W4156

Software Development Methods

Recap

We covered

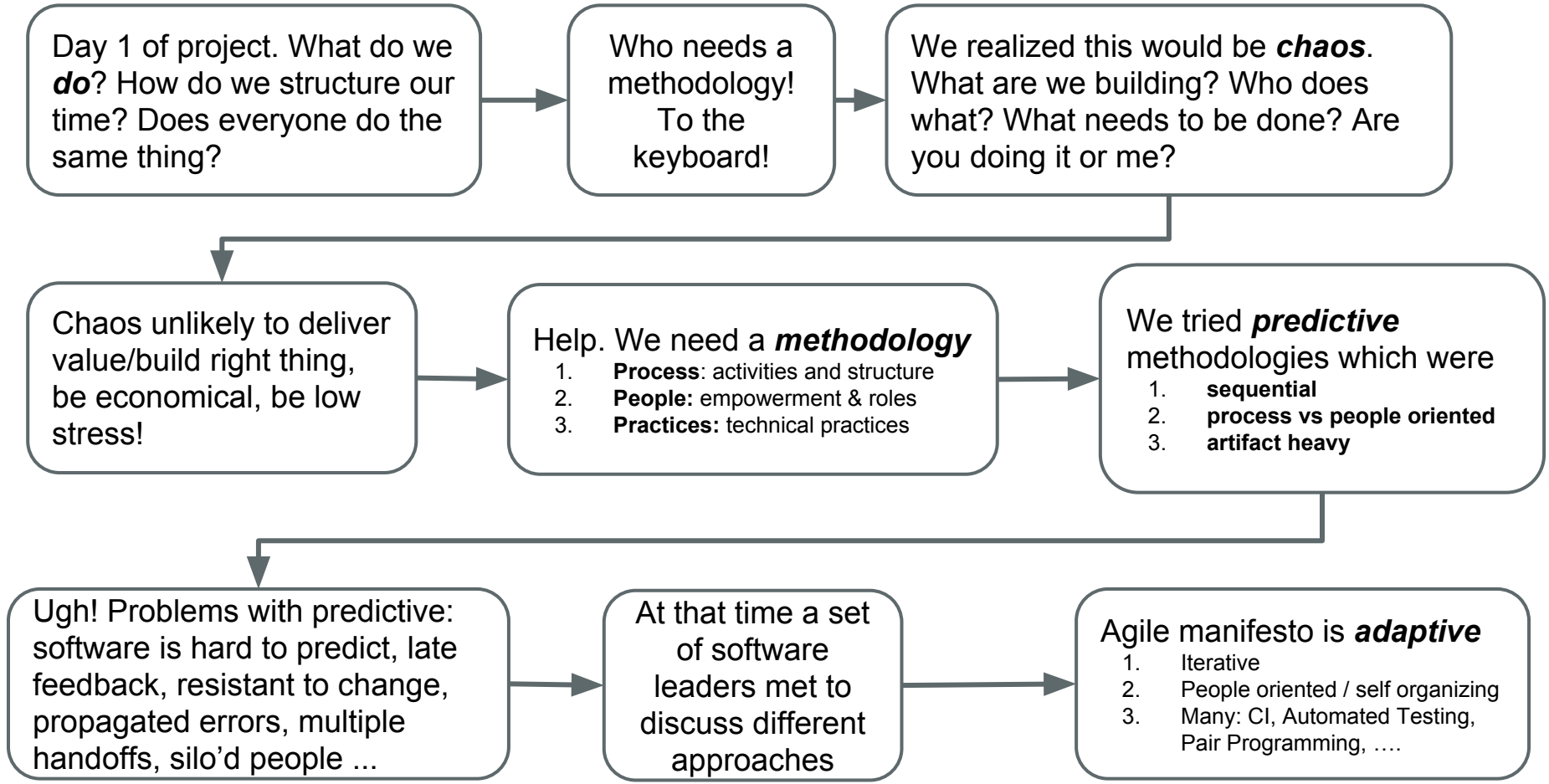
1. Plan Driven Methodologies
2. Adaptive Methodologies and Agile
3. We understand the Agile Manifesto and Umbrella
4. We now want to see more ‘implementable’ frameworks

