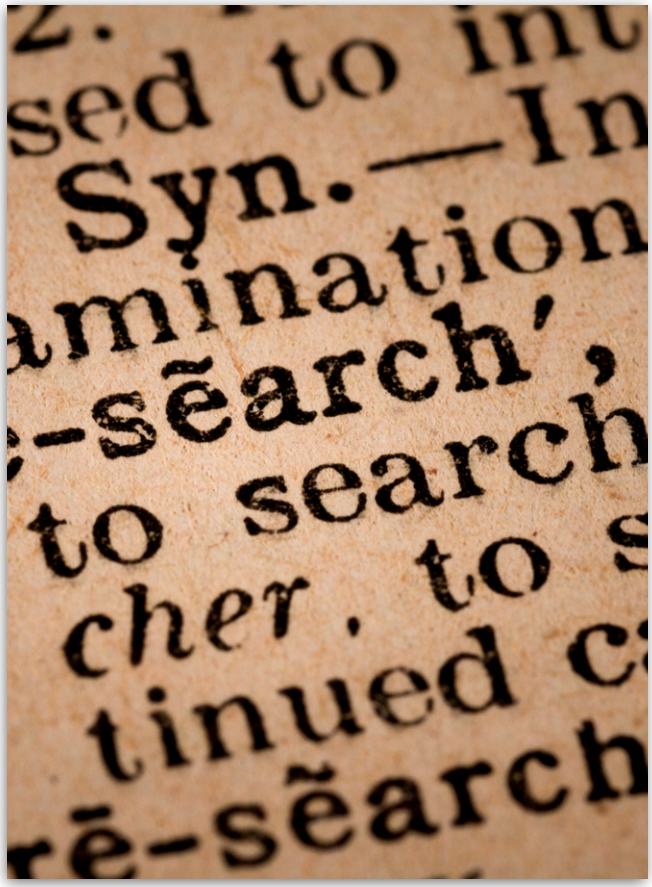
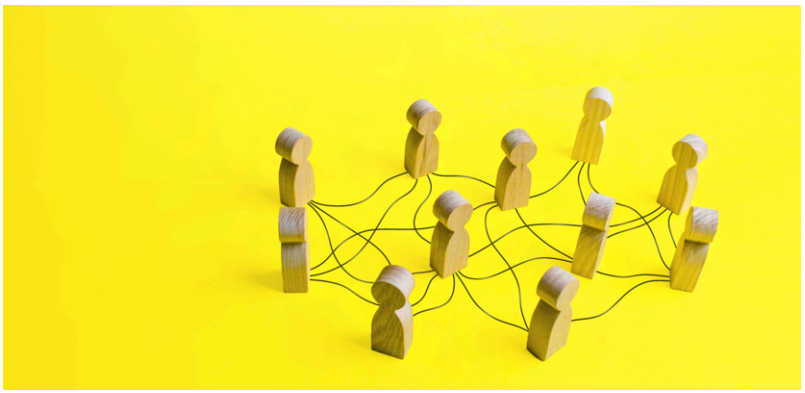


Pursuing Meaningful Software Engineering Research

Gail C. Murphy
University of British Columbia

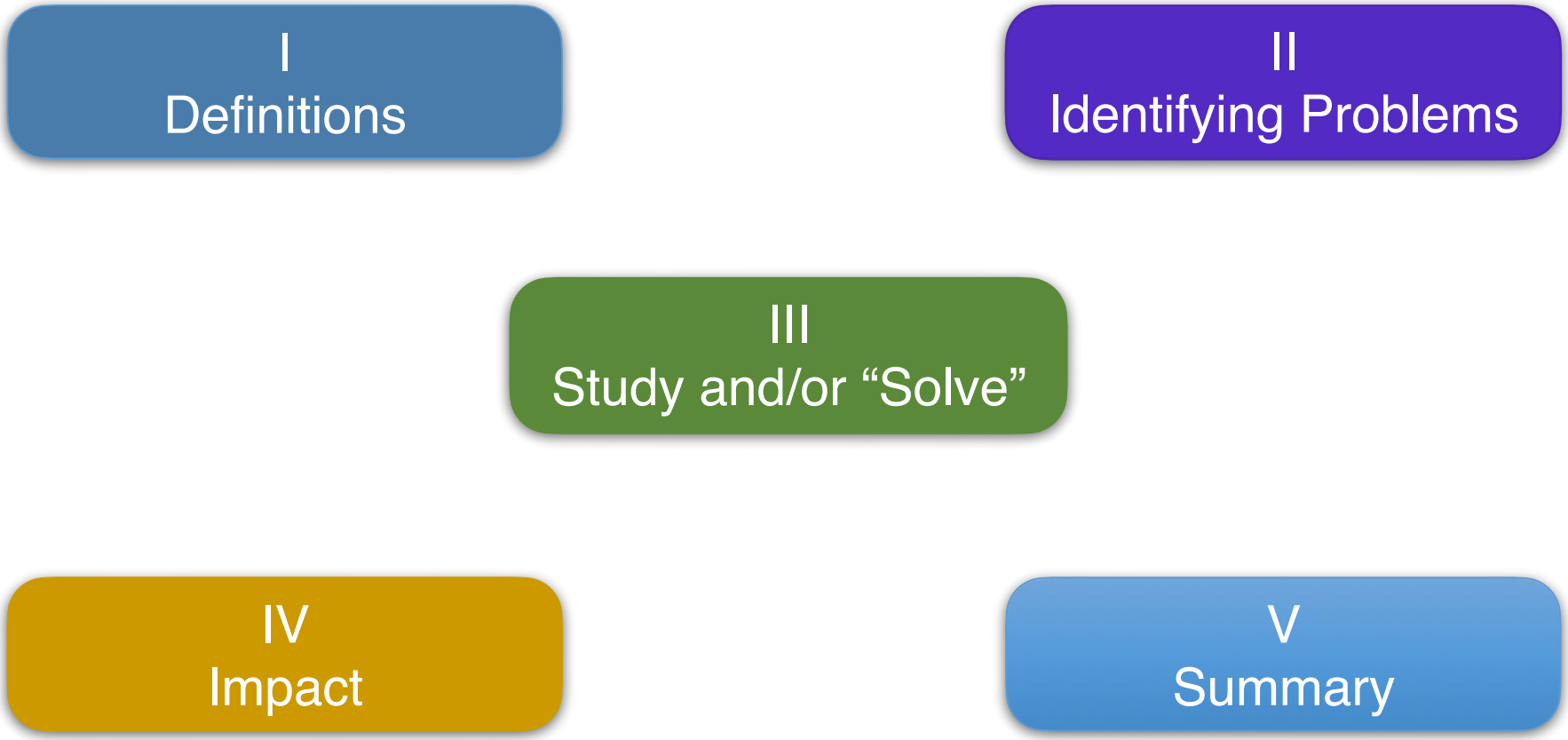


Disclaimer

One person’s views
Not exhaustive

Request

Participate! Ask questions,
provide your ideas!


