# Research Methods and Writing Research Papers

Prof. Dr. Roel Wieringa
University of Twente,
The Netherlands
www.cs.utwente.nl/~roelw

# Motivation

- Computing Sciences
  - Information Systems, Software engineering, Artificial Intelligence, Human-Computer Interaction, etc.
- Computing sciences have evolved into disciplines with both
  - a design component and
  - an empirical research component
- Research methodology must be properly aligned with this

  ⇒ Design Science research methodology

# Research methodology accross the disciplines

- Do these disciplines have the same methodology?
  - Technical science? Build cool stuff; test it; iterate
  - Social science? Observe people, interpret what they do or say; or select a sample, do a lot of statistics; iterate.
  - Physical science? Build instruments, create phenomena, analyze data, create theories; iterate.
  - Mathematics? Read, think, write, think; iterate.

# Mutual lack of appreciation

- Do they appreciate each other's methodology?
  - For social scientists, engineers are slightly autistic tinkerers
  - For technical scientists, social scientists are chatterboxes
  - For physicists, statistics is stamp collecting
  - Mathematicians think that they provide the foundations of civilization

# Our approach

- All research in all disciplines is **problem-solving**
  - **design problems:** goal is to design something useful, research method is the design cycle
  - **knowledge questions:** goal is to acquire theoretical knowledge, research method is the empirical cycle

- Wieringa, R.J. (2014) *Design science methodology for information systems and software engineering.* Springer Verlag

# Outline

1. **What is design science**
   - Research goals and problems
   - The design cycle
2. Theories
   - Conceptual frameworks
   - Generalizations
3. Empirical research
   - Scientific inference
   - Research design

- Design science is the **design** and **investigation** of **artifacts in context**

# Reality check

- What is the artifact and what is the context?
  - SIKS dissertations http://www.siks.nl/dissertations.php
  - Master theses in business informatics http://essay.utwente.nl/view/programme/60025.html
  - Master theses in computer science http://essay.utwente.nl/view/programme/60300.html
  - Master theses in human-media interaction http://essay.utwente.nl/view/programme/60030.html
- What research problem(s) are you investigating?
  - Artifact and context

- The title of your thesis is the shortest summary of your research project.
- Often, it mentions the artifact and the context.

# Two kinds of research problems in design science

**To design an artifact to improve a problem context**

→ Problems & Artifacts to investigate →

← Knowledge, Design problems ←

**To answer knowledge questions about the artifact in context**

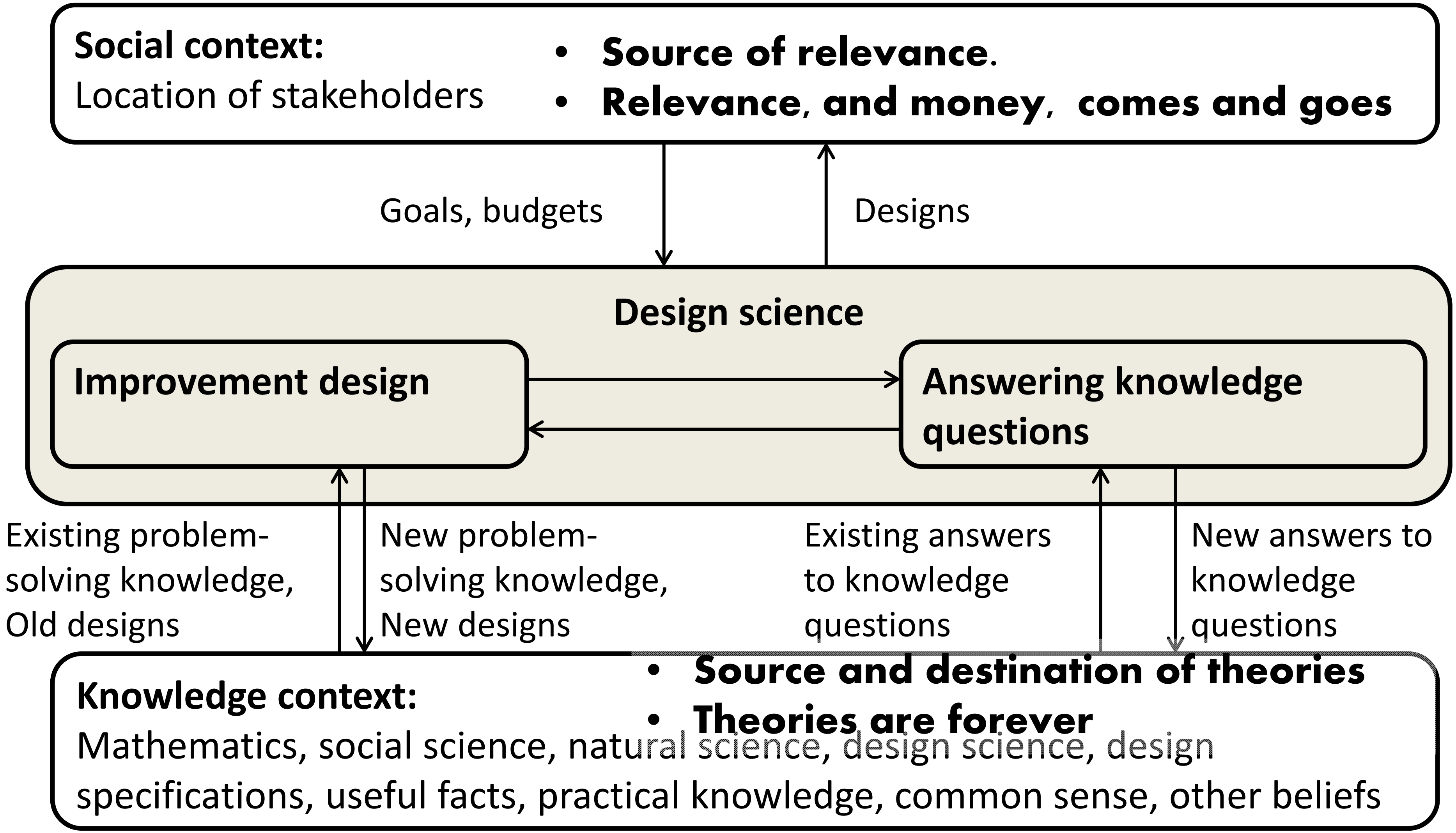
- *Design software to estimate Direction of Arrival of plane waves, to be used in satelite TV receivers in cars*
- *Design a Multi-Agent Route Planning system to be used for aircraft taxi route planning*
- *Design a data location regulation auditing method*

Is the artifact **useful**?

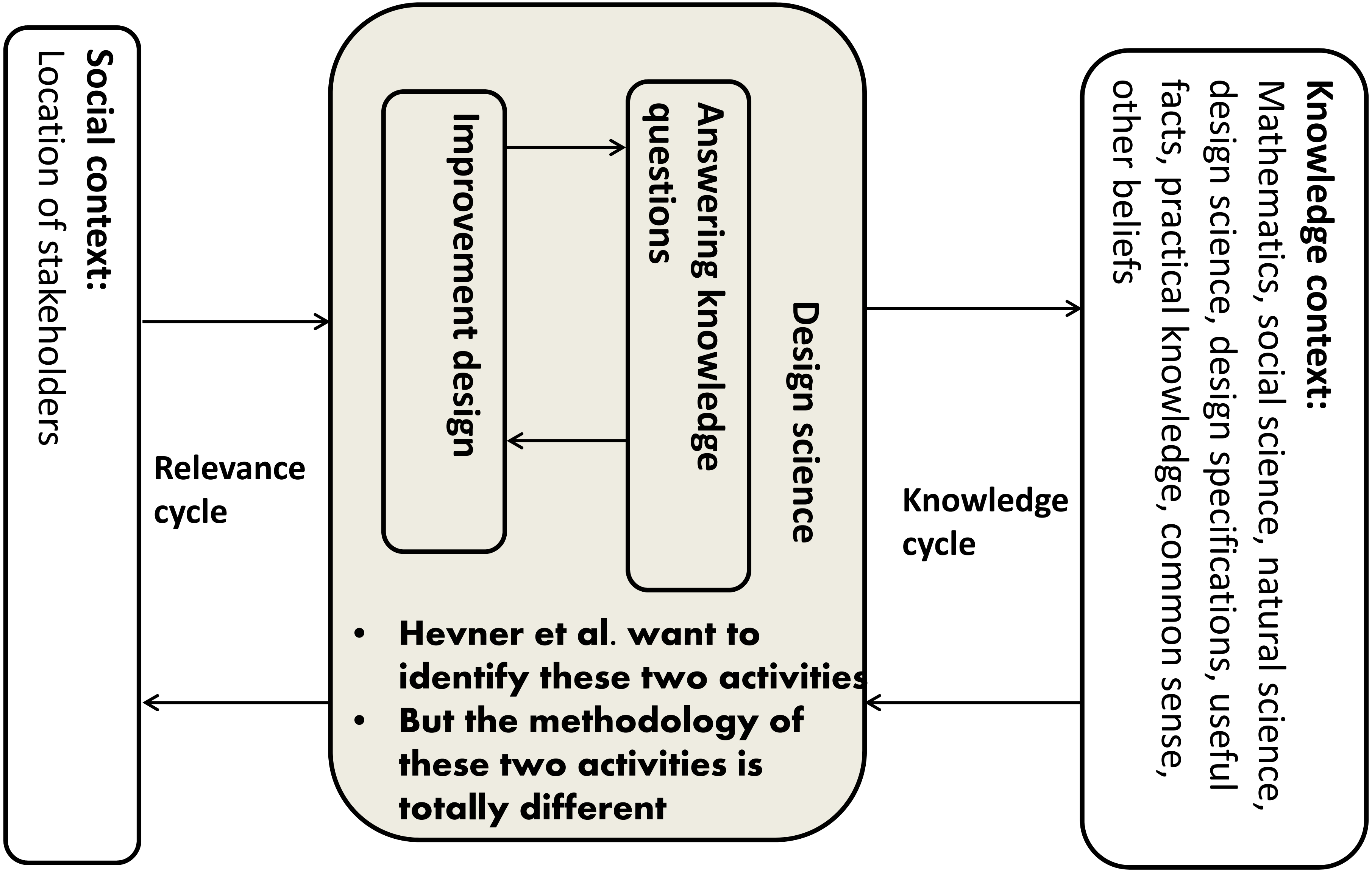
- *Is the DoA estimation accurate enough`in this context?*
- *Is it fast enough?*
- *Is this routing algorithm deadlock-free on airports?*
- *How much delay does it produce?*
- *Is the method usable and useful for consultants?*

Is the answer **true**?

# Framework for design science



# (Dis)similarity to Hevner et al. framework



# Reality check

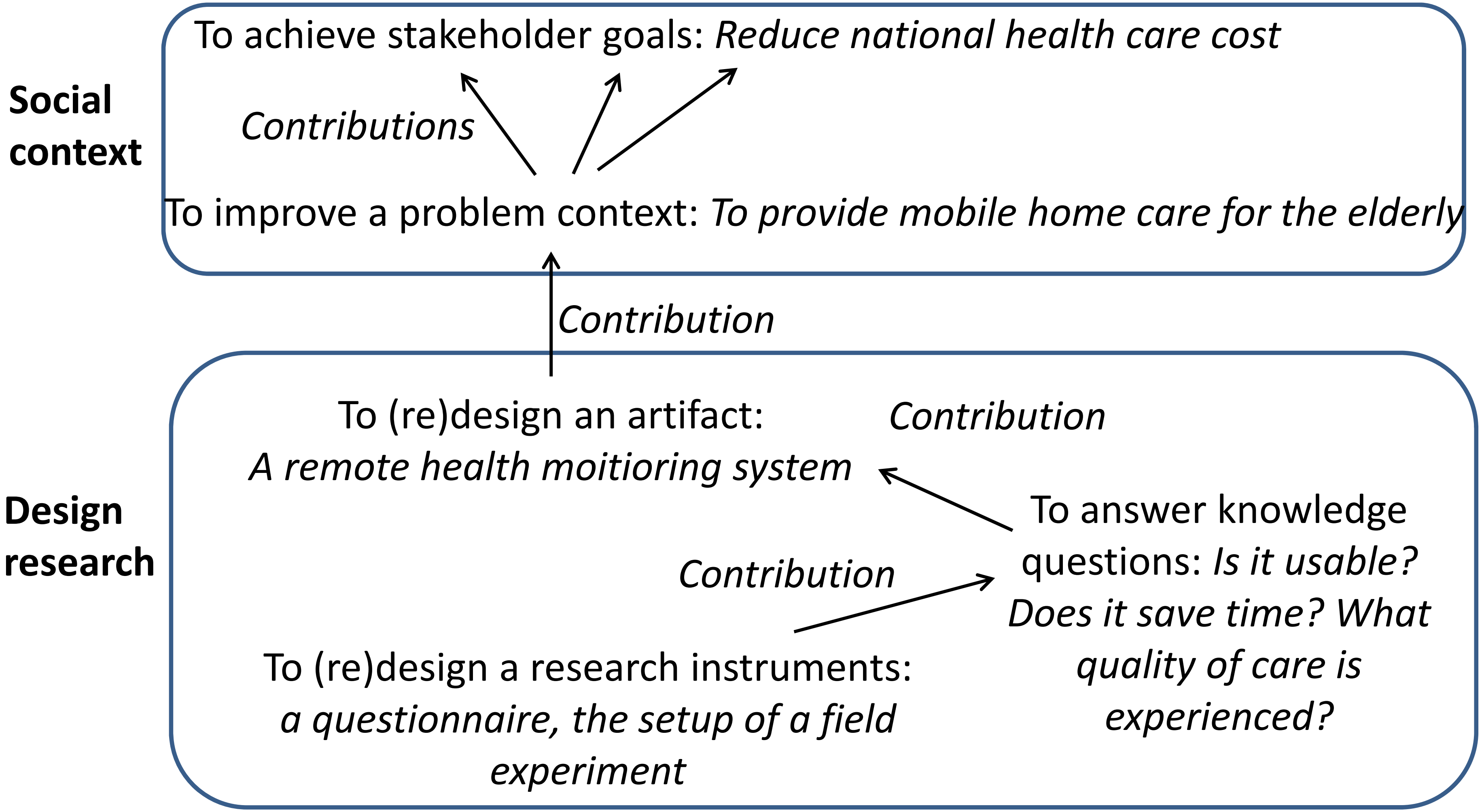
- Who are the stakeholders of your project?
  - Real or hypothetical: Stakeholders may not know they are stakeholders
- What are their goals?
  - Motivates your expectation of positive or negative utility
- What knowledge do you hope to produce?
  - Design theories, design specifications, useful facts, practical knowledge

