

# Best Practices of Agile teams

...

v2.0.2

**Today:**  
Which practices separate great  
Scrum teams from others?

Top 9 things the best Scrum  
Today: get right  
teams: Which time!  
every  
Scrum teams from You won't believe nr. 7!!  
ices separate great  
others?

# Practicalities

A barely organized list of good things to do.

Not actually a Scrum talk; it's just common.

Questions welcome.

QR-code for slides at the end

# Who am I



Jakob Buis

~~Developer~~

~~Team lead~~

~~Engineering Manager~~

Management consultant

Professional team builder

# Should you listen to me?

Yes, because:

Never been fired

Herd of 6 elePHPants

Worked with 15+ Scrum teams in various companies & industries

As a developer, manager and Scrum Master

Professional Scrum Master II

Ranking

Here you can find the top 50 collectors in the PHP community.

| Rank | Name          | Country     | Unique | Total | Updated  |
|------|---------------|-------------|--------|-------|----------|
| 1    | Damien Seguy  | Netherlands | 64     | 64    | 2 weeks  |
| 2    | Daan          | Netherlands | 60     | 117   | 3 months |
| 3    | ElePHPant_frl | Netherlands | 60     | 65    | 5 hours  |
| ...  |               |             |        |       |          |
| 33   | Jakob Buis    | Netherlands | 6      | 6     | 3 months |
| 34   | DJneo         | Netherlands | 5      | 6     | 1 year   |

Made with ❤ by [Junior Grossi](#), [Igor Duarte](#) and [contributors](#). Contribute to this project on [GitHub](#).

# Should you listen to me?

## Yes, because:

Never been fired

Herd of 6 elePHPants

Worked with 15+ Scrum teams in various companies & industries

As a developer, manager and Scrum Master

Professional Scrum Master II

## No, because:

I fuck up, a lot

Worked with 16 teams

Mostly in smaller companies  
(< 300 people, < 30 engineers)

Most of my ideas come from other people  
(links included!)

# Understanding Scrum

Daily Scrum

Sprint

Definition of Done

Sprint Backlog

Developer

Sprint Goal

Empiricism

Sprint Planning

Increment

Sprint Retrospective

Product Backlog

Sprint Review

Product Goal

Technical debt

Product Owner

Experiment beyond Scrum  
not before Scrum

# THE SCRUM Police



Daily Scrum on Mon & Wed  
Demo work not completed  
Retrospective every 3rd sprint

Daily Scrum on Mon & Wed  
Demo work not completed  
Retrospective every 3rd sprint

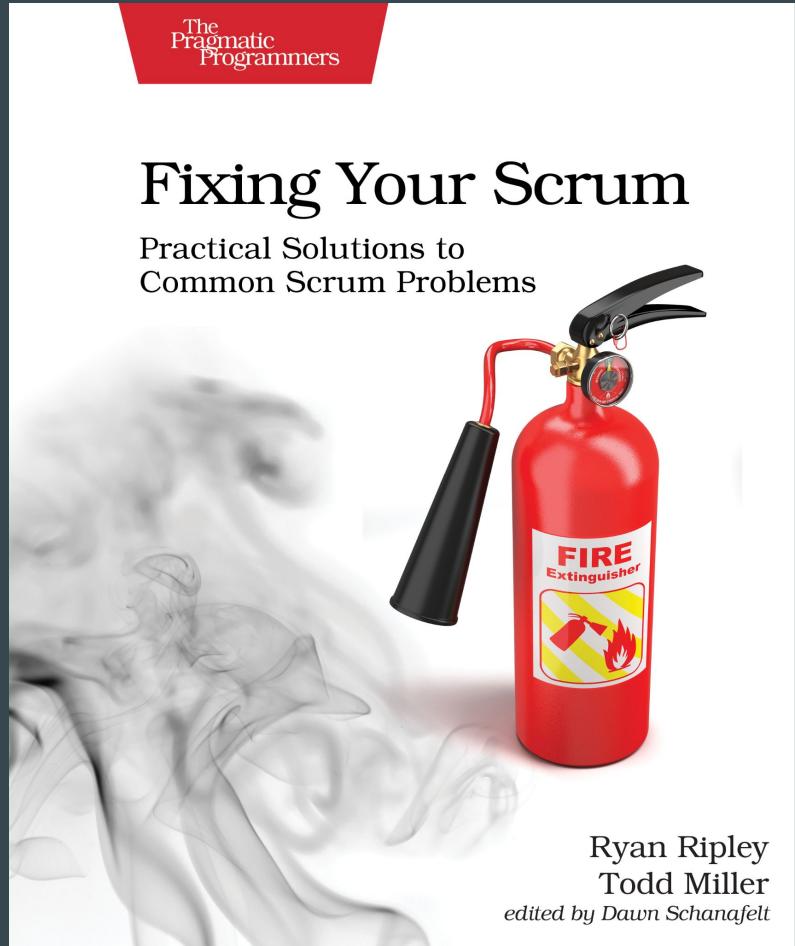
Scrum doesn't work for us

# To do

Read the Scrum Guide

Read a book

Take a course



Experiment beyond Scrum  
not before Scrum

Working, tested software  
every sprint

# Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it.

Through this work we have come to value:

- 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.

|                   |                |                  |
|-------------------|----------------|------------------|
| Kent Beck         | James Grenning | Robert C. Martin |
| Mike Beedle       | Jim Highsmith  | Steve Mellor     |
| Arie van Bennekum | Andrew Hunt    | Ken Schwaber     |
| Alistair Cockburn | Ron Jeffries   | Jeff Sutherland  |
| Ward Cunningham   | Jon Kern       | Dave Thomas      |
| Martin Fowler     | Brian Marick   |                  |

© 2001 the above authors  
this document may be freely copied in any form,  
but only in its entirety through this notice.

