Digital Transformation

Week 2

Planning and Coordination in Modern Software Processes

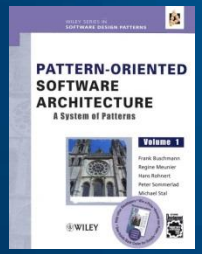
Learning Outcomes

Appreciate the need for planning regardless of software process

Plan for software process choice

Prioritisation in Iterative and Evolutionary Development

Know how to use a Kanban board



Ref: M. Cohn – Agile Estimating and Planning
D. Anderson - Kanban

Plan

- Week 1-4: Opportunity Assessment – Assessment 1 - Background Research and Innovation Plan to submit week 4
- Week 5-6: Consolidation of Requirements within Group - Assessment 2 – Assessment 2 (Software Process Choice, Software architecture and design)
- Week 7 - 11: Sprint
- Week 12: Documentation Assessment 3 (Report outlining solution delivery, critical analysis of solution, team and individual performances. Lessons learned. Week 12
- Pitch – Week 13

The problem with Planning

- **“If you fail to plan, you are planning to fail!”**

  Benjamin Franklin
- “**Give me six hours to chop down a tree and I will spend the first four sharpening the axe.”**

  Abraham Lincoln
- **Life is what happens to you while you're busy making other plans.”**

  Allen Saunders
- **“Plans are of little importance, but planning is essential.”**

  Winston Churchill
- **“The best laid schemes o' mice an' men gang aft agley.”**

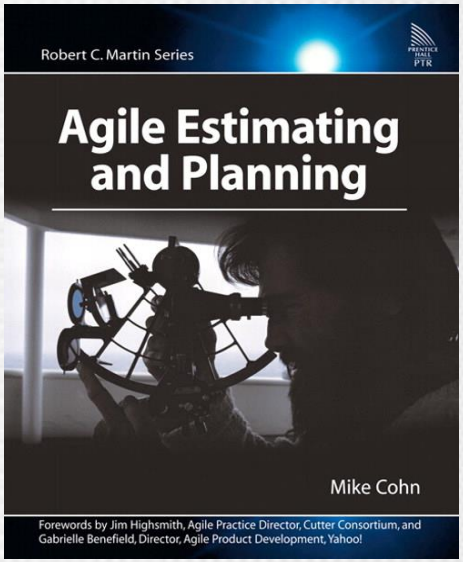
  Robert Burns
- **“If you don't know where you are going, you'll end up someplace else.”**

  Yogi Berra

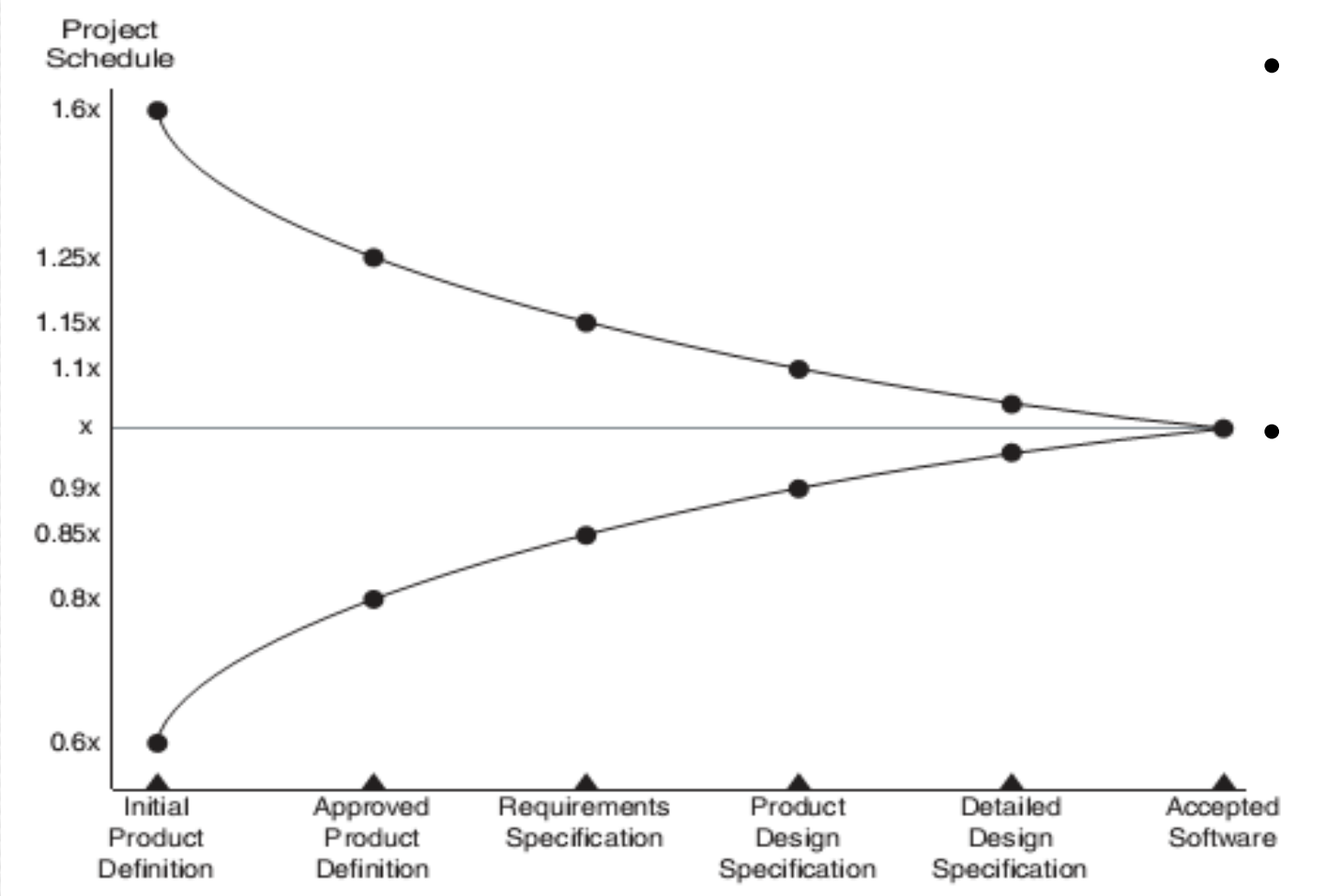
The problem with Planning

- "Planning is difficult, and plans are often wrong. Teams often respond to this by going to one of two extremes:
  - They either do *no planning at all*, or they put so much effort into their plans that they become *convinced that the plans must be right*."

Mike Cohn in Agile Estimation



Cone of uncertainty



- Feasibility stage
  - estimates x0.6 to x 1.6
  - E.g. 20 weeks could be 16-32 weeks
- Post Requirements
  - +/- 15%

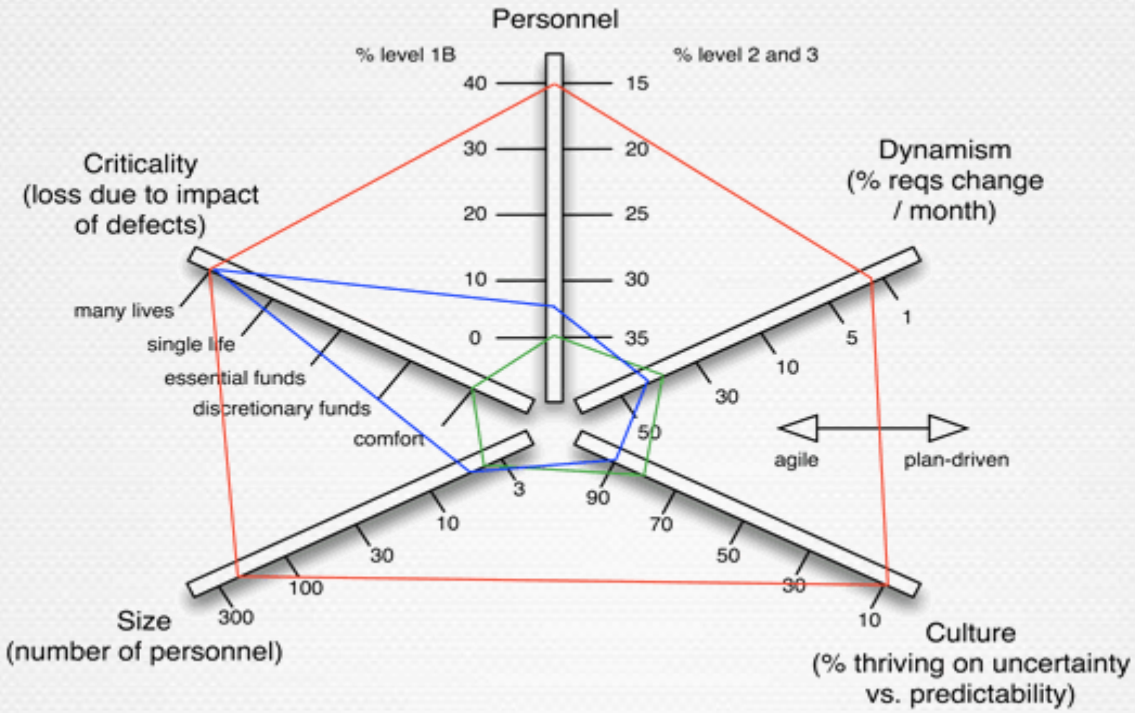
Plans get better with time/ info

- Project Management Institute (PMI)
  - initial order of magnitude estimate +75% to -25%
  - Next estimate = budgetary estimate of +25% to –10%,
  - final definitive estimate+10% to –5%
- **Plans get better as information becomes more certain**
  - **So, why do it at the start at all?**