Terminology is imprecise in industry. I will differentiate between

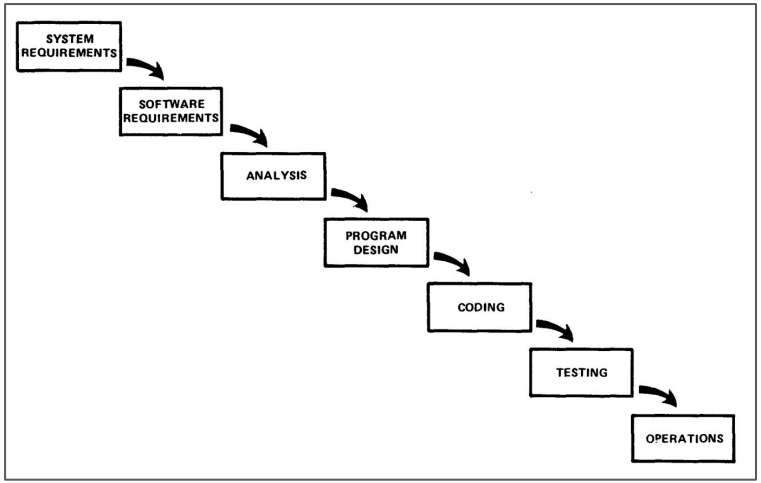
1. Methodologies: higher level set of principles (predictive, adaptive/agile)
2. Methods & Practices*: lower level implementations that can be applied

* Industry language is imprecise. Common synonyms are Frameworks/Processes/Flavors

Recap / Thought Progression

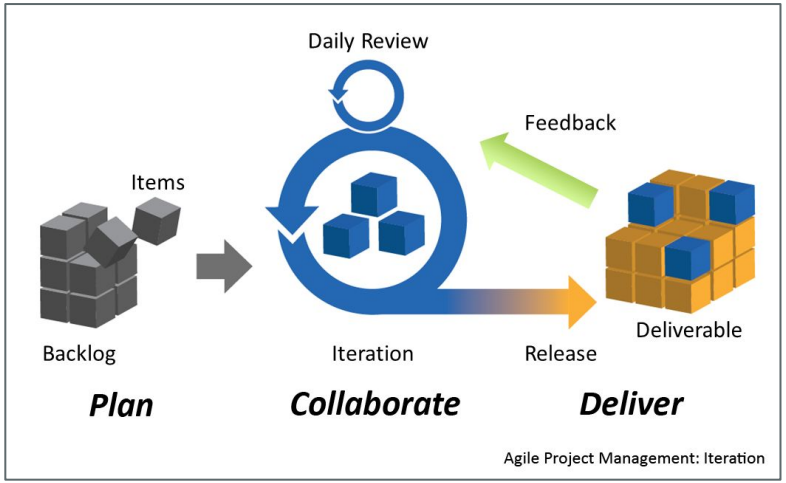


Predictive vs Agile



[Royce ICSE '87]