Tell this in your elevator pitch

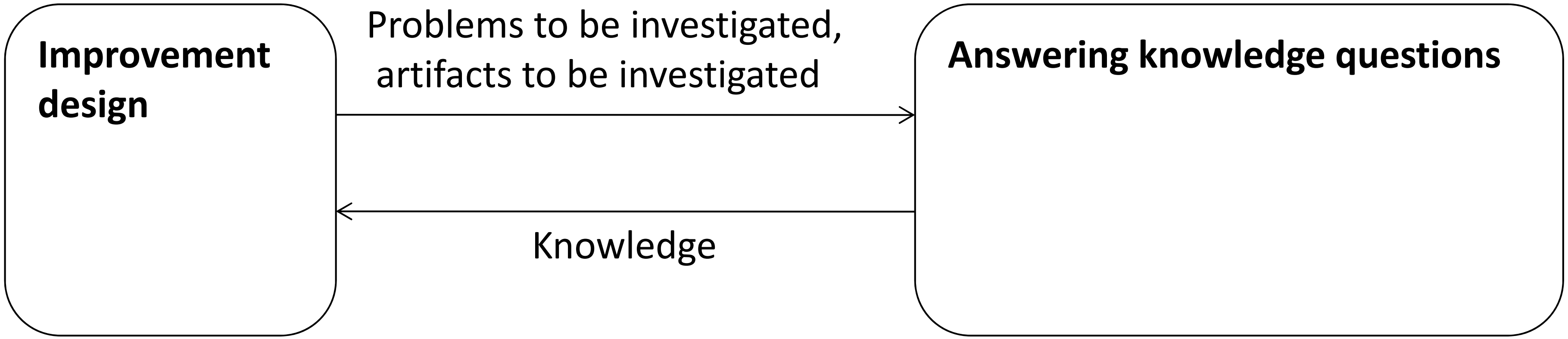
# Outline

1. What is design science
   - **Research goals and problems**
   - The design cycle
2. Theories
   - Conceptual frameworks
   - Generalizations
3. Empirical research
   - Scientific inference
   - Research design

# Goal structure: example



# **Three** kinds of design research questions




1. **Design research problems** (a.k.a. *technical research questions*)
   – To improve some kind of artifact in some kind of context.

2. **Empirical knowledge questions**
   – To ask questions about the real world.

3. **Analytical knowledge questions**
   – To ask questions about the logical consequences of definitions

# Template for design problems

- Improve <problem context>
- by <treating it with a (re)designed artifact>
- such that <artifact requirements>
- in order to <stakeholder goals>

- *Reduce my headache*
- *by taking a medicine*
- *that reduces pain fast and is safe*
- *in order for me to get back to work*

# Template for design problems

- Improve <problem context>
- by <treating it with a (re)designed artifact>
- such that <artifact requirements>
- in order to <stakeholder goals>

- ***Reduce my headache***
- *by taking a medicine*
- *that reduces pain fast and is safe*
- ***in order for me to get back to work***

**Problem context and stakeholder goals.**

Stakeholder language

# Template for design problems

- Improve <problem context>
- by <treating it with a (re)designed artifact>
- such that <artifact requirements>
- in order to <stakeholder goals>

- *Reduce my headache*
- ***by taking a medicine***
- ***that reduces pain fast and is safe***
- *in order for me to get back to work*

**Artifact and its desired properties.**

Technical language

# Template for design **research** problems

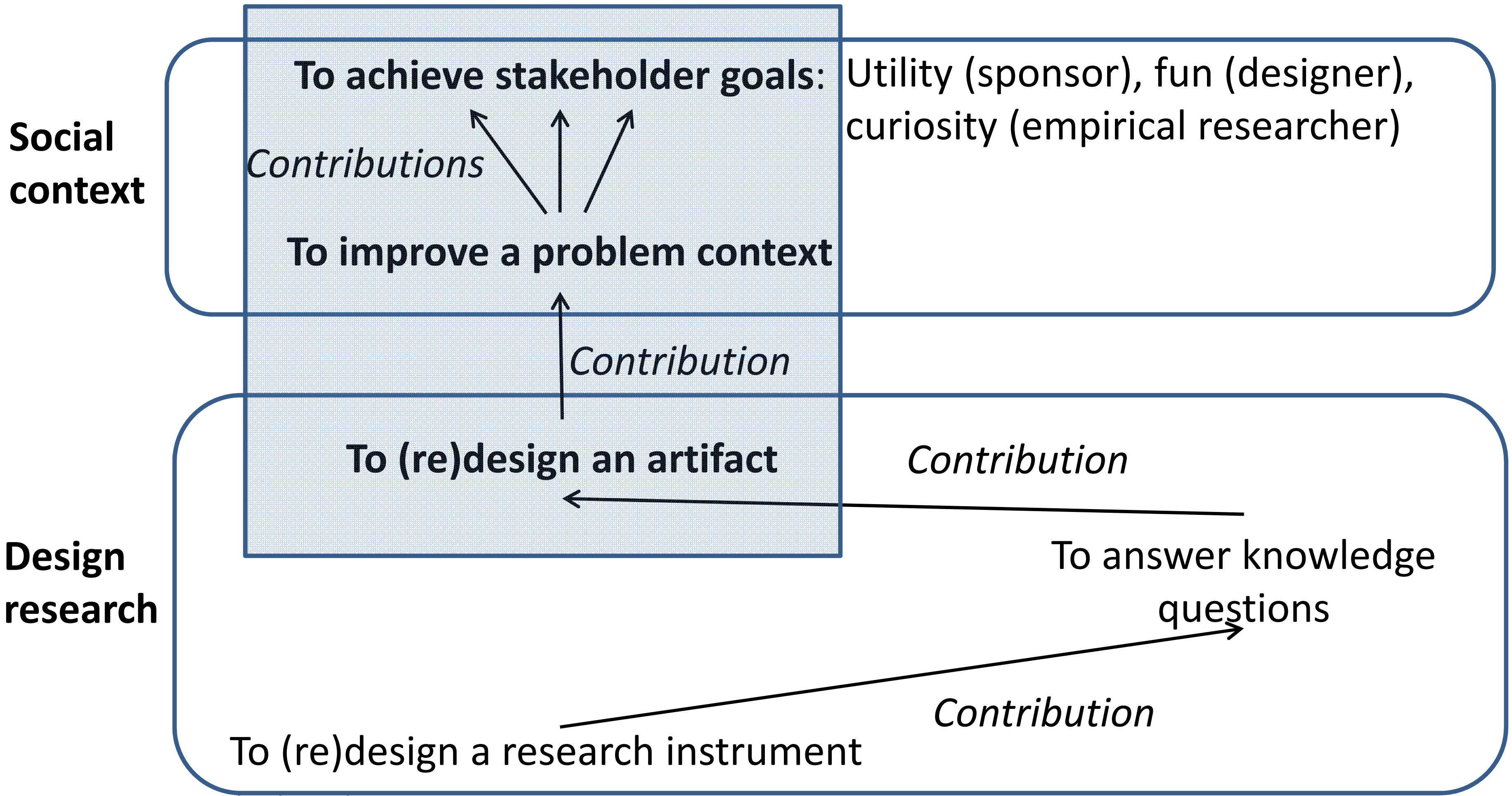
- Improve <problem context>
- by <treating it with a (re)designed artifact>
- such that <artifact requirements>
- in order to <stakeholder goals>

- *Reduce patients' headaches*
- *by treating it with a medicine*
- *that reduces pain fast and is safe*
- *in order for them to function as they wish*

The problem is now to design an artifact that helps a **class** of stakeholders achieve a **class** of goals.

# Goal structure again

- The design problem template links the artifact to the problem context and stakeholder goals



# Poster (1)
# 5 minutes

- Write down your top-level design problem, using this template.

- NB
  - some parts may be *currently* uncertain, fuzzy, or unknown.
  - But surely, some parts are currently known!

# There is no single “correct”problem statement

- A good problem statement forces the reader to think focussed about the artifact while remaining aware of the intended problem context

- Next two examples extracted from two M.Sc theses
  - http://essay.utwente.nl/67945/
  - http://essay.utwente.nl/69399/

- *BPMN Plus : a modelling language for unstructured business processes.* ← Artifact ← Context
- The objective of this study is
  - To investigate the way through which unstructured business processes can be modelled and managed without limiting their run-time flexibility.
- Research questions
  - Q1 What are the differences between structured and unstructured business processes?
  - Q2 What are the differences between Business Process Management and Case Management in dealing with unstructured business processes?
  - Q3 What are the capabilities of existing modelling notations to deal with unstructured business processes?
  - Q4 How to model an unstructured business process while providing run-time flexibility?

- **Improve <problem context in which unstructured business process is to be modelled>**
- **by <introducing a modeling language for unstructured business processes>**
- **such that <requirements such as run-time flexibility, and ... learnability etc?>**
- **in order to <stakeholder goals, e.g. provide better process improvement advice to clients>**

- *Automated generation of attack trees by unfolding graph transformation systems.* ← Context ← Artifact
  - RQ1: Can graph transformation be used as a modeling paradigm to specify systems and organizations as input models for the attack tree generation approach?
  - RQ2: Can partial-order reduction, and specifically the unfolding of a graph transformation model, be used to reduce the state-space explosion problem that occurs during the automated exploration of a model?
  - RQ3: How can the set of attacks be converted into an attack tree, what are the trade-offs and how can additional information such as sequential AND's be included in the tree?

- **Improve <attack tree generation>**
- **by <graph transformation system>**
- **such that <artifact requirements, e.g. faster generation of bigger attack trees>**
- **in order to <stakeholder goals, e.g. security risk assessment is more complete>**

# Three kinds of design research questions

1. Design problems (a.k.a. *technical research questions*)
   – To improve some artifact in some context.

- Knowledge questions
  2. Empirical knowledge questions
  3. Analytical knowledge questions (math, conceptual, logical). We ignore these in this course.

# Empirical knowledge questions

- Descriptive knowledge questions:
  - What happened?
  - How much? How often?
  - When? Where?
  - What components were involved?
  - Who was involved?
  - Etc. etc.

  **Journalistic questions. Yield facts.**

- Explanatory knowledge questions:
  - Why?
    1. What has **caused** the phenomena?
    2. Which **mechanisms** produced the phenomena?
    3. For what **reasons** did people do this?

  **Beyond the facts.**

- *BPMN Plus : a modelling language for unstructured business processes.*
- The objective of this study is
  - To investigate the way through which the unstructured business processes can be modelled and managed without limiting their run-time flexibility.
- Research questions
  - Q1 What are the differences between structured and unstructured business processes?
  - Q2 What are the differences between Business Process Management and Case Management in dealing with unstructured business processes?
  - Q3 What are the capabilities of existing modelling notations to deal with unstructured business processes?
  - Q4 How to model an unstructured business process while providing run-time flexibility?

- Explanatory questions?
- Analytical questions?