Planning for Waterfall

- Assumes (almost) everything can be known upfront
- Planning is therefore a matter of estimating and assigning times to tasks
  - Plus sign offs of each stage
- Estimation – Expert Judgement, COCOMO, Function Points, … other models
- You have probably done this before
- Most approaches nowadays are iterative and evolutionary
  - But waterfall sometimes the best approach
  - Need Risk approach

Agility-Discipline Assessment

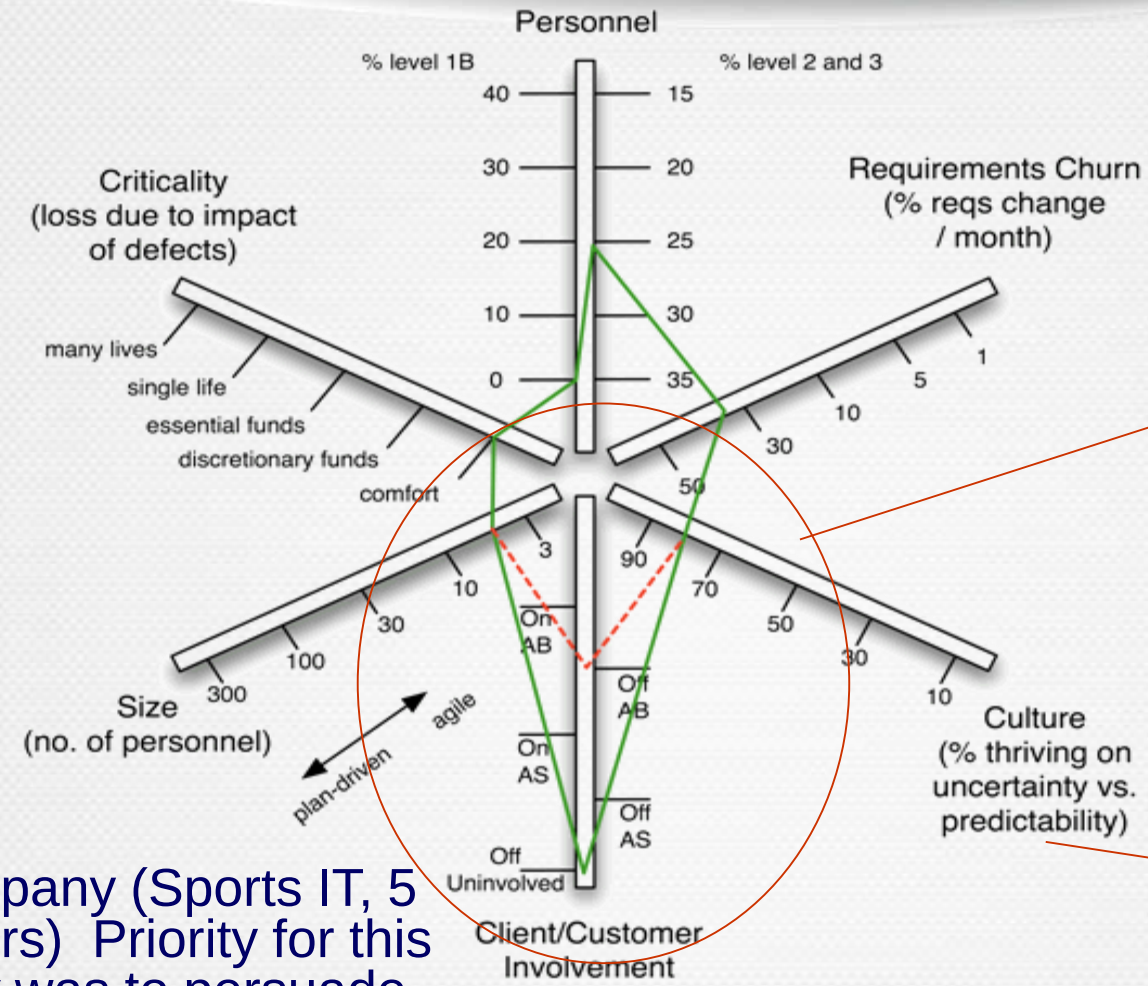


- Boehm and Turner approach
- Idea is that the process can be tailored with the right amount of agility and of discipline

Skill Level

Level	Characteristics
3	Able to revise a method (break its rules) to fit an unprecedented situation.
2	Able to tailor a method to fit a precedented new situation. Can manage a small, precedented agile or plan-driven project but would need level 3 guidance on complex, unprecedented projects.
1A	With training, able to perform discretionary method steps (e.g. sizing tasks for project timescales, composing patterns, architecture re-engineering). With experience, can become level 2. 1A's perform well in all teams with guidance from level 2 people.
1B	With training, able to perform procedural method steps (e.g. coding a class method, using a CM tool, performing a build/installation/test, writing a test document). With experience, can maser some level 1A skills. May slow down an agile team but will perform well in a plan-driven team.
-1	May have technical skills, but unable or unwilling to collaborate or follow share methods. Not good on an agile or plan-driven team.

Example Company – adapted assessment



- Project post-mortem – clearly biggest risk = customer non-involvement
- Added new axis
- OnAB = On-site Agile Believer
- OffAB = Off-site Agile Believer
- OnAS = On site Agile Sceptic
- OffAs = Off-site Agile Sceptic
- Offuninvolved = Off-site no interest

- This company (Sports IT, 5 developers) Priority for this company was to persuade customers of agile approach

Agility-Discipline Assessment Example of Risk Mitigation

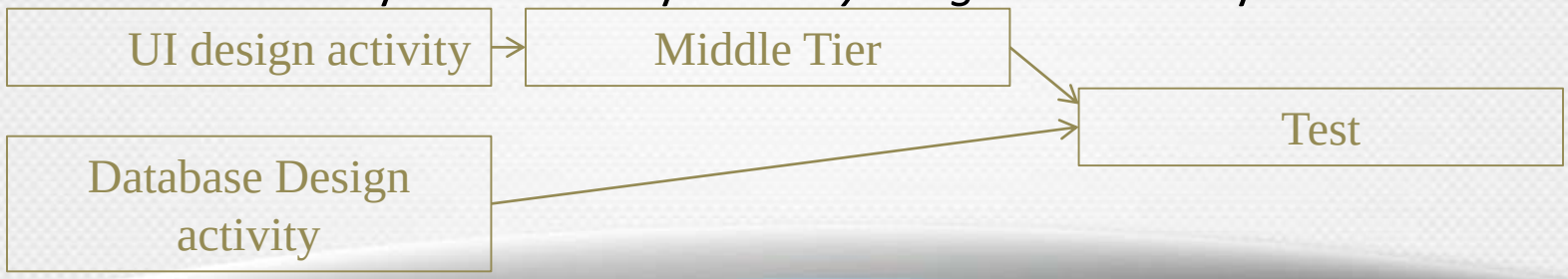
- **Off-site Uninvolved Customer Risk**
  - mitigation strategy = Off-Site Uninvolved → Off AB
    - Mechanism:
      - incremental delivery with average development cycle of 6 - 8 weeks
      - weekly incremental delivery for at least the last 3 weeks of any project. (moving an)
      - incremental releases made contractually required.
      - User acceptance testing part of the incremental approach.

12

Why Plans Fail – Activity/Feature

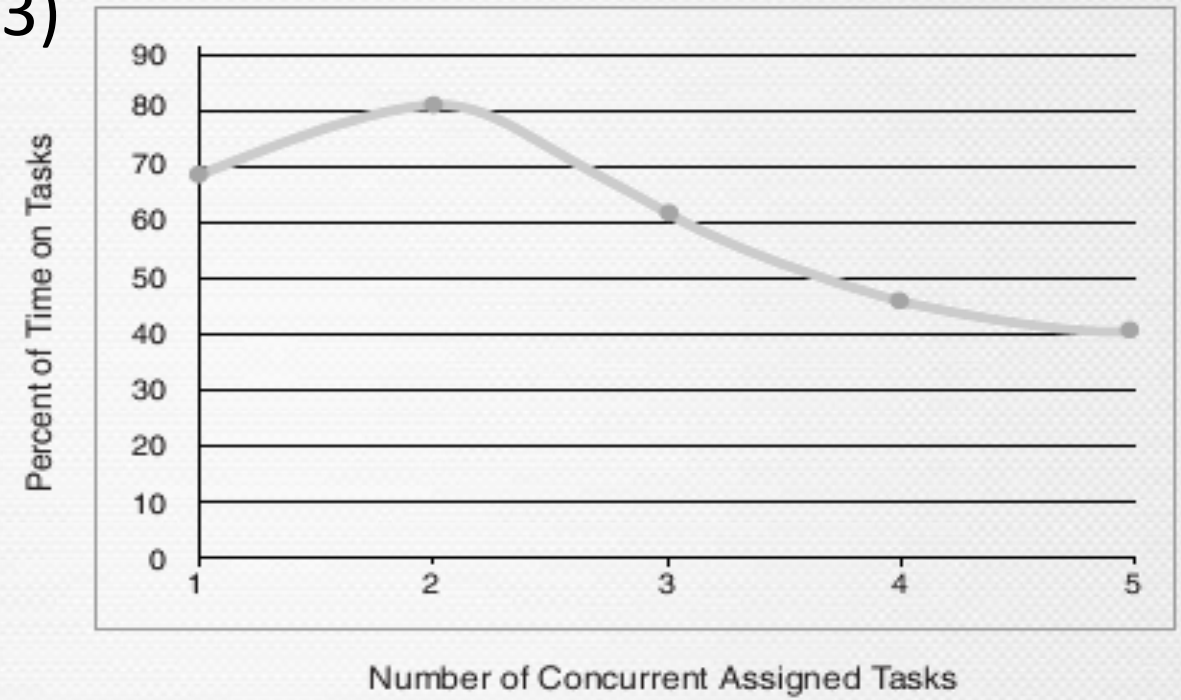
- Planning by activity – should be by feature
  - Customers don't get value from activities
  - Parkinson's Law – "Work expands to fill the time available for its completion" - nothing finishes early
  - Activities Are Not Independent (one underestimated probably means others too)
  - Lateness passed down the schedule

