

# Best Practices of Agile Teams

...

v3.0.4

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

Top 5 things the best agile  
teams get right, every time!

Today:  
Which practices separate great  
agile teams from others?

*Bonus: the 5 things  
everyone gets wrong!!*

# Practicalities

A barely organized list of good and bad things

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

Clarifying questions welcome!

# Who am I



Jakob Buis

~~Developer~~

~~Team lead~~

~~Engineering Manager~~

Management consultant

Professional team builder

[www.jakobbuis.nl](http://www.jakobbuis.nl) (now with blogging!)

# Should you listen to me?

Yes, because:

Never been fired

Herd of 7 elePHPants

Worked with 15+ agile teams in various companies & industries

Professional Scrum Master II

Ranking

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

Netherlands

Change

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 7 elePHPants

Worked with 15+ agile teams in various companies & industries

Professional Scrum Master II

## No, because:

I fuck up, a lot

Worked with 16 teams

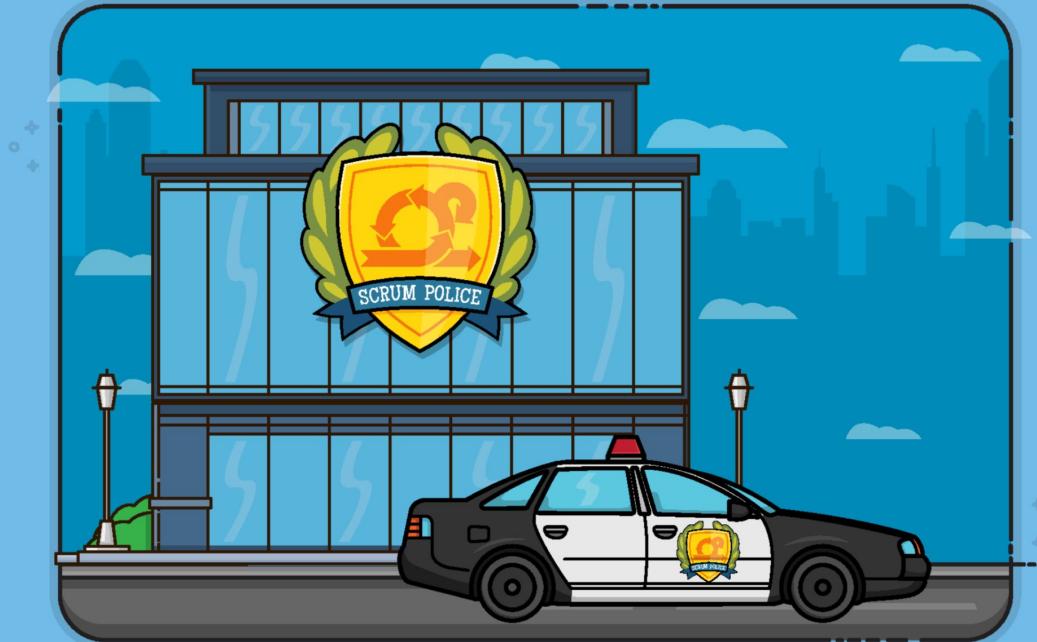
Mostly in smaller companies  
(< 300 people, < 30 engineers)