Descriptive knowledge questions; (outcome of interviews) [Q1–Q3]

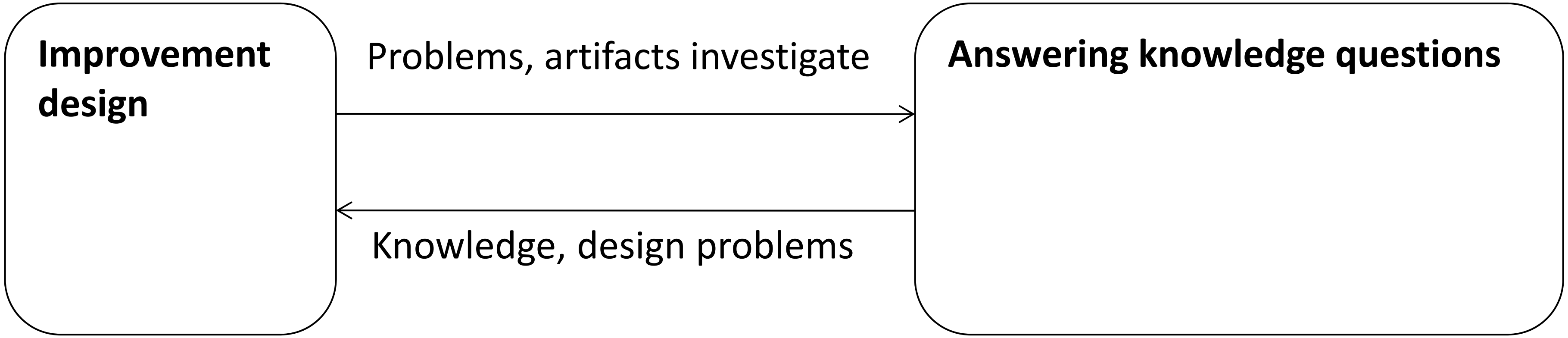
Design problem [Q4]

- *Automated generation of attack trees by unfolding graph transformation systems.*
  - RQ1: Can graph transformation be used as a modeling paradigm to specify systems and organizations as input models for the attack tree generation approach?
  - RQ2: Can partial-order reduction, and specifically the unfolding of a graph transformation model, be used to reduce the state-space explosion problem that occurs during the automated exploration of a model?
  - RQ3: How can the set of attacks be converted into an attack tree, what are the trade-offs and how can additional information such as sequential AND's be included in the tree?

} Design problems

- Descriptive questions?
- Explanatory questions?
- Analytical questions?

# Summary

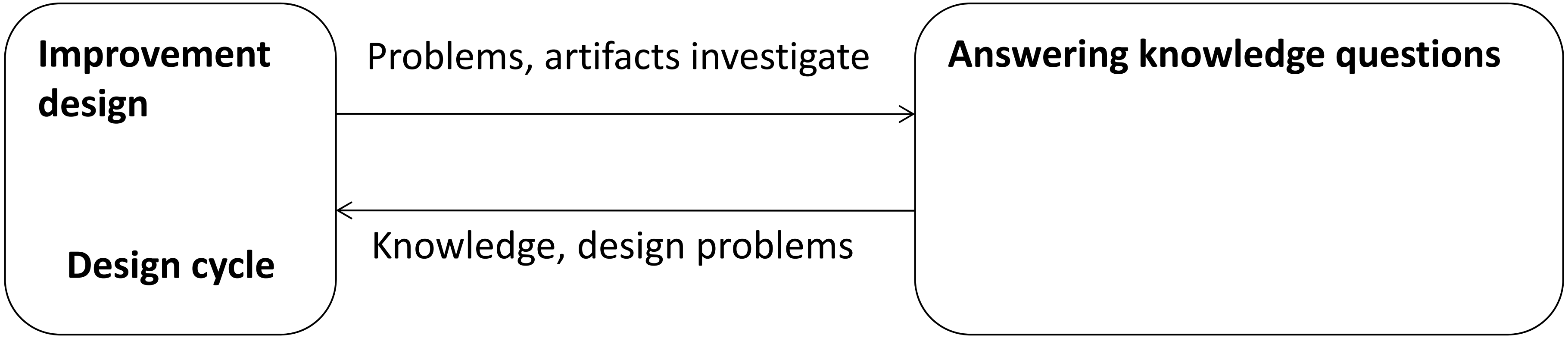



1. **Design research problems** (a.k.a. *technical research questions*)
   - **Improve <problem context>**
   - **by <treating it with a (re)designed artifact>**
   - **such that <artifact requirements>**
   - **in order to <stakeholder goals>**.
2. **Empirical knowledge questions**
   - Descriptive: what, how, when, where, who, etc.→ **Facts**
   - Explanatory: Why → **Explanations**
3. **Analytical knowledge questions**
   - Yields definitions, assumptions, theorems.

# Outline

1. What is design science
   – Research goals and problems
   – **The design cycle**
2. Theories
   – Conceptual frameworks
   – Generalizations
3. Empirical research
   – Scientific inference
   – Research design

# How to solve design problems

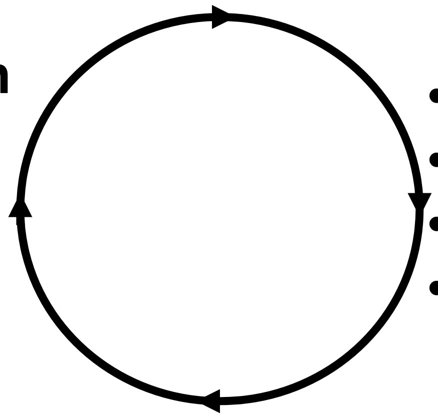



1. Design research problems (a.k.a. *technical research questions*)
   – To improve some kind of artifact in some kind of context.
2. Empirical knowledge questions
   – To ask questions about the real world.
3. Analytical knowledge questions
   – To ask questions about the logical consequences of definitions

# Engineering cycle

! = **Action**
? = **Knowledge question**

**This is a checklist. See appendix A in the book & on my web site**



## Implementation evaluation = Problem investigation

- Stakeholders? Goals?
- Conceptual problem framework?
- Phenomena? Causes, mechanisms, reasons?
- Effects? Positive/negative goal contribution?

## Treatment design

- Specify requirements!
- Requirements contribute to goals?
- Available treatments?
- Design new ones!

## Treatment validation

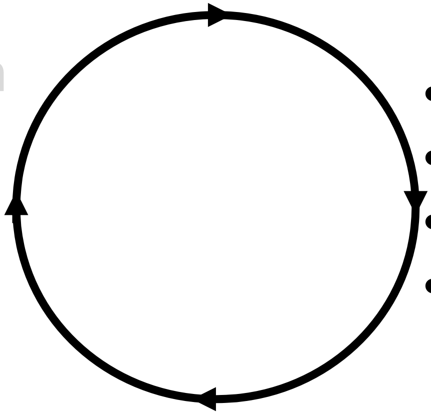
- Context & Artifact → Effects?
- Effects satisfy Requirements?
- Trade-offs for different artifacts?
- Sensitivity for different Contexts?

## Treatment implementation

# Engineering cycle

! = Action
? = **Knowledge question**

Treatment implementation



**Implementation evaluation = Problem investigation**

- Stakeholders? Goals?
- Conceptual problem framework?
- Phenomena? Causes, mechanisms, reasons?
- Effects? Positive/negative goal contribution?

**Treatment validation**

- Context & Artifact → Effects?
- Effects satisfy Requirements?
- Trade-offs for different artifacts?
- Sensitivity for different Contexts?

**Treatment design**

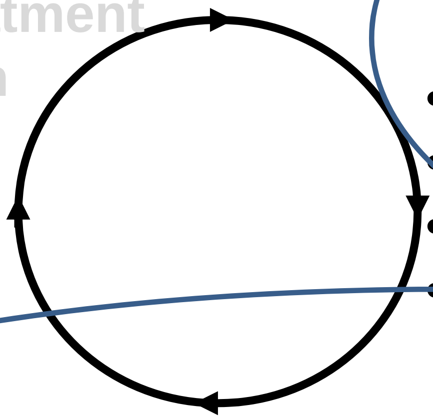
- Specify requirements!
- Requirements contribute to goals?
- Available treatments?
- Design new ones!

# Design cycle

! = **Action**
? = **Knowledge question**

**Real-world problem-oriented research**

Real-world treatment implementation



## Real-world mplementation evaluation = Real-world problem investigation

- Stakeholders? Goals?
- Conceptual problem framework?
- Phenomena? Causes, mechanisms, reasons?
- Effects? Positive/negative goal contribution?

## Treatment validation

- Context & Artifact → Effects?
- Effects satisfy Requirements?
- Trade-offs for different artifacts?
- Sensitivity for different Contexts?

## Treatment design

- Specify requirements!
- Requirements contribute to goals?
- Available treatments?
- Design new ones!

**Solution-oriented research**

# **Implementation** is introducing the treatment in the intended problem context

- If problem context is a **real-world** context…. implementation of a solution is **technology transfer to the real world**.
  - Not part of a research project

- If the problem is to learn about the performance of a design ... Implementation of a solution is the **construction of a prototype and test environment**.
  - Part of a research project

# Nesting of cycles

M.Sc. or PhD project

| | |
|---|---|
| **Problem investigation** | |
| **Treatment design** | |
| **Treatment validation** | Problem investigation (How to test the design?) |
| | Treatment design (design a prototype) |
| | Implementation (prototype construction) |
| | Evaluation (in the laboratory or field) |
| Implementation (tech transfer) | |
| Implementation evaluation (in the field) | |

This is a very special engineering cycle. Later we will call this the empirical cycle. It is performed to answer empirical knowledge questions

# Validation versus evaluation

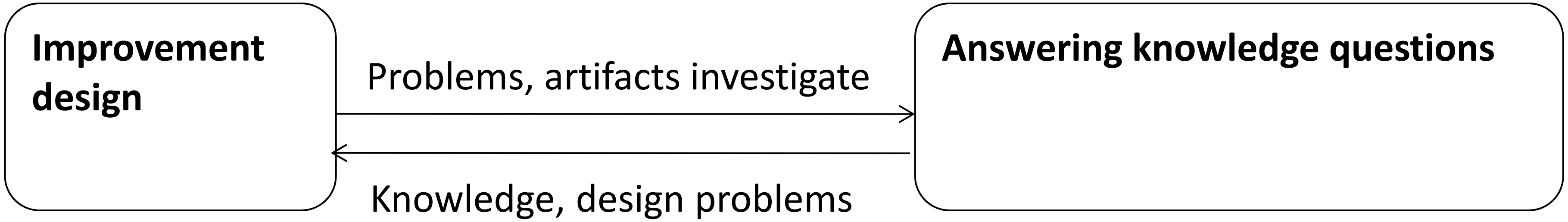
- To **validate** a design is to *predict* the effects of an implementation of the design, and the utility with respect to stakeholder goals.
  - To do this, you need a theory of the artifact in context: a design theory.
  - Many theses describe a problem or improvement opportunity, one or more treatment designs, and one or more treatment validations.
- To **evaluate** an implementation is to investigate the effects of an implementation *that have occurred* and their utility with respect to stakeholder goals
  - Now you can see what has actually happened, and possibly improve your design theory.
  - Some theses evaluate currently implemented solutions extensively before proposing and validating new ones.

# Requirements

- You specify the requirements based on your analysis of stakeholder goals
  - Even if the stakeholders do not know they are stakeholders
  - Or if they have no goals

- Your validation knowledge questions are about the requirements!
  - *Execution speed?*
  - *Memory usage?*
  - *Usability?*
  - *Reliability?*
  - …

# Summary




**1. Design research problems**

- Improve <problem context>
- by <treating it with a (re)designed artifact>
- such that <artifact requirements>
- in order to <stakeholder goals>.

**Design cycle**

- Problem investigation
- Treatment design
- Treatment validation

**Strategy**

Artifacts → Design cycle → Artifacts

2. **Empirical knowledge questions**
   - Descriptive: what, how, when, where, who, etc.→ **Facts**
   - Explanatory: Why → **Theories**
3. **Analytical knowledge questions**
   - Yields definitions, assumptions, theorems.

- *BPMN Plus : a modelling language for unstructured business processes.*
- The objective of this study is
  - To investigate the way through which the unstructured business processes can be modelled and managed without limiting their run-time flexibility.
- Research questions
  - Q1 What are the differences between structured and unstructured business processes?
  - Q2 What are the differences between Business Process Management and Case Management in dealing with unstructured business processes?
  - Q3 What are the capabilities of existing modelling notations to deal with unstructured business processes?
  - Q4 How to model an unstructured business process while providing run-time flexibility?
- "The practical usefulness of newly proposed modelling notation is investigated by demonstrating it with the help of an example.
- Moreover, the proposed modelling notation is validated by conducting interviews with experienced practitioners."

**Problem**

- Stakeholders? Goals? : *BiZZDesign consultants. To provide high-quality consultancy.*
- Conceptual problem framework? *Business process modelling, structured & unstructured. See Q1.*
- Phenomena? Causes, mechanisms, reasons? *BPMN does not allow for modelling flexible business processes; but case-management systems almost impose no constraints. Simple explanations: the languages lack facilities. See Q2.*
- Effects? Positive/negative goal contribution? *Limits to consultancy advice.*

**Treatment**

- Specify requirements! *Omitted research question. May be part of Q2.*
- Requirements contribute to goals? *Omitted too.*
- Available treatments? *See Q3.*
- Design new ones! *See Q4.*

**Validation** *Omitted questions, but done by means of interviews.*

- Context & Artifact → Effects? *Does it work?*
- Effects satisfy Requirements? *Does it work as desired?*
- Trade-offs for different artifacts? *Performance of different languages, similar cases?*
- Sensitivity for different Contexts? *Does it work in different cases*?