- Test activity will be delayed if *anything else* is delayed



Why Plans Fail - Multitasking

- Multitasking – time spent on value-adding work falls dramatically when > 2 tasks is progress (Clark and Wheelwright -1993)



Multitasking exercise

	Fibonacci	Alphabet	x 7	Roman	Consonants	Primes
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						

Why Plans Fail – Not prioritising

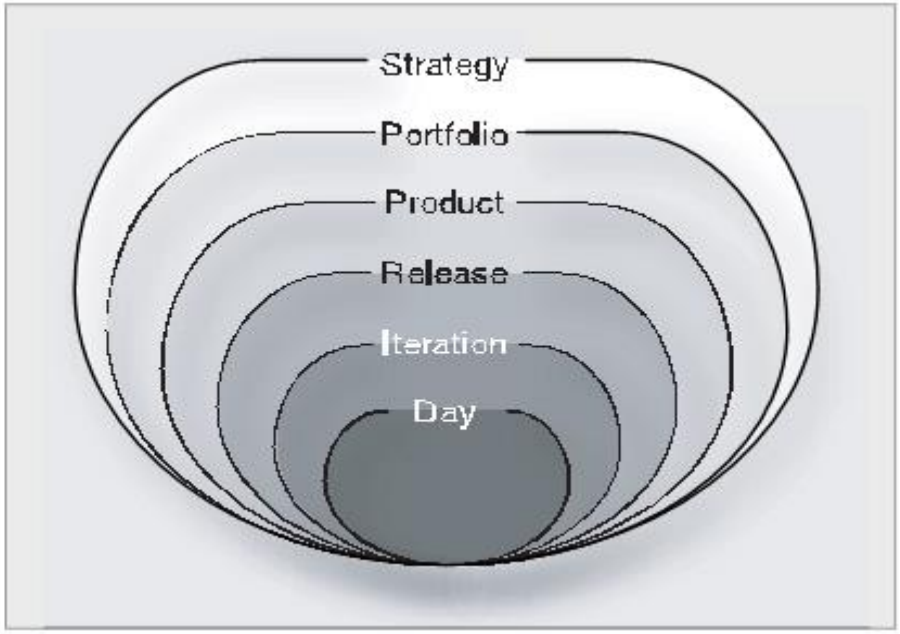
- Often plans are not prioritized value to the users and customer
- Often created assuming all activities will be completed
- What happens at the end of the project in that case?
  - De-scoping – stuff is dropped (possibly high value stuff)

Why plans fail - Ignoring Uncertainty

- Flawed Assumptions
  - Users will not change their minds?
  - Users will not refine their opinions
  - Users will not come up with new needs during the period covered by the plan
  - Uncertainty about how we will build the product
  - Thinking we can assign precise estimates to imprecise work
  - Thinking we can identify every activity needed
  - Giving exact release dates
- Handling uncertainty
  - Add probabilities to estimates/ delivery dates as ranges
  - short iterations + show and tell to customer
    - Allows feedback, adjustment, replanning

Multiple Levels of Planning

- we cannot see past the horizon
- ∴ a project is at risk if its planning extends beyond our horizon
- Every so often need to re-plan
- Horizons are
  - release
  - Iteration
  - current day.



- Release planning = which user stories will be developed for a new release of a product
  - Think about scope, schedule, and resources for a project
  - updated throughout the project (e.g. At start of iteration)
- Iteration planning = start of each iteration
  - product owner identifies high-priority stories
  - What tasks to develop these?
- Daily planning = usually daily stand-up meeting to coordinate work + synchronize efforts

Prioritisation in Agile - Factors

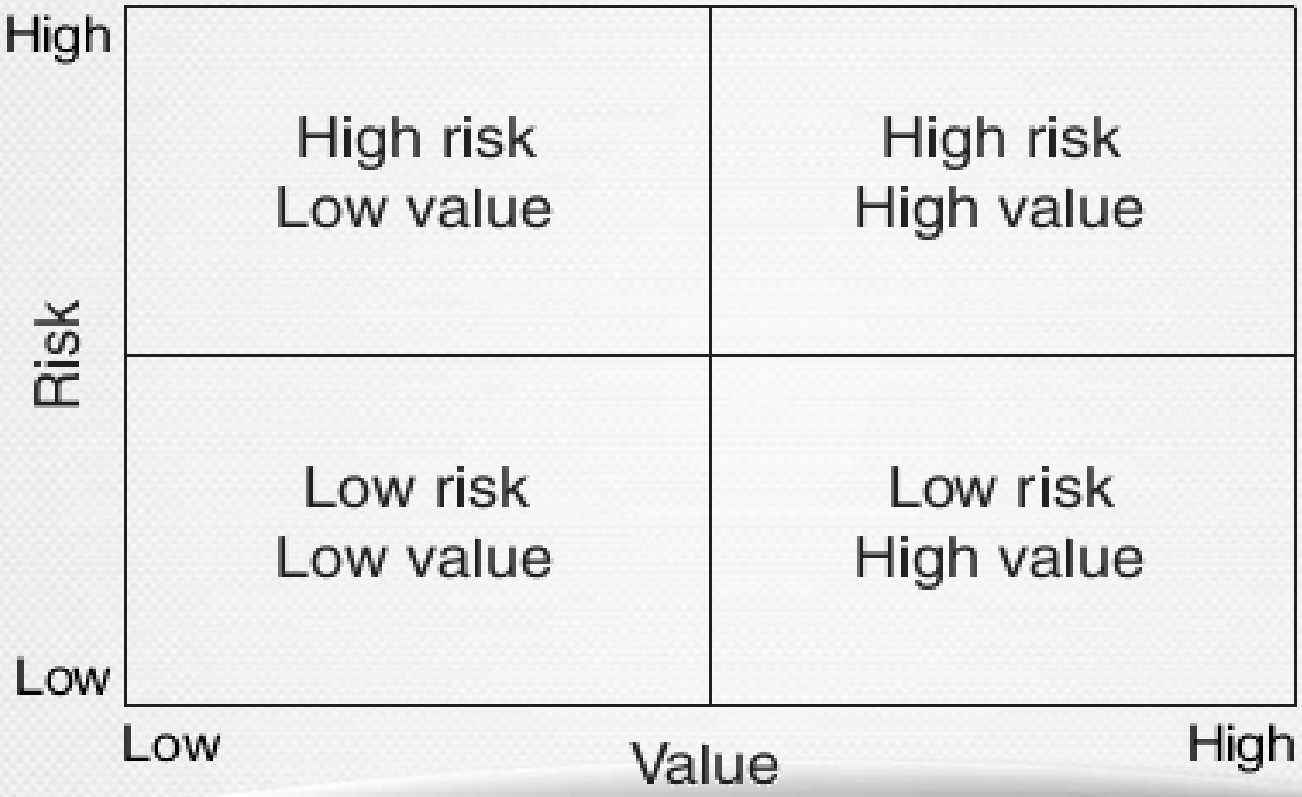
- Value: savings/ new sales
- Cost: Usually mainly effort (salaries)
  - Can change over time – requirements change
  - Reduce cost of change is by implementing a feature as late as possible
    - = no more time for change
- New Knowledge is valuable
  - Knowledge about the product
  - Knowledge about the project
  - reducing uncertainty
    - end uncertainty (product)
    - means uncertainty (project)

Prioritisation in Agile - Factors

- Risk = possibility of loss
  - Schedule risk (something is delayed)
  - Cost risk (something costs more than expected)
  - Functionality risk (something may not work)
- Should we start with the high-risk features?
  - Get them out of the way -see early on if they wreck the project
- Should we focus on the “juicy bits” i.e. high-value features
  - Get the customer on board and seeing ££££?

Prioritisation in Agile - Risk

- Need to consider Value and Risk!