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

Do this

Avoid this

Experiment before complete methodology



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

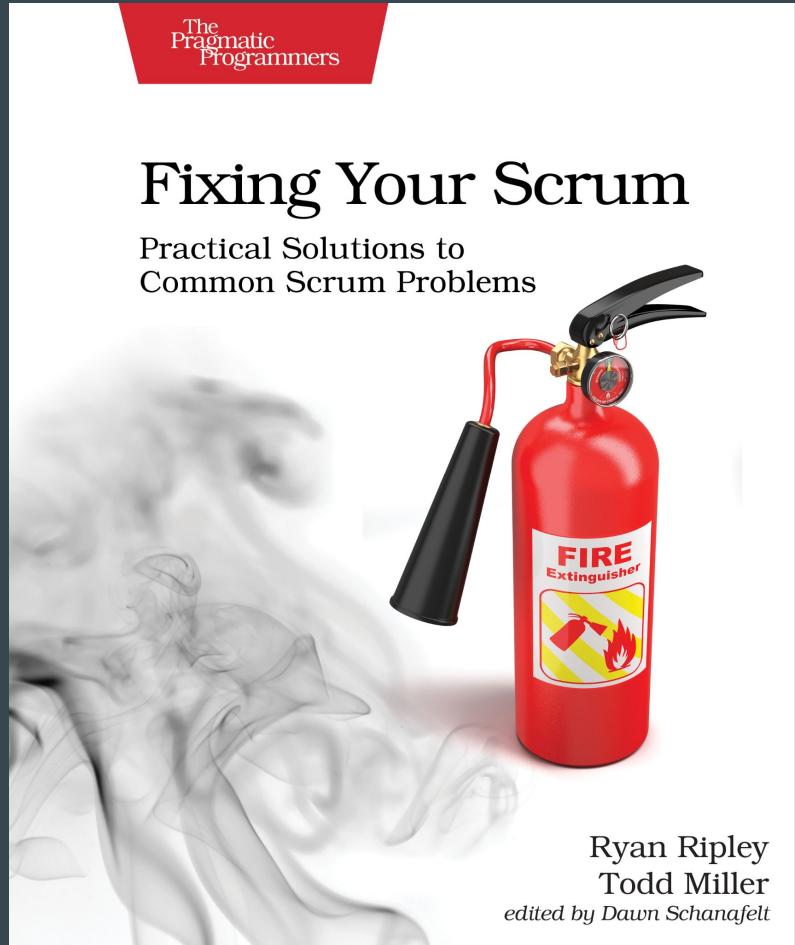
**Scrum doesn't work for us**

# Learn your method

Read the Scrum Guide

Read a book

Take a course



Do this

Avoid this

Experiment before complete methodology

## Do this

Working tested software, every sprint

## Avoid this

Experiment before complete methodology

# 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](#)

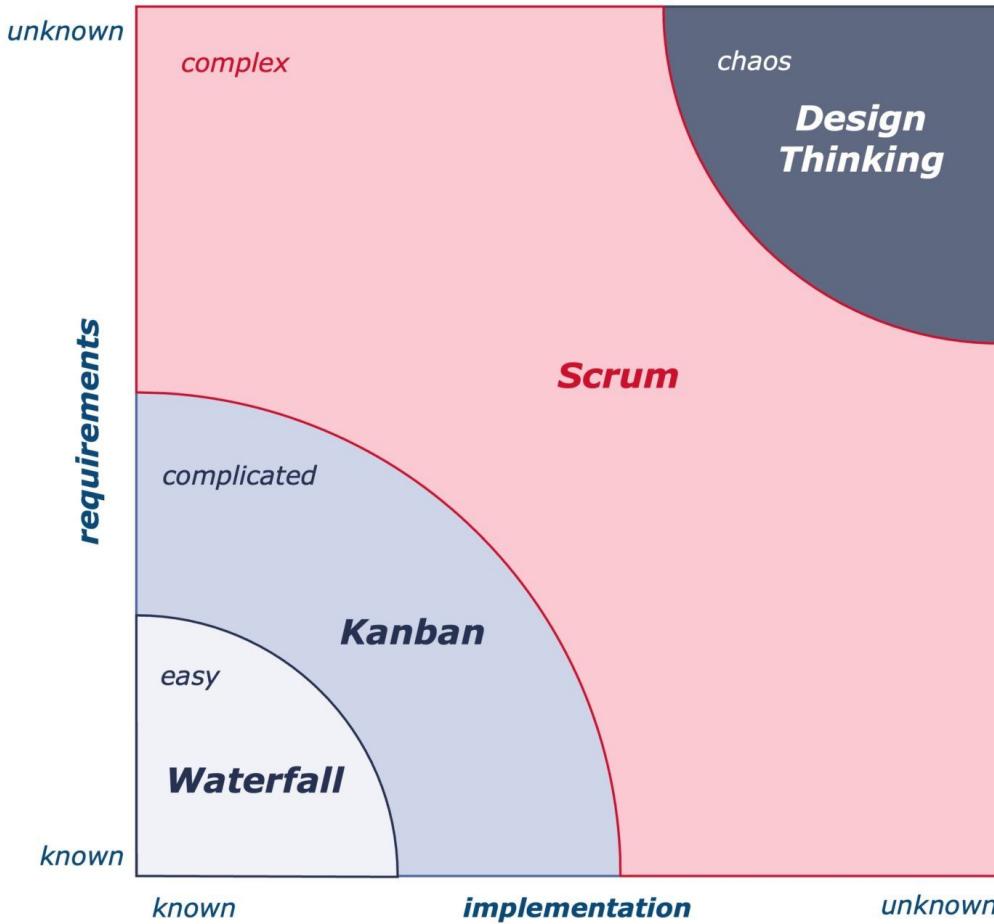
[About the Authors](#)

[About the Manifesto](#)

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

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

# *Stacey Matrix*



# SPRINT REVIEW



SCRUM TEAM



STAKEHOLDERS

# Working tested software, every sprint

Get really good at vertical slicing

Start here: <https://www.youtube.com/watch?v=urZlTIycedU>

# Working tested software, every sprint

Get really good at vertical slicing

Start here: <https://www.youtube.com/watch?v=urZlTIycedU>

Erase all dependencies

- decoupling architecture & operations
- team changes (Team Topologies)
- incur (some) technical debt