## Twelve Principles of Agile Software

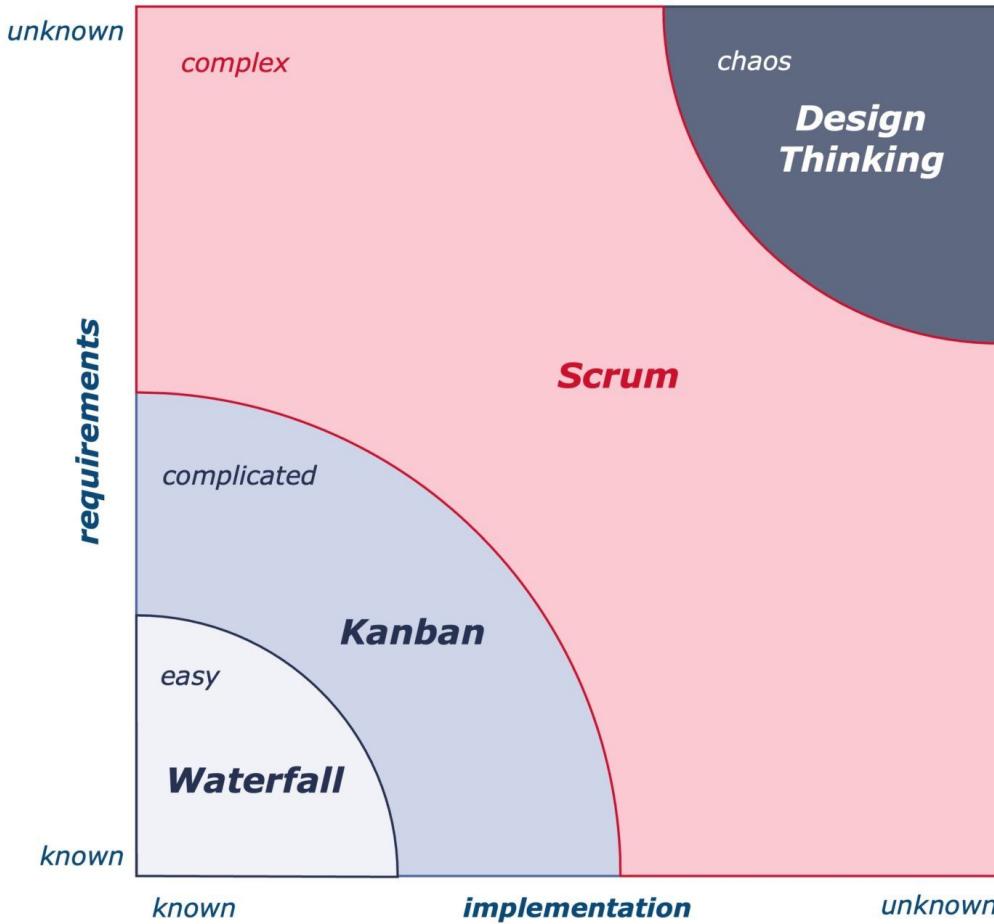
[View Signatories](#)

[About the Authors](#)

[About the Manifesto](#)

**Principle 1:**  
Our highest priority is to  
satisfy the customer through  
**early and continuous delivery**  
of valuable software.

# *Stacey Matrix*



**Principle 7:**  
**Working software** is the  
primary measure of **progress.**

# Principle 7:

**Working software** is the primary measure of **progress.**



in the **customers** hands!

# SPRINT REVIEW



SCRUM TEAM



STAKEHOLDERS

# To do

Get really good at vertical slicing

Start here: <https://www.reddit.com/r/agile/comments/lbtxpzd>

# To do

Get really good at vertical slicing

Start here: <https://www.reddit.com/r/agile/comments/lbtxpzd>

Erase all dependencies

- decoupling architecture
- team changes (#team-topologies)
- incur (some) technical debt
- feature flags

# To do

Get really good at vertical slicing

Start here: <https://www.reddit.com/r/agile/comments/lbtxpzd>

Erase all dependencies

- decoupling architecture
- team changes (#team-topologies)
- incur (some) technical debt
- feature flags

Be smart about risks, avoid big-design up-front

Working, tested software  
every sprint

3

Know how your customers  
are using the product

# SPRINT REVIEW



SCRUM TEAM



STAKEHOLDERS

There is **nothing** so useless  
as doing with great efficiency  
that which **should not be done** at all.

Peter Drucker



# To do: tracking

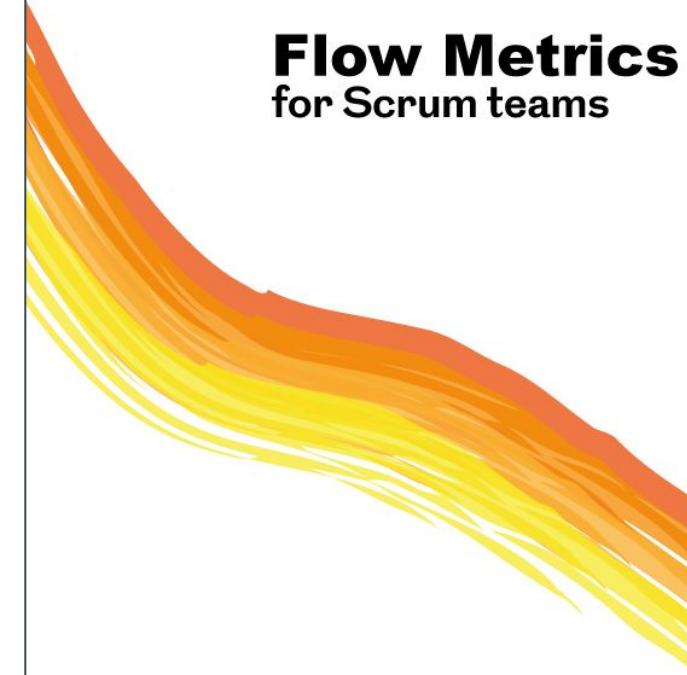
| feature_foo_clicks |                |                      |
|--------------------|----------------|----------------------|
| <b>id</b>          | <b>user_id</b> | <b>timestamp</b>     |
| 1                  | 1              | 2025-03-10T14:30:10Z |
| 2                  | 2              | 2025-03-10T14:31:23Z |
| 3                  | 1              | 2025-03-11T09:16:00Z |
| 4                  | 3              | 2025-03-12T04:10:59Z |

# To do: visualisation

1. Options (Backlog)
2. Discovery
3. Building
  - a. Not started
  - b. Coding
  - c. Code Review
  - d. Ready for release
4. Validating
5. Done