Do First
Do Second
Do Third
Avoid

Other factors

Stakeholders: Need to satisfy each group of stakeholders

Effort constraints: limited for each release

Precedence Constraints : some requirements must be delivered before others

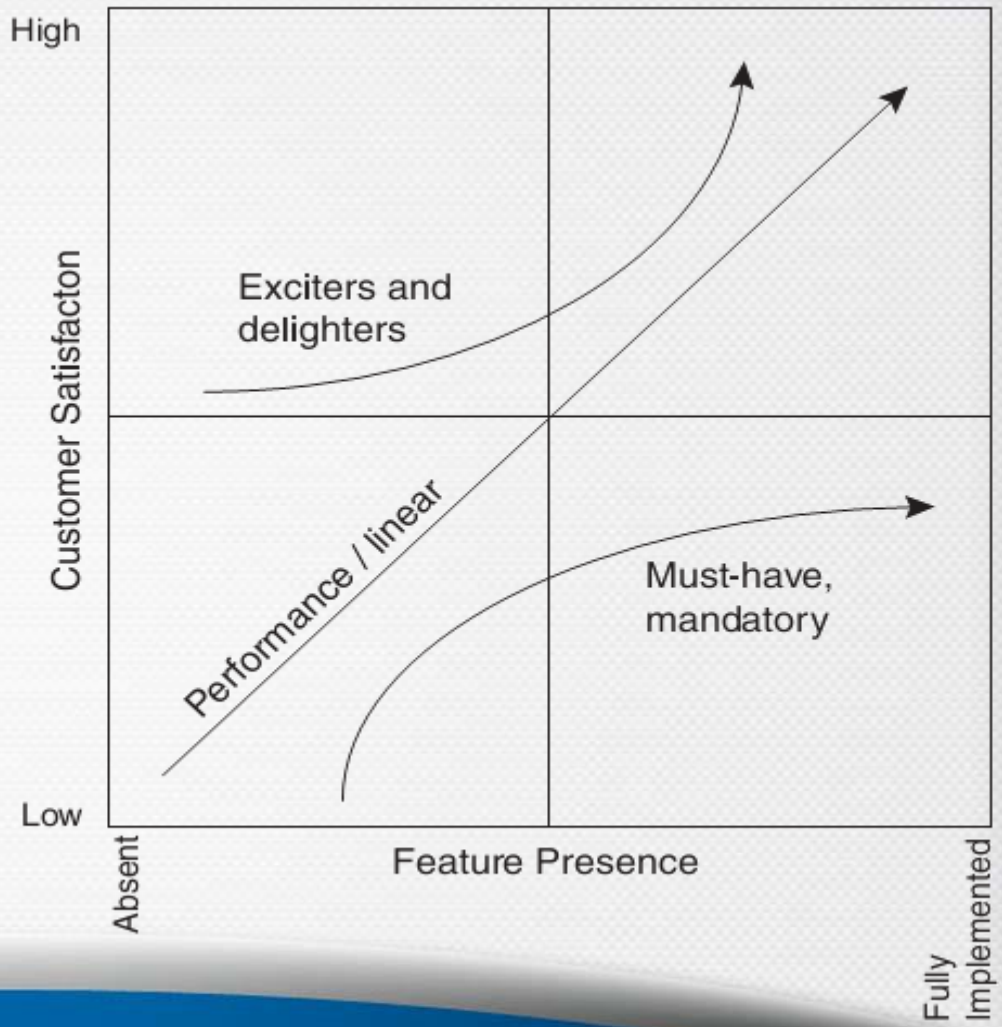
Resource constraints – some requirements must not be in the same release

Coupling Constraints : some requirements are only valuable if delivered together

Risk Constraints: Each release must be below an acceptable level of risk

Prioritisation - Desirability

- Noriaki Kano – Kano model of customer satisfaction. Feature categories:
  - Threshold (must-have) features
  - Linear features (the more, the better)
  - Exciters and delighters (provide great satisfaction – possibly would pay more for)



Is it threshold, linear or exciter?

- Kano asks 5 questions about a feature
  1. I like it that way.
  2. I expect it to be that way.
  3. I am neutral.
  4. I can live with it that way.
  5. I dislike it that way
- He then asks in two ways
  - How would you feel if that feature were present?
  - How would you feel if that feature were absent?
- We could collect these over large group and score/ analyse – or just a product owner

Release Planning in agile (briefly)

- Backlog of user stories + commitment to release frequently
- Need to plan (not necessarily commit to) what to release and when
- Why a release plan?
  - To think about how much is to be done to have a releasable product
  - Conveys expectations / timeframe e.g. For strategic planning
  - Guide for team – where they are going

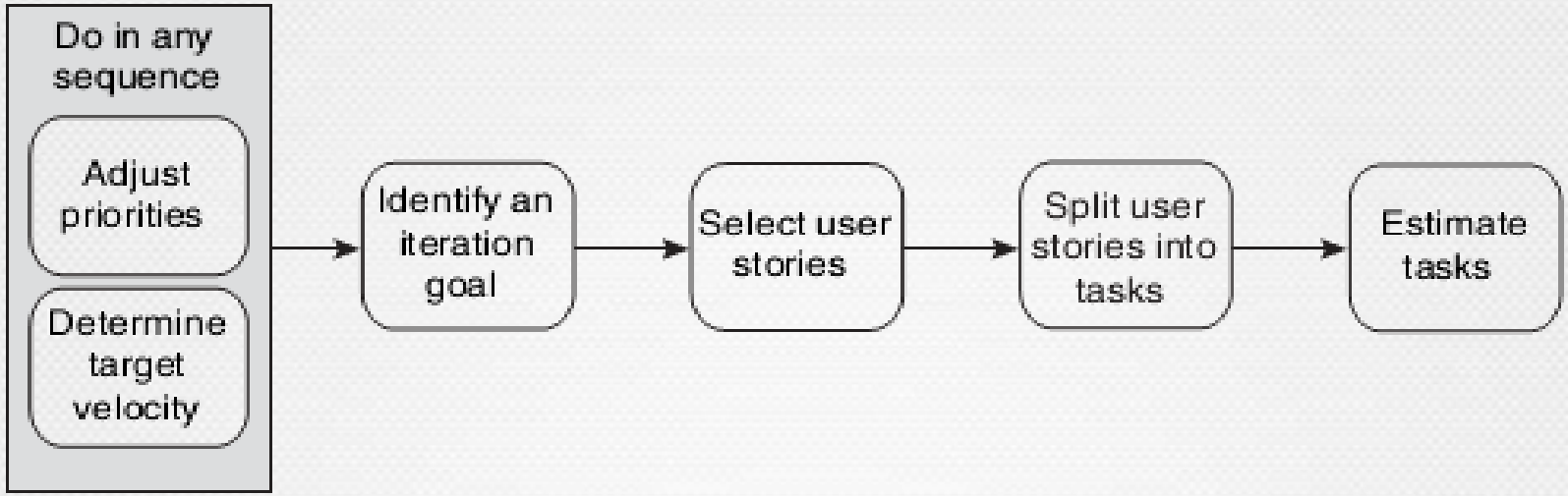
Release Planning in agile (briefly)

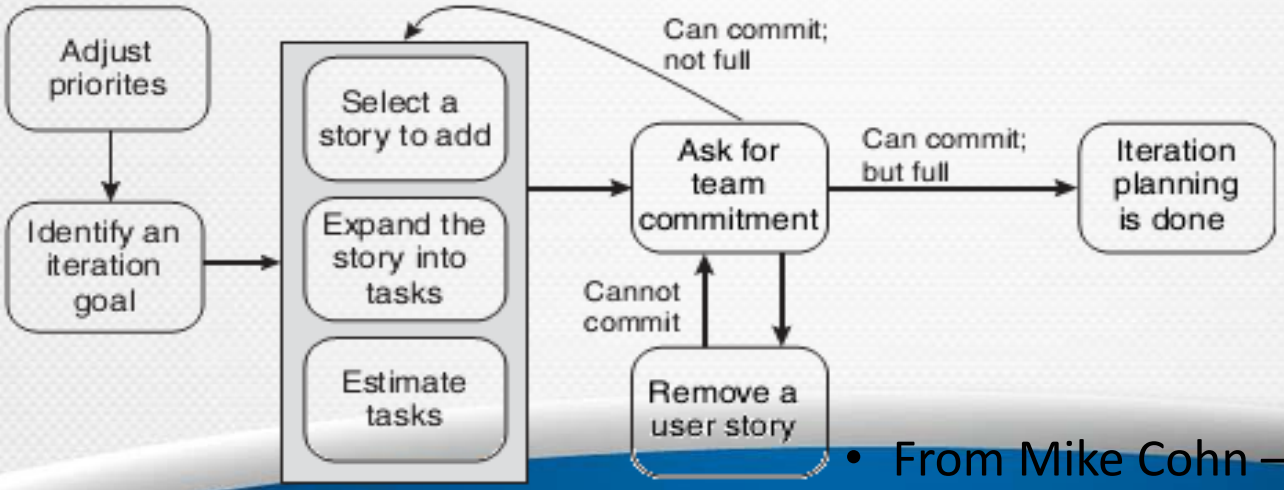
- Assign to next 2 releases based on priority
  - Decide on iteration length
  - Use story size and velocity to allocate to iterations

Iteration Planning in Agile (brief)

- Iteration planning - product owner + developers
- Spreadsheet or cards on a table /wall
- Input = User Stories - roughly estimated (story points/ ideal days)
- Process
  - Establish tasks
  - Team agrees task estimates (hours)
  - Team members create new tasks, if necessary
- Output = tasks + estimates + new knowledge
- Output ≠ allocation of tasks (only once iteration started)

