Shifting the paradigm for women in IT sector

4. Measuring performance and preventing failure

(7.5 hours)

Co-funded by the European Union

shift4T

Funded by the European Union. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or Agency for Mobility and EU Programmes. Neither the European Union nor the granting authority can be held responsible for them.

Summary

4.1 How to measure performance (2.5 hours)

4.2 How to avoid failure (2.5 hours)

4.3 Reconciling Waterfall and Agile (2.5 hours)

4.1 How to measure performance (2.5 hours)

Co-funded by the European Union

shift4T

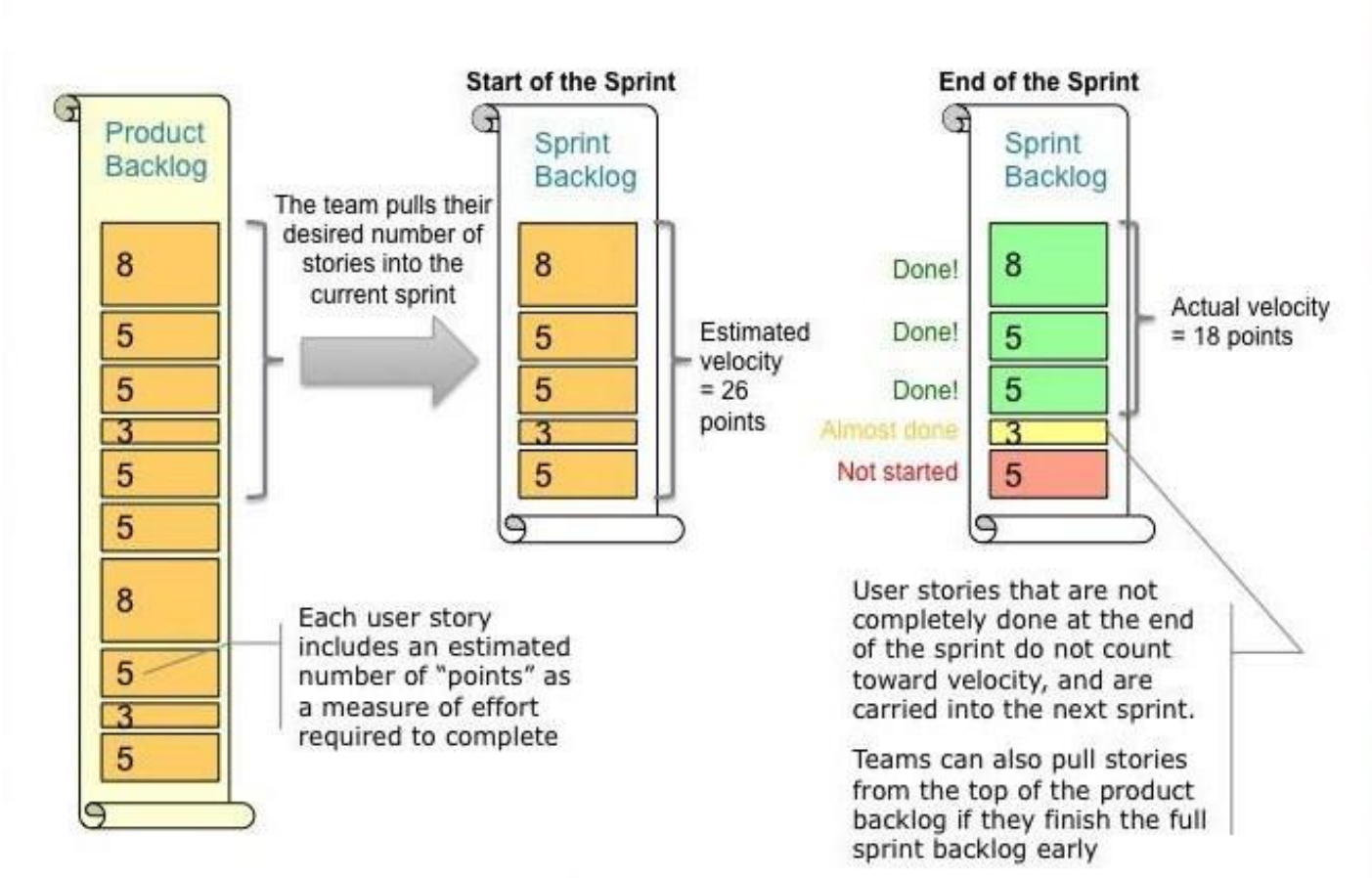
Funded by the European Union. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or Agency for Mobility and EU Programmes. Neither the European Union nor the granting authority can be held responsible for them.

Velocity: the key metric

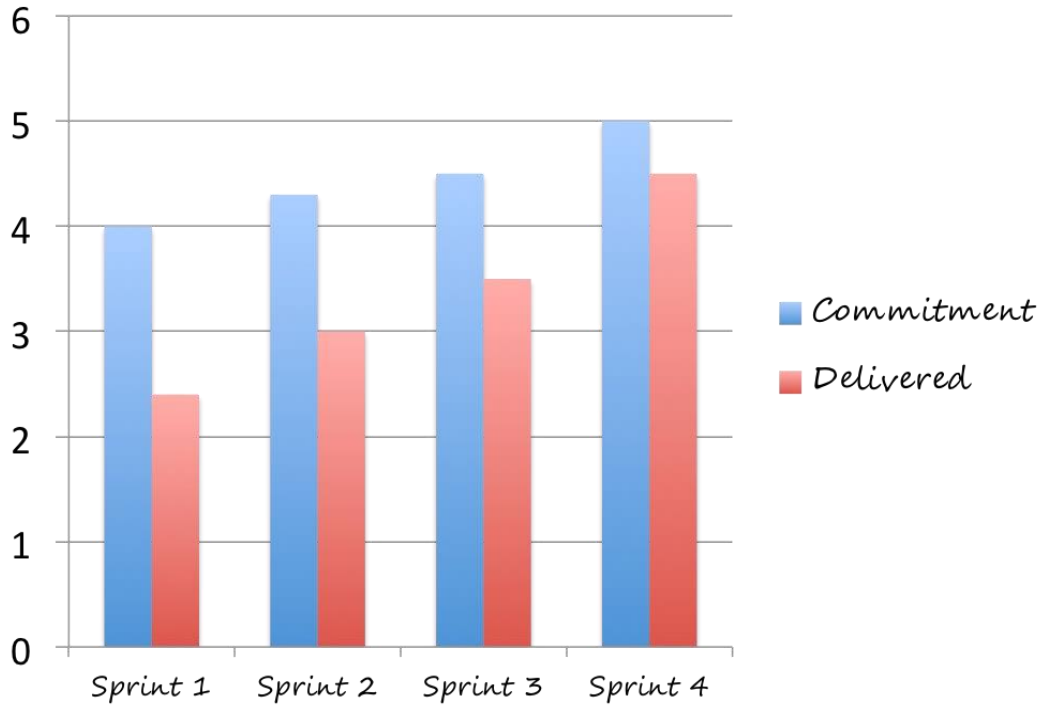
Velocity: measure of the amount of work a team manages to complete during a single cycle/sprint.

It is calculated for each cycle, summing up the effort (in hours or points) expended to complete the 'done' cards.

Velocity: the key metric



Velocity chart



Shows workflow speed and team workload per sprint.

Allows you to do forecasts for the future.

Velocity chart

Requirements:

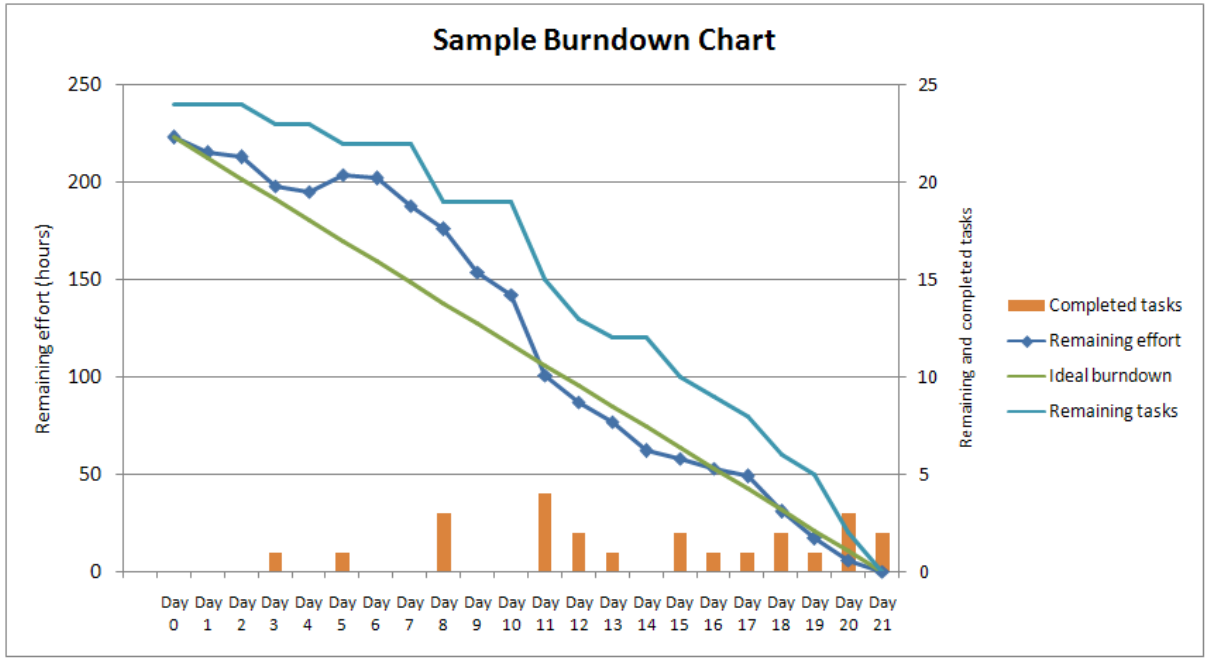
- Definition of work units (effort) for each task and duration of sprints.
- Continuous communication between team members on completed/to-be-completed tasks.
- At the end of each sprint, analysis of the number of completed work units.

Note: An actual velocity lower than the estimated one may denote:

- FF too high
- Optimistic estimates

Tip: to increase team velocity, start the retrospective from the 'blocked' cards.

Burndown chart



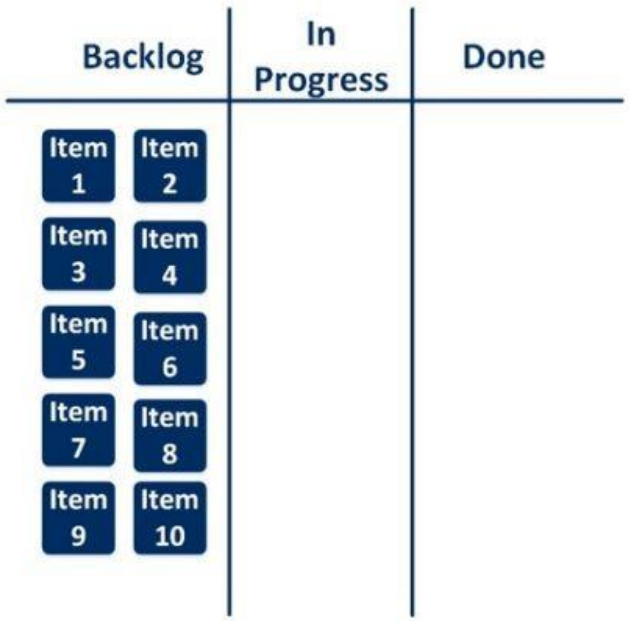
Graphical representation of the work required to complete a project.