## Flow Metrics for Scrum teams



Will Seele & Daniel Vacanti

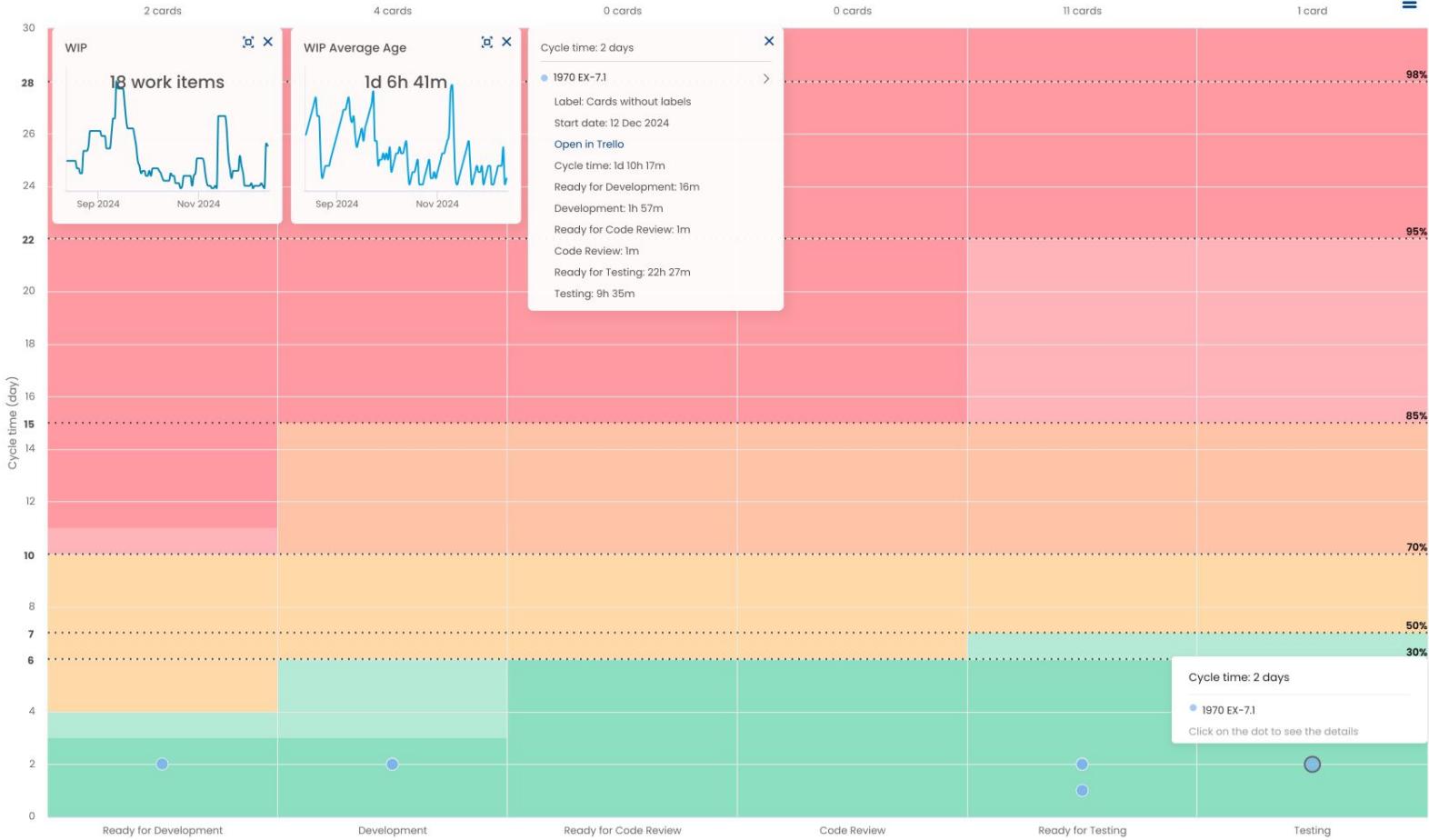
3

Know how your customers  
are using the product

5

Do all work  
on the board



Aging Chart 

13 Aug 2024 - 13 Dec 2024

Aging replay

13 Dec 2024

## Health zones

- Select all
- 30%
- 50%
- 70%
- 85%
- 95%

## Percentiles

## Group by

## Filters (18 cards)

- Select all
- Ready for Development
- Development
- Ready for Code Review
- Code Review
- Ready for Testing
- Testing

## Labels (3/3)

- Select all
- Cards without labels
- Can't be Reproduced
- Ready for Retention

## Members (6/6)

# To do

Create item: [title] + [assigned you] + [in progress]

Consider skipping ticket when:

- doing it right now
- takes < 10 minutes (and you're 99% certain)
- is a repeating action (automate it!)

Bias to having a single board per team.

Items never go back: stuck is preferable.

