A teamwork effectiveness model for agile software development

Guest lecture, CMU 17-313: Foundations of Software Engineering, 31st October 2023

Torgeir Dingsøyr
Professor, Department of Computer Science
Norwegian University of Science and Technology
Adjunct Chief Scientist, SimulaMet

NTNU – Trondheim
Norwegian University of
Science and Technology

simulamet

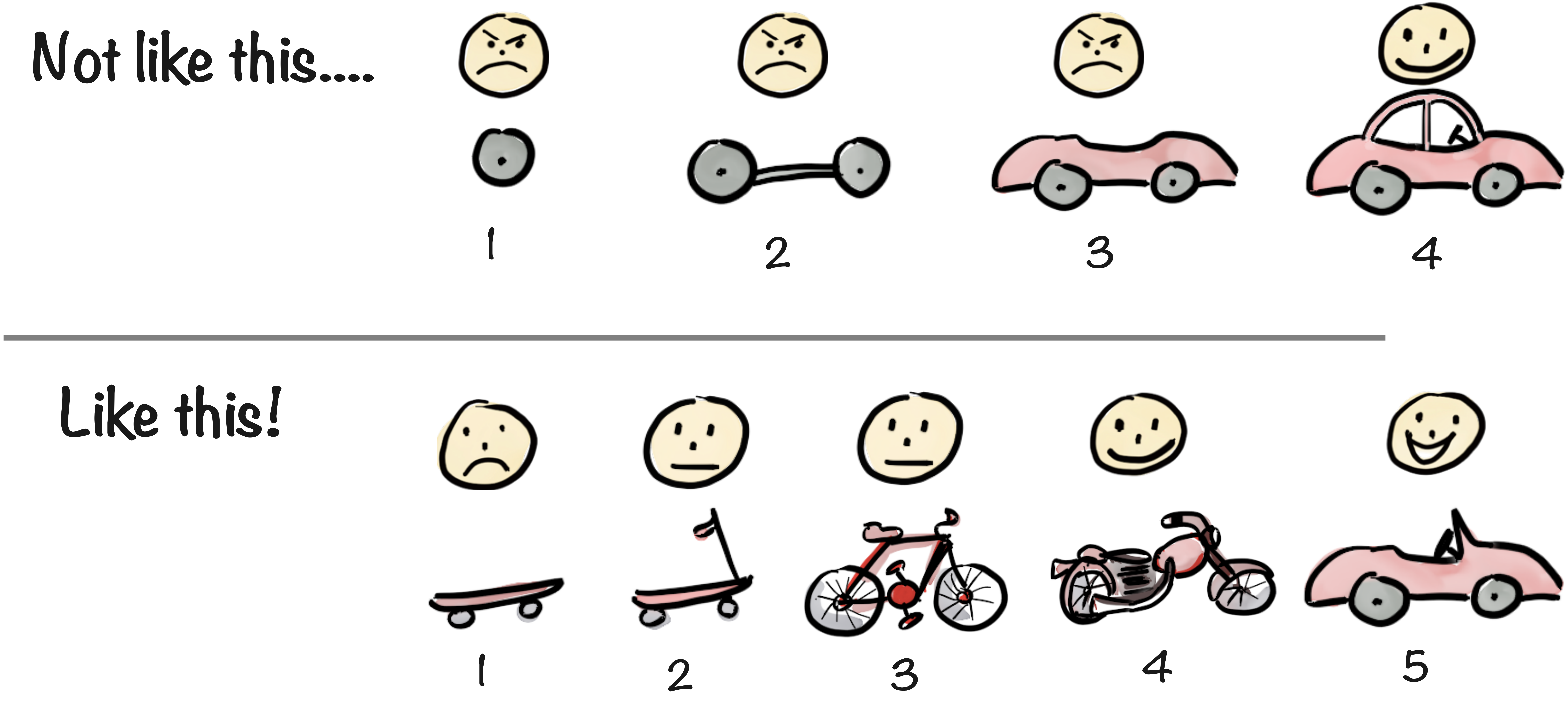
Agile software development

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

Manifesto for Agile Software Development (2001)

https://agilemanifesto.org/ Project Management Institute and Agile Alliance, Agile Practice Guide: Project Management Institute, 2017.

Agile software development



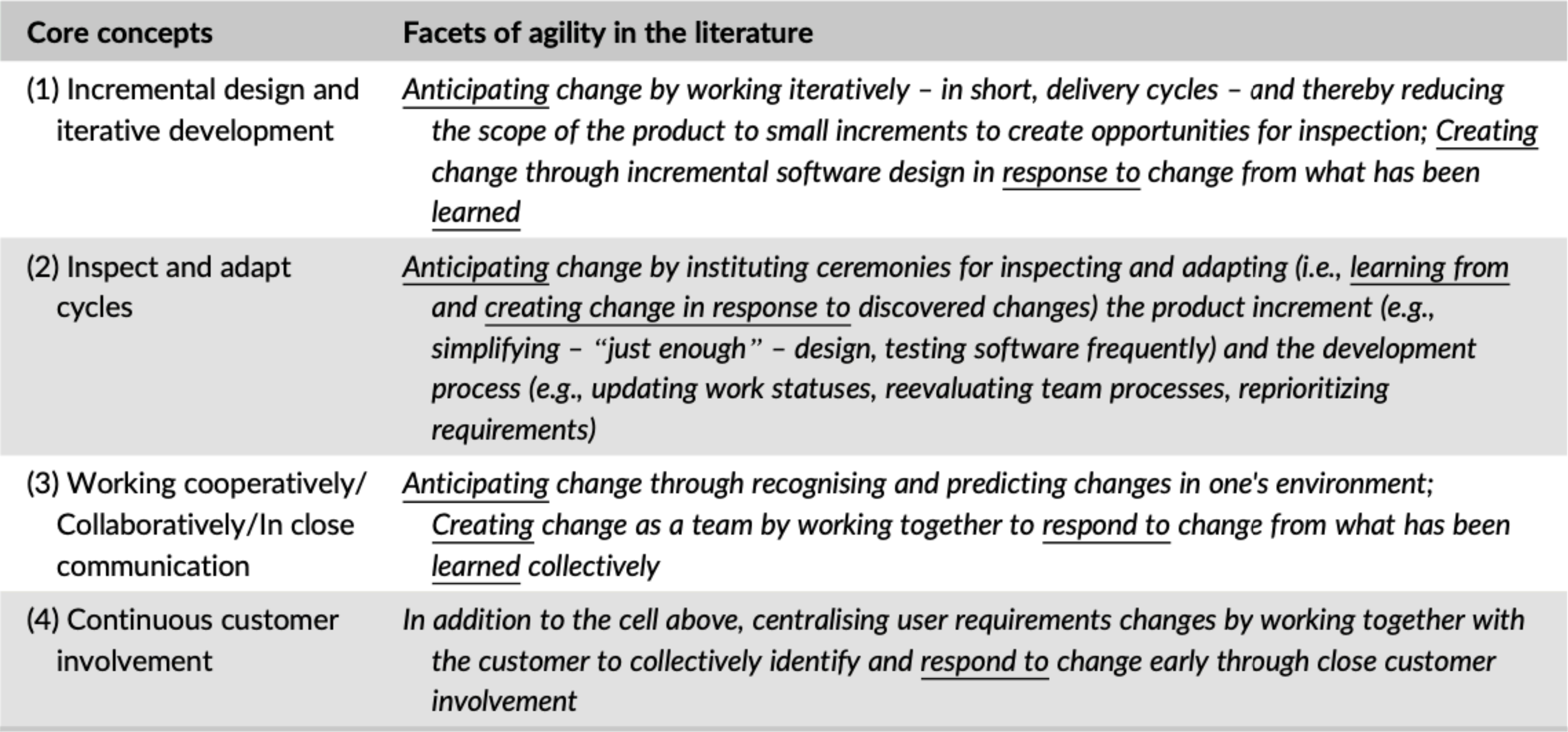
Kilde: Henrik Kniberg, 'How do you know that your product works?': Colombo Agile Conference 2014.
https://www.slideshare.net/ColomboCampsCommunity/how-do-you-know-that-your-product-works-henrik-kniberg-colombo-agile-conference-2014

On development method

“Now she speaks about method. She thinks all should follow a strict method based on structured programming. Instead, it is full anarchy, the programmers do whatever they want, everyone do what they want without considering anyone else, there is no collaboration, no holistic thinking, no harmony”

“Whatever”, Michel Houellebecq (1994)

Core concepts in agile development



Core concepts	Facets of agility in the literature
(1) Incremental design and iterative development	*Anticipating change by working iteratively – in short, delivery cycles – and thereby reducing the scope of the product to small increments to create opportunities for inspection; Creating change through incremental software design in response to change from what has been learned*
(2) Inspect and adapt cycles	*Anticipating change by instituting ceremonies for inspecting and adapting (i.e., learning from and creating change in response to discovered changes) the product increment (e.g., simplifying – "just enough" – design, testing software frequently) and the development process (e.g., updating work statuses, reevaluating team processes, reprioritizing requirements)*
(3) Working cooperatively/ Collaboratively/In close communication	*Anticipating change through recognising and predicting changes in one's environment; Creating change as a team by working together to respond to change from what has been learned collectively*
(4) Continuous customer involvement	*In addition to the cell above, centralising user requirements changes by working together with the customer to collectively identify and respond to change early through close customer involvement*

Baham, C. and Hirschheim, R., "Issues, challenges, and a proposed theoretical core of agile software development research," *Information Systems Journal*, 2021. https://doi.org/10.1111/isj.12336

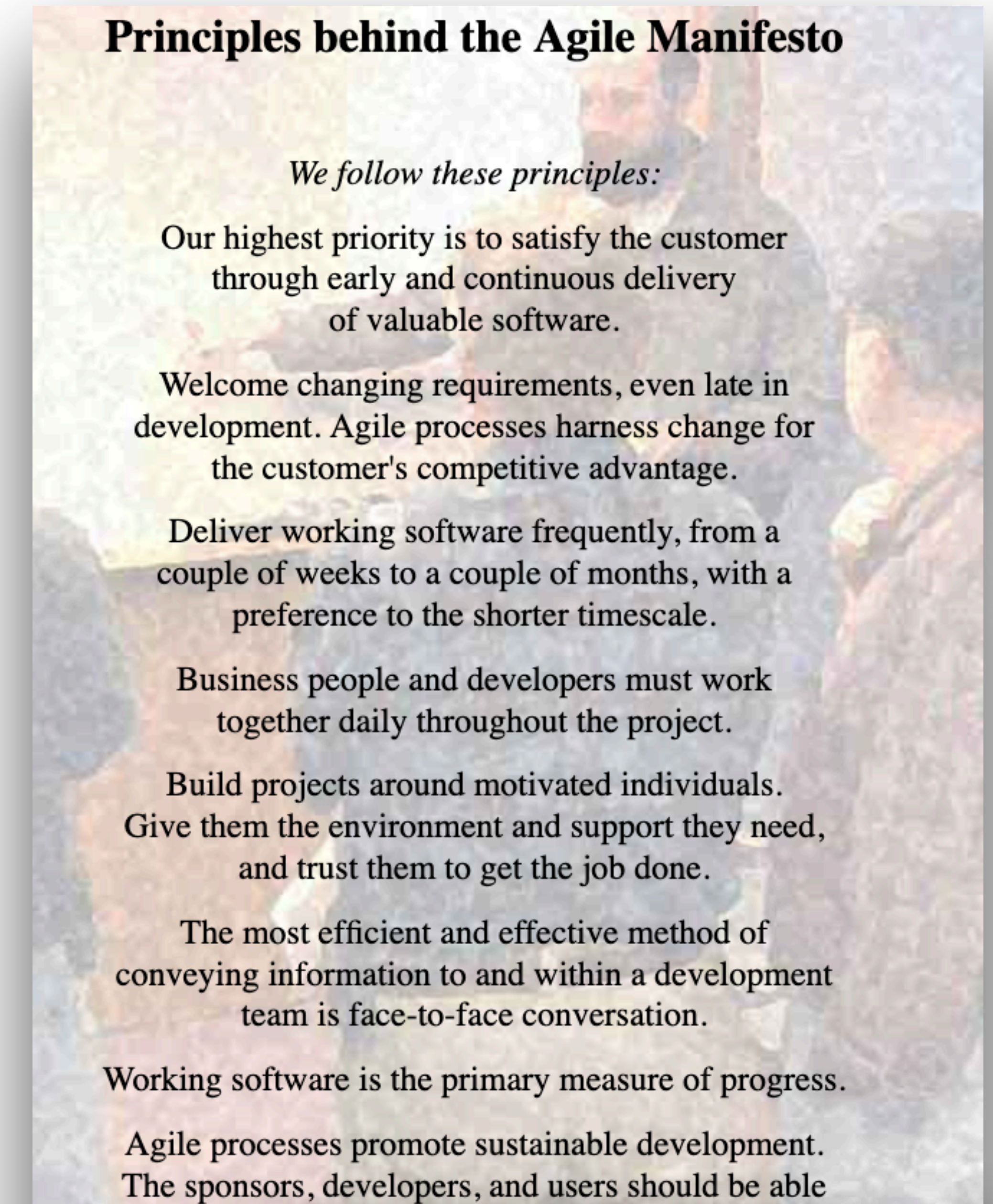
What is a team?

“(a) composed of two or more individuals, (b) who exist to perform organizationally relevant tasks, (c) share one or more common goals, (d) interact socially, (e) exhibit task interdependencies (i.e., workflow, goals, outcomes), (f) maintain and manage boundaries, and (g) are embedded in an organizational context that sets boundaries, constrains the team, and influences exchanges with other units in the broader entity”

Kozlowski and Bell (2012)

Kozlowski, S. W., & Bell, B. S. (2012). Work groups and teams in organizations. *Handbook of Psychology, Second Edition, 12*.