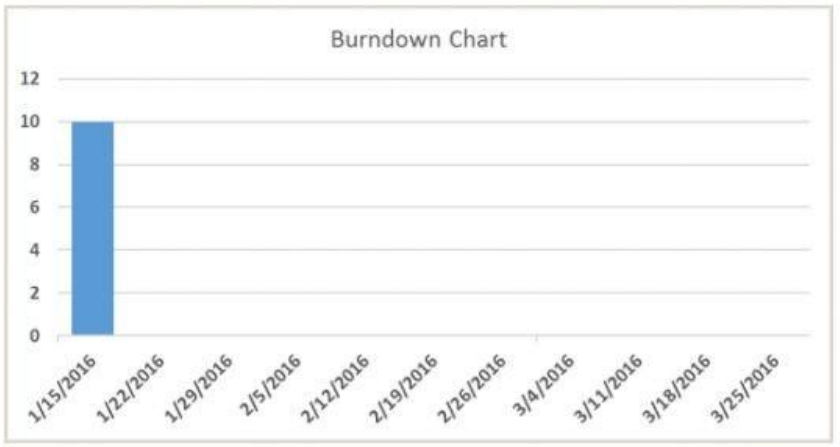
Usually the remaining work (or backlog) is indicated on the vertical axis and the time on the horizontal axis.

The diagram represents a time series of tasks to be completed.

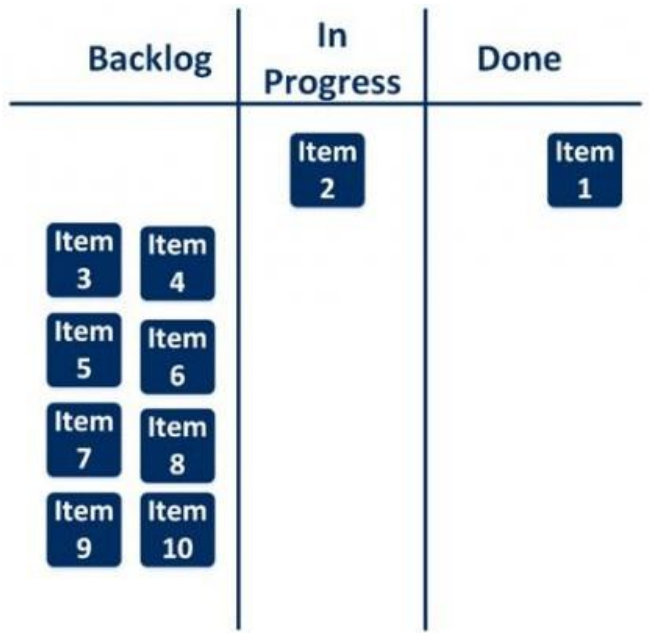
It is useful for predicting when the work will be completed.