Velocity vs. Commitment driven



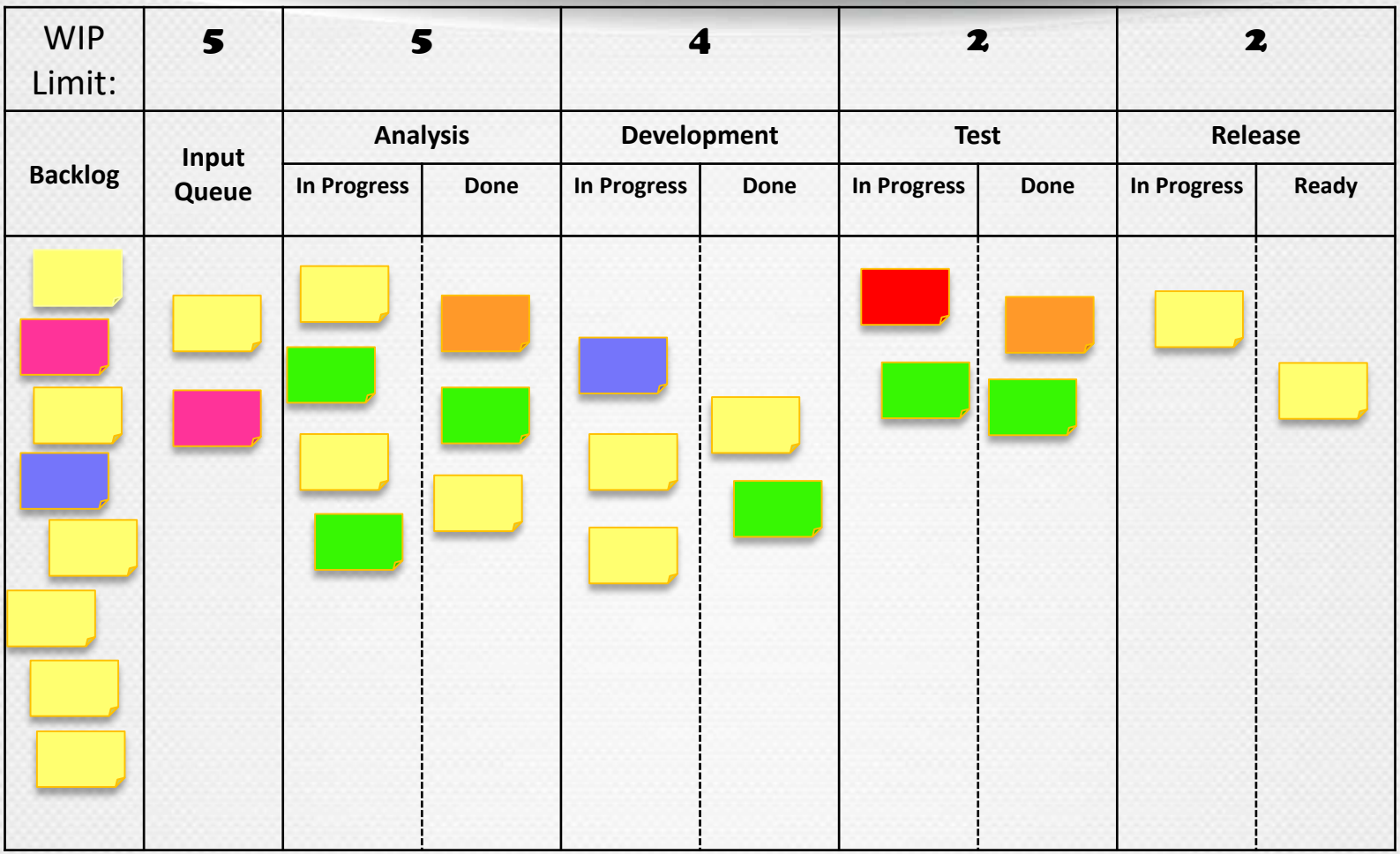


- From Mike Cohn – Agile Estimation

Kanban is different

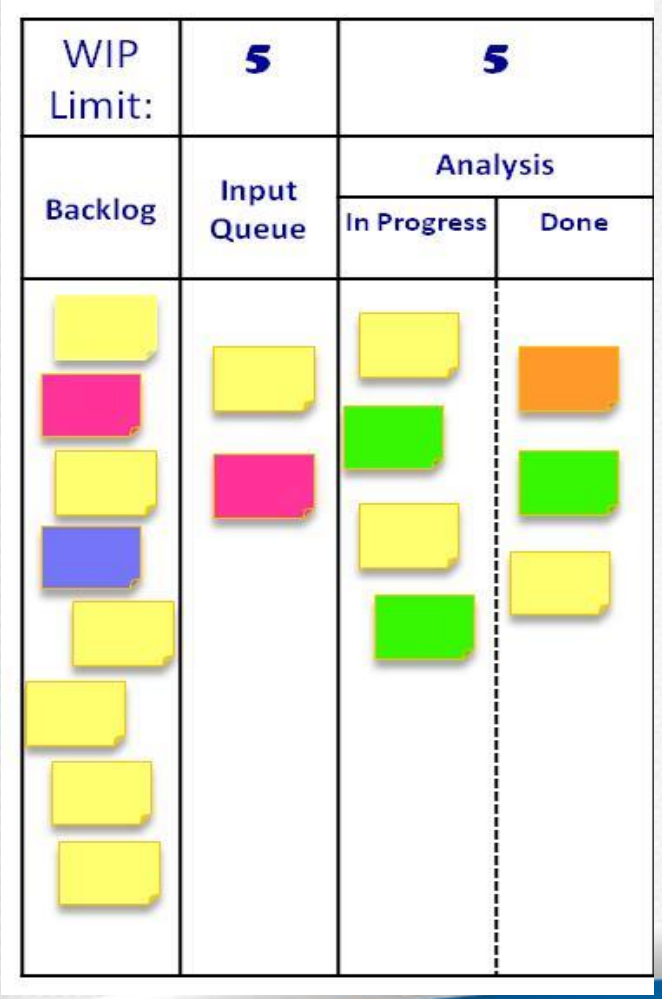
- Kanban is a pull system – so we don't really have the idea of forcing stories into an iteration or of timeboxing
- we develop as capacity is available (velocity and burndown are not so useful in Kanban)
- Kanban *has* a delivery cadence but what gets delivered is decided late on, not at the start
- Need trust of product owner/ customer in both cases
- Where priorities change frequently Kanban works well – pull in the higher priorities

Coordination via Card Wall



- Most popular form of coordination - card wall

Coordination via Card Wall

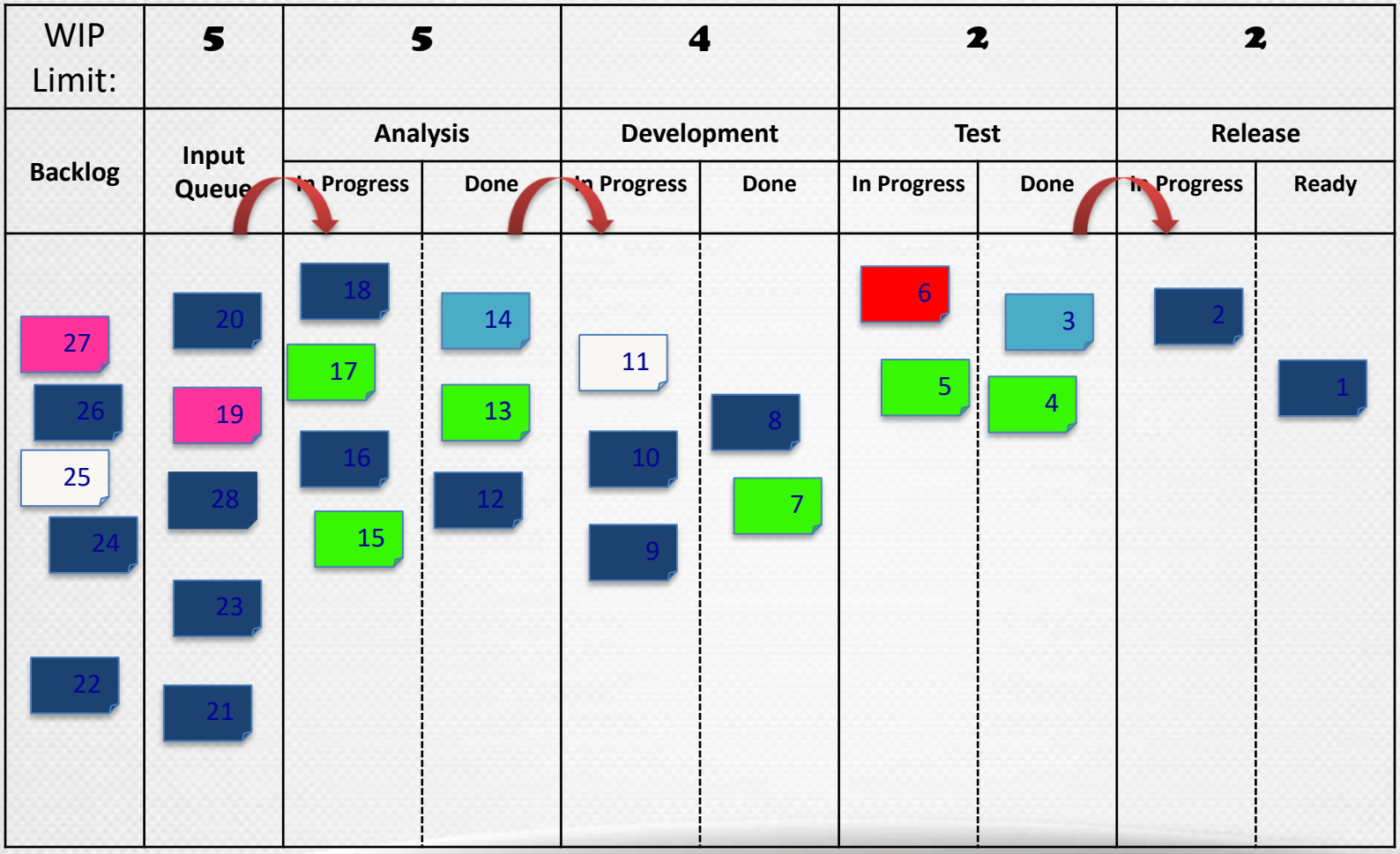


- Input Queue has a WIP limit of 5
- Analysis has WIP limit 5
  - Has currently 4 items in progress
  - So can *pull* in 1 more from the Input queue
  - '5-4=1' is the signal to pull
  - Input Queue will then have only one item left
    - » Signal to add 4 (5 -1) more to the input queue at the next prioritization meeting

