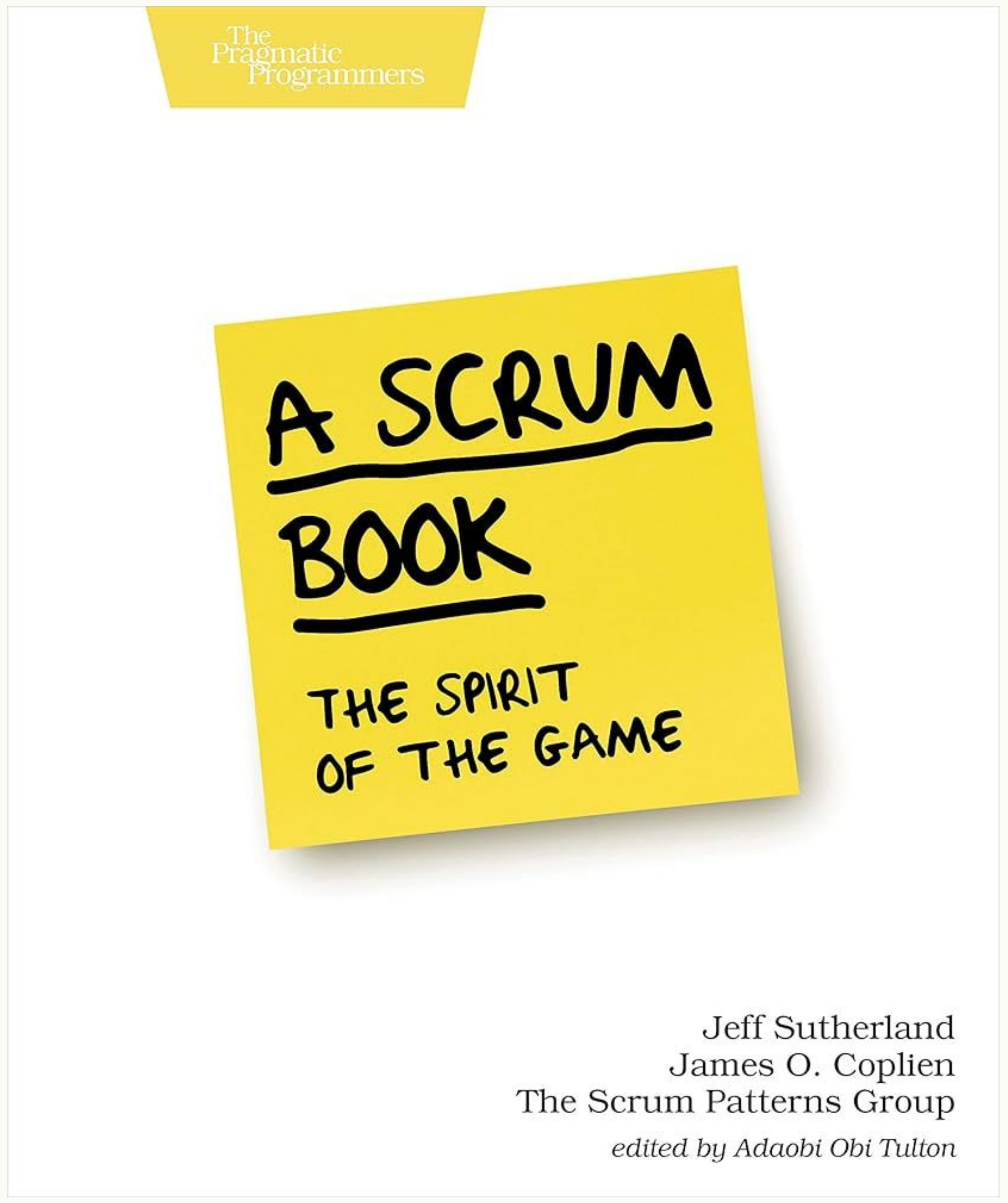
vincit

Doing Retrospectives

Veli-Pekka Eloranta

My journey with retrospectives



- Software developer
- Scrum Master
- Agile Coach
- Business director

“A retrospective”

What went well?

What didn’t go so well?

What to improve?

“A retrospective”

What went well?

- New team member is already contributing after two weeks

What didn’t go so well?