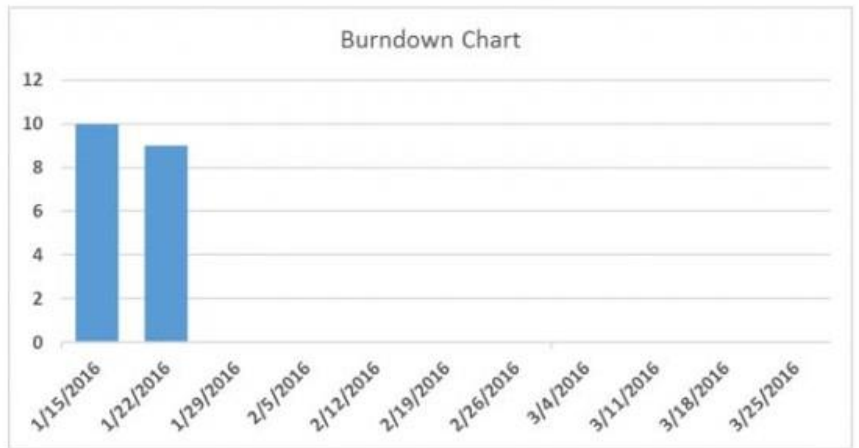
Perfect case: iteration 1



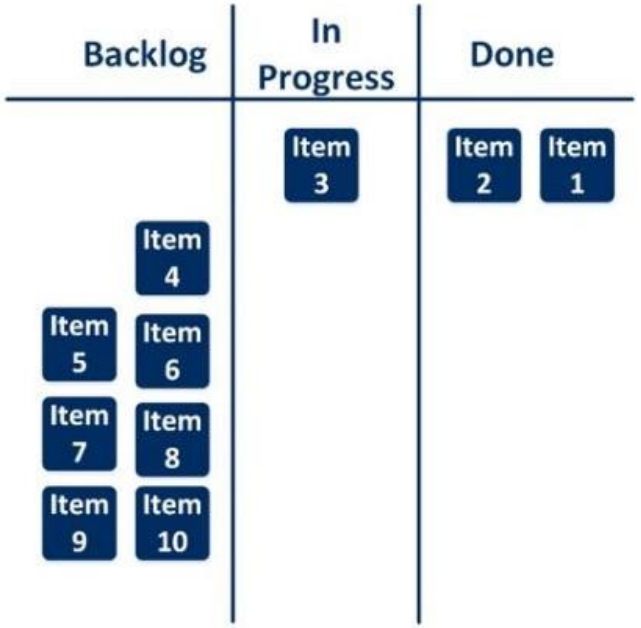


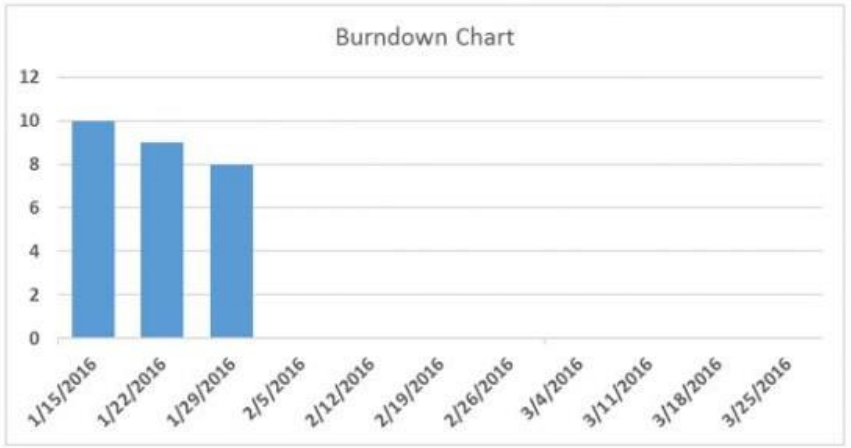
Perfect case: iteration 2



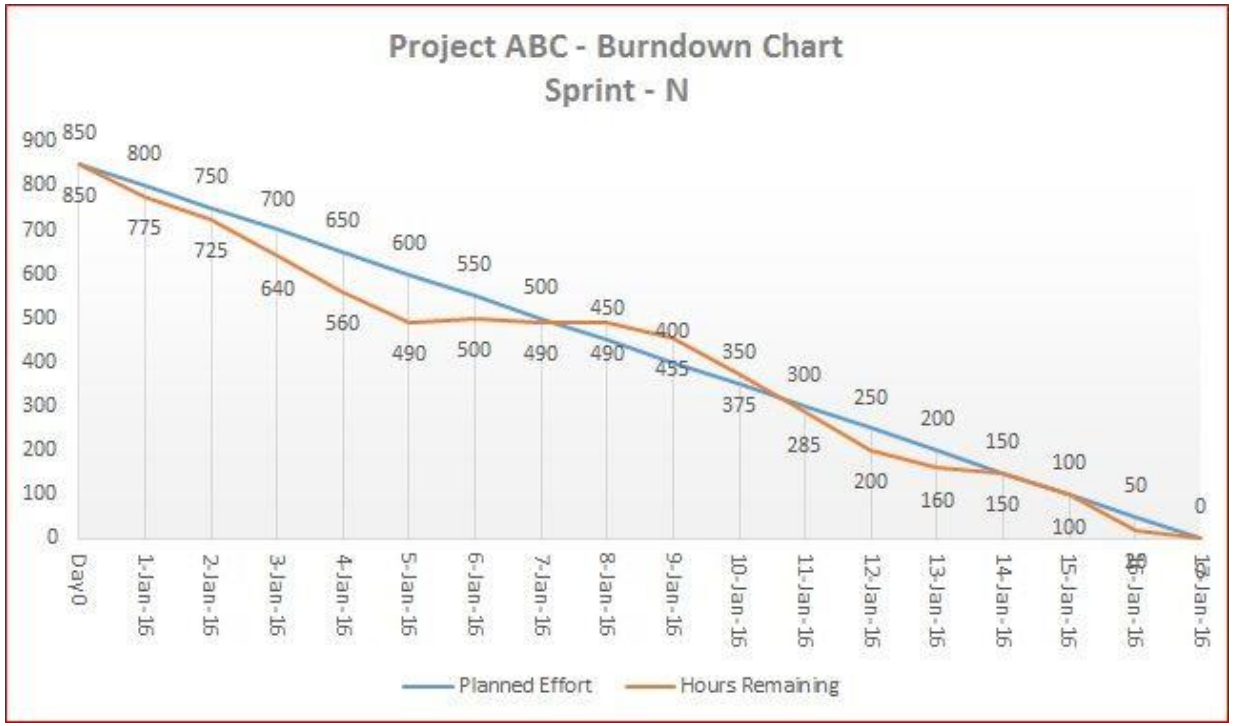


Perfect case: iteration 3

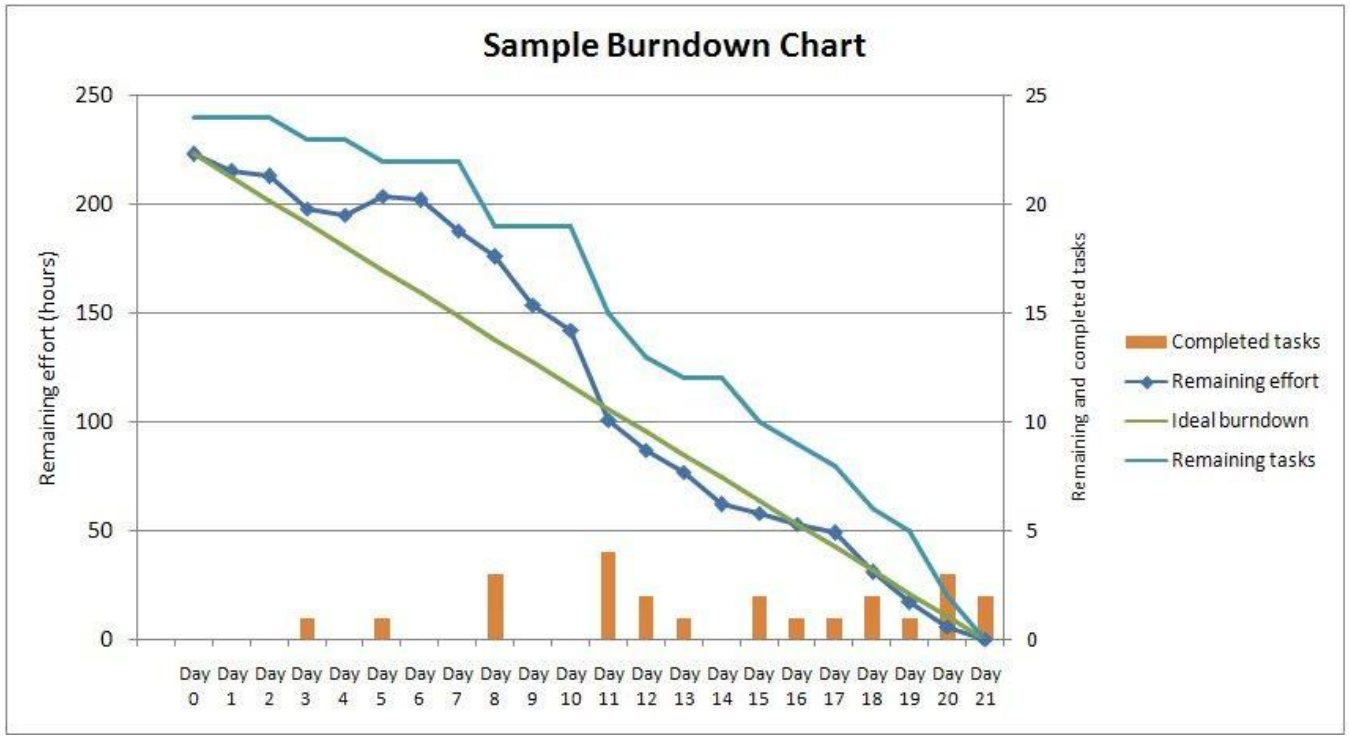




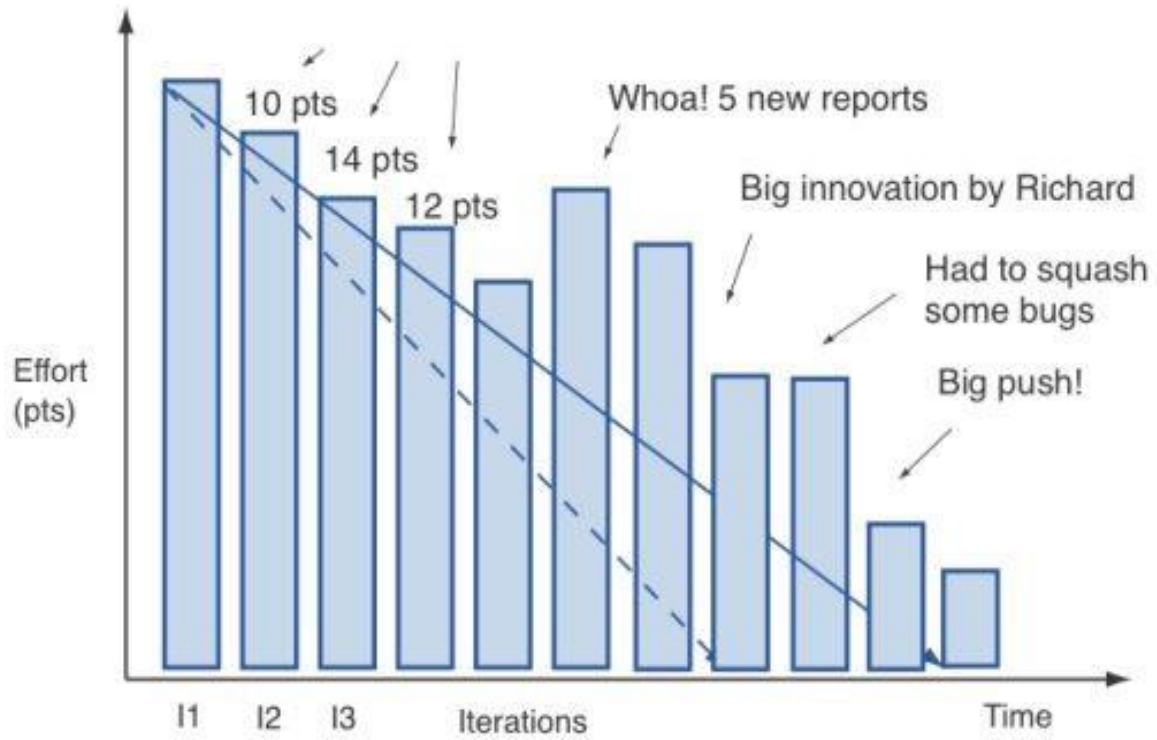
Burndown chart: example 1



Burndown chart: example 2

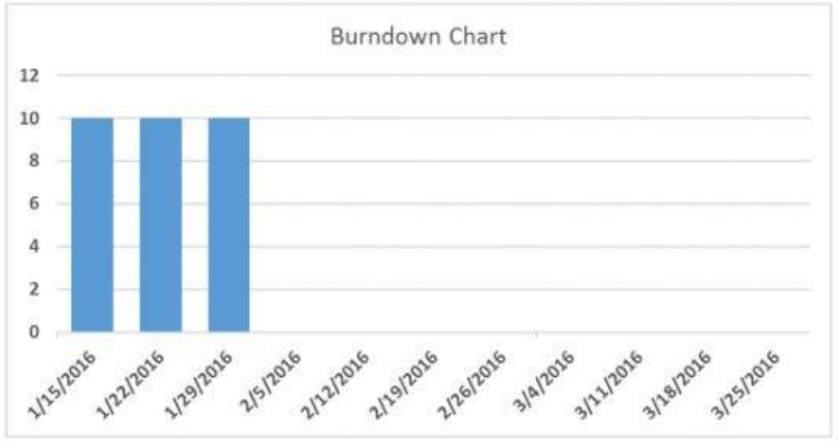


Effort may change over time

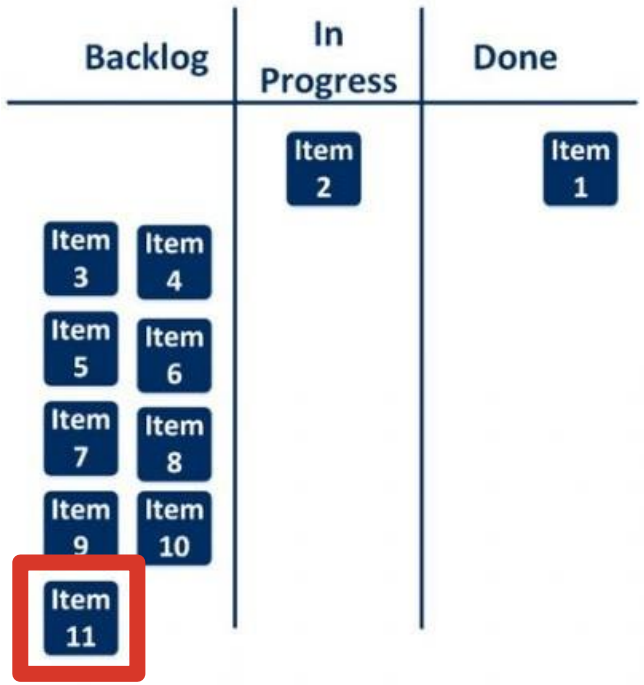


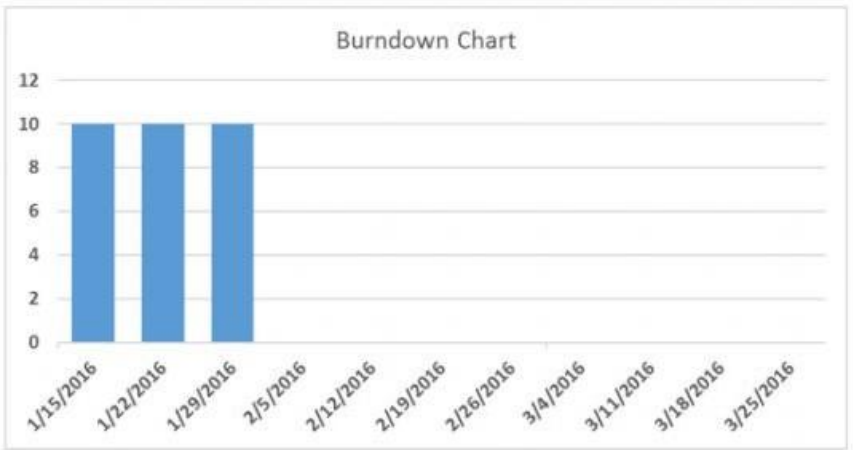
(Sad) reality: iteration 1

Backlog	In Progress	Done
Item 1, Item 2, Item 3, Item 4, Item 5, Item 6, Item 7, Item 8, Item 9, Item 10		

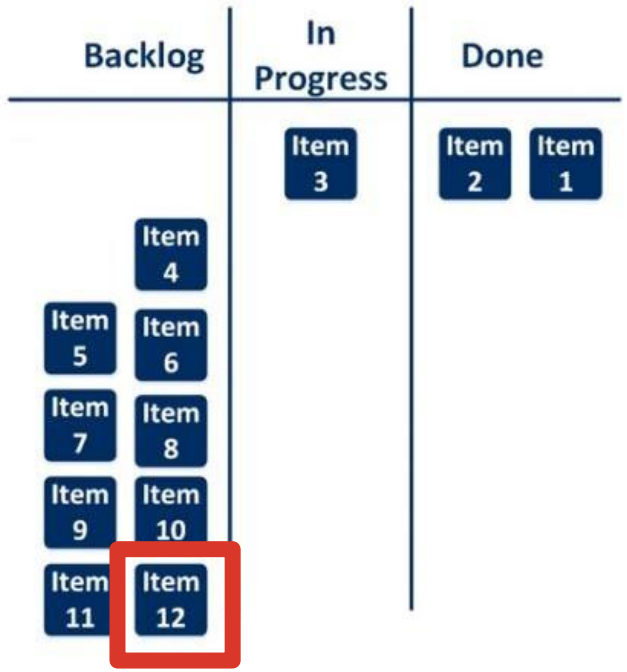


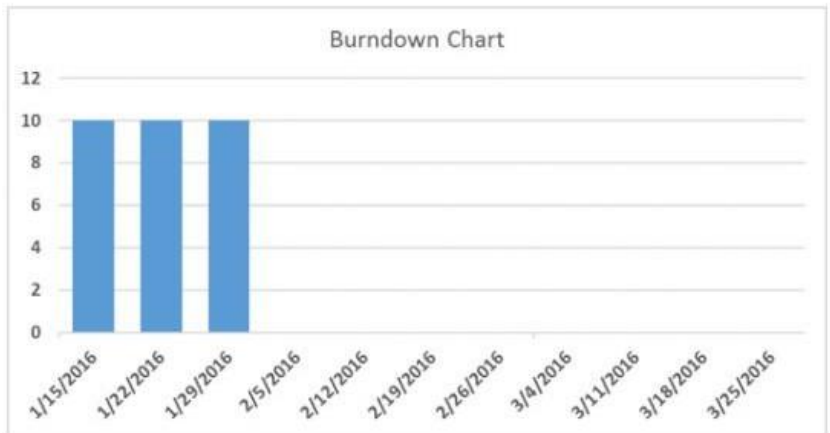
(Sad) reality: iteration 2



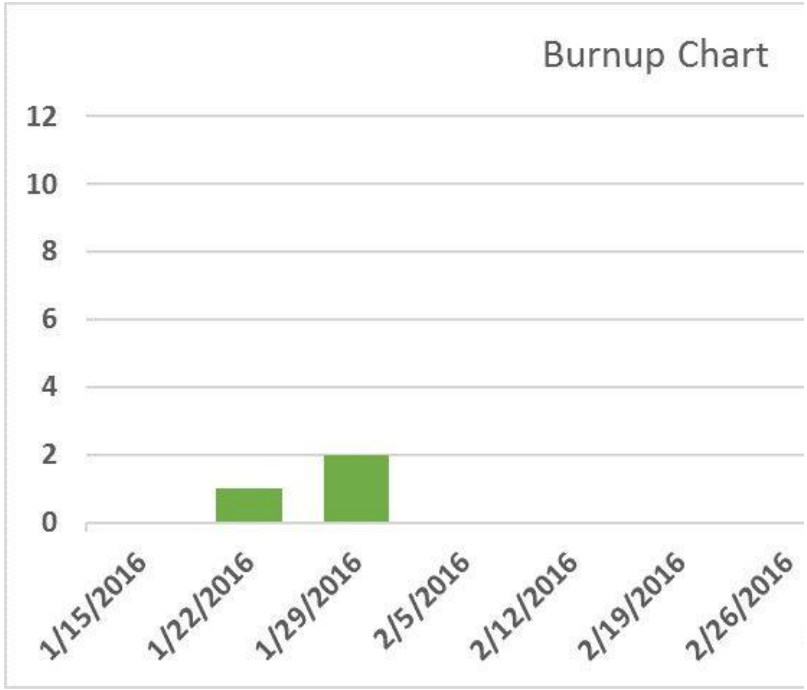


(Sad) reality: iteration 3





Burnup chart



Are there alternatives to the unedifying ‘flat’ burndown charts? The burnup charts.

