



Technische
Universität
Braunschweig



Research Software Development is not a Jenga Game!

Learnings from SURESOFT – RSE Beyond Tooling

SE24 - RSE Panel – Jan Linxweiler

Focus on two Questions

1

“Why do we need Research Software Engineering?”

or

“How to convince people that we need Research Software Engineering?”

2

“Beyond Tools - What skills are needed?”

One of the first scientific software developers



“One of our difficulties will be the maintenance of an **appropriate discipline**, so that we **do not lose track of what we are doing.**”
- 1947

[Lecture to London Mathematical Society, February 20, 1947]

Learnings from SURESOFT – RSE Beyond Tooling | SE24 – RSE Panel | Jan Linxweiler | Slide 3

“ The art of programming is the art of organizing complexity. ”

Edsger W. Dijkstra

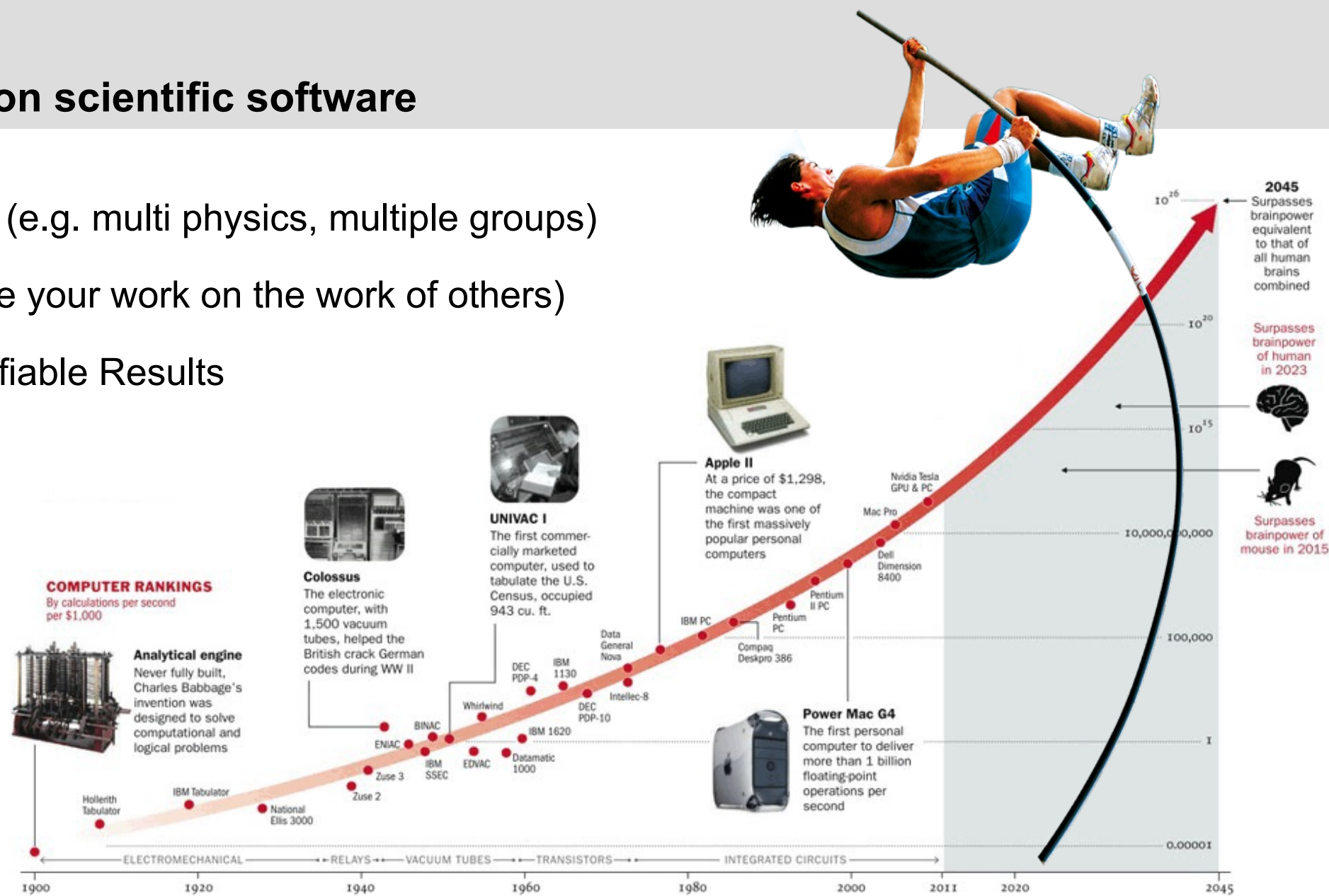


[Notes On Structured Programming, Edsger W. Dijkstra, 1970]

