Design Science Research Methods

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

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**
- The problems in design science research are design problems
  - Goal is to design something useful
  - Research method is the design cycle
- The problems in empirical research are 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

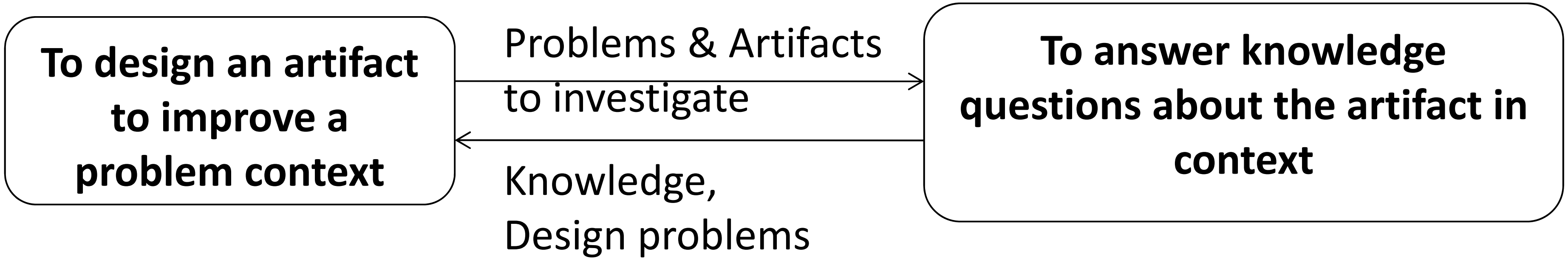
Outline

1. What is design science
   – Research goals and problems
   – The design and engineering cycles
2. Design theories
   – Scientific research design
   – Scientific theories
   – Scientific inference: from data to theories

What is design science?

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

Two kinds of research problems in design science



Design software to estimate Direction of Arrival of plane waves, to be used in satelite TV receivers in cars	• *Is the DoA estimation accurate enough in this context?* • *Is it fast enough?*
Design a Multi-Agent Route Planning system to be used for aircraft taxi route planning	• *Is this routing algorithm deadlock-free on airports?* • *How much delay does it produce?*
Design a data location regulation auditing method	• *Is the method usable and useful for consultants?*

Is the artifact **useful** in this context?

Is the answer about the artifact in context **true**?

Reality check

- What is/are the artifacts and what is/are the context(s)?
  - 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

Conclusions

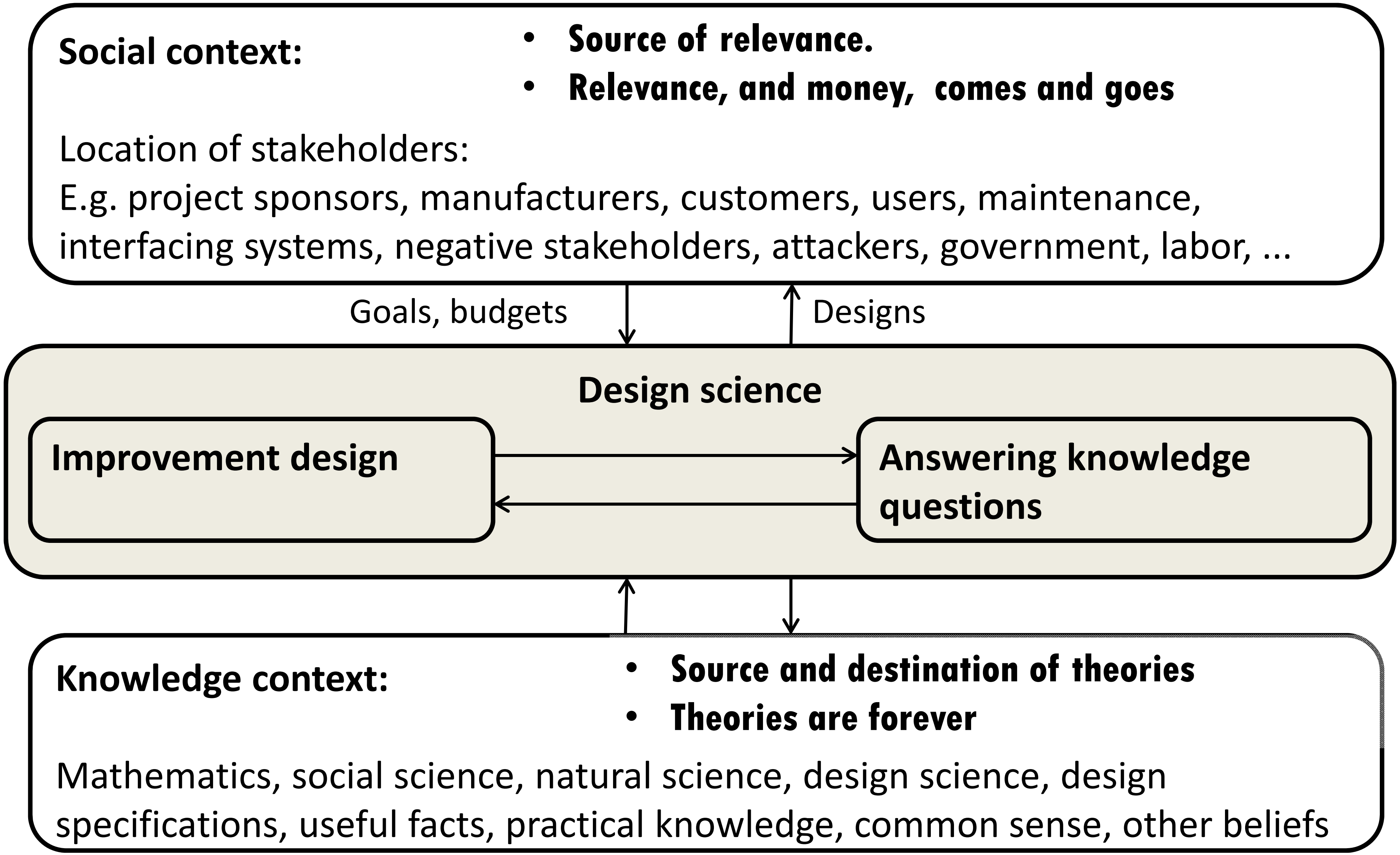
- Is the distinction between knowledge-driven and design-driven research clear?
- The title of your thesis is the shortest summary of your research project.
  - The best titles mention the artifact and the context.

Exercise:
Ingredients for your thesis title

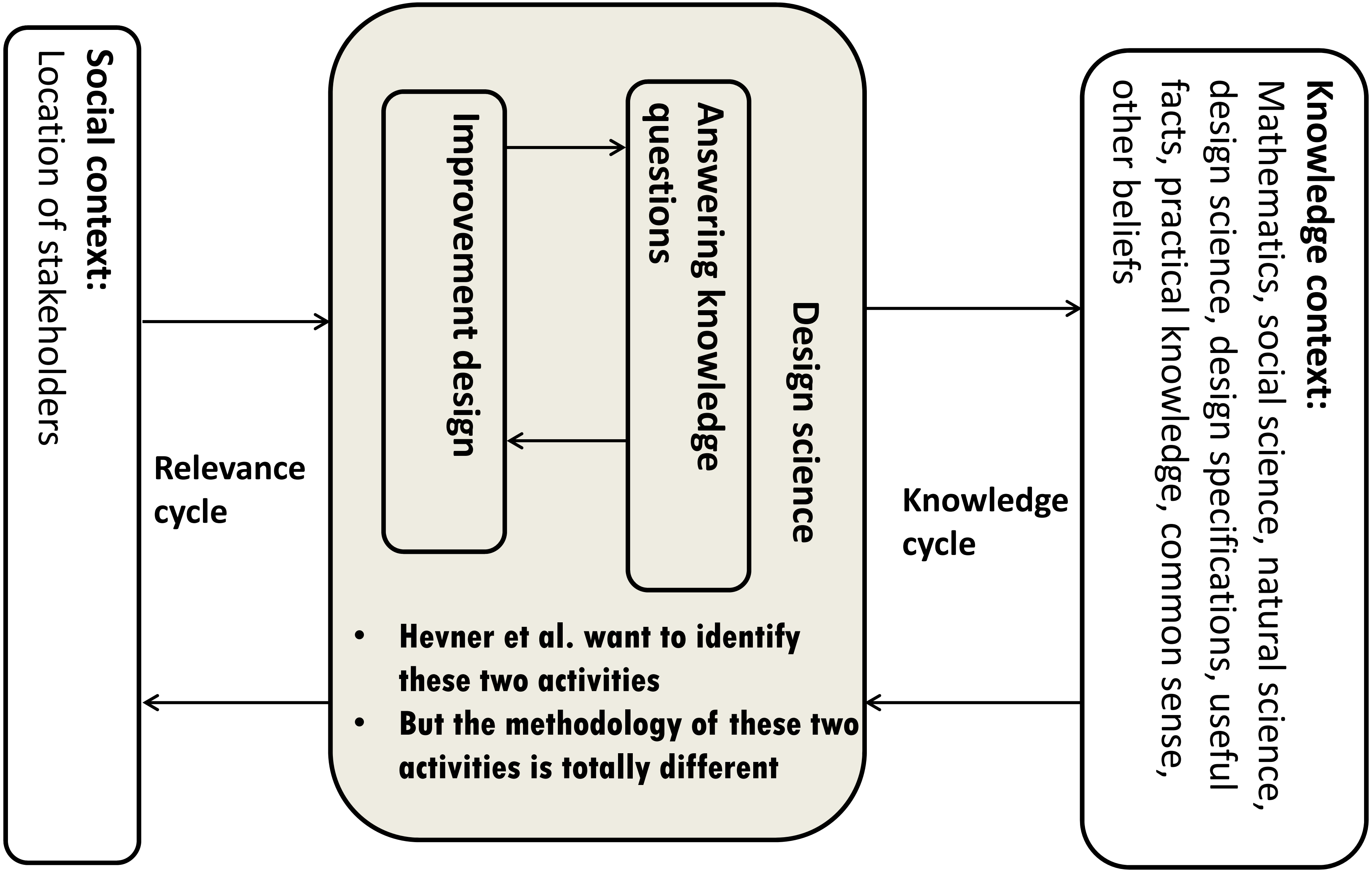
- What research problem(s) are you investigating?
  - Artifact and context