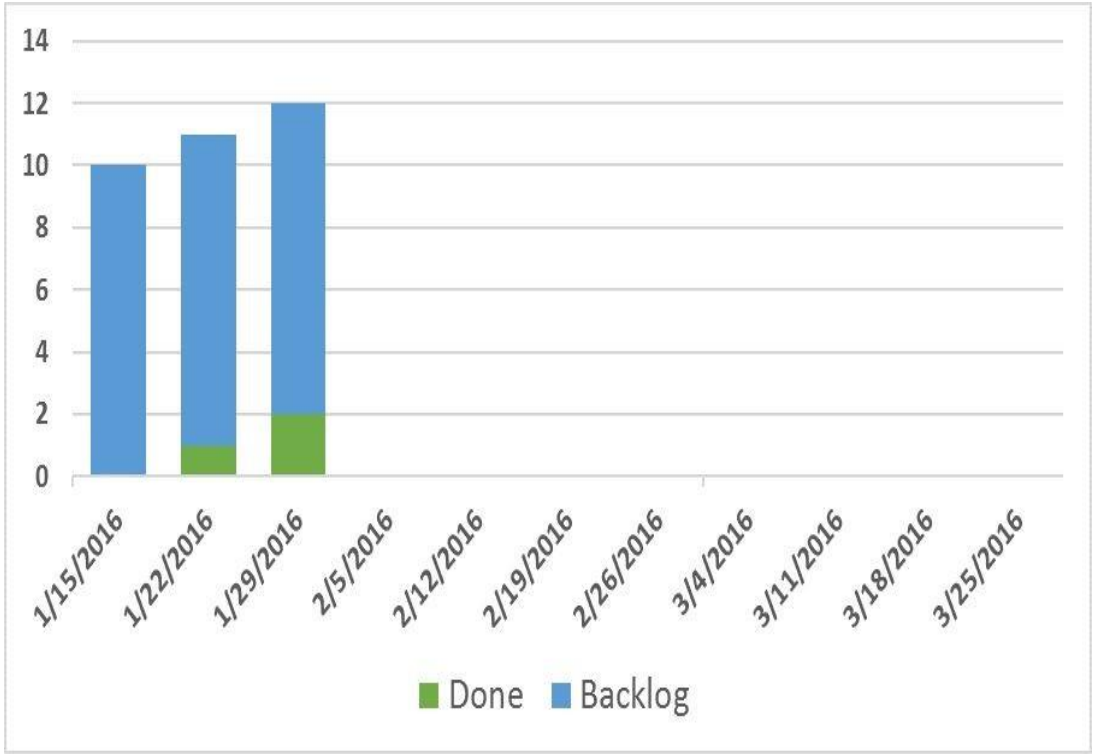
Burnup charts answer the opposite question: how much work has been completed?

Whereas a Burndown chart should always tend downwards, an ideal Burnup Chart moves upwards

But why don't we like burnup charts?

- They have no predictive utility
- They do not push velocity up (I do not know how much effort is left to finish the job, so I do not 'speed up')

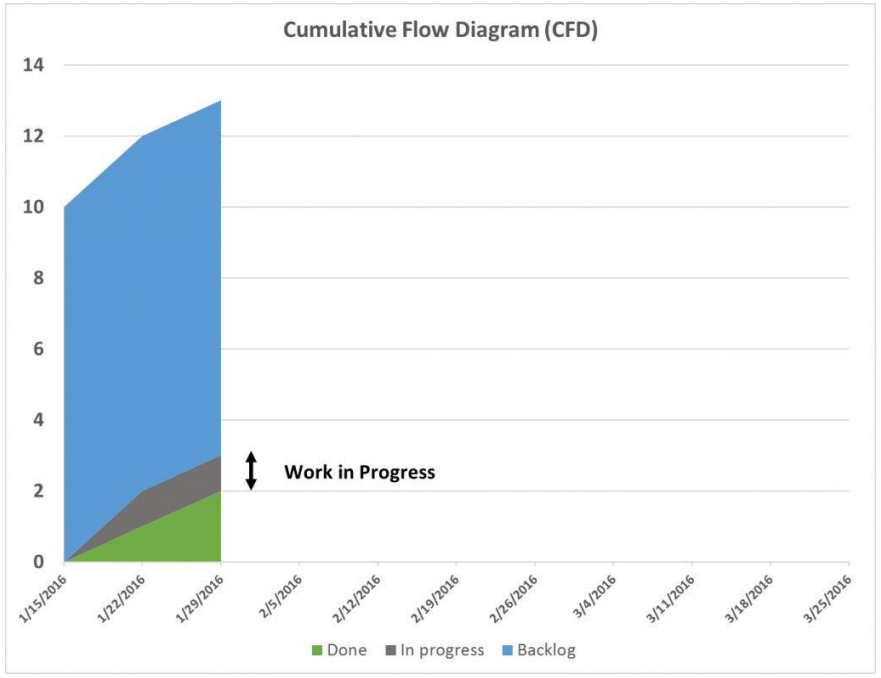
Cumulative Flow Diagram (CFD)

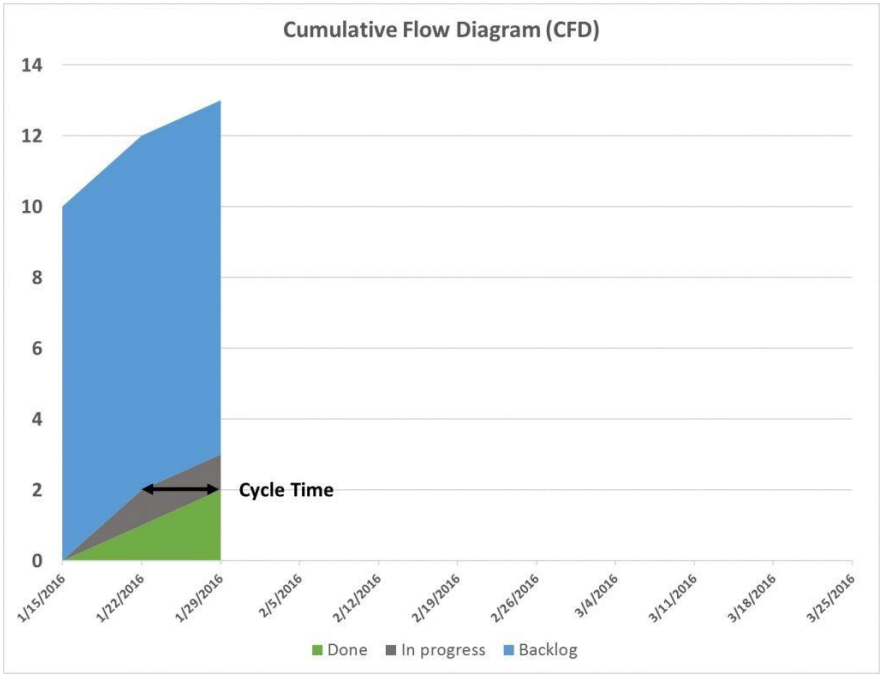


Advantages:

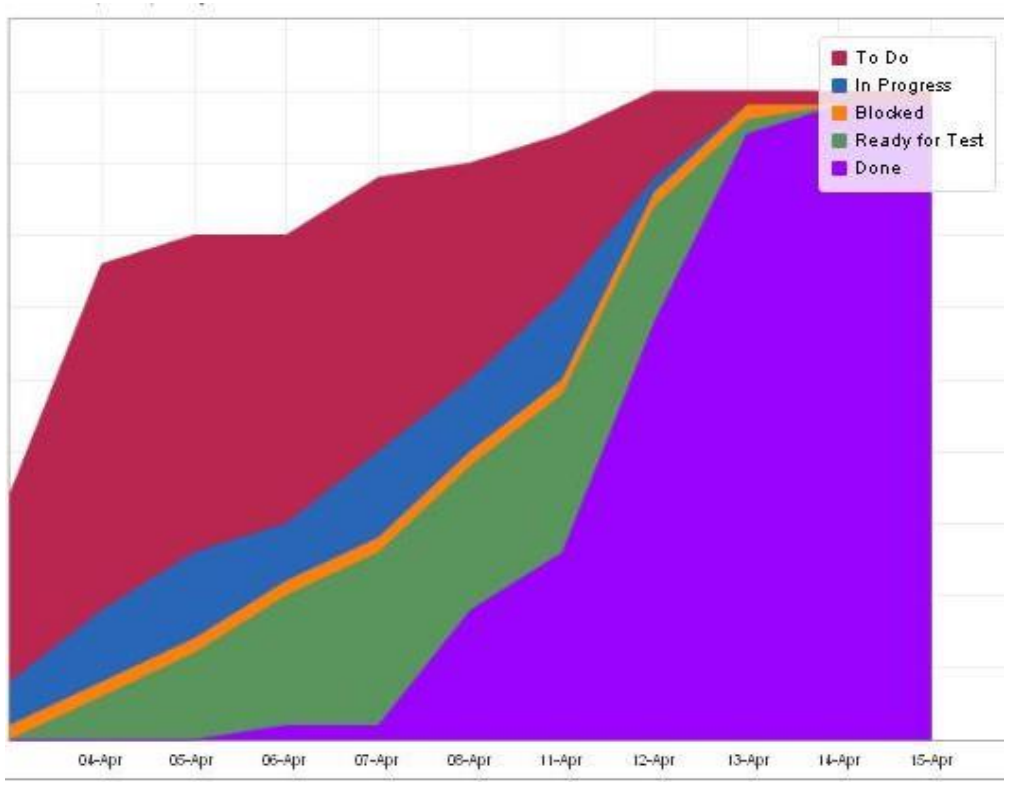
- Exceeds the limits of Burndown and Burnup
- Adds some extra data
- Displays data in a better way
- Promotes more robust decision-making processes

Cumulative Flow Diagram (CFD)