Advice on software development teamwork



Principles behind the Agile Manifesto

We follow these principles:

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

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

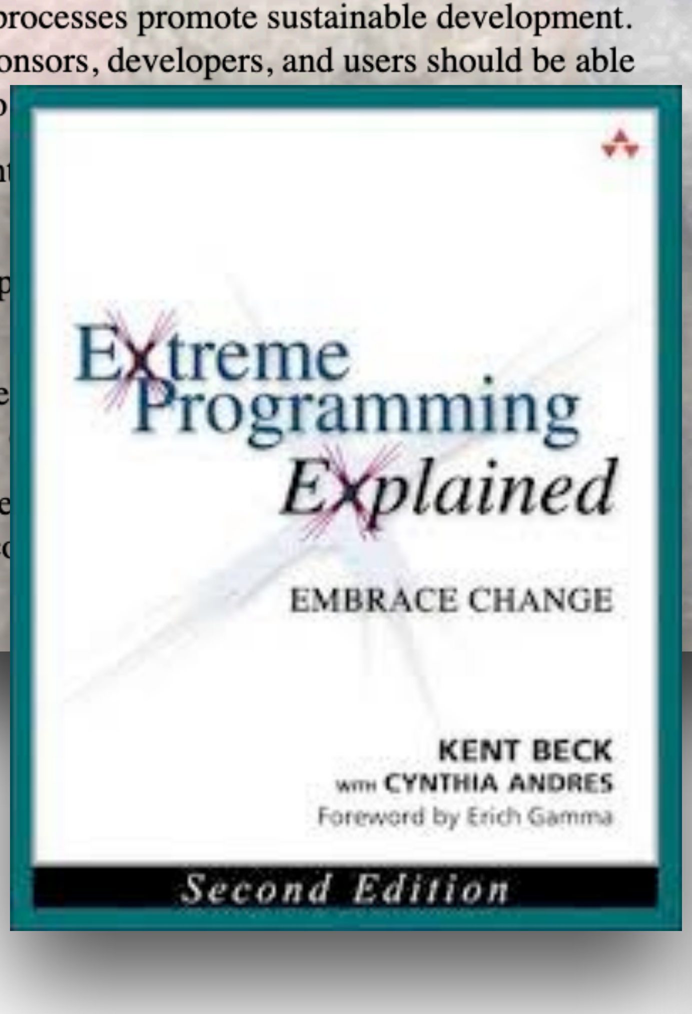
Agile processes promote sustainable development. The sponsors, developers, and users should be able to

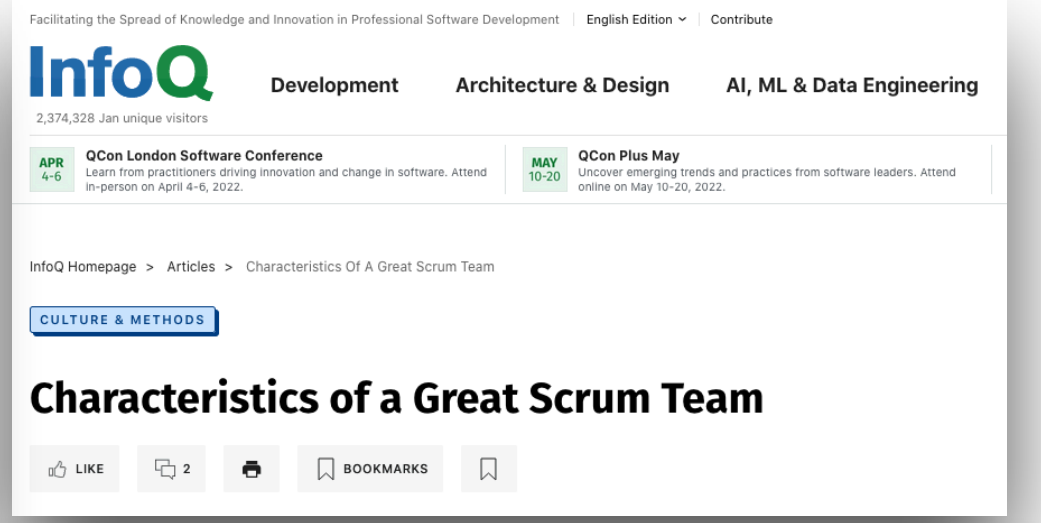
Con

Simp

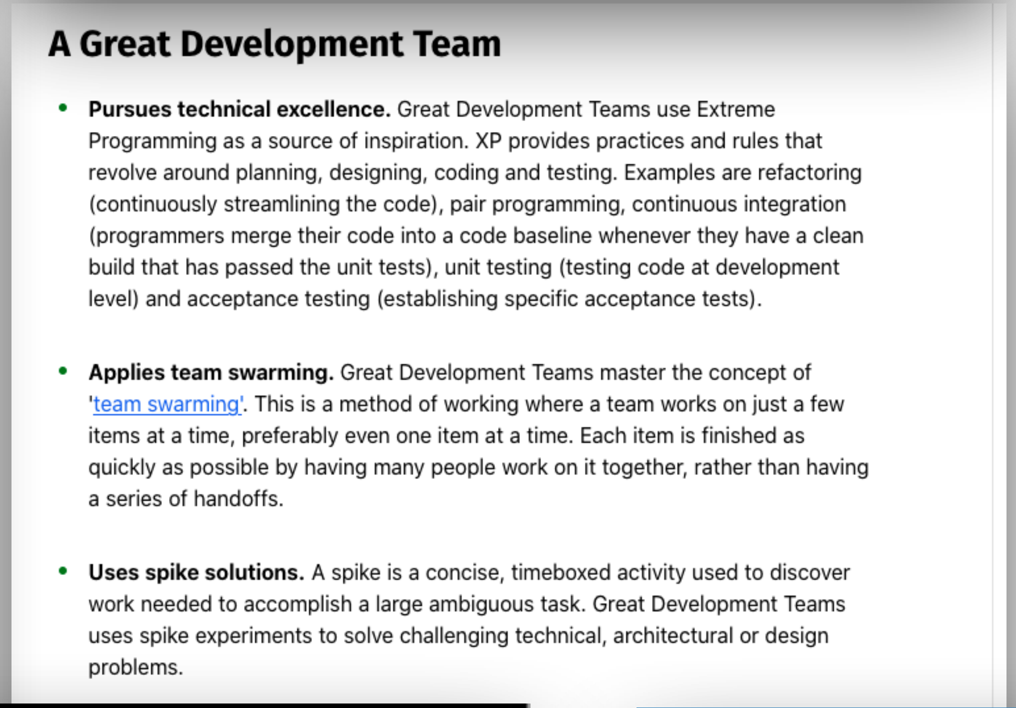
The be

At re
to bec




Facilitating the Spread of Knowledge and Innovation in Professional Software Development
English Edition
Contribute
InfoQ
Development
Architecture & Design
AI, ML & Data Engineering
2,374,328 Jan unique visitors
APR 4-6
QCon London Software Conference
Learn from practitioners driving innovation and change in software. Attend in-person on April 4-6, 2022.
MAY 10-20
QCon Plus May
Uncover emerging trends and practices from software leaders. Attend online on May 10-20, 2022.
InfoQ Homepage > Articles > Characteristics Of A Great Scrum Team
CULTURE & METHODS
LIKE
2
BOOKMARKS

Characteristics of a Great Scrum Team



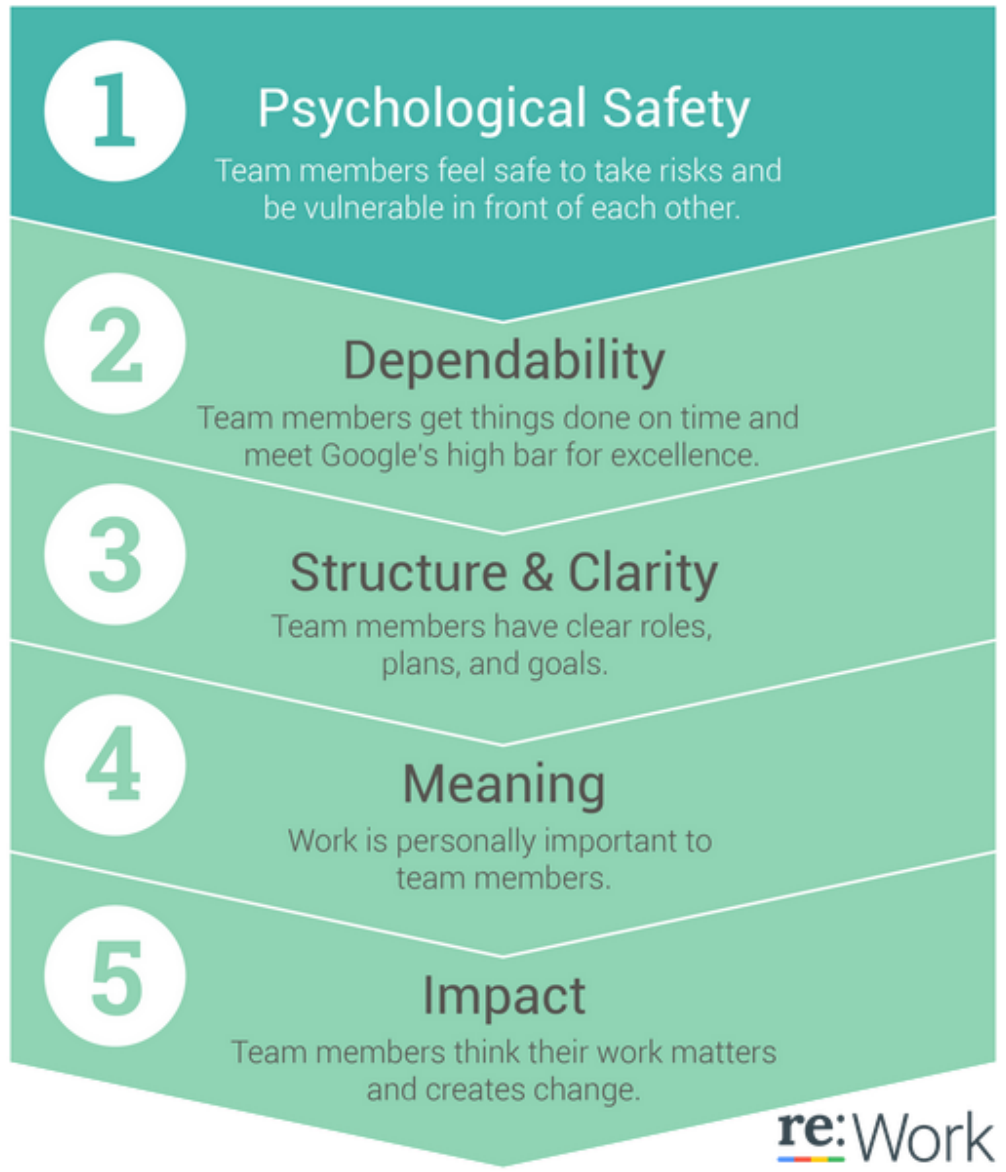
A Great Development Team

- **Pursues technical excellence.** Great Development Teams use Extreme Programming as a source of inspiration. XP provides practices and rules that revolve around planning, designing, coding and testing. Examples are refactoring (continuously streamlining the code), pair programming, continuous integration (programmers merge their code into a code baseline whenever they have a clean build that has passed the unit tests), unit testing (testing code at development level) and acceptance testing (establishing specific acceptance tests).

- **Applies team swarming.** Great Development Teams master the concept of 'team swarming'. This is a method of working where a team works on just a few items at a time, preferably even one item at a time. Each item is finished as quickly as possible by having many people work on it together, rather than having a series of handoffs.

- **Uses spike solutions.** A spike is a concise, timeboxed activity used to discover work needed to accomplish a large ambiguous task. Great Development Teams uses spike experiments to solve challenging technical, architectural or design problems.

at Developr
stand that t
nable devel
Owner is r
efine it.



1 Psychological Safety
Team members feel safe to take risks and be vulnerable in front of each other.

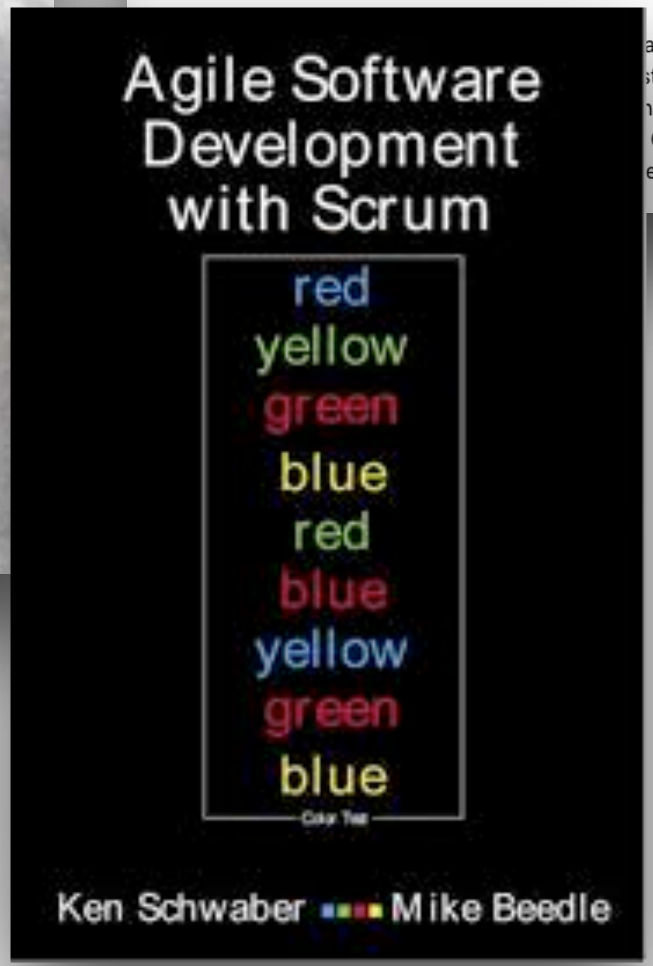
2 Dependability
Team members get things done on time and meet Google's high bar for excellence.

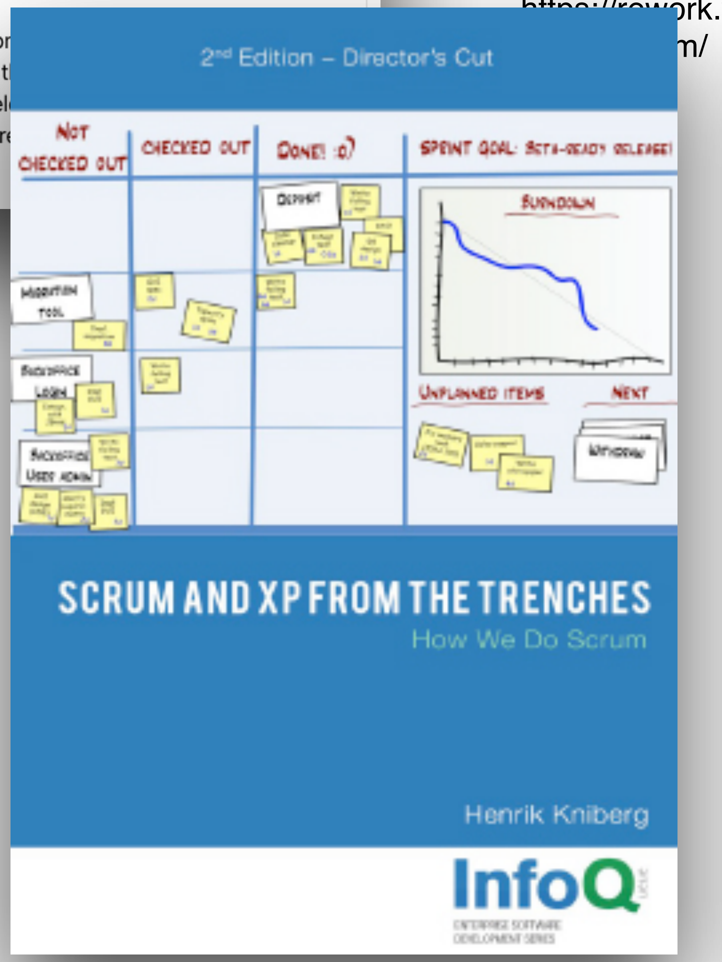
3 Structure & Clarity
Team members have clear roles, plans, and goals.

