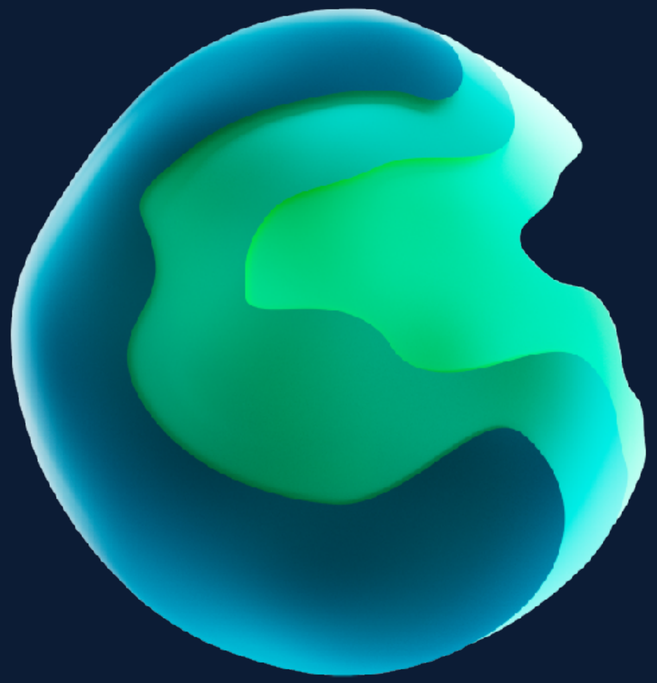


Capra

Happy people in modern and humane organisations create quality digital products

Capra

Practical use of Open Systems Theory

in the Software Industry

A Patchwork of Contradictions and Confusions: Inside the Software Industry

Merrelyn Emery

January 2023

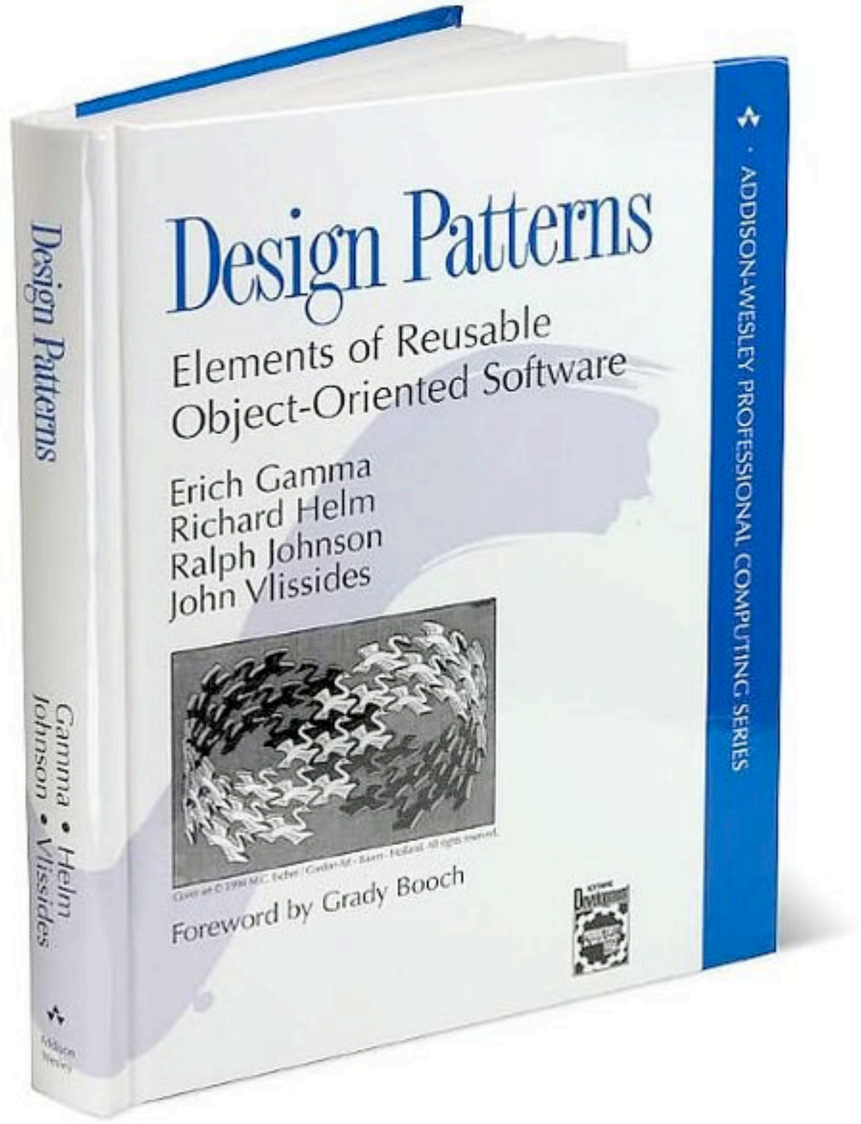
In 2021 a series of conversations took place in which concerns were raised about the software industry and what was happening to personnel in various sections of it. One of those concerned had learnt of Open Systems Theory with its genotypical design principles underlying organizational redesign and wondered whether such an option could help with analysis or present a solution if such was required. That one was Trond Hjorteland and my thanks go to him. He not only initiated these discussions, he started diffusing the concepts and involving others. He also did all the technical work on the survey to make it possible. His efforts also led to an introduction to OST course being held in Canberra. None of that would have been possible without him.

```java
/**
 * Method for testing the equality of two objects. No other modifications other than those default (see
 * {@link ObjectComparison}). The method are recursive.
 *
 * @param o1 Object 1
 * @param o2 Object 2
 * @return boolean, i.e. true or false
 * @throws InvocationTargetException
 * @throws IllegalAccessException
 */
public static boolean areEqual(Object o1, Object o2) throws InvocationTargetException, IllegalAccessException {
    if (o1 == null && o2 == null) {
        return true;
    } else if (o1 == null || o2 == null) {
        LOG.debug("Field mismatch, comparing " +
                (o1 != null ? o1.getClass().getName() : o2.getClass().getName()) + " with null value.");
        return false;
    }
    if (o1.getClass().isInstance(o2)) {
        ComparisonStrategy strategy = getComparisonStrategy(o1);
        if (strategy != null) {
            boolean result = strategy.checkEquality(o1, o2);
            if (!result) {
                LOG.debug("Field mismatch between " + o1.getClass().getName() + "=" + o1 + " and " +
                        o2.getClass().getName() + "=" + o2 + ".");
            }
            return result;
        } else {
            throw new RuntimeException("No equality handler was found for " + o1.getClass().getName());
        }
    } else {
        LOG.debug("Field mismatch, " + o1.getClass().getName() + " is not an instance of  " + o2.getClass().getName());
        return false;
    }
}

/**
 * Locates the correct {@link ComparisonStrategy} based on the incoming object.
 *
 * @param o object
 * @return The corresponding {@link ComparisonStrategy}
 */
private static ComparisonStrategy getComparisonStrategy(Object o) {
    ComparisonStrategy result = null;
    Iterator it = COMPARISON_STRATEGIES_LIST.iterator();
    while (it.hasNext() && result == null) {
        ComparisonStrategy strategy = (ComparisonStrategy) it.next();
        if (strategy.isRightStrategy(o)) {
            result = strategy;
        }
    }
    return result;
}
```

@trondhjort

Capra

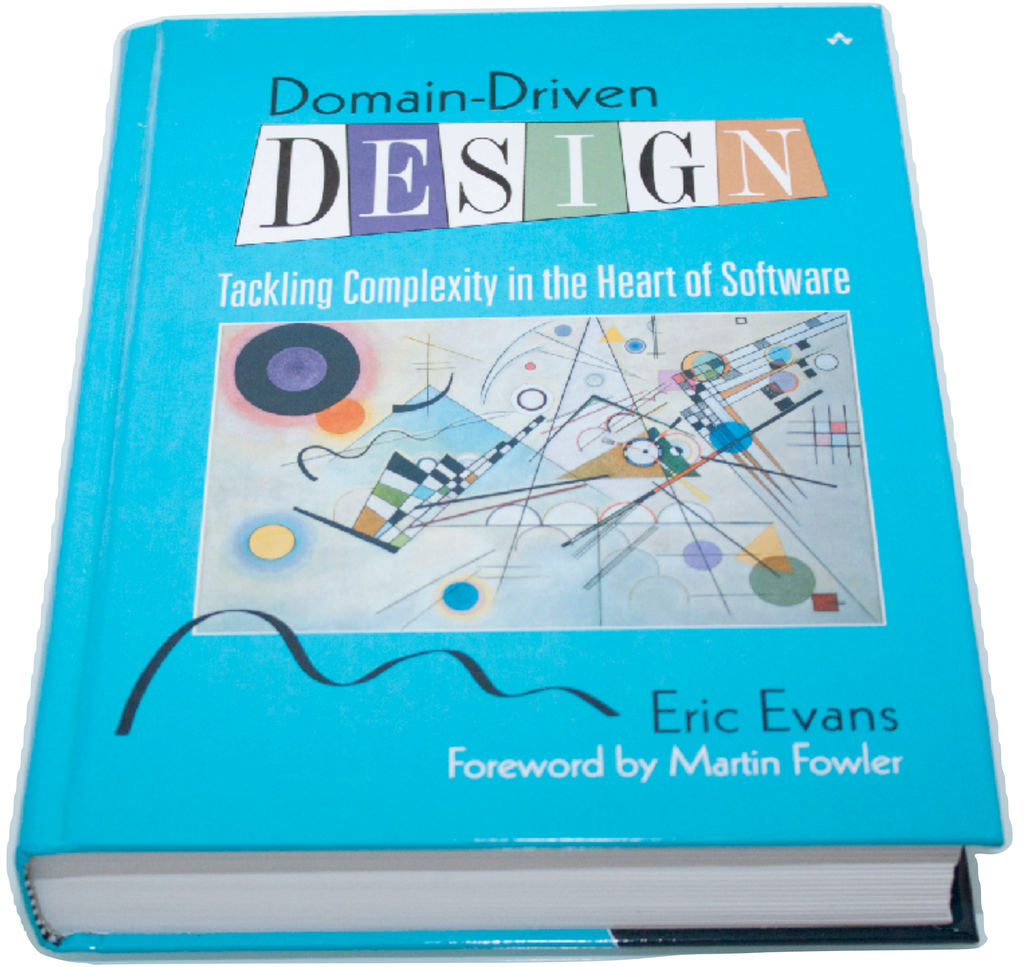
```java
/**
 * Method for testing the equality of two objects. No other modifications other than those default (see
 * {@link ObjectComparison}). The method are recursive.
 *
 * @param o1 Object 1
 * @param o2 Object 2
 * @return boolean, i.e. true or false
 * @throws InvocationTargetException
 * @throws IllegalAccessException
 */
public static boolean areEqual(Object o1, Object o2) throws InvocationTargetException, IllegalAccessException {
    if (o1 == null && o2 == null) {
        return true;
    } else if (o1 == null || o2 == null) {
        LOG.debug("Field mismatch, comparing " +
                (o1 != null ? o1.getClass().getName() : o2.getClass().getName()) + " with null value.");
        return false;
    }
    if (o1.getClass().isInstance(o2)) {
        ComparisonStrategy strategy = getComparisonStrategy(o1);
        if (strategy != null) {
            boolean result = strategy.checkEquality(o1, o2);
            if (!result) {
                LOG.debug("Field mismatch between " + o1.getClass().getName() + "=" + o1 + " and " +
                        o2.getClass().getName() + "=" + o2 + ".");
            }
            return result;
        } else {
            throw new RuntimeException("No equality handler was found for " + o1.getClass().getName());
        }
    } else {
        LOG.debug("Field mismatch, " + o1.getClass().getName() + " is not an instance of  " + o2.getClass().getName());
        return false;
    }
}

/**
 * Locates the correct {@link ComparisonStrategy} based on the incoming object.
 *
 * @param o object
 * @return The corresponding {@link ComparisonStrategy}
 */
private static ComparisonStrategy getComparisonStrategy(Object o) {
    ComparisonStrategy result = null;
    Iterator it = COMPARISON_STRATEGIES_LIST.iterator();
    while (it.hasNext() && result == null) {
        ComparisonStrategy strategy = (ComparisonStrategy) it.next();
        if (strategy.isRightStrategy(o)) {
            result = strategy;
        }
    }
    return result;
}
```

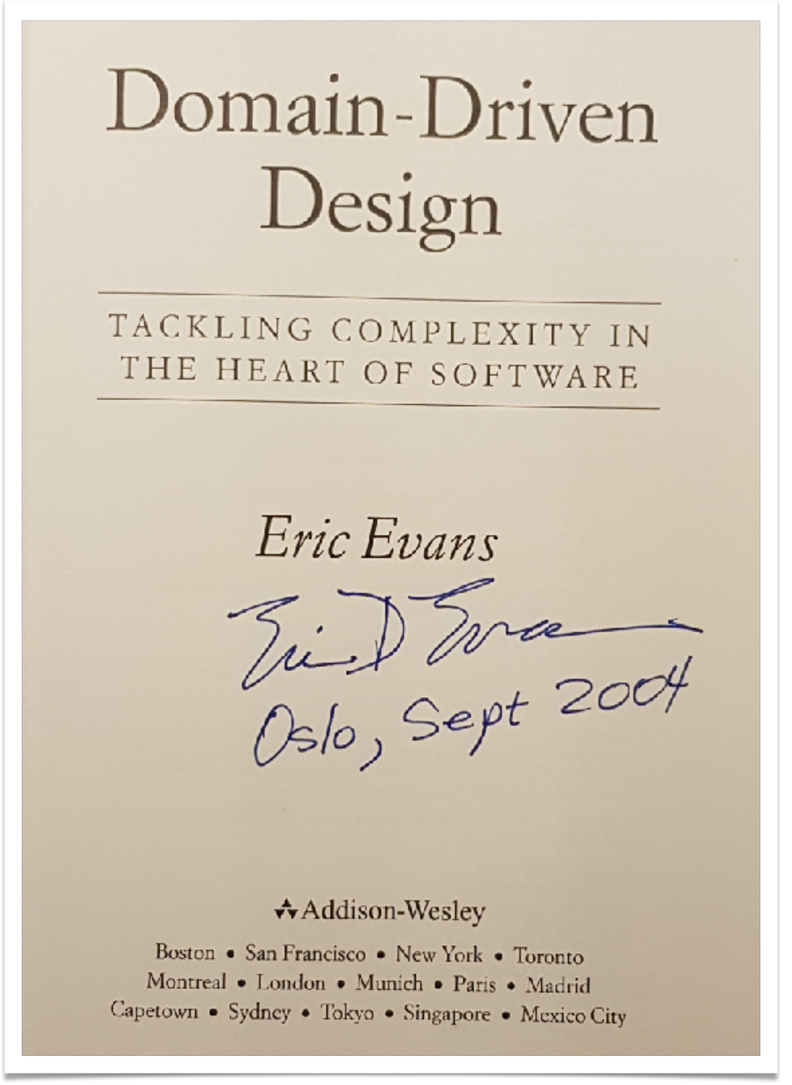


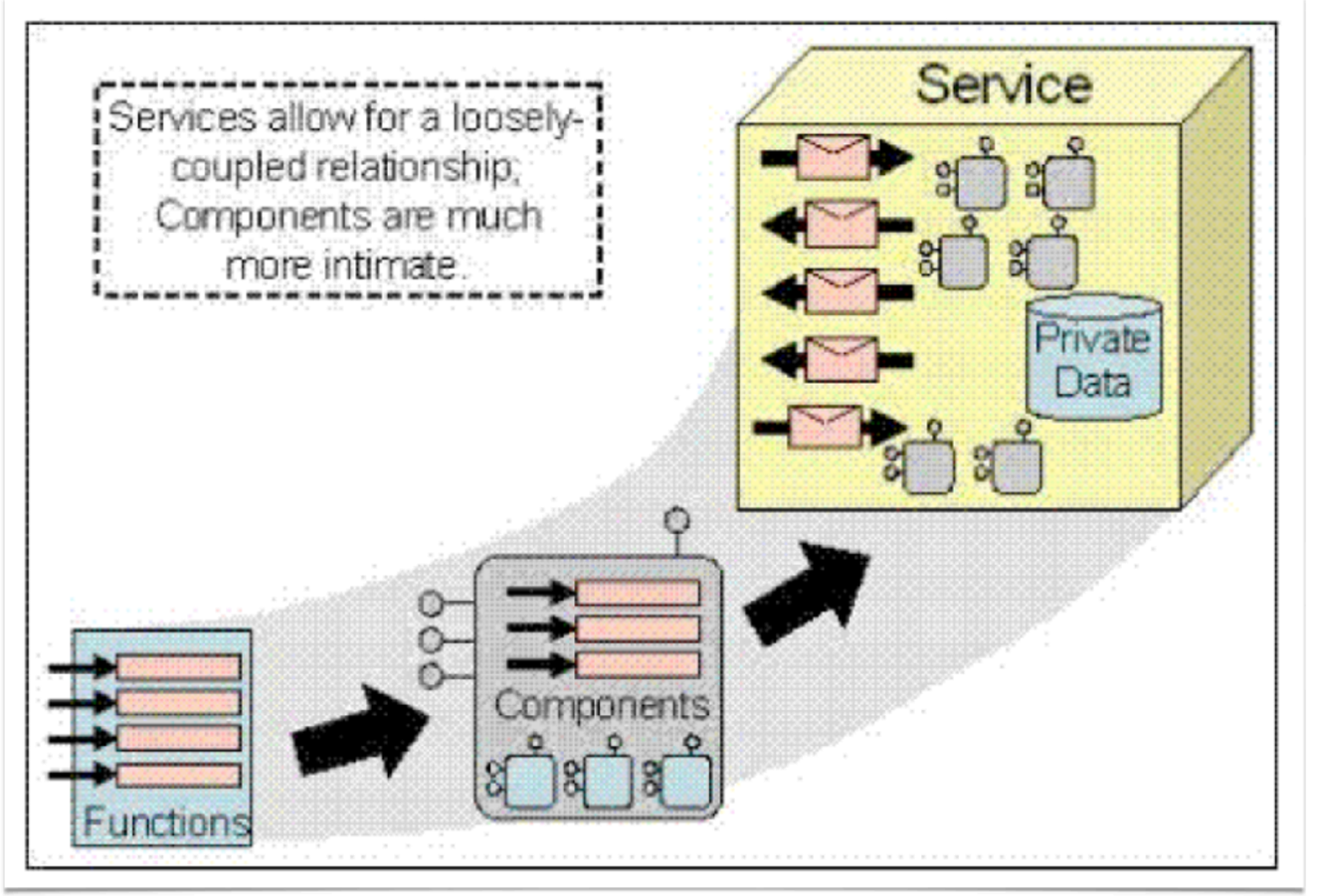
@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra







@trondhjort

Source: "Data on the Outside vs. Data on the Inside," Pat Helland, MSDN 2004

scienta

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

Hvem er vi?