I
Definitions
II
Identifying Problems
III
Study and/or “Solve”
IV
Impact
V
Summary

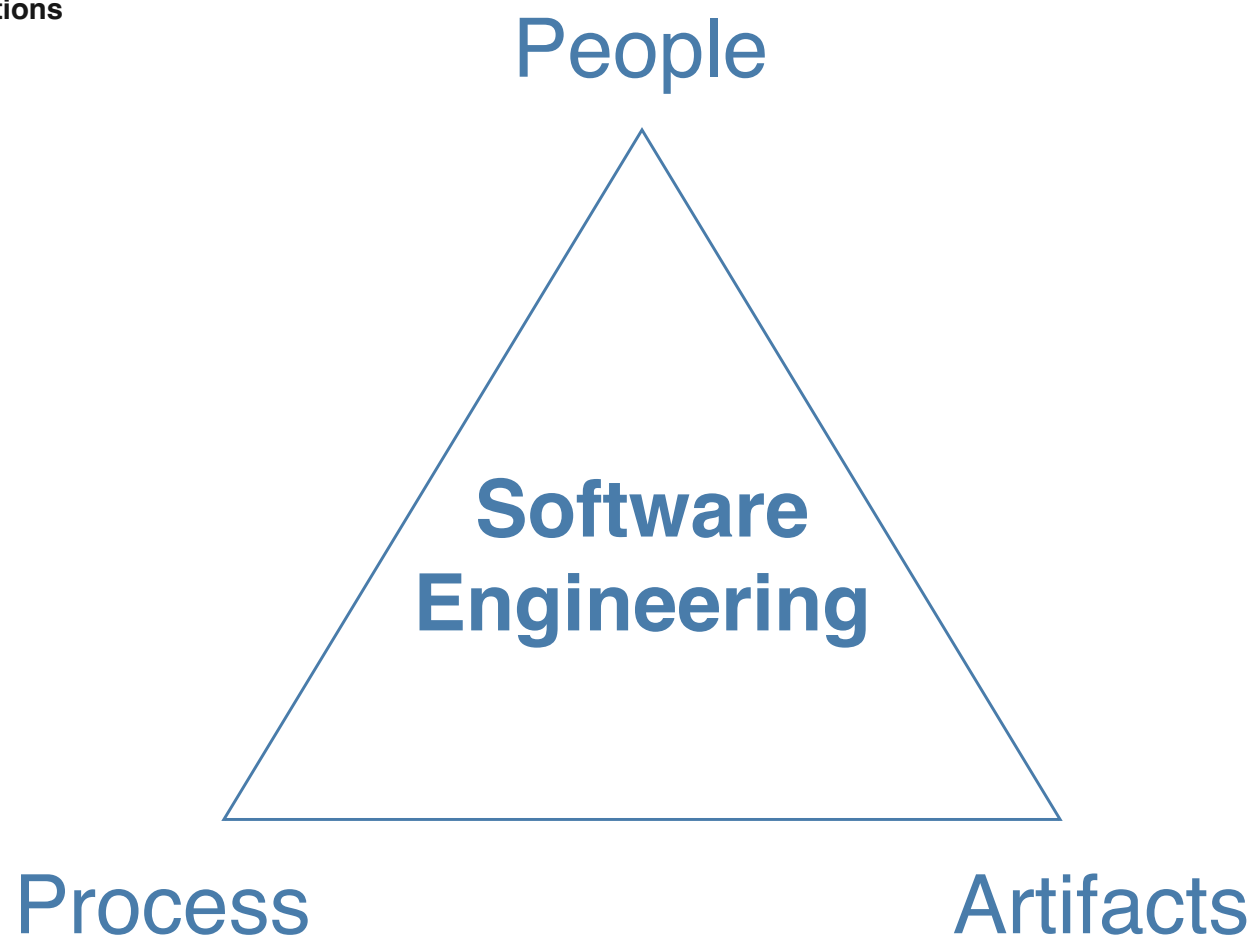
I
Definitions

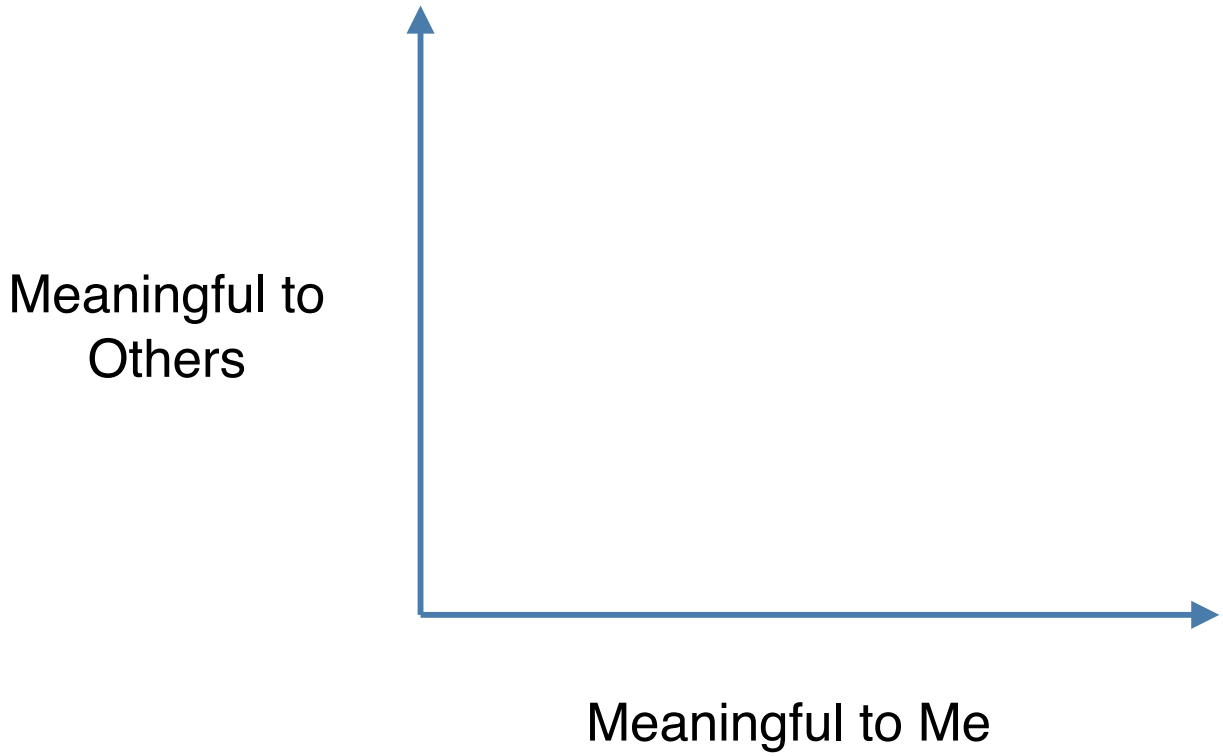
Pursuing Meaningful Software Engineering Research