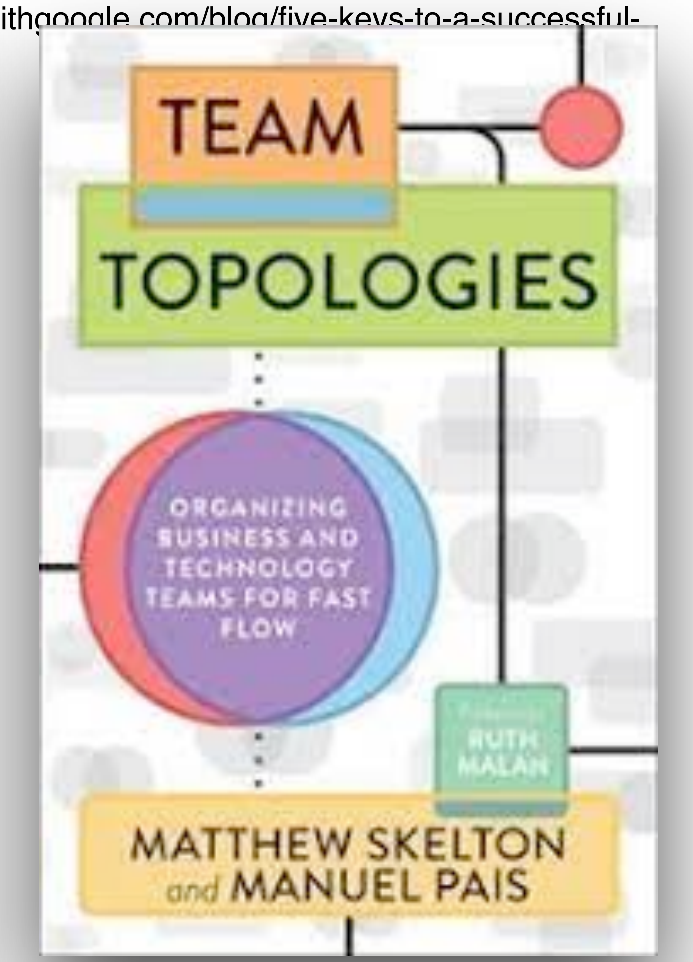
4 Meaning
Work is personally important to team members.

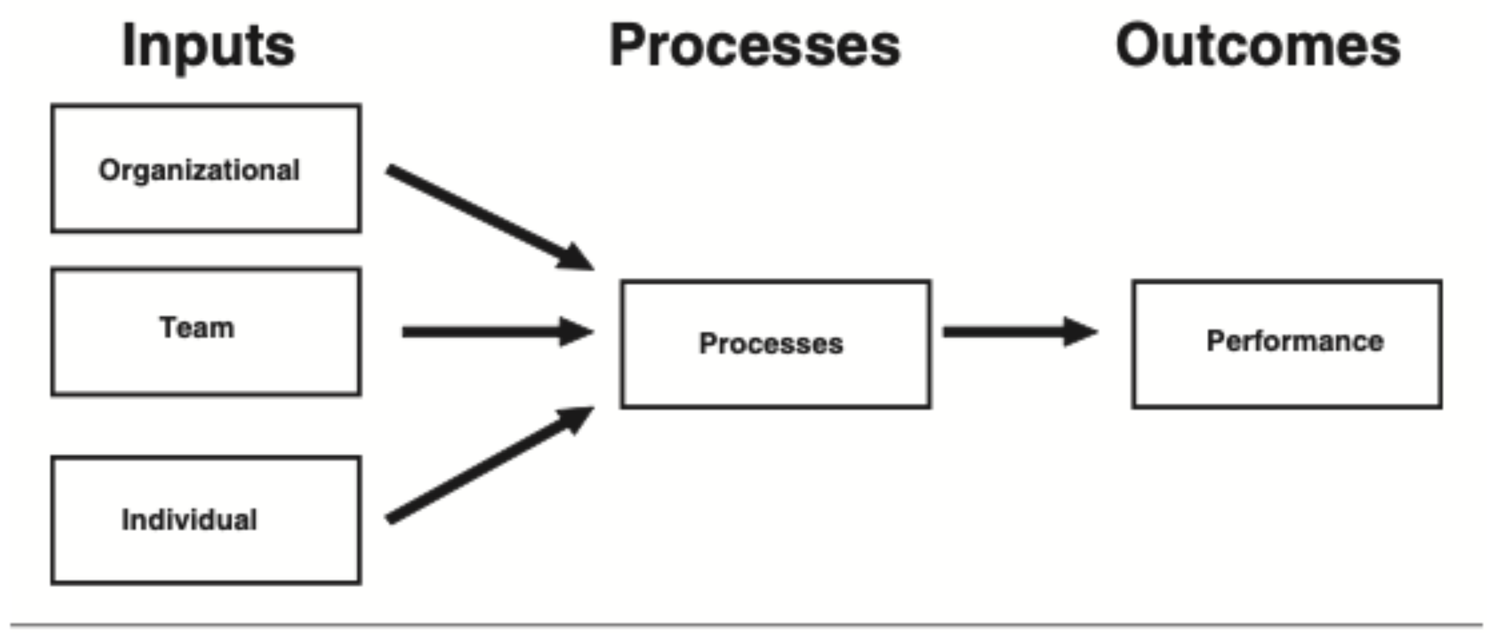
5 Impact
Team members think their work matters and creates change.

Julia Rozovsky, Analyst, Google People Operations
https://rework.withgoogle.com/blog/five-keys-to-a-successful-
m/



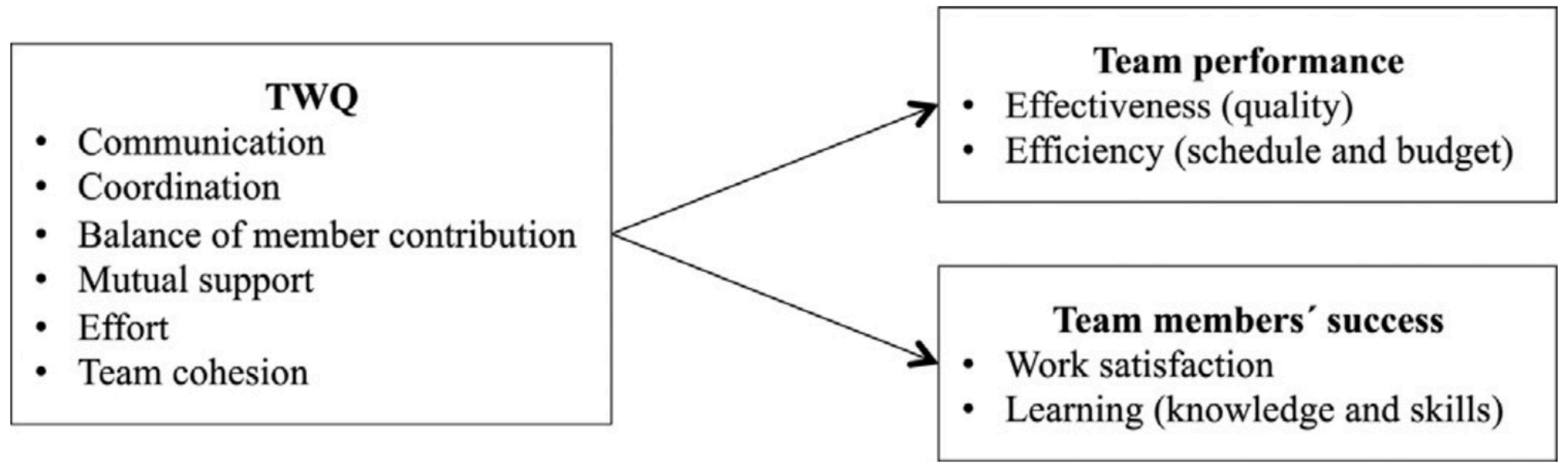






Inputs — Processes — Outcomes

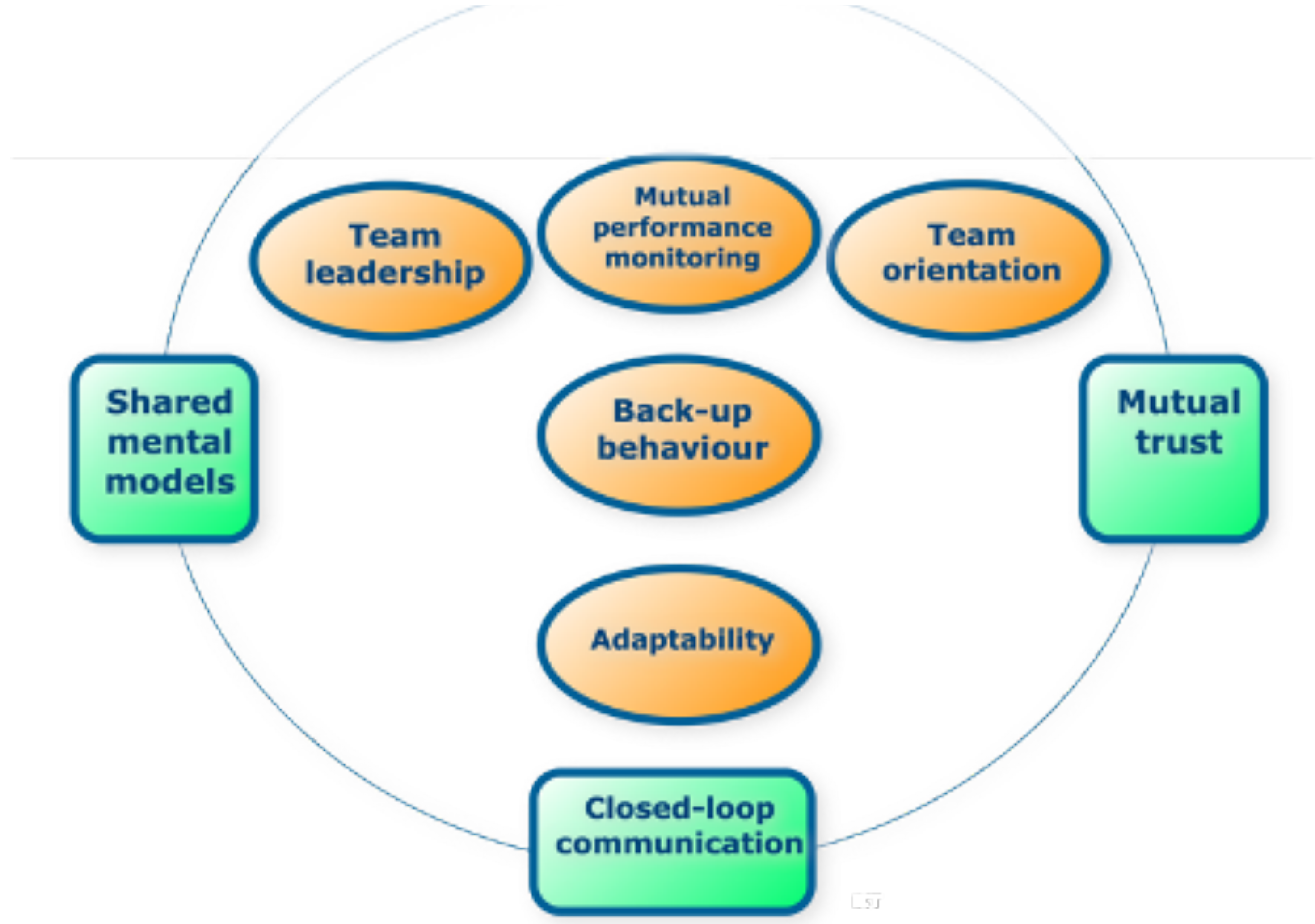
Organizational, Team, Individual → Processes → Performance



TWQ
- Communication
- Coordination
- Balance of member contribution
- Mutual support
- Effort
- Team cohesion

Team performance
- Effectiveness (quality)
- Efficiency (schedule and budget)

Team members´ success
- Work satisfaction
- Learning (knowledge and skills)



EXERCISE

1

- Individually (2 minutes):
  - What factors do you think *foster* team effectiveness for a software development team?

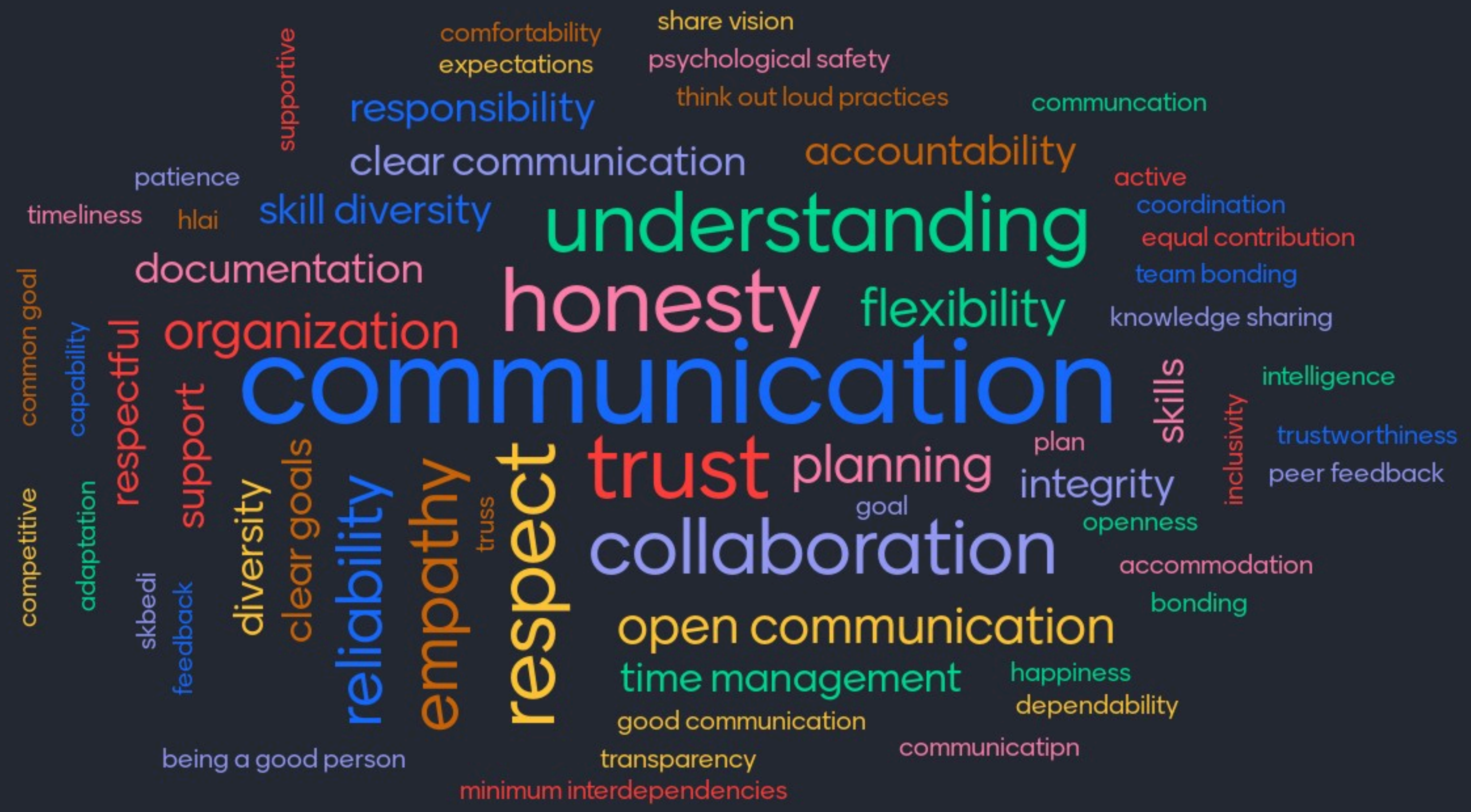
2

- Post the three factors you think are most important menti.com code 4776 6307

Mentimeter

What factors do you think foster team effectiveness?