[Frederick P. Brooks, No Silver Bullet: Essence and Accidents of Software Engineering, 1987]

Growing demands on scientific software

- Increasing Complexity (e.g. multi physics, multiple groups)
- Longer Life Span (base your work on the work of others)
- Reproducible and Verifiable Results

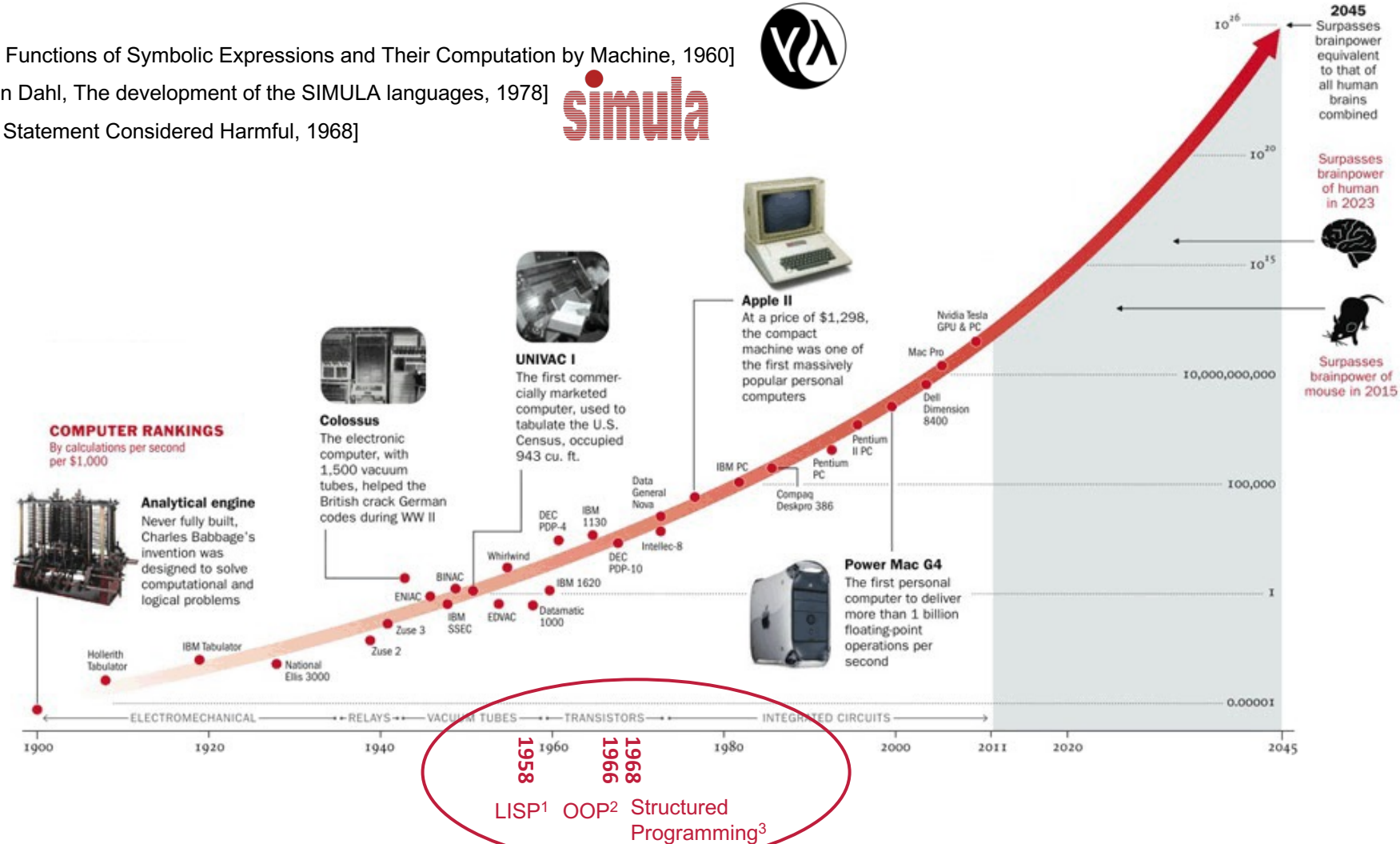


Growth in technology vs. software development paradigms

¹ [John McCarthy, Recursive Functions of Symbolic Expressions and Their Computation by Machine, 1960]

