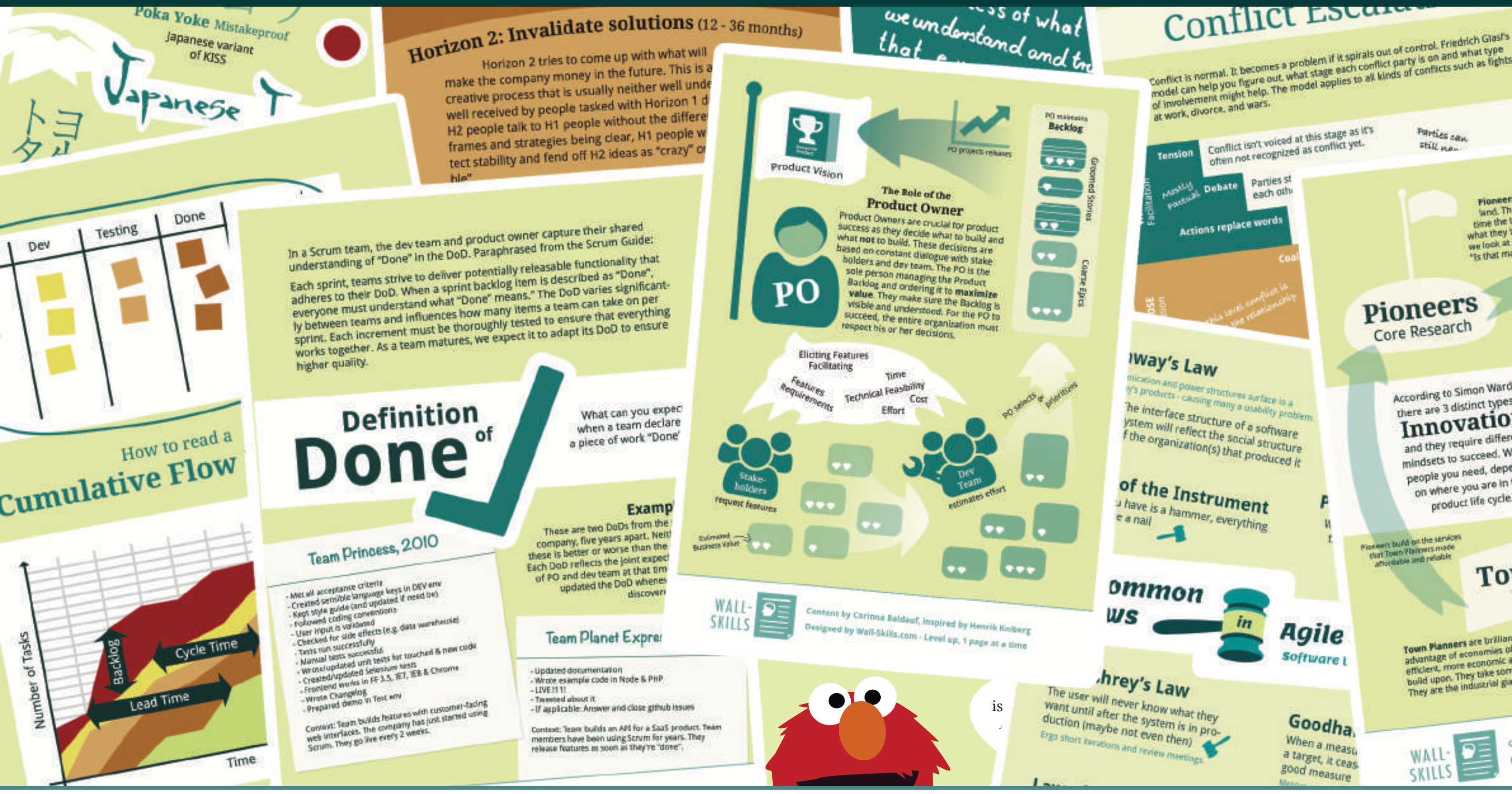
Skills for successful PRODUCT OWNERS

Quickly get an overview of Agile, Scrum, Product Mgmt & Facilitation with 1 topic per page



WALL-SKILLS

Corinna Baldauf

Welcome!