Framework for design science

- Stakeholders may not know they are stakeholders



(Dis)similarity to Hevner et al. framework



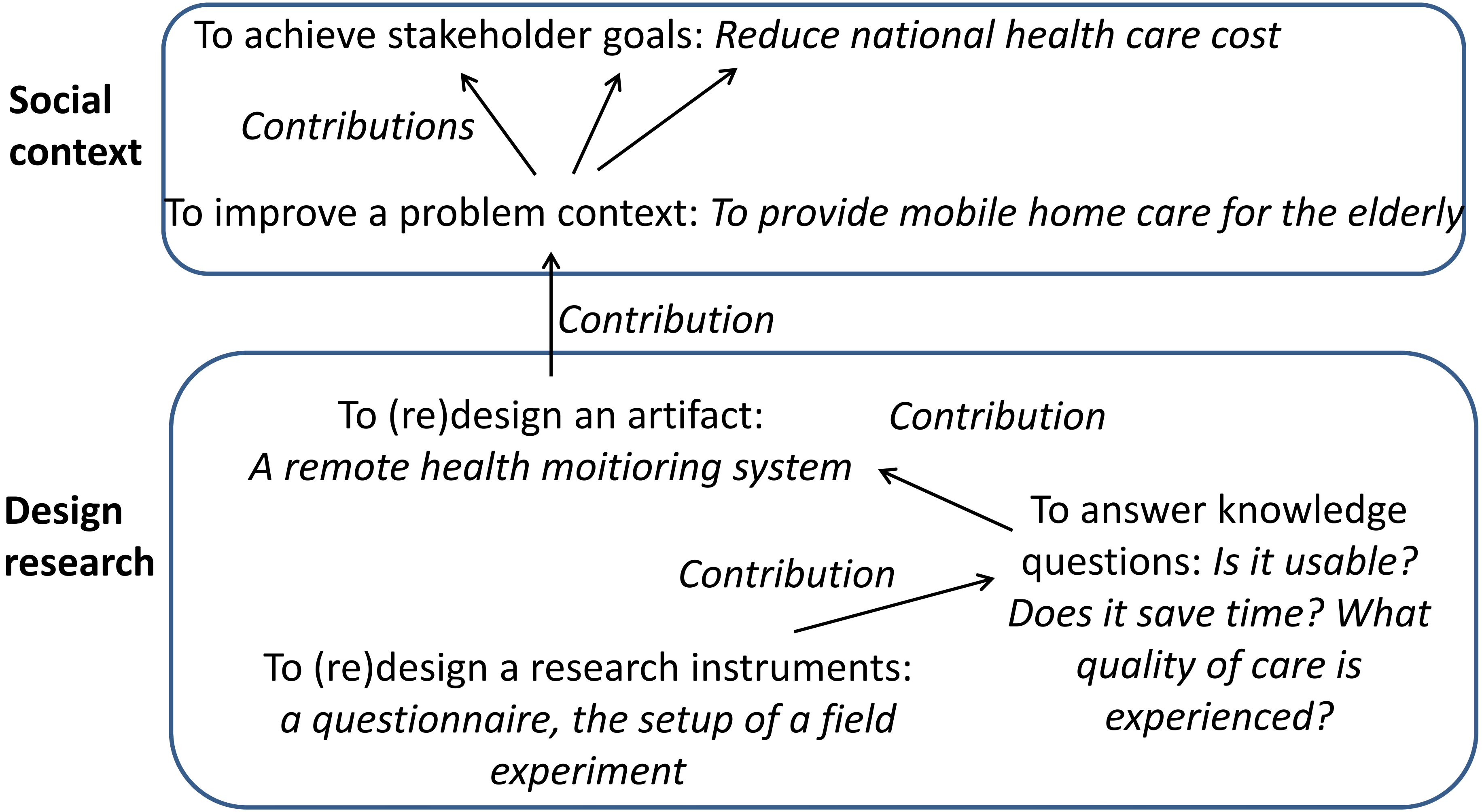
Exercise: Material for your elevator pitch

1. What design(s) will be delivered by your project?
   - What is new?
2. Who are the stakeholders of your project?
   - What are their goals?
3. What knowledge will be produced by your project?
   - What is new?

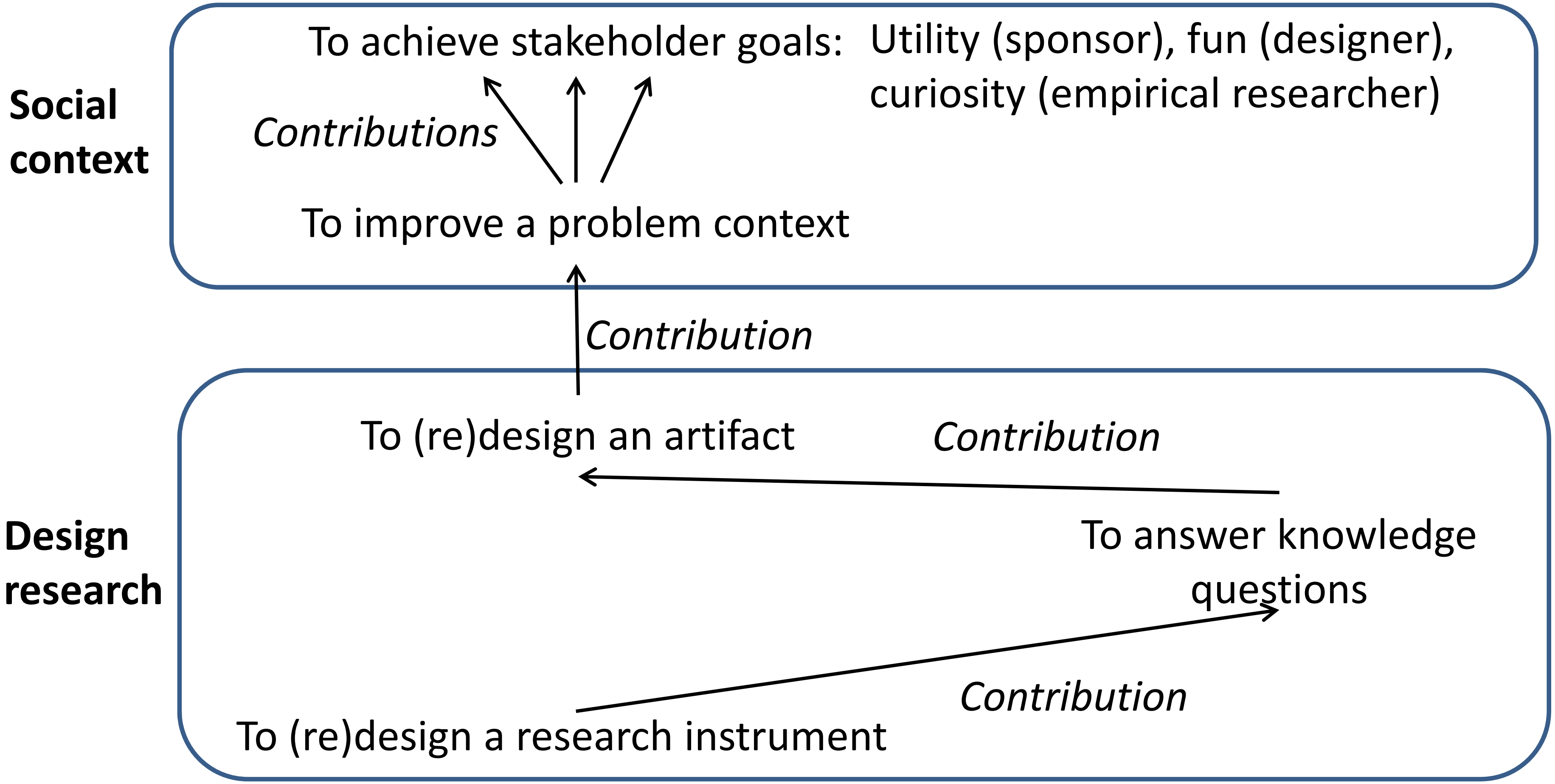
Outline

1. What is design science
   - **Research goals and problems**
   - The design and engineering cycles
2. Design theories
   - Scientific research design
   - Scientific theories
   - Scientific inference: from data to theories