What to improve?

“A retrospective”

What went well?

- New team member is already contributing after two weeks

What didn’t go so well?

- In team area there is a server room. The fire alarm went off and rang for one day - it was disrupting

What to improve?

“A retrospective”

What didn't go so well?

- In team area there is a server room. The fire alarm went off and rang for one day - it was disrupting

What to improve?

- Next time the alarm goes off, we know the number, it is on a note in team area

Raises a question how to do it better?

- It should be fun
- No prisoners, no vacationers
- Everybody should be engaged and contributing
- Concrete actionable outcomes
- Changes should have impact (that lasts)

Why continuous improvement?

Because Agile manifesto says so?

From 12 Principles behind the Agile Manifesto:

“At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.”

IF YOU ARE NOT IMPROVING, YOU ARE DECAYING

- Improving by 1% doesn’t (usually) require a lot of effort
- Improving only by 1 % is often not notable

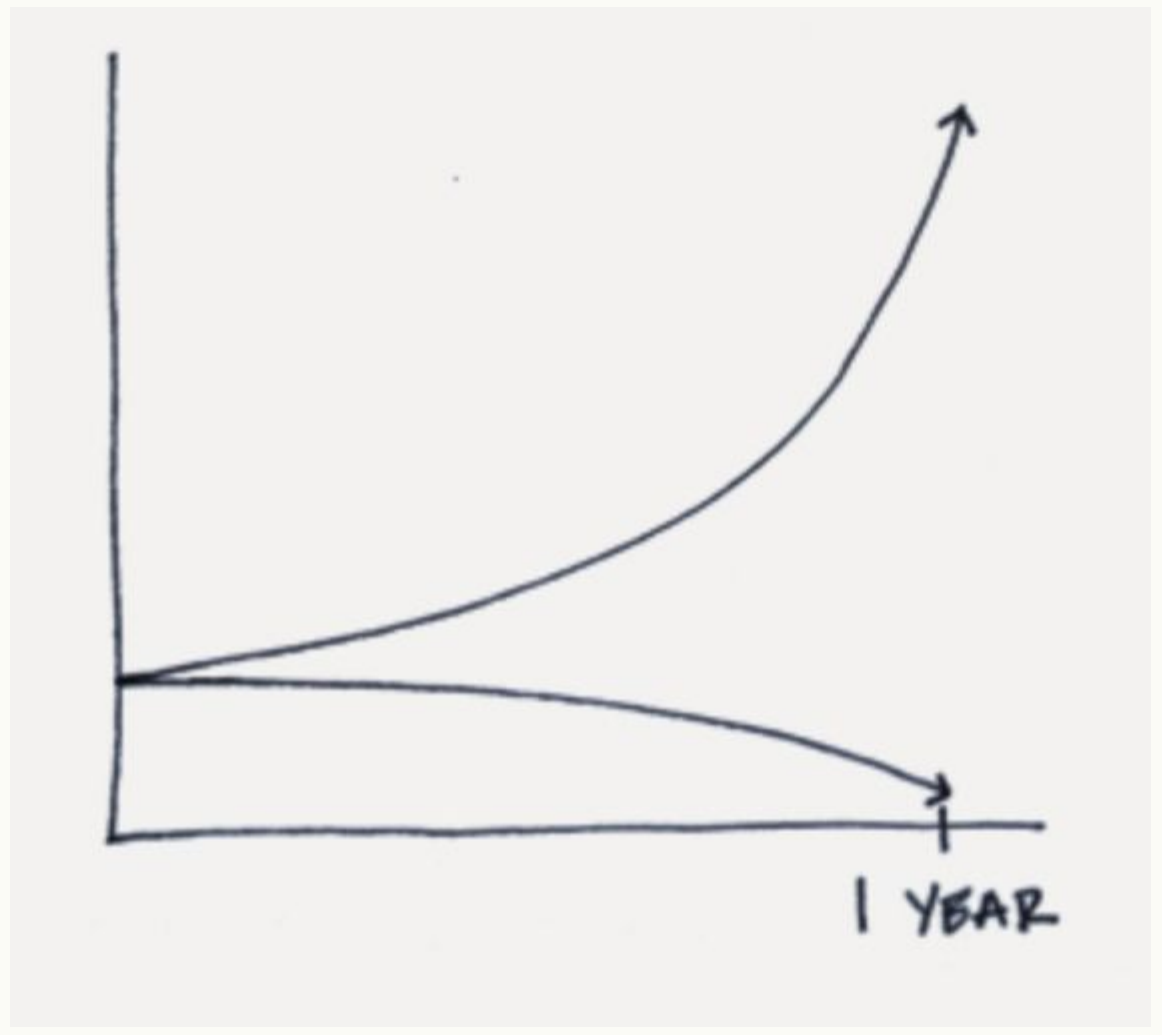
Math is simple

1 % better every day:

1.01ˆ365 = 37.783

1 % worse every day

0.99ˆ365 = 0.026



Continuous improvement

- World changes everyday – users' needs, markets, and technologies evolve, focus shifts.
- Continuous improvement ensures the team can adapt, rather than rigidly follow outdated plans.
- Usually in team work we deal with complex problems. In complex domain[1] no plan survives first contact with reality
- Frequent reflection and adjustment help detect small problems before they grow into unmanageable technical debt or product failures.

1) HBR article "A Leader's Framework for Decision Making" by David J. Snowden and Mary E. Boone

Continuous improvement

- World changes everyday – users' needs, markets, and technologies evolve, focus shifts.
- Continuous improvement ensures the team can adapt, rather than rigidly follow outdated plans.
- Usually in team work we deal with complex problems. In complex domain[1] no plan survives first contact with reality
- Frequent reflection and adjustment help detect small problems before they grow into unmanageable technical debt or product failures.

1) HBR article "A Leader's Framework for Decision Making" by David J. Snowden and Mary E. Boone

Continuous improvement

- Teams are social systems: people join, leave, change roles, and develop new dynamics
- Retrospectives and small process tweaks maintain psychological safety, collaboration quality, and motivation
- Continuous improvement spreads the cost of change over time and keep delivery flowing
- Continuous improvement fosters a culture of experimentation and learning
    - Teams are more likely to try new tools, architectures, or practices if improvement is part of the norm rather than a disruption

INDIVIDUAL VS. TEAM

- When working alone, you can improve your process, tools, etc anytime
- When working in a team, it might be a good idea to discuss the improvements together
- Easiest when there is regular time reserved to discuss improvements => **Retrospective**

Anatomy of high-impact retrospective

TRUST & PSYCHOLOGICAL SAFETY

- Prerequisite for successful retrospectives
- If there is "boss" who causes people not to say things aloud, remove the shade
- Needs to be built consciously
- Takes time
- You need to cherish trust
    - "Half-time of trust is 6 weeks"