- *Automated generation of attack trees by unfolding graph transformation systems.*
  - RQ1: Can graph transformation be used as a modeling paradigm to specify systems and organizations as input models for the attack tree generation approach?
  - RQ2: Can partial-order reduction, and specifically the unfolding of a graph transformation model, be used to reduce the state-space explosion problem that occurs during the automated exploration of a model?
  - RQ3: How can the set of attacks be converted into an attack tree, what are the trade-offs and how can additional information such as sequential AND's be included in the tree?

**Problem** *Implied, no further details.*

- Stakeholders? Goals?
- Conceptual problem framework?
- Phenomena? Causes, mechanisms, reasons?
- Effects? Positive/negative goal contribution?

- **Treatment**
- Specify requirements! *Omitted RQ, presumably scalability (RQ2).*
- Requirements contribute to goals?
- Available treatments?
- Design new ones! *RQ1, RQ2, RQ3.*

- **Validation** *Omitted RQs*
- Context & Artifact → Effects?
- Effects satisfy Requirements?
- Trade-offs for different artifacts?
- Sensitivity for different Contexts?

# Poster (2)
# 10 minutes

- Make an outline of the table of contents of your thesis, following the design cycle.
  - Include the top-level design problem, including the problem context and stakeholder goals that motivate your design problem.
  - In the empirical research chapters (implementation evaluation/problem investigation, treatment validation), include the knowledge questions as far as you can now (fore)see them.

# Outline

1. What is design science
   – Research goals and problems
   – The design cycle
2. **Theories**
   – Conceptual frameworks
   – Generalizations
3. Empirical research
   – Scientific inference
   – Research design

# Facts, explanations, theories

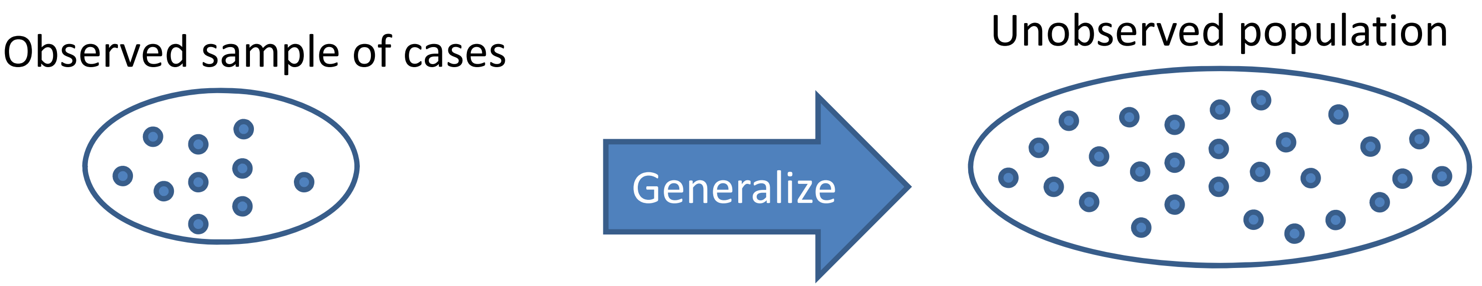
- Descriptive knowledge questions:
  - What happened?
  - How much? How often?
  - When? Where?
  - What components were involved?
  - Who was involved?
  - Etc. etc.

- Facts.
- Generalizable to **descriptive theories.**

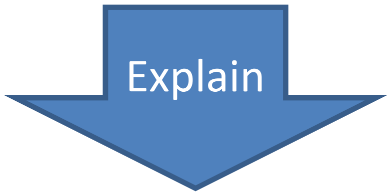
- Explanatory knowledge questions:
  - Why?
    - What caused this phenomenon?
    - What mecanisms produced it?
    - Why did people do this?

- Explanations.
- Generalizable to **explanatory theories**

# Two ways to go beyond the facts

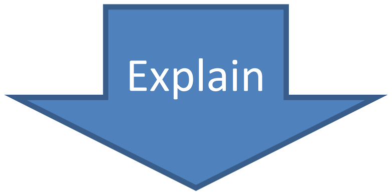



- What happens in these **cases**?
- What average, variance in this **sample**?




- Why?

- What happens in all cases?
- What average, variance in this population?

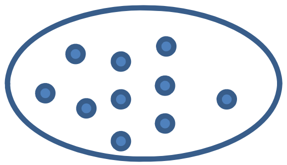



- Why?

# Facts versus theories

**Facts**

Observed sample of cases

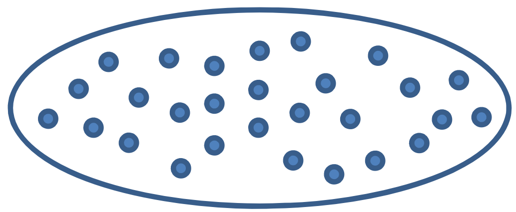


- What happens in these **cases**?
- What average, variance in this **sample**?

Generalize

**Descriptive theory of the population**

Unobserved population



- What happens in all cases?
- What average, variance in this population?

Explain

- Why?

**Explanatory theory of the case/sample**

Explain

- Why?

**Explanatory theory of the population**

# What is a theory?

- A **theory** is a belief that there is a pattern in phenomena.
  - Idealizations: *“Merging two faculties reduces cost.” “This works in theory, but not in practice.”*
  - Speculations: *“The NSA is monitoring all my email.”*
  - Opinions: *“The Dutch lost the soccer competition because they are not a team.’’*
  - Wishful thinking: *``My technique works better than the others.’’*
  - **Scientific theories:** *Theory of electromagnetism*

# Scientific theories

- A **scientific** theory is a belief that there is a pattern in phenomena, that has survived
  - Tests against experience:
    - Observation, measurement
    - Possibly: experiment, simulation, trials
  - Criticism by critical peers:
    - Anonymous peer review
    - Publication
    - Replication

- Examples
  - *Theory of electromagnetism*
  - *Technology acceptance model*
  - *Theory of the UML*

- Non-examples
  - *Religious beliefs*
  - *Political ideology*
  - *Marketing messages*
  - *Most social network discussions*

# Anti-twitter

- The discussion is conducted with everyone
  - Not only those who already agree with you
- All parties must agree on facts
  - Subject opinions to evidence
- They must criticize your explanations and generalizations constructively
  - The force of an argument does not equal the loudness with which it is expressed,
  - nor the number of times it is repeated.
- They must even criticize their own explanations and generalizations.
  - And you too.
- Facts, generalizations and explanations are value-free
  - Even if they are unpleasant or unethical.

# Design theories and problem theories

- A **scientific design theory** is a scientific theory that there is a pattern in the interaction between an artifact and its context
- Examples:
  - *Theory of the UML in software engineering projects*
  - *Theory of your design in the intended problem context*

- A **scientific problem theory** is a scientific theory that there is a pattern in phenomena in a problem domain
- Examples:
  - *A theory of causes of large SE project failure*

# The structure of scientific theories

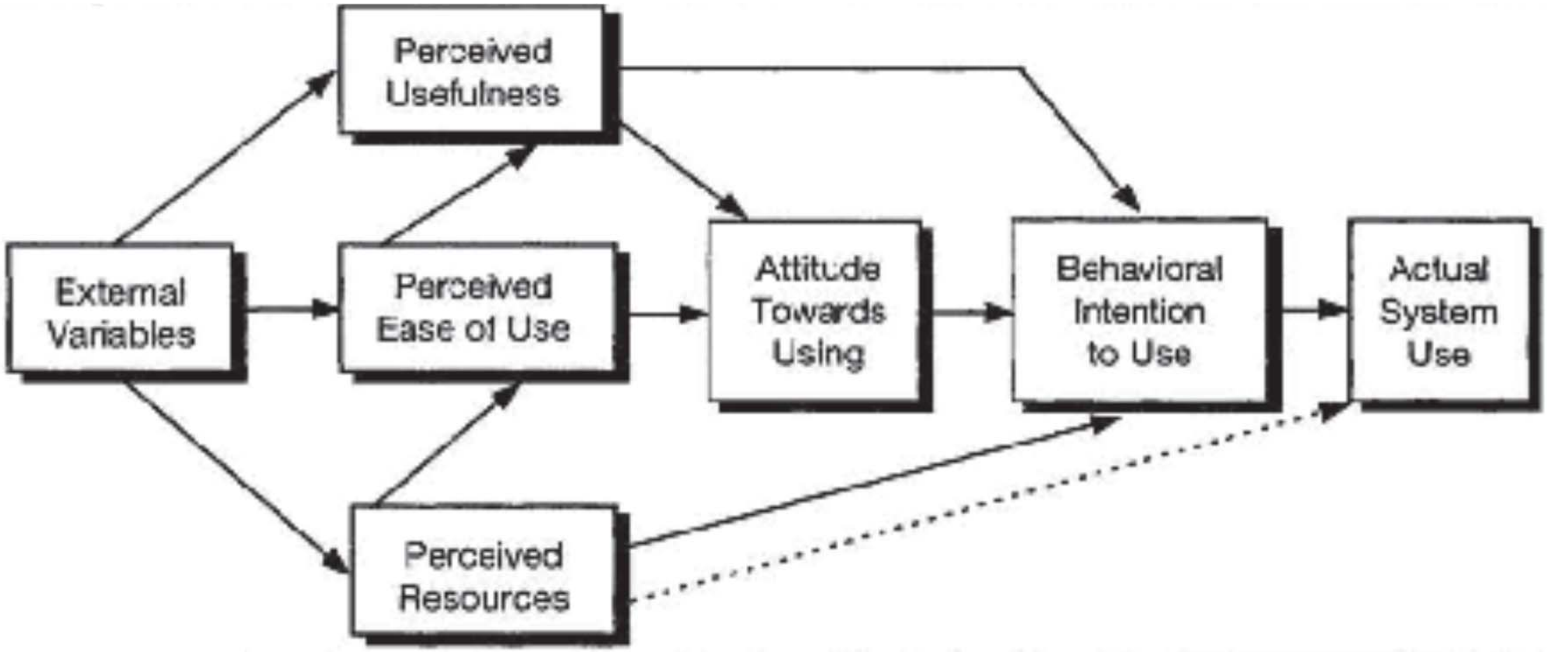
1. **Conceptual framework**
   – Definitions of concepts.
2. **Generalizations**
   – Express beliefs about patterns in phenomena.

# Theory of electromagnetism

- *Conceptual framework (concepts defined to describe and explain the relevant phenomena):*
  - *Definitions of electric current, electric charge, potential difference, electric resistance, electric power, capacitance, electric field, magnetic field, magnetic flux density, inductance, …, … and their units.*
- *Generalizations*
  - *Electric charges attract or repel one another with a force inversely proportional to the square of their distance.*
  - *Magnetic poles attract or repel one another in a similar way and always come in North-South pairs.*
  - *An electric current inside a wire creates a corresponding circular magnetic field outside the wire.*
  - *A current is induced in a loop of wire when it is moved towards or away from a magnetic field*

# Technology Acceptance Model

- *Conceptual framework*
  - *Definitions of perceived usefulness, perceived ease of use, perceived resources, attitude towards using, behavior intention to use, actual system use*
- *Generalization*




- K. Mathieson, E. Peacock, W. W. Chin - Extending theTechnology Acceptance Model: The Influence of Perceived User Resources. SIGMIS Database, 2001.

# Two more examples

- *Descriptive UML theory*
  - *Concepts: UML concepts, definitions of software project, of software error, project effort.*
  - *Descriptive generalization: (UML) X (SE project) → (Less errors, less effort than similar non-UML projects)*

- *Explanatory UML theory:*
  - *Concepts: definition of concept of domain, understandability*
  - *Explanatory generalization:*
    - *UML models resemble the domain more than other kinds of models;*
    - *they are easier to understand for software engineers;*
    - *So they they make less errors and there is less rework (implying less effort).*

# Even more examples

- *Theory of cognitive dissonance:*
  - *people tend to mutually adjust facts, beliefs and intentions to reduce stress*
- *The Balance theorem in social networks:*
  - *Social networks tend to decompose into two giant groups who like themselves and hate each other*
- *Theories X, Y, Z, and W of (project) management*
  - *Productivity increases by scientific management (X), stimulating creativity (Y), creating shared culture (Z), creating win-win (W).*
- *The theory that agile development delivers software faster than waterfall development*

# The use of theories in the design cycle

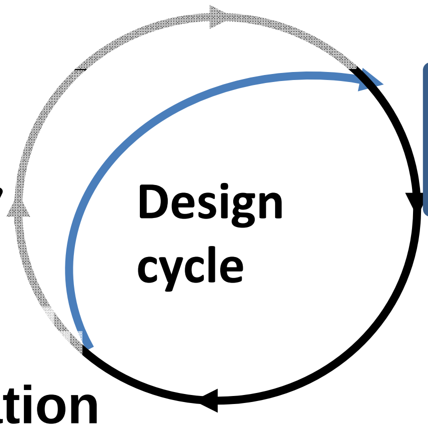
**Treatment implementation**

**Implementation evaluation = Problem investigation**

- Stakeholders? Goals?
- Conceptual problem framework?
- Phenomena? Causes, mechanisms, reasons?
- Effects? Positive/negative goal contribution?