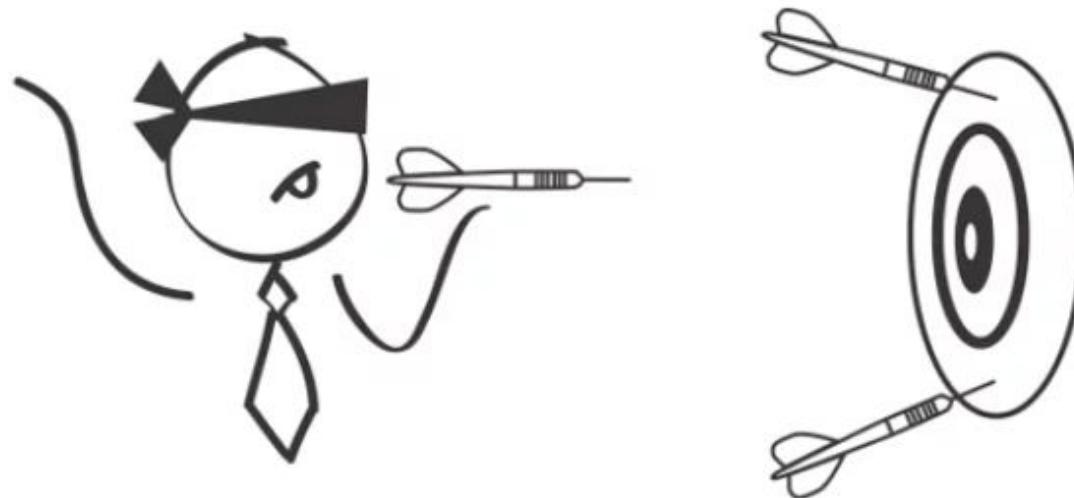
5

Do all work  
on the board

All estimates  
are bullshit

# Estimation

The fine art of guessing



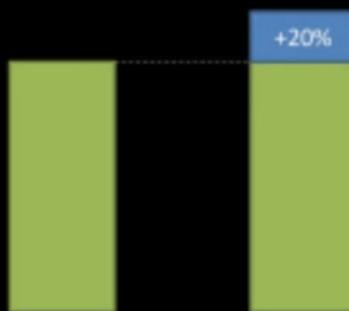
# #NoEstimates

# After just 3 sprints

## Story Points predictive power

The true output:  
349,5 SPs  
completed

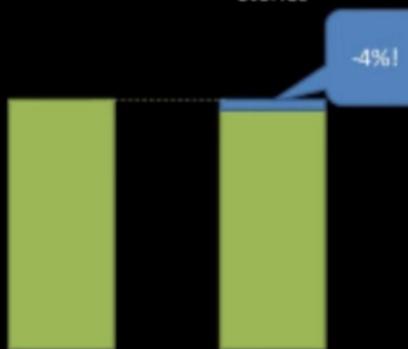
The predicted  
output: 418 SPs  
completed



## # of Stories predictive power

The true output:  
228 Stories  
completed

The predicted  
output: 220  
Stories



#NoEstimates (Allen Holub)

<https://www.youtube.com/watch?v=QVBIInCTu9Ms>

# To do

Good:

same-sizing everything: "1 story point" and "too big"

<https://mdalmijn.com/p/roman-estimation-a-simple-easy-and>

Better:

use data

# Monte Carlo simulation

Record throughput per day:

0    7    2    6    6    3    7    2    9    1    13    0    0    2

# Monte Carlo simulation

Record throughput per day:

0    7    2    6    6    3    7    2    9    1    13    0    0    2

Sample next 7 days:

2    0    2    7    0    3    2

# Monte Carlo simulation

Record throughput per day:

0    7    2    6    6    3    7    2    9    1    13    0    0    2

Sample next 7 days:

2    0    2    7    0    3    2                 = 16

Next week, we'll finish **16** stories.

## Simulation controls

Start Date  
15 Sep 2018

End Date  
15 Oct 2018

Trials  
10000

## Lists

- Select all
- To do
- Development
- Code review
- Code review (Done)
- Testing
- Testing (Done)
- Deployment
- Done

## Labels

- Select all
- Cards without labels
- Expedite
- Fixed Delivery Date
- Intangible
- Standard

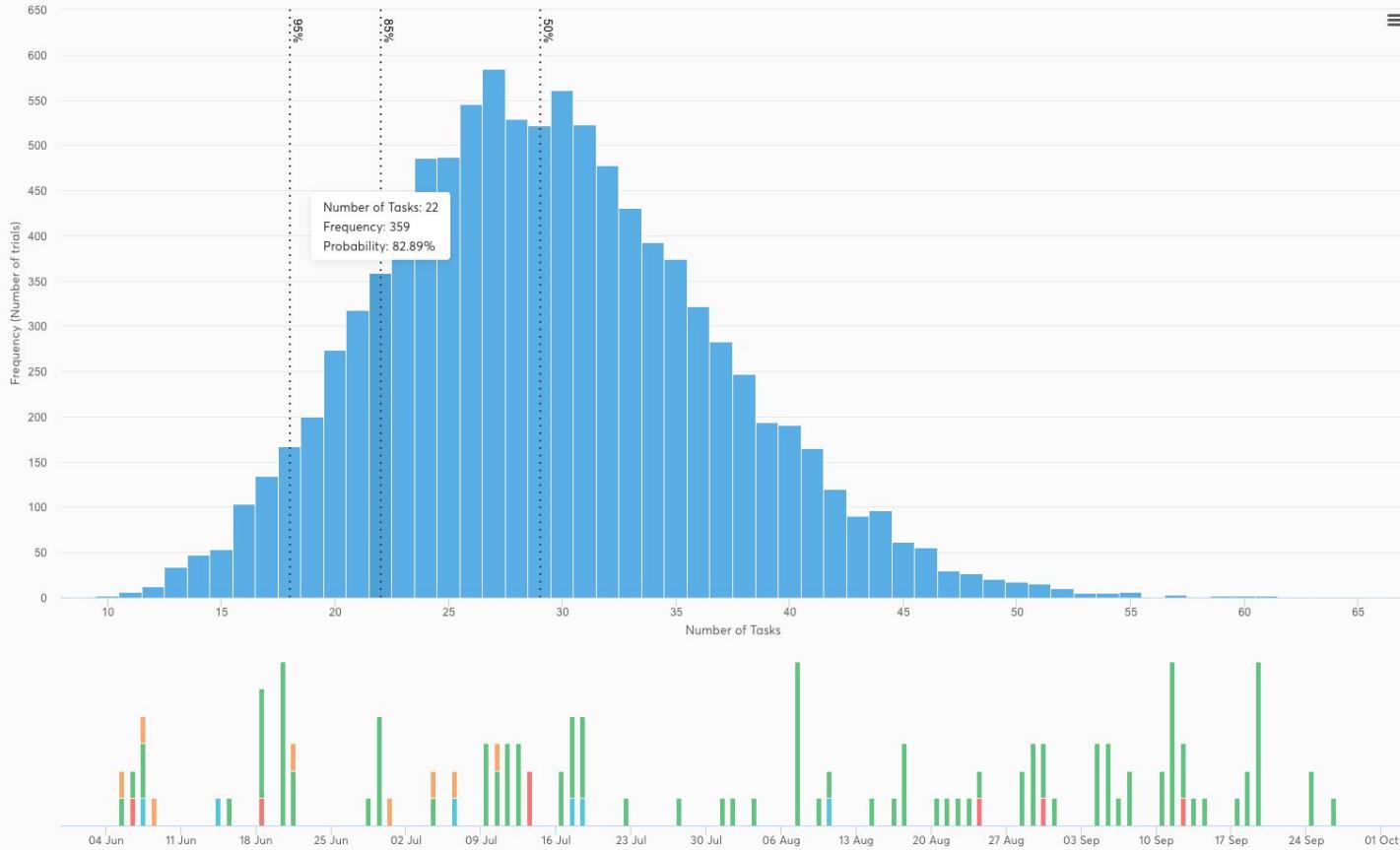
## Members

## Percentiles

- Select all
- 30%
- 50%



Monte Carlo: Number of Tasks



## Simulation controls

Start Date: 15 Sep 2018

Items to complete: 10

Trials: 10000

## Lists

- Select all
- To do
- Development
- Code review
- Code review (Done)
- Testing
- Testing (Done)
- Deployment
- Done

