



# Team projects - best practices

Wojciech Barczyński

# Wojciech Barczyński

- VP of Engineering at Spacelift
- Developer -> System Engineer -> Tech Lead
- [wojciech.barczynski@wroclaw.merito.pl](mailto:wojciech.barczynski@wroclaw.merito.pl)

# Best practices

1. Design docs,
2. UX/UI design,
3. Customer feedback,
4. Initial Planning and ETA,
5. Do work,
6. Working as a team,
7. Track progress + update every 1-2 weeks the plan,
8. Release checklists,
9. Alpha/Beta/RC/Release.

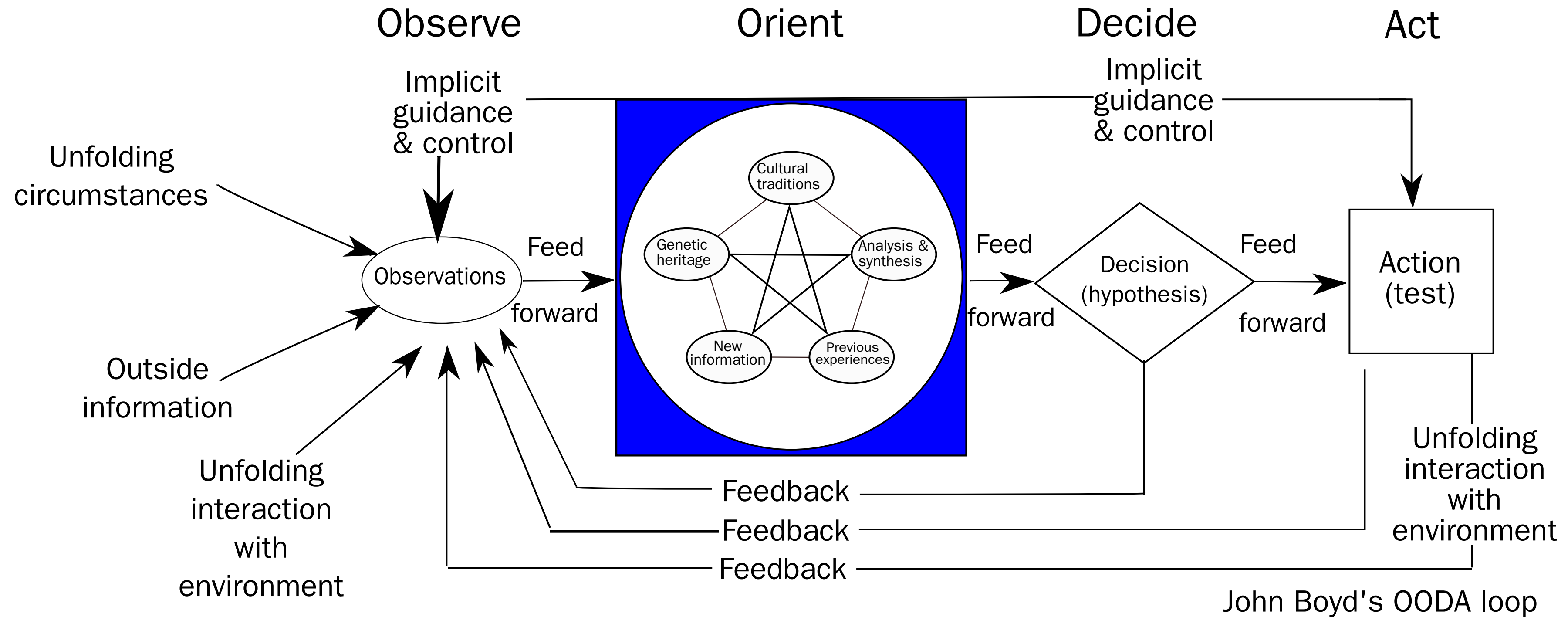
# Everything is an experiment & everything is iteration

Do-and-learn — driven by the team:

1. Retrospective per project  
(along with the team retros),
2. Keep and update project how-to / internal [wiki](#),
3. Pre-mortems,
4. Post-mortems,
5. Communication with the user / data.