- Trond Hjorteland
- @trondhjort

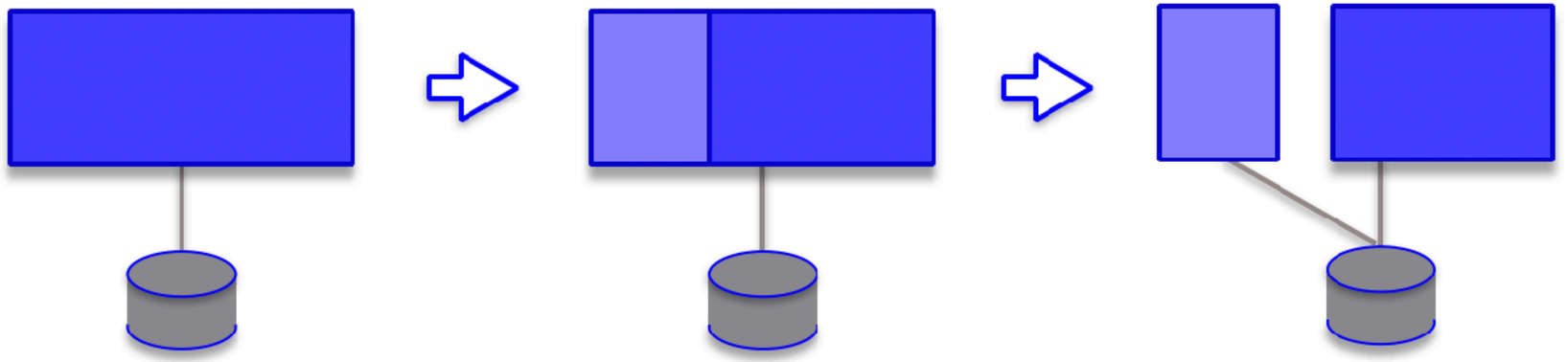
- Nils Christian Haugen
- @nchaugen

scienta

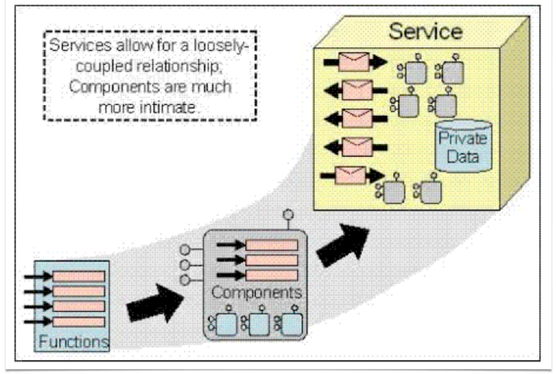
STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

Stegvis splitting



scienta



@trondhjort

Source: "Data on the Outside vs. Data on the Inside," Pat Helland, MSDN 2004

scienta

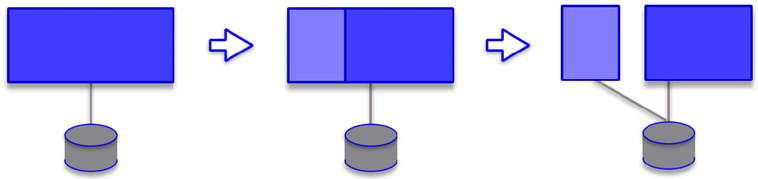
Hvem er vi?

- Trond Hjorteland
- @trondhjort

- Nils Christian Haugen
- @nchaugen

scienta

Stegvis splitting



scienta

STS RT SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

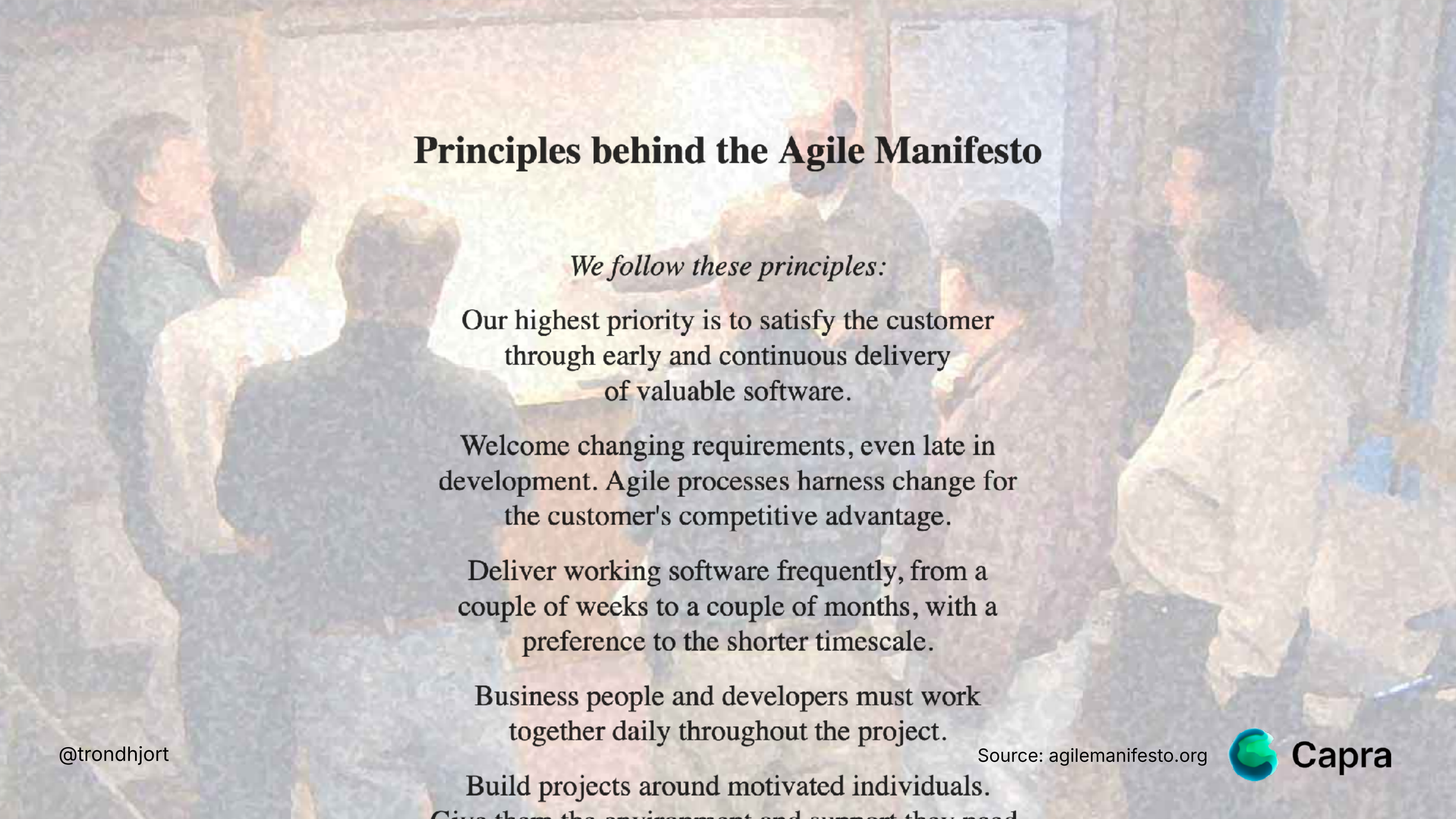
Capra

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.



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 maintain a constant pace indefinitely.

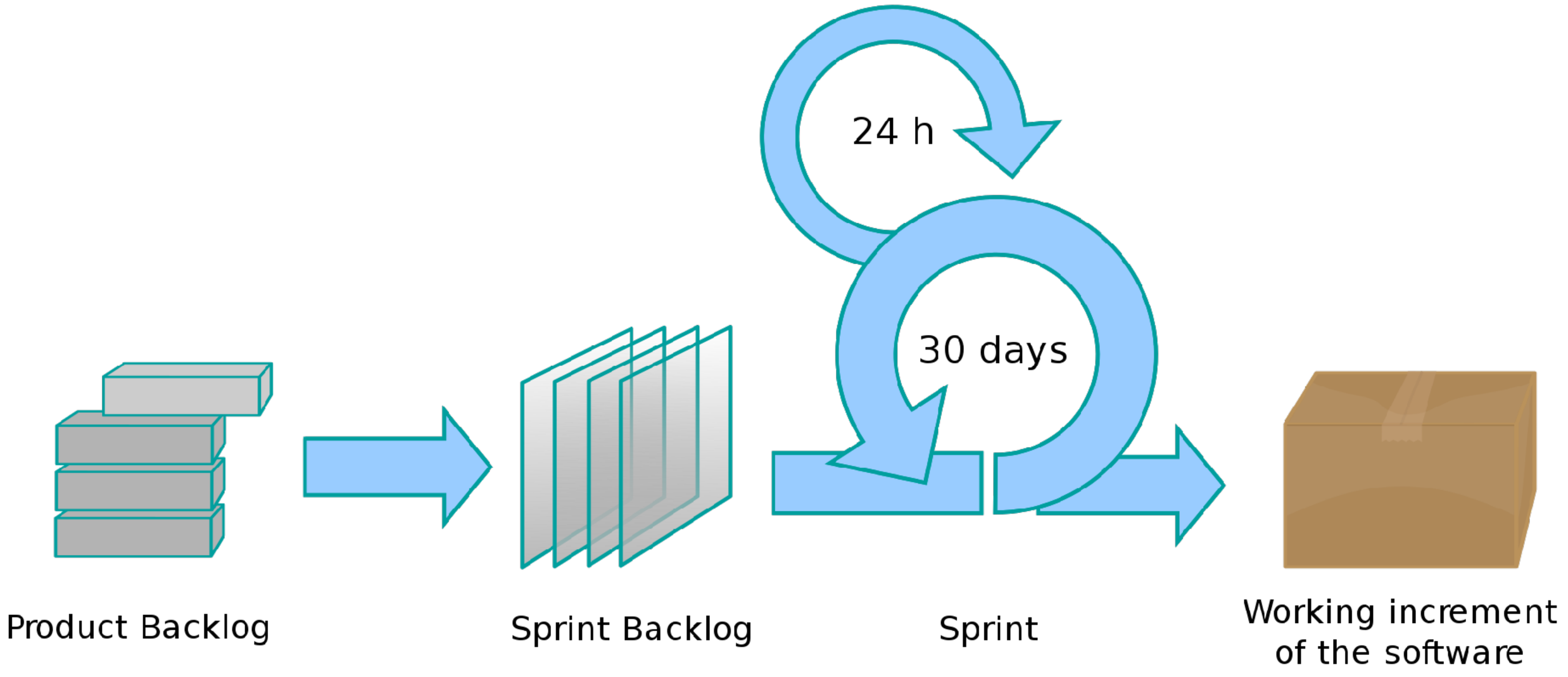
Continuous attention to technical excellence
and good design enhances agility.

Simplicity--the art of maximizing the amount
of work not done--is essential.

**The best architectures, requirements, and designs
emerge from self-organizing teams.**

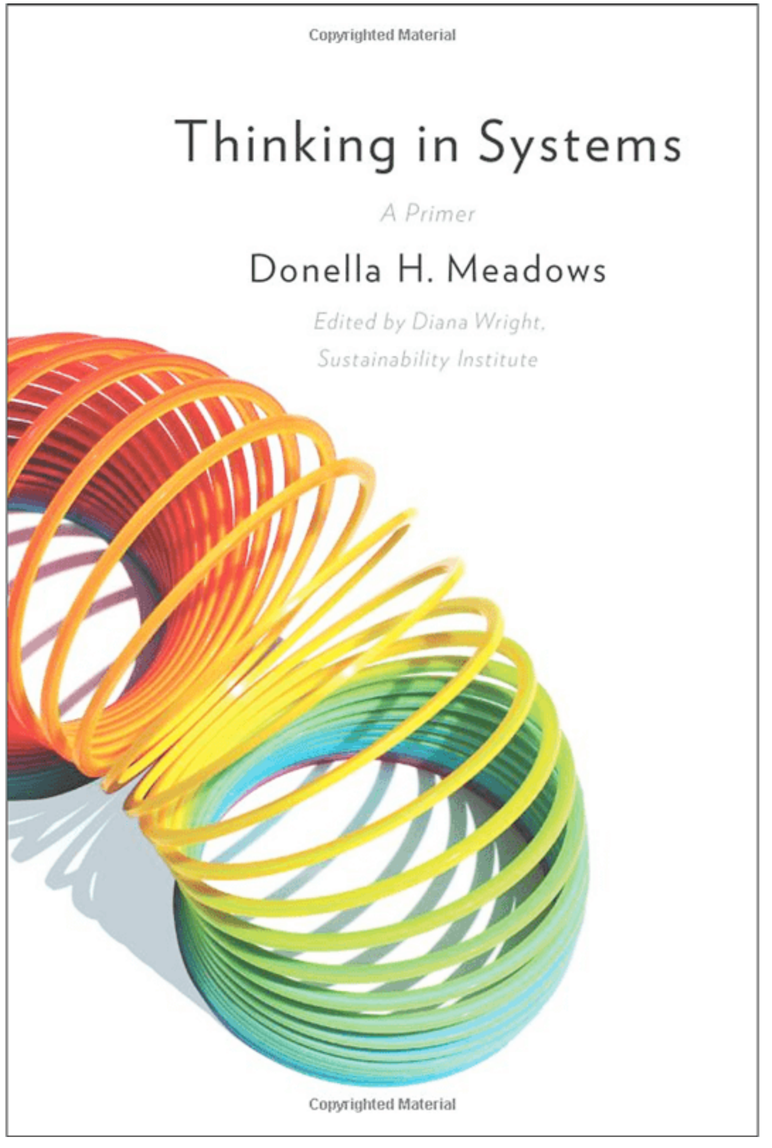
At regular intervals, the team reflects on how
to become more effective, then tunes and adjusts
its behavior accordingly.

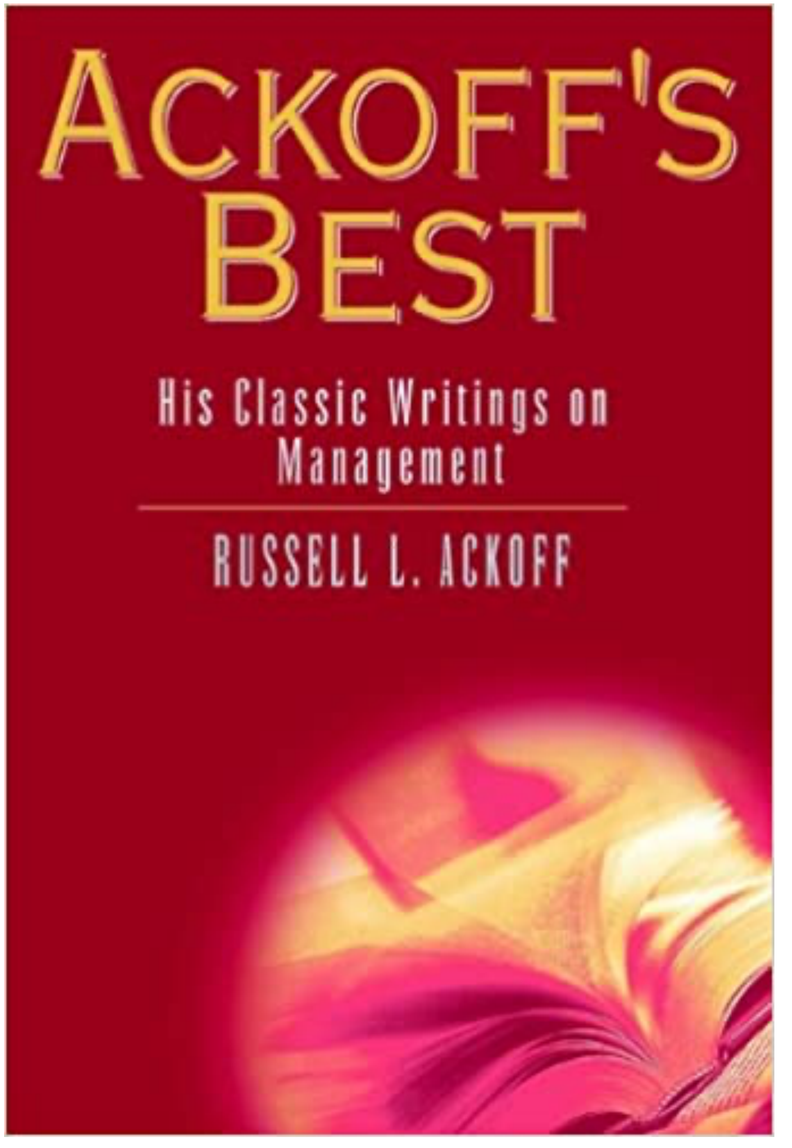
Source: agilemanifesto.org



STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

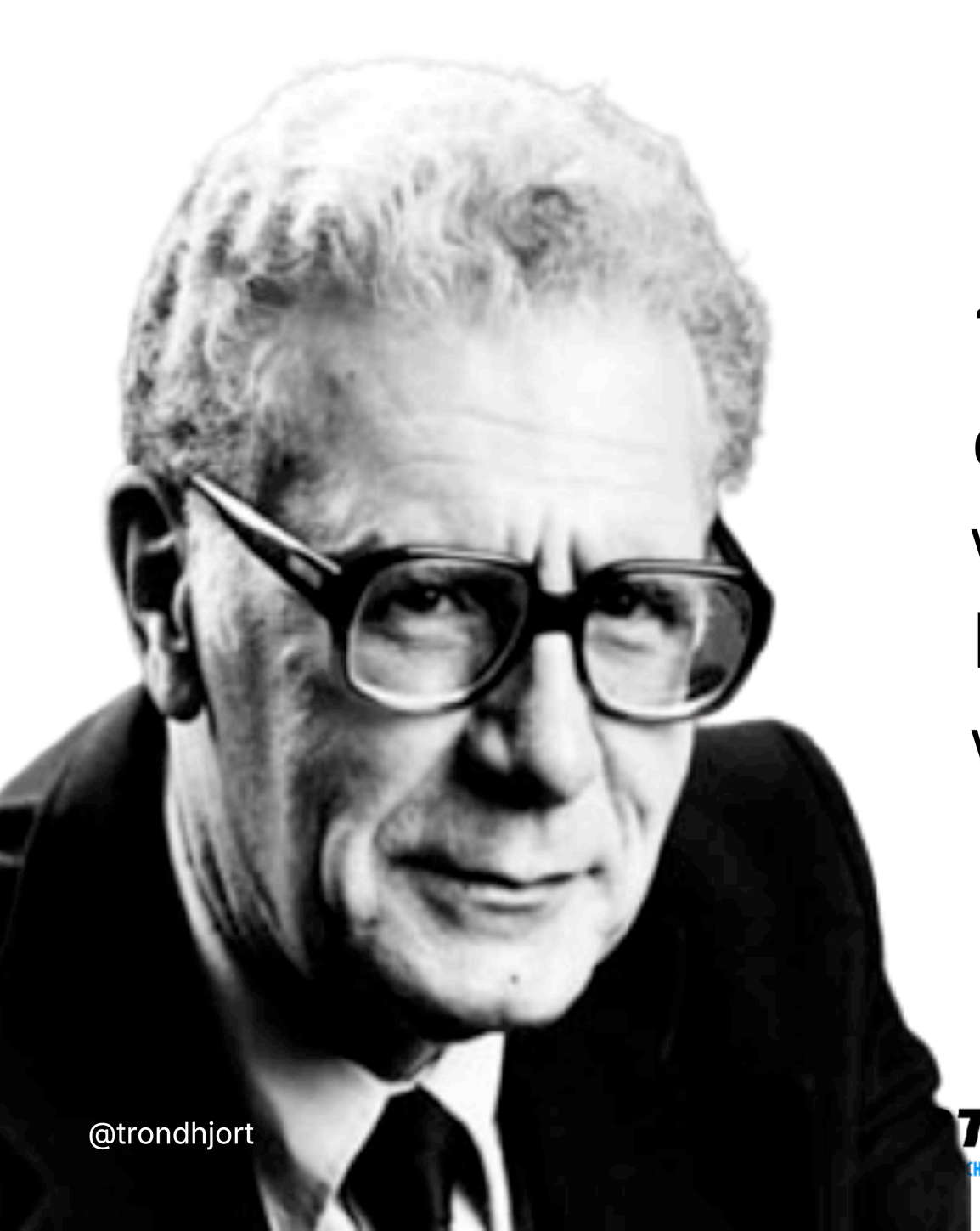
Capra





STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra



@trondhjort

"A system is more than the sum of its parts; it is an indivisible whole.
It loses its essential properties when it is taken apart."