## Labels

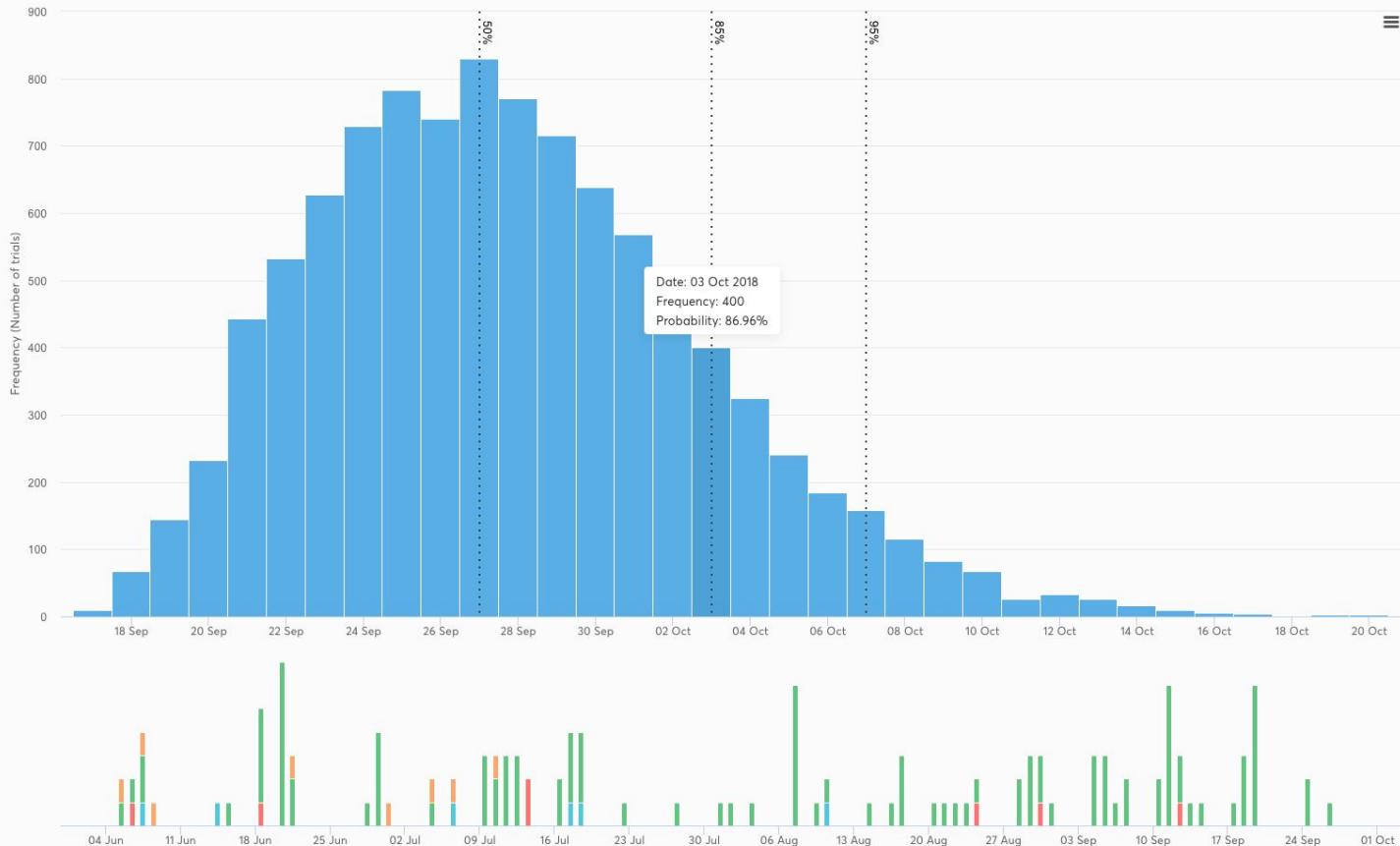
- Select all
- Cards without labels
- Expedite
- Fixed Delivery Date
- Intangible
- Standard

## Members

## Percentiles

- Select all
- 30%
- 50%

Monte Carlo: Delivery Date



Cumulative Flow Diagram

Cycle Time Scatterplot

Cycle Time Breakdown Chart

Cycle Time Histogram

Aging Chart

Throughput Run Chart

Throughput Histogram

Flow Efficiency Chart

Monte Carlo: Delivery Date

Monte Carlo: Number of Tasks

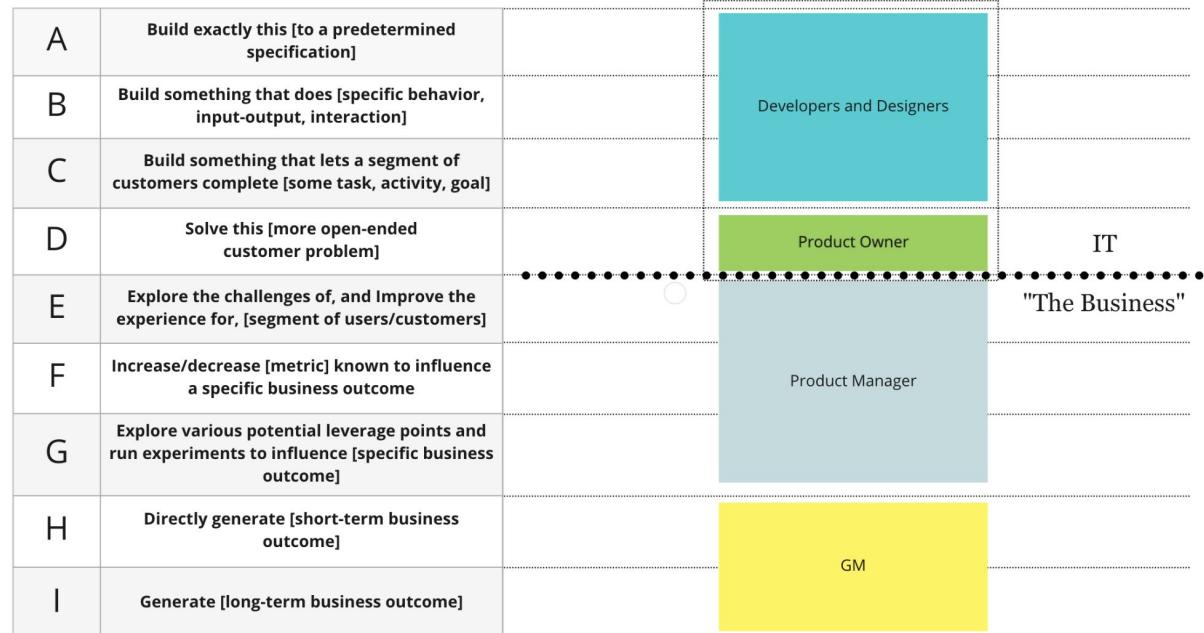
All estimates  
are bullshit

Solve business problems

# MANDATE LEVELS

@johncutlefish

Effort is happening at all of these levels concurrently. It is all connected (explicitly, and often implicitly).