# Best practices

## OODA:



# Engineering Culture

Strong foundation:

1. Safe environment (see [1](#) and [2](#)),
2. [Ownership](#),
3. Drum beat by the people who do the most of the work,
4. Pragmatism,
5. [Product teams](#)!

# Engineering Culture

Quality of communication:

- 2x yes, and...;
- Strong opinion, weak held (read: [original post](#), not-a-silver-bullet);
- [number of f-given](#);
- trust / you do not need to be involved in all the discussions.

# Consider

- Size of the team and the company,
- Experience of the engineers,
- Product maturity,
- etc.



# 0. Validate idea

Test/Validate the idea before writing code:

1. Mockups,
2. Code examples / API,
3. Share design.

You might also want to check Lean UX.

# 1. Design (engineering) docs

- With **DRI**,
- Thinking on paper,
- Opportunity to discuss and give feedback,
- **Example**,
- My favorite format: Google Docs and slack,
- A good place to track design decisions later.

## 2. UX/UI design

- for example, Figma,
- Good to setup what consitute a [good UX feedback](#).

# 3. Customer feedback

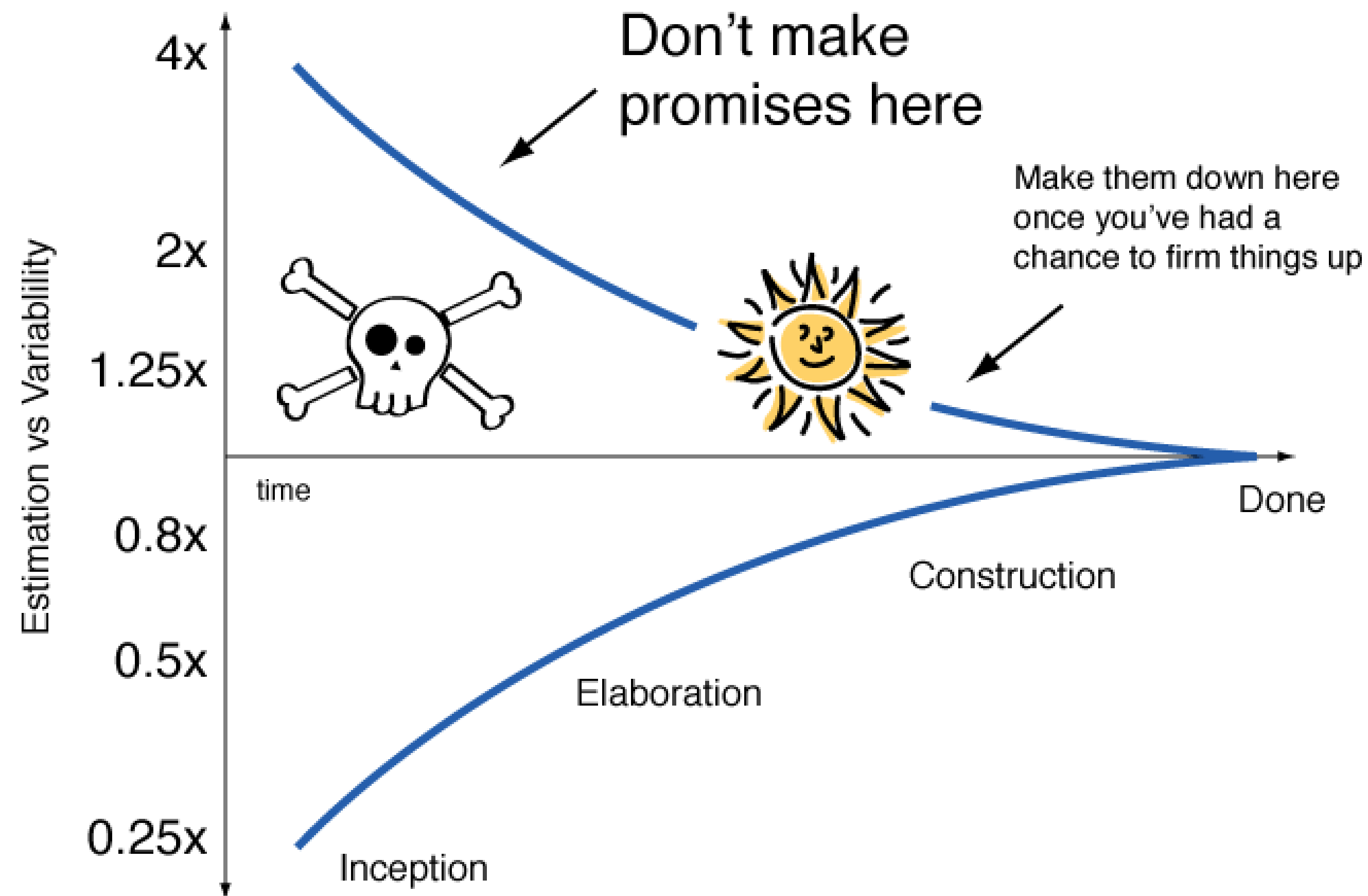
- Not only in the beginning, continuously reach out for feedback;
- Does not need to be a complex process (see [book](#));
- Design Doc, code example, ...;
- [Qualitative and Quantitative](#);
- Check tips from [YC](#).