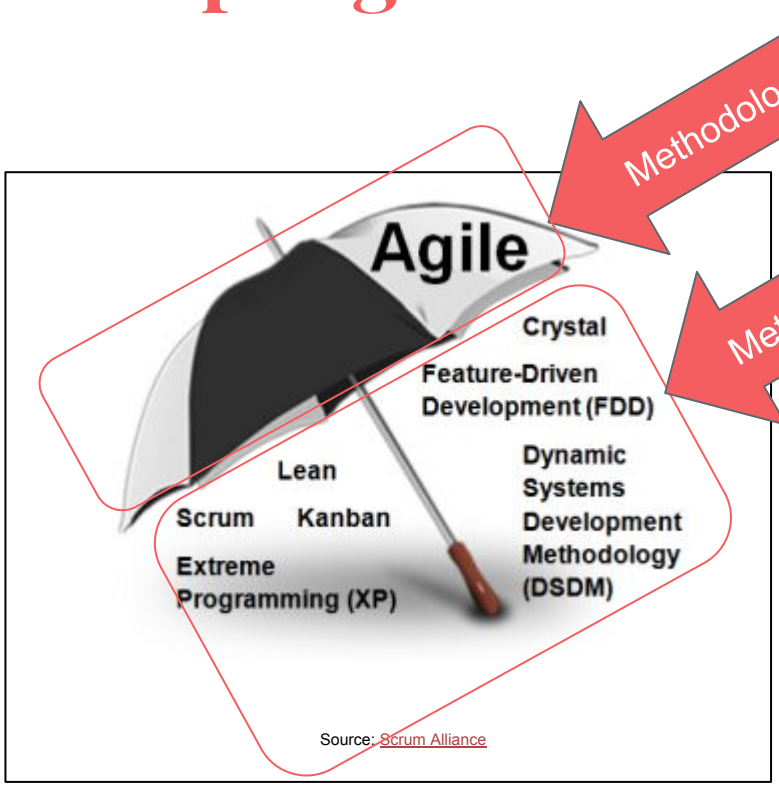
- Project is a single sequential arrangement of activities
- In extremis each activity has a person fulfilling that role



[source]

- Project a series of time boxed iterations
- Each iteration has requirements, design, construction, testing
- Each iteration delivers a version of software
- Feedback on each iteration can change direction
- Team is collaborative

Recap: Agile Umbrella



Agile is an “umbrella term” to describe ***a set*** of software development **methods and practices** ***based on the values and principles*** of the Agile Manifesto

Agenda

- ❏ Software development method
- ❏ Scrum Walkthrough
- ❏ Kanban
- ❏ Applied to Project