Software Engineering

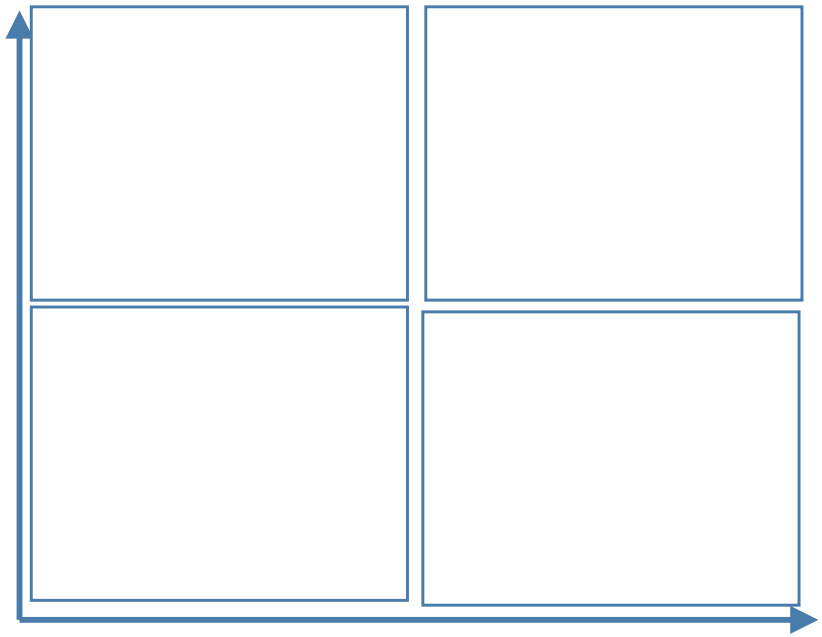
Multi-person multi-version software

- B. Randell


People
Software Engineering
Process
Artifacts

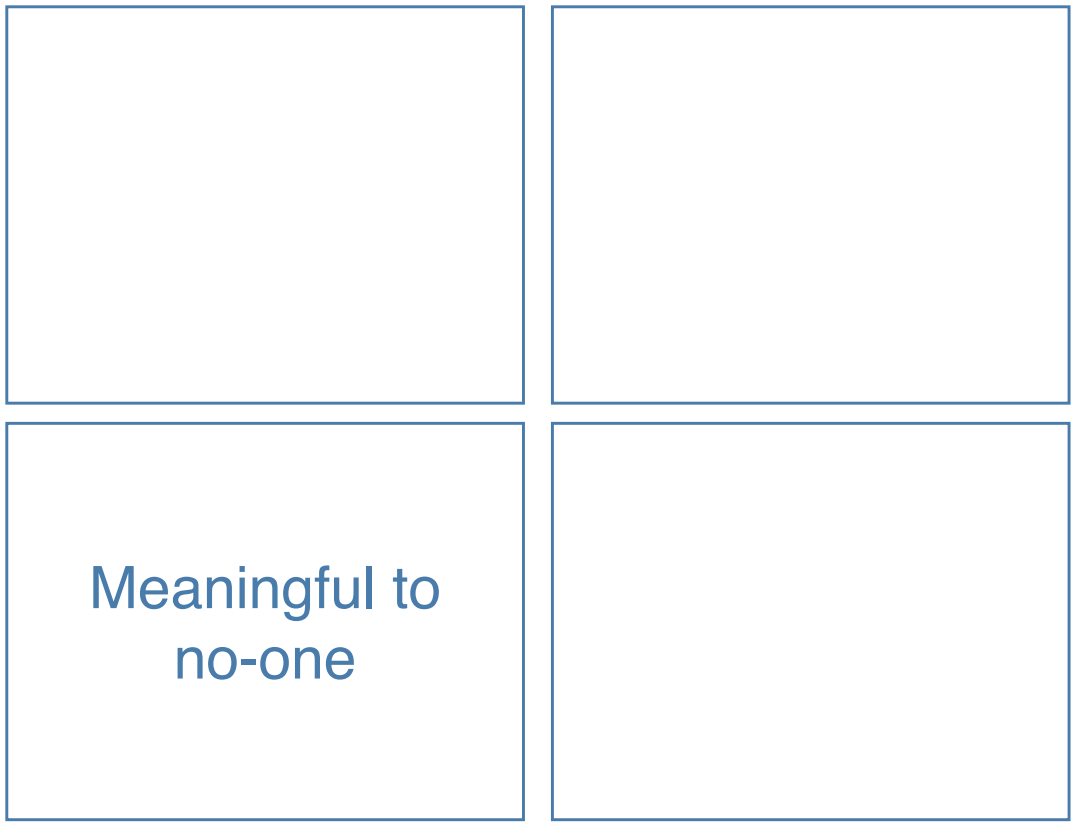

Meaningful to Others
Meaningful to Me

Meaningful to Others



Meaningful to Me

I - Definitions


Meaningful to
no-one


Meaningful to others
Meaningful to me

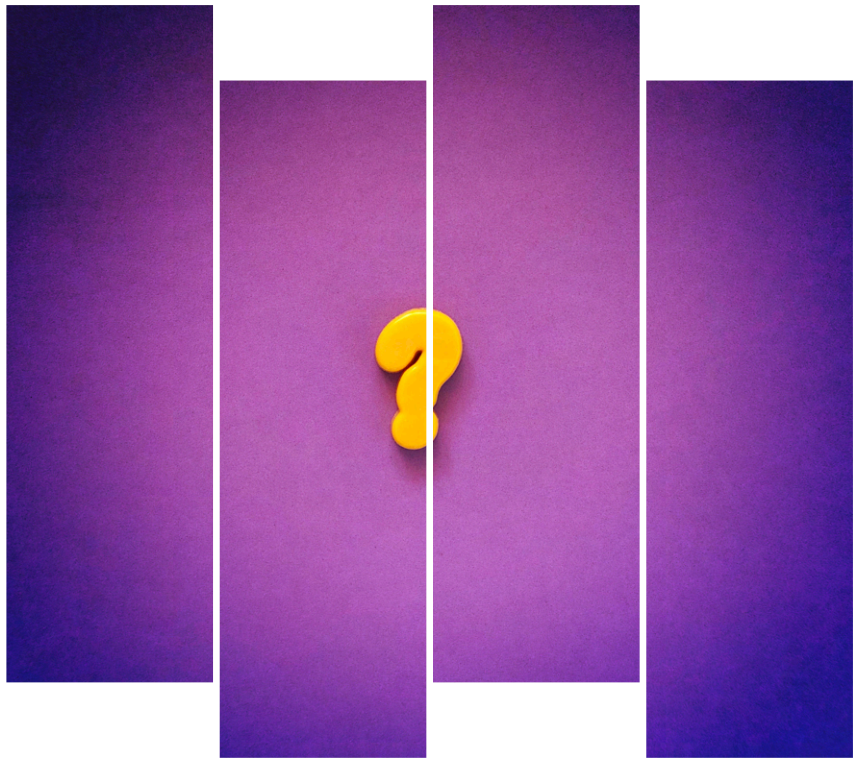
This occurs when you are passionate about the questions you are pursuing and the ways that you are providing insights to those questions impacts academic and/or industry communities.

Meaningful to others and me

Software Engineering

Meaningful

II
Identifying Problems



Ways to Identify Problems

01 Overly hard?

02 Unrealistic assumptions?

03 Relax constraints?

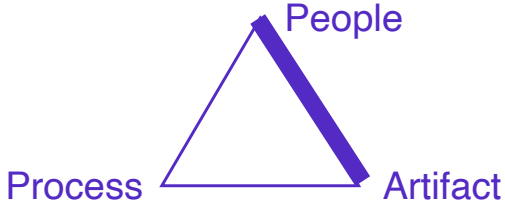
01 Overly Hard

1990s: developer tools often relied on call graphs parsed from source code. These tools were brittle and often didn't work across systems.

Lightweight source model extraction aimed to ease extraction of information like call graphs by bypassing parsing.

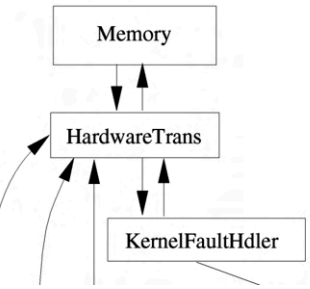
[⟨type⟩] ⟨functionName⟩ \([{ ⟨formalArg⟩ }+] \) [{ ⟨t

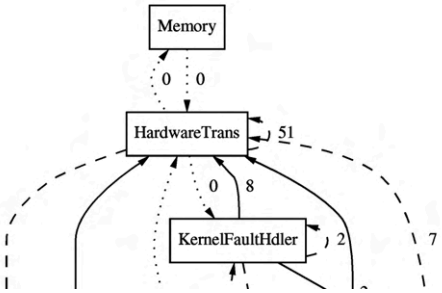
Figure 2: A Pattern for Locating Func



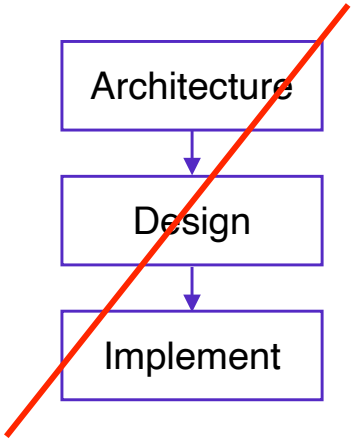
Lightweight Source Model Extraction
Murphy and Notkin, FSE '95