## 4. Initial plan

- Planning is everything;
- The first iteration is a sketch;
- ETA;
- The further into the future, the larger the unit of measurement.

# 4. Initial plan

Be aware / communicate:



## 4. Initial plan

- Lay few alternatives scenarios out - completeness/quality/speed;
- Break up the project in many iterations, each iteration should be deliverable;
- Deliver -> patch/patch/patch / Tracer Bullet Dev;
- v1/v2.

## 4. Initial plan

- what we need first to learn, risk to address, key value to deliver;
- MVP?
- Uncharted territory? Spike or PoC.



## 5. Do work

- 1 week perspective,
- Focus and cutting Work-In-Progress,
- Important project/feature -> more focus time + take the best engineers.

Kanban, Scrum

## 6. Work as a team

1. Daily syncs (driven by the team);
2. Working in pairs+ or pair programming;
3. Swarming;
4. Do-ocracy / owning inputs.

## **6. Work as a team**

- Blocked or struggle -> let the team know.

## 7. Track progress & update the plan

Every 2 weeks:

1. Demo session that every project stream presents what they done,
2. Update the plan.

## 7. Track progress & update the plan

Demo:

- the best way to present the progress;
- Sync;
- Drum-beat;
- everything is demoable!

## 7. Track progress update the plan

Every 2 weeks;

- Add new tickets/task;
- Update plan and the ETA for v1, v2...;
- Be honest with yourself.

# **7. Track progress update the plan**

Live demo!

## 8. Release checklist

- Getting the business on the same page;
- Testing;
- Pre-mortem if the rollout is complex;
- Release details;
- Metrics.



## 8. Release checklists



- Recording know-how;
- Keep quality;
- Do not make unnecessary mistakes..
- ...

# 7. Release

Recommended:

- Feature flags.

# 7. Release

Remember, you have many options to release your software to manage customers' expectations:

- Alpha, (Closed) Beta, RC.