# Working tested software, every sprint

Get really good at vertical slicing

Start here: <https://www.youtube.com/watch?v=urZlTIycedU>

Erase all dependencies

- decoupling architecture & operations
- team changes (Team Topologies)
- incur (some) technical debt

Avoid big-design up-front

## Do this

Working tested software, every sprint

## Avoid this

Experiment before complete methodology

## Do this

Working tested software, every sprint

Know how the product is used

## Avoid this

Experiment before complete methodology

# 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



# Add tracking tables

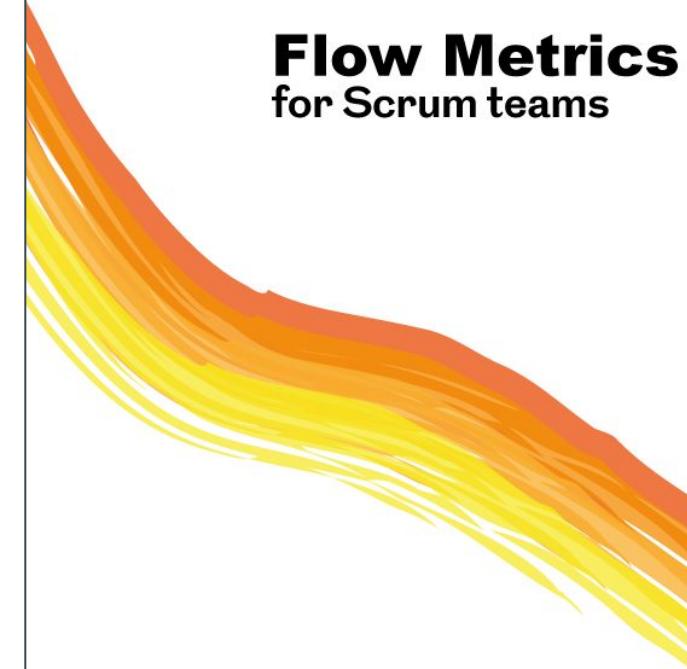
feature_foo_clicks		
<b>id</b>	<b>user_id</b>	<b>timestamp</b>
1	1	2025-03-10T14:30:10Z
2	2	2025-03-10T14:31:23Z
3	1	2025-03-11T09:16:00Z
4	3	2025-03-12T04:10:59Z