Coordination Illustration

WIP Limit:	5	5		4		2		2	
Backlog	Input Queue	Analysis		Development		Test		Release	
		In Progress	Done	In Progress	Done	In Progress	Done	In Progress	Ready
28, 27, 26, 25, 24, 23, 22, 21	20, 19	18, 17, 16, 15	14, 13, 12	11, 10, 9	8, 7	6, 5	3, 4	2	1

Coordination Illustration



Coordination Illustration

WIP Limit:	5	5		4		2		2	
Backlog	Input Queue	Analysis		Development		Test		Release	
		In Progress	Done	In Progress	Done	In Progress	Done	In Progress	Ready
27, 26, 25, 24, 22	20, 19, 28, 23, 21	18, 17, 16, 15	14, 13, 12	11, 10, 9	8, 7	6, 5	3, 4	2	1

Some features completed (Done)

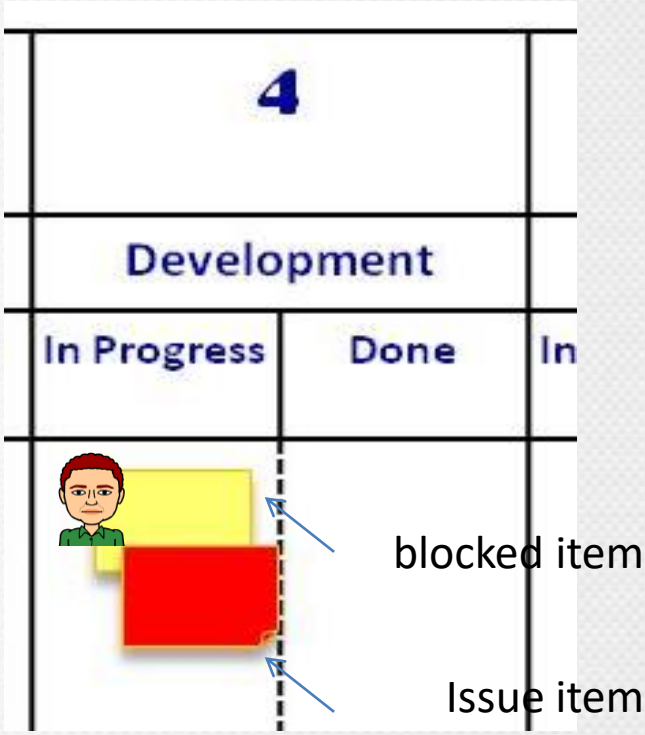
Need to define 'Done'

WIP Limit:	5	5		4		2		2	
Backlog	Input Queue	Analysis		Development		Test		Release	
		In Progress	Done	In Progress	Done	In Progress	Done	In Progress	Ready
27, 26, 25, 24, 22	20, 19, 28, 23, 21	18, 17, 16, 15	14, 13, 12	11, 10, 9	8, 7	6, 5	3, 4	2	1
		1) Goal is clear 2) Tasks defined		1) Code is Clean 2) Checked-in 3) Integrated + Tested		1) All tests pass		1) Customer accepts 2) Ready for prodcution	

Take definition of done as far as possible

Which item to pull?

- Team can choose for themselves
- Based on visual information e.g.
  - work item type (urgent bug?)
  - the class of service (e.g. “Expedite”, “Fixed Delivery”)
  - the due date (if there is one)
  - the age of the work item
- To help – more information can be added
  to the board
  - Combination work item type/ class of service.
  - Who is working on what?

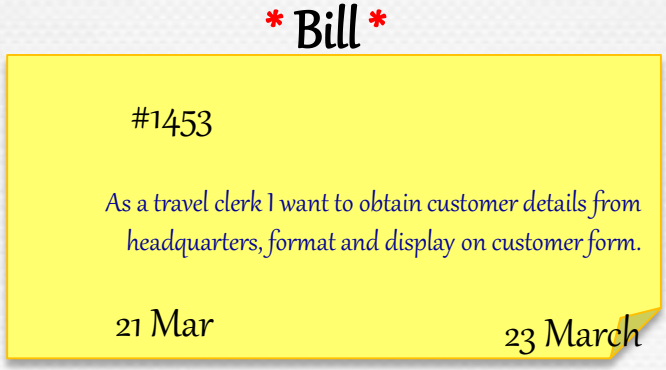


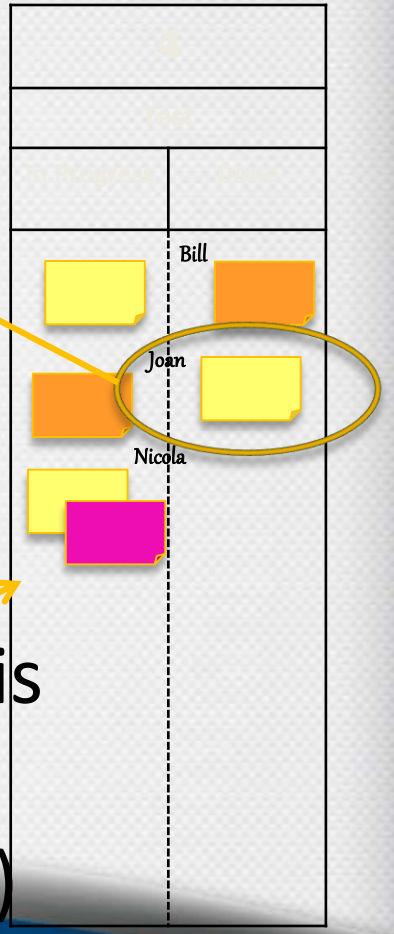
Which Items to Pull?

1) Expedited items
2) Fixed Date in danger
3) Oldest

Visual Clues

- E.g. This one has exceeded its SLA *



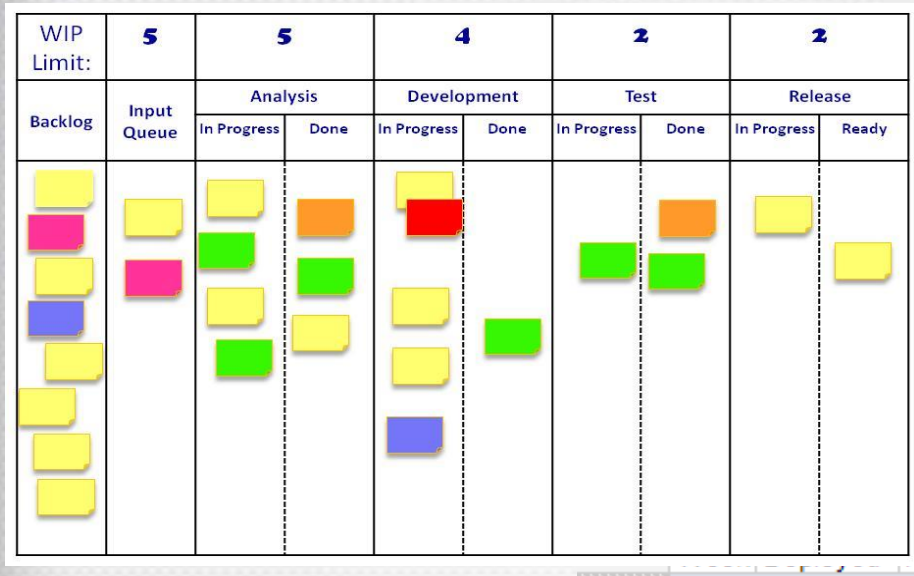


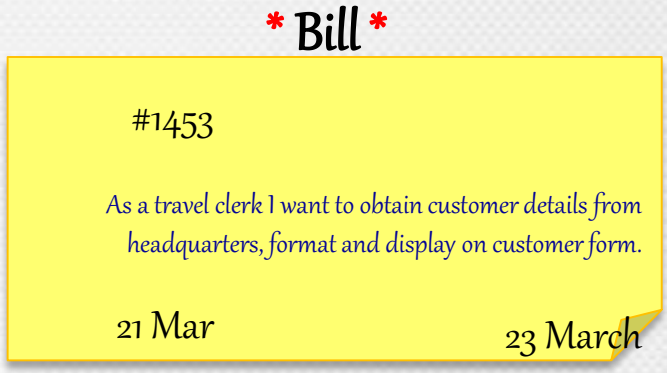
- overdue ?– this entered input queue on 21 March – deduce its age
- If an item is of a class of service that is a guaranteed delivery date – we can see if it is late * *
- Higher priority is blocked (pink issue ticket)

Self organising teams

- Kanban board = visual control mechanism
- enables team members to pull work *without* direction from their manager