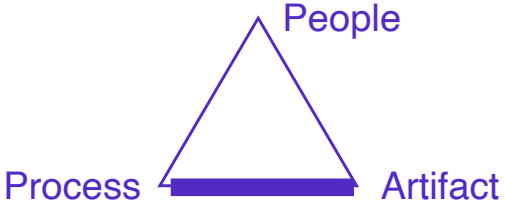
02 Unrealistic Assumptions





1990s assumption:



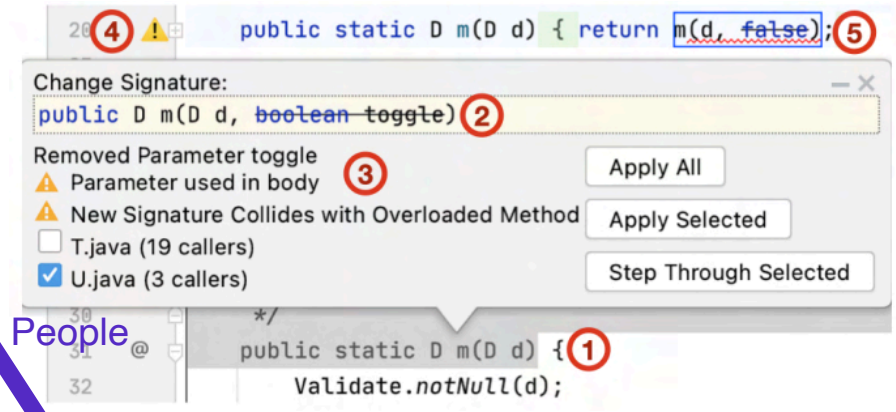


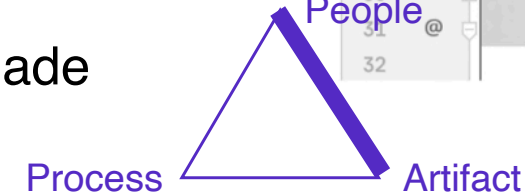
Software Reflexion Models
Murphy, Notkin and Sullivan, FSE '95

03 Relax Constraints

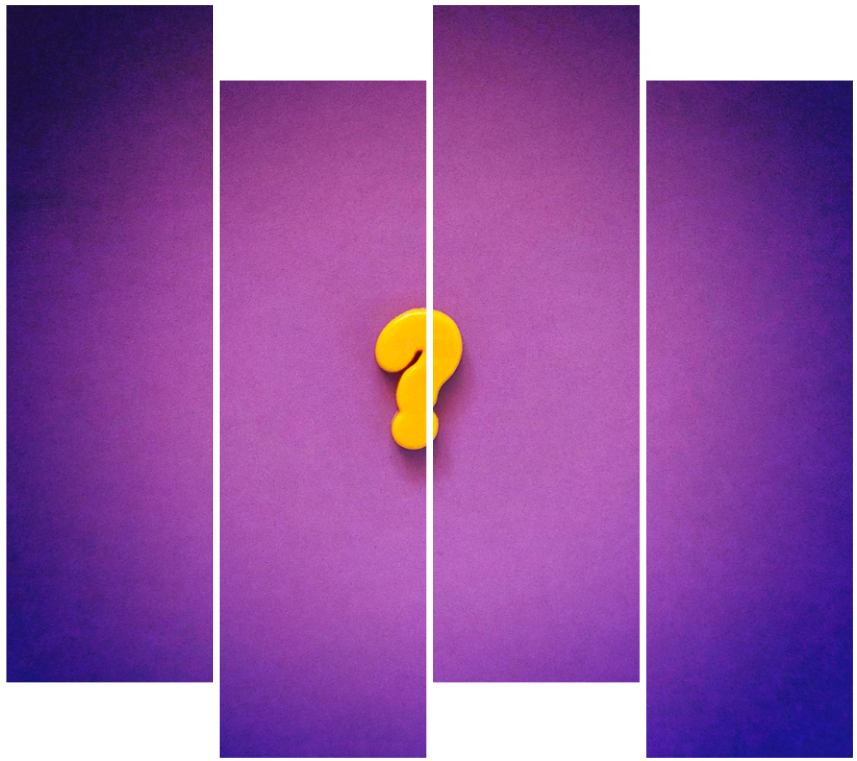
Refactoring tools take an an all or nothing approach, making potential large-scale code changes invisibly to the developer.

Make refactoring operations like a debugger so the developer can see the changes being made and be “in control”





Stepwise Refactoring Tools
Eilertsen and Murphy, ICSME ‘21



Ways to Identify Problems

01 Overly hard?

02 Unrealistic assumptions?

03 Relax constraints?

Other ways?

Let’s Try Identifying Some Potential Problems

1-2-4-All

I’ll assign groups of 4 to one of two scenarios about a company building software.

Think (1 min) about the software engineering problems there might be in the scenario. You’ll have to stretch and imagine what might be going wrong for the companies in the scenarios.

Pair with one person in your group of 4 and share the problems (2 min) you think might be a cause of the company’s problems.

Talk about the problems in your group of 4 (4 min).

We’ll share some of the potential problems identified with the group.

Let’s Try Identifying Some Potential Problems

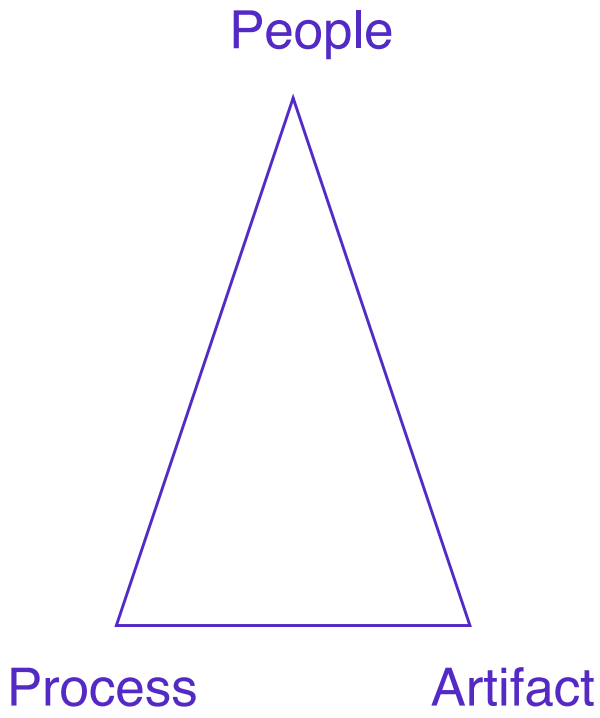
1-2-4-All

Ways to Identify Problems

01 Overly hard?

02 Unrealistic assumptions?

03 Relax constraints?



Let's Try Identifying Some Potential Problems

Scenario A

Company A builds popular mobile phone apps. It uses a continuous delivery to push updates and new features fast to its customers. Those fast deliveries provide substantial value to the customer, but also risk as security vulnerabilities related to the frameworks and libraries used in the software too often leak to customers.

Scenario B

Company B builds software for the competitive financial services industry. It has great teams of developers who work together well, but it is unable to deliver software a fast as its competitors.

Meaningful?

Impact more than one company?

Impact more than one kind of software domain?

Apply across languages? Teams? Processes?

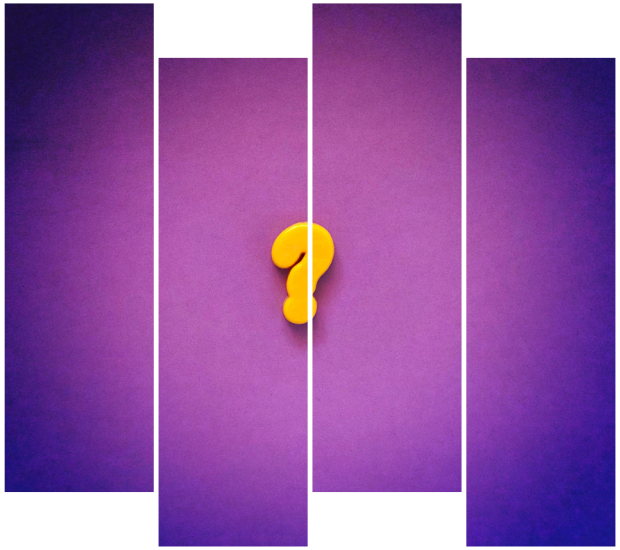
Who/what/where does the problem apply/occur?