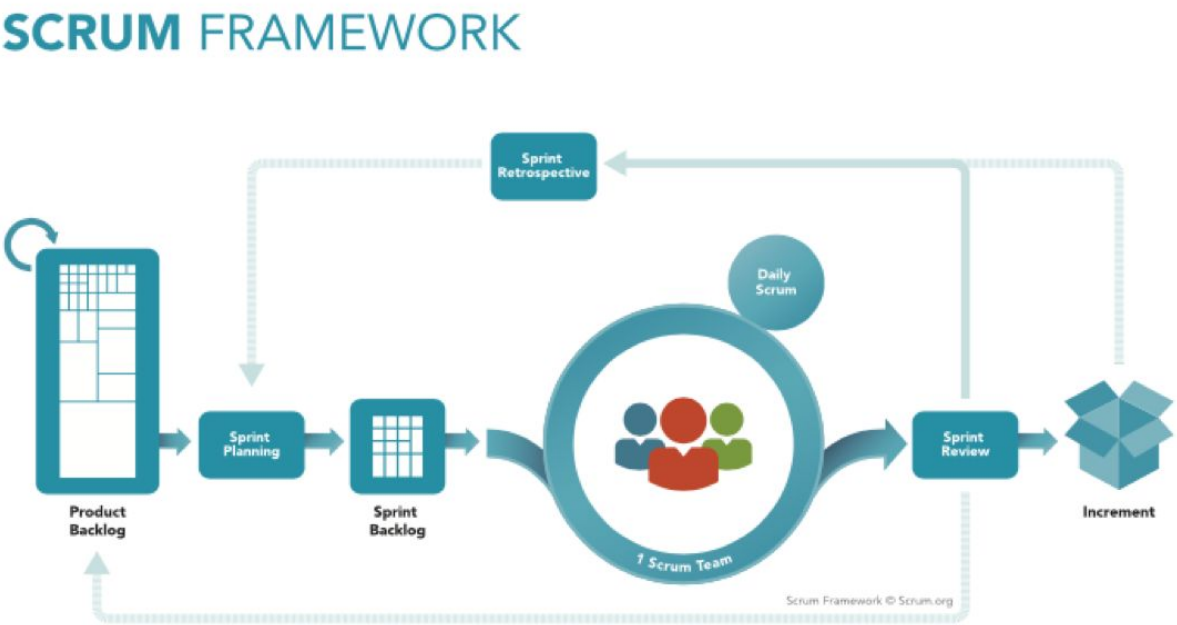
Software Development Methods

- **Theory:** Did this method come from any theoretical underpinnings?
- **Structure:** How is a project/iteration structured?
- **Activities:** Does it define activities?
- **Roles:** Do individuals take on any roles and if so what are the responsibilities?
- **Ceremonies:** Are there any prescribed meetings? What is the input, decisions, output and attendees of those meeting?
- **Artifacts:** Are there any artifacts (documents, plans)?
- **Technical Practices:** Is it associated or require technical practices?

(In OOAD a method is a concrete subclass of an abstract methodology superclass)

Scrum

Scrum Process



https://www.scrumalliance.org/

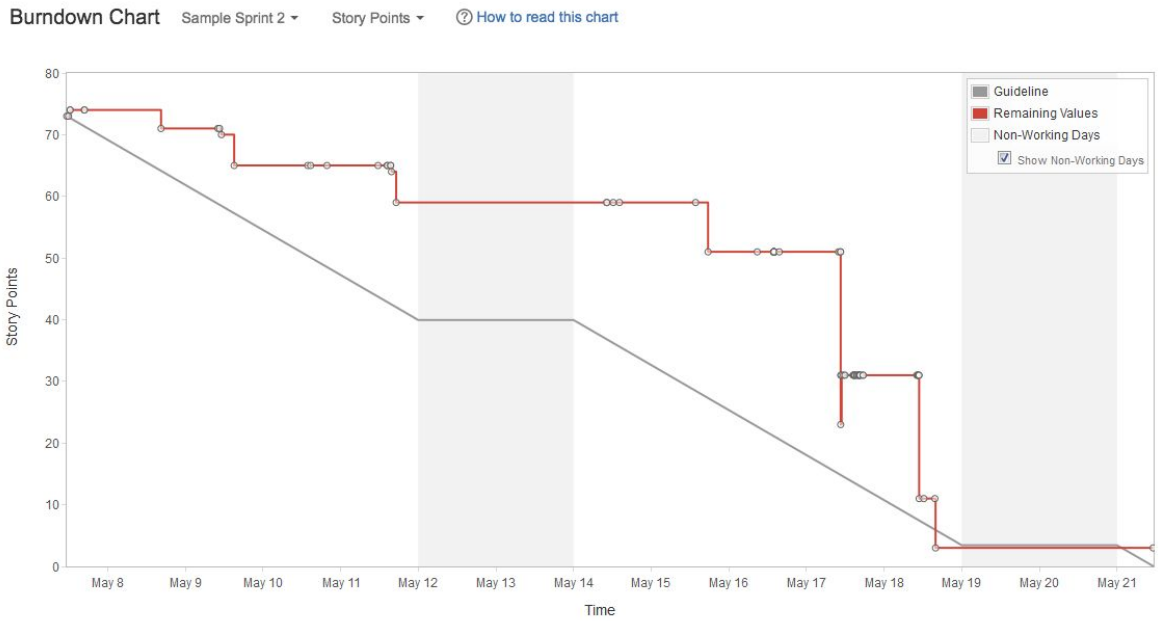
Roles

Role	Comments
Product Owner	• "maximizing the value of the product" • "Clearly expressing Product Backlog items" • "The Product Owner may do the [work or dev but the] Product Owner remains accountable" (see RACI) <Note - no specification of where the backlog came from …>
Scrum Master	
Development Team	• They are self-organizing. No one (not even the Scrum Master) tells the Development Team how to turn Product Backlog into Increments of potentially releasable functionality; • Development Teams are cross-functional, with all the skills as a team necessary to create a product Increment; • Scrum recognizes no titles for Development Team members, regardless of the work being performed by the person; <How do we contrast these statement with predictive methodology approach to people?>