172 responses

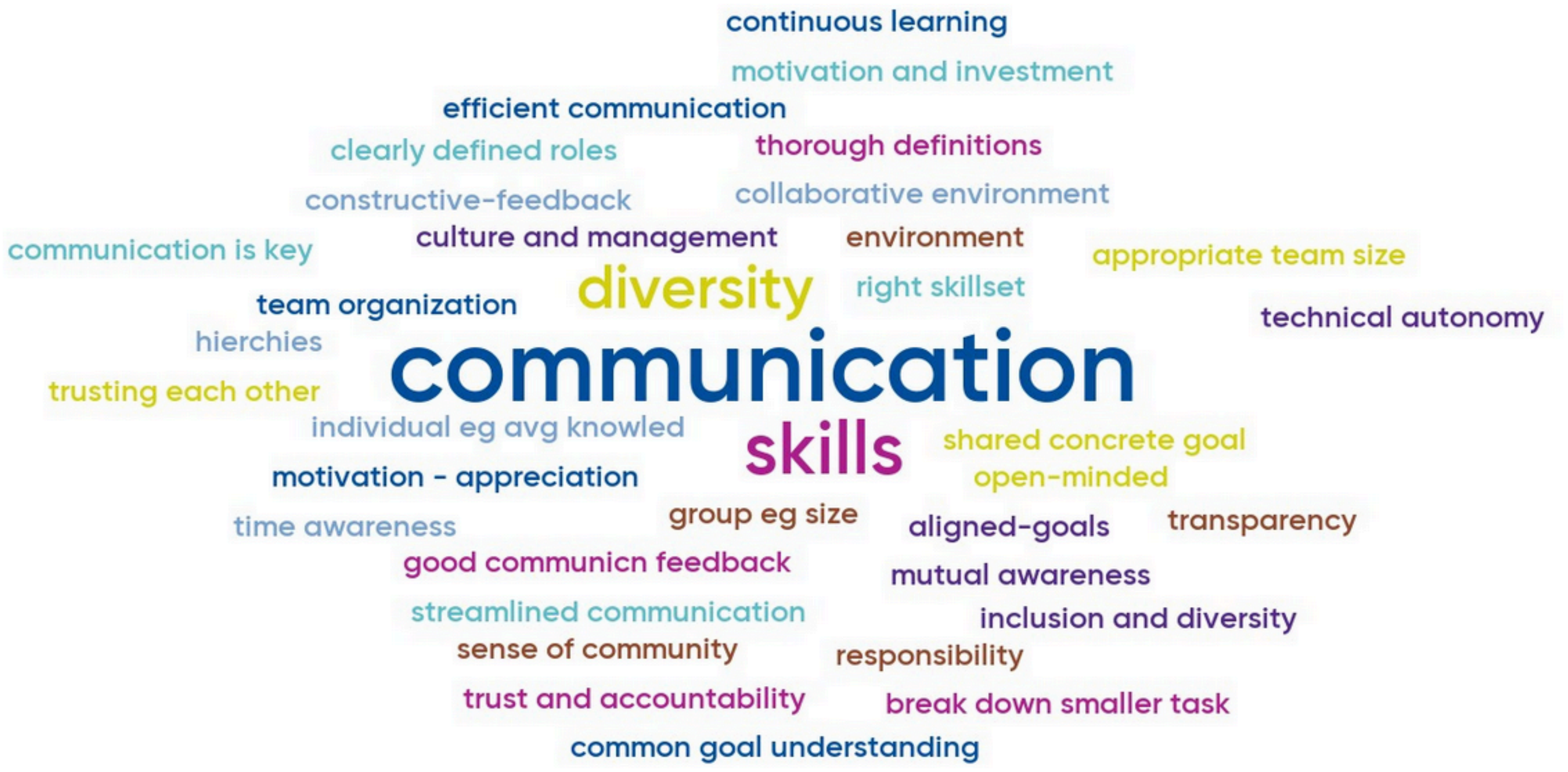


share vision
comfortability
expectations
psychological safety
supportive
responsibility
think out loud practices
communcation
clear communication
accountability
patience
active
coordination
timeliness
hlai
skill diversity
understanding
equal contribution
team bonding
documentation
honesty
flexibility
knowledge sharing
common goal
capability
respectful
organization
communication
skills
intelligence
inclusivity
trustworthiness
support
trust
planning
plan
integrity
peer feedback
competitive
adaptation
skbedi
feedback
diversity
clear goals
reliability
empathy
truss
respect
goal
openness
collaboration
accommodation
bonding
open communication
time management
happiness
dependability
good communication
communicatipn
being a good person
transparency
minimum interdependencies

14

61

What fosters team effectiveness?



continuous learning

motivation and investment

efficient communication

clearly defined roles

thorough definitions

constructive-feedback

collaborative environment

communication is key

culture and management

environment

appropriate team size

diversity

right skillset

team organization

technical autonomy

hierchies

communication

trusting each other

individual eg avg knowled

skills

shared concrete goal

open-minded

motivation - appreciation

group eg size

time awareness

aligned-goals

transparency

good communicn feedback

mutual awareness

streamlined communication

inclusion and diversity

sense of community

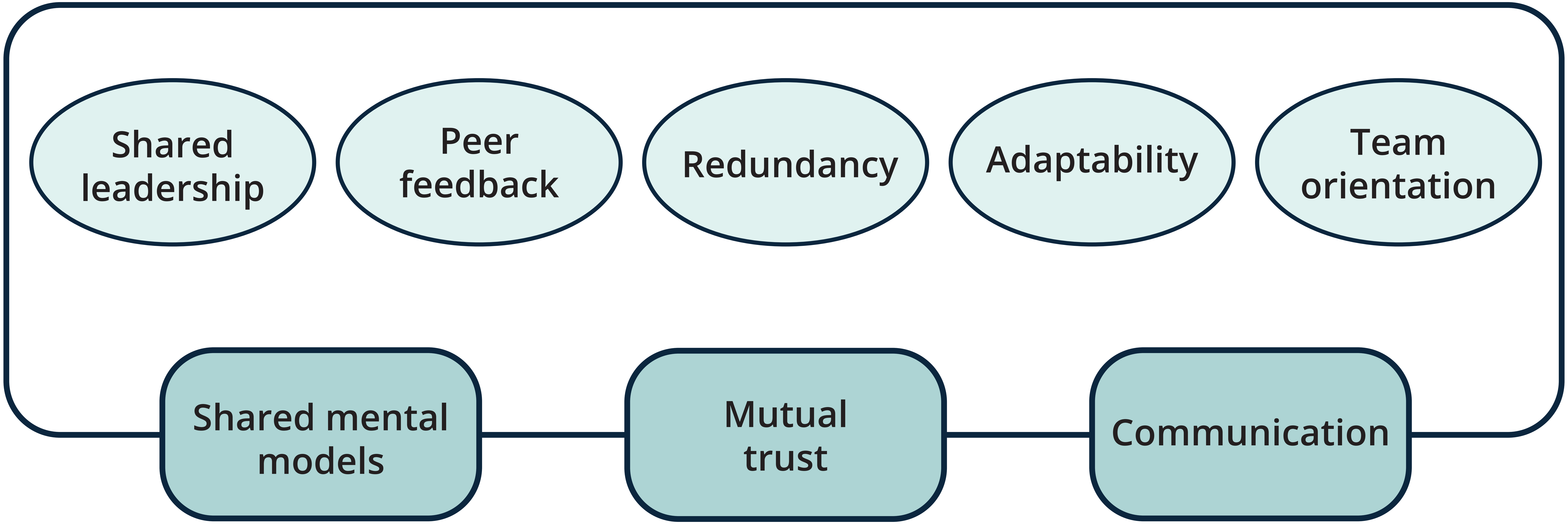
responsibility

trust and accountability

break down smaller task

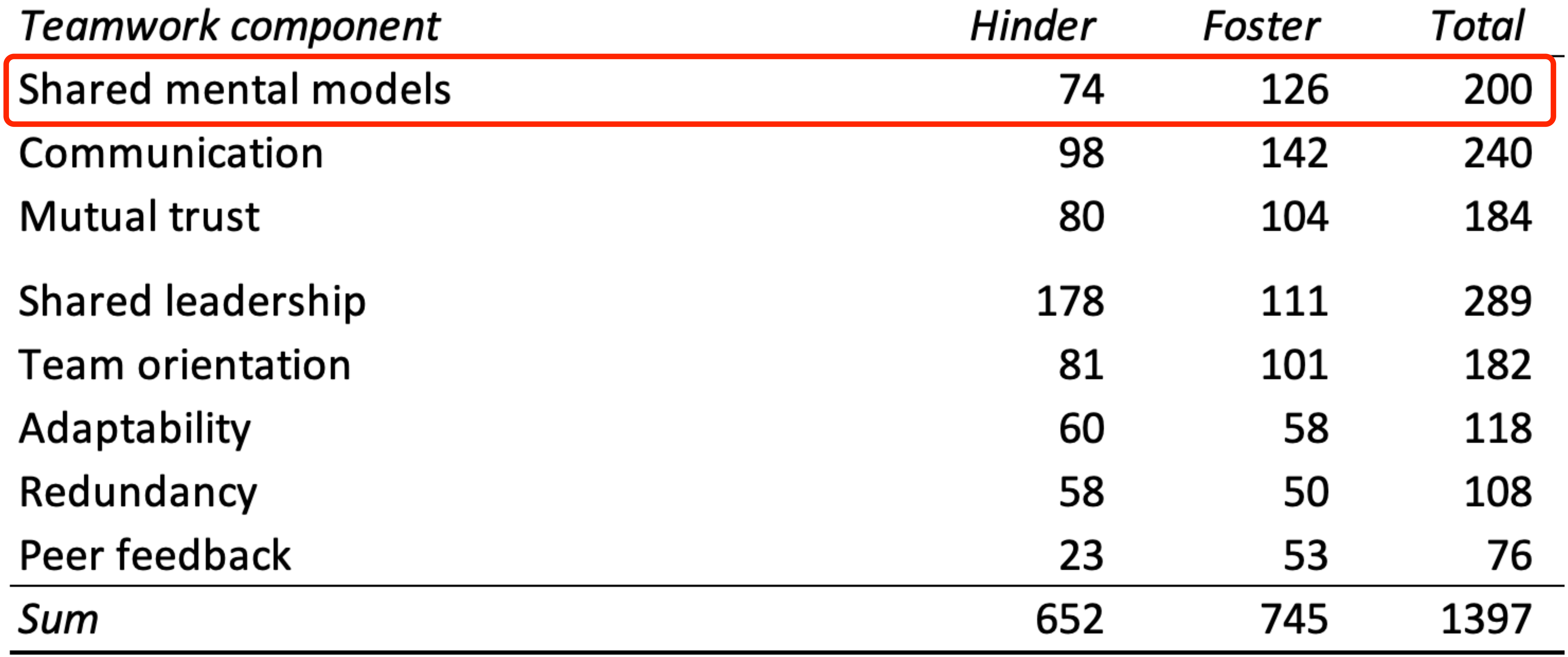
common goal understanding

Agile team effectiveness model (ATEM)



Strode, D., Dingsøyr, T., & Lindsjørn, Y. (2022). A Teamwork Effectiveness Model for Agile Software Development. *Empirical Software Engineering.* Modified from: Salas, E., Sims, D. E., and Burke, S. C., "Is there a "Big five" in teamwork?," *Small Group Research,* vol. 36, pp. 555-599, 2005.

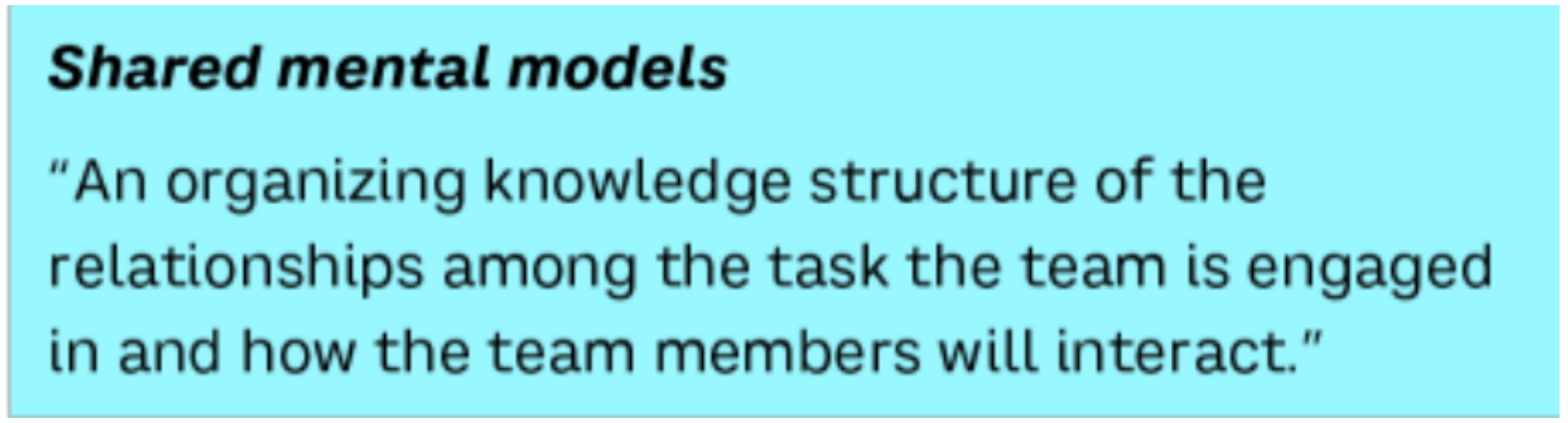
What fosters or hinders team performance?