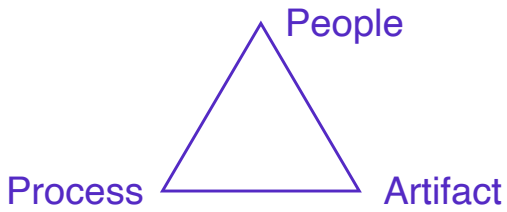
Ways to Identify Problems

01 Overly hard?

02 Unrealistic assumptions?

03 Relax constraints?





Meaningfulness

Who/what/where does this problem apply/occur?

III
Study and/or “Solve”

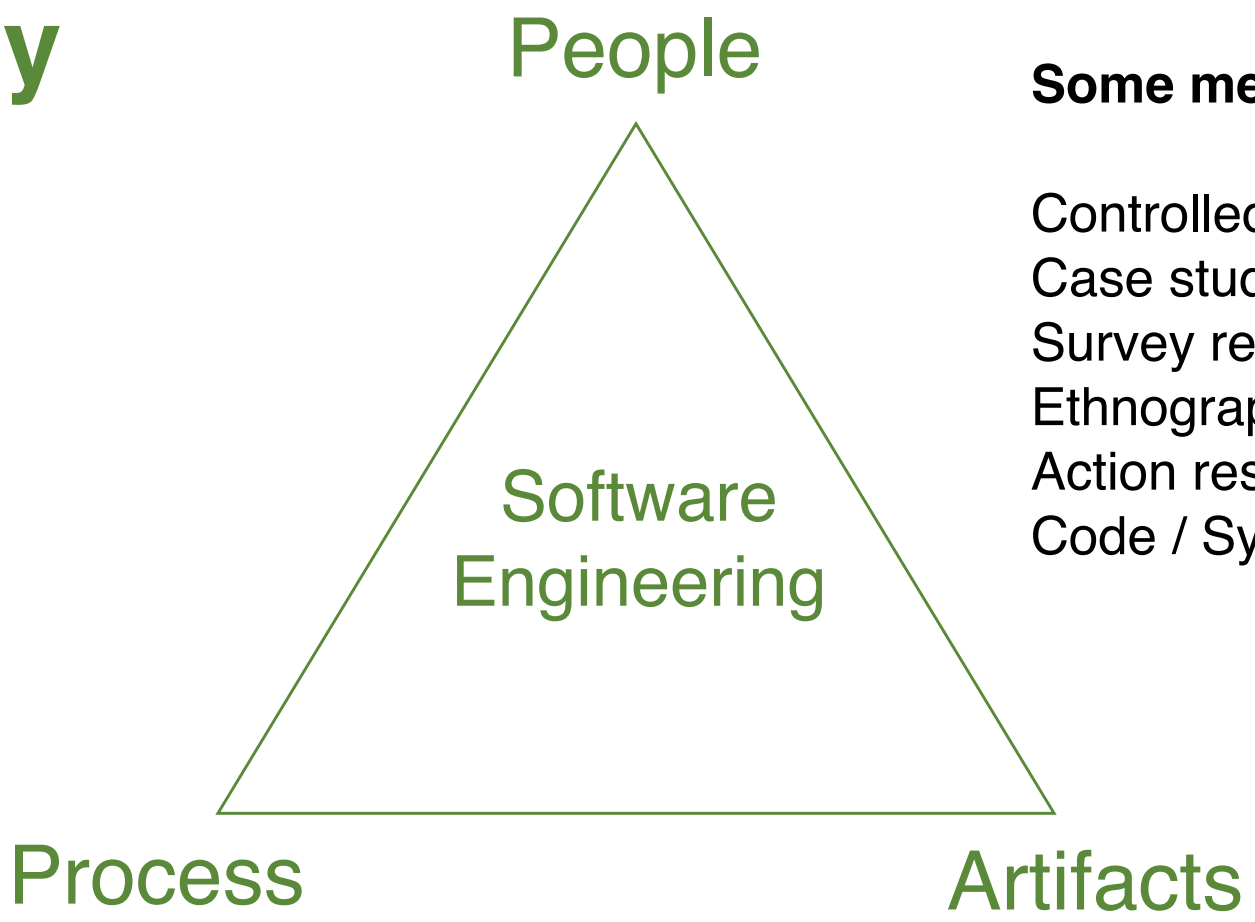
Study

characterize the phenomenas

Solve

demonstrate a new idea that addresses one or more phenomenon

Study



Some methods:

Controlled experiments
Case studies
Survey research
Ethnographies
Action research
Code / System analyses

Picking a study method

Wait, what do you know about the phenomenon to be studied?

Picking a research question

Do you first need to ***explore*** the phenomenon asking questions like “Does X exist?” or “What is X like?” or “How is X different than Y?”

Do you know enough about the phenomenon to ask about ***normal patterns of occurrence*** like “How often does X occur?” or “How does X normally occur?”

See “Selecting Empirical Methods for Software Engineering Research”, Easterbrook et al.

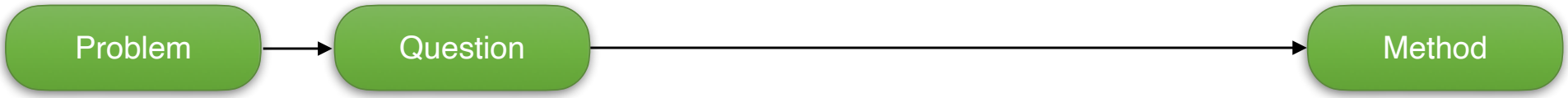
Picking a research question

Do you want to know more about the ***relationship*** between two phenomenon as in “is there a correlation between X and Y?”

Do you want to explore ***cause and effect*** as in “does X cause Y?” rr “is X better at preventing Y than Z?”

See “Selecting Empirical Methods for Software Engineering Research”, Easterbrook et al.

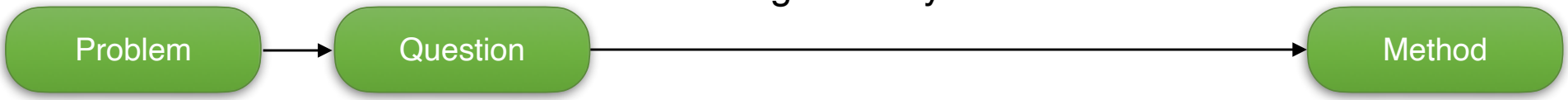
Study



Study

Other factors to consider:

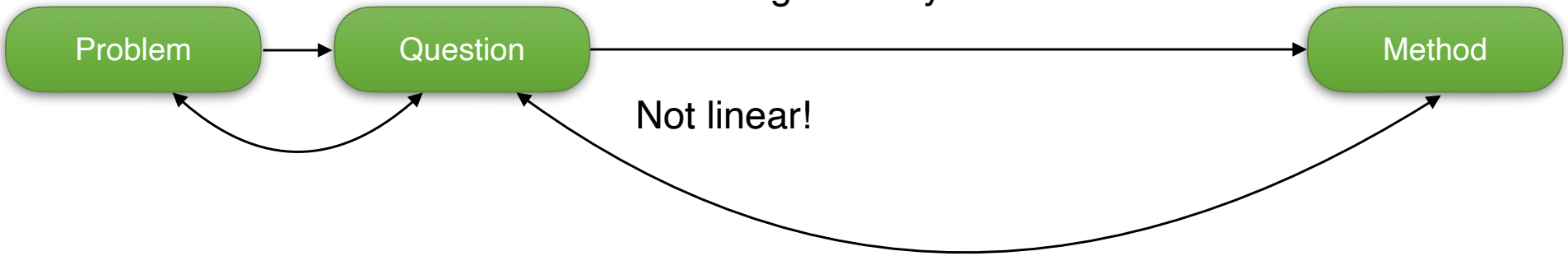
- how do you think about scientific truth?
  (e.g., positivist, constructionist? Etc.?)
- how might theory fit in?