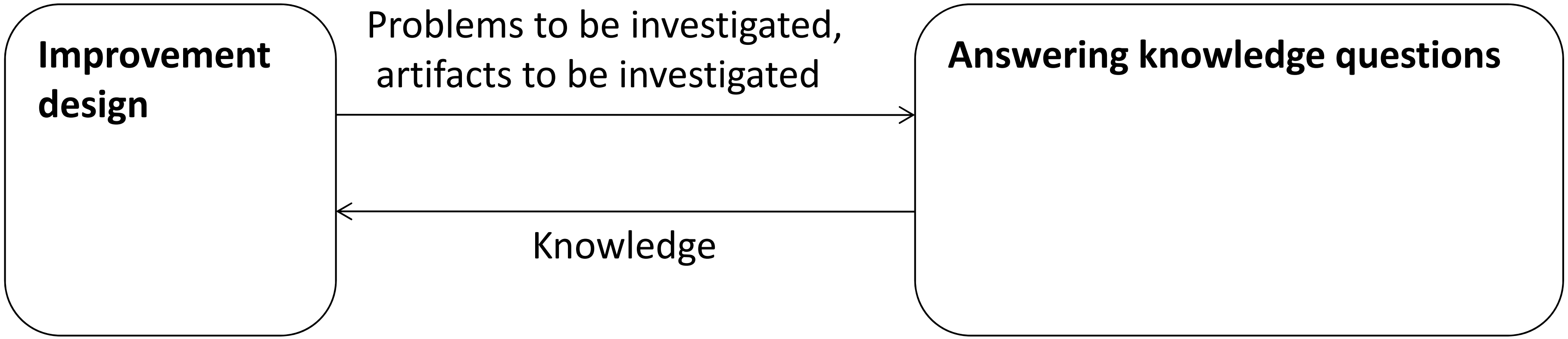
Goal structure: example



Goal structure

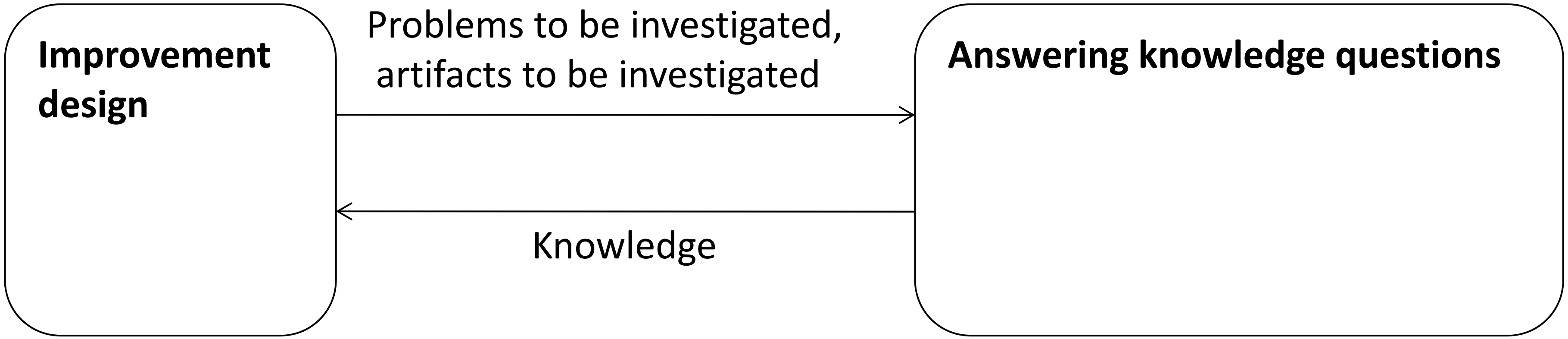


Two kinds of knowledge questions



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

Our focus



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

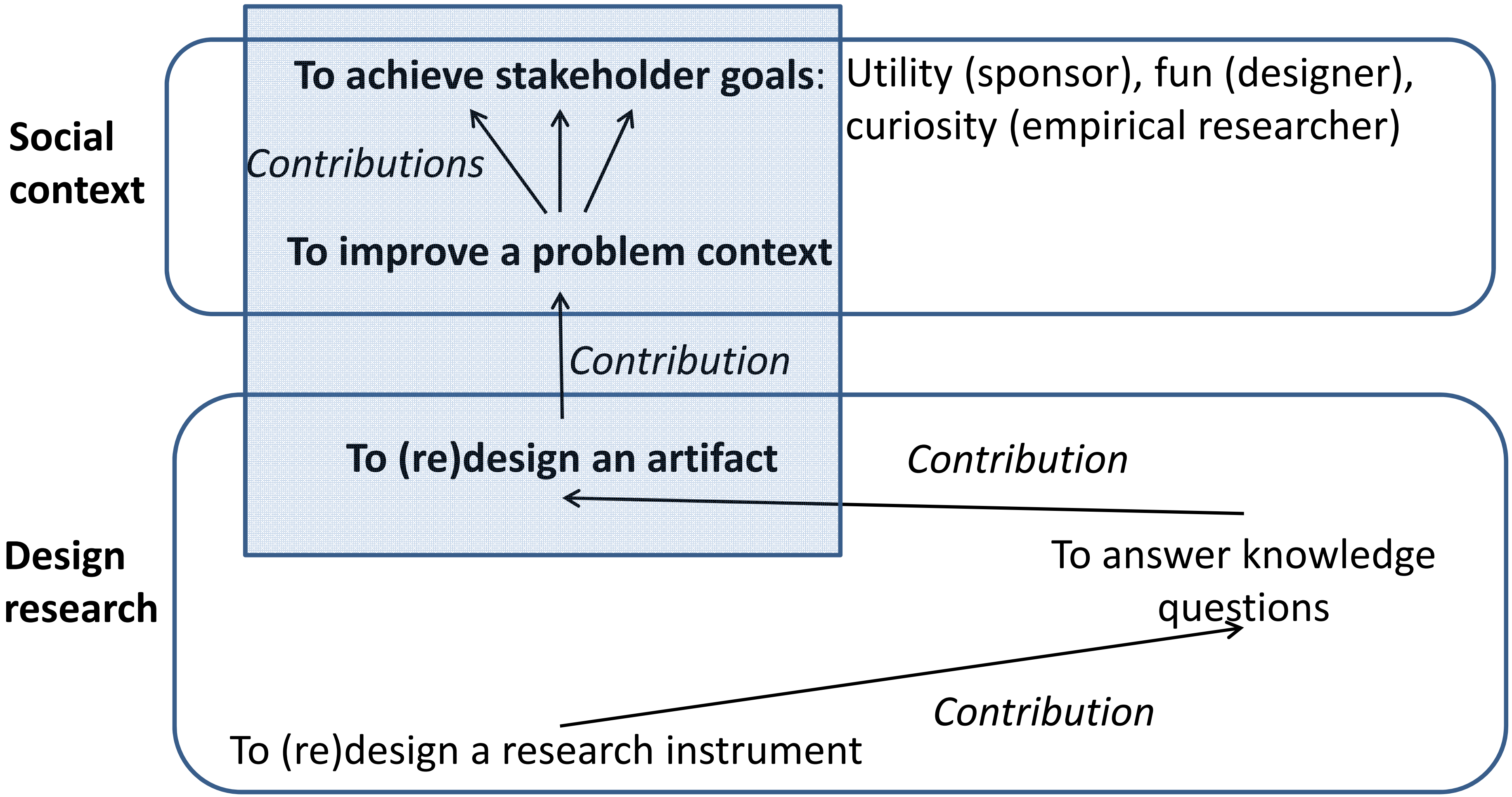
- Improve <problem context>
- by <treating it with a (re)designed artifact>
- such that <artifact requirements>
- in order to <stakeholder goals>

- *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



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.*
- 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?