As a Product Owner you have chosen an exciting and challenging path! This book will make the trip smoother by pointing out important waypoints.

In base camp, you find what you need to know about Agile & Scrum itself, before we head out to cover topics from Product Management.

You'll be much more effective as a PO if you can help people talk to each other and collaborate. That's why we cover Facilitation next.

We end our journey on a mountain top looking at useful concepts from surrounding areas.

May you have an inspiring journey!
All the best, Corinna Baldauf

Role of the PO

Product Owners have a very important role: They maximize product value for the customers and the business. This requires many trade-offs, clear communication and an unwavering vision for the product. Sounds like a tall order? It is, but you can master it! This book is here to help you hit the ground running.

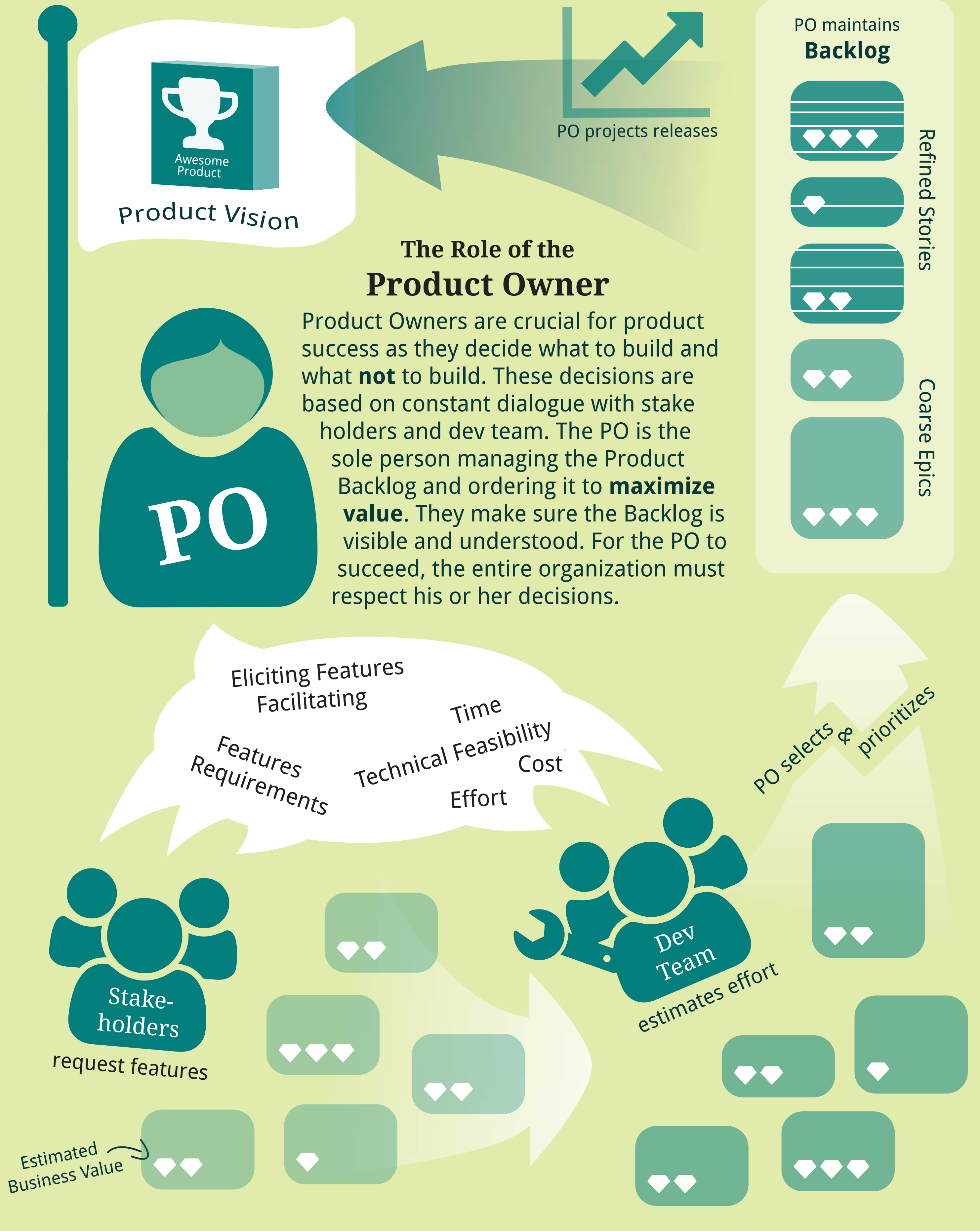
More Information

- Henrik Kniberg's excellent video "Agile Product Ownership in a Nutshell": https://www.youtube.com/watch?v=502ILHjX9EE
- The official Scrum Guide: http://www.scrumguides.org/scrum-guide.html#team-po

The Role of the

Product Owner

Product Owners are crucial for product success as they decide what to build and what **not** to build. These decisions are based on constant dialogue with stake holders and dev team. The PO is the sole person managing the Product Backlog and ordering it to **maximize value**. They make sure the Backlog is visible and understood. For the PO to succeed, the entire organization must respect his or her decisions.



Agile & Scrum

Many companies try to adopt working in an Agile way.
The Scrum framework is the most popular way to do so.

More Information

"Kanban and Scrum - Making the most of both"
by Henrik Kniberg & Mattias Skarin is by far my favourite book for beginners. A nice, short read and actionable.
http://www.infoq.com/minibooks/kanban-scrum-minibook

Content

Sprint Goal

A Sprint Goal allows the dev team to make good tradeoffs even if they can’t check back with the PO.

More Information

- Roman Pichler created a Sprint Goal Template: http://www.romanpichler.com/tools/sprint-goal-template/

Sprint Goal

Why are you doing this iteration?

What is a sprint goal?

Each sprint should have an explicit sprint goal, e.g. written out across the Scrum board. Think of the goal as the elevator pitch of a sprint. During planning a Scrum team commits to the sprint goal, not a bunch of stories.