Teamwork component	*Hinder*	*Foster*	*Total*
Shared mental models	74	126	200
Communication	98	142	240
Mutual trust	80	104	184
Shared leadership	178	111	289
Team orientation	81	101	182
Adaptability	60	58	118
Redundancy	58	50	108
Peer feedback	23	53	76
Sum	652	745	1397

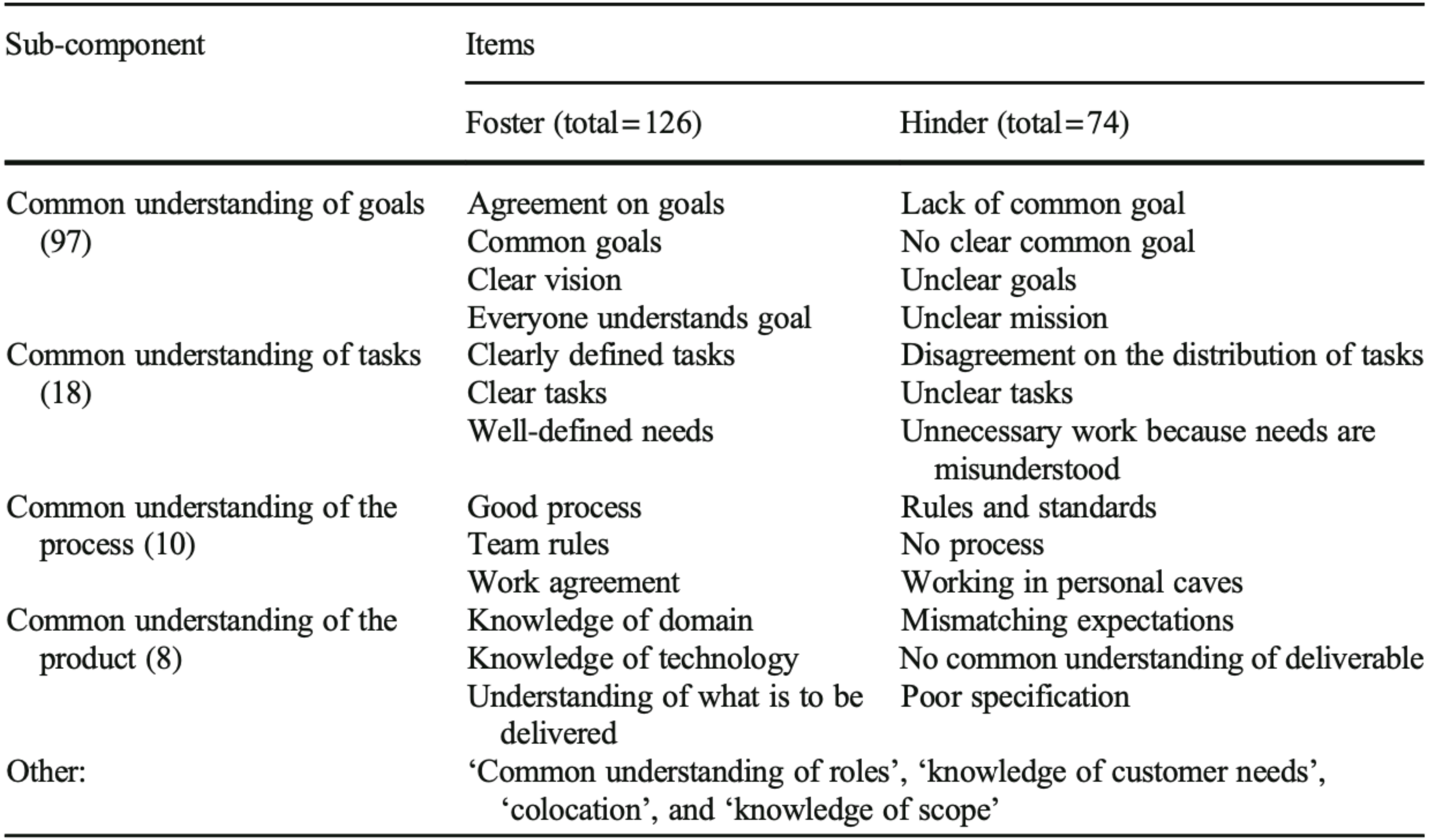
Strode, D., Dingsøyr, T., & Lindsjørn, Y. (2022). A Teamwork Effectiveness Model for Agile Software Development. *Empirical Software Engineering.*

Shared mental models



Shared mental models

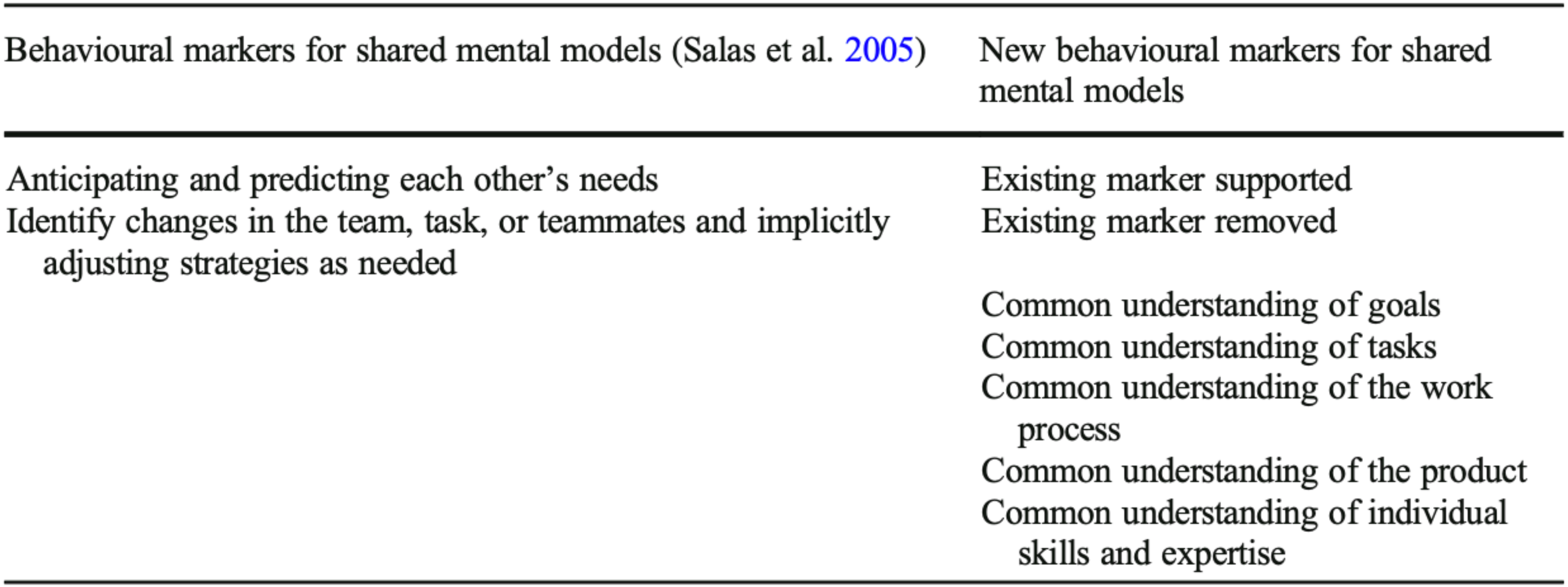
"An organizing knowledge structure of the relationships among the task the team is engaged in and how the team members will interact."



Sub-component	Items	
	Foster (total = 126)	Hinder (total = 74)
Common understanding of goals (97)	Agreement on goals	Lack of common goal
	Common goals	No clear common goal
	Clear vision	Unclear goals
	Everyone understands goal	Unclear mission
Common understanding of tasks (18)	Clearly defined tasks	Disagreement on the distribution of tasks
	Clear tasks	Unclear tasks
	Well-defined needs	Unnecessary work because needs are misunderstood
Common understanding of the process (10)	Good process	Rules and standards
	Team rules	No process
	Work agreement	Working in personal caves
Common understanding of the product (8)	Knowledge of domain	Mismatching expectations
	Knowledge of technology	No common understanding of deliverable
	Understanding of what is to be delivered	Poor specification
Other:	'Common understanding of roles', 'knowledge of customer needs', 'colocation', and 'knowledge of scope'	

Strode, D., Dingsøyr, T., & Lindsjørn, Y. (2022). A Teamwork Effectiveness Model for Agile Software Development. *Empirical Software Engineering*..

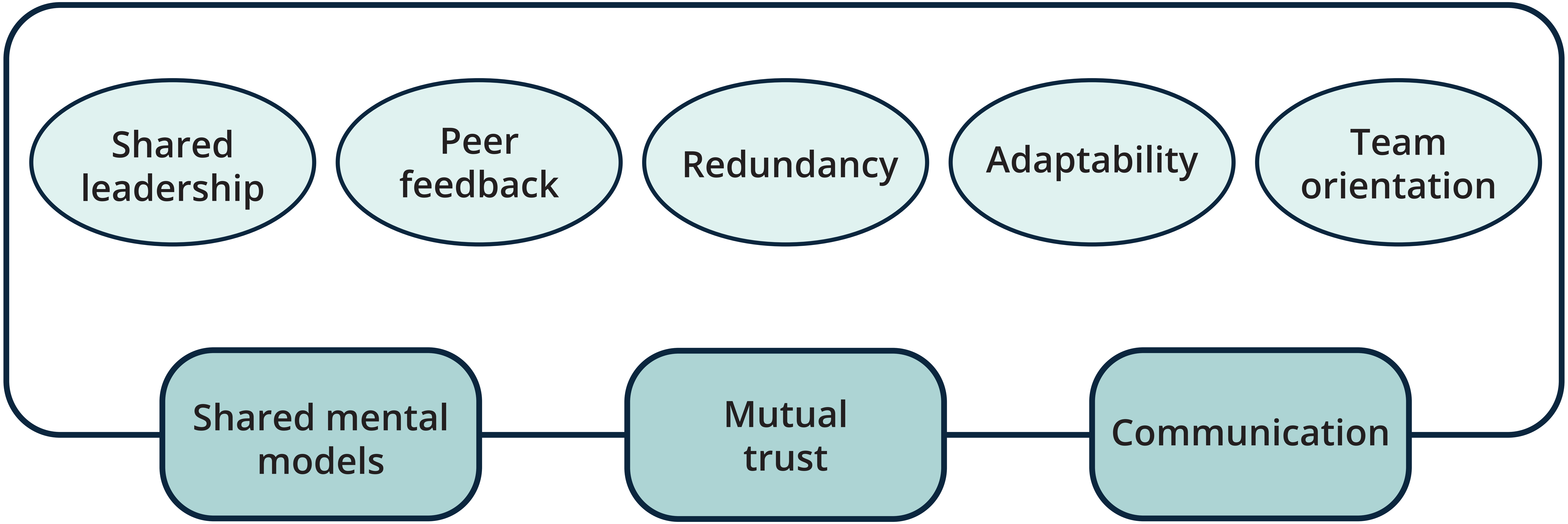
Revised behavioural markers



Behavioural markers for shared mental models (Salas et al. 2005)	New behavioural markers for shared mental models
Anticipating and predicting each other's needs	Existing marker supported
Identify changes in the team, task, or teammates and implicitly adjusting strategies as needed	Existing marker removed
	Common understanding of goals
	Common understanding of tasks
	Common understanding of the work process
	Common understanding of the product
	Common understanding of individual skills and expertise

Strode, D., Dingsøyr, T., & Lindsjørn, Y. (2022). A Teamwork Effectiveness Model for Agile Software Development. *Empirical Software Engineering*..

Agile team effectiveness model (ATEM)



Strode, D., Dingsøyr, T., & Lindsjørn, Y. (2022). A Teamwork Effectiveness Model for Agile Software Development. *Empirical Software Engineering.* Modified from: Salas, E., Sims, D. E., and Burke, S. C., "Is there a "Big five" in teamwork?," *Small Group Research,* vol. 36, pp. 555-599, 2005.

EXERCISE: TEACHING CASE

1

- Individually: Read teching case (7 minutes)
- Make a note for yourself on:
  - What are the main challenges to team effectiveness in the case?
  - Do you recognise challenges from the ATEM model?
  - What advice would you give to the team to improve their effectiveness?

2

- In pairs:
  - Present notes to each other
  - Identify one main recommendation you would give to the team
  - Post your recommendation on Mentimeter menti.com code 4776 6307

CHALLENGES IN THE TEACHING CASE:

- Little openness on problems
- Individual goals given priority over team goals
- Little insight into what other team mambers are working on

Recommendations to the team

90 responses

create a shared doc with defined standards that all teams can reference to reduce need for meetings across teams

Improve sprint planning to create a stable and achievable backlog.

Encourage active participation and discussion in meetings for better collaboration.

mplement more rigorous review processes to avoid unnecessary rework.

Emphasize the importance of timely problem reporting and resolution.

Foster a team environment where members actively seek and provide feedback.

communication

I recommend the team to communicate better so they don't do stuff like takign 100 hours to rewrite a module

I would recommend to better communicate.

👍 7 👤 76

Recommendations to the team

90 responses

Create an async method for them to provide updates without in person meetings

The team should hold scheduled meetings between the development project and business project so they can be on page better.

Needs to have better communication so that each team member is on same page.

My recommendation to the team is to improve open communication. It says on page 10 that "get insight into and obtain an understanding of what was happening" and "like delivering to a black box"

I'd say that it's of utmost importance to improve on communication, it's stated in page 10 that it was "impossible to...gain an understanding of what was going on".

Create a system where team members can give and receive feedback on their work frequently. This will help in early identification of issues and will reduce the chances of significant rework.

Set expectations early

I think they should continue to work together in teams and continuously share knowledge as it crops up

Improve inter-project communication to streamline decision-making and enhance efficiency during the "digital" phase

Recommendations to the team

90 responses

A reccomendation to the team is to try and schedule meetings based on the necessity of it or to do quick stand ups with a max time of 30 minutes or so. Also you could have designated work days

It's important to ask for help and identify problems early. They also spent too much time discussing things abstractly instead of doing work.

Have daily stand ups with both the business and development teams included. Because theres a lack of communication between them

Implement enhance communication strategies to stream line decision making and reduce meeting overload.

The perspectives of all sides should be considered and the manager should have more trust in the team

Weekly open-discussions where each team members report their progress along with any other issues that they ran into that week.

There should be a mutual trust between the team and leadership (ie Scrum master), so establishing measures to maintain that trust (better communication, for example) would help.

Better communication between developer and team

Have multiple leaders for specific issues that can make those decisions quickly and effectively, so that their planning process doesn't take as long.