# Board expansion

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

## Do this

Working tested software, every sprint

Know how the product is used

## Avoid this

Experiment before complete methodology

## Do this

Working tested software, every sprint

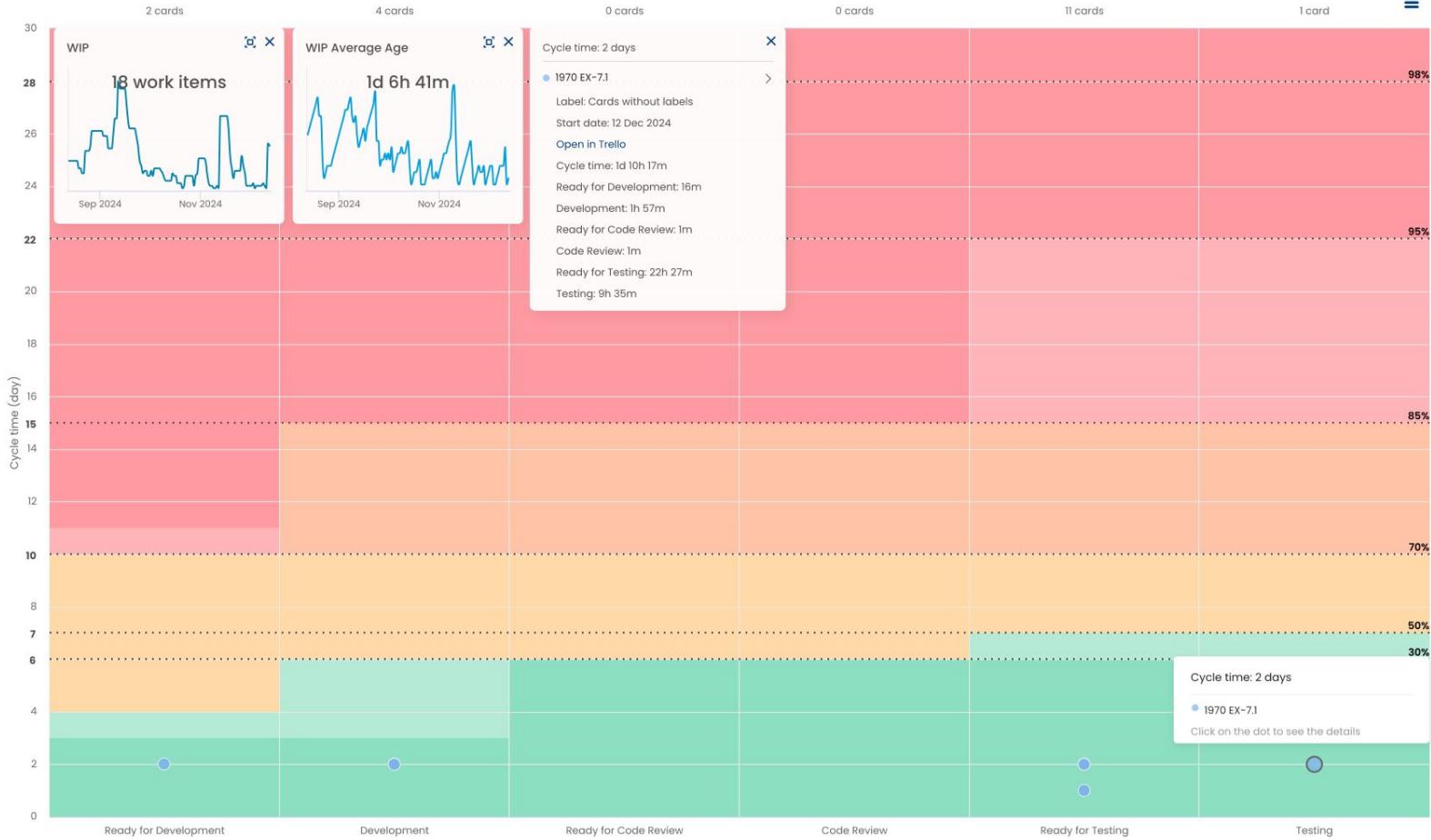
Know how the product is used

Do all work on the board

## Avoid this

Experiment before complete methodology



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)

## Lists (6/6)

- 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)

# Do all the work on the board

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!)

Try a physical board

Bias to having a single board per team.

Items never go back: stuck is preferable.

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

## Avoid this

Experiment before complete methodology

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

## Avoid this

Experiment before complete methodology

You're not smarter than the customer



## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

This guy is a software engineer,  
you can tell by his awesome  
estimation skills



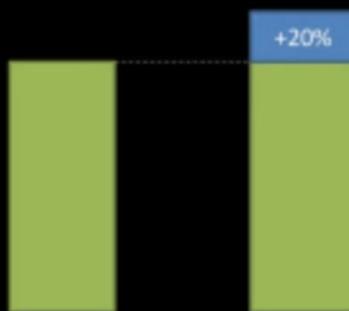
# #NoEstimates

# After just 3 sprints

## Story Points predictive power

The true output:  
349,5 SPs  
completed

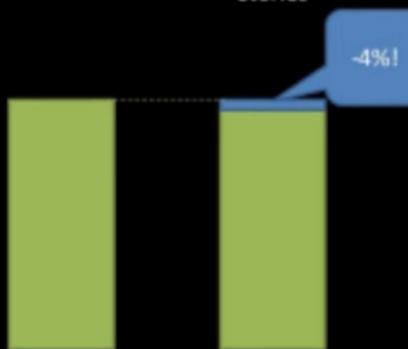
The predicted  
output: 418 SPs  
completed



## # 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>

# Don't estimate

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    4

# Monte Carlo simulation

Record throughput per day:

0    7    2    6    6    3    7    2    9    1    13    0    0    2    4

Sample next 5 days:

2    0    2    7    0

# Monte Carlo simulation

Record throughput per day:

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

Sample next 5 days:

2    0    2    7    0    = 11

Next week, we'll finish **11 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