Why set one?

The goal helps the team focus and make good trade-offs, even when the product owner is not available to help. A team might reach the sprint goal even if they can't finish all stories.

Who sets the goal?

The product owner can choose the goal beforehand to help her order the backlog according to the goal. Alternatively the whole team can phrase the goal at the end of planning, when they know which stories will be part of the sprint.

Examples of good and bad goals

- Bad: "Complete all 4 stories we pulled"
  Doesn't add information to enable good team decisions
- Bad: "Improve the dubbledigook widget"
  Completely fuzzy. How can you to tell when / if you're done?
- Better (i.e. enabling and measurable):
  "Customers can pay with bank transfer"
  "Time to first byte on the landing page is less than 0.7s"
  "Increase conversion rate by 10%".

WALL-SKILLS
Content by Corinna Baldauf - http://finding-marbles.com
Designed by Wall-Skills.com - Spread Knowledge with 1-Pagers

Definition of Done

What does it mean when something is “Done”? Especially in software development people’s expectations vary significantly when they hear “Done”. It could mean anything from “finished coding” to “written and tested” to “documented, released and blogged about it”. In Scrum, a dev team and their product owner compare expectations and agree on a shared Definition of Done.

More Information

The DoD is part of the Scrum Guide:
http://scrumguides.org/scrum-guide.html

In a Scrum team, the dev team and product owner capture their shared understanding of "Done" in the DoD. Paraphrased from the Scrum Guide:

Each sprint, teams strive to deliver potentially releasable functionality that adheres to their DoD. When a sprint backlog item is described as "Done", everyone must understand what "Done" means." The DoD varies significantly between teams and influences how many items a team can take on per sprint. Each increment must be thoroughly tested to ensure that everything works together. As a team matures, we expect it to adapt its DoD to ensure higher quality.

Definition of Done

What can you expect when a team declares a piece of work "Done"?

Examples

These are two DoDs from the same company, five years apart. Neither of these is better or worse than the other. Each DoD reflects the joint expectations of PO and dev team at that time. They updated the DoD whenever they discovered gaps.