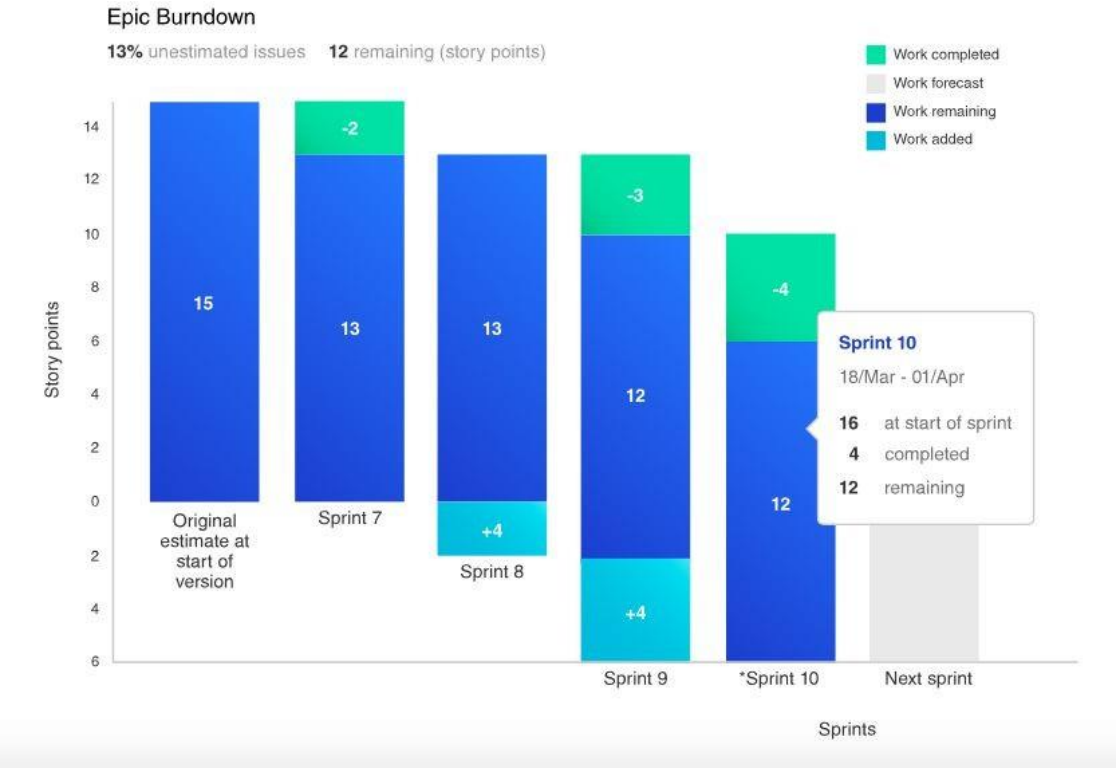


Cumulative Flow Diagram (CFD)



A CFD can have as many 'coloured bands' as there are states into which the process is divided (the columns of our board)

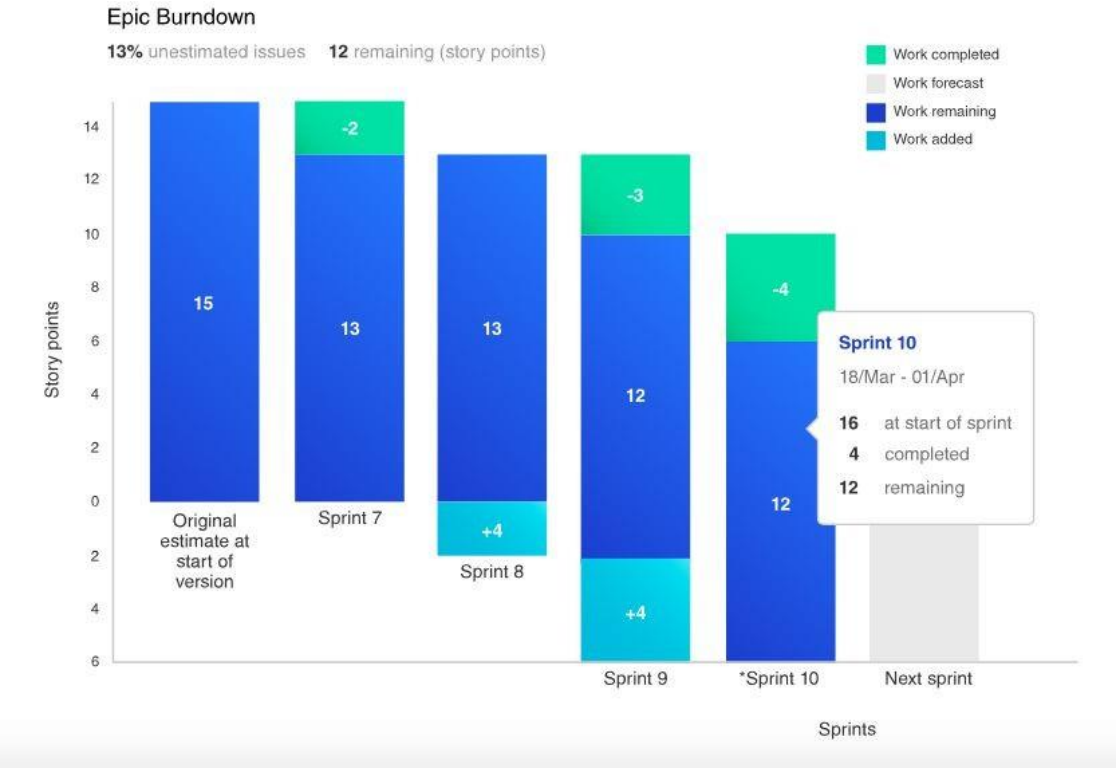
Epic and release burndown



Track project progress over different sprints, highlighting the flow of work and any rework within an epic or version (even spread over several sprints)

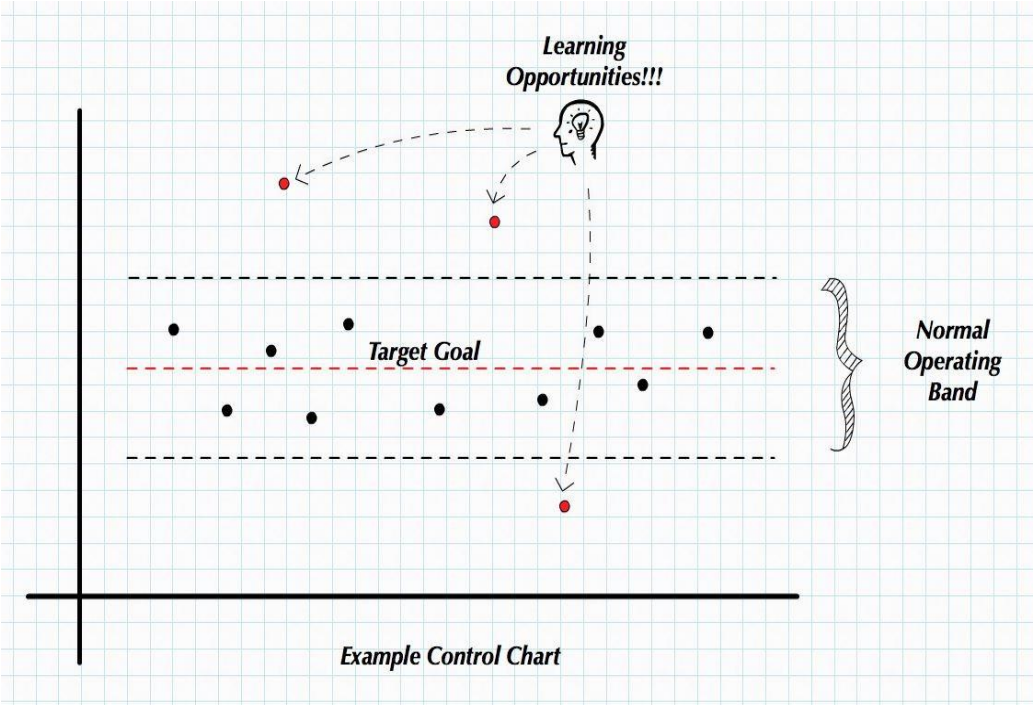
Scope creep: adding new requirements to an already defined project

Epic and release burndown



Note: If scope creep between several epic is a problem during the sprint, it is a natural consequence of Agile development (the client can decide to accept or not accept the work done based on what emerges during the project)

Control chart



It highlights the cycle time of the cards, i.e. the time between 'doing' and 'done'.

In an ideal situation, all cards/issues should be below the red line (average cycle time)

Control chart

In general:

- Short cycle time → High productivity.
- Constant cycle time → predictable delivery times.

The ideal situation is therefore to have teams with short cycle times and constant cycle times.

Other metrics

- Quality

  No. of reported defects (during the project, after release, from outside the team)

  No. of customers submitting support requests

  % coverage of tests carried out automatically

- Frequency, Speed and Difficulty of releases

Other metrics

Metrics are only one step in the process of building a team culture.

They provide evidence of the team's performance and measurable objectives for its members but it is important to be able to collect feedback during retrospectives to increase the team's confidence, the quality of products/services and speed of execution of activities.

Both quantitative and qualitative data are needed to drive change!

4.2 How to avoid failure (2.5 hours)

Co-funded by the European Union

shift4T

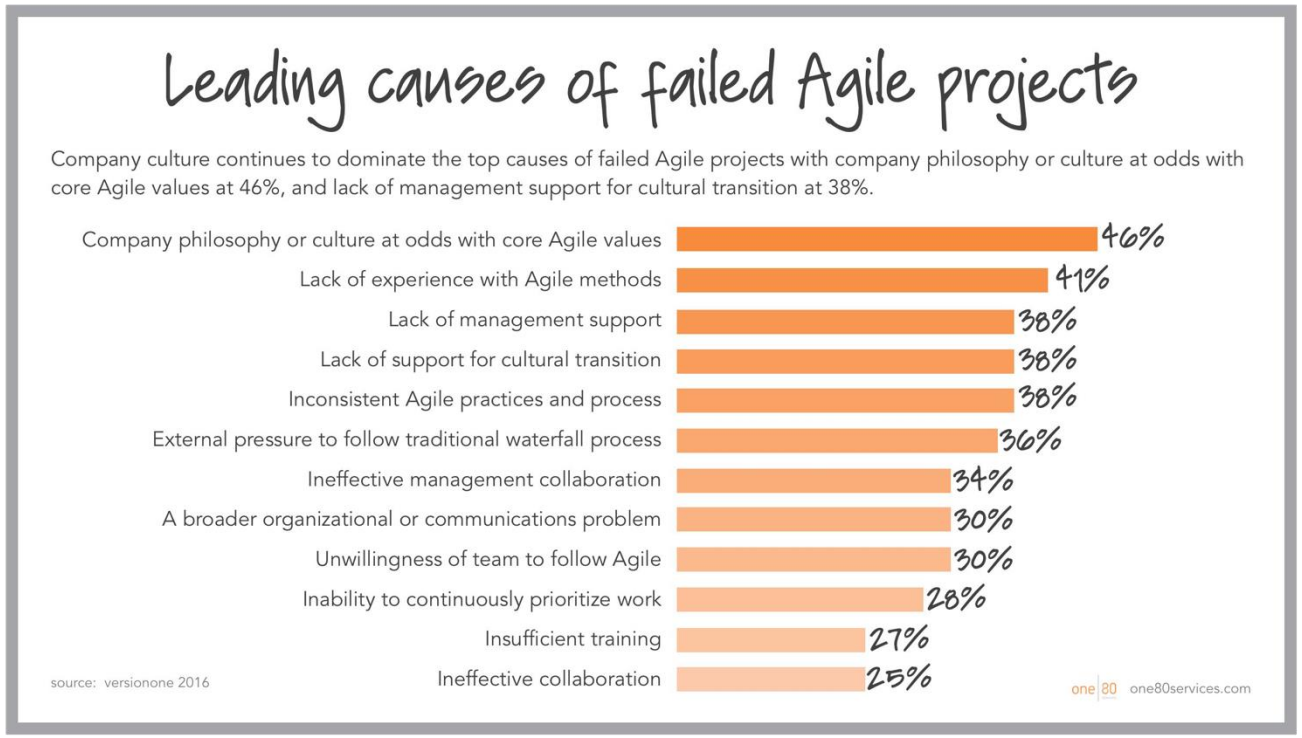
Funded by the European Union. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or Agency for Mobility and EU Programmes. Neither the European Union nor the granting authority can be held responsible for them.