-Russell L. Ackoff

TS RT
TECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

"...organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations."

-Mel Conway, 1968

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

Medium Search Get app Write Sign up Sign in

Nick Tune's weird ideas

An Introduction to Sociotechnical Architecture Patterns

Nick Tune Follow 7 min read · Mar 28, 2018

161 2

@trondhjort

Capra

Software systems are inextricably linked to the organisation that builds them, aka the system of work. By co-designing and co-evolving the software system and the system of work, modern organisations gain competitive advantage through faster iterations and higher quality discovery cycles.

As a result, we must all think of ourselves as sociotechnical architects. When we are architecting a software system, we must consider the impact on the teams in the organisation and vice-versa.

A lackadaisical design choice in either the system of work or the system of software can introduce severe bottlenecks that cause weeks of lost effort and exasperating political wrangling. I have the mental scars to prove it, and I'm sure you have as well.

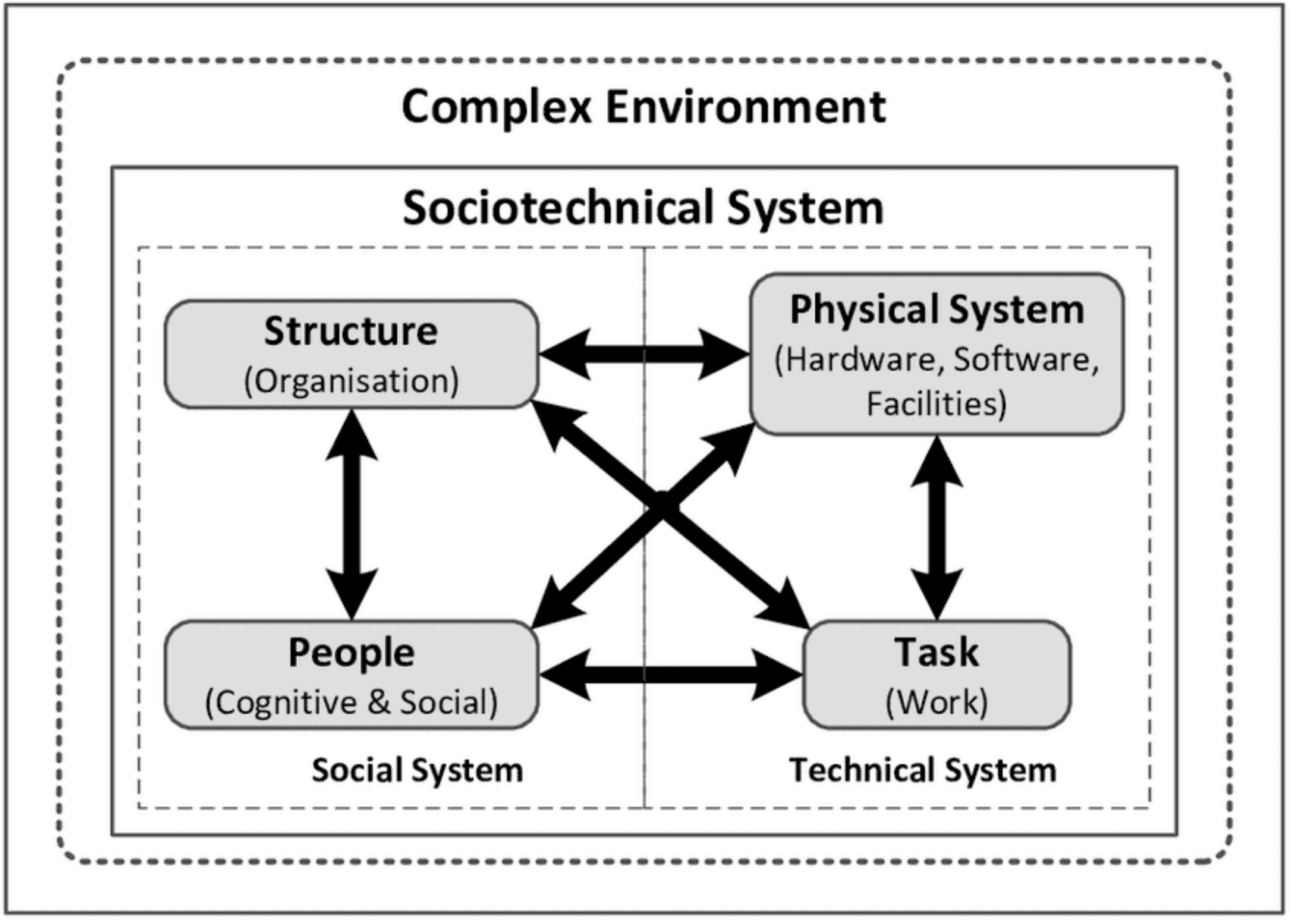
We can avoid these pains and consciously design complex sociotechnical systems through the use of patterns. The patterns help us to understand the challenges of an existing organisation and help us to design and evolve new sociotechnical systems.

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra


JERN-ARBEIDERNE KREVER
ØKT DEMOKRATI i SAMFUNN
OG PÅ ARBEIDSPLASSEN!

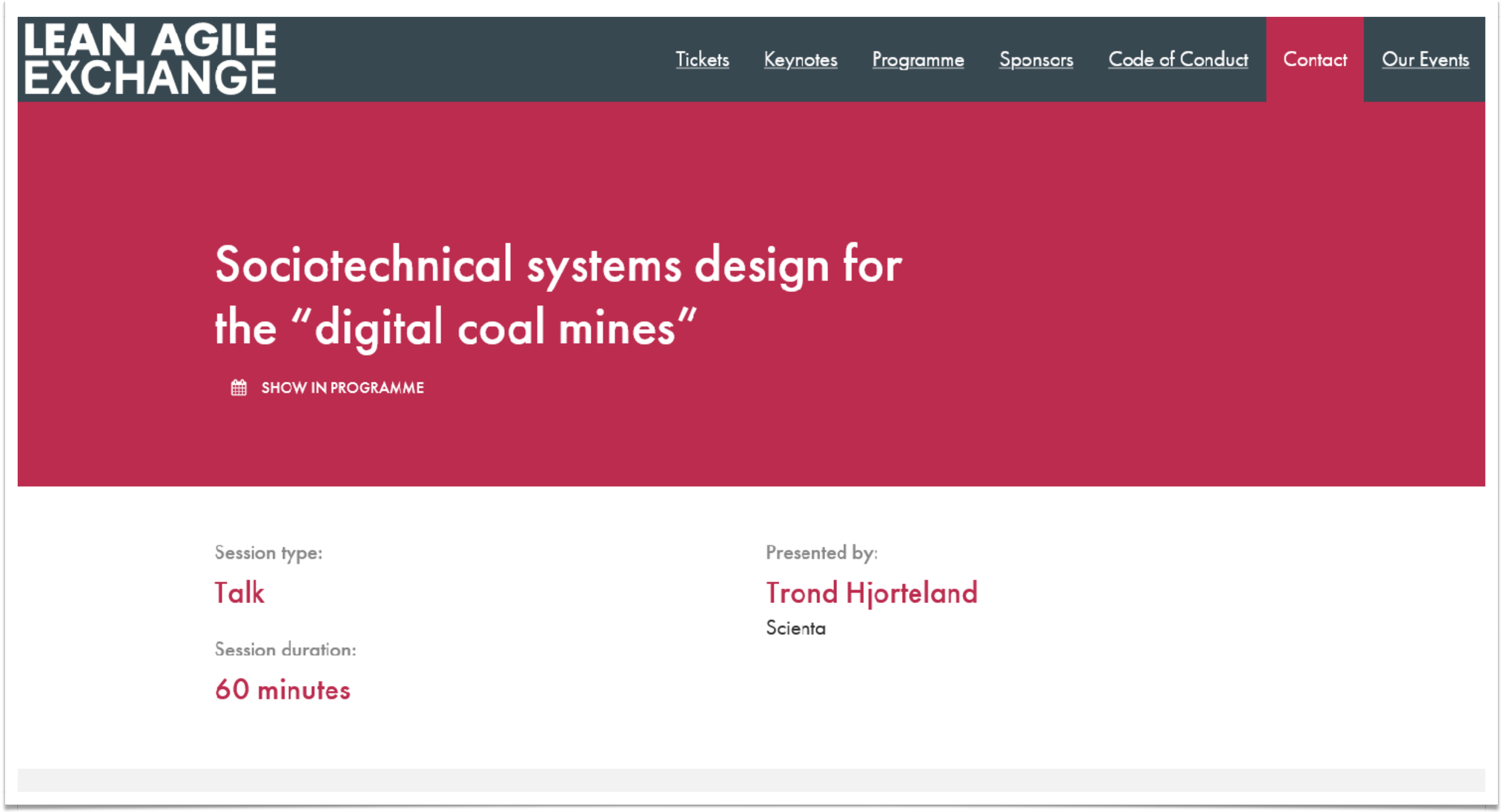


@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: Oosthuizen & Pretorius, Assessing the impact of new technology on complex sociotechnical systems

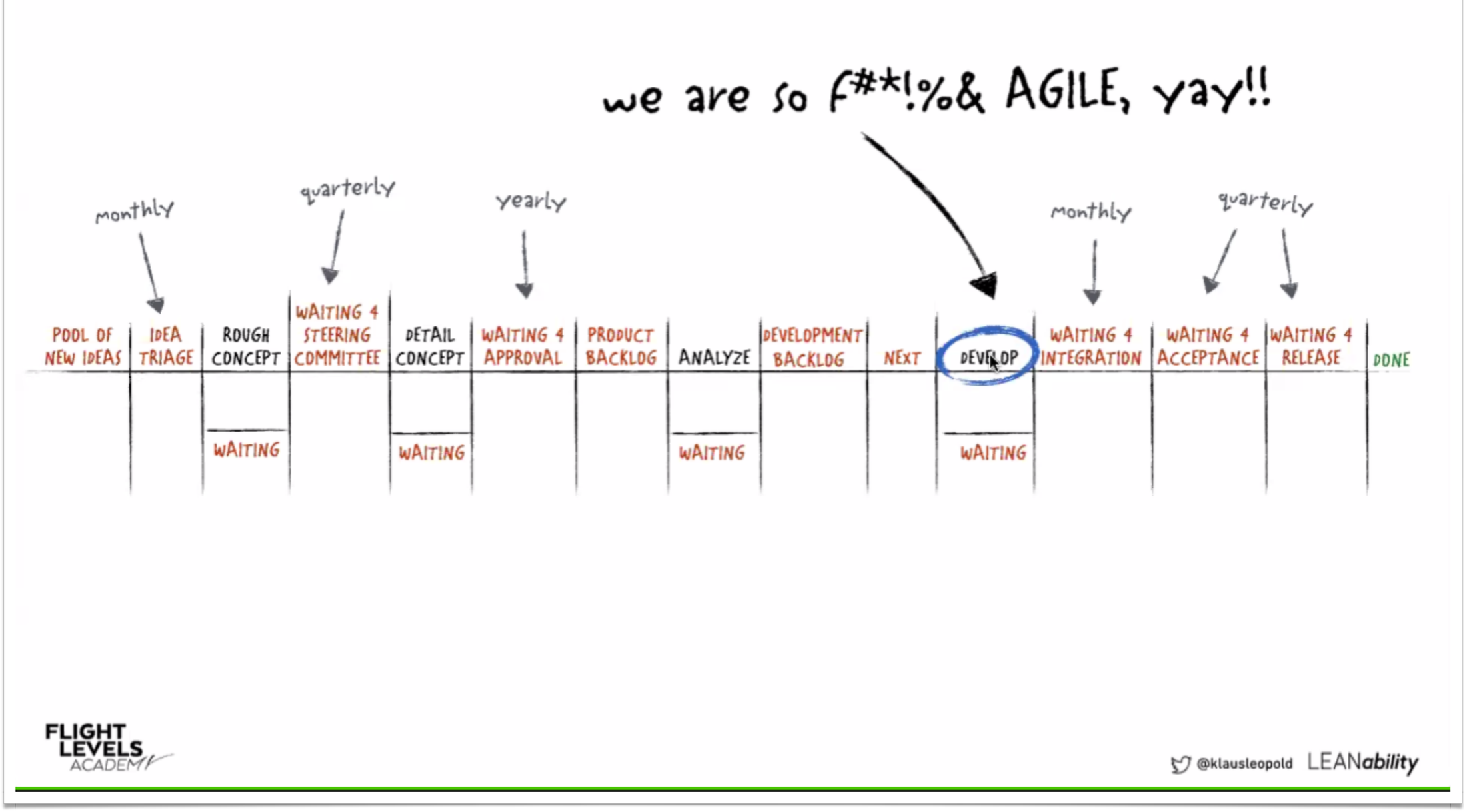
Capra



@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

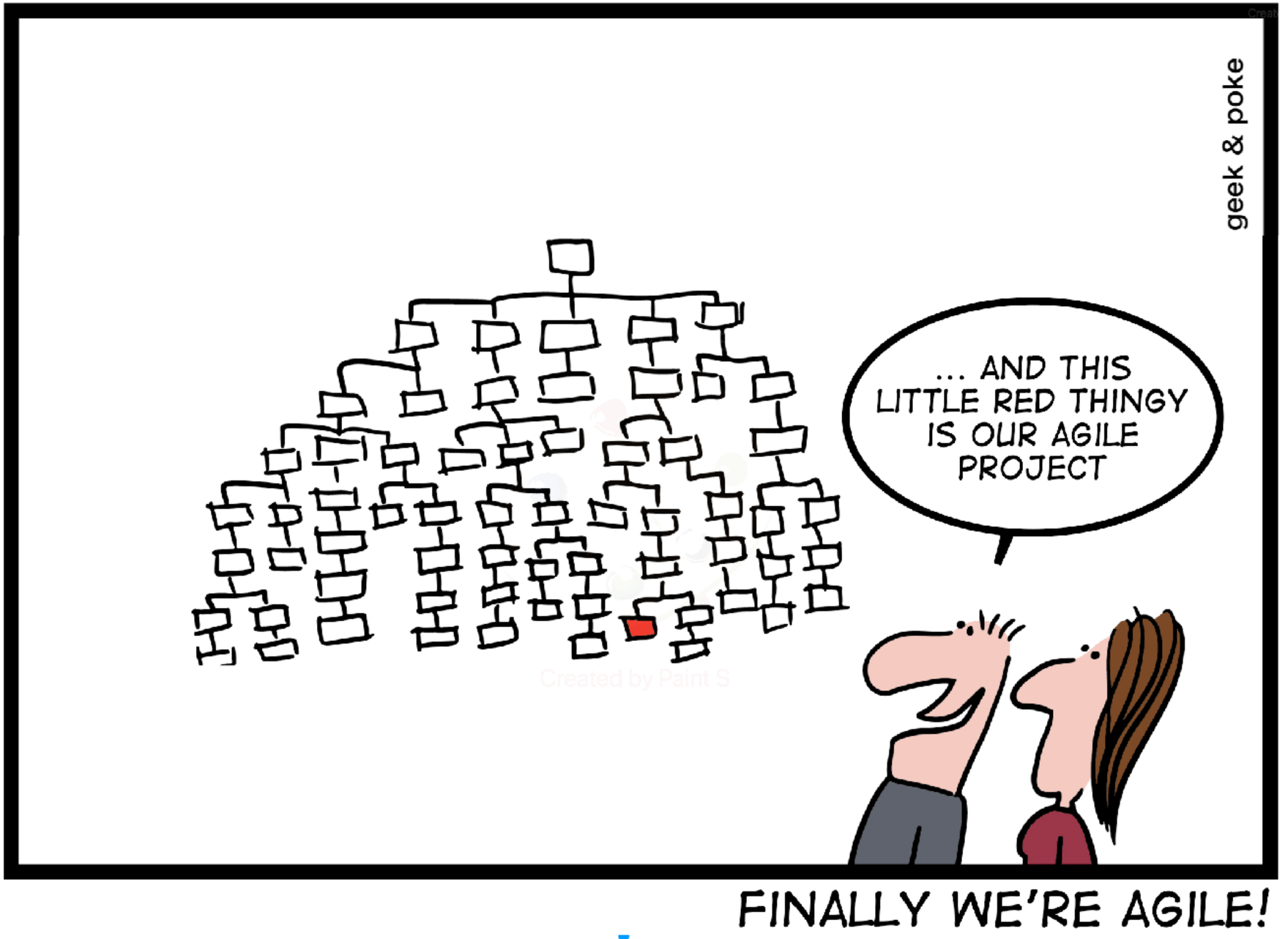
Capra



STS RT SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: Klaus Leopold, “Rethinking Agile: Why Agile Teams Have Nothing To Do With Business Agility”

Capra

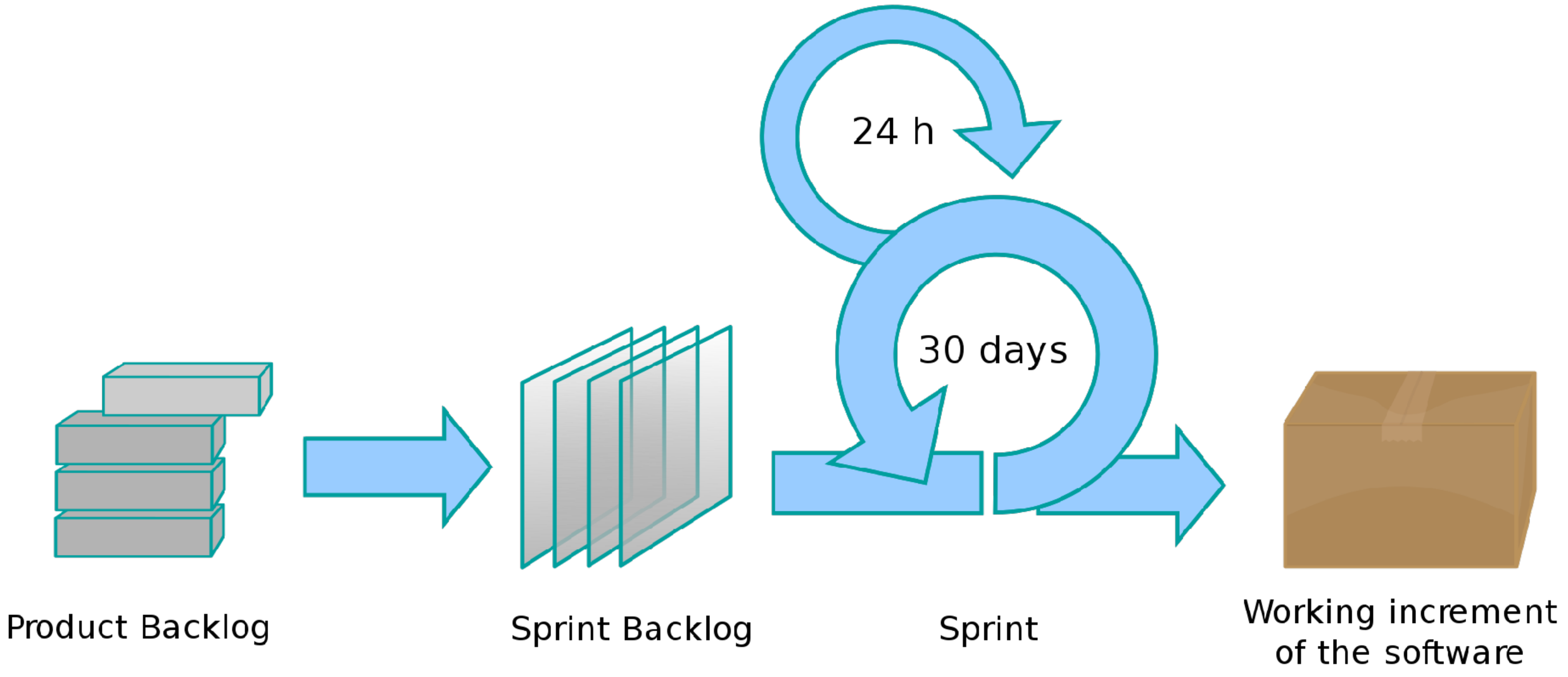


@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: https://geek-and-poke.com/geekandpoke/2016/4/26/finally-agile