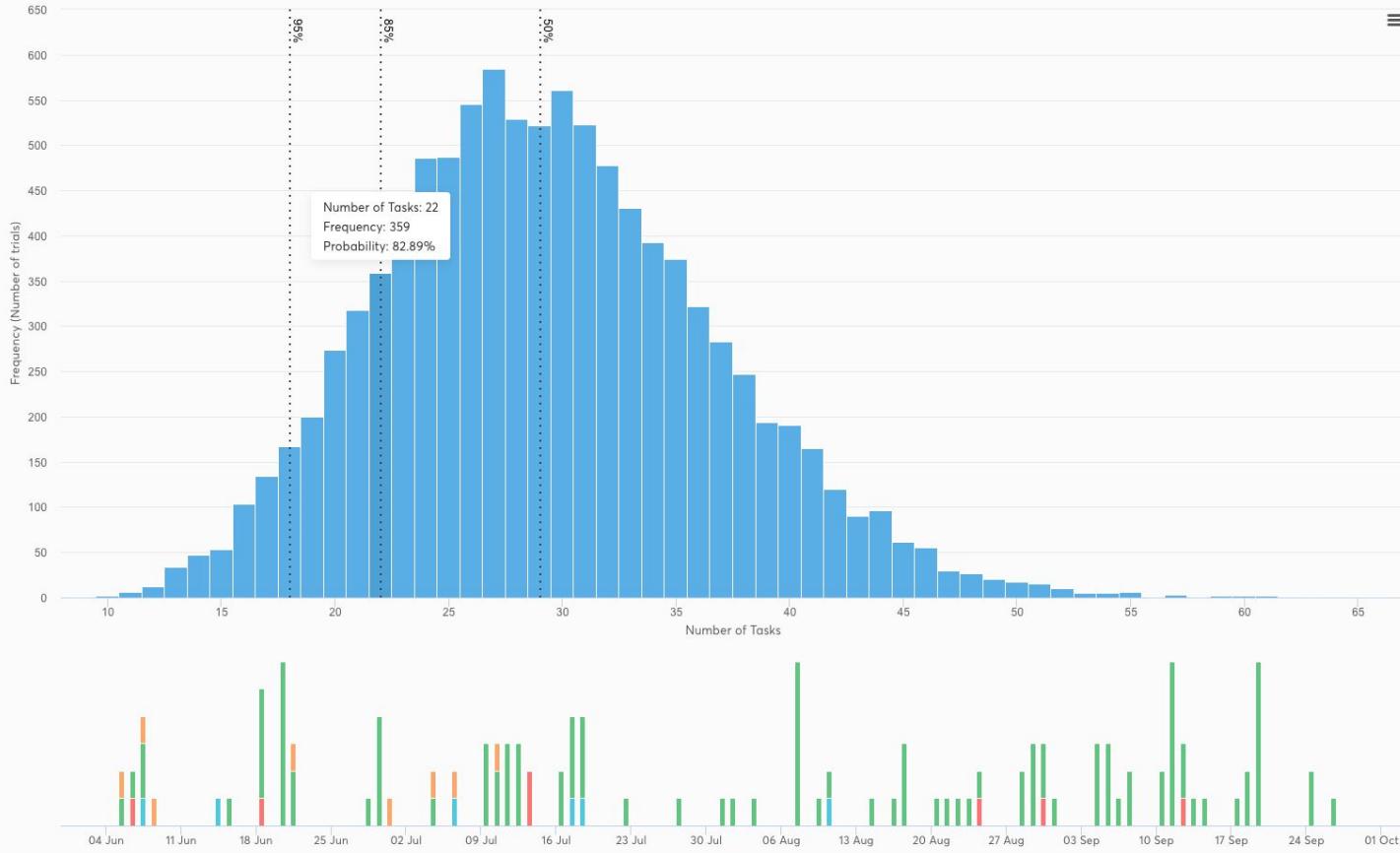
## Members

## 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