Artifact

Context

- **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>**

Exercise: your top-level design problem

- What is/are your top-level design problem(s), using our template?
  - **Improve <problem context>**
  - **by <treating it with a (re)designed artifact>**
  - **such that <artifact requirements>**
  - **in order to <stakeholder goals>**

- For a knowledge-oriented thesis, think of a top-level design problem that motivates your knowledge question

Three kinds of design research questions

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

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. Yields theories.

- *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?

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

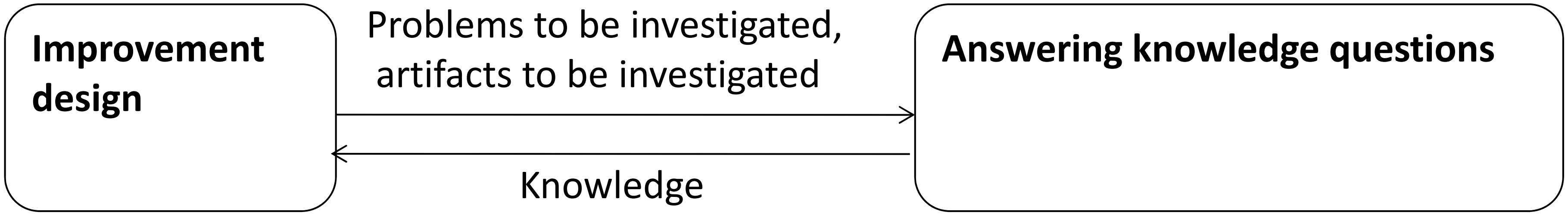
Q4: Design problem

- *[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



Design research problems

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

2. **Empirical knowledge questions**
3. **Analytical knowledge questions**

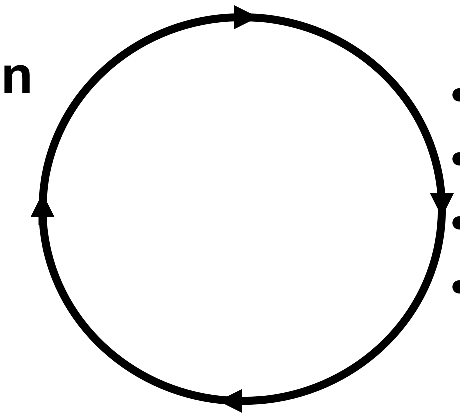
Outline

1. What is design science
   – Research goals and problems
   – **The design and engineering cycles**
2. Design theories
   – Scientific research design
   – Scientific theories
   – Scientific inference: from data to theories

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?

Design implementation

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!

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, and using it**.
  - Part of a research project

Nesting of cycles

Research project: **design cycle**

Problem investigation	
Treatment design	
Treatment validation	Problem investigation (How to do the validation?)
	Experiment design & validation (design and validate a prototype & test environment)
	Implementation (construction of prototype & test environment, lab or field)
	Evaluation (analyze results)
Implementation (tech transfer)	
Implementation evaluation (in the field)	

This is a very special engineering cycle, called the **empirical cycle**.

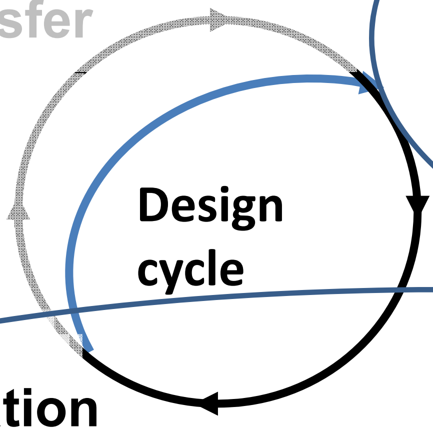
Design cycle

Real-world problem-oriented research

Real-world implementation evaluation = Real-world problem investigation

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

Real-world design Implementation: Technology transfer



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

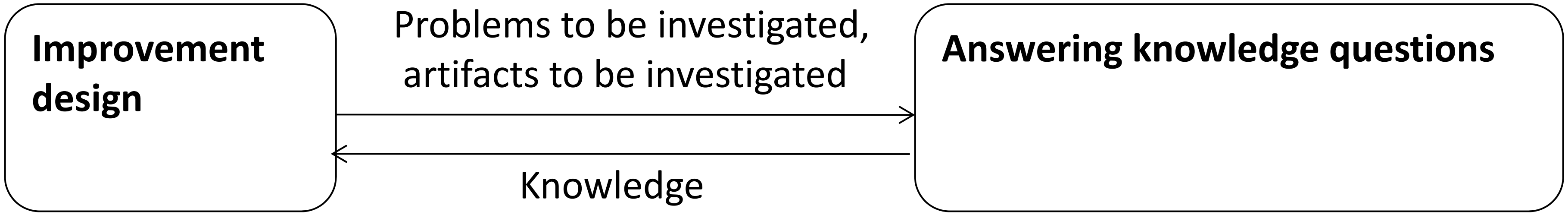
Two kinds of design science research projects

- Problem-oriented: empirical social-science-like research
  - Investigate real-world implementations.
    - E.g. How is the UML used in small and medium sized companies?
    - What is the cause if large SE projects being late?
    - How is RE done in large-scale agile projects?
- Solution-oriened: technical research
  - Design and validate an artifact
    - Design a multi-agent system for autonomous route planning
    - Design a system for remote health monitoring for the elderly
    - Design a requirements engineering technique for agile global software engineering projects

Sequence of cycles

- Design the product idea
  - Sketch the problem – design principle of operation –validation soundness of the idea
- Sketch the product
  - Describe problem – sketch product architecture – provide argument that this exhibits the necessary mechanisms
- Feasibility study
  - Same, but now validate by building small prototype in test environment
- Specify the product
  - Describe problem mechanisms and goals – Specify product requirements and structure – validate analytically and empirically
- Etc. in a sort of risk management process

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

Artifacts → Design cycle → Artefacts

Empirical knowledge questions

- Descriptive: what, how, when, where, who, etc.→ **Facts**
- Explanatory: Why → **Explanations**

Analytical knowledge questions

Questions?

Exercise (design-driven thesis) your table of contents

- Make a poster with the outline of the table of contents of your thesis, following this pattern:
  1. Introduction: Societal improvement problem, stakeholders and their goals, current designs, gap with improvement needs.
  2. Research problem: top-level design problem; decomposition into subproblems; knowledge questions
  3. State of the art: existing designs
  4. Requirements for a new design; motivation in terms of stakeholder goals; evaluation of current designs against the requirements
  5. New design
  6. Validation of new design: prototypes, simulations, field experiments, etc.
  7. (More designs and validations)
  8. Conclusions, recommendations, and further work

Exercise (knowledge-driven thesis): your table of contents

- Make a poster with the outline of the table of contents of your thesis, following this pattern:
  1. Introduction: Societal improvement problem, stakeholders and their goals, current knowledge, gap with desired knowledge.
  2. Research problem: Top-level knowledge question; decomposition into sub-questions
  3. State of the knowledge: existing knowledge
  4. Research methods followed
  5. Study: observational study, experimental, case-based, sample-based, etc.
  6. (More studies)
  7. Conclusions, recommendations, and further work

Outline

1. What is design science
   – Research goals and problems
   – The design and engineering cycles
2. **Design theories**
   – Scientific research design
   – Scientific theories
   – Scientific inference: from data to theories

Descriptions, generalizations, explanations

- Descriptive knowledge questions:
  - What happened?
  - How much? How often?
  - When? Where?
  - What components were involved?
  - Who was involved?
  - Etc. etc.
- Explanatory knowledge questions:
  - Why?
    - What caused this phenomenon?
    - What mecanisms produced it?
    - Why did people do this?

Descriptive knowledge questions:

- Yield **facts** about cases or samples.
- May be **generalized** beyond the facts, to **descriptive theories** about a population

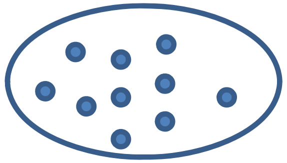
Explanatory knowledge questions:

- Beyond the facts: explanatory theories about cases/samples.
- May be **generalized** to **explanatory theories** about a population

From facts to theories

Facts

Observed sample of cases

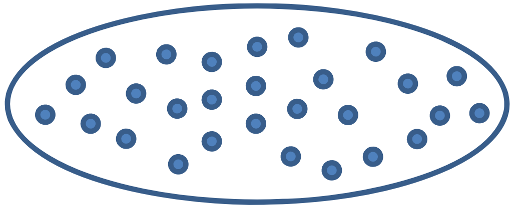


- What happens in these cases?
- What average, variance in this sample?