<https://cutlefish.substack.com/p/tbm-2752-mandate-levels>



# To do

Don't start here

Working software in the customers hands

Build a prototype

Get users in the room while designing

Solve business problems

# Deal with technical debt

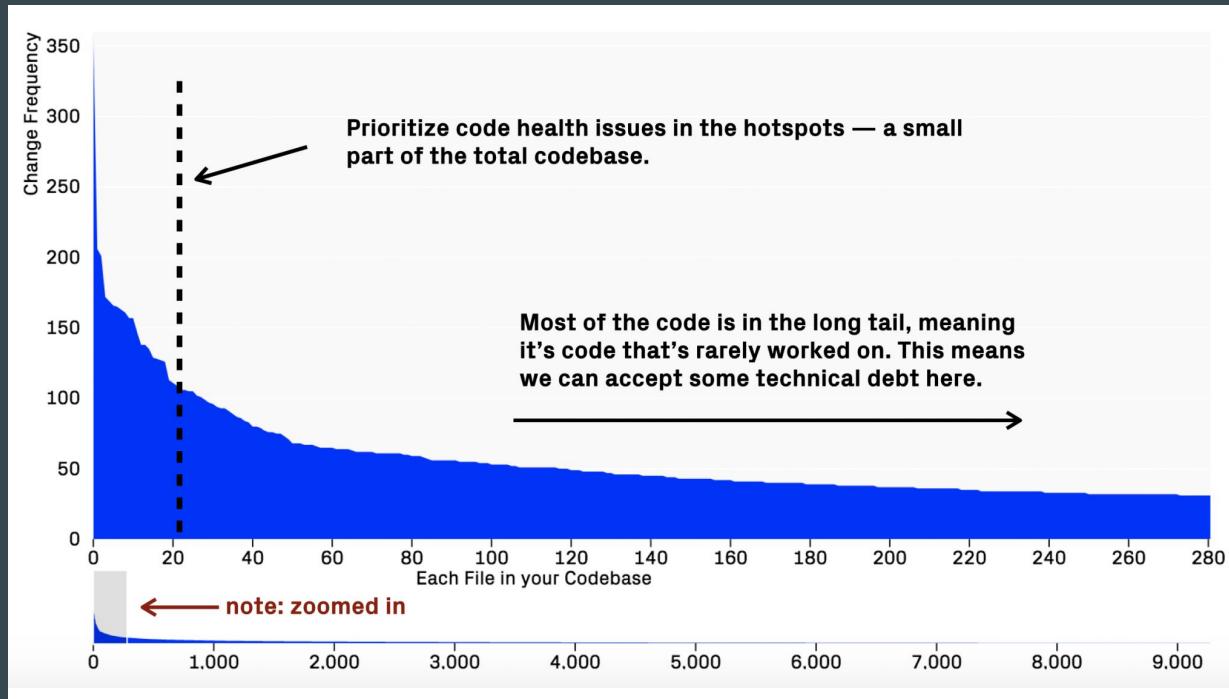


QUALITY



CONSISTENCY

# To do



<https://www.getunleash.io/blog/manage-technical-debt-measure-the-impact-and-prioritize-improvements-guided-by-development-data>

# To do

Minimum:

1. error monitoring service (sentry)
2. static analysis (psalm, phpstan)
3. existing coding standard enforced by linter (pint)