Team Princess, 2010

- Met all acceptance criteria
- Created sensible language keys in DEV env
- Kept style guide (and updated if need be)
- Followed coding conventions
- User input is validated
- Checked for side effects (e.g. data warehouse)
- Tests run successfully
- Manual tests successful
- Wrote/updated unit tests for touched & new code
- Created/updated Selenium tests
- Frontend works in FF 3.5, IE7, IE8 & Chrome
- Wrote Changelog
- Prepared demo in Test env

Context: Team builds features with customer-facing web interfaces. The company has just started using Scrum. They go live every 2 weeks.

Team Planet Express, 2015

- Updated documentation
- Wrote example code in Node & PHP
- LIVE !11!
- Tweeted about it
- If applicable: Answer and close github issues

Context: Team builds an API for a SaaS product. Team members have been using Scrum for years. They release features as soon as they're "done".

WALL-SKILLS

Content by Corinna Baldauf - http://finding-marbles.com

Designed by Wall-Skills.com - Spread Knowledge with 1-Pagers

Product Management

The skills in this chapter are specific for the PO role. They cover agile concepts, such as User Stories as well as ideas from traditional Product Management and User Experience.

Content

Saying “No”

Saying “No” is an essential skill for Product Owners. You get bombarded with feature requests and ideas. And most of them sound good enough. But having ideas is easy and – mostly – free. Implementing ideas is not. They cost time and money.

And even if they didn’t, if you just put every possible feature into your product it will become bloated, confusing and unusable. What you choose not to include is at least as important as what you do implement.

It’s your job to have a strong vision of what problem the product solves and for whom. And then execute on that, protecting the backlog from requests that do not serve that vision.

How does this idea fit into our long term strategy?

As a PO, you should be very clear about the product strategy. New requests are viewed as not supporting the strategy unless the requester convinces you that their idea will help reaching significant strategic goals faster. Tip: Try "Impact Mapping"

You need to convince the other stakeholders.

When you put an item into the backlog at a certain priority it pushes down other items. Items whose stakeholders ask you to justify that decision.Don't put yourself into that position. Let the requester go into the line of fire. Tip: Try "Buy-a-feature"

A Product Owners most important word: 7 Ways to say "No" as a Product Owner – using valid reasons instead of the word

"No."

This is going to cost 10x as much as you'd want it to.

Random ideas congest the path to maximum value. Stakeholders often have little understanding of how much work a "small favor" is. Always ask the team for estimates before committing to anything.

"Which of your requests should I scrap to put the new one in?"

We have limited WIP.

People are good at coming up with ideas. People are bad at understanding that limiting Work In Progress is key to getting things done and delivering a valuable product. Avoid becoming a "request manager" tracking endless lists of things that have no chance of ever getting done. Tip: Try a Visual Dashboard with WIP limit 3. Requests are only accepted once something got done.

Our backlog is a prioritized list.

Don't waste time figuring out whether something is worth doing, delegate that to the requester. You are a bottleneck. Don't be pressured into making something Priority 1 just because it's the Group CEO's idea. Arrange your backlog items so that they make sense from an economic perspective. Tip: Google "Weighted shortest job first"

"Do you have any data that warrants putting it into a position where it has a chance to be done in the next half year?"

This has no chance of being done anytime soon.

Why would you track things so far into the future that your product won't fit to the idea any more should you get there? If the idea is good, it will come back later. If it isn't - well, why is it important to keep? Any backlog item moving beyond a predictable horizon wastes your capacity if you think about it at all. Tip: Cap the backlog to contain at most 6 months of work

We discuss things when they become important.

Here's a polite way of saying: "This item is wasting my time". Focus on the top few priorities in the backlog. Avoid getting dragged into meetings for things that may never happen. If others want to discuss these things, let them. Once they can prove it's worth your time, you join. Until then, your product is fine without the requested item.

"We will consider your suggestion at an appropriate time. Thanks."

User Stories

In agile environments you reduce big upfront design, because you know that customer requirements will change. Capturing requirements as User Stories is the most common approach. User stories are short and easy to use.

Jeff Patton says they are called stories because we're meant to have a conversation – tell a story – about them.

In agile environments you reduce big upfront design, because you know that customer requirements will change. Capturing requirements as User Stories is the most common approach.