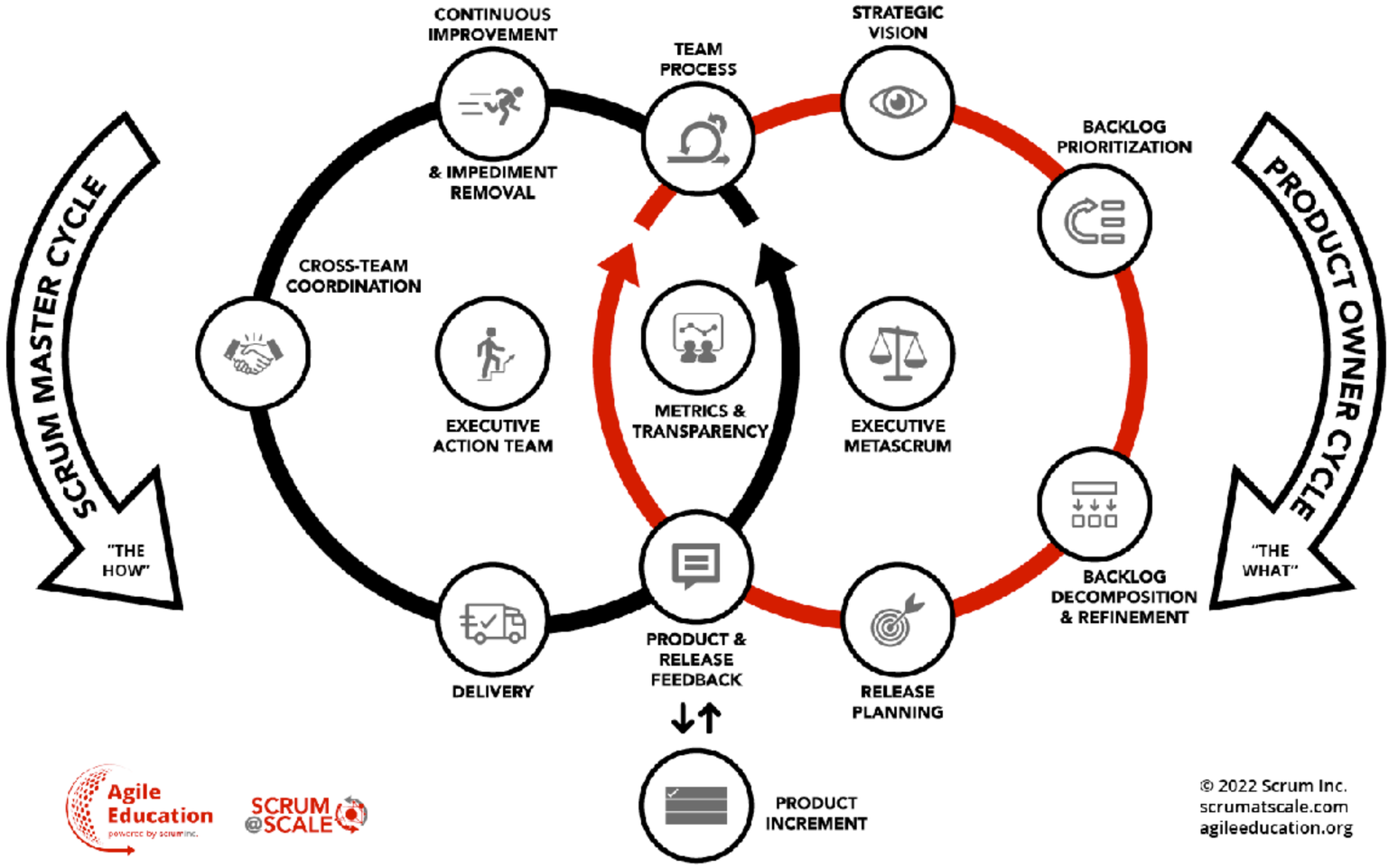
Capra



STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

Scrum@Scale Framework

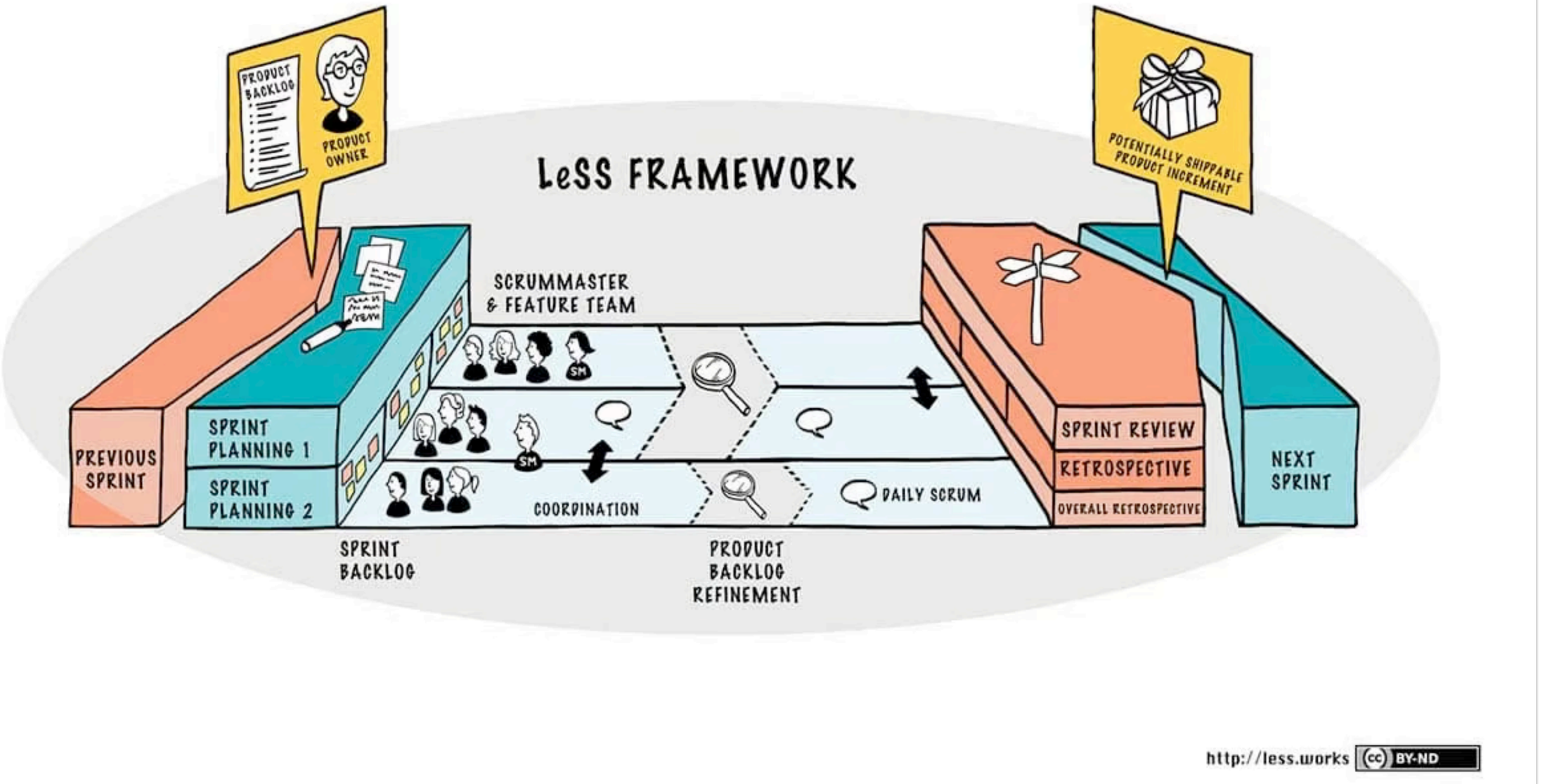


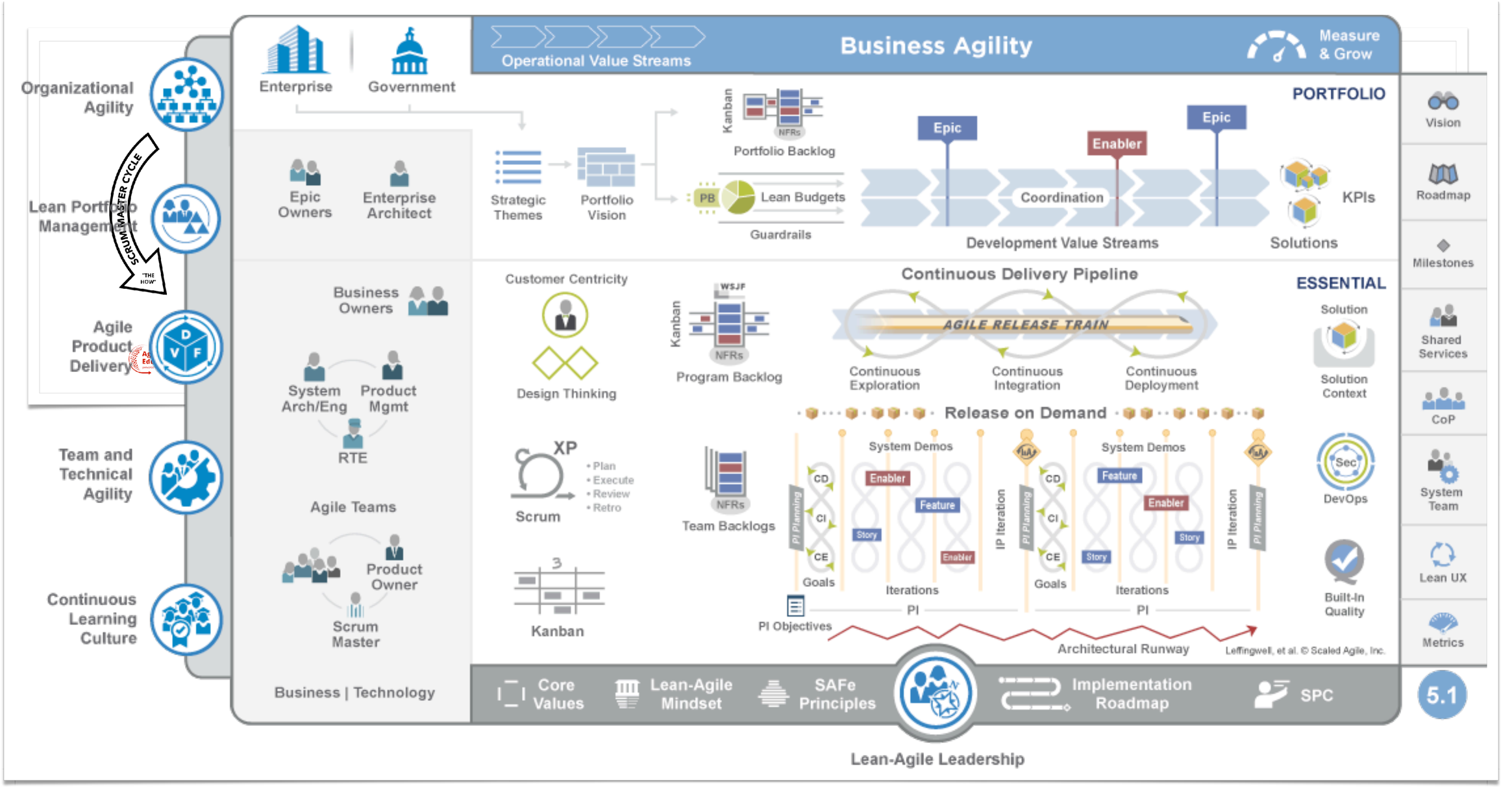
Agile Education powered by scrum inc.

SCRUM @SCALE

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra


LeSS FRAMEWORK
PRODUCT BACKLOG
PRODUCT OWNER
POTENTIALLY SHIPPABLE PRODUCT INCREMENT
SCRUMMASTER & FEATURE TEAM
PREVIOUS SPRINT
SPRINT PLANNING 1
SPRINT PLANNING 2
SM
SM
COORDINATION
DAILY SCRUM
SPRINT REVIEW
RETROSPECTIVE
OVERALL RETROSPECTIVE
NEXT SPRINT
SPRINT BACKLOG
PRODUCT BACKLOG REFINEMENT
http://less.works
BY-ND

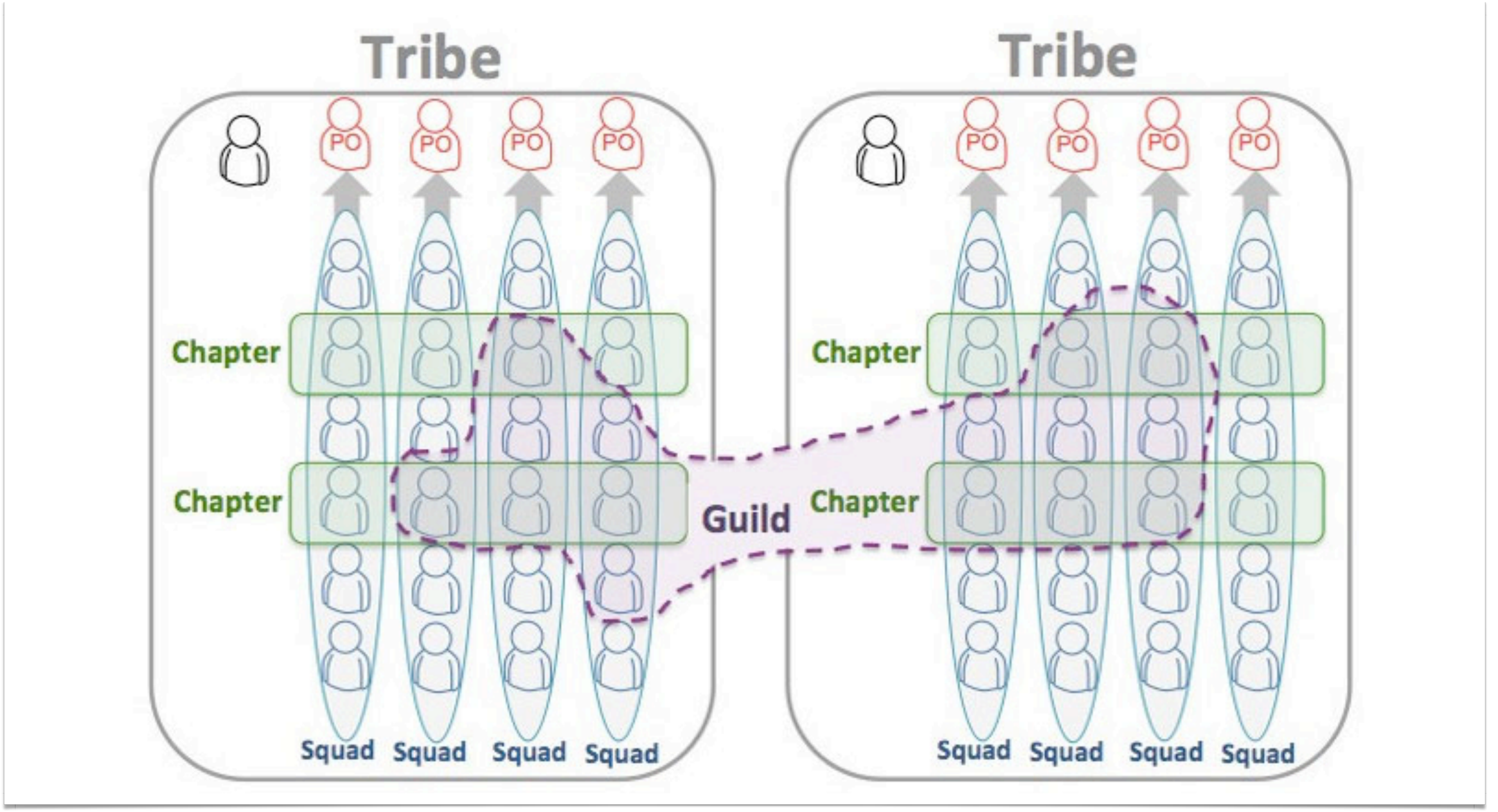


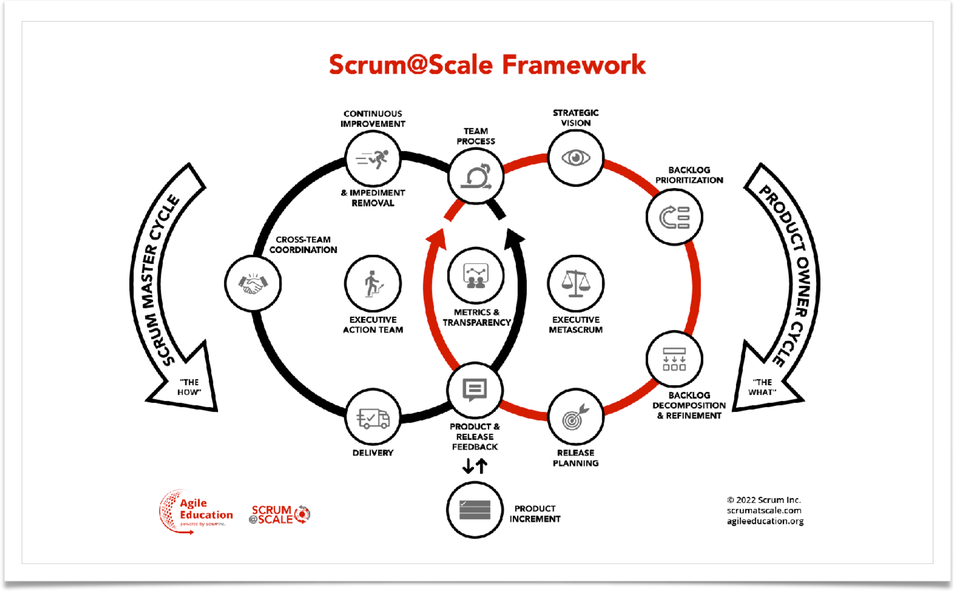
@trondhjort

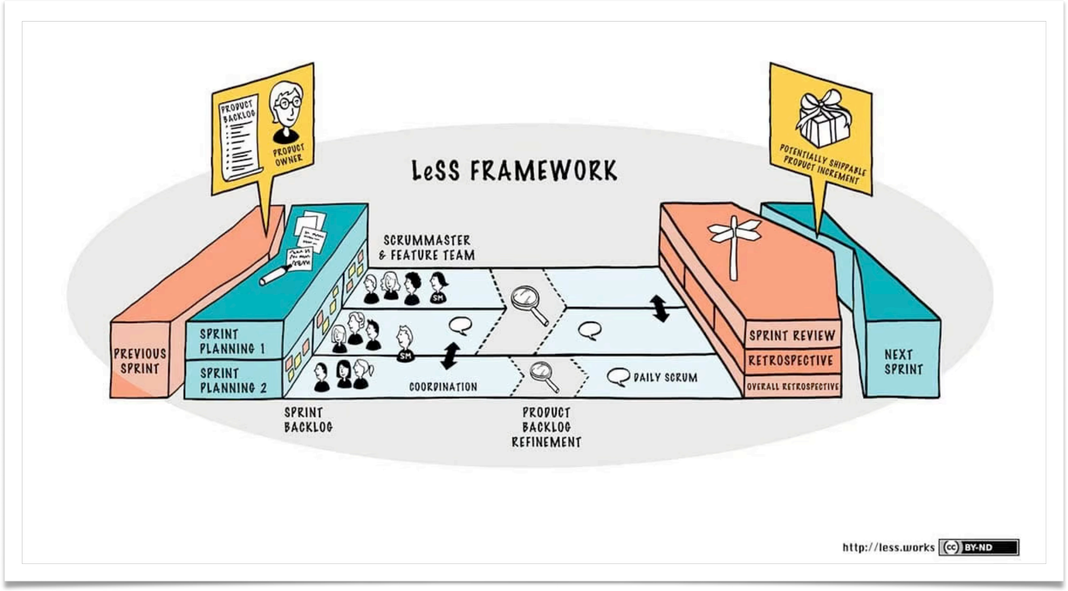
STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