Fix bugs

# Deal with technical debt

Have a strong  
Definition of Done



# To do

Absolute

Automated

Agreed with PO

Have a strong  
Definition of Done

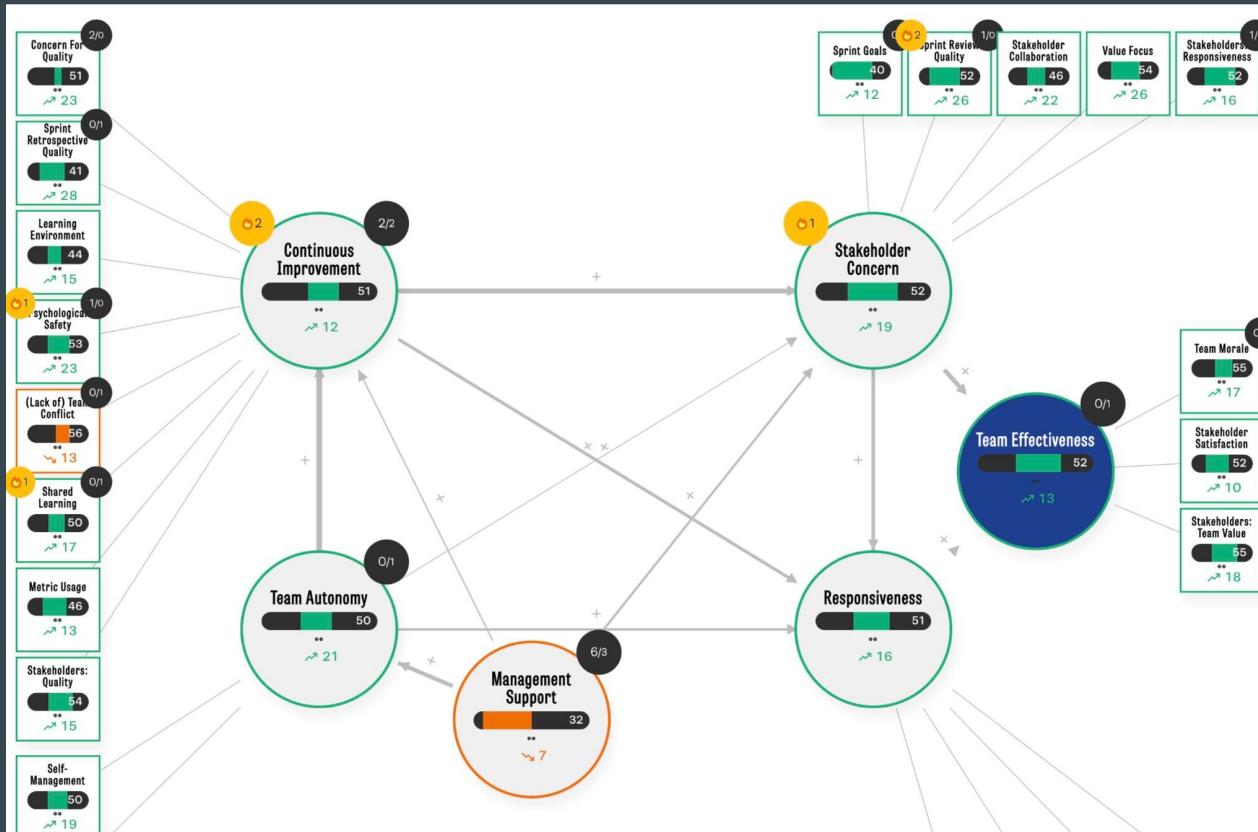
Make retrospectives effective



# To do

Escalate what you cannot solve

Data-driven decision making



**nave**

- Process Improvement Dashboard
- Planning & Forecasting
- Prioritization & Dependency Management
- Flow Metrics & Analytics
- Home
- Cumulative Flow Diagram
- Cycle Time Scatterplot
- Cycle Time Breakdown
- Cycle Time Histogram
- Aging Chart
- Throughput Run Chart
- Throughput Histogram
- Flow Efficiency Chart
- Due Date Performance Chart
- Monte Carlo: Delivery Date
- Monte Carlo: Number of Tasks
- Executive View
- Executive Dashboard
- Executive Report

Development (CONWIP: 2 Planned + 1 Unplanned)  
Last Updated by Sonya



#### Controls for all charts

##### Colors

- Issue Type
- Status
- Priority
- Severity of Impact

##### Cycle time precision

##### Filters (70 cards)

###### Statuses (1/4)

- Development
- Code Review
- Testing
- Deployment

###### Priorities (4/4)

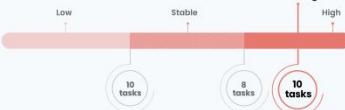
- Expedite
- Fixed Date
- Standard
- Intangible

###### Sprints (1/4)

- Sprint 01 - Launch Prep
- Sprint 02 - UI Improvements
- Sprint 03 - Bug Fixes
- Sprint 04 - UX Updates

### Work in Progress

HIGH

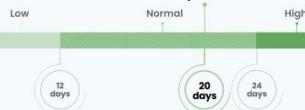


Your WIP is too high! Focus on moving current tasks along, especially the ones that are almost done:

- Under Review [DEV] 2222-738
- Ready for Review [DEV] 5302-730
- Ready for Review [DEV] 9983-283

### Cycle Time

STABLE

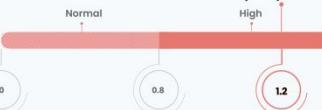


Your cycle time is stable! If you're still looking for improvement opportunities, consider reviewing the items with the longest cycle time:

- 16 days [DEV] 7382-849
- 12 days [DEV] 6638-839
- 12 days [DEV] 2673-098

### Throughput

HIGH



Your throughput is too high! Review these tasks to identify what factors contributed to this boost and assess if they can be sustained in the long run:

- Aug 4th [DEV] 8293-283
- Aug 4th [DEV] 8983-103
- Aug 4th [DEV] 8112-099

### Work in Progress Age

AT RISK



Your WIP age is at risk! Consider prioritizing the tasks with the highest work in progress age to bring your workflow back under control:

- 16 days [DEV] 1232-190
- 12 days [DEV] 3320-827

### Cycle Time per Process Step

HIGH



Your Development cycle time is too high! Focus on assessing the issue that caused the delay to bring cycle time on that status back in line:

- 16 days [DEV] 8982-772

### Flow Efficiency

STABLE



Your flow efficiency is stable! If you want to further reduce wait times, consider analyzing the following items with the lowest flow efficiency and highest cycle time:

- 16 days 23% [DEV] 537-009
- 12 days 26% [DEV] 346-098

Make retrospectives effective

To do

# To do

1. Experiment beyond Scrum, not before Scrum
2. Working tested software, every sprint.
3. Know how your customers are using the product
4. Do all work on the board
5. All estimation is bullshit
6. Solve business problems
7. Deal with technical debt
8. Having a strong Definition of Done
9. Make retrospectives effective

# How to get started

RICHARD RUMELT

"A giant in the field of strategy"—*McKinsey Quarterly*

GOOD  
STRATEGY  
BAD  
STRATEGY

The Difference  
and Why it Matters

The kernel of a strategy  
contains three elements:  
a **diagnosis**,  
a **guiding policy**,  
and **coherent action**.

RICHARD RUMELT

"A giant in the field of strategy"—*McKinsey Quarterly*

GOOD  
STRATEGY  
BAD  
STRATEGY

The Difference  
and Why it Matters

# That's all



Slides  
Feedback  
Assessment

Don't forget to like,  
subscribe and hit  
the bell button.