Study

Other factors to consider:

- how do you think about scientific truth?
  (e.g., positivist, constructionist? Etc.?)
- how might theory fit in?

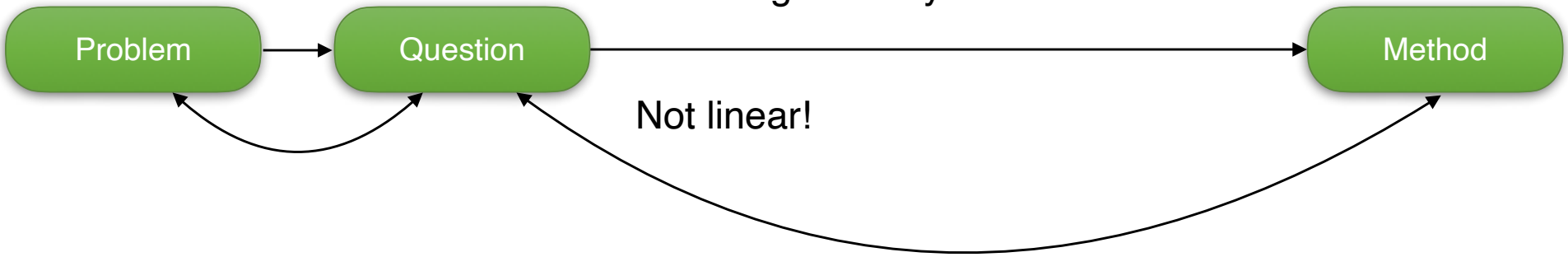


Study

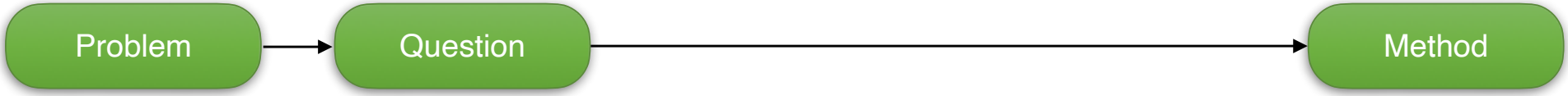
It is often hard work to figure out what you are studying exactly and how to go about studying it!

Other factors to consider:

- how do you think about scientific truth?
  (e.g., positivist, constructionist? Etc.?)
- how might theory fit in?



Some abstract examples



People	Explore	Case Study
Artifacts	Patterns of Occurrence	Code Analyses
Process	Relationship	Case Study
Process	Cause and Effect	Controlled Experiment

One of the aspects that makes software engineering research challenging and fun is how much thought you’ll need to put into the phenomenon you are studying and how you study it

Study

characterize the phenomenas

Solve

demonstrate a new idea that addresses one or more phenomenon

“Solve”

Sometimes we understand a problem well and we have new ideas for how to “solve” the problem

We introduce a new tool, method, process, etc.

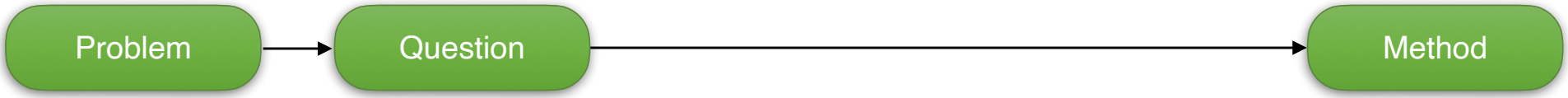
We then need to show that it “solves” the problem

Solve

More likely to use…

Some methods:

Controlled experiments
Case studies
~~Survey research~~
~~Ethnographies~~
Action research
~~Code / System analyses~~



More likely to be about…

Relationship

Cause and Effect

Note, I am making some gross generalizations!

Let’s Try Studying or “Solv”ing…

Appreciative Interview (Modified)

Find someone near you to pair with

Pick one of the problems I put up on the next screen as a pair

Think to yourself for how you might study or solve the problem (2 min)

In pairs, take 2 min each to share your approach and the other person will interview you about the benefits for 1 min. Then switch. (6 min total)

I might ask for some volunteers to share some observations you have about your discussions (depending on how we are doing on time)

Let’s Try Studying or “Solv”ing…

Appreciative Interview (Modified)

Problem #1

A developer at Company A is unaware of whether there are outstanding or new vulnerabilities in libraries on which the code they write relies.

Problem #2

A developer at Company B integrates new features once per month into a release that is manually tested.

What might you “tweak” or “alter” to move between quadrants?