How to avoid failure

Agile offers benefits ranging from more engaged employees to higher performance, but not so fast.

Why do Agile adoption projects often fail?

Typical causes of failure



A study conducted on more than 4,000 individuals highlighted the main causes of Agile failure.

Typical causes of failure

1. Contrasting organisational culture

Understanding that agile impacts organisational values and facilitating transformation is the first step towards wider adoption of agile

2. Lack of experience with the method

Investing in solid training in basic agile techniques and coaching for their correct application is money well spent

Typical causes of failure

3. Poor support in cultural transition

Tangible and active involvement of all management is crucial for cultural transformation

4. External pressure to Waterfall

Especially in large organisations, in order to facilitate coexistence between Agile and Waterfall teams, it is useful to involve people 'outside' Agile in meetings and to agree on mutual 'interfaces' for exchanging information

Typical causes of failure

5. Organisational and/or communicational issues

The effectiveness of Agile depends on a broad and deep acquisition of values and principles at organisational level

6. Team reluctance

Occurs when team members fear losing identity or control

7. Insufficient training

Ensure that all those involved (including leaders) receive solid training on Agile

A model for change management

Adapted from Knoster, T. (1991) Presentation in TASH Conference. Washington, D.C. Adapted by Knoster from Enterprise Group, Ltd.

Agile adoption vs Agile transformation

Agile adoption: use of tools and techniques in line with Agile principles.

Agile transformation: process of evolution of the culture and nature of the organisation.

Agile adoption vs Agile transformation

	ADOPTION	TRANSFORMATION
Answers the question	What do we do?	Who are we?
Emphasis on	Practices	Values and principles
Speed of change	Fast	Slow
Time horizon	Short period	Long Period
Impact on results (productivity)	100%	300%
Changes in organisational structure	Few or none	Significant
Changes in organisational culture	At most localised	Broad-spectrum

4.3 Reconciling Waterfall and Agile (2.5 hours)

Co-funded by the European Union

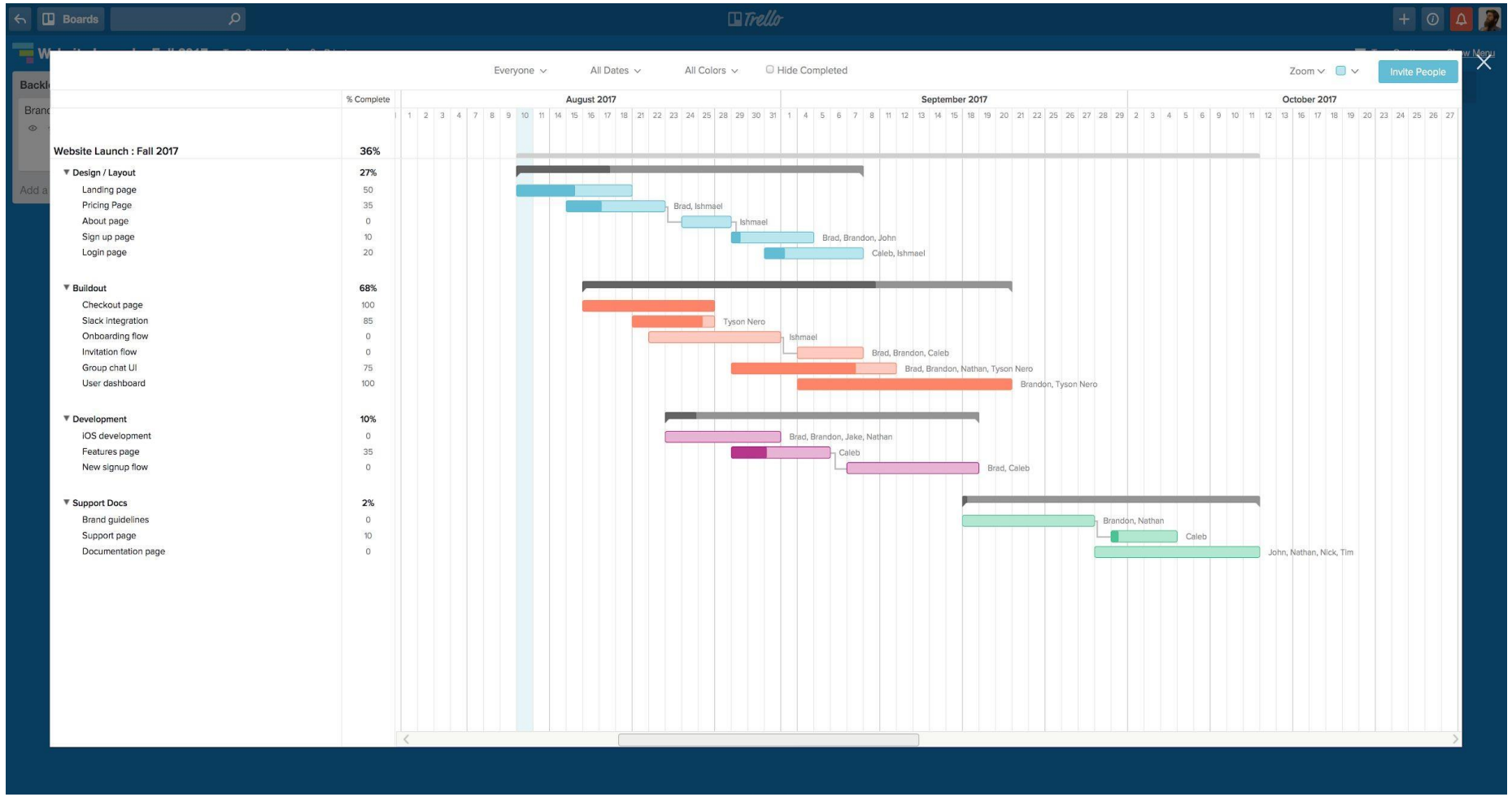
shift4T

Funded by the European Union. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or Agency for Mobility and EU Programmes. Neither the European Union nor the granting authority can be held responsible for them.

Creating the Gantt from cards

Regardless of the approach taken, we are often asked to produce a project Gantt. With respect to Agile, we can try to do this by starting with the very cards on the board.

A digital tool: TeamGantt

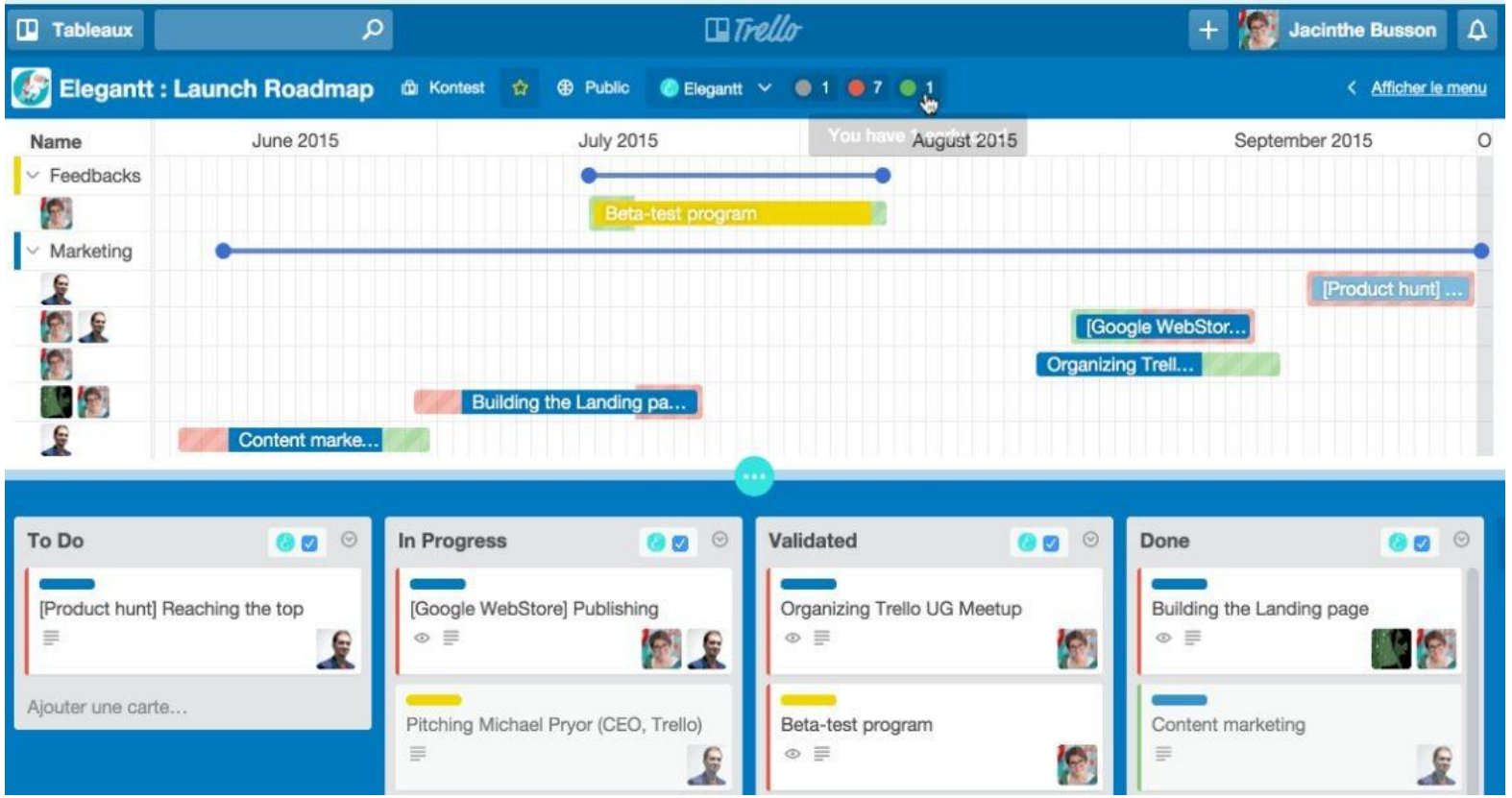


A digital tool: TeamGantt

TeamGantt is a digital tool that allows you to:

- create, share, update Gantt
- reorganise tasks
- display all projects in one view
- manage resources, control budget and make sure to avoid overload
- create reports
- check progress against plan and make projections
- list tasks and schedule them
- track activities