**Problem theory (a.k.a. diagnosis)**

**Design theory**




**Treatment validation**

- Context & Artifact → Effects?
- Effects satisfy Requirements?
- Trade-offs for different artifacts?
- Sensitivity for different Contexts?

**Treatment design**

- Specify requirements!
- Requirements contribute to goals?
- Available treatments?
- Design new ones!

# Scaling up of design theories

- A design theory
  - may start out as a hopeful belief,
  - become a hypothesis supported by an argument
  - evolve as an initial theory supported by laboratory validations
  - and end up as generally accepted theory evaluated in the field

# Theories are fallible

- Fallibilism: All theories may be wrong!
  - Outside mathematics there is no certainty
  - And even there, mathematicians make mistakes (Lakatos' *Proofs and Refutations*)
- No theory can be proven correct
- All theories are improvable

# Falsificationism

- Introduced by the philosopher Karl Popper
  - ~~Mechanical version: If the prediction of a theory is contradicted by observation, then reject the theory.~~
  - Sophisticated version: If the prediction of a theory is contradicted by observation, then
    - try to replicate the contradicting phenomenon,
    - try to understand why it happens,
    - try to improve the theory so that it can deal with this phenomenon,
    - describe whatever comes out of this and submit to critical peer review,
    - shelve this as a problem to be solved later.

# The result of your technical research is a design theory

- Your research results:
  - An artifact design,
  - A generalization about the effects of placing this design in a context that satisfies some assumptions.
- The generalization is fallible and you must provide as much evidence as possible for it, and indicate the limits of this evidence
- All your prototypes will probably get lost, or will be changed

# Poster (3)
# 5 minutes

- Which theory about (your artifact) x (context) do you hope/have you shown to be true?
  - Descriptions, explanations provided by the theory?
- What evidence do you have, and what do you still intend to produce?

# Outline

1. What is design science
2. Research goals and problems
3. The design and engineering cycles
4. Theories
   – Conceptual frameworks
   – Generalizations
5. **Scientific inference**
6. Research design

# Outline

1. What is design science
   – Research goals and problems
   – The design cycle
2. Theories
   – Conceptual frameworks
   – Generalizations
3. Empirical research
   – **Scientific inference**
   – Research design

# From design to empirical research

**Improvement design**

Problems, artifacts investigate →

← Knowledge, design problems

**Answering knowledge questions**

The design of useful artifacts

Metaphor: the craftsman

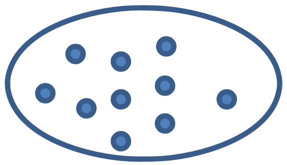
The design of sound arguments

Metaphor: the lawyer

# How to get from your measurements to a theory?

**Facts**

Observed sample of cases

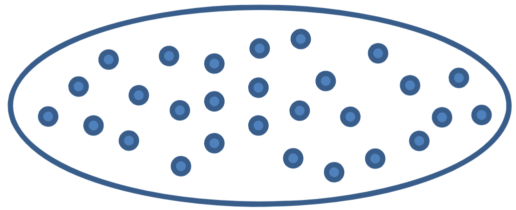


- What happens in these **cases**?
- What average, variance in this **sample**?

Generalize

**Descriptive general theory**

Unobserved population



- What happens in all cases?
- What average, variance in this population?

Explain

- Why?

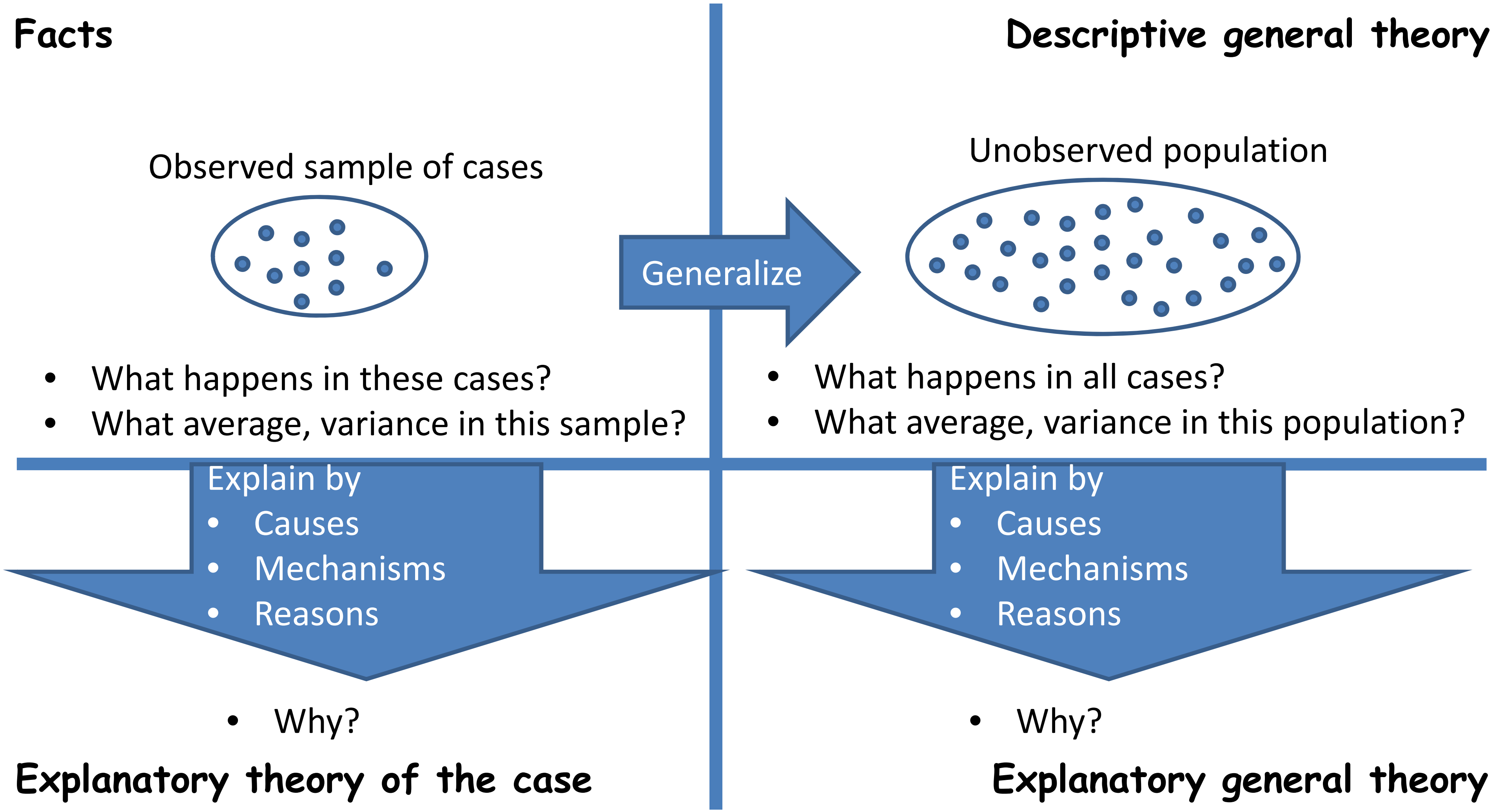
**Explanatory theory of the case/sample**

Explain

- Why?

**Explanatory general theory**

# Three kinds of explanation



# Example

- *Descriptive question: Is the light on?*
  - *Based on observation: Yes.*
  - *When? Now.*
  - *Where? Here.*
- *Explanatory question: Why is it on?*
  1. ***Cause:*** *because someone turned the light switch, it is on (and not off).* Explains difference with off-state.
  2. *Why does this cause the light to switch on?* ***Mechanism:*** *because the switch and light bulbs are connected by wires to an electricity source, in this architecture …, and these components have these capabilities …..* Explains how on-state is produced.
  3. *By why did someone turn the light on?* ***Reasons:*** *Because we wanted sufficient light to be able to read, and it was too dark to read.* Explains which stakeholder goal is contributed to.

# Another example: software

- *Descriptive question: What is the performance of this program?*
  - *Execution time for different classes of inputs?*
  - *Memory usage?*
  - *Accuracy?*
  - *Etc. etc.*
- *Explanatory question: Why does this program have this performance (compared to others)?*
  1. ***Cause:*** *Variation in execution time is caused by variation in input; etc.*
  2. ***Mechanism:*** *Execution time varies this way because it has this architecture with these components*
  3. ***Reasons:*** *Observed execution time varies this way because users want to be on-line all the time, and therefore provide these inputs*

# Another example: method

- *Descriptive question: What is the performance of this method for developing software?*
  - *Understandability for practioners*
  - *Learnability*
  - *Quality of the result*
  - *Perceived utility*
  - *Etc. etc.*
- *Explanatory question: Why does this method have this performance?*
  1. ***Cause:** Difference in understanding of methods by software engineers is attributed to differences in the methods.*
  2. ***Mechanism:** These differences are explained by the structure of the method and/or the structure of cognition.*
  3. ***Reasons:** Developers use this method because it is currently a hype among developers*

# Keywords in the three kinds of explanations

- *Descriptive question: What is happening?*
  - What, How much, How often, When? Where? What components, Who involved, etc. <u>Facts.</u>
- *Explanatory question: Why did this happen?*
  1. ***Cause:*** *effect <u>attributed</u> to a cause. Explain difference in outcomes by difference in interventions.*
  2. ***Mechanism:*** *Outcome <u>produced</u> by interaction among components. Explain capability of system in terms of capabilities of components.*
  3. ***Reasons:*** *Outcome <u>contributes</u> to a goal. Explain outcome in terms of rational takeholder choices.*

# One more example

- **Causal explanation:** effect <u>attributed</u> to a cause. Explain difference in outcomes by difference in interventions. Causation is difference-making.
  - *The coffee made me stay awake late.*
- **Architectural explanation:** Outcome <u>produced</u> by interaction among components. Explain capability of system in terms of capabilities of components
  - *Caffeine has a psychostimulant effect because it antagonizes adenosine, which normally inhibits neurotransmitters such as dopamine.*
- **Rational explanation:** Outcome <u>contributes</u> to a goal. Explain outcome in terms of rational takeholder choices.
  - *I worked late because I wanted to finish the paper before the deadline.*