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

- How to answer a knowledge question?
  - Use existing knowledge: experts, literature
  - Do you own research

Outline

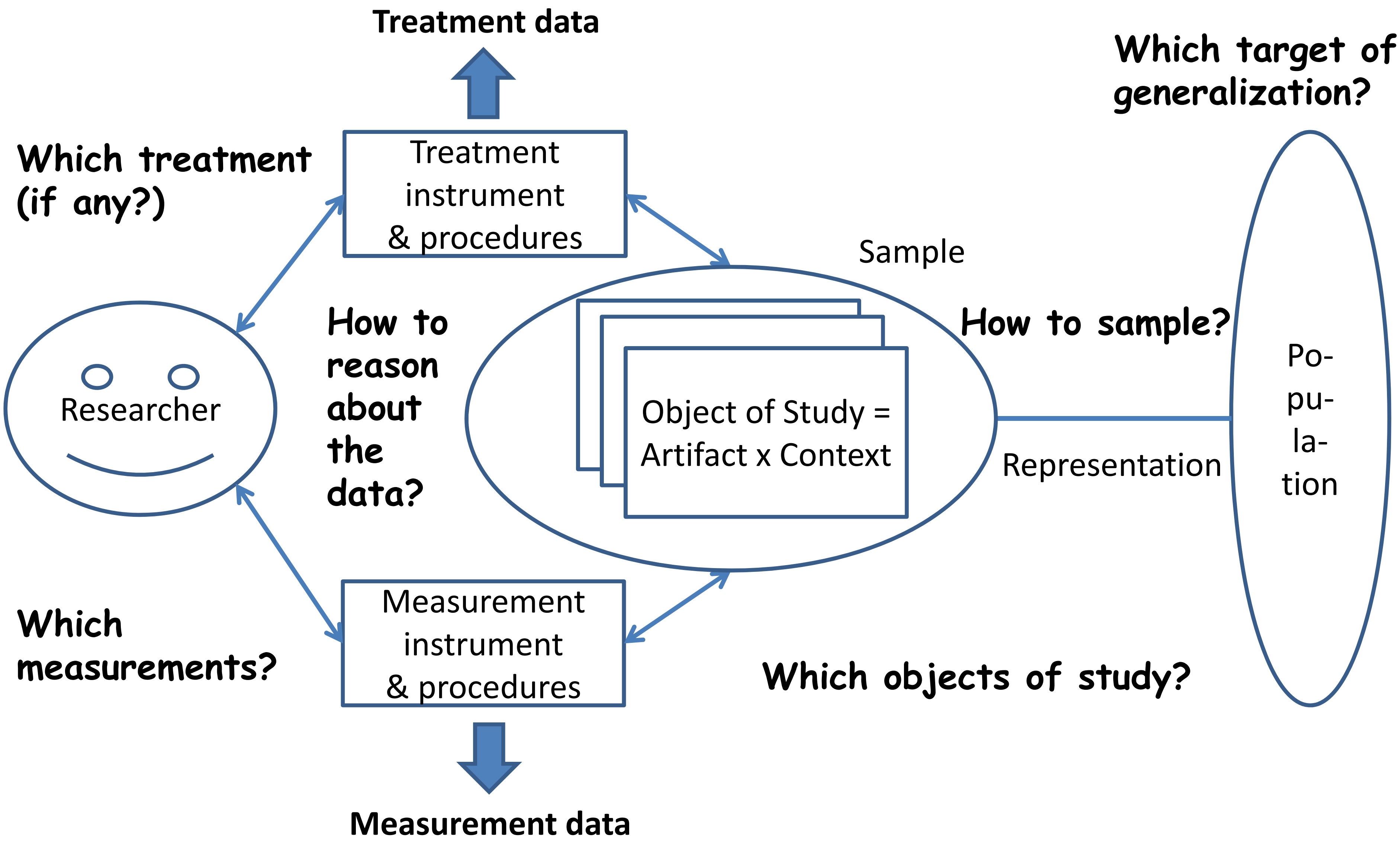
1. What is design science
   - Research goals and problems
   - The design and engineering cycles
2. Design theories
   - **Scientific research design**
   - Scientific theories
   - Scientific inference: from data to theories

Research designs

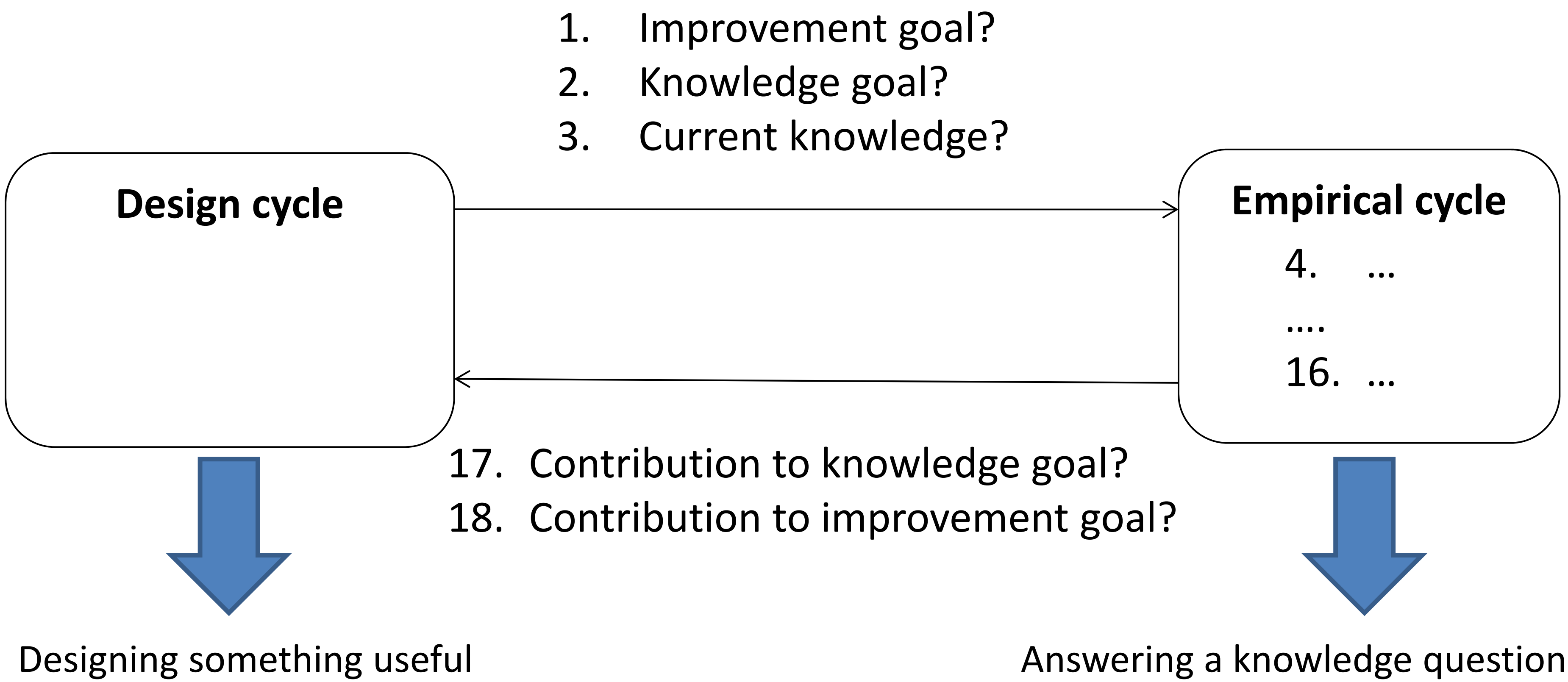
	Observational study (no treatment)	Experimental study (treatment)
Case-based: investigate single cases, look at architecture and mechanisms	**Observational case study**	• **Expert opinion** (mental simulation by experts), • **Mechanism experiments** (simulations, prototyping), • **Technical action research** (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)

Validation methods (depends on budget)

Design decisions for research setup



Checklist for research design: context



This is a checklist for
- **research design,**
- **research reporting,**
- **reading a report.**

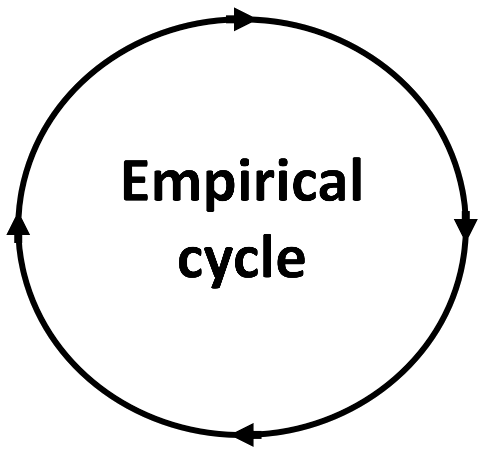
App. B in my book & my web site

Data analysis

12. Descriptions?
13. Statistical conclusions?
14. Explanations?
15. Generalizations?
16. Answers?

Research execution

11. What happened?



Research problem analysis

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

Design validation	Research & inference design	
7. Objects of study validity?	7. Objects of study?	Research setup
8. Treatment specification validity?	8. Treatment specification?	
9. Measurement specification validity?	9. Measurement specification?	
10. Inference validity?	10. Inference?	Inference

- The research setup will produce data
- Scientific inference is
  - reasoning from these data to the answers of your knowledge questions, and
  - from these answers to conclusions about theories.