A User Story has 3 components (3Cs): The physical **C**ard contains title and description. It's a reminder to have a **C**onversation to create a shared understanding between everyone involved. You discuss what is really needed and agree on acceptance criteria to **C**onfirm that the team achieved the benefit.

User Stories

Capture requirements as User Stories - Concise and easy to use!

Classic format

As a <role>
I want <goal>
so that <benefit>

Focus on benefit

In order to <benefit>
as a <role>
I want <goal>

A User Story contains:

- The **Role** - **Who** wants something. Ideally a customer (persona).
- The **Goal** - **What** does the user want to do?
- The **Benefit** - **Why** the customer wants this. What does she want to achieve? This is the most important part of the story, because you may discover that there's a better way to reach the benefit.

Example stories for a CMS:

- Bad: "As the database I want a field 'status' so that I can store if a post is a draft or published"
  Databases can't want anything. What's the benefit for customers?
- Good: "As an author I want to save posts as drafts so that I can finish them later"

(Mini) Design Studio

A Design Studio is a 1 day workshop invented in industrial design. The 1-pager describes a shortened version tailored to design single elements such as a graphical interface, short text, etc. It's useful for product teams that jointly want to shape their product's face.

Solutions created during Design Studios tend to have broad support because many people were involved.

A Design Studio is a 1 day workshop invented in industrial design. Here we present a shortened version tailored to design single elements such as a graphical interface, short text, etc. Solutions created during Design Studios tend to have broad support because many were involved. It's useful for product teams that jointly want to shape their product's face.

Mini Design Studio

Quickly gather interface ideas from many people and evolve them into a sketch of the solution

How to run a design studio in less than 1 hour:

1) Invite a diverse group of people. Invite at least 1 UX person, if possible

2) Hand out stacks of paper and thick markers

3) Present the problem to be solved

4) Round #1 - Set a timer to 5 minutes. Now everyone individually sketches as many solutions as they can think of. The short time prevents people from adding unnecessary details
When time is up, everybody posts their interfaces to the wall and describes them in less than 2 minutes.
Tell everyone to combine whichever elements they liked in all presented interfaces.

5) Round #2 - Another 5 minutes to sketch a refined solution. Another round of presentations.

6) Have a 5 minute discussion on which interfaces work for you and why. You can still incorporate suggestions to sketches.

7) If no favorite emerges you can dot vote to find the most promising sketches.

Variant: After step #5 (round #2) have a round of critiquing all sketches. This feedback can only be **negative**! Have a 3rd round of sketching, followed by critiquing. Now feedback can only be **positive**. Continue with step #6.

Facilitation

Facilitators support a discussion with structure. They reign in rambling people and encourage others to say what needs to be said to bring important matters out into the open.

As a PO you will attend and sometimes run a number of meetings. Here are some Facilitation basics on how to structure discussions and meetings so that they are time well spent.

Content

PO3

As an alternative to the first Meeting Checklist, check out PO3 to prepare your meetings. PO3 helps you gain clarity into why you are running this particular meeting (Purpose), what you want to achieve during it (Objectives), how you'll document it (Outputs) and its intended overall effect (Outcomes).

More Information

More about PO2 (no Objectives in this one):
MeetingMagic.co.uk/blog/purpose-outcomes-and-outputs/

Planning a meeting? Gain clarity for yourself and set expectations for the participants by filling out a PO3 and including it in the invite.

Name ______________________________

Purpose

Why do this? What would be missing without this meeting?

PO3

Planning Meetings with Purpose

Objectives

What do you want to achieve during the meeting? (Similar to an agenda)

Outputs

How will you document the meeting or its results? Do you need details notes? Will a photo of stickies suffice?

Outcomes

If it works, what effect(s) will you able to observe? What is going to change? Short term? Long term?

Ways to Vote

A big part of the magic of Agile Software Development are self-organizing teams. These teams take a lot of decisions themselves that used to taken by managers. When first trying to work this way, it can be difficult to agree after a group discussion. Here are some basic ways to vote to help teams take decisions.

You’ve got a list of options and want to pick the top ones fast

Dot Voting

All participants get the same number of votes to split and distribute on the options however they like. Fewer votes mean faster counting. 3 votes per person is a popular number. In this case 1:1:1, 1:2 and 3 votes for the same topic are all valid.