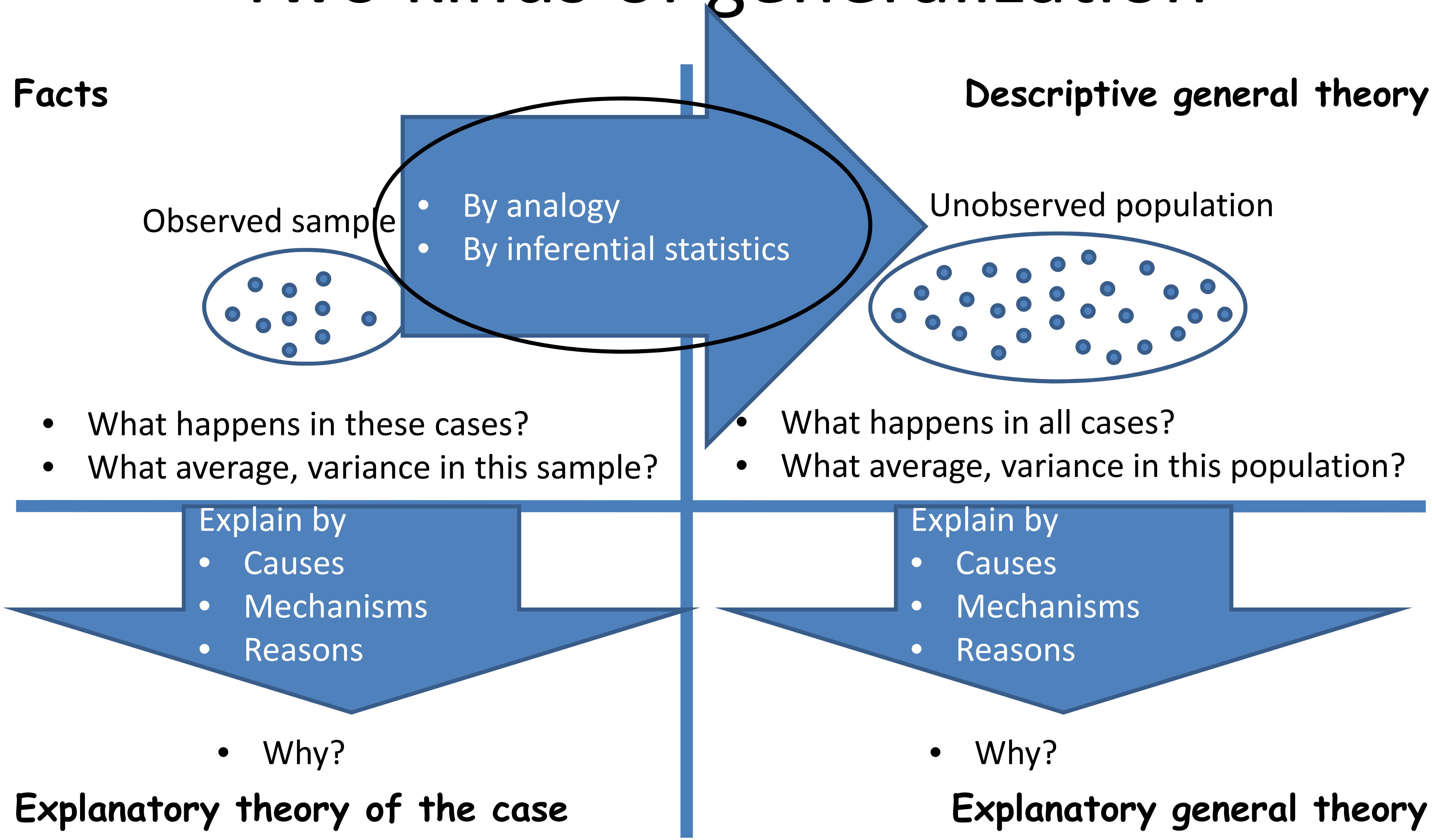
# Internal validity

- Degree of support for an explanation
- Threats that decrease support:
  - Outcome of an exceriment may have many causes
    - Which one is most plausible?
    - Which ones can and cannot be ruled out?
  - Effect of a cause may be produced by various mechanisms
    - Which components played a role, and which did not?
    - How did they interact? How do we know?
  - An action may have many reasons
    - Which ones were operative?
    - What evidence do we have for it?

# Summary of explanations

- Causal explanation:
  - Event Y happened because earlier, event X happened.
  - A difference in X **makes a difference to** Y
- Architectural explanation:
  - System S has an architecture with components C1 … Cn that have some capabilities to interact with each other
  - When stimulus s occurs, response r is produced by an interaction among C1 … Cn, called a **mechanism**
- Rational explanation:
  - Actor A performs action a because it has goal G.
  - G is the **reason** that A does a

# Two kinds of generalization



# Generalization by analogy Example 1

- *Observation:*
  - *Artifact: This prototype implementation of the MUSIC algorithm,*
  - *Context: when interacting with a simulation of an antenna array receiving plane waves in the presence of only white noise, running on a Montium 2 processor,*
  - *Effect: has execution speed less than 7.2 ms and accuracy of at least 1 degree.*
- *Generalization by analogy:*
  - *All **similar** implementations*
  - *Running in **simila**r contexts*
  - *Will show **similar** performance*

Descriptive generalization. Implicit assumptions:

1. The mechanisms that explain this performance will be present in all similar artifacts and contexts, and
2. Will not be undone by other mechanisms.

- Analogic generalization  must be based on architectural explanations

- In all architecturally similar situations, similar mechanisms will lead to similar phenomena

- Assumptions:
  1. The mechanisms that explain the phenomena will be present in all architecturally similar situations, and
  2. will not be undone by other mechanisms.

# Example of an unsound analogic generalization

- Wallnuts look like brains.
- Brains can think.
- Therefore .... Wallnuts can think

- This is only superficial, feature-based similarity
- There is no mechanism that produces thinking in brains and wallnuts!

- If it walks and talks like a duck, it may be John Cleese!

# Generalization by analogy example 2

- *Observations:*
  - *Artifact: this version of the UML*
  - *Context: Used in this software project*
  - *Effect: Produces software with less errors and less effort than in similar projects without the UML,*
  - *Explanation: UML models  are easier to understand for software engineers because they resemble the domain more than other kinds of models, and*
  - *So the software engineers make less errors and there is less rework.*
- *Generalization*
  - *In similar projects, UML will have similar effects by the same mechanisms,*
  - *Unless there are other mechanisms that undo the UML-effect*

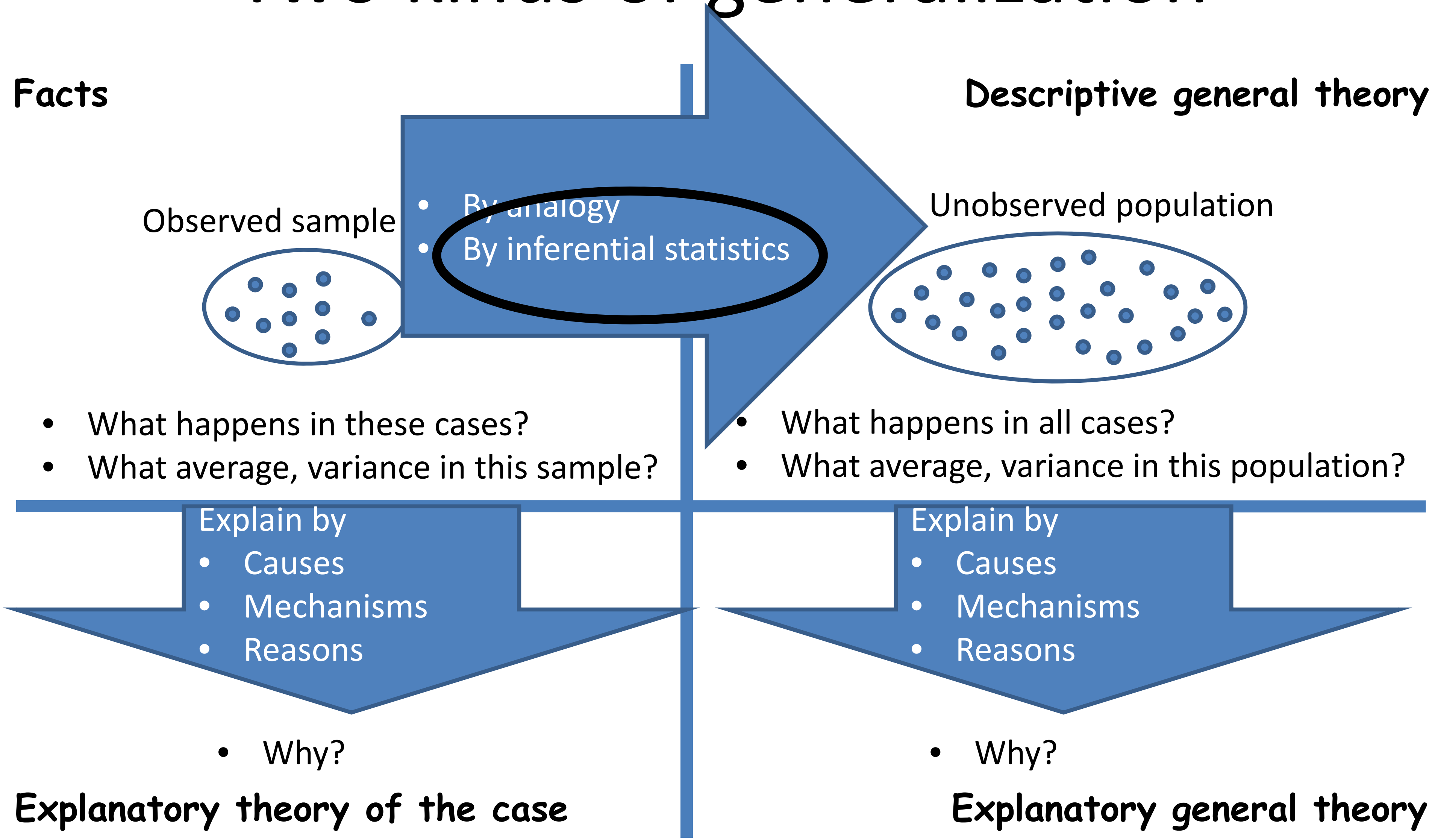
Descriptive and explanatory generalization. Assumptions:

1. The mechanisms that explain this performance will be present in all uses of UML in software projects, and
2. Will not be undone by other mechanisms.

# External validity

- Degree of support for generalization by analogy
- Support increases when there is previously established theory that explains phenomena in terms of architecture.
- Threats that decrease support:
  - Cases that look superficially similar may be **architecturally different**.
  - Analogic generalization is not universal: it may be falsified by **interfering mechanisms**.
- Mitigate this by analytic induction: Study cases one by one, update theory in between
  - Start with an initial theory about how mechanisms produce phenomena
  - Update the theory after each case
  - Look for confirming as well as falsifying cases

# Two kinds of generalization



# Statistical inference

- Define theoretical population
- Construct a sampling frame: list of study population
- Randomly select a sample from the study population
- Collect sample statistics
- Conclude something by statistical inference about the study population (fallible conclusion)

# Generalization by statistical inference Example 1

- T. Huynh, J. Miller, An empirical investigation into open source web applications' implementation vulnerabilities. Empir. Softw. Eng. **15**(5), 556–576 (2010)

- *Theoretical population: Open source web applications*
- *Study population: not reported.*
- *Sample of 20 open source web applications*
- *Sample statistic: The average percentage of vulnerabilities caused by coding errors (rather than by design flaws or configuration errors) per OS web application* ***in the sample*** *is 73%.*
- *Statistical inference:*
  - *with roughly 95% confidence, the average percentage of vulnerabilities caused by coding errors* ***in the study population*** *is roughly 73% ± 4%*

# Example 1: statistical conclusion validity

- *Unstated assumptions of statistical inference:*
  - *Assuming a random sample from the study population, and*
  - *assuming that the proportion of coding errors is constant and independent across web applications,*
  - *with roughly 95% confidence, the average percentage of vulnerabilities caused by coding errors* ***in the study population*** *is roughly 73% ± 4%*

- *This means that 95% of the times we estimate a confidence interval this way, the conclusion is correct*

# Example 1: external validity

- *Assuming that the mechanisms by which implementtion vulnerabilities introduced in the study population are the same in the entire theoretical population,*

- *we infer by analogy that*

- *with roughly 95% confidence, the average percentage of vulnerabilities caused by coding errors* ***in the study population*** *is roughly 73% ± 4%*

# Statistical inference combined with causal inference

- Define theoretical population
- Construct a sampling frame: list of study population
- Randomly select a sample from the study population
- **Allocate two treatments randomly to sample elements**
- **Apply the treatments**
- Collect sample average and variance of the two treatment groups
- Conclude by statistical inference whether a differences exists in the average of the two treated study populations (fallible conclusion validity)
- **If it exists, attribute this to the difference in treatments (fallible internal validity)**
- Generalize by analogy to the theoretical population (fallible external validity)

# Generalization by statistical inference

- Hypothetical example:

- *Four groups of 9 to 26 students made UML domain model from Use case model for two systems, with or without using System Sequence Diagrams (SSDs) and System operations contracts (SOCs). Four-group crossover design.*
- *Observation:*
  - ***In the observed samples**, when SSDs and SOCs were used, average correctness of models was higher, and effort to produce them was lower.*
- *Generalization by statistical inference:*
  - *Pairwise t-test, simple repeated measures ANOVA and mixed repeated measures ANOVA support the generalization that average correctness of models and effort to produce them is better when SSDs and SOCs are used **in the population** of all software engineering students. This conclusion is plausible but not always correct.*
- *Explanation:*
  - *By listing all possible causes, and assessing them on their plausibility, the use of SSDs and SOCs is the most plausible cause of these effects (and not the competence of the students or the positive expectation of the experiments, or …)*

# Example continued

- *We may want to* ***generalize by analogy*** *to similar populations, e.g. the population of professional software engineers.*
  - *Need to discuss if the social or cognitive mechanisms that produce the results in the student population, are the same as those in the population of professional software engineers.*

- NB the setup of the experiment resembles the classical Randomized Controlled Trial used to validate the effect new drugs

# An aside

- L. Briand, Y. Labiche, R, Mardazo-Rivera. “An experimental evaluation of the impact of systems sequence diagrams and system operation contracts on the quality of the domain model”. *ESEM 2011,* Page 157-166. ACM Press.
- They did this ….. but unfortunately found hardly any support for a statistically significant difference.

# Statistical conclusion validity

- Degree of support for a statistical inference
- Threats:
  – The study population may be undefined
  – Sampling may not be random
  – Assumptions of statistical techniques may not be satisfied

- NB
  – All models are wrong! They are abstractions.