👍 7 👤 76

Recommendations to the team

90 responses

Reducing the number of meetings and instead only having weekly team and mini team syncs so that people make progress

Empower Leaders to take decisions certain decisions quicker

Have a set meeting agenda and make sure that the meetings are efficient within a shorter amount of time

Proper meeting agendas; agile kanban method; minimal dependencies; extend time frame to remove pressure; increased communication. between teams

I would suggest that the team implements and agile workflow, specifically using sprints and standup to ensure that everyone in the team is on the same page which will help reduce delays and confusion

Since coordination was frustrating, I'd recommend planning less formal meetings.

The team seemed to be working in a siloed manner so, I would recommend improving communication and collaboration. People need to communicate clearly and transparently during meetings to stay informed.

Make sure to improve and focus on communication as much as possible

Improve flexibility and communication

7 76

Recommendations to the team

90 responses

Have stand-up meetings with both the business and development teams. this should help address the lack of inter-team communication.

Plan more meetings between each team to allow for more chances for communication

communcation, empathy, diversity

Honestly, the scrum master should probably back off a little bit and trust the developers more - it seemed like there was an environment of fear that hurt open discussion

Make the programme manager set up a task force to evaluate and recommend changes to enhance efficiency for the last release.

It's challenging to keep developers satisfied with projects with changes in human resources. Also by communicating more, people share the pressure from failure and risks.

Create a more streamlined and organized process for the sprint backlog to improve coordination.

Shrink down team numbers and size to reduce communication costs and be more effective in meetings, since the current issue is difficult to coordinate and communicate with so many teams.

I would recommend that the teams work independently. Having all teams participate in sprint meetings took time away from the deliverables. It would be better to segment the work to be more effective.

Recommendations to the team

90 responses

It seemed like during the scrum meetings, everyone in the team just focused on presenting what they were working on but ignored what each other said. I recommend enforcing feedback after each speaker.

Have a shared scheduling between the business and dev departments, and have an assigned scheduling person.

My recommendation is for the team to encourage open communication and regular feedback during daily stand-up meetings to ensure that all team members actively.

I recommend only doing meetings as necessary, and also be less stringent on who needs to come to the meeting (they can just review meeting notes).

I recommend that the team has better communication. Also, the team could have worked on the same floor to foster better communication and understood better what each team was doing.

Reduce scope of each team to focus on separable tasks, include only representatives for each team in a meeting to reduce overhead

Honestly, the scrum master probably needs to back off a little: there seemed like a lack of trust there that lead to developers being alienated or feeling afraid to bring up problems.

Create proper meeting agenda; increase communication between teams; include minimal dependencies for tasks; extend time frame for deadlines to eliminate time pressure.

Establish daily cross-project meetings to ensure both teams are aware of each other's progress, challenges, and objectives

7 76

Recommendations to the team

90 responses

I think the team should strive to implement open communication to help with the transition between the business project and the development project as it was a source of frustration.

It seemed like in the scrum meetings people only focused on presenting what they worked on but ignored each other's speech. I recommend enforcing feedback sessions after each speaker.

Establishing a clear and efficient handover process of tasks between business and development side to make sure there is a shared understanding of requirements.

Foster communication within the team

My recommendation to the team is to have better communication among team members. One way they could do this is to provide better documentation. This means noting any new changes or developments.

each group should have a leader, or PM of some sorts that understands the projects that they are assigned as well as the other teams. This makes communication between teams more smooth and clear.

Establish regular cross-project meetings or communication channels to ensure that both teams are aware of each other's progress, challenges, and objectives.

One thing I would recommend to the team is to foster communication between the business and development teams by developing a better plan beforehand and increasing the documentation made by both sides

Both business and software team lacks info from the other party. Poor coordination and communication was an issue here. I suggest that team have more meeting with each other

👍 7 👤 76

Recommendations to the team

90 responses

1) Late Problem Discovery2) lack of multiskilling 3) Implement regular feedback mechanisms

The team should prioritize having more productive and planned out meetings (even if that means they will be less frequent) because it is more effective to use time together more constructively.

Create some norm which greater coordinates management and discussion making from the business development with actual development teams. Possibly providing more autonomy.

clarify roles and responsibilities

Limit meeting time - making sure that people have a concept of what's on the agenda and what questions people have that need to get answered ahead of time (esp. to get feedback on overall confusion)

The team should have more frequent meetings to discuss what they are working on and give updates on their work.

have a clear feedback mechanism

Reduce the numbers of tasks each team works on. Also initiate more regular discussion between business and development teams so that nobody feels they're missing information.

Meetings should be more intentional and efficient to make the most of the times that everyone can meet together

Recommendations to the team

90 responses

To plan for work time and meeting time, so that members are not always in meetings.

Someone from each team (like a PM) should communicate with other PMs to get team progress and figure out next steps. Should be someone who understands both business and technical jargon.

Consider overlapping some of the development phases. For example, while one team is doing analysis of needs for the next phase, another team can start the solution description of the current phase.

Encourage transparent communication across all parties in order for everyone to be on the same page before moving forward.

Honesty, collaboration, diversity

Have one or more designated employees to act as a bridge between business and development projects so it's not like working with a black box.

Communication between different projects was not effective, so maybe add more in-between roles for projects to keep people on the same page

My recommendation to the team is to improve communication among team members. They can do this by having better documentation, like noting down any new development and sending this to everyone.

To designate clear protected individual work times and meeting times to allow for a balance between collaboration and getting work done

👍 7

76

Mentimeter

Recommendations to the team

90 responses

Recommendation: team members should speak up on what tasks they do or don't want to do, and the scrum master should also ask for feedback on whether members want to work on tasks.(Discussed with Yuhe)

Yuhe and Luna: 1. The developers should be more vocal and open during team meetings.2. Team members should also provide more feedback on others' code

Enhance transperency for honest and truthful communication

Create a document where everyone can make process updates and give feedback so that work does not have to be visited again

Foster a collaborative + supportive environment, so that problems can be discussed early and easily. This will prevent larger issues in the future.

-

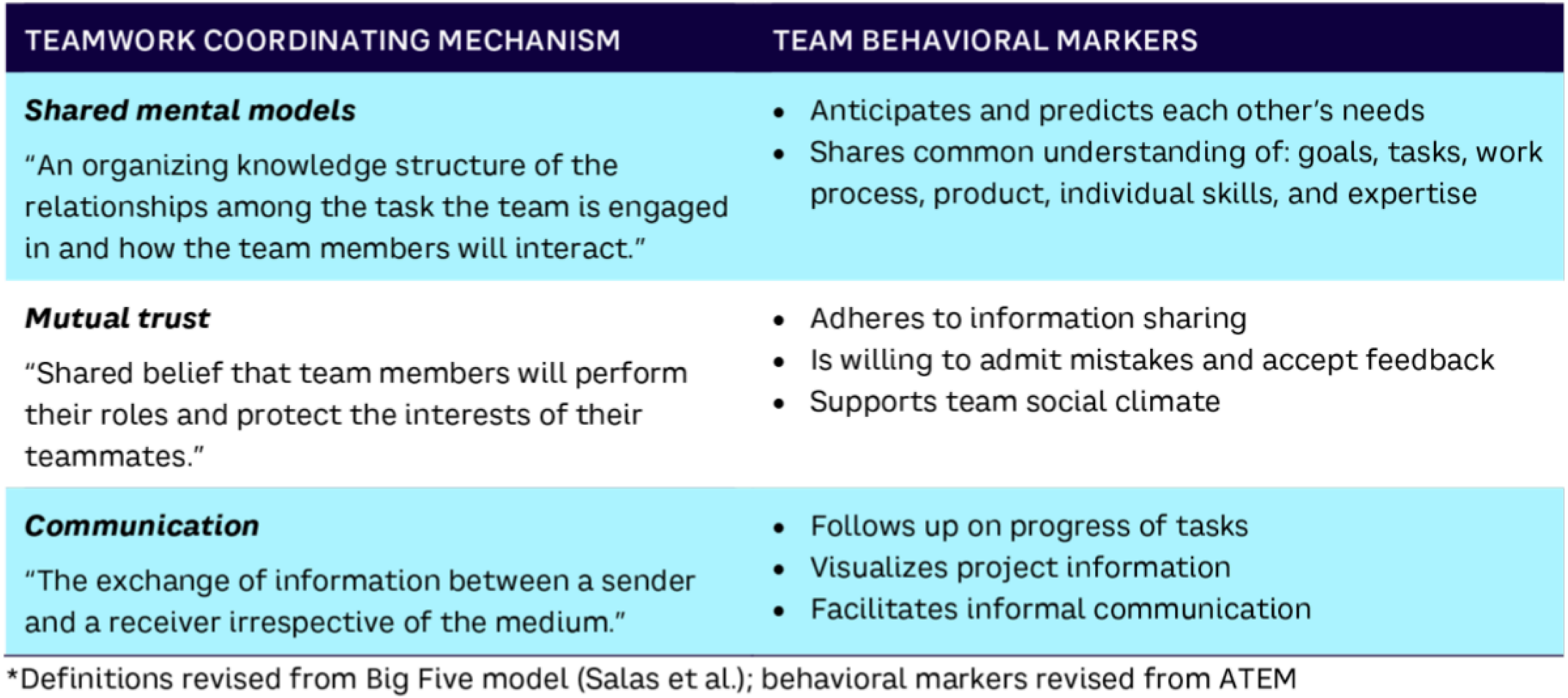
Problems are all indicators of meeting and planning not seen as important and responsibilities individualized. So internalize planning and scum meetings, follow schedule, have tasks as group effort.

I would recommend the team to hire people with more experience either in this tech field or with efficient communication

Encourage active participation and discussion in meetings for better collaboration.

7 👍 76

How to use the ATEM



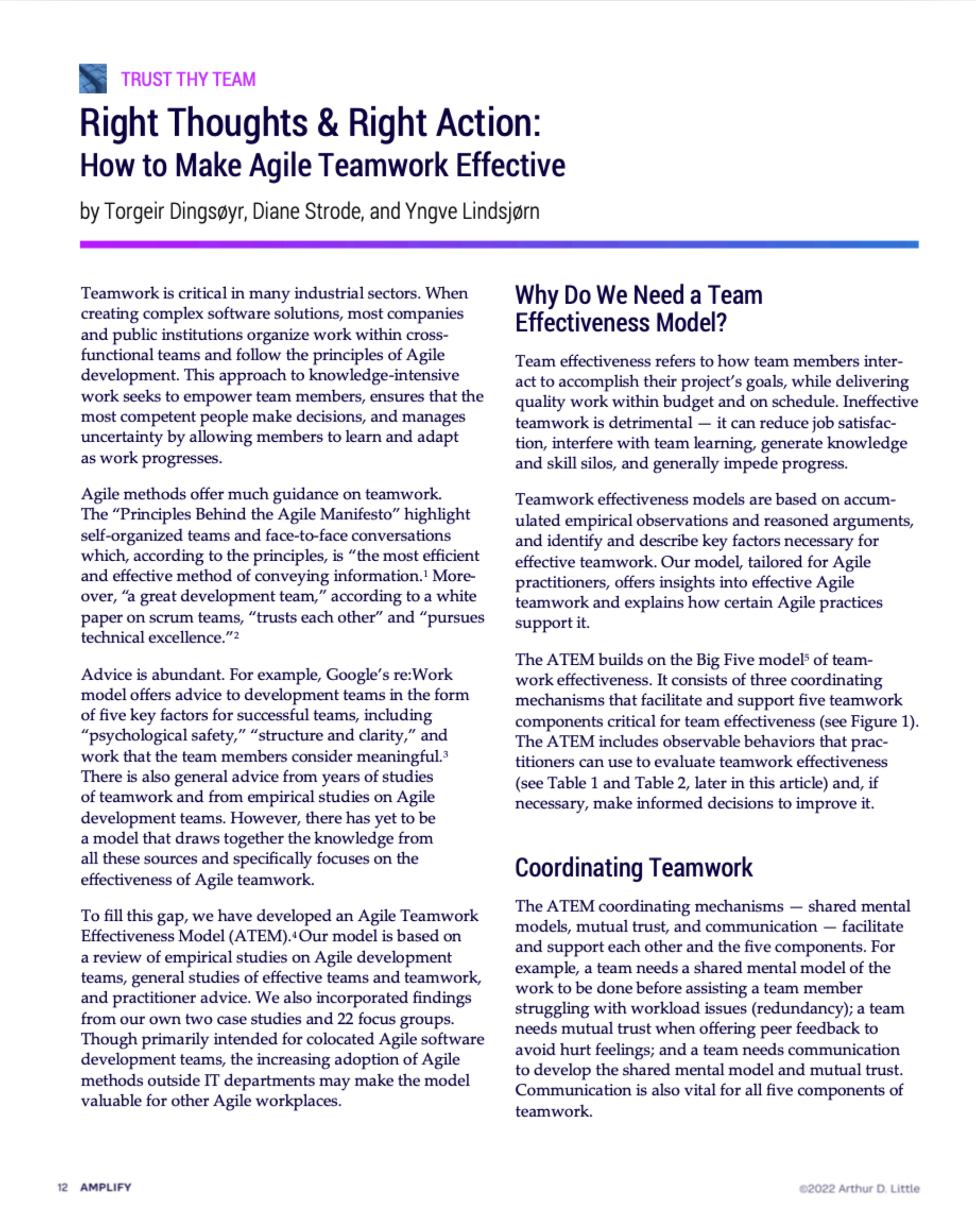
TEAMWORK COORDINATING MECHANISM	TEAM BEHAVIORAL MARKERS
Shared mental models “An organizing knowledge structure of the relationships among the task the team is engaged in and how the team members will interact.”	• Anticipates and predicts each other’s needs • Shares common understanding of: goals, tasks, work process, product, individual skills, and expertise
Mutual trust “Shared belief that team members will perform their roles and protect the interests of their teammates.”	• Adheres to information sharing • Is willing to admit mistakes and accept feedback • Supports team social climate
Communication “The exchange of information between a sender and a receiver irrespective of the medium.”	• Follows up on progress of tasks • Visualizes project information • Facilitates informal communication

*Definitions revised from Big Five model (Salas et al.); behavioral markers revised from ATEM

Dingsøyr, T., Strode, D., and Lindsjørn, Y. (2022) Right Thoughts & Right Action: How to Make Agile Teamwork Effective. *Amplify* 12-17. Available: https://bit.ly/3vSiffB

Franz Ferdinand. “Right Action (Official Video).” YouTube, 7 July 2013.

Read more



TRUST THY TEAM

Right Thoughts & Right Action: How to Make Agile Teamwork Effective

by Torgeir Dingsøyr, Diane Strode, and Yngve Lindsjørn

Teamwork is critical in many industrial sectors. When creating complex software solutions, most companies and public institutions organize work within cross-functional teams and follow the principles of Agile development. This approach to knowledge-intensive work seeks to empower team members, ensures that the most competent people make decisions, and manages uncertainty by allowing members to learn and adapt as work progresses.

