PM methodology?
3. Why agile is more suitable for software development?
4. What are the advantages of using agile ?
5. What is the primary barrier when moving from waterfall to agile?
6. What is a sprint?
7. What are the roles of PO, SM, TM?
8. What is a scrum board?
9. How to create a user story? Epic? Feature? Release?
10. What is a product backlog? Sprint backlog?
11. How do you estimate a story?
12. What is velocity?

What we have learned

- We plan to predict and deliver!
- Waterfall vs Agile
- When unknowns and/or flexibility – Agile!

Product Owner, Scrum Master, Team Member

Stories, epics, tasks

Backlog

Planning, scrums, retrospectives

Sprint

Board