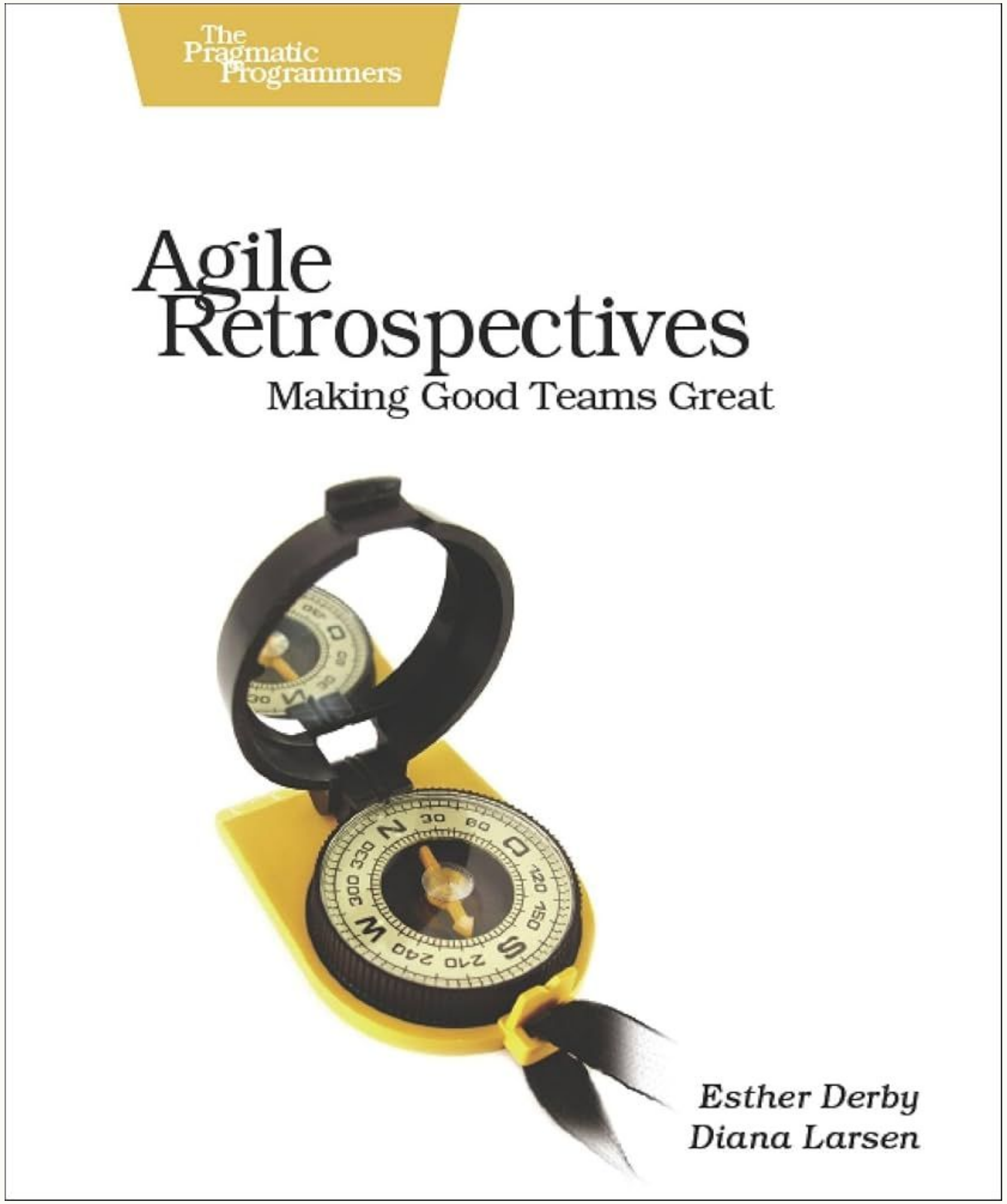
PRIME DIRECTIVE

"Regardless of what we discover, we understand and truly believe that everyone did the best job they could, given what they knew at the time, their skills and abilities, the resources available, and the situation at hand."