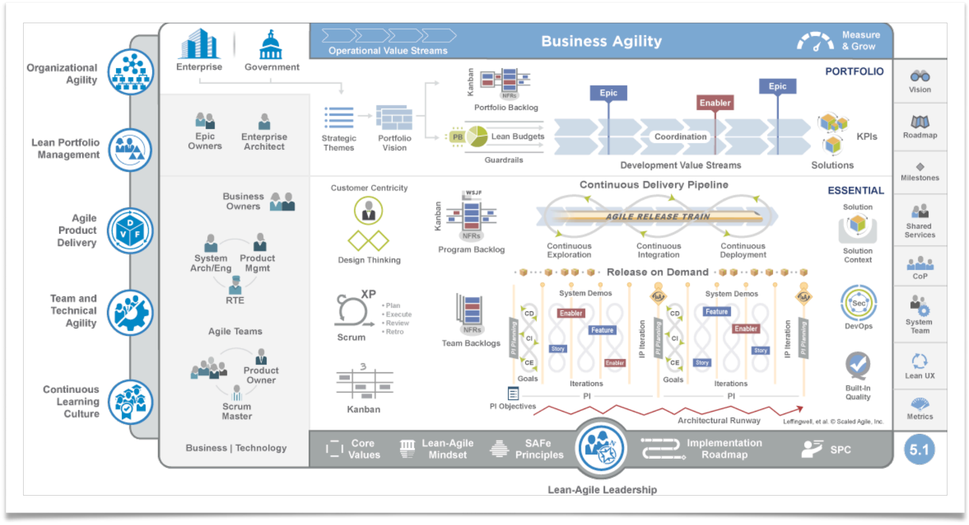
Source: scaledagileframework.com

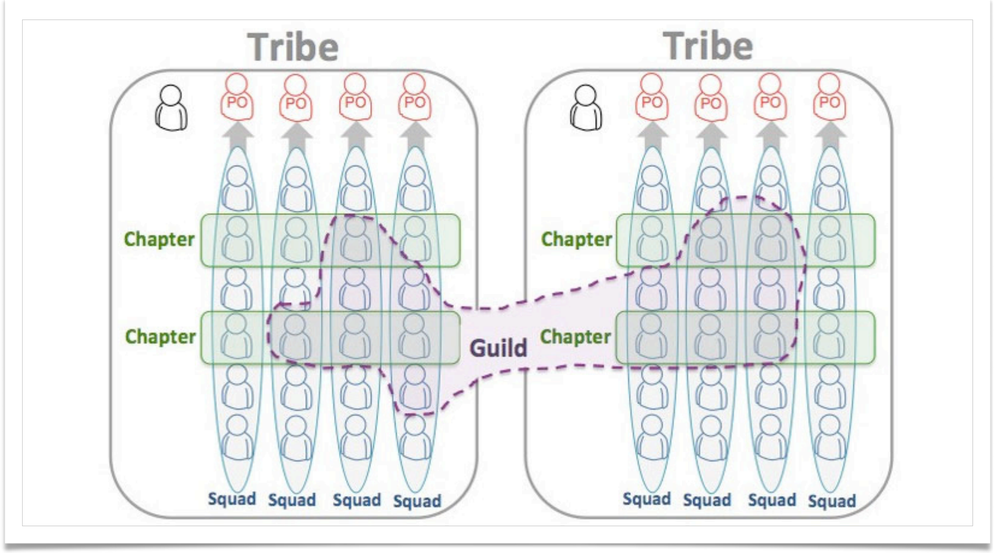
Capra







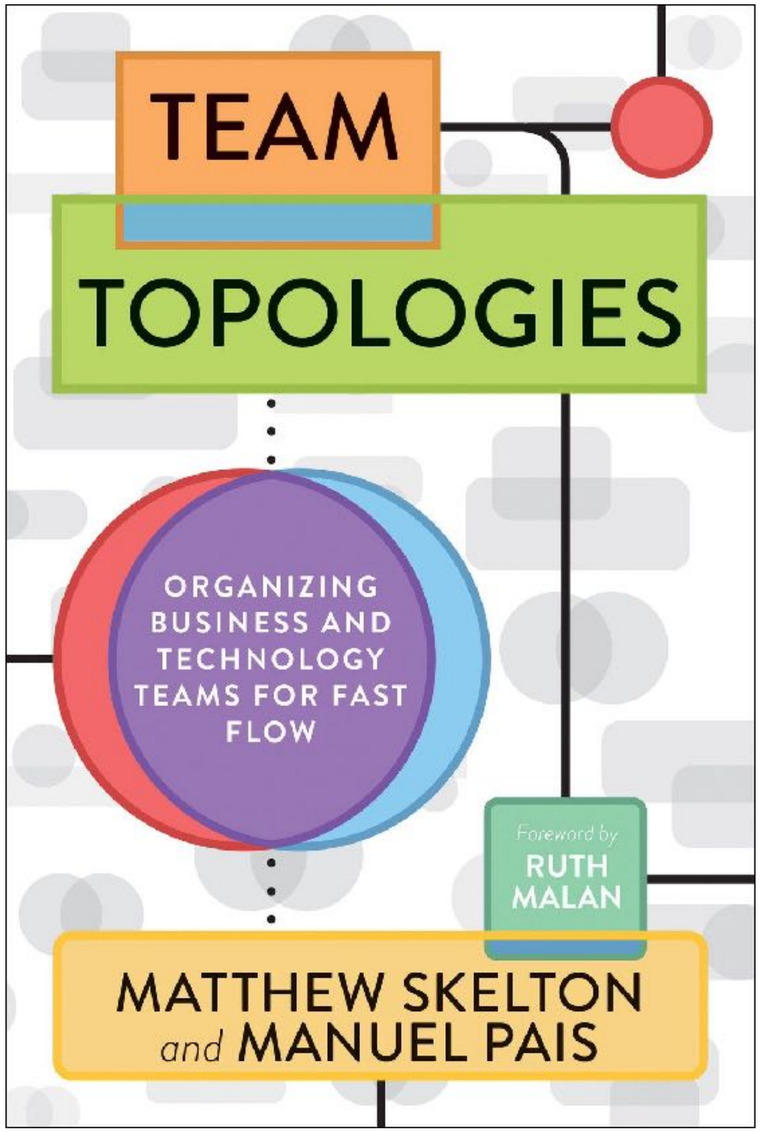




@trondhjort

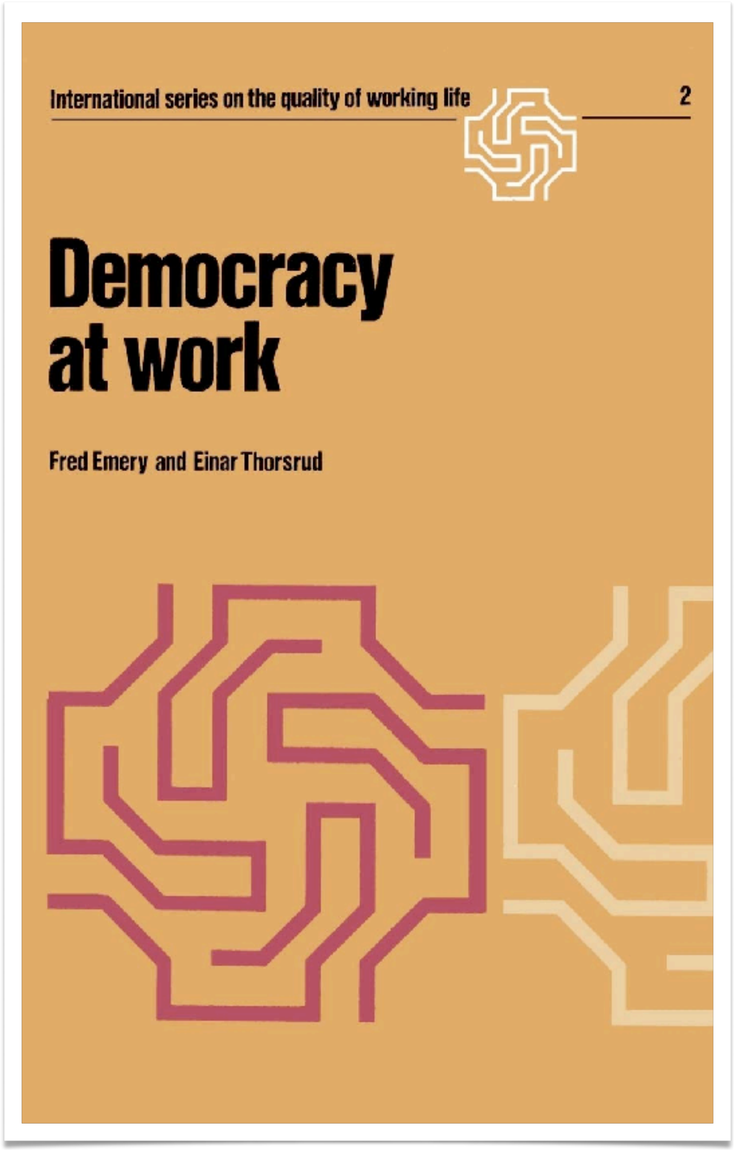
STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra



STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

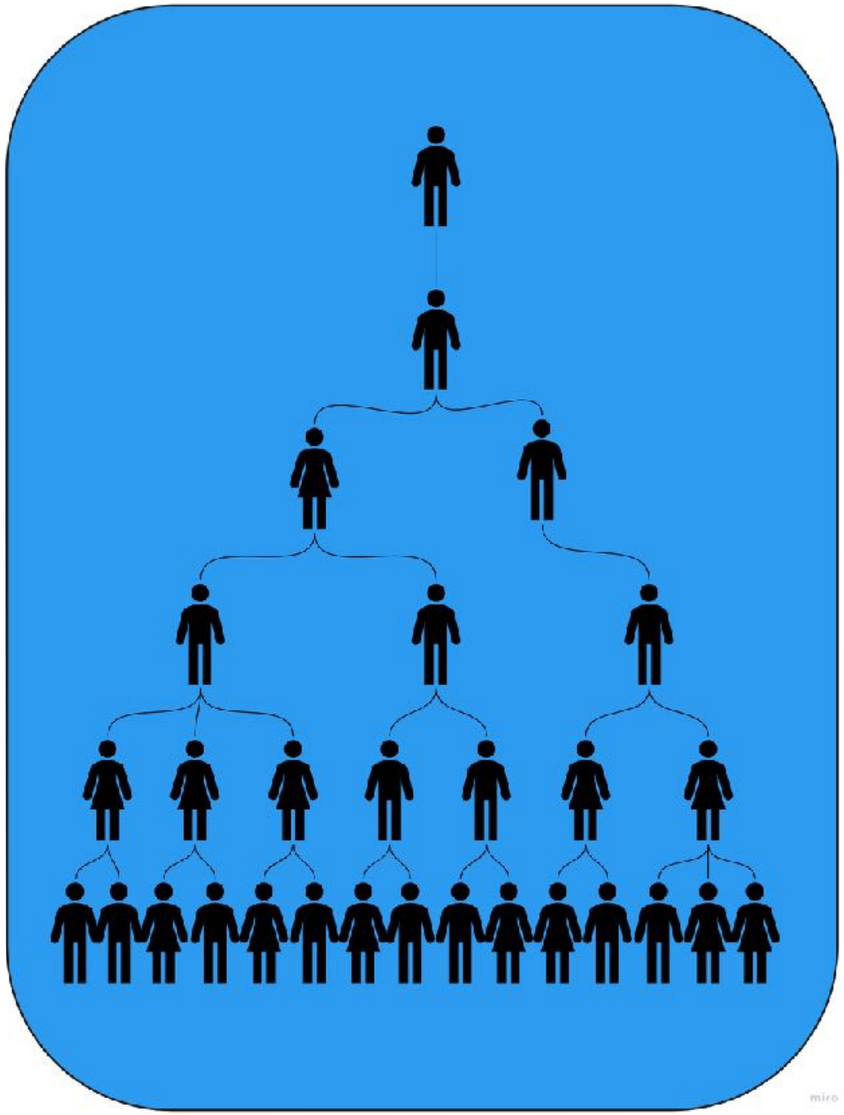
Capra



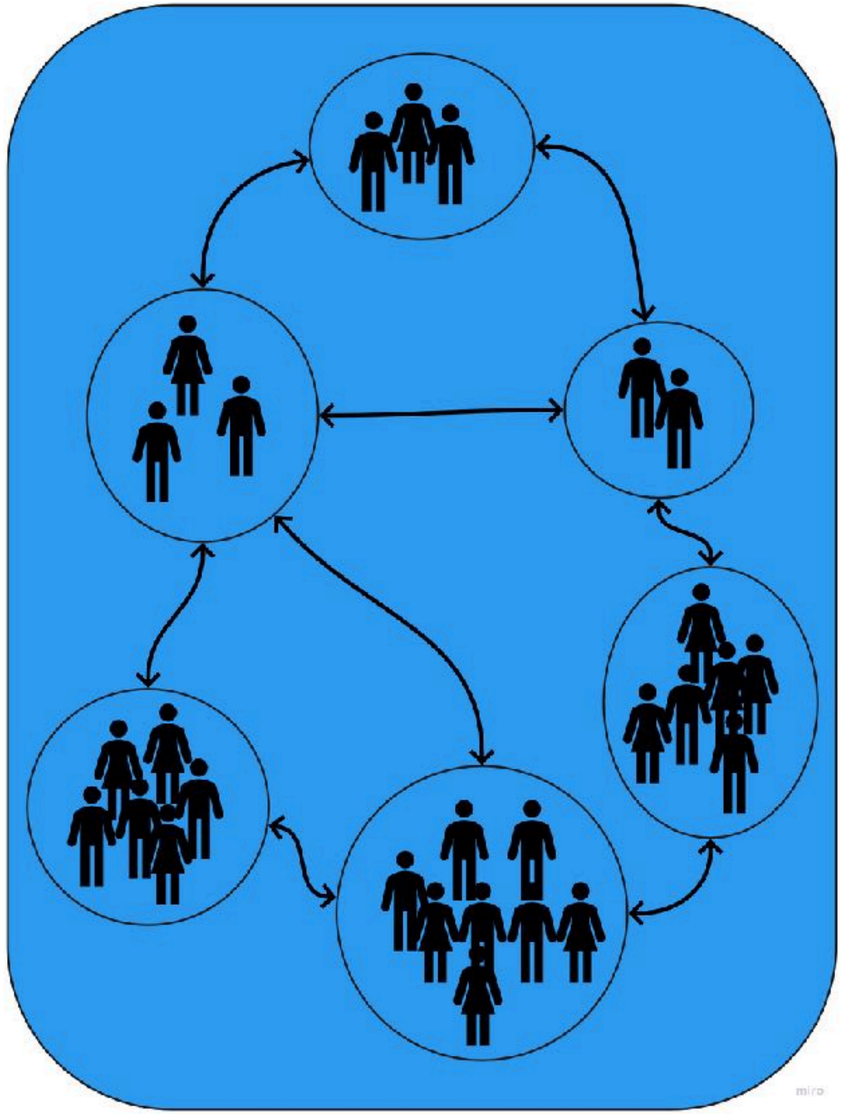
@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra



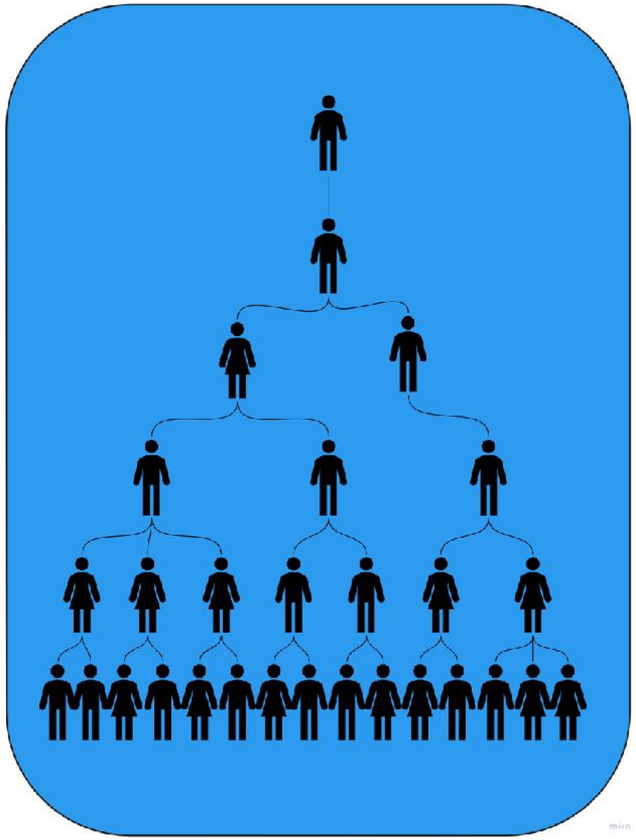
DP1



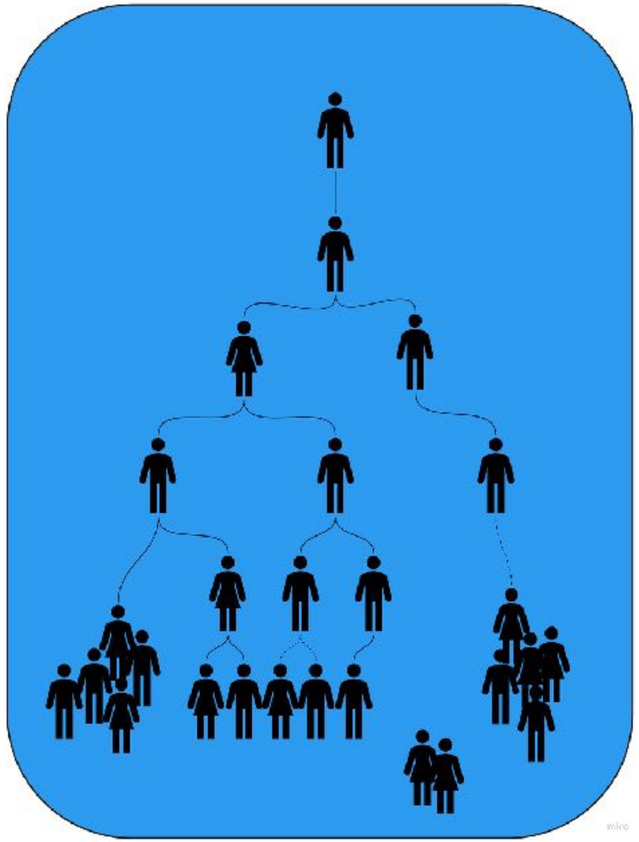
DP2

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

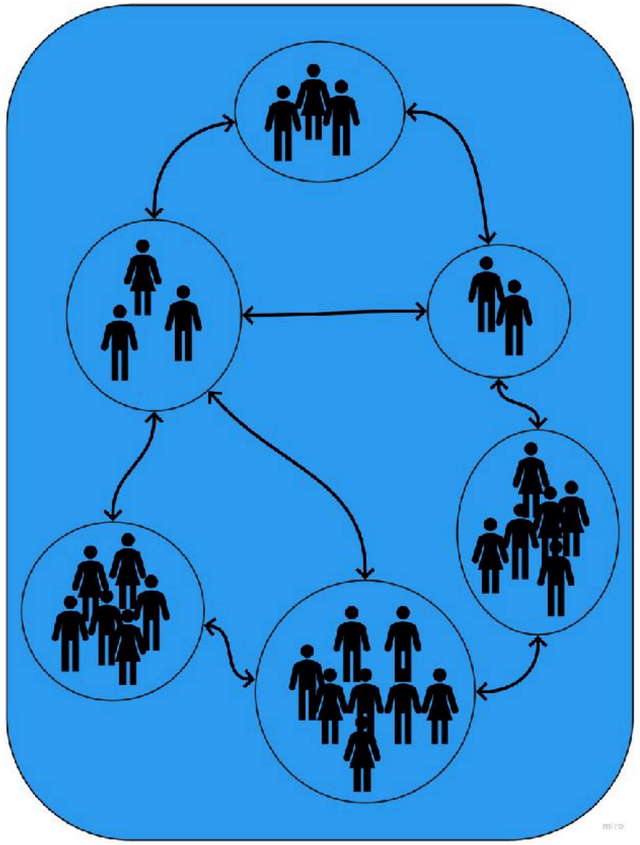


DP1



Laissez-faire

DP1 → DP2

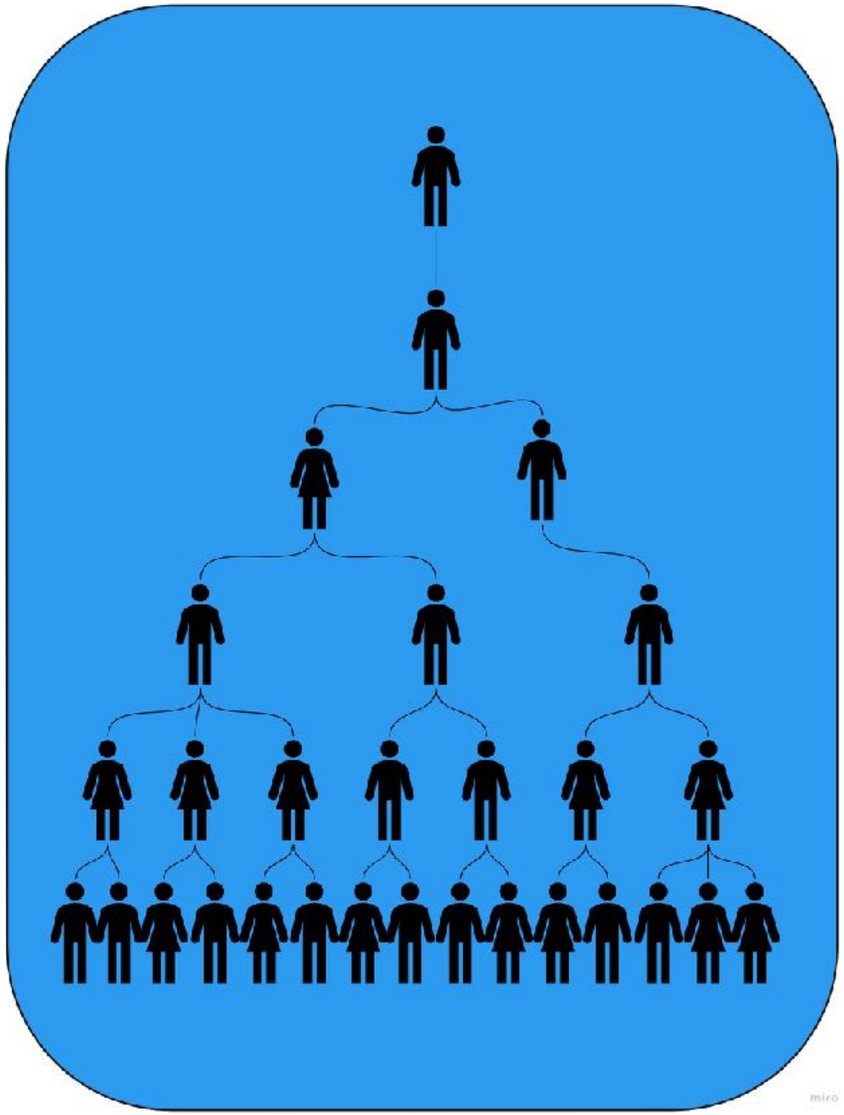


DP2

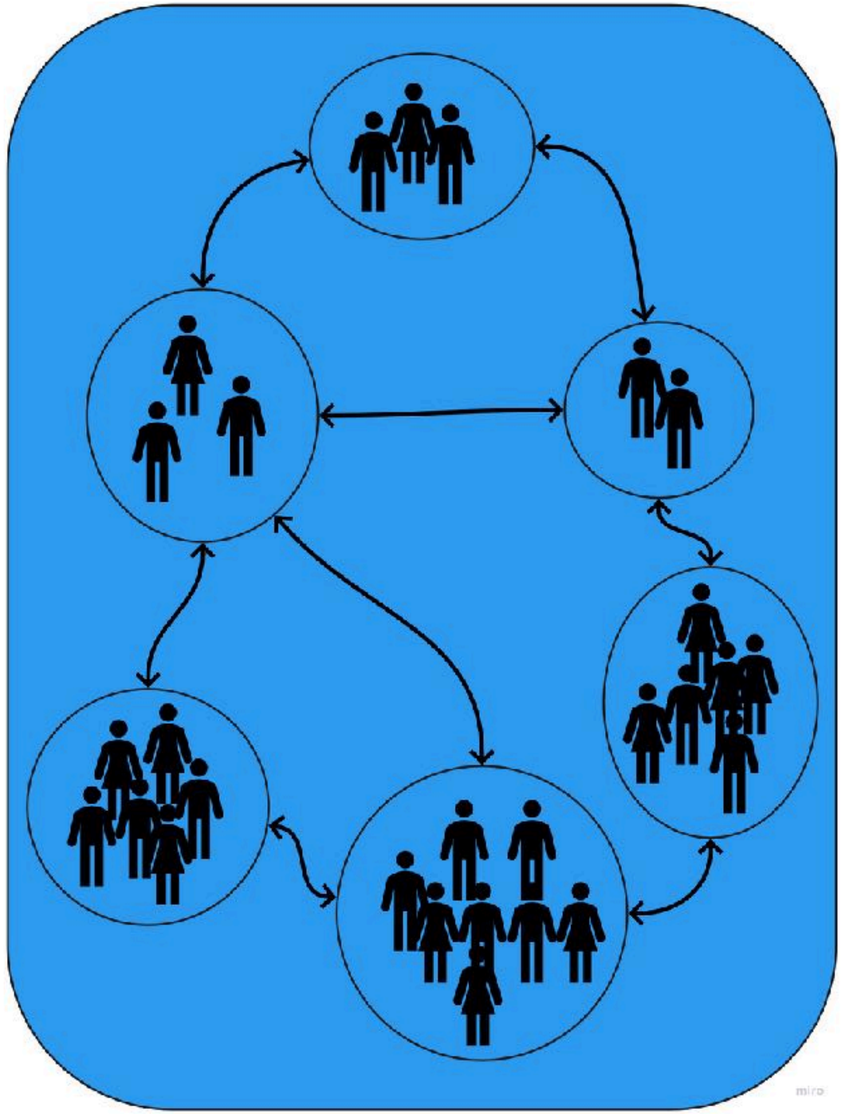
@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

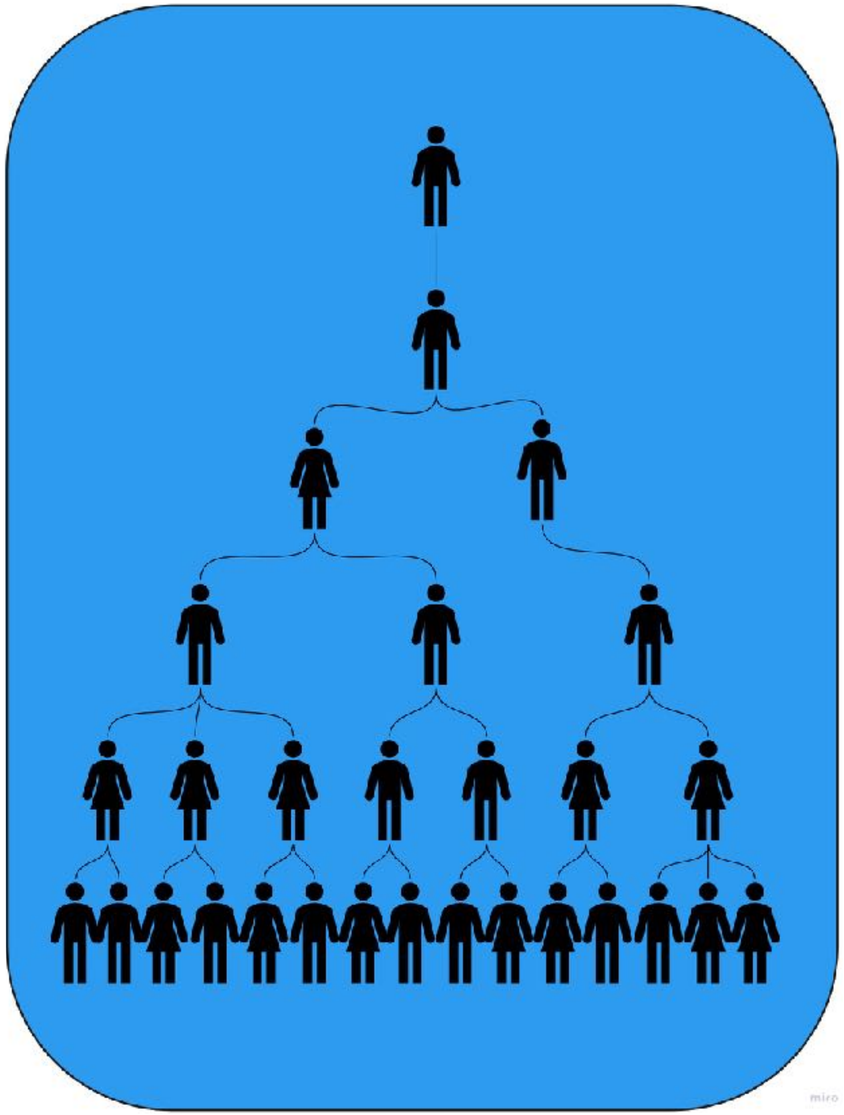
Capra



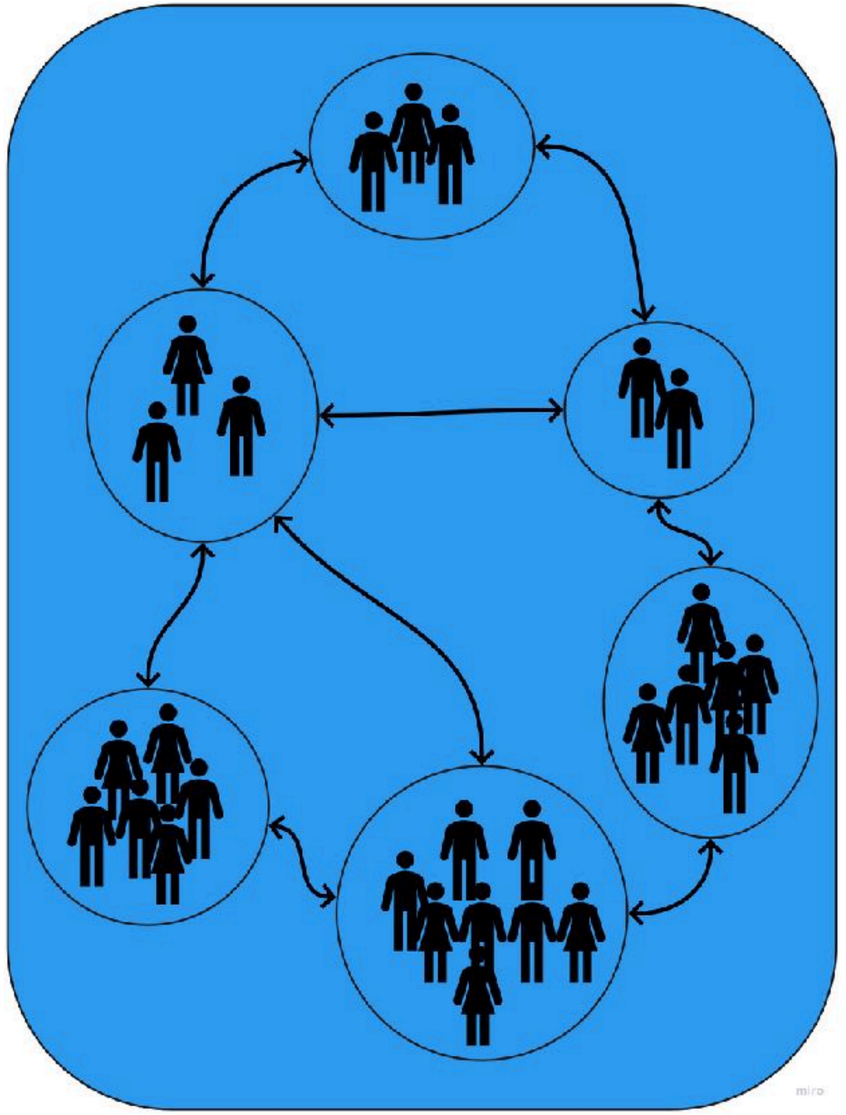
DP1



DP2



DP1

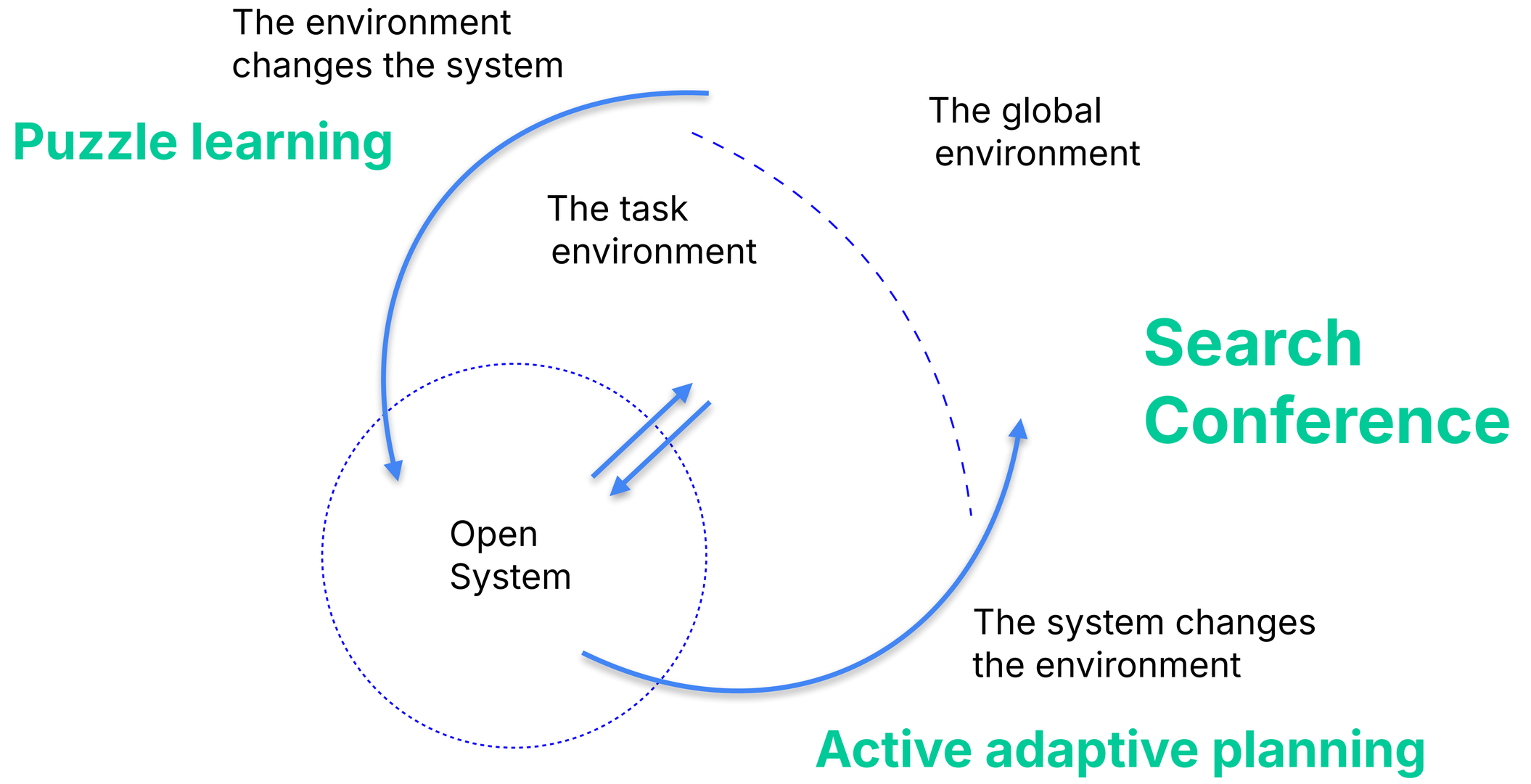


DP2

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra



@trondhjort

STS RT SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: Merrelyn Emery, "Self managing management of the self managing organization: an update"