² [Kristen Nygaard, Ole-Johan Dahl, The development of the SIMULA languages, 1978]

³ [Edsger W. Dijkstra, Go To Statement Considered Harmful, 1968]



Take Home Messages

In accordance to Wirth's law one can argue:

“Software systems grow faster in size and complexity than methods to handle complexity are invented.”

[Niklaus Wirth, "A Plea for Lean Software", 1995]



We need to **make the best possible use of the software development techniques available** to cope with the growth in complexity.

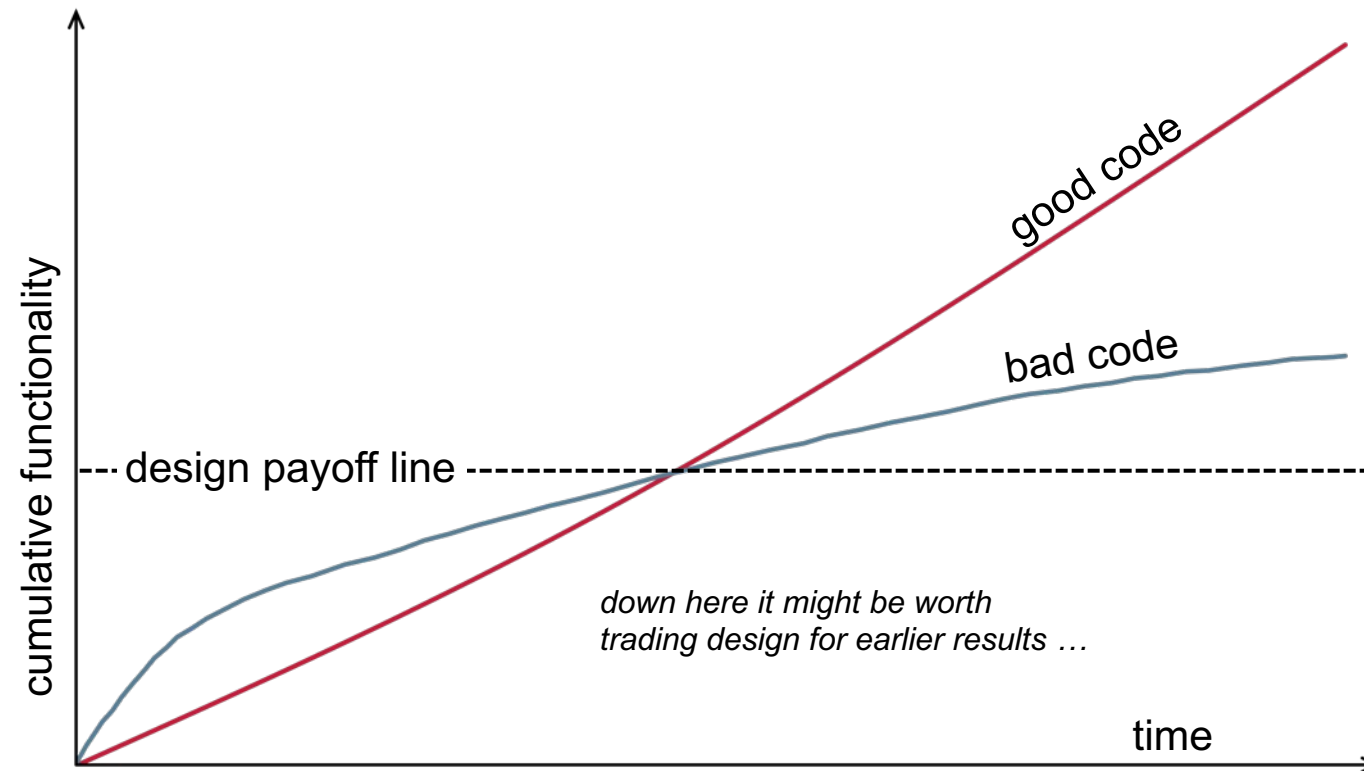
“The gap between the best software engineering practice and the average practice is very wide — perhaps wider than in any other engineering discipline. [...] **The difference between the the great and the average approach an order of magnitude.**”

[Frederick P. Brooks, No Silver Bullet: Essence and Accidents of Software Engineering, 1987]

We are stuck!