# Big data

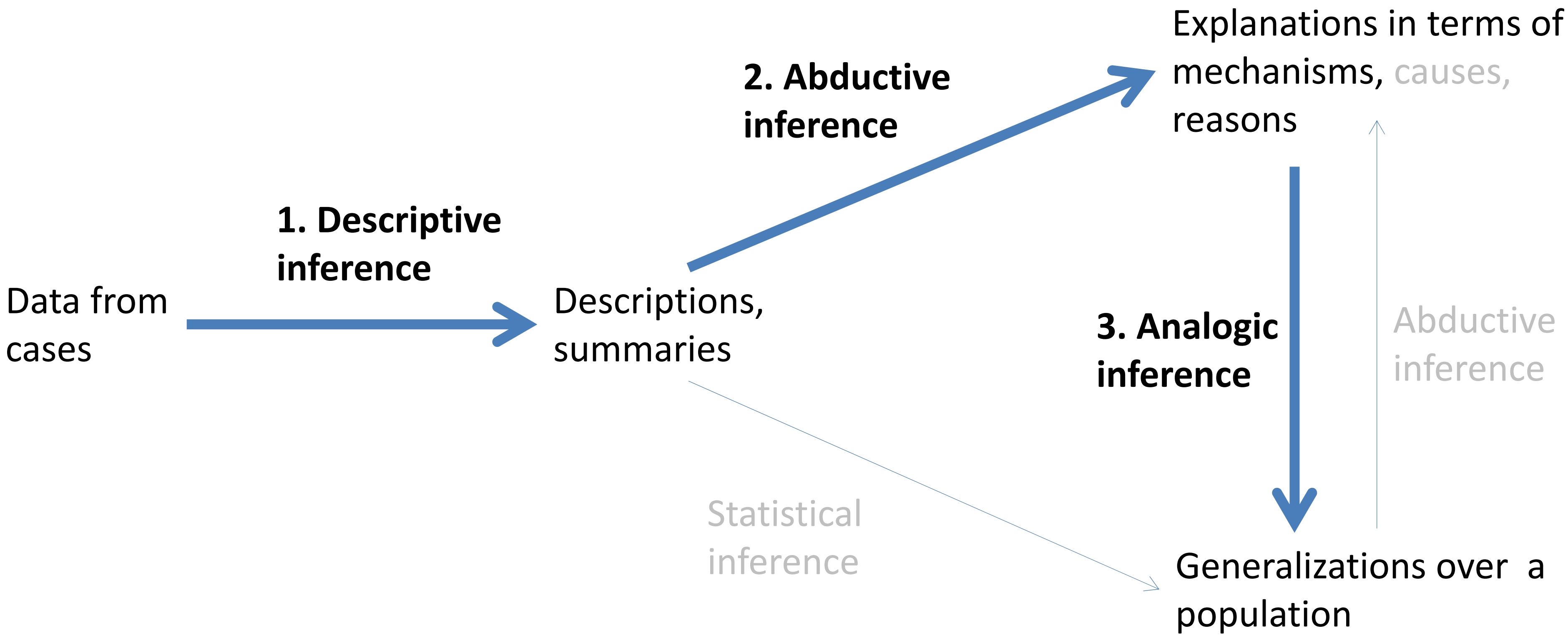
- If the sample equals the study population, statistical inference can be skipped.
  - descriptive statistics
  - statistical learning (e.g. regression or classification)
- Still need to argue external validity of generalization to theoretical population

- *Example*
  - *Based on an analysis of data about 90% of the Dutch male population, you compute an average height of 1m75.*
  - *However, your sample excluded all males taller than 2m.*
  - *The real average is 1m83*

# Summary of scientific inferences

Remember: we are constructing arguments ... You are a lawyer defending your case

# Case-based inference



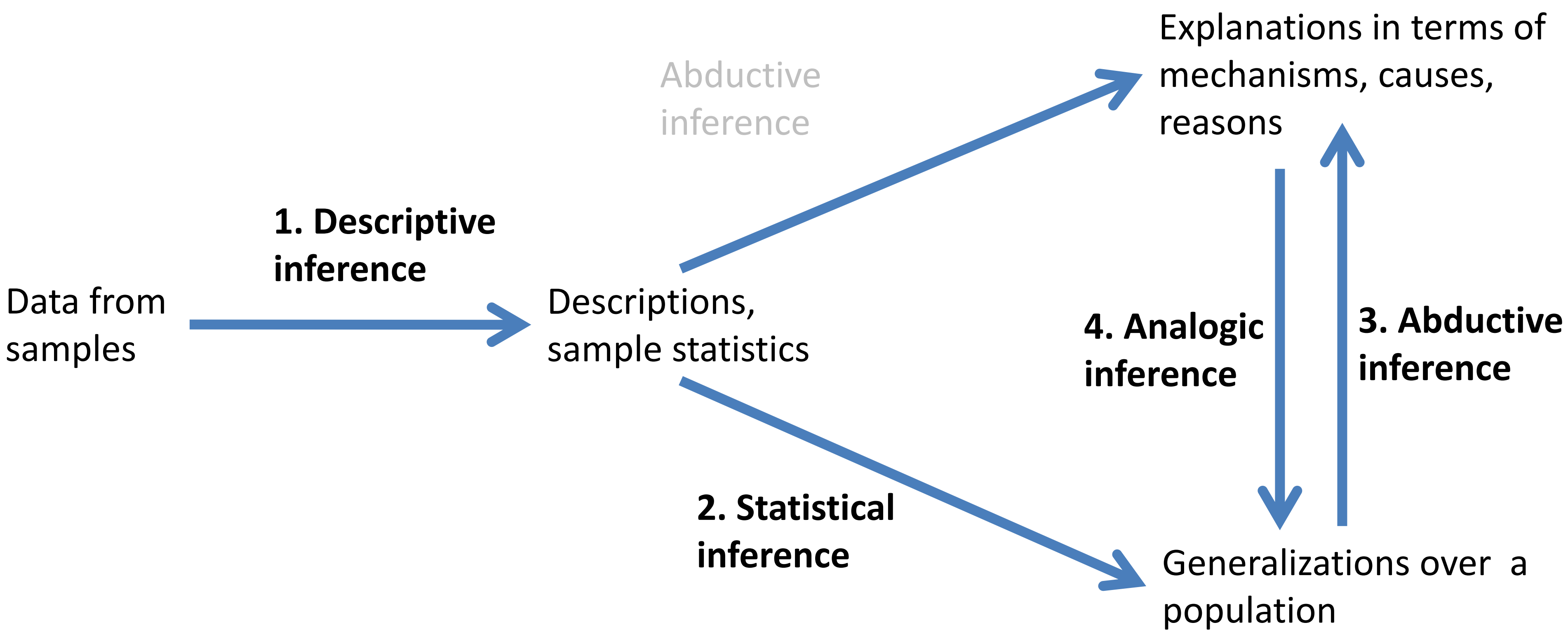

- Analogic inference to similar cases/samples/populations must be based on architectural explanations (in terms of mechanisms or reasons)

# Analytic induction

- Study cases one by one

- Start with an initial theory about how mechanisms produce phenomena
- Select cases expected to confirm or falsify the theory
- If data falsify your expectation,
  - update your conceptual framework to define the falsification away, or
  - update your generalization to explain all cases studied so far

# Sample-based inference




- Abduction:
  - Explanation of effects in terms of causes
  - Explanation of causes in terms of mechanisms or reasons

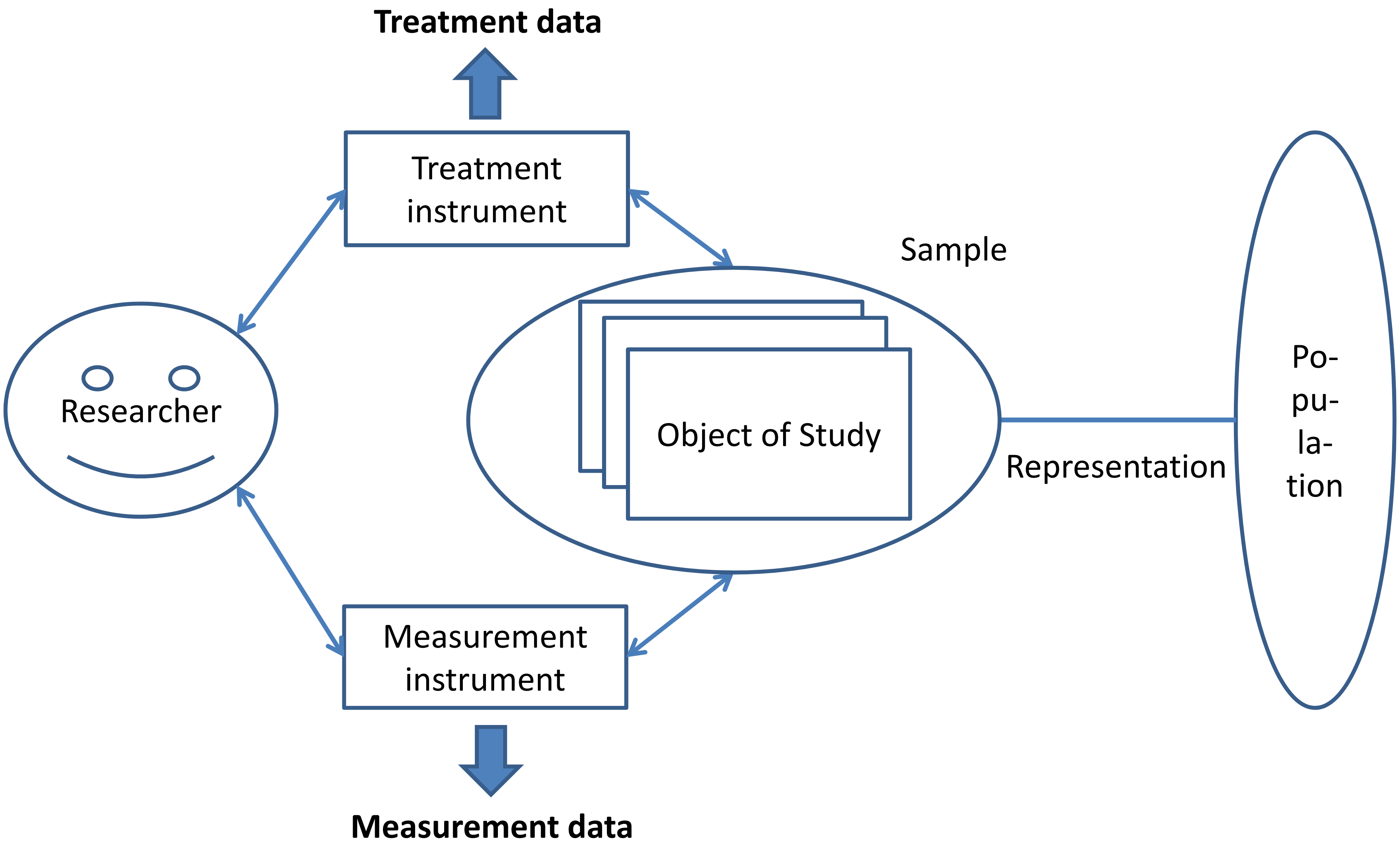
# Outline

1. What is design science
   – Research goals and problems
   – The design cycle
2. Theories
   – Conceptual frameworks
   – Generalizations
3. Empirical research
   – Scientific inference
   – **Research design**

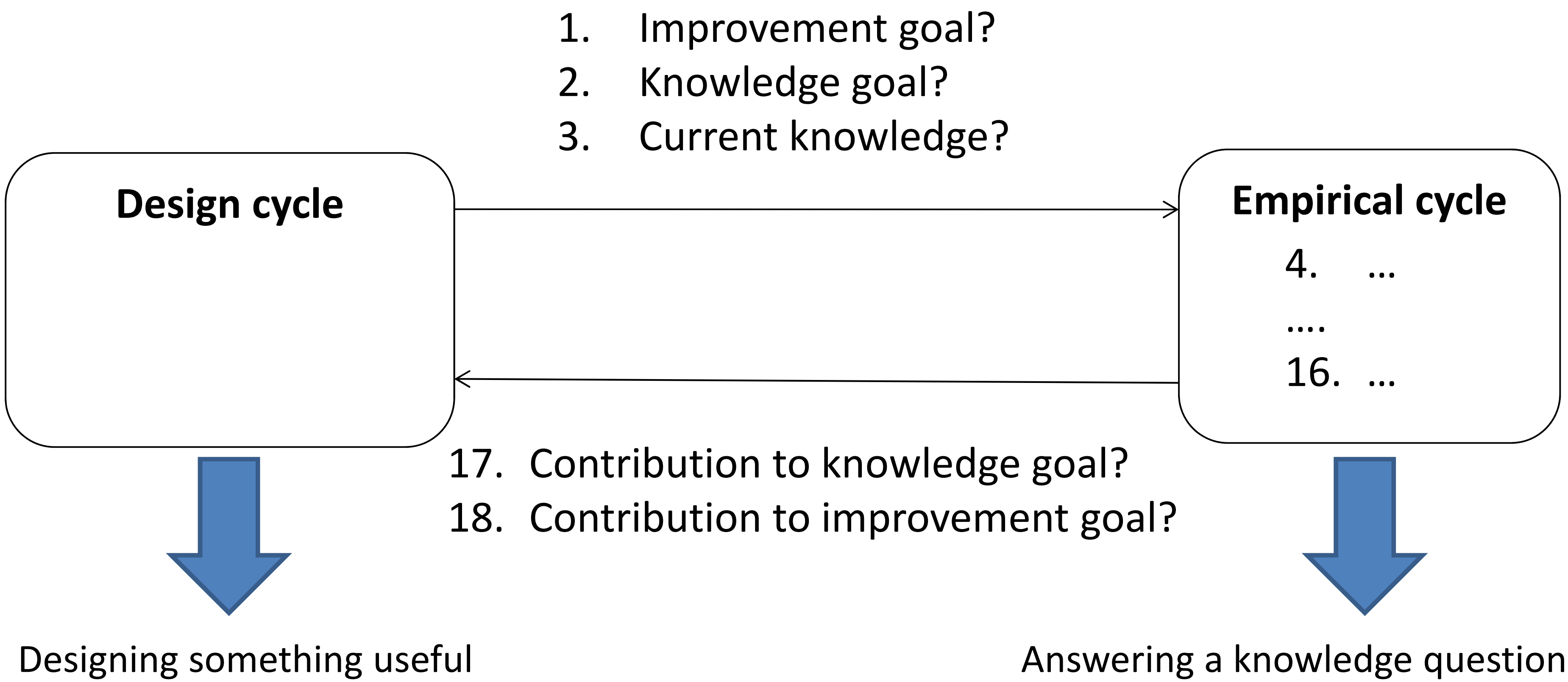
# Research designs

| | Observational study (no treatment) | Experimental study (treatment) |
|---|---|---|
| **Case-based:** investigate single cases, look at architecture and mechanisms | **Observational case study** | • **Expert opinion** (mental simulation by experts),<br>• **Mechanism experiments** (simulations, prototyping),<br>• **Technical action research** (experimental use of the artifact in the real world) |
| **Sample-based:** investigate samples drawn from a population, look at averages and variation | **Survey** | • **Statistical difference-making experiment** (treatment group – control group experiments) |