Agile methods offer much guidance on teamwork. The “Principles Behind the Agile Manifesto” highlight self-organized teams and face-to-face conversations which, according to the principles, is “the most efficient and effective method of conveying information.[1] Moreover, “a great development team,” according to a white paper on scrum teams, “trusts each other” and “pursues technical excellence.”[2]

Advice is abundant. For example, Google’s re:Work model offers advice to development teams in the form of five key factors for successful teams, including “psychological safety,” “structure and clarity,” and work that the team members consider meaningful.[3] There is also general advice from years of studies of teamwork and from empirical studies on Agile development teams. However, there has yet to be a model that draws together the knowledge from all these sources and specifically focuses on the effectiveness of Agile teamwork.

To fill this gap, we have developed an Agile Teamwork Effectiveness Model (ATEM).[4] Our model is based on a review of empirical studies on Agile development teams, general studies of effective teams and teamwork, and practitioner advice. We also incorporated findings from our own two case studies and 22 focus groups. Though primarily intended for colocated Agile software development teams, the increasing adoption of Agile methods outside IT departments may make the model valuable for other Agile workplaces.

Why Do We Need a Team Effectiveness Model?

Team effectiveness refers to how team members interact to accomplish their project’s goals, while delivering quality work within budget and on schedule. Ineffective teamwork is detrimental — it can reduce job satisfaction, interfere with team learning, generate knowledge and skill silos, and generally impede progress.

Teamwork effectiveness models are based on accumulated empirical observations and reasoned arguments, and identify and describe key factors necessary for effective teamwork. Our model, tailored for Agile practitioners, offers insights into effective Agile teamwork and explains how certain Agile practices support it.

The ATEM builds on the Big Five model[5] of teamwork effectiveness. It consists of three coordinating mechanisms that facilitate and support five teamwork components critical for team effectiveness (see Figure 1). The ATEM includes observable behaviors that practitioners can use to evaluate teamwork effectiveness (see Table 1 and Table 2, later in this article) and, if necessary, make informed decisions to improve it.

Coordinating Teamwork

The ATEM coordinating mechanisms — shared mental models, mutual trust, and communication — facilitate and support each other and the five components. For example, a team needs a shared mental model of the work to be done before assisting a team member struggling with workload issues (redundancy); a team needs mutual trust when offering peer feedback to avoid hurt feelings; and a team needs communication to develop the shared mental model and mutual trust. Communication is also vital for all five components of teamwork.

12 AMPLIFY

©2022 Arthur D. Little

https://bit.ly/3vSiffB

Empirical Software Engineering _######################_
https://doi.org/10.1007/s10664-021-10115-0

Check for updates

A teamwork effectiveness model for agile software development

Diane Strode[1] · Torgeir Dingsøyr[2,3] · Yngve Lindsjorn[4]

Accepted: 29 December 2021/[redacted]

Abstract
Teamwork is crucial in software development, particularly in agile development teams which are cross-functional and where team members work intensively together to develop a cohesive software solution. Effective teamwork is not easy; prior studies indicate challenges with communication, learning, prioritization, and leadership. Nevertheless, there is much advice available for teams, from agile methods, practitioner literature, and general studies on teamwork to a growing body of empirical studies on teamwork in the specific context of agile software development. This article presents the agile teamwork effectiveness model (ATEM) for colocated agile development teams. The model is based on evidence from focus groups, case studies, and multi-vocal literature and is a revision of a general team effectiveness model. Our model of agile teamwork effectiveness is composed of shared leadership, team orientation, redundancy, adaptability, and peer feedback. Coordinating mechanisms are needed to facilitate these components. The coordinating mechanisms are shared mental models, communication, and mutual trust. We critically examine the model and discuss extensions for very small, multi-team, distributed, and safety-critical development contexts. The model is intended for researchers, team members, coaches, and leaders in the agile community.

Keywords Agile leadership · Agile methods · Agile teamwork model · Agile teams · Big five model of teamwork · Mutual performance monitoring · Peer feedback · Redundancy · Scrum teams · Teamwork model · Teamwork theory

Communicated by: Vittorio Cortellessa

✉ Diane Strode
diane.strode@whitireia.ac.nz

Torgeir Dingsøyr
torgeir.dingsoyr@ntnu.no

Yngve Lindsjorn
ynglin@ifi.uio.no

Extended author information available on the last page of the article

Springer

https://rdcu.be/cIINu

https://bit.ly/3w9Ydhd

Agile Development at Scale: Challenges and Success Factors

Guest lecture, CMU 17-313: Foundations of Software Engineering, 31st October 2023

Torgeir Dingsøyr
Professor, Department of Computer Science
Norwegian University of Science and Technology
Adjunct Chief Scientist, SimulaMet

simulamet

NTNU – Trondheim
Norwegian University of Science and Technology

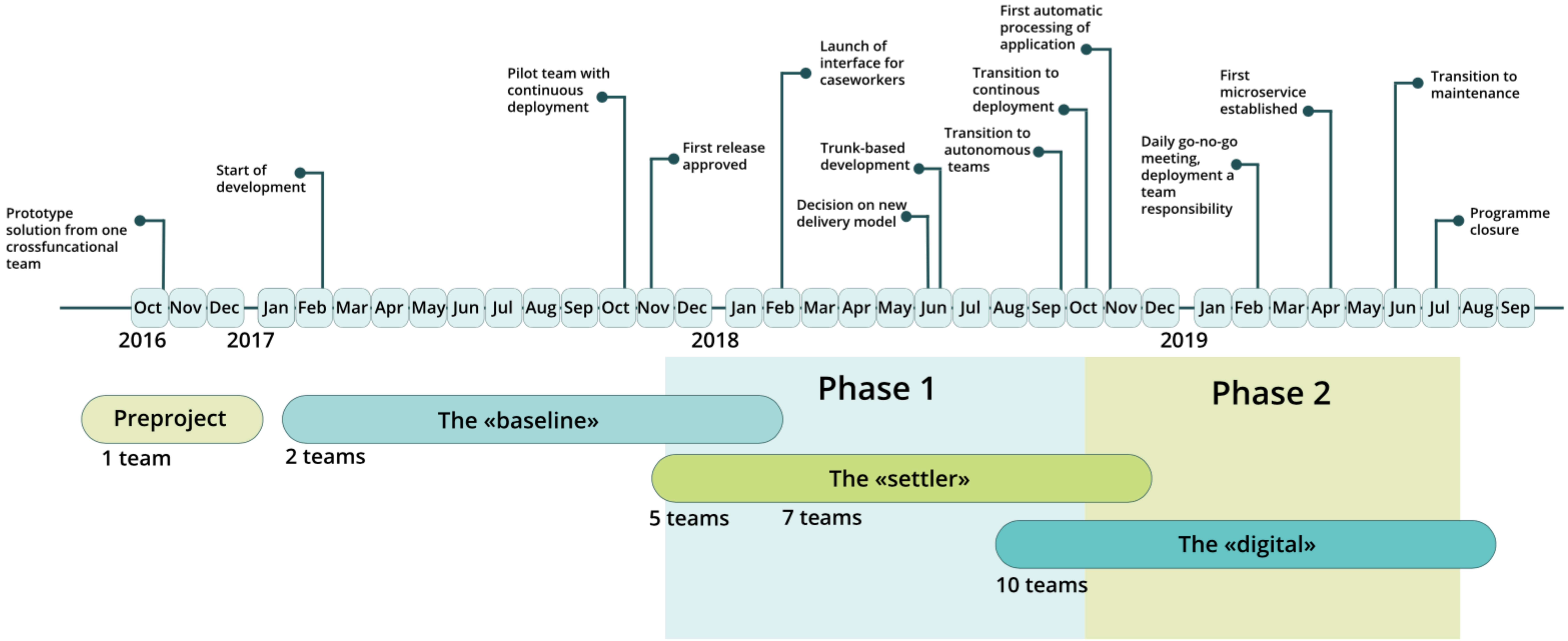
Home ground of Agile Methods

“agile value set and practices best suit colocated teams of about 50 people or fewer who have easy access to user and business experts and are developing projects that are not life-critical”

Williams and Cockburn, 2003

Williams, L. and Cockburn, A., "Agile Software Development: It’s about Feedback and Change," *IEEE Computer,* vol. 36, pp. 39-43, 2003.

Case: Parental benefit programme



Prototype solution from one crossfuncational team

Start of development

Pilot team with continuous deployment

First release approved

Launch of interface for caseworkers

Decision on new delivery model

Trunk-based development

Transition to autonomous teams

Transition to continous deployment

First automatic processing of application

Daily go-no-go meeting, deployment a team responsibility

First microservice established

Transition to maintenance

Programme closure

Phase 1

Phase 2

Preproject

1 team

The «baseline»

2 teams

The «settler»

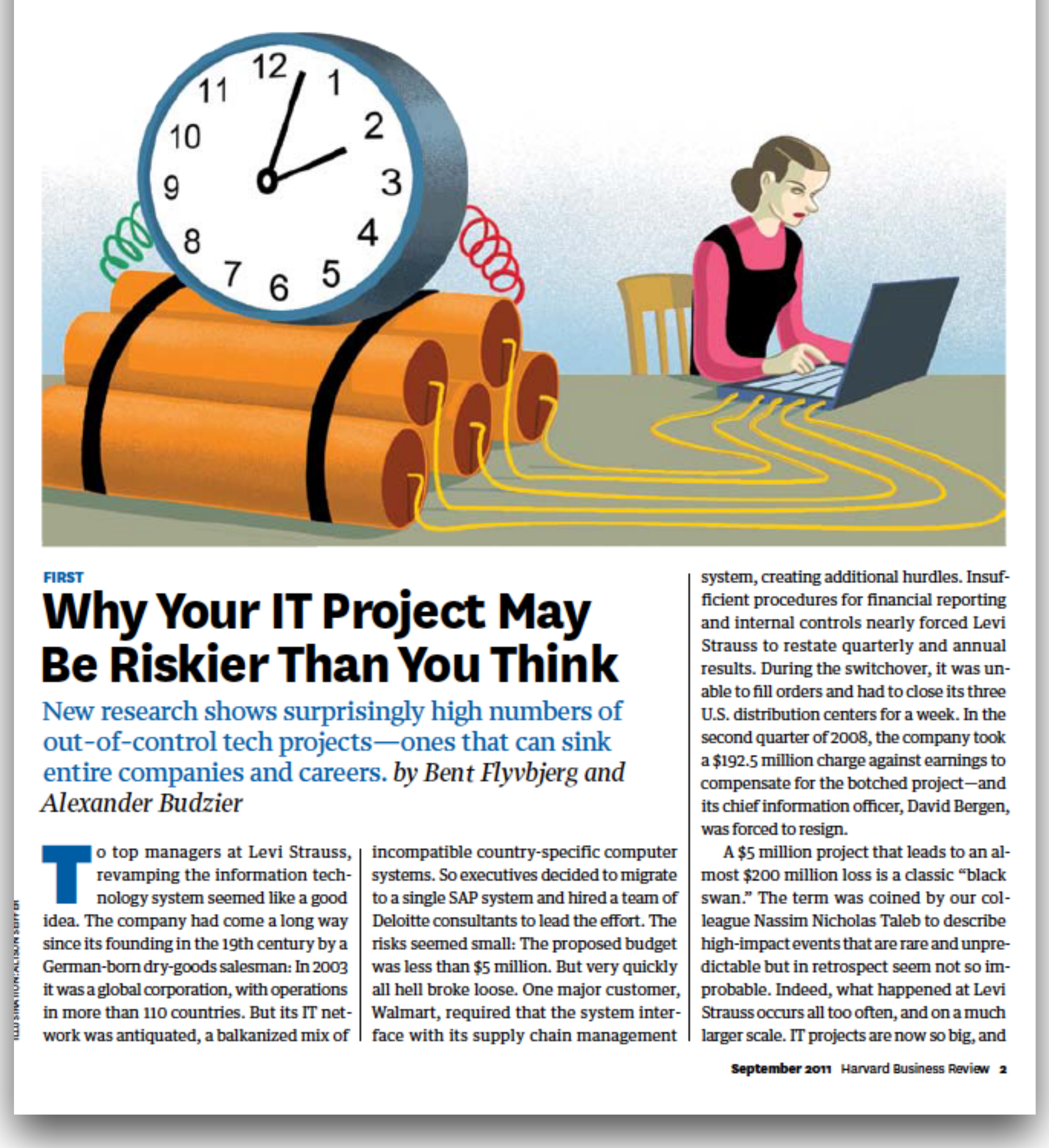
5 teams

7 teams

The «digital»

10 teams

Dingsøyr, T., Bjørnson, F. O., Schrof, J., and Sporsem, T., "A longitudinal explanatory case study of coordination in a very large development programme: the impact of transitioning from a first- to a second-generation large-scale agile development method," *Empirical Software Engineering*, vol. 28, p. 49, 2022/11/08 2023. 10.1007/s10664-022-10230-6. https://rdcu.be/c3FQ4



FIRST

Why Your IT Project May Be Riskier Than You Think

New research shows surprisingly high numbers of out-of-control tech projects—ones that can sink entire companies and careers. *by Bent Flyvbjerg and Alexander Budzier*

ILLUSTRATION: ALISON SEIFFER

To top managers at Levi Strauss, revamping the information technology system seemed like a good idea. The company had come a long way since its founding in the 19th century by a German-born dry-goods salesman: In 2003 it was a global corporation, with operations in more than 110 countries. But its IT network was antiquated, a balkanized mix of incompatible country-specific computer systems. So executives decided to migrate to a single SAP system and hired a team of Deloitte consultants to lead the effort. The risks seemed small: The proposed budget was less than $5 million. But very quickly all hell broke loose. One major customer, Walmart, required that the system interface with its supply chain management system, creating additional hurdles. Insufficient procedures for financial reporting and internal controls nearly forced Levi Strauss to restate quarterly and annual results. During the switchover, it was unable to fill orders and had to close its three U.S. distribution centers for a week. In the second quarter of 2008, the company took a $192.5 million charge against earnings to compensate for the botched project—and its chief information officer, David Bergen, was forced to resign.