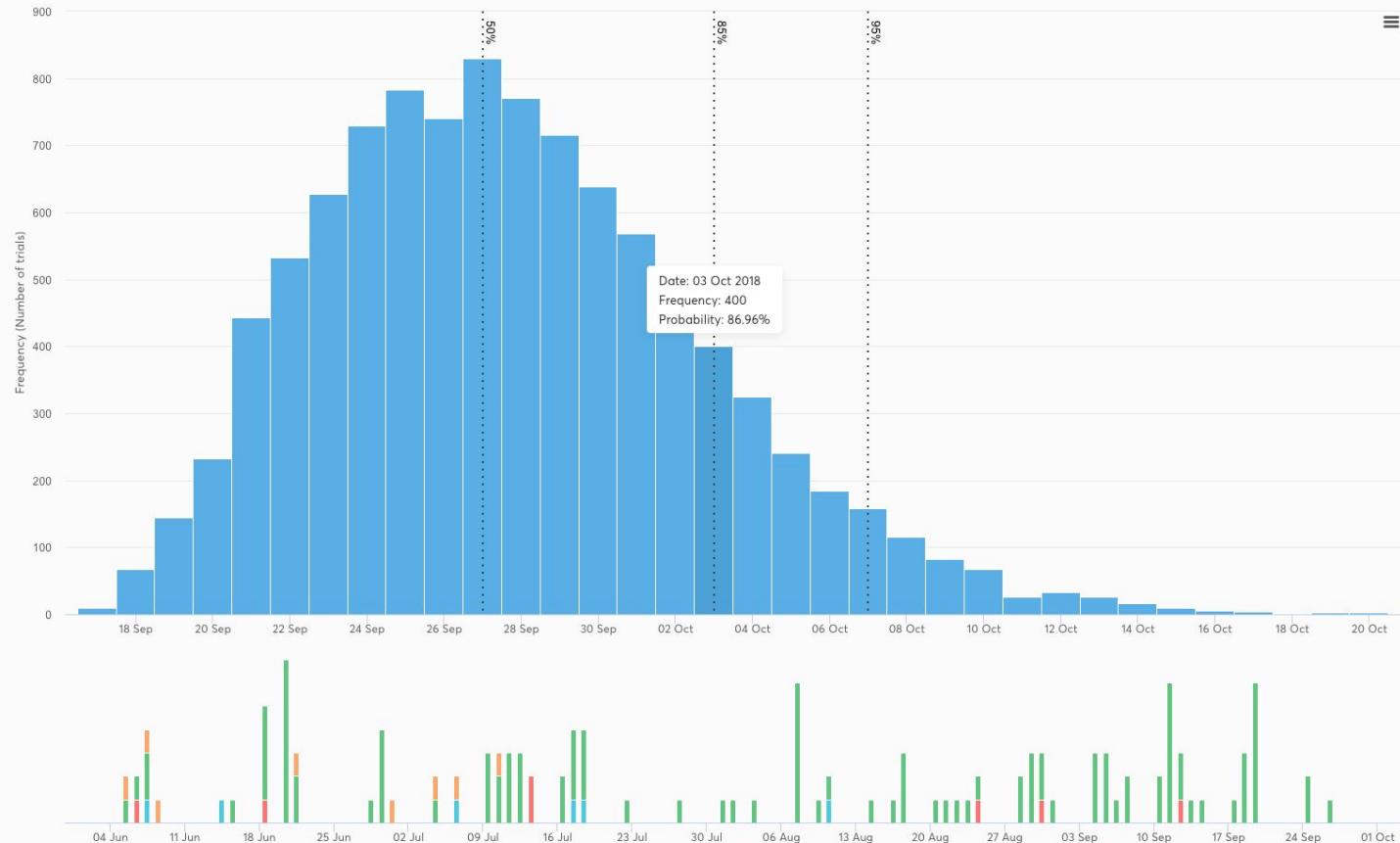
- Select all
- Cards without labels
- Expedite
- Fixed Delivery Date
- Intangible
- Standard