Scrum by Construction

How do I ….. ?	Ceremony	Artifact
Keep track of *everything* that needs to be done?		
Prioritize what to put in this iteration?		
- Estimate the time taken to complete a feature?		
- Know how ‘much’ the team fit into an iteration?		
How will we build what we agreed?		
During the sprint keep the team connected & productive?		
During the sprint work out if we are on track to complete?		
We built something. How do we get feedback?		
That was fun/horrid sprint. How do we make it better next time?		
What do we do next sprint?		

Velocity



				Story Points		
Date	**Issue**	**Event Type**	**Event Detail**	**Inc.**	**Dec.**	**Remaining**
07/05/2012 11:22	GHS-4767	Sprint start		8		
	GHS-4862			5		
	GHS-4910			8		
	GHS-4913			20		
	GHS-4920			5		

Kanban

Theory

Roots in manufacturing. Toyota engineers were trying to optimize manufacturing processes (TPS and TOC)

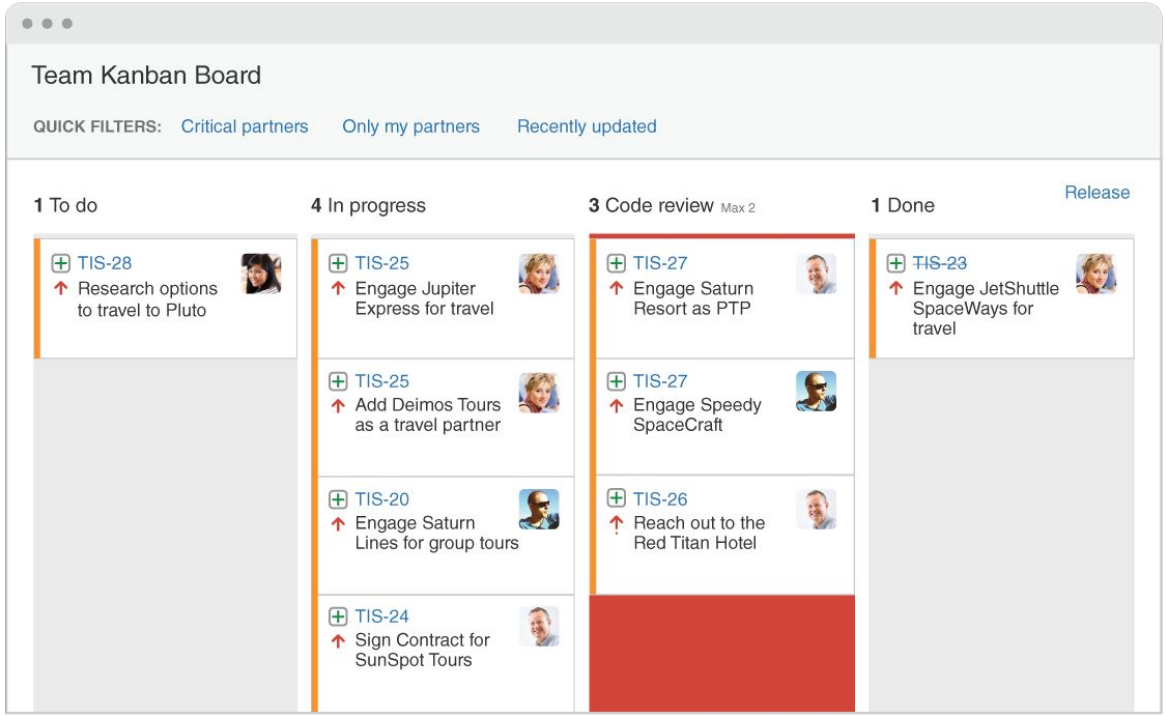
- How can you efficiently flow raw materials to product[1]?
- This led to a series of advances (TPS, TOC and Kanban as a supporting tool)

Kanban was then used as an inspiration/applied to the software problem domain

[1]Showerthought: Is software engineering flow of value idea to production?