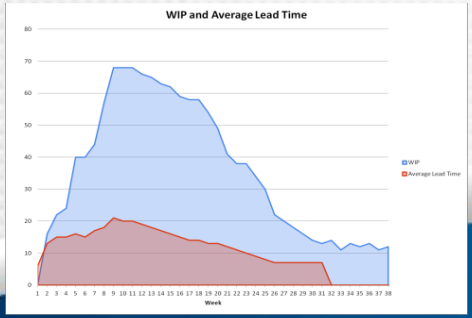
Electronic Tracking

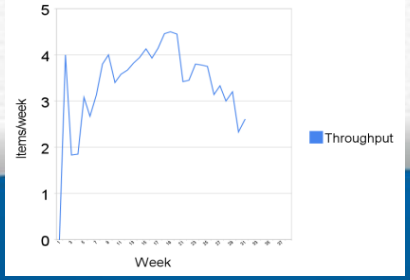


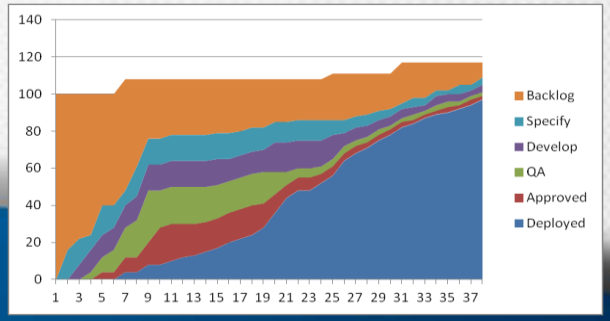


Snapshot
1 per week

			C	D	E	F	G	I	M	R
			Approved	QA	Develop	Specify	Backlog	WIP	Average Lead Time	Throughput
2	1	0	0	0	0	0	100	0	6	0.00
3	2	0	0	0	0	16	84	16	13	1.23
4	3	0	0	0	8	14	78	22	15	1.47
5	4	0	0	4	12	8	76	24	15	1.60
6	5	0	4	8	12	16	60	40	16	2.50
7	6	0	4	12	12	12	60	40	15	2.67
8	7	4	8	16	12	8	60	44	17	2.59

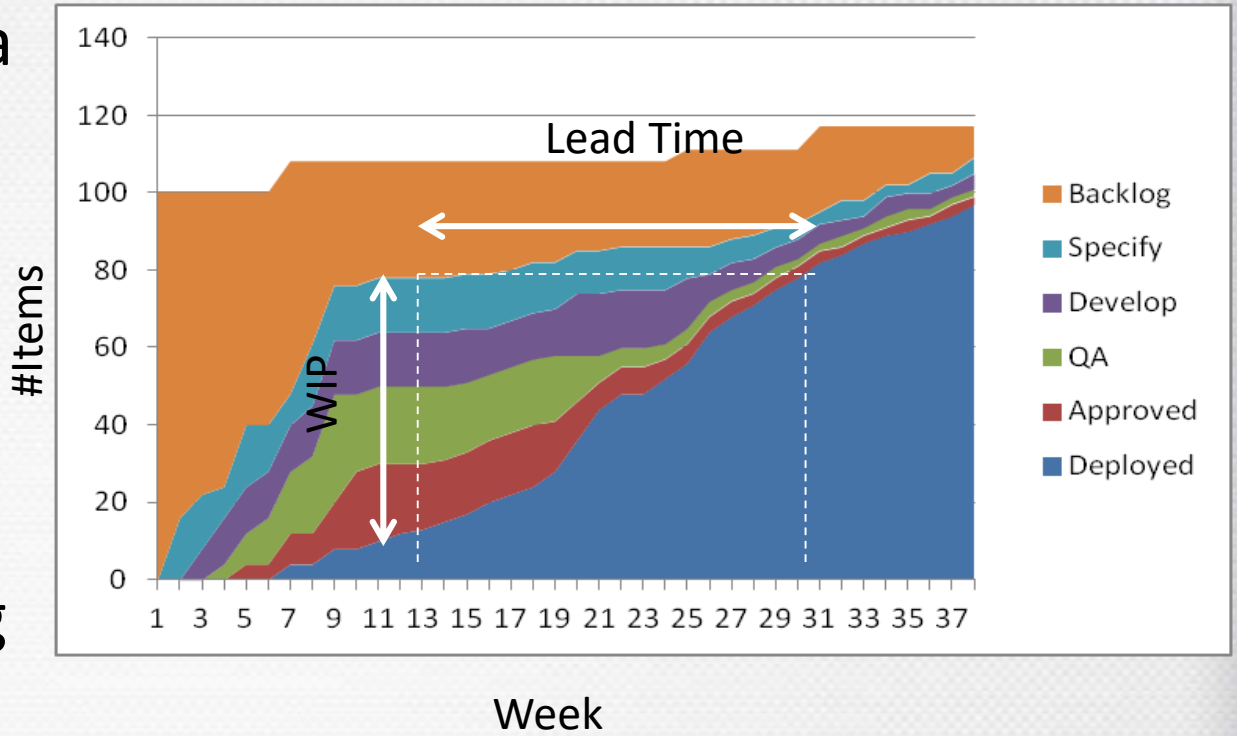






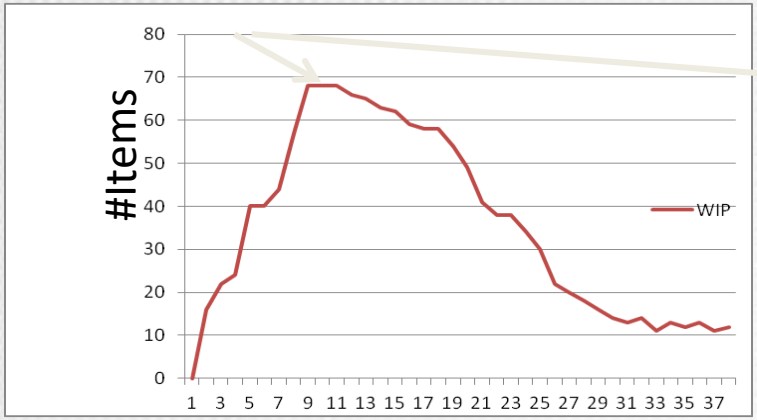
Tools allow monitoring of progress

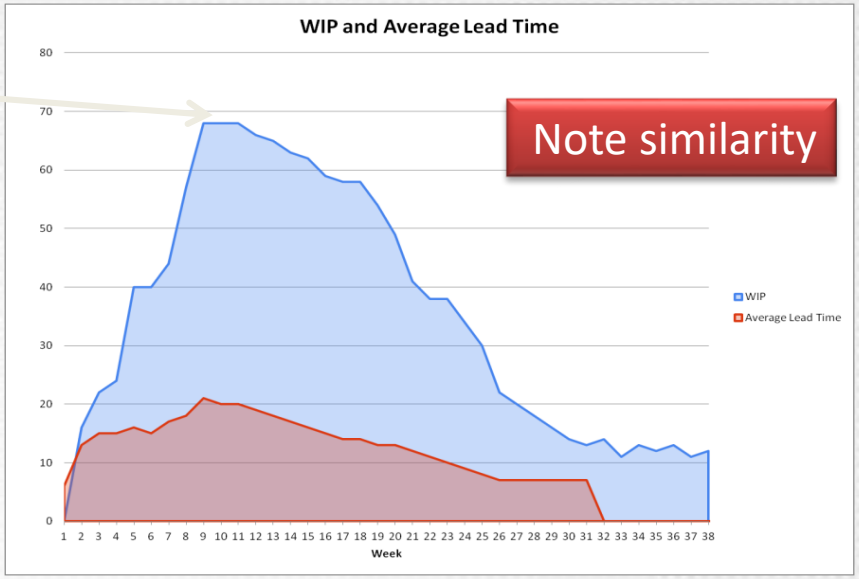
- CFD is a stacked area chart - each area represents a station in the development process
  - Lead time = entry to exit
- WIP – anything being worked on

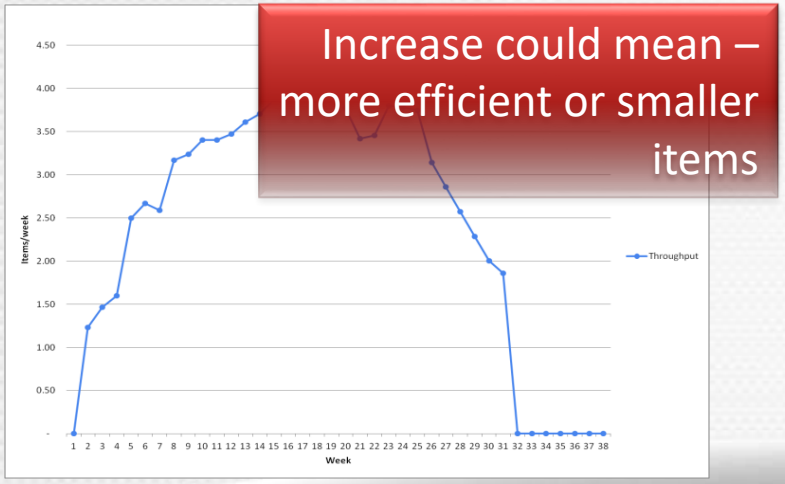


Tools allow monitoring of progress

Team reduces WIP!