- Norm Kerth, Project Retrospectives: A Handbook for Team Review

Agile Retrospectives

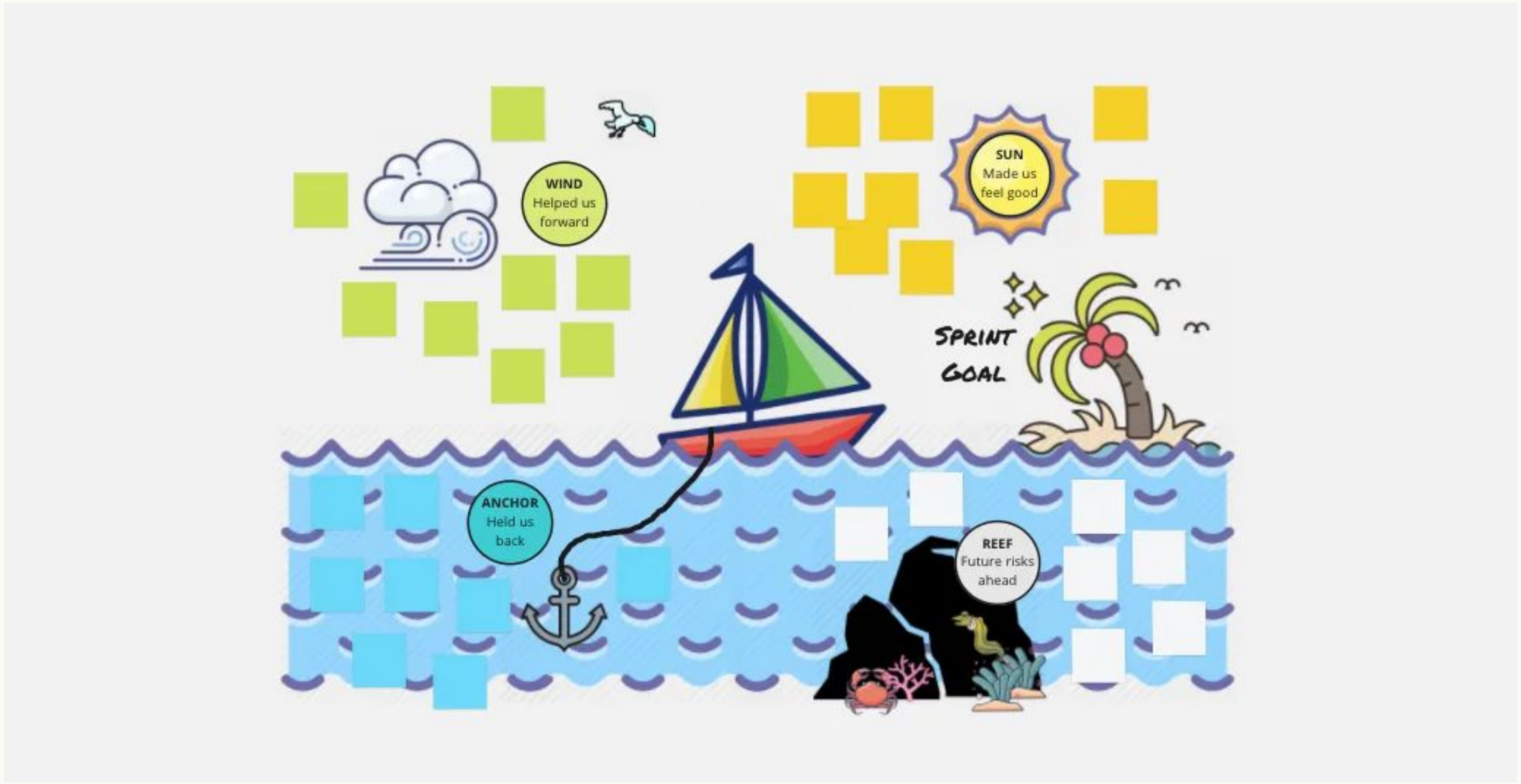
- Good starting point for learning retrospectives



STRUCTURE OF RETROSPECTIVE

1. Set the stage
2. Gather data
3. Generate insights
4. Decide what to do
5. Close the retrospective

Gather data - example



Gather data - example

Glad Sad Mad Worried

SMART GOAL

S*pecific* – target a specific area for improvement

M*easurable* – quantify or at least suggest an indicator of progress

A*ssignable* – specify who will do it

R*ealistic* – state what results can realistically be achieved, given available resources