Meaningful to others

Meaningful to myself and others

Meaningful to me

Study

Question

Explore

Patterns of Occurrence

Relationship

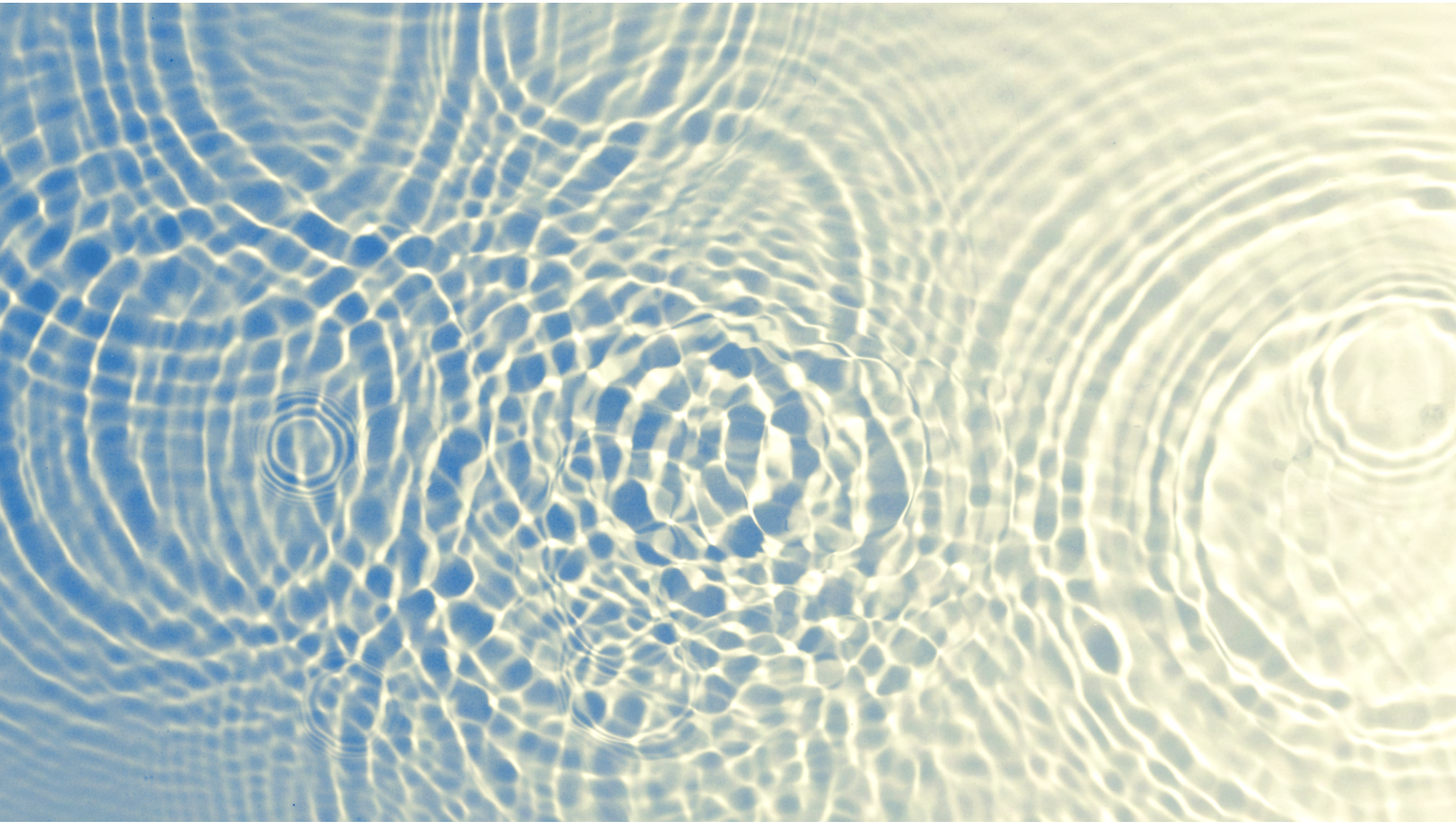
Cause and Effect

Solve

Methods

Controlled experiments
Case studies
Survey research
Ethnographies
Action research
Code / System analyses

IV
Impact



Impact through publishing

IEEE TRANSACTIONS ON
SOFTWARE ENGINEERING

Journal of Systems and Software
Supports open access

ISSTA 2025

Meaningful to others

Meaningful to others and me

Meaningful to me

The publication venue(s) you choose can impact “meanginfulness”

Each venue has a different (but not distinct) community

How else might you have impact with your research and how might you achieve that impact?

Not every piece of research you will do will have broad impact academically or impact industrial practice, but it can be helpful to reflect about if you want broader impact and if so how to achieve it

Let’s brainstorm some ways to have impact

1-2-4-All

I’ll assign groups of 4 to brainstorm how we might have impact with software engineering research.

Think (1 min) about how you might go about increasing impact of a result.

Pair with one person in your group of 4 and share the ways to increase impact (2 min).

Talk about the ways to have impact in your group of 4 (4 min).

We’ll share some of the potential ways to have impact across everyone.

Let’s brainstorm some ways to have impact

1-2-4-All

Solution #1

You have developed a new tool that checks source code when it is committed against a database of known vulnerabilities and alerts the developer if there is a new version available for software depended on with a vulnerability. The tool works for Java code.

Solution #2

You have defined a new process that enables a company to analyze their toolchain and identify improvements to speed the delivery of new features. The process is presented as a series of questions that help an organization identify and fill toolchain gaps. The process works across a wide variety of software organizations.

Some possible ways…

Tutorials

Blogs

Social Media

Start a company

Workshops

License your Technology

Talks

Mainstream media stories

A company example…

Mik Kersten

3 co-founders

Rob Elves

Tasktop Technologies

2004 - Tech development
~2005 - Started planning company
2007 - Mik Kersten finishes Ph.D.
Incorporate company
2014 - $11m Series A funding
2017 - $11m Series B funding
2019 - $7.25m Series C funding
2021 - $100m Private Equity
2022 - Acquired by Planview

A company example: Tasktop Technologies Some lessons learned

Research problems aren't the same as industry problems

Selling a new idea means showing ROI and creating a category

Customer success is critical (it isn't obvious how to use software well)

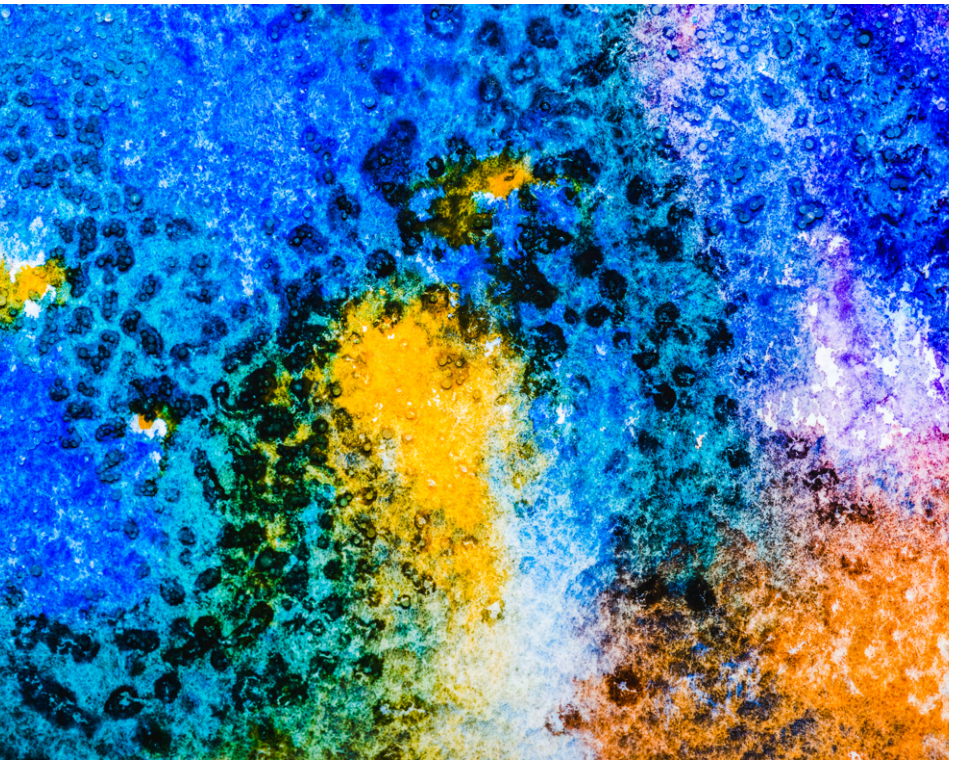
It takes a multi-talented team of more than developers to be successful

Patience is key

Continual problem solving is fun!


Need to reflect on different perspectives on your work to have broad impact

V
Summary



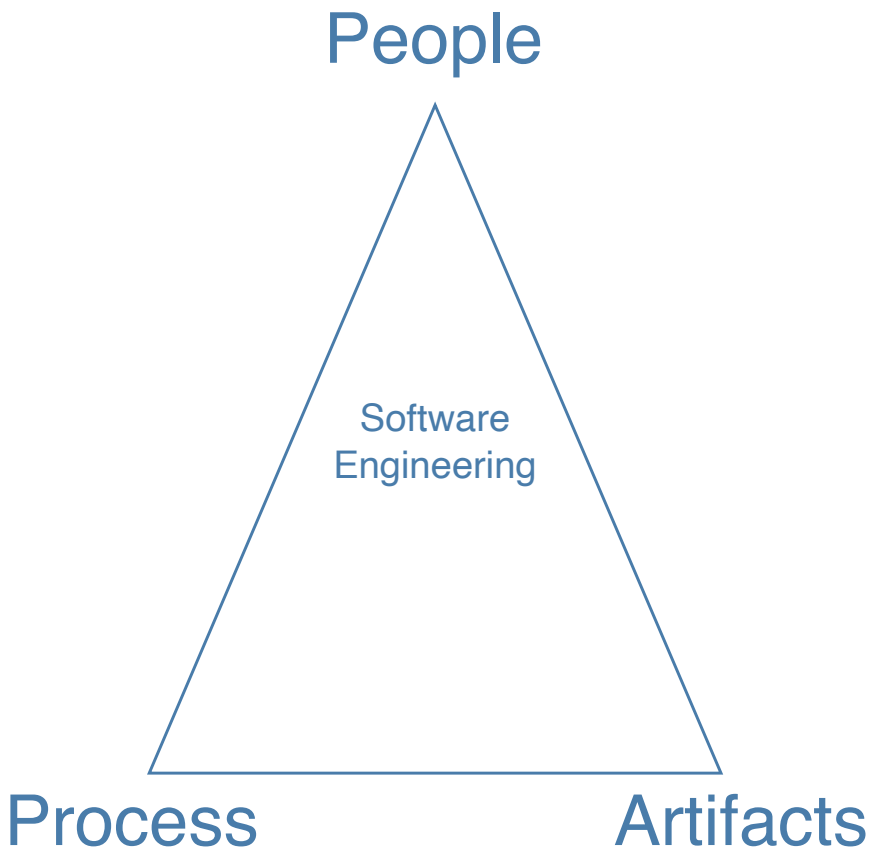
Software Engineering Research
is not abstract