- floating point performance is constantly rising
- time-to-solution is inceasing
- scientists spend 50% of the time finding bugs

[P. Prabhu, A Survey of the Practice of Computational Science, 2011]



Design Stamina Hypothesis - <https://bit.ly/2A64CAR>

Learnings from SURESOFT – RSE Beyond Tooling | SE24 – RSE Panel | Jan Linxweiler | Slide 8

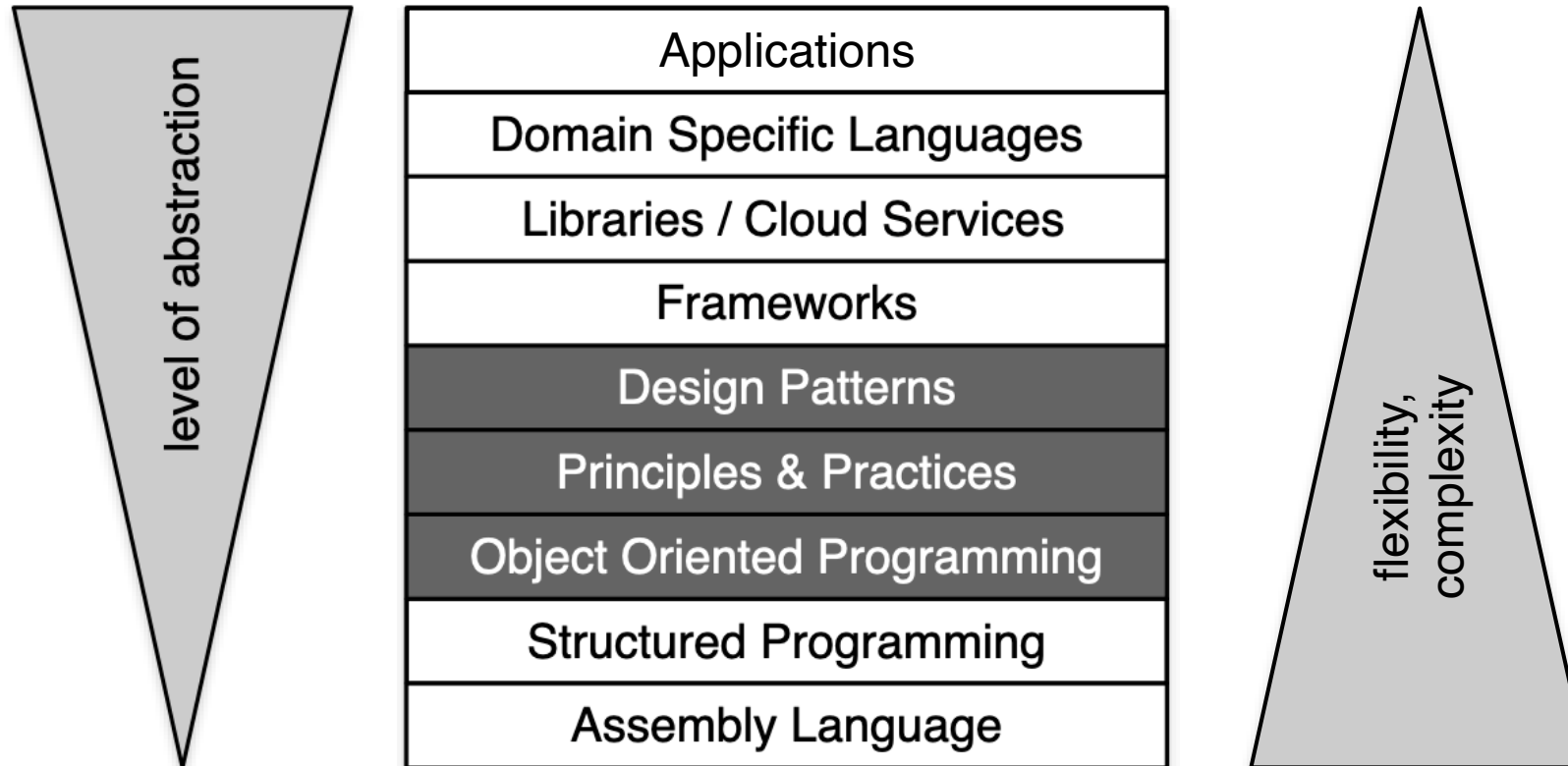


Suresoft -Approach

SUSTAINABLE RESEARCH SOFTWARE

“The function of good software is to make the complex appear to be simple. ”