In the name-coining form of Dot Voting the facilitator hands out stick-on dots to the participants. 1 dot per vote. If you're not concerned about someone manipulating the votes, you can also use X's, drawn with big markers. That way it also works on whiteboards.

Ways to Vote

In agile, self-organizing teams often take decisions for themselves instead of relying on managers. This can call for votes to settle a discussion. Check out these basic ways to vote!

“Was that a decision?”

In a discussion the participants seem to have reached consensus yet it's not explicit

You don't always need a vote. In unclear situations, encourage everyone to loudly ask “Was that a decision?”. This question often gets enough silent people to voice a “Yes” so that people know that they indeed just took a decision. If they object, people are forced to speak up.

Gauge the support for one proposal

Roman Vote

Everybody votes at the same time by showing a thumb up, horizontal or down. Thumb up shows support, thumb to the side means "I'll support whatever the majority wants" and thumb down means that you want to speak.Beware of lots of horizontal thumbs. Support is lukewarm at best.

Visual Cues & ELMO

Have you ever been in a discussion and someone just keeps talking and talking? You are wondering to yourself: “Is this still relevant? Are the others still interested? Do they look bored? But I don't wanna interrupt. What if the others want to hear it?”

The visual cues on this 1-pager need less effort learning and remembering than the Hand Signals. They do need some material, though.

Have you ever been in a discussion and someone just keeps talking and talking? You are wondering to yourself: “Is this still relevant? Are the others still interested? Do they look bored? But I don't wanna interrupt. What if the others want to hear it?”

Depending on your cultural background it might be considered rude to interrupt. Visual cues to the rescue! They convey the message to the person speaking (and everybody else) without breaking the flow. It allows people to self-adjust. Most people want to stay relevant and welcome these hints.

Visual cues in discussions:

Jeff Patton's Cups & Elmo

In a real life meeting you use white paper cups as cues: Write messages such as "tangent" or "sold" on them and place them on the table as hints. Everybody can create additional messages as needed.

In a remote session you can you hold up cards with messages. Buy special sets of postcards for this or just create your own on the fly. Additionally, some tools have built-in mechanisms for non-disruptive feedback such as the reactions in Zoom.

Visual cues enable a group of people to keep each other on track – in discussions with or without a dedicated facilitator.

The **E.L.M.O.** technique is similar but uses only one visual cue: An Elmo doll (or drawing). When a topic has run its course but people are still talking, hold up Elmo to signal "Enough, let's move on!"

Beyond Agile & Scrum

Right now, Scrum is the dominant software development process. Knowing about approaches outside of Scrum will make you a more well-rounded Product Owner, someone who is able to pull ideas from many different sources.