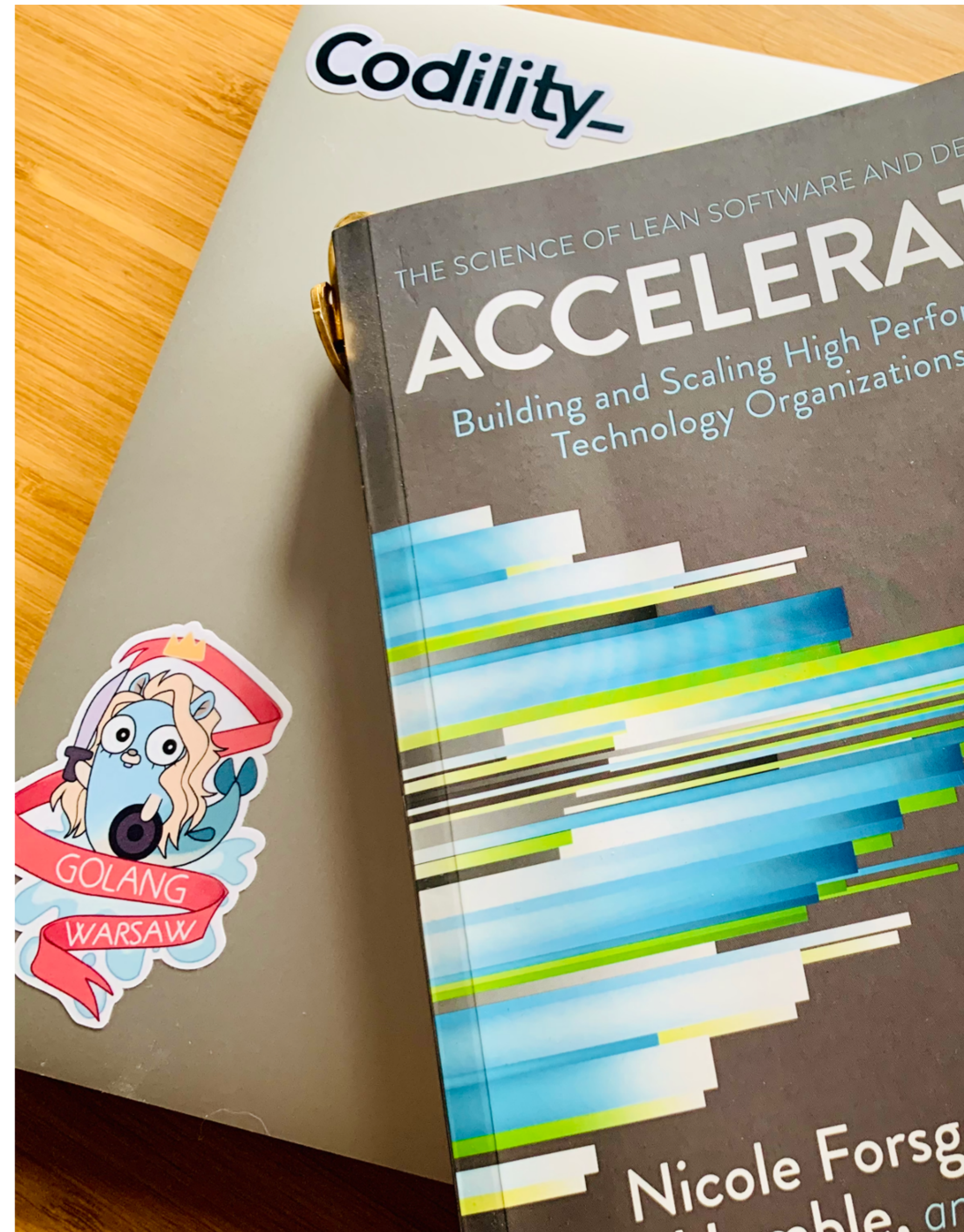
## 8. Metrics

- Start with **DORA**;
- Deployent frequency and lead time.



# High delivery performance

- Lead Time
- Deployment frequency
- Mean time to Recovery
- Change Fail Percent





## 9. Transparency and communication

- communicate only on public channels (slack),
- documents only on shared GDrives,
- starting work? Open a PR in draft.

# Metodyka?

- Kanban with weekly cycles following [prokanban.org](https://prokanban.org)  
- my preference,
- **SCRUM** with 1-week or max 2-weeks cycles.