Grady Booch



SURESOFT Approach for Sustainable Software

Essential
Complexity

Methods

Documentation

Software Engineering
Principles

Testing

Infrastructure, Tools & Process

Version Control

Archiving &
Publication

CI &
Automated Testing

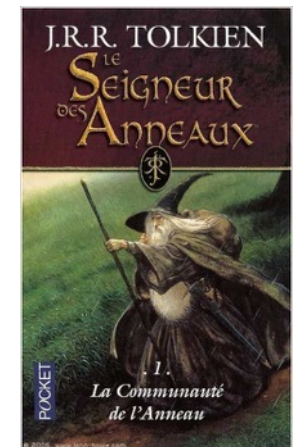
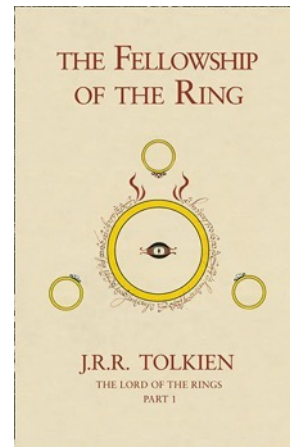
Virtualization

Issue Reporting

Installation &
Deployment

Accidental
Complexity

It's all about higher-level concepts...



It needs more than knowing a language to write a novel!

J.R.R. Tolkien (1892 – 1973)

The one constant in **Soft**ware development is

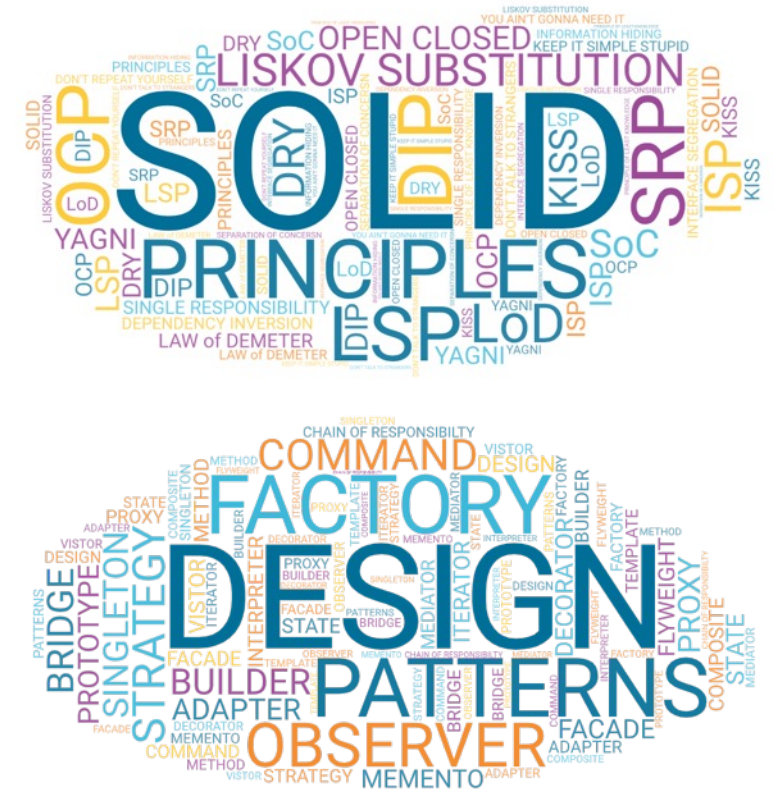
CHANGE

Workshop-Series



HeFDI Code School 'Sustainable Research Software'