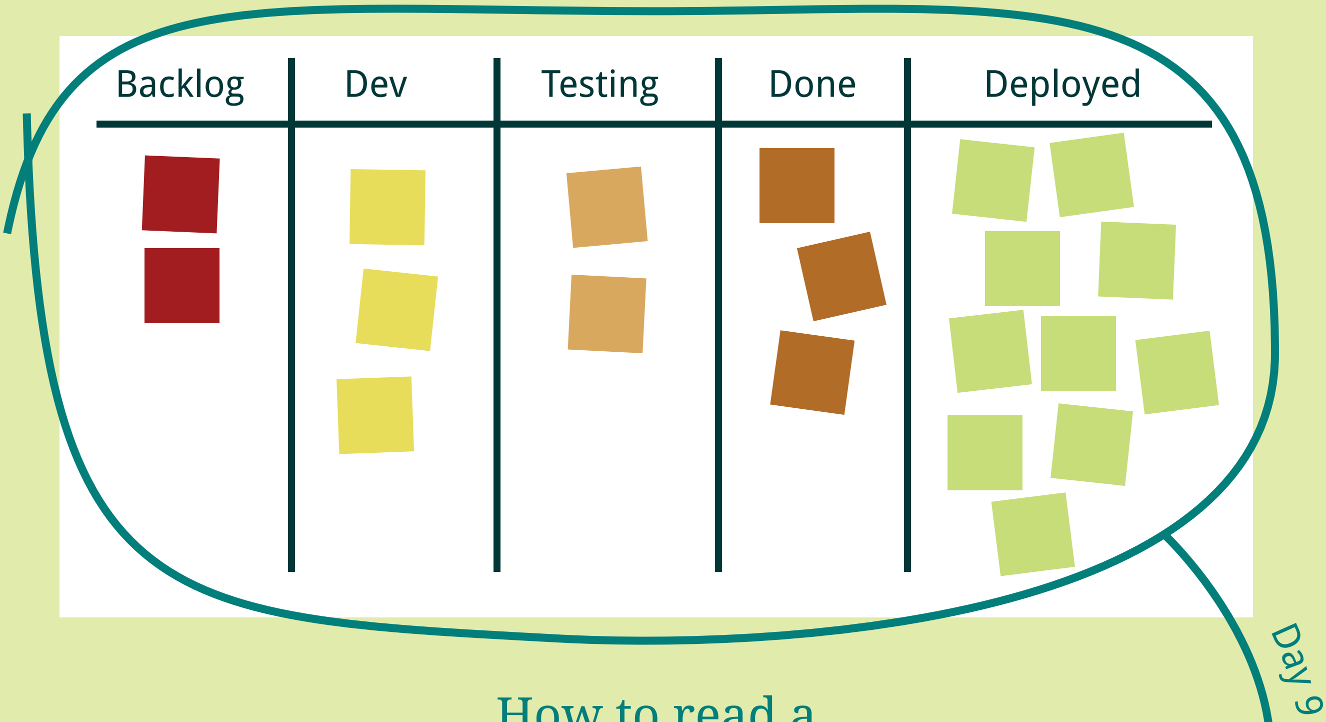
Content

Cumulative Flow

Cumulative Flow Diagrams are a wonderful tool to see trends and find bottlenecks in your delivery process. They are often used in Kanban environments.