## 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

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation



# Have a strong Definition of Done

Absolute

Automated

Agreed with PO

# Have a strong Definition of Done

Absolute

Automated

Agreed with PO

Never lie about Done

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

Ineffective retrospectives



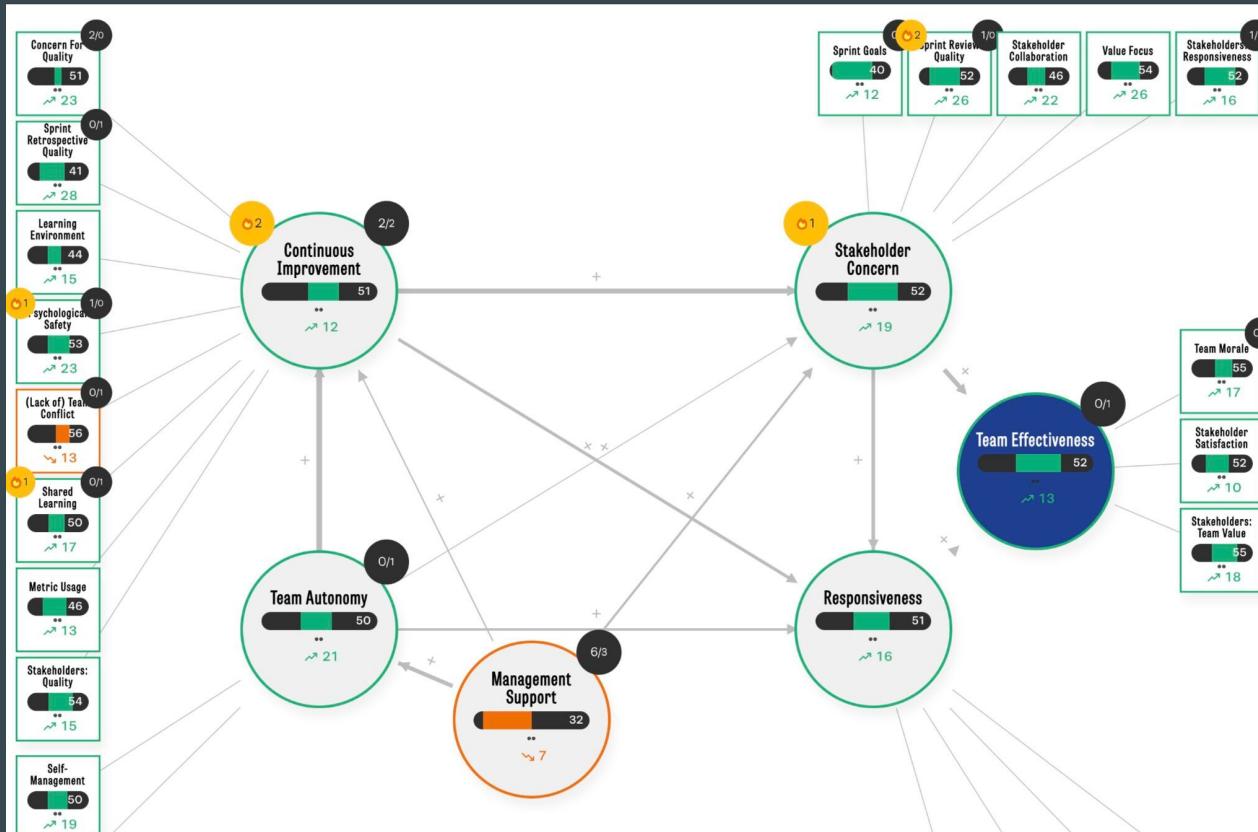
# Make retrospectives effective

Escalate what you cannot solve

# Make retrospectives effective

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)

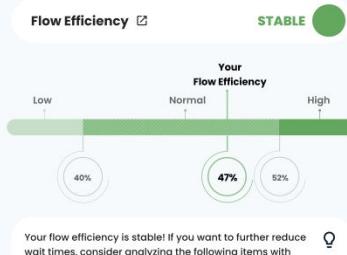
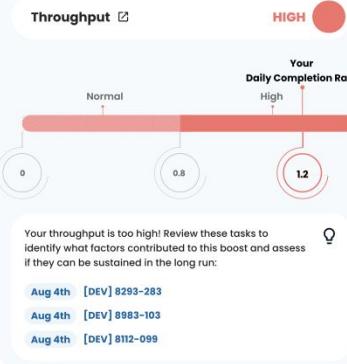
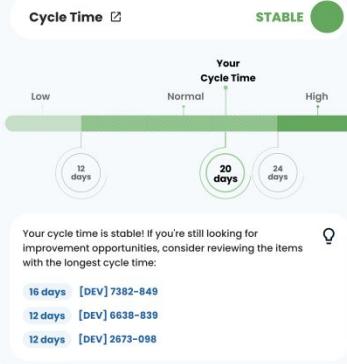
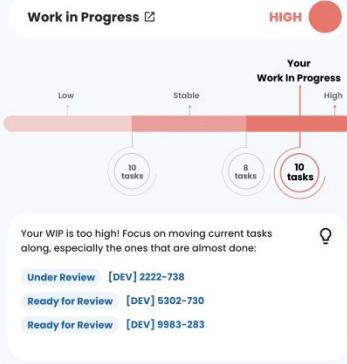
- Development
- Code Review
- Testing
- Deployment

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



[getnave.com](http://getnave.com)