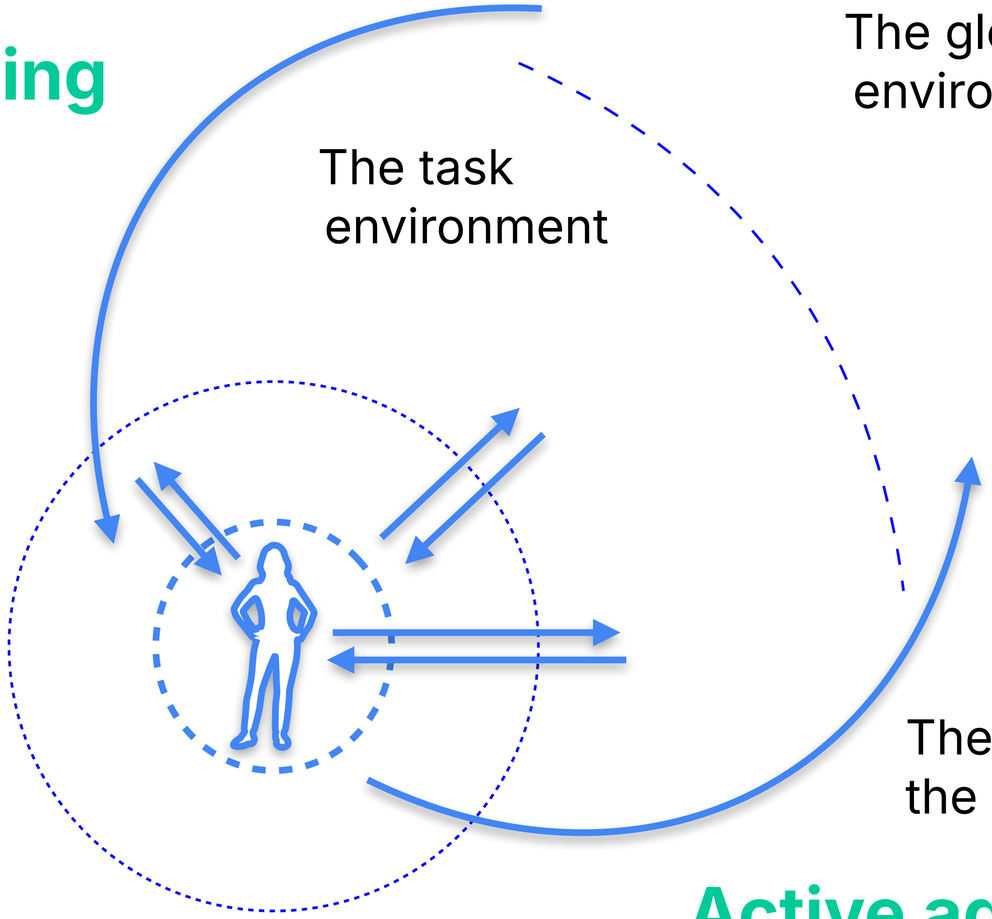
Capra

The environment changes the system

Puzzle learning

The global environment

The task environment



Search Conference

Participative Design Workshop

The system changes the environment

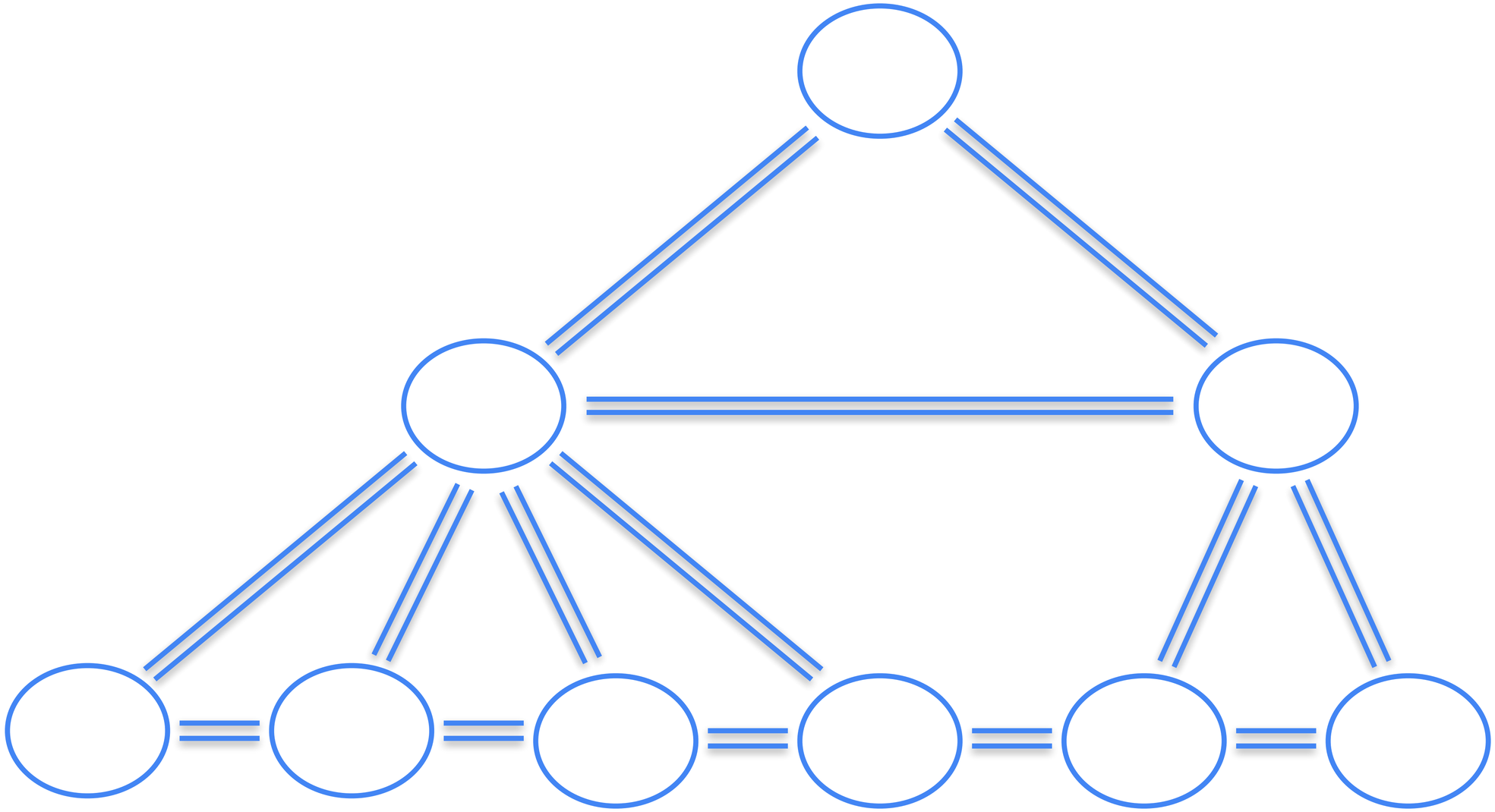
Active adaptive planning

@trondhjort

STS RT SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: Merrelyn Emery, "Self managing management of the self managing organization: an update"

Capra



@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: Merrelyn Emery, "Self managing management of the self managing organization: an update"

Capra

Merrelyn Emery

From Wikipedia, the free encyclopedia

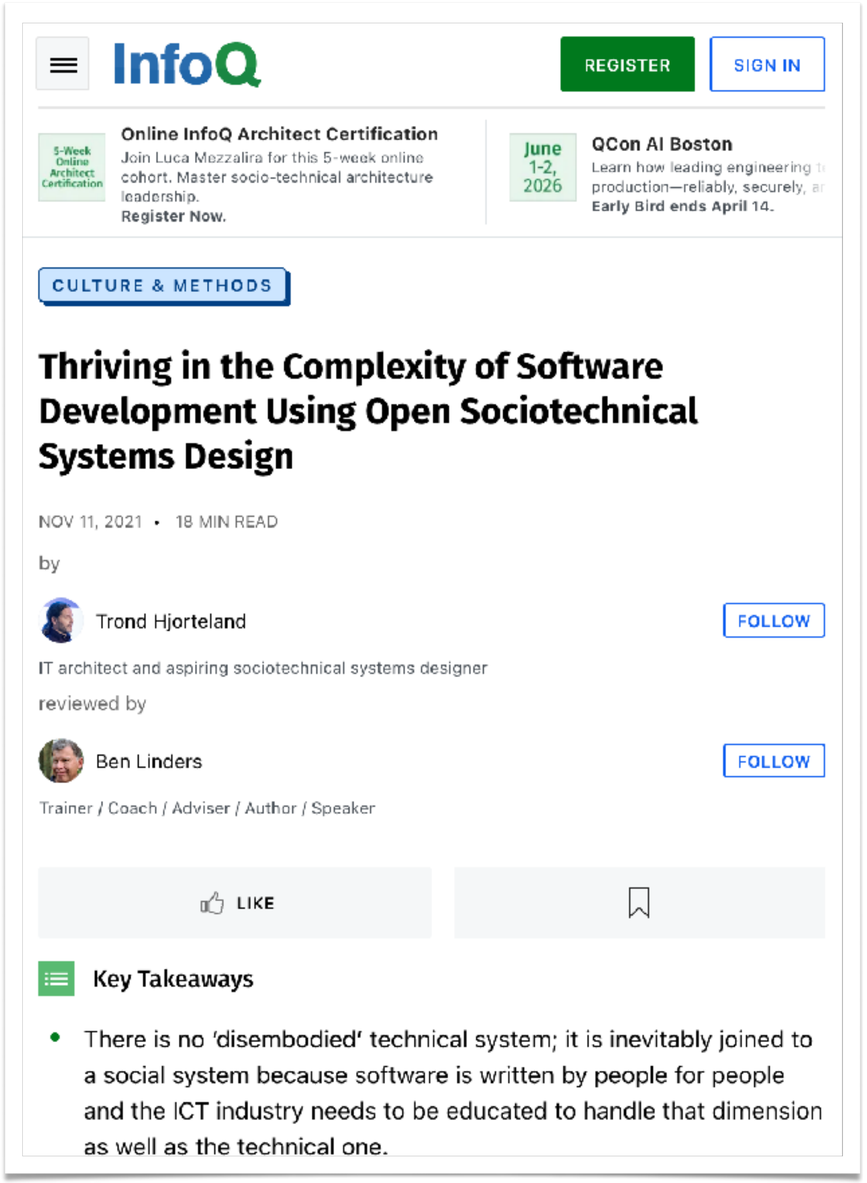
Merrelyn Emery (née Butterfield; born April 27, 1940) is an Australian social scientist and adjunct professor at Department of Applied Human Sciences, Concordia University, and has a first class honours degree from University of New England (Australia) and a PhD from University of New South Wales. She is best known for her work with Fred Emery on the theoretical framework called Open Systems Theory (OST),[1] including refinement of the related Search Conference participative planning process and Participative Design Workshop methodology,[2] both which are part of the dialogic organization development toolset.[3]

Since 1970, she has worked to develop both the theory and methods of open systems using action research with organizations and communities.[4] She was a faculty member at Australian National University and has served on several diverse community and educational organizations around the world.[5]

She regards humans as "innately social animals; we grow only according to the density of interconnections we share with a group," and that "the basic unit of society is the group, not the individual".[6] These groups, be it families, communities, and organizations, is in OST seen as an open social system that transacts with its environment, the external social field, and co-evolution and active adaptation is needed for sustainability and harmony. Socioecology captures the notion of people-in-environments. Included within this is the concept of open, jointly optimized, sociotechnical systems, optimizing human purposefulness and creativity, and the best options afforded by changing technologies. This can be achieved by employing the two-stage model, the Search Conference and the Participative Design Workshop, whose purpose is to create change toward a world that is consciously designed by people, and for people, living harmoniously within their ecological systems, both physical and social. "As the purpose of the two-stage model is to build a community, most work takes place in plenary, generating excitement, joy, and the energy that powers diffusion."[1]

Publications [edit]

Merrelyn Emery is the author or coauthor of ten books, eight edited books, 35 book chapters, 60 journal articles, and contributed 29 institutional research reports including several national studies (e.g. Project Australia, the National Telecom Study, Workplace Australia and Future Directions).[7]



InfoQ

REGISTER SIGN IN

5-Week Online Architect Certification

Online InfoQ Architect Certification
Join Luca Mezzalira for this 5-week online cohort. Master socio-technical architecture leadership.
Register Now.

June 1-2, 2026

QCon AI Boston
Learn how leading engineering t production—reliably, securely, a
Early Bird ends April 14.

CULTURE & METHODS

Thriving in the Complexity of Software Development Using Open Sociotechnical Systems Design

NOV 11, 2021 • 18 MIN READ

by

Trond Hjorteland FOLLOW

IT architect and aspiring sociotechnical systems designer

reviewed by

Ben Linders FOLLOW

Trainer / Coach / Adviser / Author / Speaker

LIKE

Key Takeaways

- There is no 'disembodied' technical system; it is inevitably joined to a social system because software is written by people for people and the ICT industry needs to be educated to handle that dimension as well as the technical one.

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

@trondhjort

Capra

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra

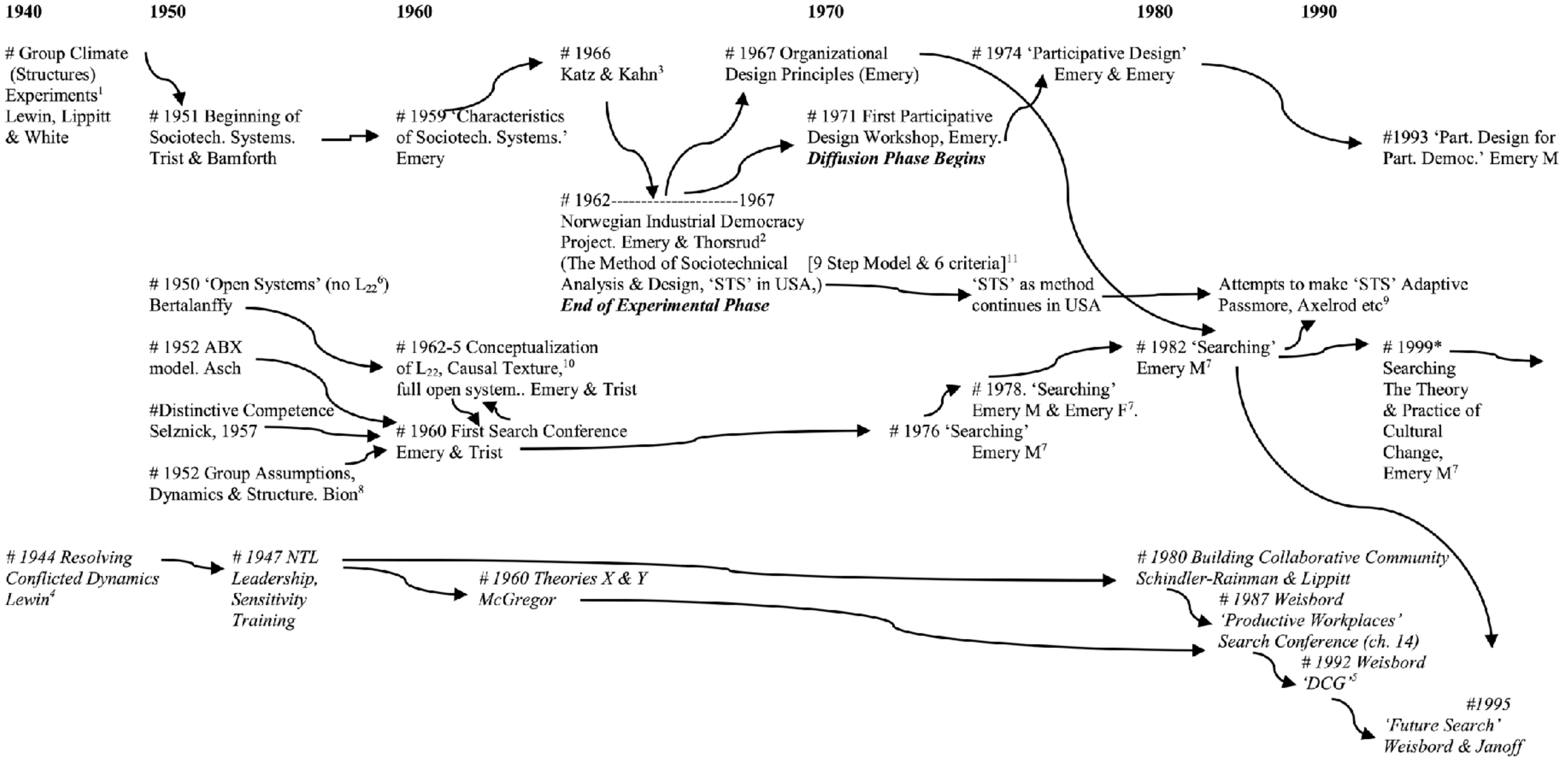

@trondhjort
STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.
Capra

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: www.opensystemstheory.org

Capra

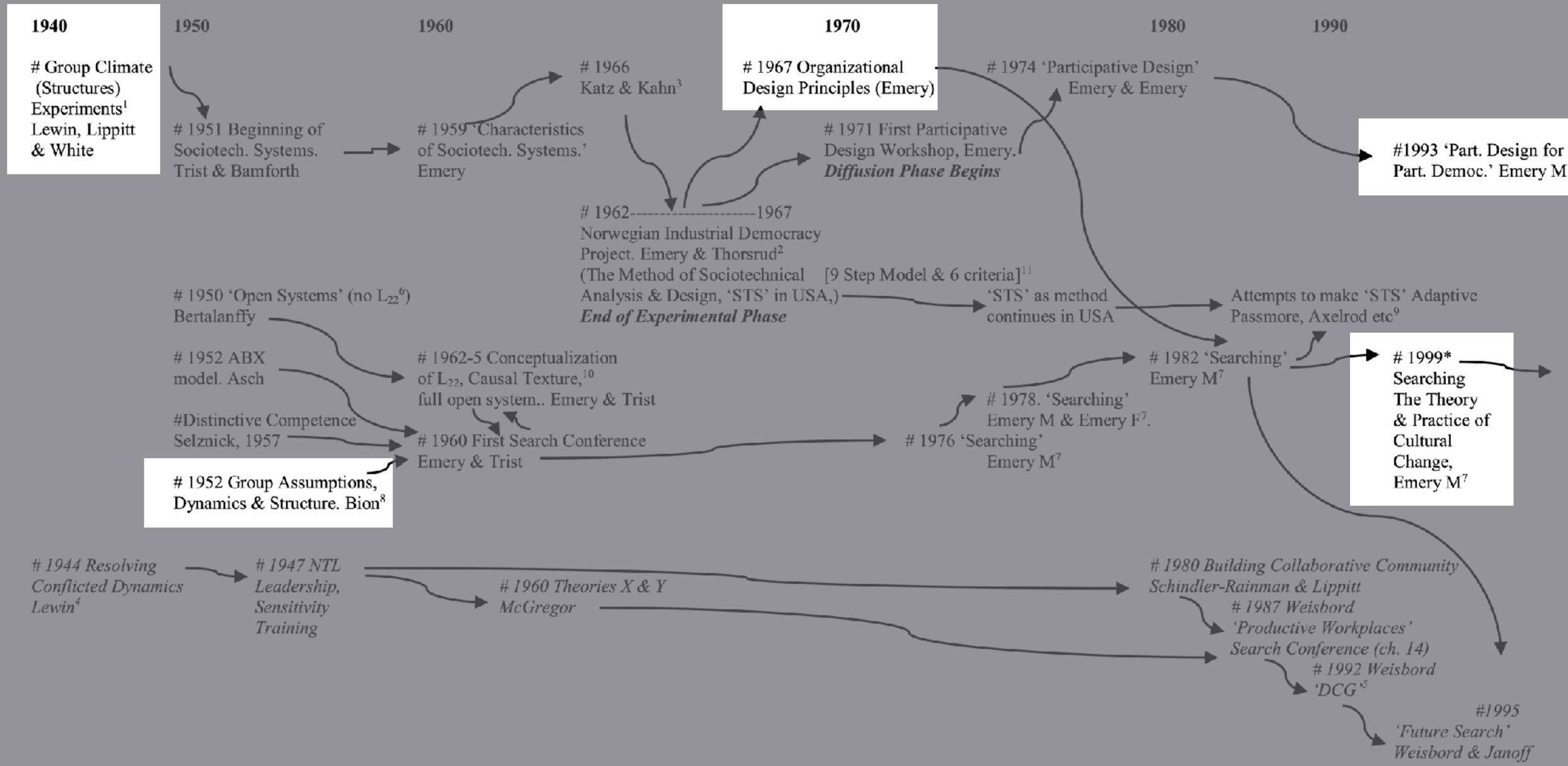


@trondhjort

STS RT SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: Merrelyn Emery, The Evolution of Open Systems Theory

Capra



@trondhjort

STS RT SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Source: Merrelyn Emery, The Evolution of Open Systems Theory