# Research setup



# Checklist for the empirical cycle: context



**Data analysis**

12. Data?
13. Observations?
14. Explanations?
15. Generalizations?
16. Answers?

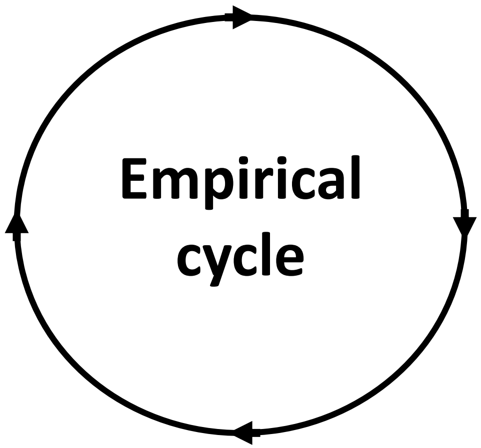
**This is a checklist for**

- ***research design,***
- ***research reporting,***
- ***reading a report.***

**App. B in my book & my web site**

**Research execution**

11. What happened?




**Research problem analysis**

4. Conceptual framework?
5. Knowledge questions?
6. Population?

**Design validation**

7. Object of study validity?
8. Treatment specification validity?
9. Measurement specification validity?
10. Inference validity?

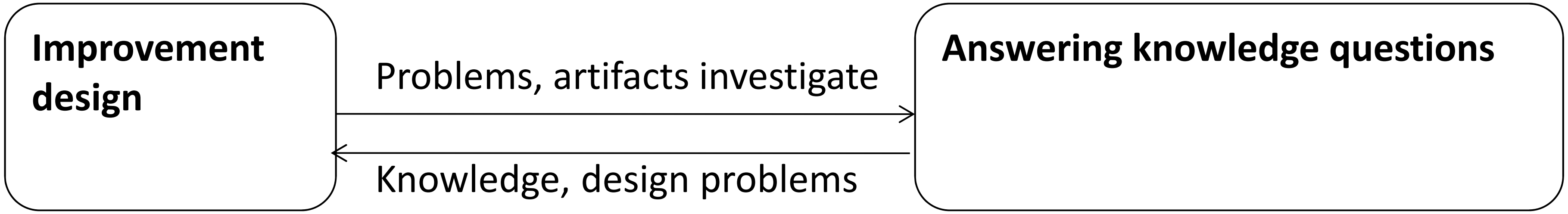
**Research & inference design**

7. Object of study?
8. Treatment specification?
9. Measurement specification?
10. Inference?

Research setup (items 7–9)

Inference (item 10)

# Summary




**Design research problems**
- Improve <problem context>
- by <treating it with a (re)designed artifact>
- such that <artifact requirements>
- in order to <stakeholder goals>.

**Design cycle**
- Problem investigation
- Treatment design
- Treatment validation

**Strategy**

Artifacts → Design cycle → Artifacts

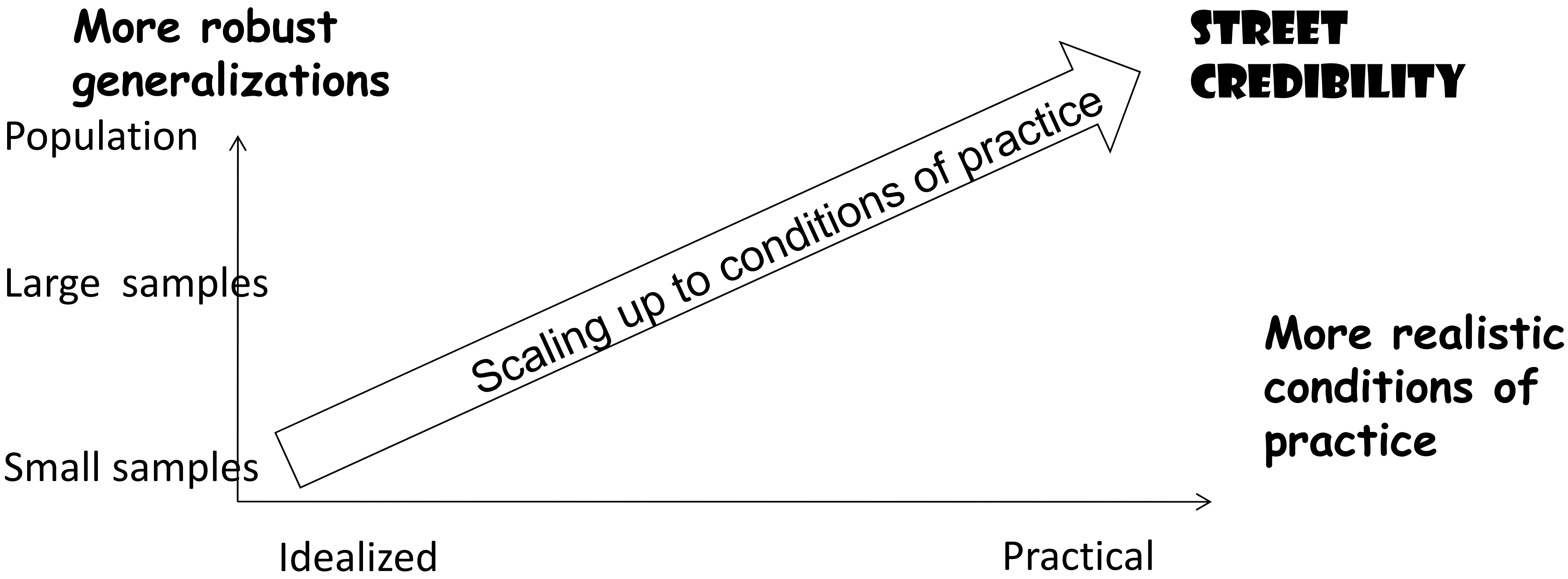
**Empirical knowledge questions**
- Descriptive: what, how, when, where, who, etc.→ **Facts**
- Explanatory: Why → **explanations**

**Empirical cycle**
- Research problem analysis
- Research design & validation
- Research execution
- Data analysis

**Strategy:**

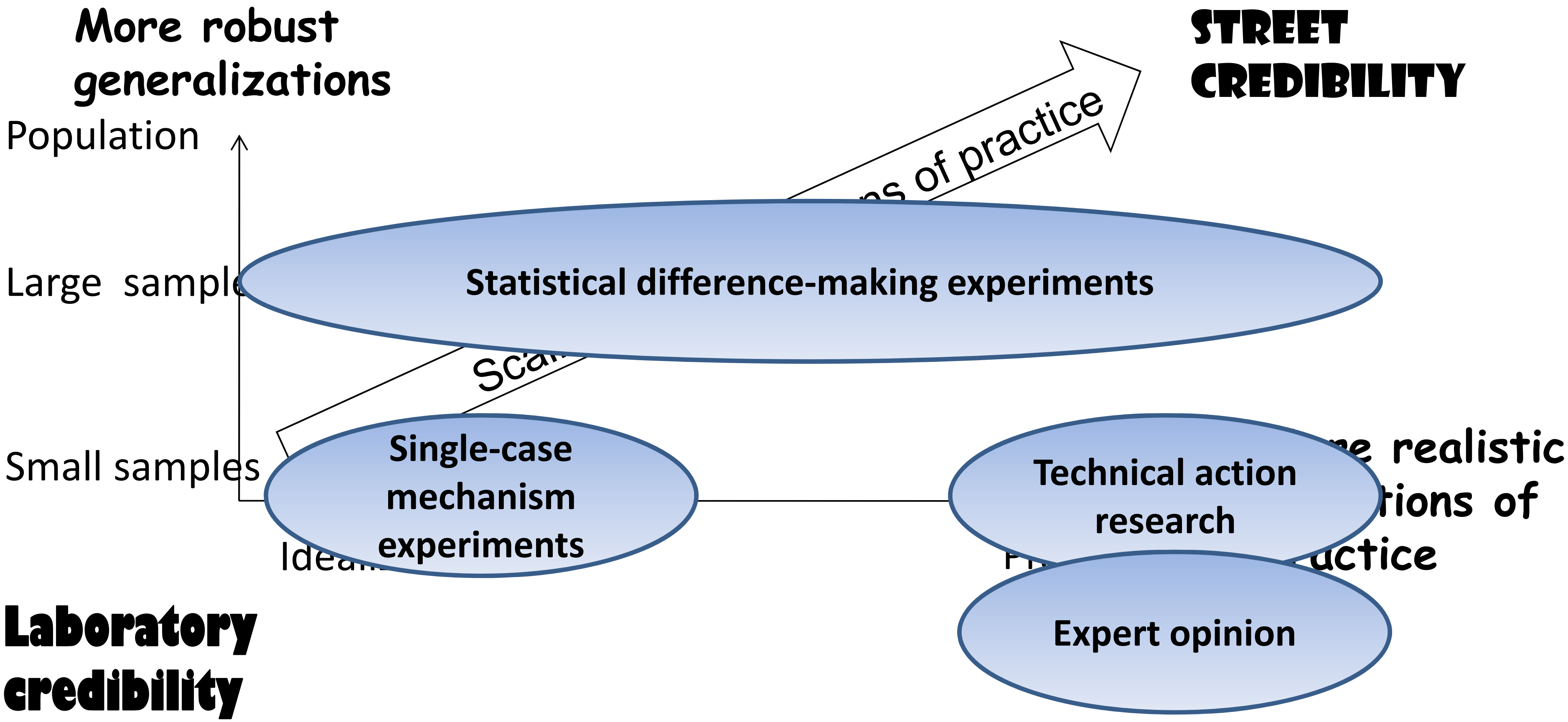
Theories → Empirical cycle → Theories

# Wnen to use these methods in design science research?




**Laboratory credibility**

- Just like New Drug Research




- Scaling up:
  - Single-case mechanism experiment (laboratory simulation)
  - Expert opinion
  - Single-case mechanism experiment (field simulation)
  - TAR (apply technique in a real-world project)

# Poster (4)

- Finish your poster with information about the kinds of empirical research that you intend to do

- Wieringa, R.J. and Daneva, M. (2015) *Six strategies for generalizing software engineering theories.* Science of computer programming, 101. pp. 136-152.

- Wieringa, R.J. (2014) *Design science methodology for information systems and software engineering.* Springer Verlag

- Wieringa, R.J. (2014) *Empirical research methods for technology validation: Scaling up to practice.* Journal of systems and software, 95. pp. 19-31.

- Wieringa, R.J. and Morali, A. (2012) *Technical Action Research as a Validation Method in Information Systems Design Science.* In: *Design Science Research in Information Systems. Advances in Theory and Practice 7th International Conference*, DESRIST 2012, 14-15 May 2012, Las Vegas, USA. pp. 220-238. Lecture Notes in Computer Science 7286. Springer.

- Wieringa, R.J. (2010) *Relevance and problem choice in design science.* In: *Global Perspectives on Design Science Research (DESRIST). 5th International Conference*, 4-5 June, 2010, St. Gallen. pp. 61-76. Lecture Notes in Computer Science 6105. Springer.

- Wieringa, R.J. (2009) *Design Science as Nested Problem Solving.* In*: Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology*, Philadelphia. pp. 1-12. ACM.