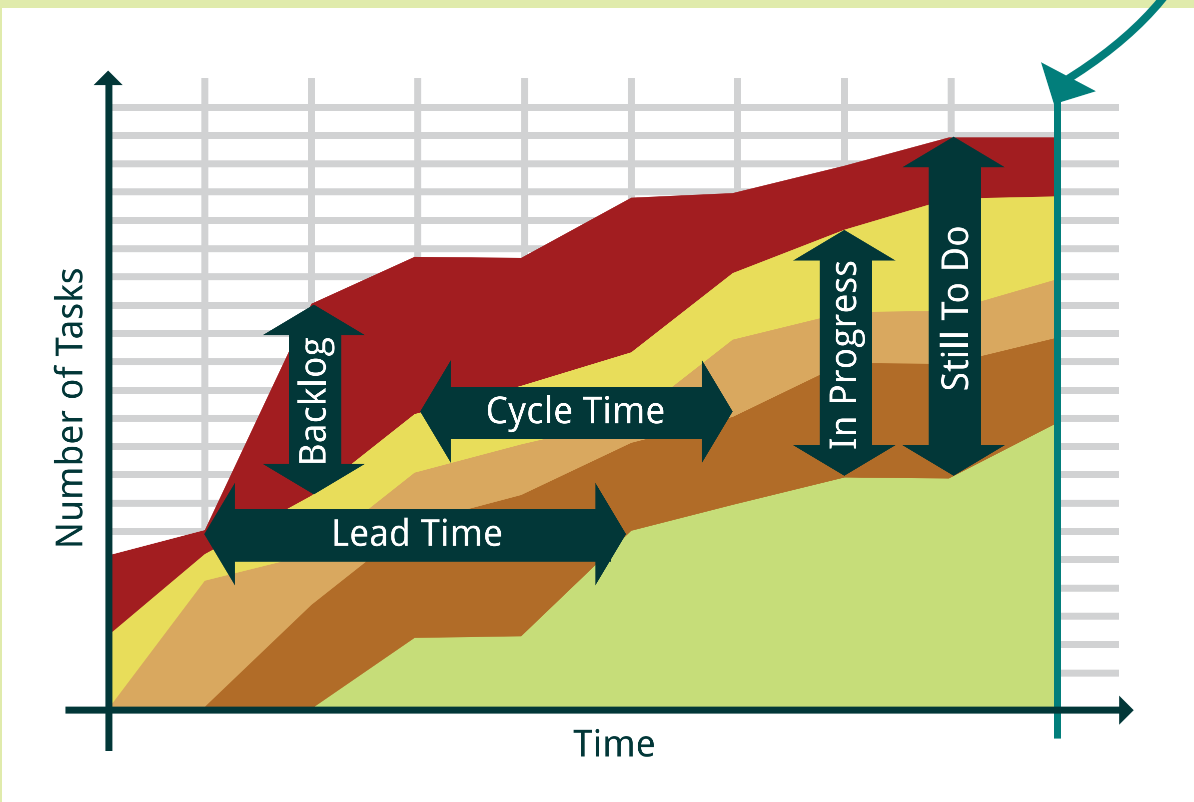
More Information

An excellent article by Pawel Brodzinski's on Cumulative Flow Diagrams inspired the 1-pager. Find it here: http://brodzinski.com/2013/07/cumulative-flow-diagram.html



How to read a

Cumulative Flow Diagram



Content adapted from Paul Klipp & Pawel Brodzinski

Designed by Wall-Skills.com - Spread Knowledge with 1-Pagers

Five Love Languages

The marriage counselor Gary Chapman popularized the concept of Love Languages: The concept that people express their love in different ways and that sometimes the love gets lost in translation. Which may leave one or both partners feeling unloved, which damages the relationship.

Why is this topic in a book for professionals? Replace "love" with "appreciation" and the parallels are striking.

More Information

"The 5 Love Languages" by Gary Chapman or "Love Languages in the Workplace"

Gifts

A gift is a sign that you thought of them. Gifts don't have to be big and fancy. At work it can be small tokens like bringing an extra bagel or via a kudos programme in which employees can give each other bonusses.

This love language includes the "Gift of Self" – being there in a crisis, when it really counts.

Words of Affirmation

People with this primary love language thrive on the spoken or written word – words of encouragement, compliments etc. A "Thank you" goes a long way.

Speak kind words to them, to others in their presence or about them behind their back.

5 Love Languages

Why do marriages fail? Many fail because a spouse doesn't feel loved by their partner. Sometimes to the utter surprise of said partner. People express love in different ways and gestures of love can get lost in translation, leaving spouses feeling unloved.

Most people have a primary and secondary love language. They express their love in these languages and also best understand love in these. If you want to fill your partner's “love tank” and make them feel loved, show your love in their love language, not your own.

The same concept applies at work regarding appreciation. Many people quit their jobs when they do not feel appreciated.

Physical Touch

If you're thinking "I love sex! This must be my love language!", think again. Do you also thrive on hugs, kisses, holding hands, a pat on the back?

This one is the most difficult at work. Maybe a handshake, fist bumps, ...?

Quality Time

This love language is about undivided attention (drop that phone!). It has two major dialects: 1) Talking to really understand the other person and 2) "Quality Activities" – doing something together, building nice memories.

At work, have team events or 1on1s.

Acts of Service

Some people let actions speak. To fill their love tank, help them solve a problem. Take over a task you know they hate. If you can, anticipate their needs.

WALL-SKILLS

Adapted from "The 5 Love Languages" by Gary Chapman

Designed by Wall-Skills.com - Spread Knowledge with 1-Pagers

About

Corinna Baldauf has filled every Scrum role there is and then some. She's on an eternal mission to make knowledge accessible. She's interested in lean, agile, systems thinking, communication, leadership & UX.

Find more 1-pagers like the ones in this book at Wall-Skills.com – her resource for condensed educational content. The topics range from Agile to DevOps to Soft Skills.

Got a question or suggestion? Email hallo@corinnabaldauf.de or contact @CorinnaBaldauf@nrw.social on Mastodon.

Acknowledgements

My sincere thanks go to my Wall-Skills partner Roland Wolters, to my husband Tobias, and to my parents, who instilled in me the deep wish to pass on whatever knowledge I acquire.