T*ime-related* – specify when the result(s) can be achieved

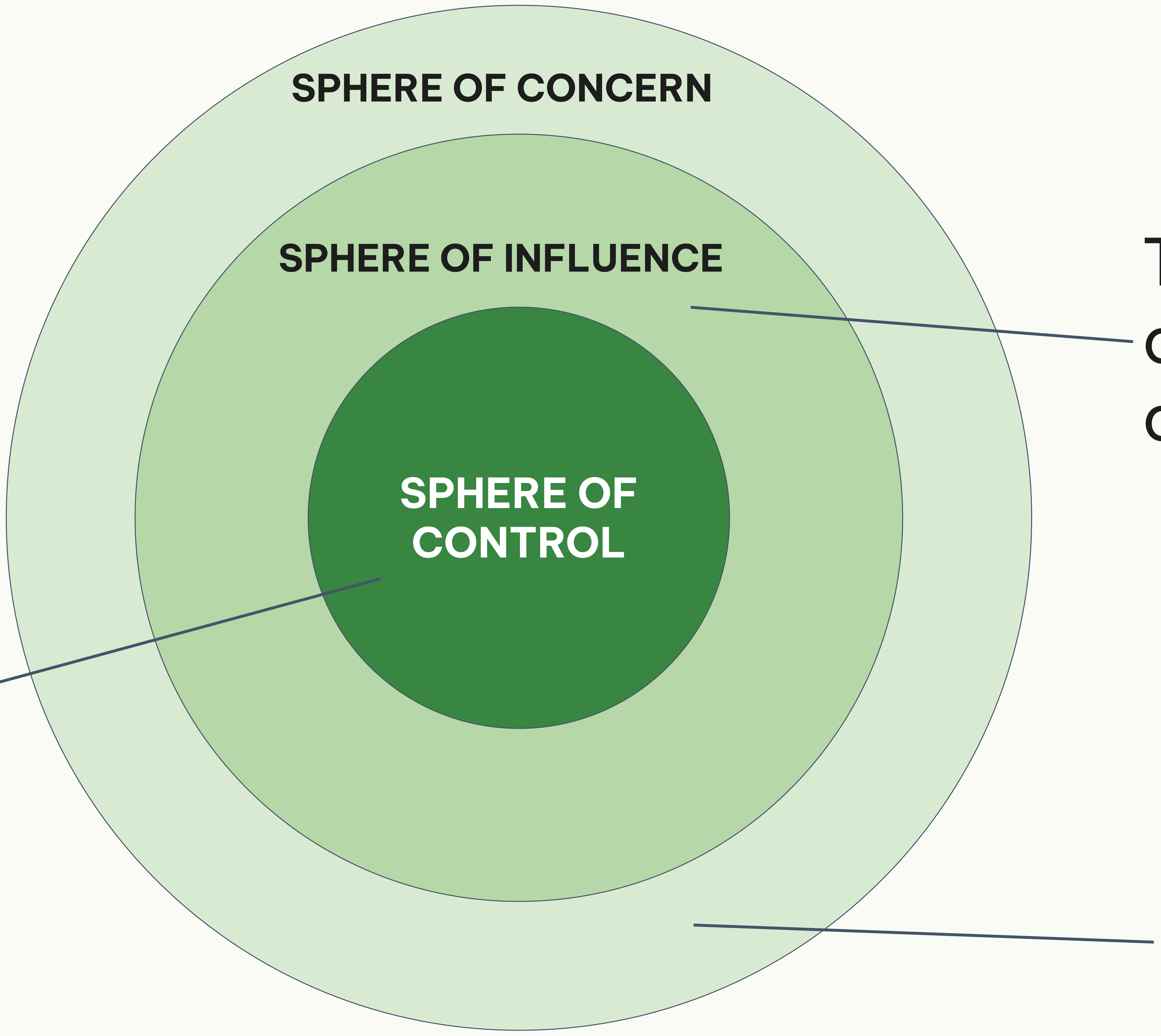
SMART GOAL EXAMPLE

“We should do more pair programming”

vs.

“Our goal is to do pair programming at least 5 hours a day starting next Monday. We will switch pairs daily and Scrum Master will create a schedule with pairs named and post it on the wall in the team’s room. Our next retrospective is dedicated to discussion on our experiences on pair programming and we will decide if this practice should be continued or not. We can also make changes to the agreed practices in the next retrospective.”

SPHERE OF INFLUENCE

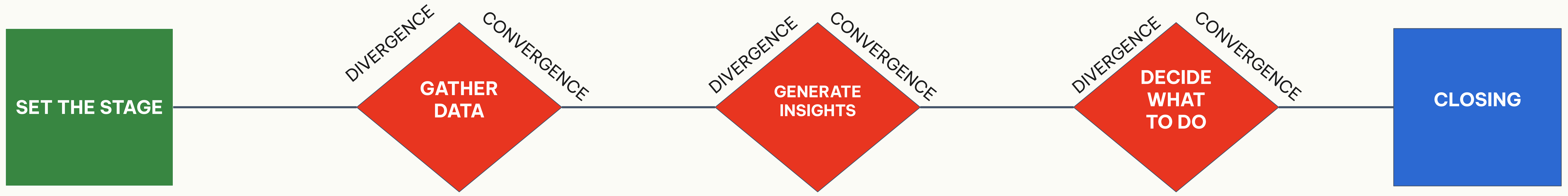


Things we can't change, but we can influence

Things we own, things we can change

Things we can't affect at all, need to take them as is

CONVERGENCE & DIVERGENCE IN RETROS



E.g. Glad/Sad/Mad (divergence) and then sorting and grouping (convergence)

SLOWING DOWN THE CONVERGENCE

- People love divergence phases and tend to hate convergence
- It is important to slow down convergence and explore ideas
  - People tend to jump in to the solutions and conclusions
- Introverts need more time to get on board to chip in their ideas
- Different techniques, e.g. me-we-us

EXAMPLE PLAN FOR 2 HOUR SPRINT RETROSPECTIVE

Focus on convergence => Takes time!!