<https://suresoft.dev/workshops/>



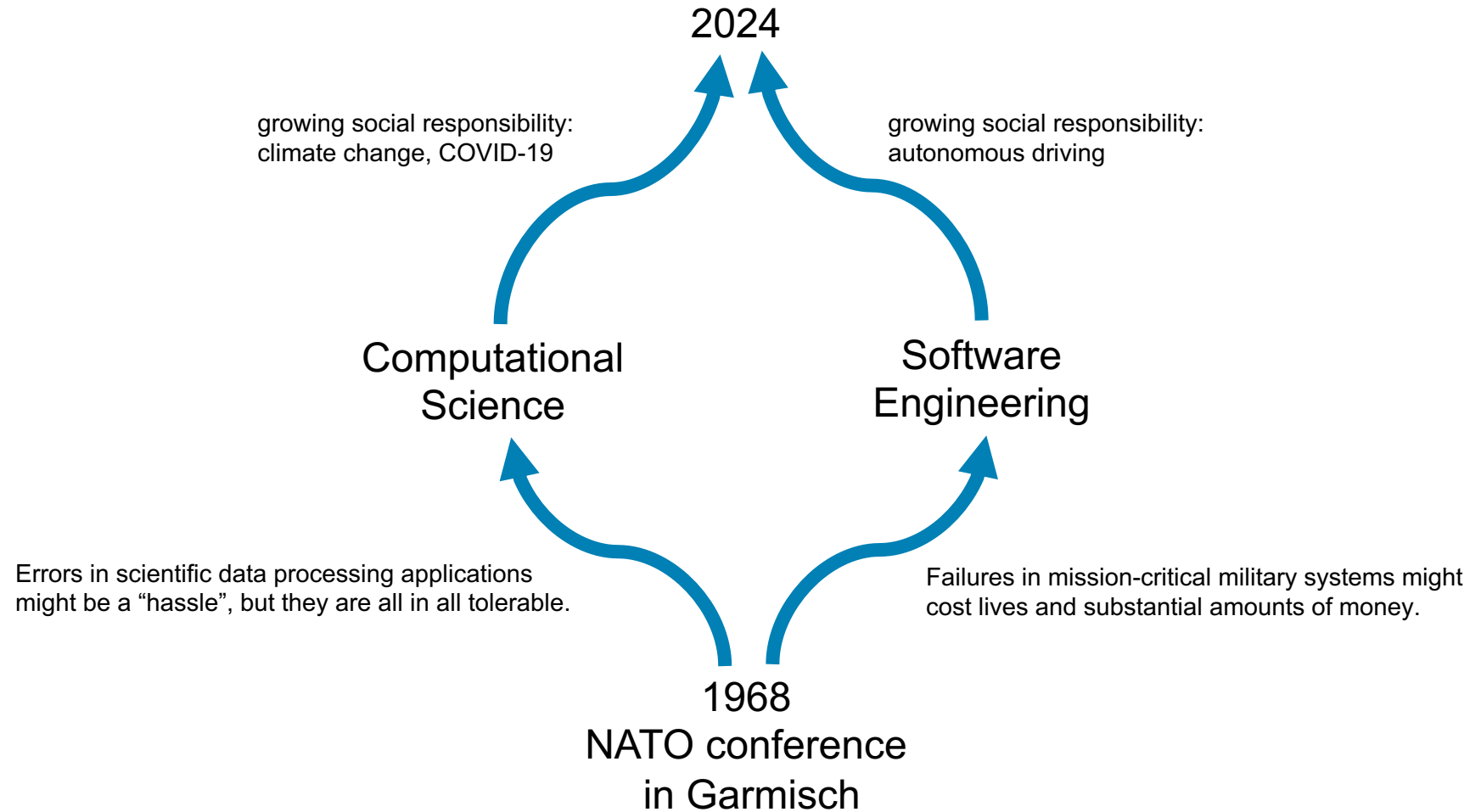
F A I R
EUSABLE

[Notes On Structured Programming, Edsger W. Dijkstra, 1970]

[Frederick P. Brooks, No Silver Bullet: Essence and Accidents of Software Engineering, 1987]

Learnings from SURESOFT – RSE Beyond Tooling | SE24 – RSE Panel | Jan Linxweiler | Slide 15

Birth of Software Engineering discipline in 1968



[Naur, Software Engineering: Report of a Conference Sponsored by the NATO Science Committee, Garmisch, Germany, 1968]

Learnings from SURESOFT – RSE Beyond Tooling | SE24 – RSE Panel | Jan Linxweiler | Slide 16

“Those who cannot remember the past
are condemned to repeat it.”

George Santayana