Kanban Applied to Software

1. Visualize Workflow
2. Limit WIP
3. Focus on Flow
4. Continuous Improvement



https://www.atlassian.com/agile/kanban

Comparison

Kanban & Scrum: Two Different Agile Methods

Aspect	Scrum	Kanban
Theory	empirical process control: transparency, inspection, and adaptation	“Pull Systems” / Theory of Constraints
Activity Scope	Planning -> Production (does not really specify where the backlog comes from but often used with product vision to initially populate backlog)	Work to be done -> Production (does not really specify where the backlog comes from but often used with product vision to initially populate backlog)
Structure	Sprints/Batches	Continuous
Roles	Product Owner, Scrum Master, Engineer	(Doesn’t dictate)
Ceremonies	Sprint Planning, Daily Scrum, Retrospective	(Doesn’t dictate)
Artifacts	Backlog, Epics, Stories, Sprint Goal, Sprint Backlog, Velocity	Cards (Features/Tasks), Cycle Time
Technical Practices	Both associated with CI, CD, TDD	

Interview Question

A typical interview question is “tell me about the methodology/ SDLC/ process at your previous job?”

- “We were agile” as a *standalone* response does not make sense
- “We did Scrum” makes slightly more sense but did you really do everything? There was nothing adapted and what about the other aspects Scrum does not dictate? What was working well or not? Were you ‘doing scrum’ but not ‘being agile”?

More mature answer is a more nuanced discussion and evaluation of strengths and weaknesses ….

- We were loosely based on <Kanban/Scrum/Other> but we adapted it. Our business is mainly <X> so <Kanban/Scrum/Other> was a good fit. We were pretty mature at <practices A,B,C, Backlog, Sprint Planning, etc>. We didn’t use <planning poker, pair programming, etc> and were experimenting with <Cucumber/etc>. Our areas of improvement were <retrospectives or code review>

Project Advice

Disclaimer

“Ewan - you extol engineering judgement and selecting the right solutions depending on the situation. Yet …. You are proposing a methodology for teams? Are you a raging hypocrite?”

Not entirely:

- We are learning so I want to give guide rails
- Your projects are different but share certain characteristics
  (team size, duration, experience, complexity)
- Experience with ‘a’ methodology is the key learning outcome
- You are free/*encouraged* to adapt the framework
- My advice is to keep it *very simple* and grow

Project Advice: Kanplan

1. Agree roles (product owner, engineers, perhaps scrum master or collapse into PO)
2. Load up the first set of features as product backlog in Trello

   (PRD roadmap 1st column)
3. Pick a *very* pragmatic goal for your first sprint[1] select from backlog
4. Have a design meeting (yes - we have not taught arch or design. The assumption is we start with web-app)
5. Define your Kanban workflow

   (I suggest starting with 'backlog, sprint, analysis, ready, development, done'. Analysis is where features ↠ tasks)
6. Start to pull tasks from sprint to done
7. Tasks are only done once tests are done

[1] your first sprint has a lot of setup cost & you may be surprised by difficulty of producing larger software.
You also don't know your velocity from previous sprints. You are encouraged to introspect each card and perform some complexity estimation
Remember sprint != all backlog and you are unlikely to reach MVP in a single sprint

Reading

Reading	Optionality
Scrum Official Guide	Required
Velocity	Required
Atlassian Kanban	Required
Kanplan	Required
Sprint Goal	Required
Scrum vs Kanban	Optional