Outline

1. What is design science
   - Research goals and problems
   - The design and engineering cycles
2. Design theories
   - Scientific research design
   - **Scientific theories**
   - Scientific inference: from data to theories

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*

Scientific design theories

- A **scientific design theory** is a belief 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*

The structure of scientific theories

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

The structure of scientific design theories

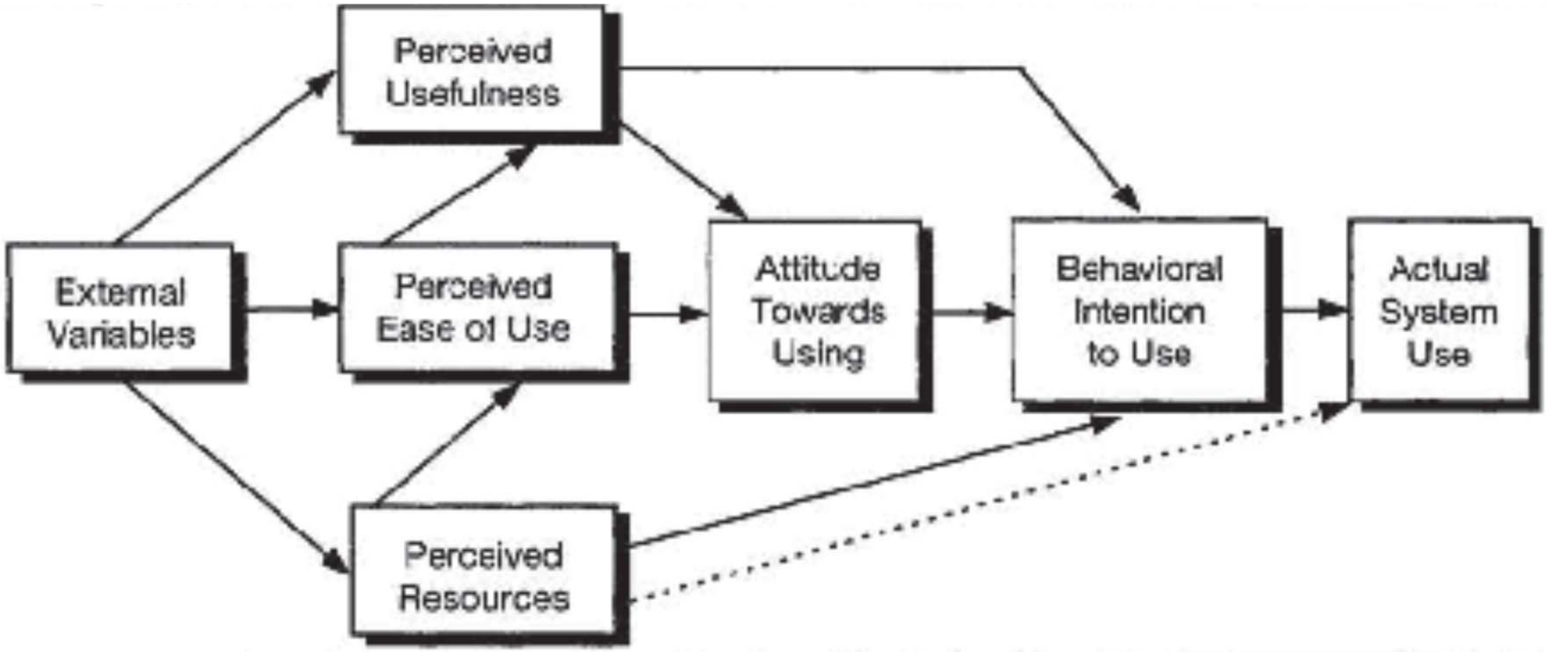
1. **Conceptual framework**
   – Definitions of concepts.
2. **Generalizations**
   – Express beliefs about patterns in interactions between artifact and context.

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.

Design theory

Theory of an algorithm

- *Concepts: definitions of concepts to specify a direction-of-arrival recognition algorithm, and of concepts to describe antenna array, and of accuracy and excution time*
- *Descriptive generalization: (Algorithm MUSIC) x (antenna array, plane waves, white noise) → (execution time less than 7.2 ms.)*
- *Explanatory generalization: qualitative explanation by analysis of the algorithm.*

Another design theory

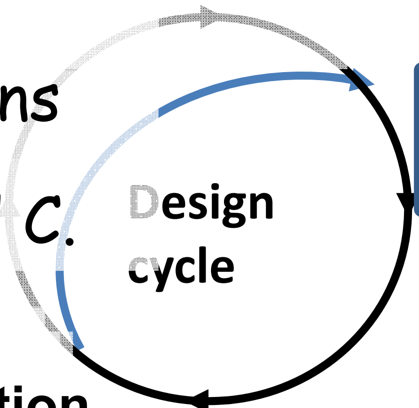
- *Descriptive UML theory*
  - *Concepts: UML concepts, definitions of software project, of software error, project effort.*
  - *Descriptive generalization: (UML) X (SE project) → (Less errors, less effort than similar non-UML projects)*

- *Explanatory UML theory:*
  - *Concepts: definition of concept of domain, understandability*
  - *Explanatory generalizations:*
    - *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).*

The use of theories in the design cycle

Design implementation

Design theory describes and possibly explains interaction between A and C.



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!

Problem theory describes and explains the problem: Symptoms and diagnosis.

All generalizations can be used to make predictions

- A **general problem theory** describes and explains a type of problem. General symptoms and diagnosis.
- **A general design theory** describes and possibly explains interaction between Artifact and Context in general.

- Both theories generalize, and so may be used to predict.
  - What will happen if the problem is untreated?
  - What will happen if the treatment is applied?

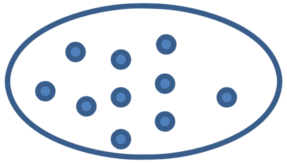
Outline

1. What is design science
   – Research goals and problems
   – The design and engineering cycles
2. Design theories
   – Scientific research design
   – Scientific theories
   – **Scientific inference: from data to theories**

From facts to theories

Facts

Observed sample of cases

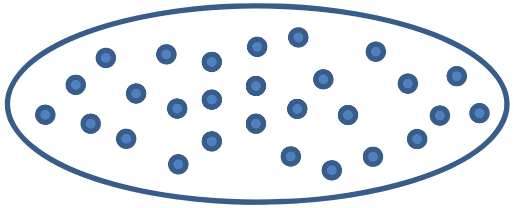


- What happens in these cases?
- What average, variance in this sample?

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

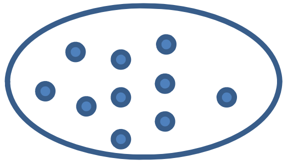
- Why?

Explanatory theory of the population

Three kinds of explanation

Facts

Observed sample of cases

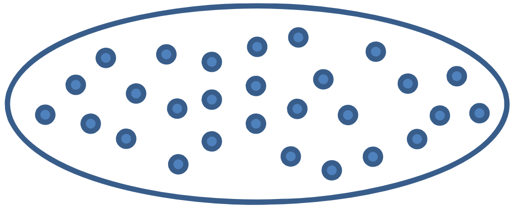


- What happens in these cases?
- What average, variance in this sample?

Generalize

Descriptive theory of the population

Unobserved population



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

Explain by
- Causes
- Mechanisms
- Reasons

- Why?

Explanatory theory of the case/sample

Explain by
- Causes
- Mechanisms
- Reasons

- Why?

Explanatory theory of the population

Example explanations (1)

- *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.