Slides  
Feedback  
Assessment

# Bonus content

# Flow Metrics for Scrum teams

| PBI | Started    | Finished   |
|-----|------------|------------|
| 1   | 2025-03-01 | 2025-03-10 |
| 2   | 2025-02-16 | 2025-03-18 |
| 3   | 2025-03-17 |            |
| 4   |            |            |

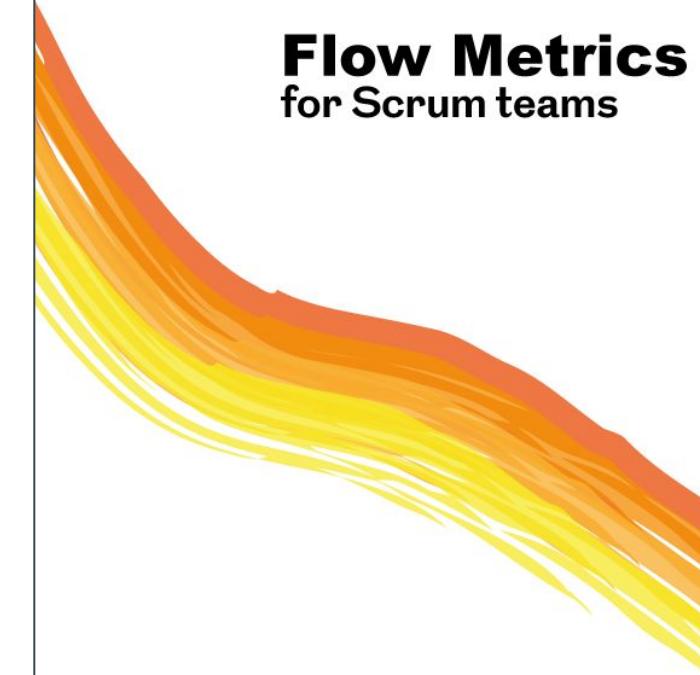
## Measure

- WIP
- Cycle time
- Work item age
- Throughput



ProKanban.org

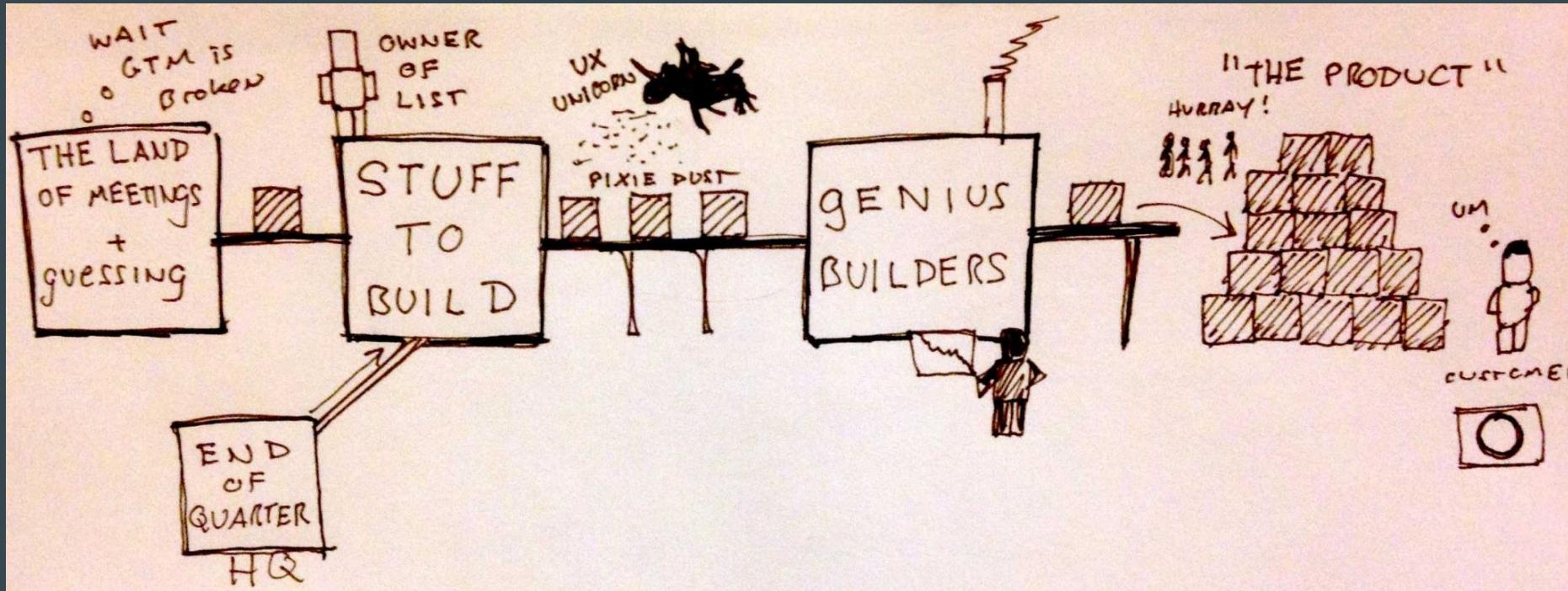
## Flow Metrics for Scrum teams



Will Seele & Daniel Vacanti

# Feature addiction

# Feature addiction



# Feature addiction

Unclear goals. The sprint goal != the work.

# Feature addiction

Unclear goals. The sprint goal != the work.

You almost certainly have a product management problem.

# Feature addiction

Unclear goals. The sprint goal != the work.

You almost certainly have a product management problem.

Read this: <https://kellanem.com/notes/faqs-from-coaching>  
on "the team isn't working hard enough"

# Feature addiction

Unclear goals. The sprint goal != the work.

You almost certainly have a product management problem.

Read this: <https://kellanem.com/notes/faqs-from-coaching>  
on "the team isn't working hard enough"

Run away.

You are not smarter  
than the customer



# To do

... don't do that

Respect the customer

Regular contact helps

# DORA metrics

# Key DORA Metrics



## Deployment Frequency

The number of times per day that a release is deployed into production.



## Lead Time for Changes

Total time between the initiation of a feature request to the delivery of that feature to a customer.



## Mean Time to Recovery (MTTR)

Average time it takes the team to restore service when a failure/outage occurs in production.



## Change Failure Rate

Percentage of changes that resulted in degraded services, like the service impairment or outage, and need to be fixed.

| Software delivery performance metric  | Elite                                | High                                     | Medium   | Low                            |
|---|--------------------------------------|--|--|--------------------------------|
| <p>⌚ <b>Deployment frequency</b></p> <p>For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?</p>  | On-demand (multiple deploys per day) | Between once per week and once per month | Between once per month and once every 6 months | Fewer than once per six months |
| <p>⌚ <b>Lead time for changes</b></p> <p>For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?</p>   | Less than one hour                   | Between one day and one week             | Between one month and six months               | More than six months           |
| <p>⌚ <b>Time to restore service</b></p> <p>For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?</p>   | Less than one hour                   | Less than one day                        | Between one day and one week                   | More than six months           |
| <p>⌚ <b>Change failure rate</b></p> <p>For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?</p> | 0%-15%                               | 16%-30%                                  | 16%-30%  | 16%-30%                        |

<https://cloud.google.com/blog/products/devops-sre/using-the-four-keys-to-measure-your-devops-performance>