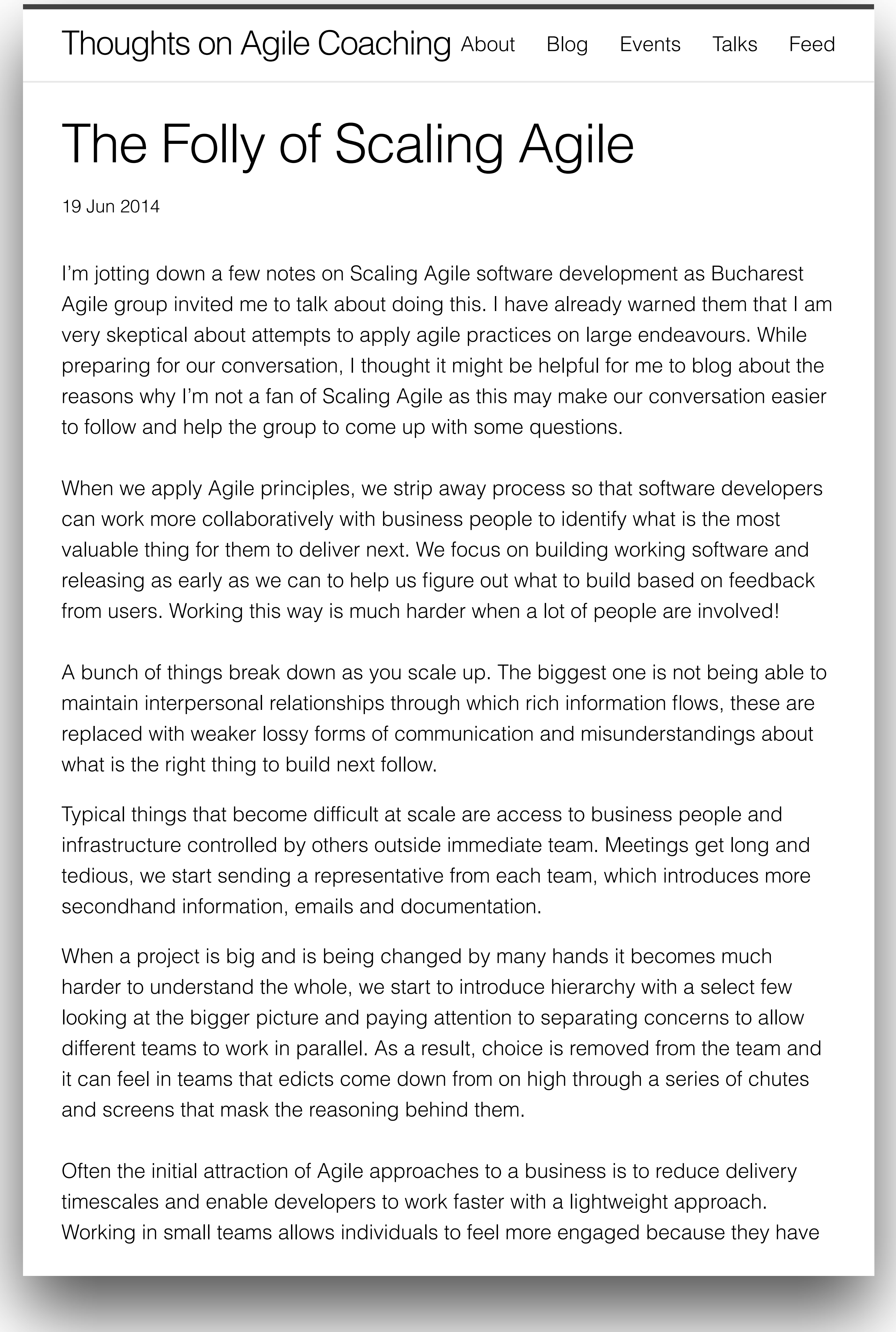
A $5 million project that leads to an almost $200 million loss is a classic "black swan." The term was coined by our colleague Nassim Nicholas Taleb to describe high-impact events that are rare and unpredictable but in retrospect seem not so improbable. Indeed, what happened at Levi Strauss occurs all too often, and on a much larger scale. IT projects are now so big, and

Flyvbjerg, B. and Budzier, A., "Why Your IT Project May Be Riskier Than You Think," *Harvard Business Review*, vol. 89, pp. 23-25, Sep 2011.

Arguments against Scaling Agile

«…meetings gets long and tedious, we start sending a representative from each team, which introduces more secondhand information, emails and documentation.»

«…not being able to maintain interpersonal relationships through which rich information flows …»



The Folly of Scaling Agile

19 Jun 2014

I'm jotting down a few notes on Scaling Agile software development as Bucharest Agile group invited me to talk about doing this. I have already warned them that I am very skeptical about attempts to apply agile practices on large endeavours. While preparing for our conversation, I thought it might be helpful for me to blog about the reasons why I'm not a fan of Scaling Agile as this may make our conversation easier to follow and help the group to come up with some questions.

When we apply Agile principles, we strip away process so that software developers can work more collaboratively with business people to identify what is the most valuable thing for them to deliver next. We focus on building working software and releasing as early as we can to help us figure out what to build based on feedback from users. Working this way is much harder when a lot of people are involved!

A bunch of things break down as you scale up. The biggest one is not being able to maintain interpersonal relationships through which rich information flows, these are replaced with weaker lossy forms of communication and misunderstandings about what is the right thing to build next follow.

Typical things that become difficult at scale are access to business people and infrastructure controlled by others outside immediate team. Meetings get long and tedious, we start sending a representative from each team, which introduces more secondhand information, emails and documentation.

When a project is big and is being changed by many hands it becomes much harder to understand the whole, we start to introduce hierarchy with a select few looking at the bigger picture and paying attention to separating concerns to allow different teams to work in parallel. As a result, choice is removed from the team and it can feel in teams that edicts come down from on high through a series of chutes and screens that mask the reasoning behind them.

Often the initial attraction of Agile approaches to a business is to reduce delivery timescales and enable developers to work faster with a lightweight approach. Working in small teams allows individuals to feel more engaged because they have

https://agilecoach.typepad.com/agile-coaching/2014/06/the-folly-of-scaling-agile.html

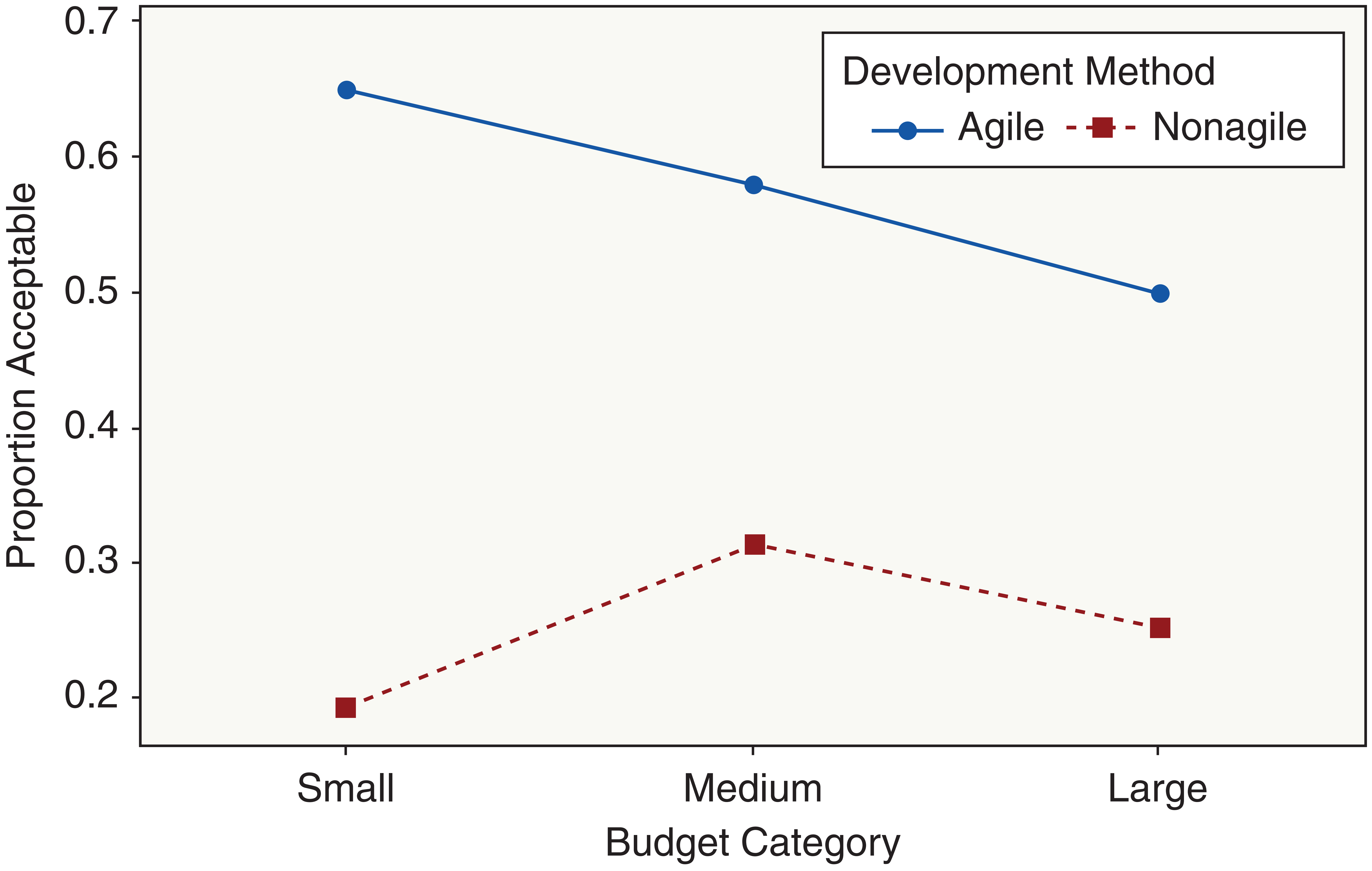
WARNING

DO NOT TRY
THIS AT HOME.

Project Success and Size and Method

«Large-scale software development succeeds more often when using agile methods»

(Jørgensen 2019)



Jørgensen, M., "Relationships Between Project Size, Agile Practices, and Successful Software Development: Results and Analysis," *IEEE Software,* vol. 36, pp. 39-43, 2019.

Challenges with Scale

- Autonomy
- Sharing knowledge in a large project / programme
- Coordinating many development teams

Edison, H., Wang, X., & K., C. (2021). Comparing Methods for Large-Scale Agile Software Development: A Systematic Literature Review. *IEEE Transactions on Software Engineering*, 1-1. doi:10.1109/TSE.2021.3069039
Dingsøyr, T. and Moe, N. B., "Research Challenges in Large-Scale Agile Software Development," *ACM Software Engineering Notes*, vol. 38, pp. 38-39, 2013.

Coordination

Importance of Coordination

> *«While there is no single cause of the software crisis, a major contribution is the problem of coordinating activities while developing large software systems. We argue that coordination becomes much more difficult as project size and complexity increases»*

Kraut and Streeter, *Communications of the ACM,* 1995

Kraut, R. E. and Streeter, L. A., "Coordination in software development," *Communications of the ACM*, vol. 38, pp. 69-81, 1995.

Change in coordination practices

Personal

Individual

Horisontal

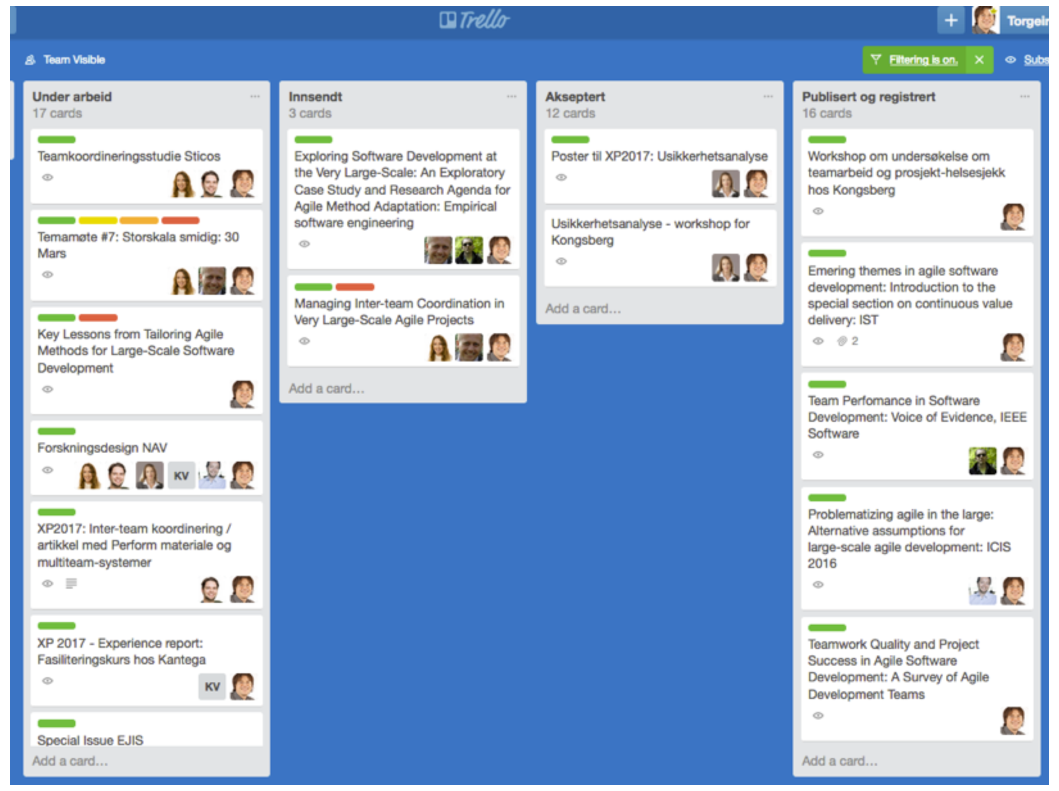
vertical

Group

Scheduled meetings

Unscheduled meetings

Impersonal



Under arbeid
17 cards

- Teamkoordineringsstudie Sticos
- Temamøte #7: Storskala smidig: 30 Mars
- Key Lessons from Tailoring Agile Methods for Large-Scale Software Development
- Forskningsdesign NAV
- XP2017: Inter-team koordinering / artikkel med Perform materiale og multiteam-systemer
- XP 2017 - Experience report: Fasiliteringskurs hos Kantega
- Special Issue EJIS

Innsendt
3 cards

- Exploring Software Development at the Very Large-Scale: An Exploratory Case Study and Research Agenda for Agile Method Adaptation: Empirical software engineering
- Managing Inter-team Coordination in Very Large-Scale Agile Projects

Akseptert
12 cards

- Poster til XP2017: Usikkerhetsanalyse
- Usikkerhetsanalyse - workshop for Kongsberg

Publisert og registrert
16 cards

- Workshop om undersøkelse om teamarbeid og prosjekt-helsesjekk hos Kongsberg
- Emering themes in agile software development: Introduction to the special section on continuous value delivery: IST
- Team Perfomance in Software Development: Voice of Evidence, IEEE Software
- Problematizing agile in the large: Alternative assumptions for large-scale agile development: ICIS 2016
- Teamwork Quality and Project Success in Agile Software Development: A Survey of Agile Development Teams

Plans

Routines

Van de Ven, A. H., Delbecq, A. L., & Koenig Jr, R. (1976). Determinants of coordination modes within organizations. *American sociological review*, 322-338.
Dingsøyr, T., Bjørnson, F. O., Schrof, J., and Sporsem, T., "A longitudinal explanatory case study of coordination in a very large development programme: the impact of transitioning from a first- to a second-generation large-scale agile development method," *Empirical Software Engineering*, vol. 28, p. 49, 2022/11/08 2023. 10.1007/s10664-022-10230-6. https://rdcu.be/c3FQ4