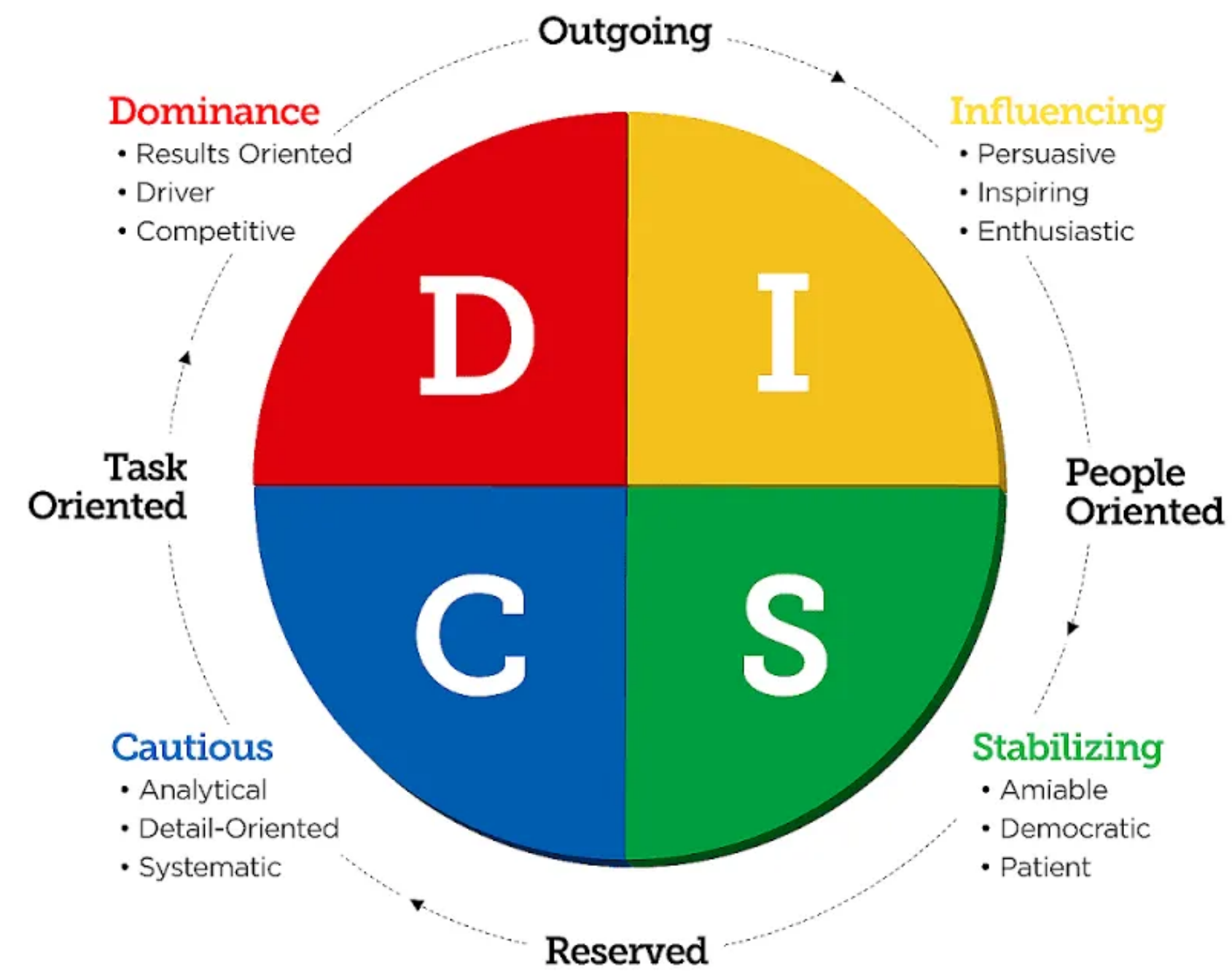
For larger companies, the concept of [flight levels](#) is worth exploring.

# Questions?





# Bonus



To learn more: [manager-tools podcast](#)

# Misc

Facilitate growth:

- Ed Batista - the art of self coaching,
- yes, and...,
- masterclass,
- or other initiatives.