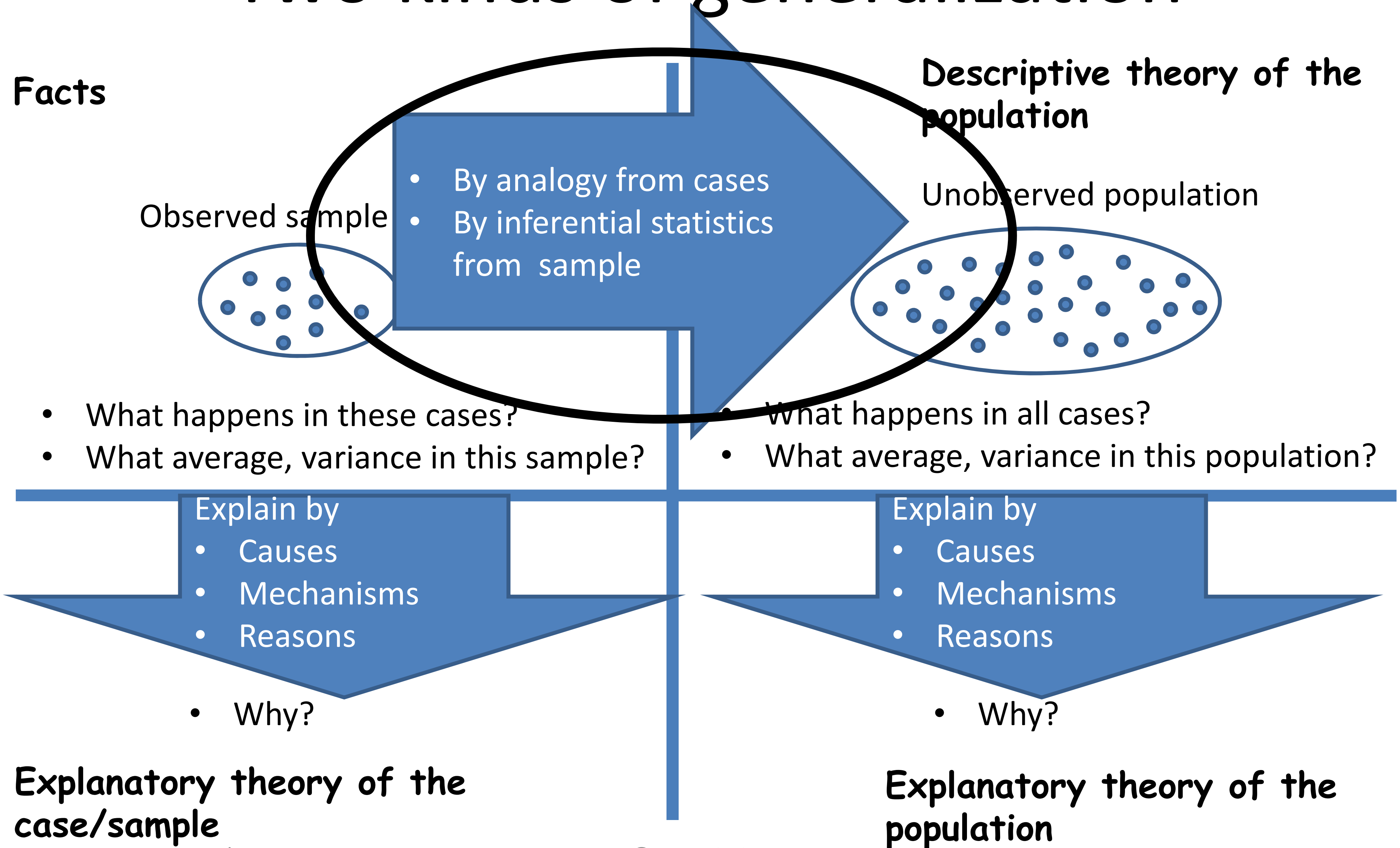
Example explanations (2)

- *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 choose to drive on busy roads with a lot of signal interference*

Example explanations (3)

- *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 project performance is attributed to difference between UML and non-UML methods.*
  2. ***Mechanism:*** *The difference in effects is by the match between UML and the structure of cognition.*
  3. ***Reasons:*** *Difference in performance may be explainable by difference in motivation of developers to use UML or something else.*

Two kinds of generalization



Case-based generalization (1)

- *Observation:*
  - *Artifact: A light switch*
  - *Context: next to the door in the wall of a room with ceiling lights*
  - *Effect: toggles the ceiling light on and off.*
- *Explanation:*
  - *The switch and context architectures produce this behavior*
- *Generalization by analogy:*
  - *All **similar** switches*
  - *Running in **similar** contexts*
  - *Will show **similar** effects*

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.

Case-based generalization (2)

- *Observation:*
  - *Artifact: This prototype implementation of the MUSIC algorithm,*
  - *Context: when used to recognize direction of arrival of plane waves received by an antenna array, 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.*
- *Explanation:*
  - *Algorithm structure*
- *Generalization by analogy:*
  - *All **similar** implementations*
  - *Running in **similar** 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.

Case-based generalization (3)

- *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,*
  - *so the software engineers make less errors and there is less rework.*
- *Generalization*
  - *In similar projects, UML will have similar effects*
  - *Assumptions: The mechanisms that produced these effects will be present in all similar projects, i.e. UML is used in the same way, and any relevant social and cognitive mechanisms are present in similar projects too, and*
  - *The effects will not be undone by other mechanisms*

Case-based generalization

- Must be based on architectural similarity
  - Similar components, with similar capabilities
  - Similar mechanisms involving these components
- Analogy based in similarity of superficial features, without knowledge of underlying mechanisms, is too weak a basis for generalization.
  - *Wallnuts look like brains.*
  - *Brains can think.*
  - *Therefore .... Wallnuts can think*
- *There is no shared mechanism that produces thinking in brains and wallnuts!*

Sample-based generalization

1. **By big data:** If the sample is almost the size of the population, then the population probably has similar statistics.
   – Only true if the sample is random. Law of large numbers.
2. **By statistical learning:** Use a sample of (X, Y) values to estimate Y as a function of X in the population.
   – E.g. regression. Different methods come with different assumptions.
3. **Bayesian inference.** Use a sample to update a hypothesized distribution of a variable over the population
   – Need to start with an initial hypothesized distribution.
4. **Frequentist statistical inference:** In repeated random sampling from the same population, the sample averages are approximately normally distributed around the population mean.
   – Central-limit theorem. Assumes random samples.

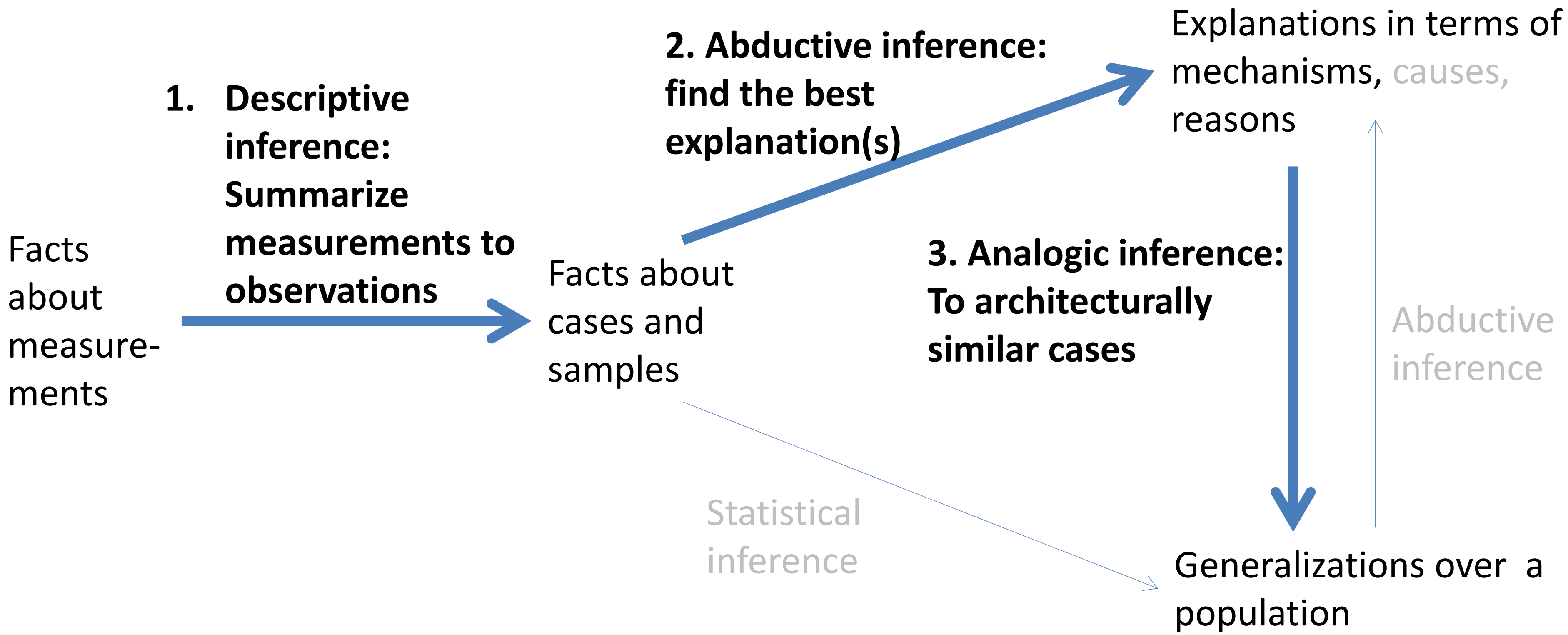
Four varieties of frequentist statistical inference