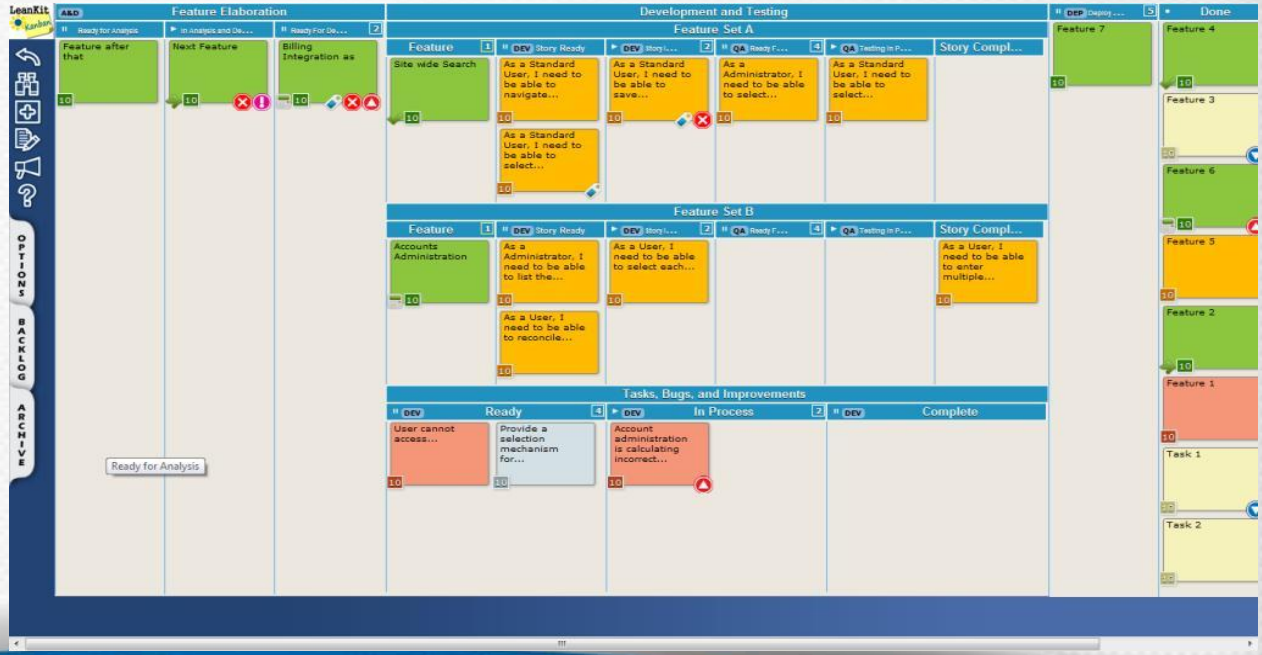


Throughput = rate of work items leaving

- **WIP = Arrival Rate * Lead Time**
  - **Arrival time is constant**
  - **Reduce WIP, reduce Lead time**

Electronic tools

- There are several more elaborate electronic devices which include virtual Kanban boards and instant distribution and analysis of data



Daily Standup Meetings

- You have probably seen this in Scrum
  - What have you done since yesterday's meeting?
  - What are you going to get done today?
  - What obstacles do you need to be removed?
- Kanban card wall obviates some of these
  - You can see what has been done
  - You may be able to see obstacles
- Instead – focus on **flow of work**

Daily Standup Meetings – how?

- Facilitator e.g. project manager will “walk the board.”
- Usually – Right to Left
  - i.e. in the direction of pull—through the cards on the board.
  - Facilitator can ask for a status update on a ticket or for more information on something not shown
- Particular emphasis on items that are blocked (pink ticket attached) or delayed due to defects (series of blue tickets attached).

Attention to blocking items

- Attention to items that seem stuck (obviously haven't moved or entered the input queue a long time ago)
  - Some teams may put a dot on tickets for each day in a single location
    - Is it blocked?
  - Discussion on who is working an issue and when it will be resolved.
  - Call for any other blocking issues not on the board/ impediments or help needed.

After Meeting

- After the standup
- Typically groups of 2 or 3 people.
  - Discuss a blocking issue
  - technical design/architecture / process issue.
    - Improvement in current product or current/future process.
- Scrum v Kanban standup
  - Scrum, the teams meet first and then send a delegate to a Scrum-of-Scrums for overall large project
  - Kanban – its the reverse —the overall project meeting happens first

Delivery Cadence

- In agile and in lean we are committed to regular deliveries of working software
- Lean has the idea of ‘cadence’
- “delivery cadence” = a pattern of delivery of working software at regular intervals
- E.g. We could decide to deliver every two weeks/ 26 times per year
- Or ‘every second Wednesday’

Delivery Cadence – in agile

- Agile development - time-boxed iterations
  - Iterations typically 1-4 weeks in length
  - Idea is that a steady “heartbeat” to a project is a good thing
    - Assumption: way to achieve this is strict time-boxes
  - At the start of the timebox - analysis, test plans, design, development, tests etc performed
  - At end of timebox, committed scope is completed (or with some de-scoping) + software delivered.
  - Cycle repeats in next timeboxed iteration
- = regular cadence

Delivery Cadence – in Kanban

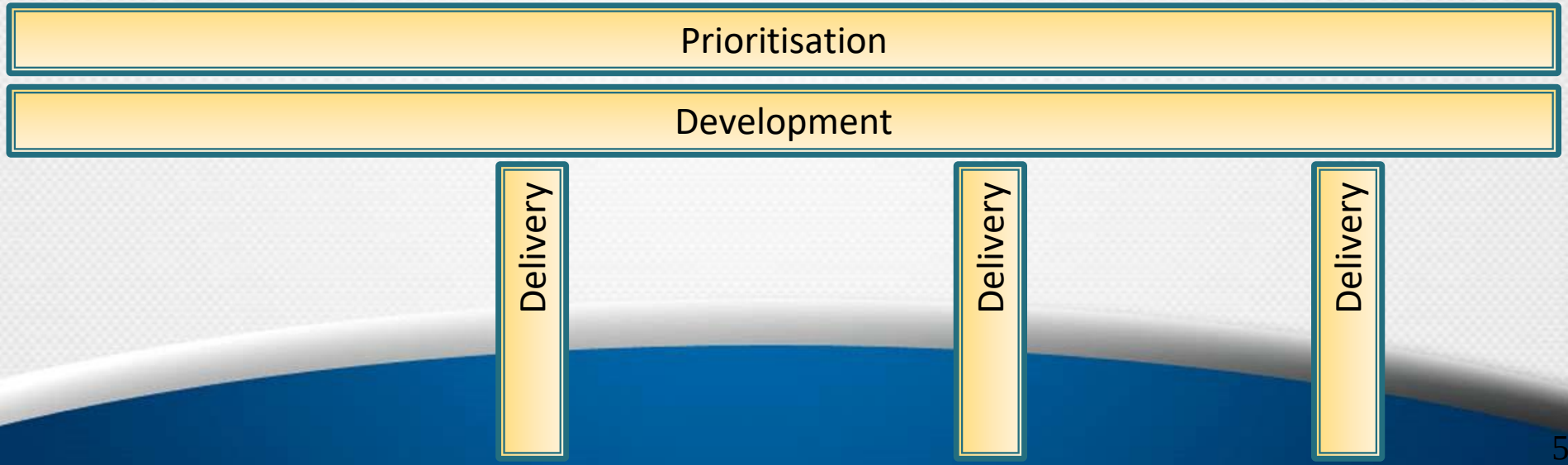
- Kanban does NOT use time-boxes
- Instead
  - prioritization, development, and delivery. The cadence of each is allowed to adjust to its own natural level.
  - Kanban teams still deliver software regularly, preferring a short timescale
  - No artificial forcing of things into time-boxes.

Agile vs Kanban cadence

- Decoupling of prioritisation, development, delivery

Agile:

Kanban: no forcing into time boxes; Deliver what is ready to be pulled into production



Some agile problems

- XP and Scrum have gradually adopted shorter iteration lengths
  - Typical Scrum was typically 4 weeks, now 2
  - Extreme Programming teams, was 2 weeks, now 1
  - BUT how to analyse work into small enough units to fit in the available time window
  - Approach adopted = reduce the size of stories to fit them into smaller iterations OR go back to bigger iterations

agile problems with timeboxes

- Smaller iterations - write stories on some aspect of the architecture or part-requirement.
  - E.g. story for the UI, a story for the persistence layer etc.
- Or break stories across several iterations in phases – 1st iteration analysis and test planning 2nd code development 3rd system testing and bug fixing
- These are all dysfunctions to fit the strategy of timeboxed iterations

Decouple development and delivery

- Some work is complete and ready for delivery
- Other work is in progress
- Unlikely that planning, estimation, and prioritisation discussions should all need to happen at the same pace as delivery and software release
  - different functions, often requiring the attention of different groups of people
  - coordination effort around delivery different from the coordination effort around prioritization of new work
  - Prioritization cadence ≠ delivery cadence

Digital Transformation

Week 2

Planning and Coordination in Modern Software Processes

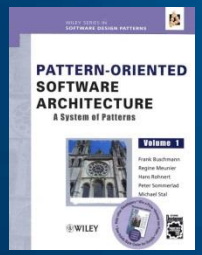
Learning Outcomes

Appreciate the need for planning regardless of software process

Plan for software process choice

Prioritisation in Iterative and Evolutionary Development

Know how to use a Kanban board



Ref: M. Cohn – Agile Estimating and Planning
D. Anderson - Kanban