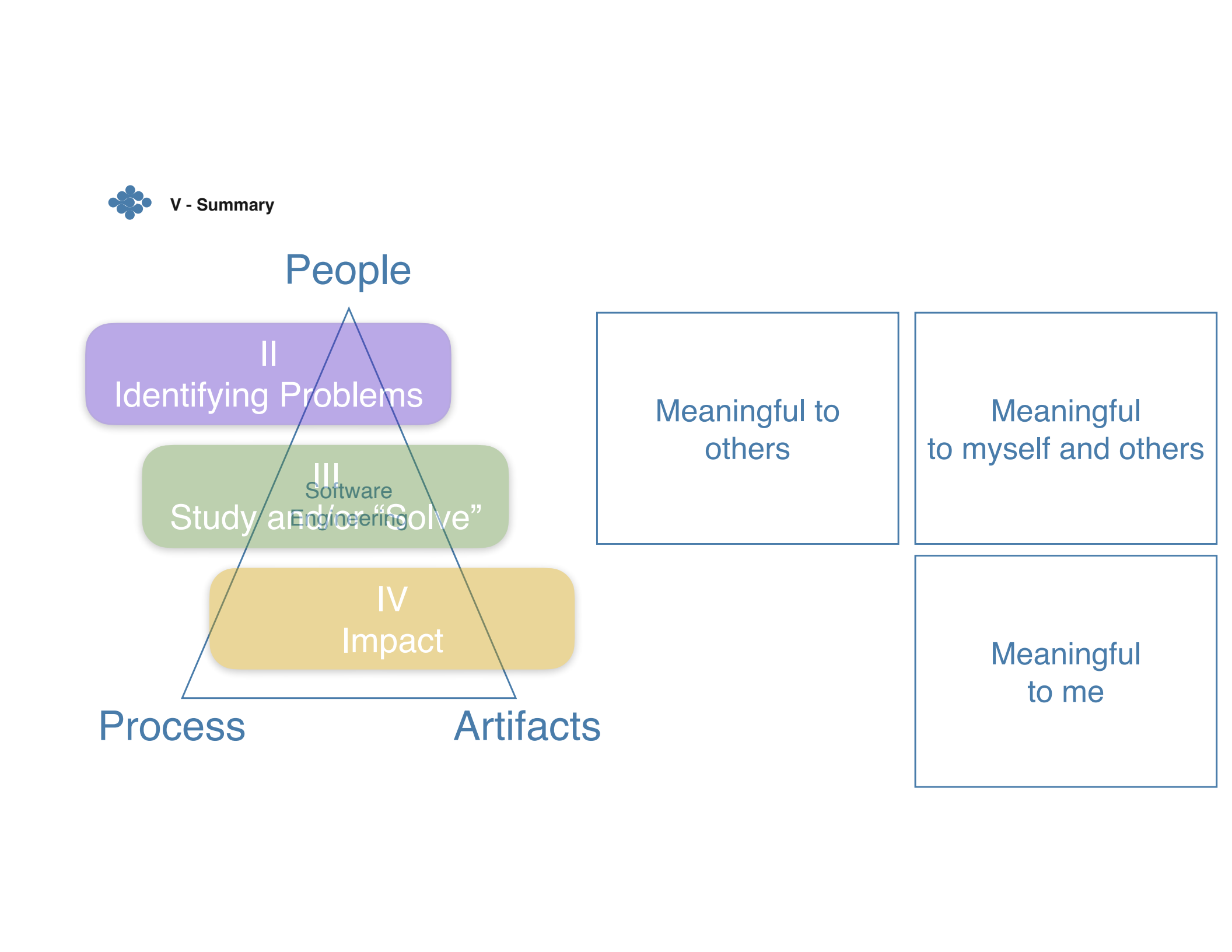
It is very concrete
as its about
multiple people
building complex
systems better

II
Identifying Problems

III
Study and/or “Solve”

IV
Impact




V - Summary
People
II
Identifying Problems
III
Study and/or "Solve"
Software
Engineering
IV
Impact
Process
Artifacts
Meaningful to
others
Meaningful
to myself and others
Meaningful
to me

You can’t tackle everything at once: Some tips

Approach your work rigorously

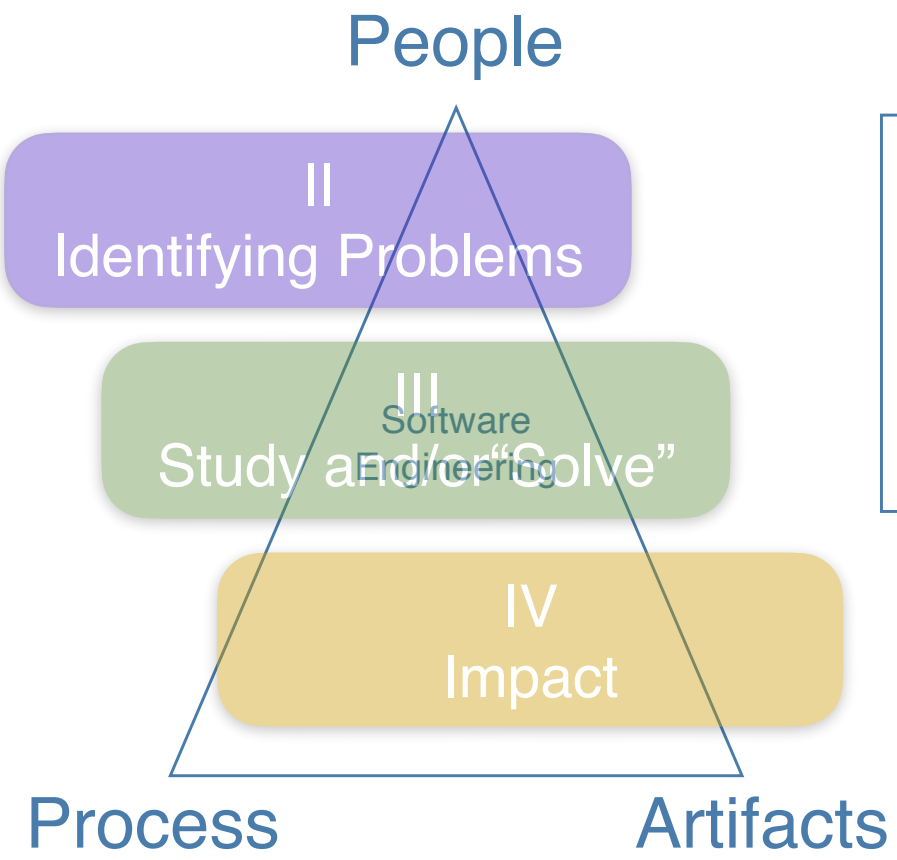
Try to hit different notes over your career

Each day is ~ 100 blocks of 10 min, use one block to reflect*

Its about continuous improvement

* Tim Urban: https://waitbutwhy.com/2016/10/100-blocks-day.html

V - Summary



Meaningful to others

Meaningful to myself and others

Meaningful to me