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

Ineffective retrospectives

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

Ineffective retrospectives

Deal with technical debt

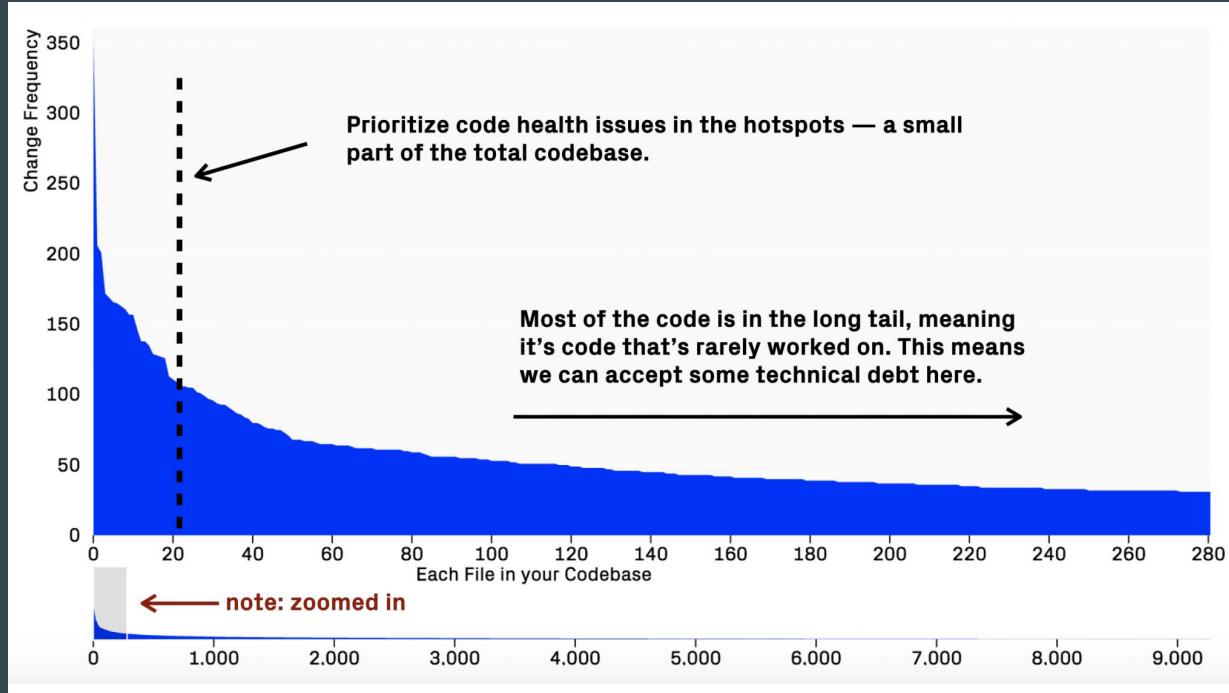


QUALITY



CONSISTENCY

# Prioritize changing files



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

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

Ineffective retrospectives

Deal with technical debt

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

Solve business problems

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

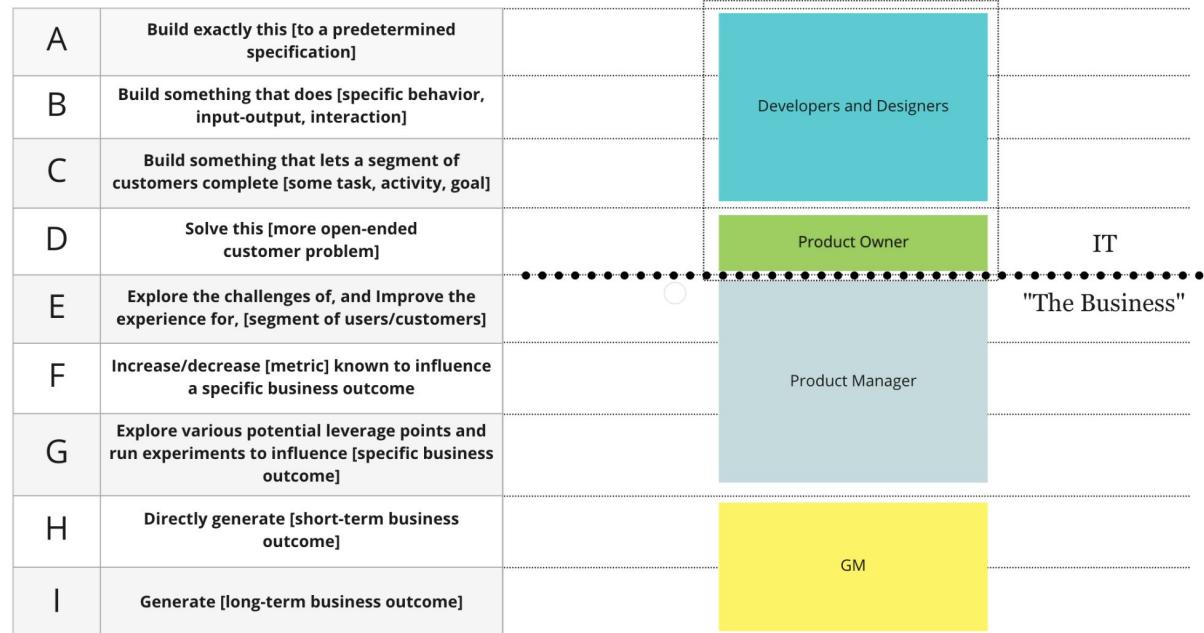
Ineffective retrospectives

Deal with technical debt

# 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>



# Solve business problems

Don't start here

Working software in the customers hands

Build a prototype

Get users in the room while designing

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

Solve business problems

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

Ineffective retrospectives

Deal with technical debt

To do

## Do this

Working tested software, every sprint

Know how the product is used

Do all work on the board

Have a strong Definition of Done

Solve business problems

## Avoid this

Experiment before complete methodology

You're not smarter than the customer

Estimation

Ineffective retrospectives

Consistent architecture

# 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!

• • •

Contact, blog & slides @  
[www.jakobbuis.nl](http://www.jakobbuis.nl)