A digital tool: EleGantt

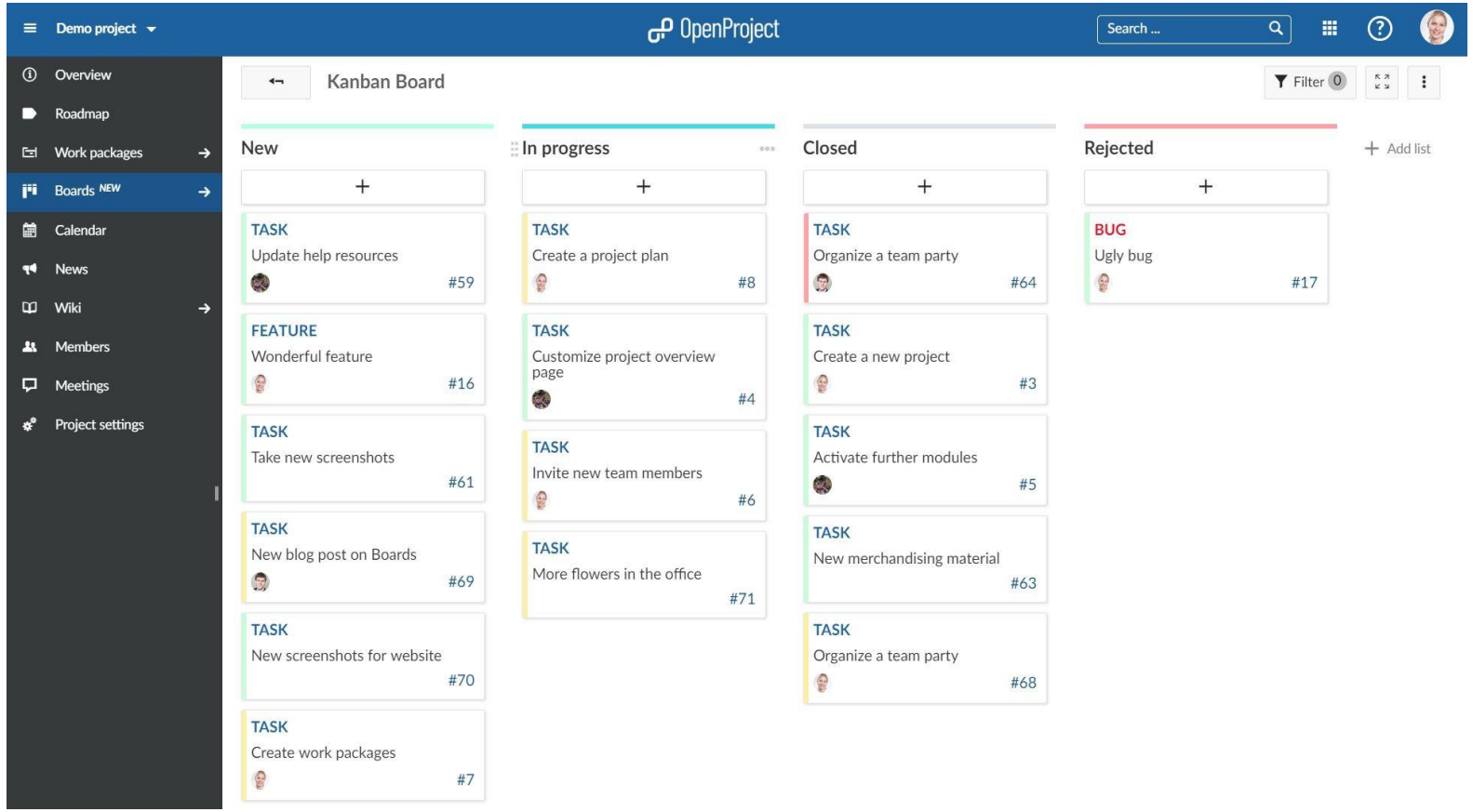


A digital tool: EleGantt

EleGantt is a digital tool that allows you to:

- display the Gantt directly in each board starting from the cards present
- automatically update the Gantt based on changes to cards, lists, assignees
- display dependencies between cards
- display deadlines and send automatic reminders to the team
- group, filter or reorder cards

A digital tool: OpenProject



A digital tool: OpenProject

OpenProject is a project management digital tool that allows you to:

- create product and sprint backlogs
- estimate activities in story points
- use card templates
- create burndown charts
- track 'impediments' for sprints
- assign and prioritise activities
- display activities on calendar or board

Monitor overall progress

How to monitor the progress of activities?

Define Sprint Goals (max 3 or 5) and ask the PM to indicate the % achievement of each one.

Sprint Goals: sub-goals corresponding to sets of cards (≈ Epic)

Monitor progress with sprint goals

Use Agile as a tool for the micro-management of the team (also in waterfall projects):

- Organise team work by weekly sprints and hold regular meetings (1 planning and 1 retrospective)!
- Keep the team's velocity under control
- Use goals as milestones and evaluate overall progress by % of their completion

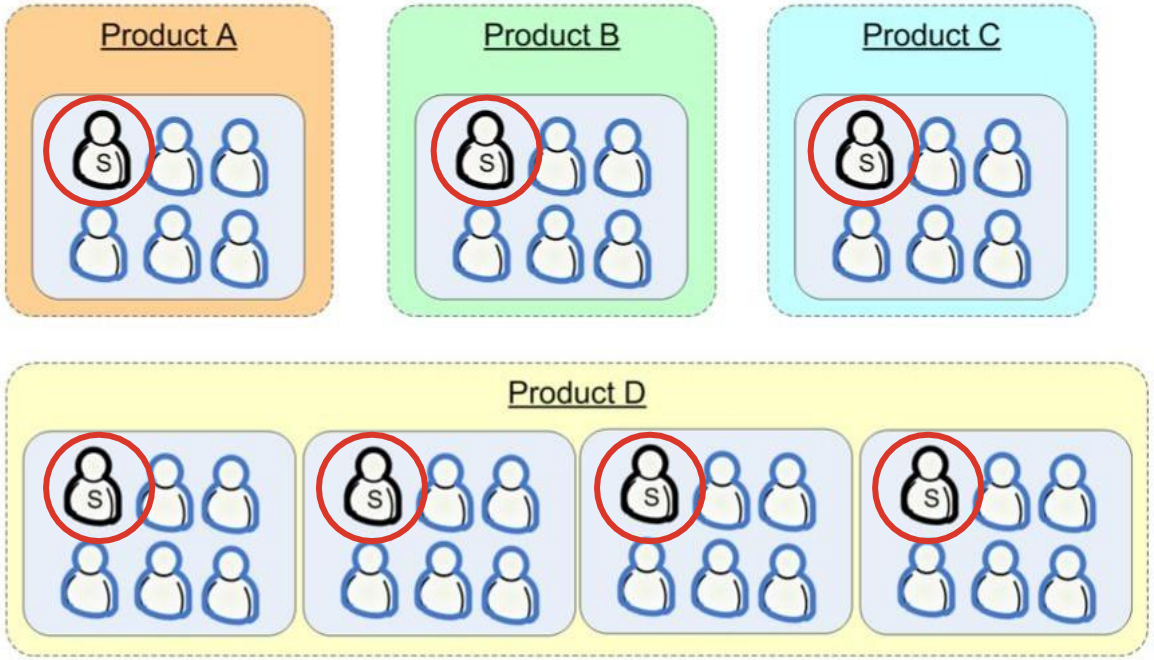
Scrum of scrum

How to apply Agile to projects involving many people?

Scrum of scrum (general meaning): technique for applying Agile to large organisations.

Scrum of scrum (specific meaning): meeting held regularly between the MS of each team to share information.

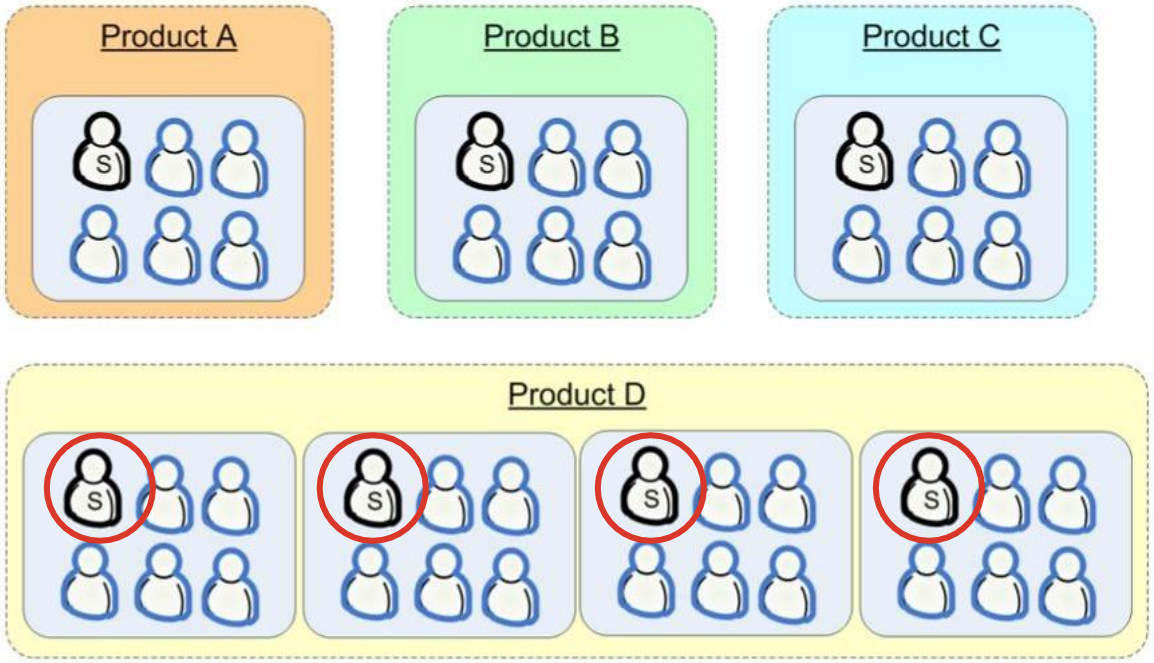
Scrum of scrum



Two levels different:

1. 'Corporate scrum of scrum' (all MS of all products)

Scrum of scrum



2. 'Product scrum' (all MS of product 'D')

Scrum of scrum

Product/project scrum: 30-minute meeting, once a week where:

1. Each MS describes what the team did last week, what it will do in the current week and what obstacles it foresees
2. MS share issues that affect all teams (e.g. related to integration)

Product/project scrum

30-minute meeting, once a week where:

1. Each MS describes what the team did last week, what it will do in the current week and what obstacles it foresees
2. MS share issues that affect all teams (e.g. related to integration)

Corporate scrum

15-30 minute plenary meeting, perhaps once a week where:

1. All team members receive news from the management (e.g. about upcoming events of a general nature)
2. "Round-robin': in turn, one member reports what their team did last week, what they will do in the present week, and what problems they foresee
3. Anyone else is free to add information or ask questions

Note: Intersperse the meetings, allowing those who must to participate in more than one meeting

Multiple Scrums

What happens when several teams work on the same project?

In general, the rule applies: more members = more complications.

That is why it is essential to understand

- how many teams to use
- how to allocate people between teams

Multiple Scrums

As a rule, it is preferable not to have teams larger than 9 persons in order to avoid:

- Long meeting (>15 min)
- People who don't know what others do
- Distractions
- Tight deadlines to resolve unforeseen issues

Multiple Scrums

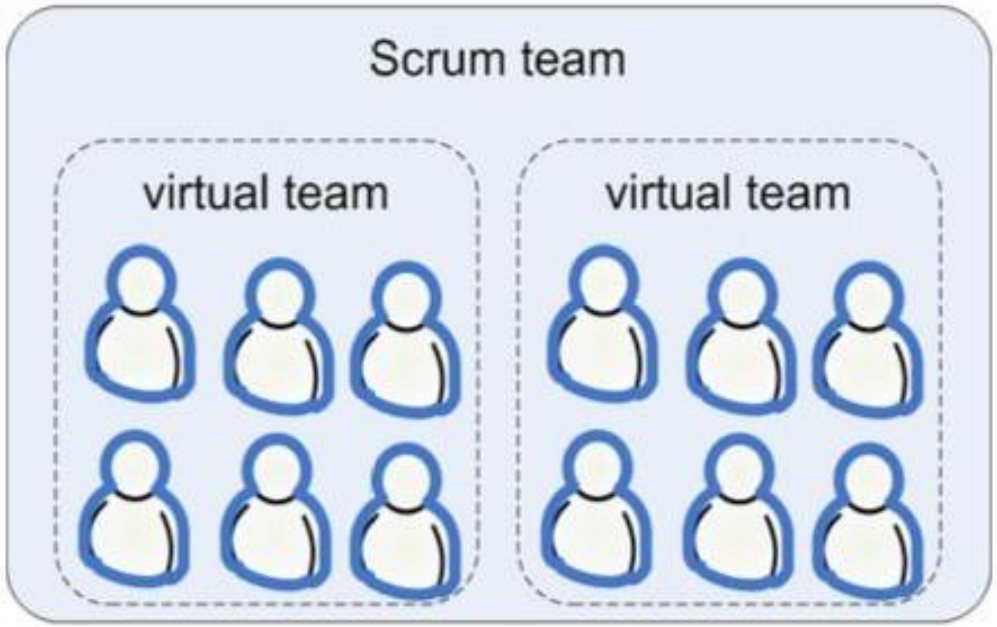
Is it enough to split up the larger teams?