Capra

EventStorming

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

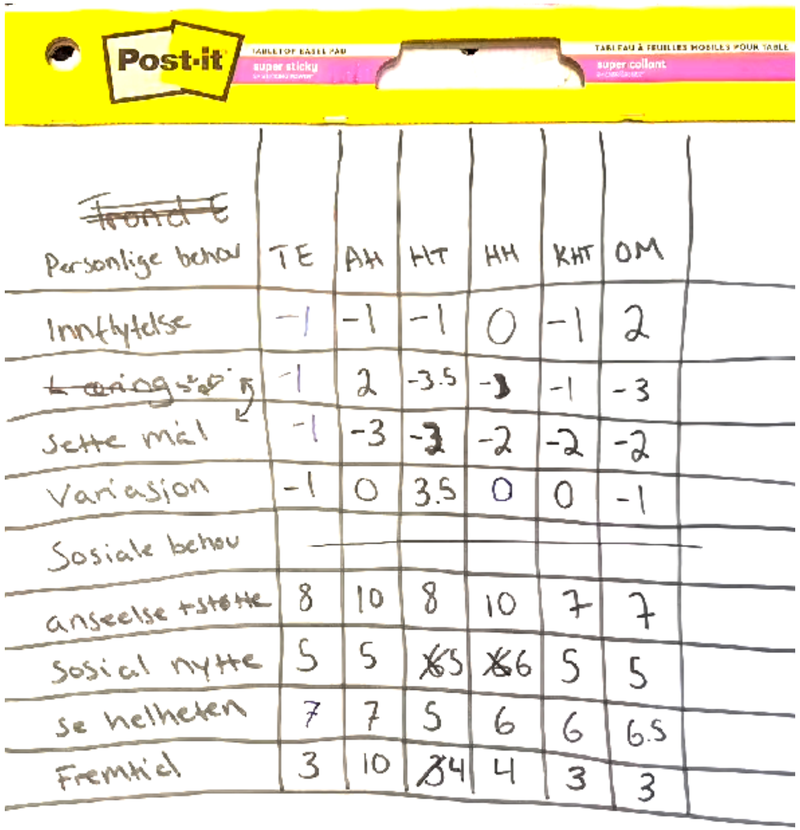
Capra

Pre-mortem

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra



Personlige behov	TE	AH	HT	HH	KHT	OM
Innflytelse	-1	-1	-1	0	-1	2
~~Lærings~~	-1	2	-3.5	-3	-1	-3
Sette mål	-1	-3	-3	-2	-2	-2
Variasjon	-1	0	3.5	0	0	-1
Sosiale behov						
anseelse +støtte	8	10	8	10	7	7
Sosial nytte	5	5	~~6~~5	~~7~~6	5	5
se helheten	7	7	5	6	6	6.5
Fremtid	3	10	~~4~~4	4	3	3

Partcipative Design Workshop

@trondhjort

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

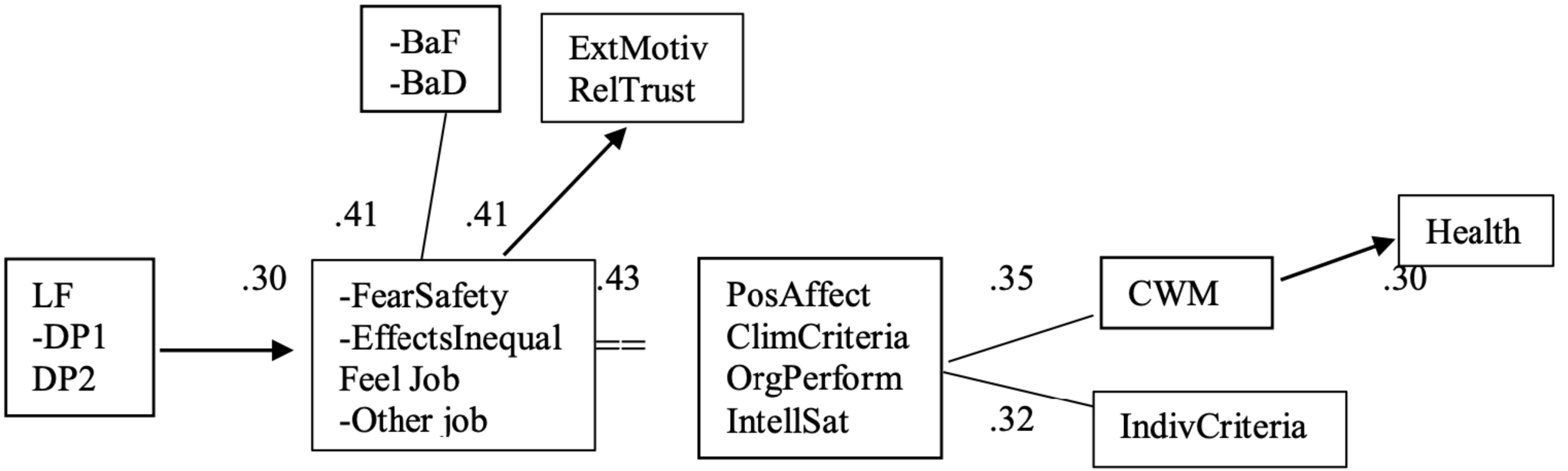
Capra

A Patchwork of Contradictions and Confusions: Inside the Software Industry

Merrelyn Emery

January 2023

In 2021 a series of conversations took place in which concerns were raised about the software industry and what was happening to personnel in various sections of it. One of those concerned had learnt of Open Systems Theory with its genotypical design principles underlying organizational redesign and wondered whether such an option could help with analysis or present a solution if such was required. That one was Trond Hjorteland and my thanks go to him. He not only initiated these discussions, he started diffusing the concepts and involving others. He also did all the technical work on the survey to make it possible. His efforts also led to an introduction to OST course being held in Canberra. None of that would have been possible without him.



N=87, r=.21 @p<.05, r=.27 @p<.01, r=.34 @p<.001

Figure 3. Causal Path for Managerial Sample (from M4)

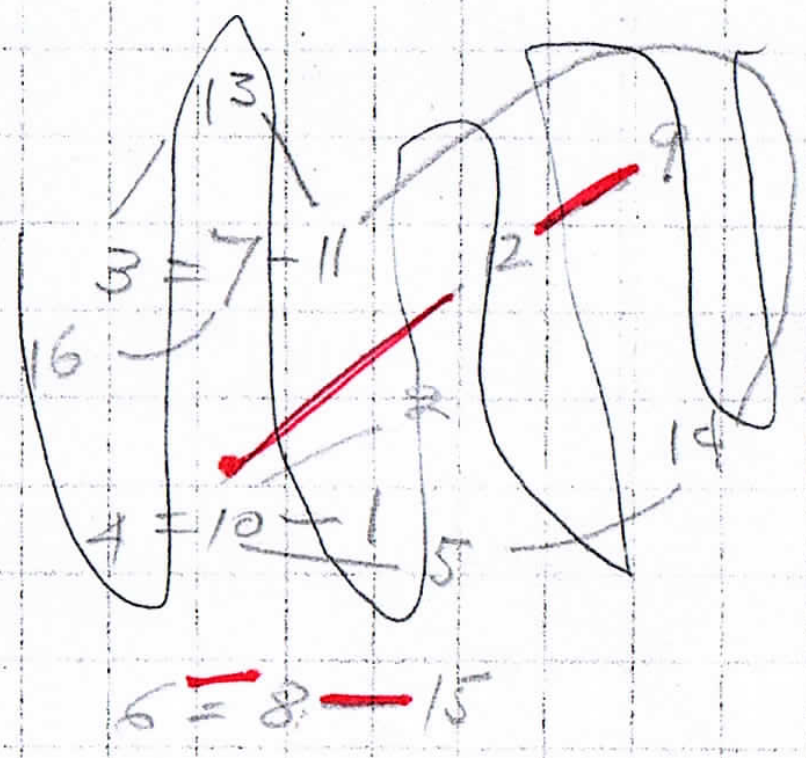
Extract from ~~Orig~~ Combined file.

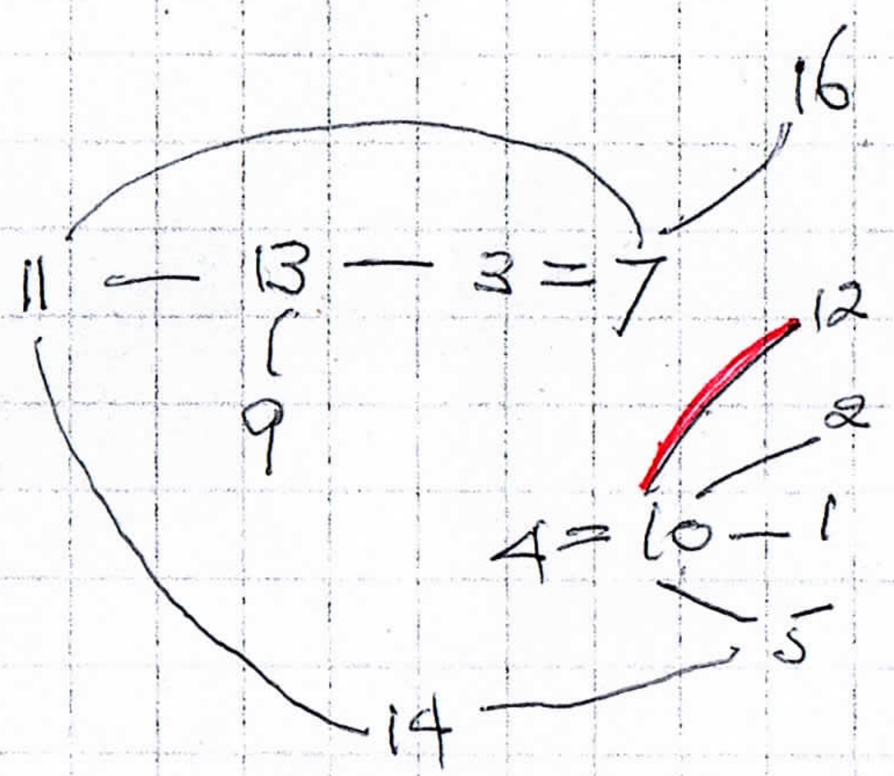
r = .12, .16, .20.

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	L
Ignore	1	—	15	26	48	54	04	23	04	04	62	0	21	14	12	04	25	10
Sick	2		—	07	39	10	09	08	01	05	46	12	17	06	05	01	01	10
Com	3			—	21	27	17	72	09	03	33	22	29	21	01	07	17	7
baF	4				—	63	24	25	21	07	71	07	15	04	13	09	14	10
baD	5					—	20	32	14	09	67	25	21	04	17	0	26	10
DP2	6						—	25	89	02	24	24	20	09	03	12	07	8
Pos. Aff.	7							—	20	01	47	32	36	22	01	02	29	3
DP1	8								—	10	18	26	16	07	05	16	0	6
LF	9									—	01	06	15	18	08	14	04	~~12~~ 13
Neg Aff	10										—	14	59	12	10	06	29	4
Intell. Satis	11											—	09	24	17	08	16	7
Hygiene	12												—	03	13	03	14	10
SES	13													—	11	04	11	3+11
Sex	14														—	09	14	5+11
Age	15															—	02	8
Productive	16																—	7

R

3 = 7
4 = 10
6 = 8





@trondhjort

Capra

“As soon as people are forced to compete, they have to look after their own interests and so self-interest comes to dominate life in a DP1 [autocratic/bureaucratic] structure.

All the team building in the world cannot change this dynamic.”

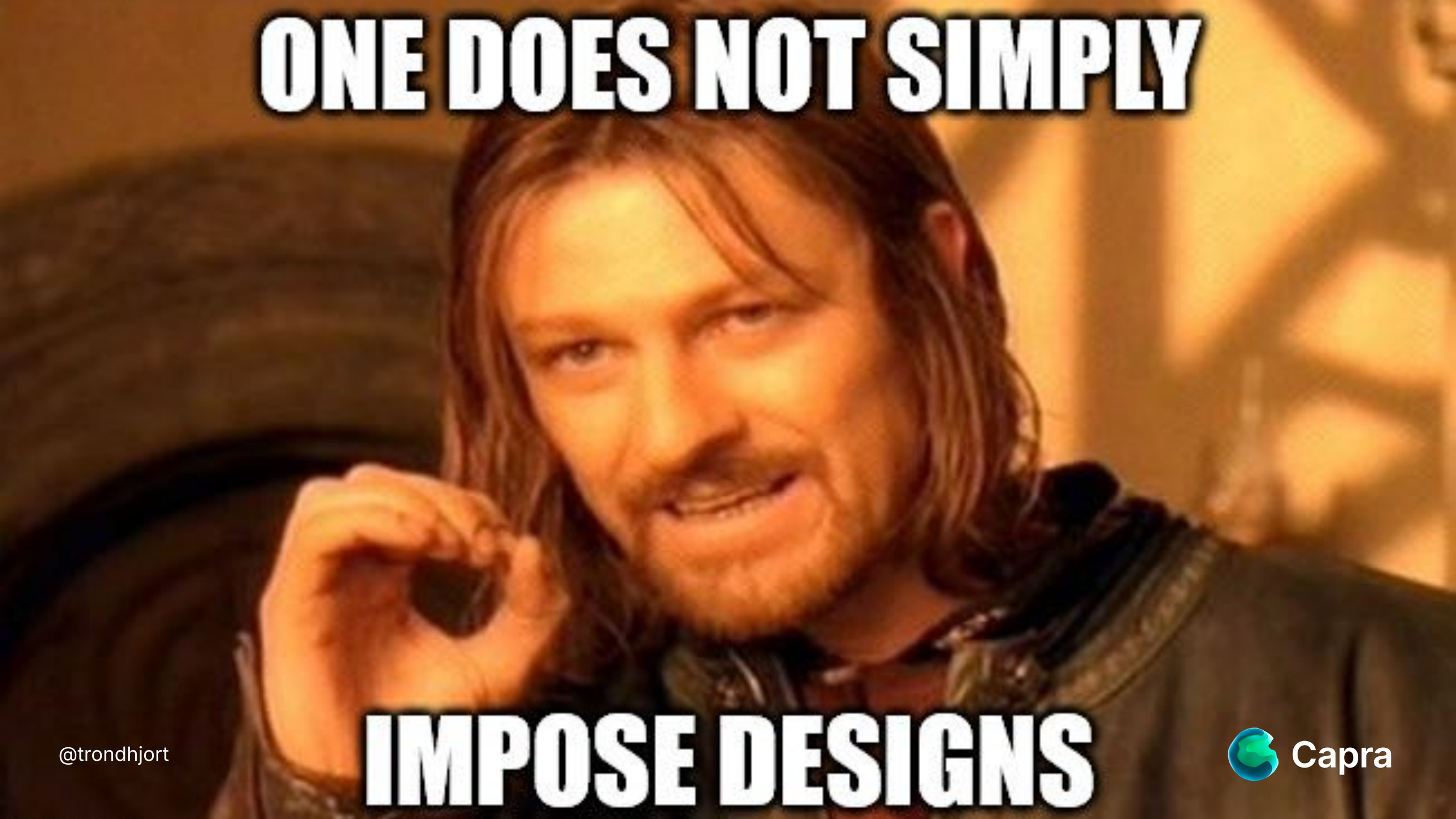
-Merrelyn Emery

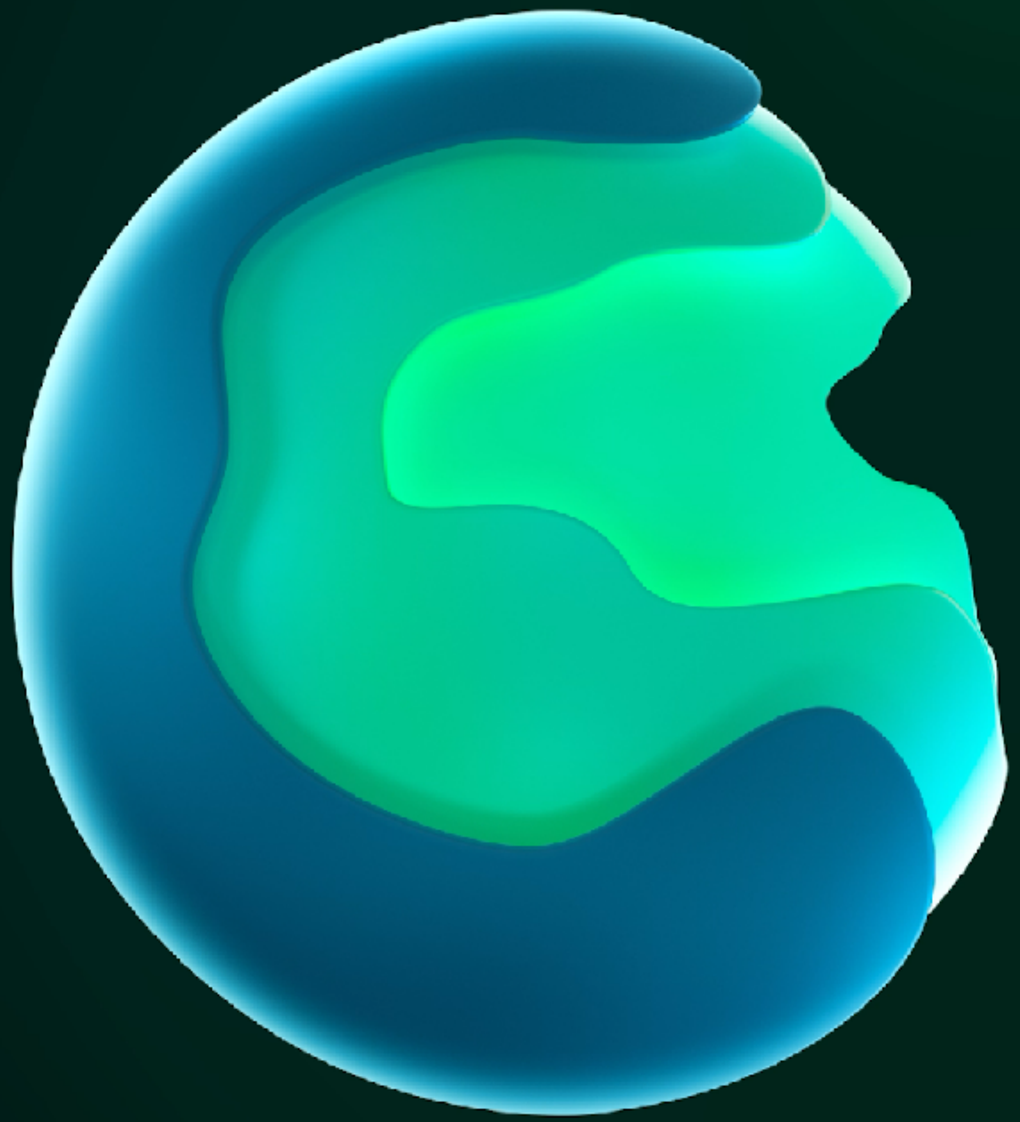
“It is only when the people involved work out their own designs that the necessary motivation, responsibility and commitment to effective implementation is present.”

-Merrelyn Emery

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Capra


ONE DOES NOT SIMPLY
IMPOSE DESIGNS
@trondhjort
Capra



Thank you

@trondhjort

www.opensystemstheory.org

UPCOMING EVENTS

THANK YOU FOR COMING!

JOIN THE STS ROUNDTABLE

STS RT
SOCIOTECHNICAL SYSTEMS ROUNDTABLE INC.

Socio-Technical Systems Roundtable

www.stsroundtable.com