1. Set the stage (5-10 minutes)
2. Gather data (20-30 minutes)
3. Generate insights (30 minutes)
4. Decide what to do (30-40 minutes)
5. Closing the retro (5-10 minutes)

FACILITATOR

- Retrospective needs a facilitator
- Can be person outside the team
  - Does not need to know anything about the contents
- Can be a team member
  - Should be neutral
  - Should not contribute to the contents

LIVE, REMOTE OR HYBRID?

- Live and remote works well
- Hybrid is the hardest

THE 5 STAGES OF A REMOTE RETROSPECTIVE
CLICK.
1. SET THE STAGE
HI EVERYONE. WELCOME TO OUR RETRO...
Hi!
HELLO
Hi
HELLO
Hi.
2. GATHER DATA
UM... IS THAT... IS THAT YOUR CAT'S ARSE?
HA HA!
HA!
HA!
HA! HA!
3. GENERATE INSIGHTS
OK... WELL.. LET'S DO "MAD, SAD, GLAD"
HELLO?
1, 2, 3...
I CAN'T HEAR...
HELLO! TESTING... HELLO.
4. DECIDE WHAT TO DO
HELLO? CAN YOU HEAR ME?
I'M GOING TO RESTART
I CAN SEE YOU BUT I CAN'T HEAR YOU
THIS IS STUPID. IM GOING.
5. CLOSE THE RETROSPECTIVE
SORRY I'M LATE. ARE WE DOING THE RETRO?
BROKENBAYSOFTWARE.CO

Vincit

FOOLS WITH TOOLS

- Engineer mind wants to automate with tools. Tools are ok as long as they serve the greater cause
- Whiteboards + Post its ❤️
- For remote sessions, Miro is super good.
- For lazy (remote) teams Echometer is superb but costs
- You can have a retrospective with Trello, Google Spreadsheet or free tools you find from Google
- Tools make it easier to further process the results

TEAM SIZE

- Retrospectives work well for teams from 3 to 9 people
- Larger groups needs to be divided and then results brough together
- Larger the group, more time you will need
- In large groups, you should avoid open discussions as it takes a lot of time

PITFALLS

- “It’s just a meeting at the end of a sprint”
- “We already know what’s wrong”
- “We are doing improvements all the time even without retros?”
- Irregular retrospectives (e.g. once a year, just before release)
- Do it always using the same methods
  => It will get boring

AFTER THE RETRO

- Document the results, they already should have been assigned to someone who takes care that the team makes progress

- Follow up at the beginning of the next retro
  - If something is not done, analyze why
  - Often something is not done, because
    - What to do wasn’t clear
    - World changed and the thing became irrelevant
    - We didn’t analyze very well in the last retro what should actually be done

NOT ONLY FOR SOFTWARE TEAMS

- Continuous improvement should be done on all levels at an organization - not only in software teams
- At Vincit, our operational model is built around the core of continuous improvement
    - We require all functions to do continuous improvement
- Most of the teams have retrospectives, including sales and leadership team
- My leadership team has retros every 6 weeks

AFTER 100 THE FIRST RETROS AT VINCIT'S SOFTWARE TEAMS

- Small experiment - data from 100 retros at Vincit
- Summarized which are the common pain points our software teams are having
- Top 3 most often repeating problems were:

  1. Lack of vision
  2. Unclear specifications / user stories
  3. No feedback from real users

- Might be a good source for a publication :)

TO SUMMARIZE

- Retrospectives are not about the meeting itself, but about the feedback loop
- Time spent in retrospective is an investment to learning
- If the team has a coach or lead, they should keep an eye on what the team needs to unlock the potential and plan the retrospective accordingly.
- Use data about the process as an input for retrospective
- Have metrics to follow if the desired impact has been reached

THANK YOU

Veli-Pekka Eloranta
Business Director

+358 44 557 0591
firstname.lastname@vincit.com

Vincit