The idea of splitting the team can only be effective if:

- The team has experience with Agile
- The roadmap can be subdivided (without compromising the output)

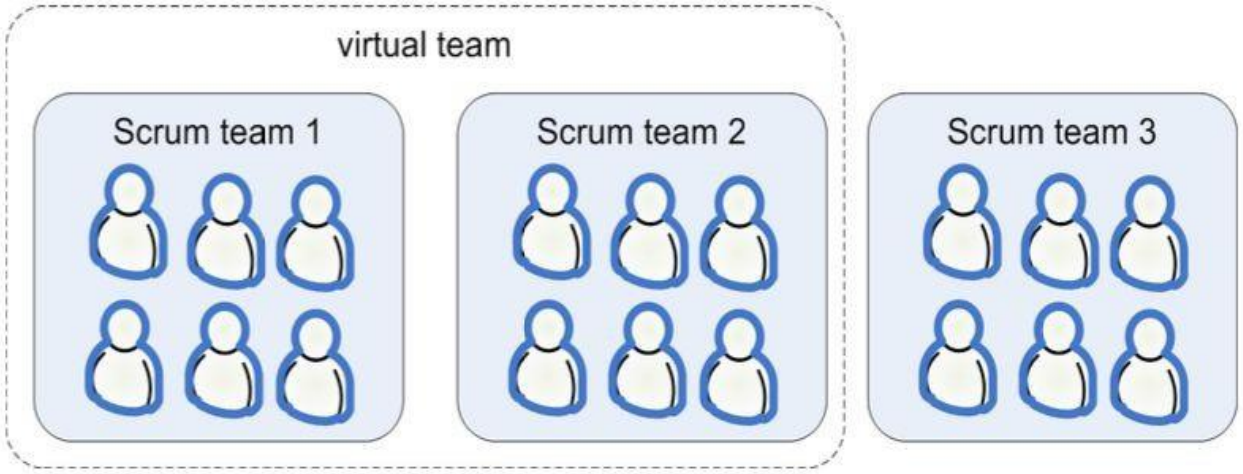
Beware of virtual teams! By observing whether and how members interact, one can see the existence of 'virtual teams'.

Virtual teams: case 1



You opt for one big team, but observing who talks to whom during the sprint, you notice a division into 2 sub-teams

Virtual teams: case 2

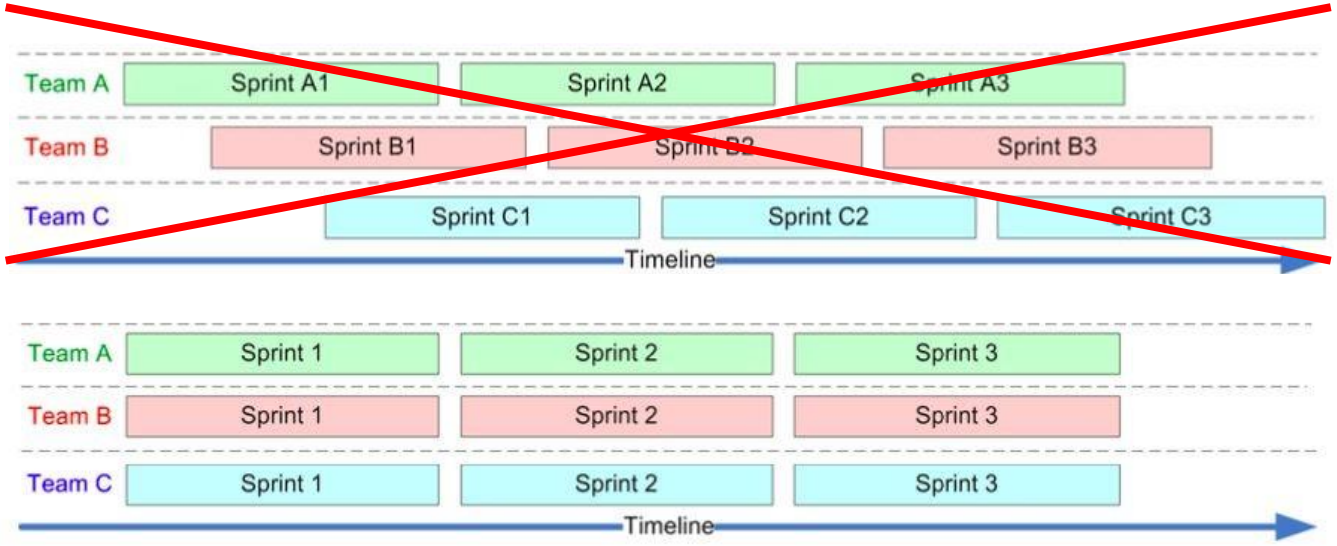


Three smaller teams are used, noting however that while the first and second interact frequently, the third remains isolated

Does this mean that the team size is wrong? whether virtual teams are permanent

- Yes, if they are permanent
- No, if they are temporary

Multiple Scrums



What to do instead if several teams are working on the same project?

1. Synchronising Sprints

Advantages:

- Shared meetings
- A single moment to redefine teams (between sprints)
- Fewer administrative tasks

Multiple Scrums

2. Appoint a Team Lead (Scrum Master of all teams)

The Team Lead decides on the composition of the teams and their changes between several sprints.

Note: before any change to the team, do not underestimate the so-called 'team gel' (bonds created by working in a team)

What about part-time members? In general, a few full-time people are better than a larger team of part-time members

Multiple Scrums

How to maintain the balance between planned and emerging activities?

One solution is to create 2 types of teams:

- Scrum Team
- Fire Team

Scrum Team: under the guidance of the PM carries out activities trying to prevent emergencies.

Fire Team: support team dedicated to resolving emergencies and protecting against other disruptive elements

Multiple Scrums

How to manage geographically distributed teams?

If the effectiveness of Agile depends a lot on the intensive and direct collaboration of the team it is necessary that this can also be done remotely, enabling members to:

- working at the same time
- participate in daily meetings
- have direct, spontaneous and immediate contacts
- have visibility of the board, the backlog and access to all useful information (e.g. diagrams)

Multiple Scrums

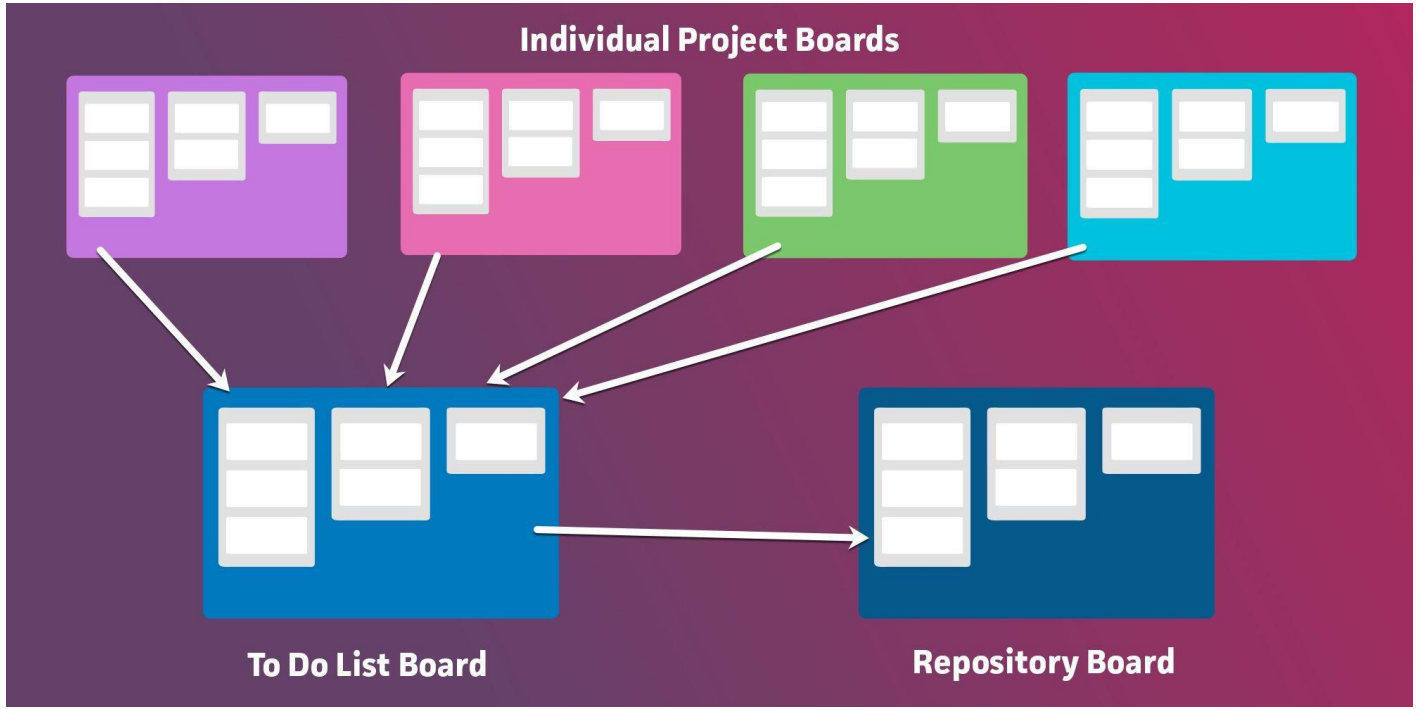
How to do it?

- Conference room with workstation(s) equipped with connection, webcam, microphone, collaboration software
- Remote windows: screens placed in each location showing other members and/or project progress data

If possible, it is preferable to have one dispersed team rather than several 'co-located' teams

Multiple Scrums

Which tools should be used to manage multiple Scrums?



Multiple Scrums

Which tools should be used to manage multiple Scrums?

- Create a 'To Do List' board where, at each sprint, you move (or copy) cards from your personal boards
- Create a 'board repository' to store project materials, resources (and cards) common to all members
- If useful, create automatisms for recurring card movements (e.g. with Zapier or Butler)

Scrum master checklist

At the beginning of each Sprint:

- After Sprint planning, create a Sprint information document (estimated speed, team size, duration, etc.), print it out and hang it on the team's most prominent wall.
- Send an e-mail to all stakeholders announcing that the new sprint has started (include the objective of the sprint and a link to the information document).
- Update the information document with the sprint statistics.

Scrum master checklist

Every day:

- Ensure that the Daily Scrum is started and finished on time.
- Ensure that cards are added/removed from the backlog only when necessary and that the PM is informed.
- Ensure that the sprint backlog and burndown are kept up to date by the team.
- Ensure that problems are resolved or reported to the PM.

Scrum master checklist

At the end of each Sprint:

- Make a demo of the Sprint open to everyone and announcing it 1-2 days in advance.
- Do a retrospective of the Sprint with the whole team and the PM.
- Update the Sprint statistics document. Add the actual speed and key points from the retrospective.