- Fisher: Test a null hypothesis that is unlikely, given what you know
- Neyman-Pearson: Decide between alternative hypotheses, based on a previously set of error rates
- Neyman: Estimate a confidence interval of a distribution parameter
- Social sciences: Null Hypothesis Significance Testing (NHST). Misconceived and logically incorrect mix of Fisher & Neyman-Pearson

Sample-based inference

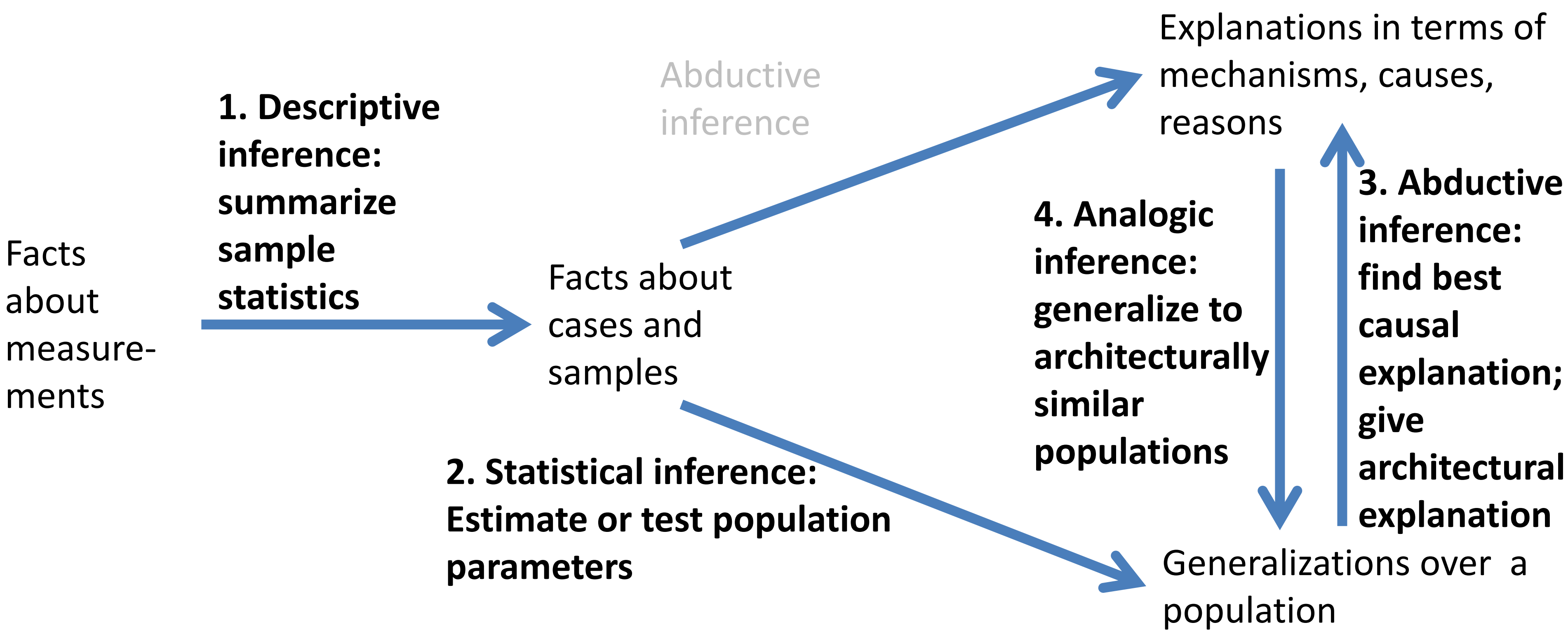
- T. Huynh, J. Miller. An empirical investigation into open source web applications' implementation vulnerabilities. *Empir. Softw. Eng.* **15**(5), 556–576 (2010)
- *Sample of 20 open source web applications from the population of all OS web applications. Count the number of security vulnerability caused by coding errors (rather than by design flaws or configuration errors).*
- *Observation: The average percentage of vulnerabilities caused by coding errors per OS web application* ***in the sample*** *is 73%.*
- *Generalization by statistical inference:*
  - *Assuming a random sample, and*
  - *assuming that the proportion of coding errors is constant and independent across web applications,*
  - *the average percentage of vulnerabilities caused by coding errors in any OS web application* ***in the population*** *is roughly 73% ± 4% with roughly 95% confidence (95% of the times we conclude this, the conclusion is correct)*

Case-based inference



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

Sample-based inference

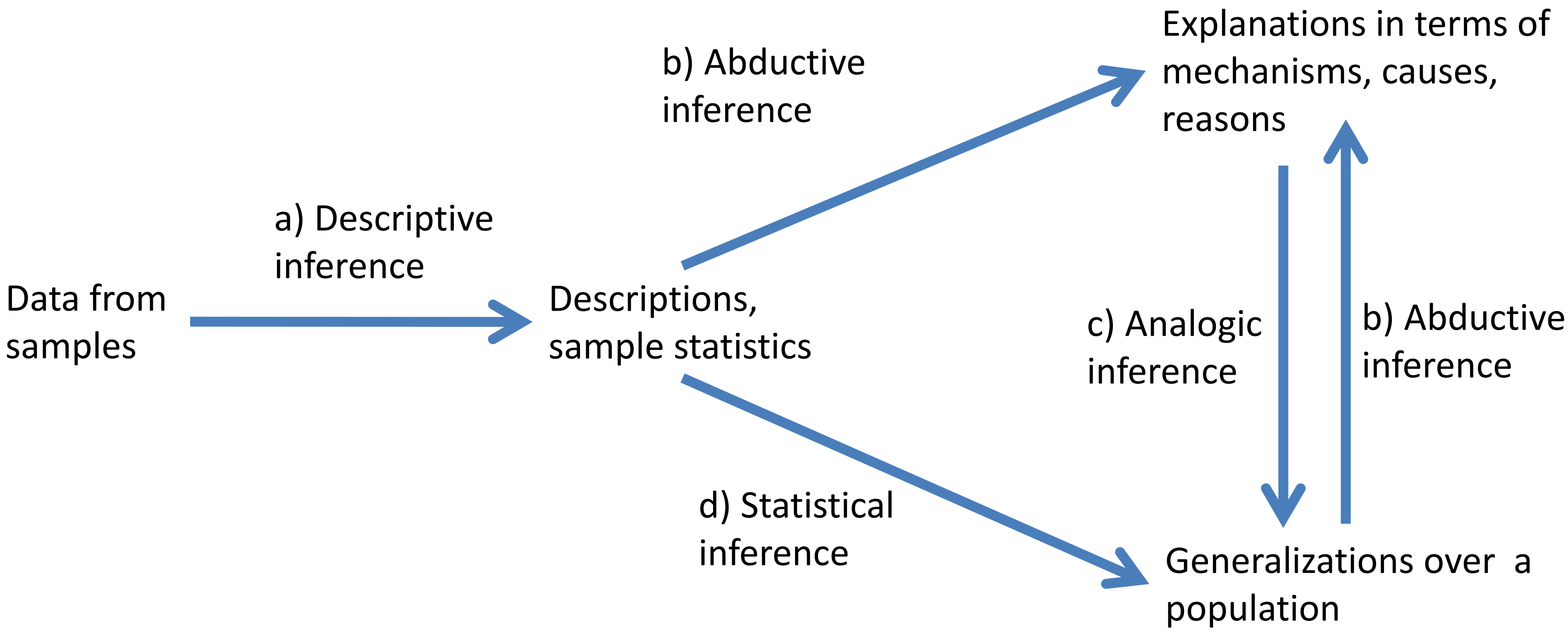


- Statistical inference yields descriptive generalization over a study population.
- Differences in outcome may be explainable by causes
- Analogic generalization to similar populations must be based on architectural explanation of those causes.

Research designs and inferences

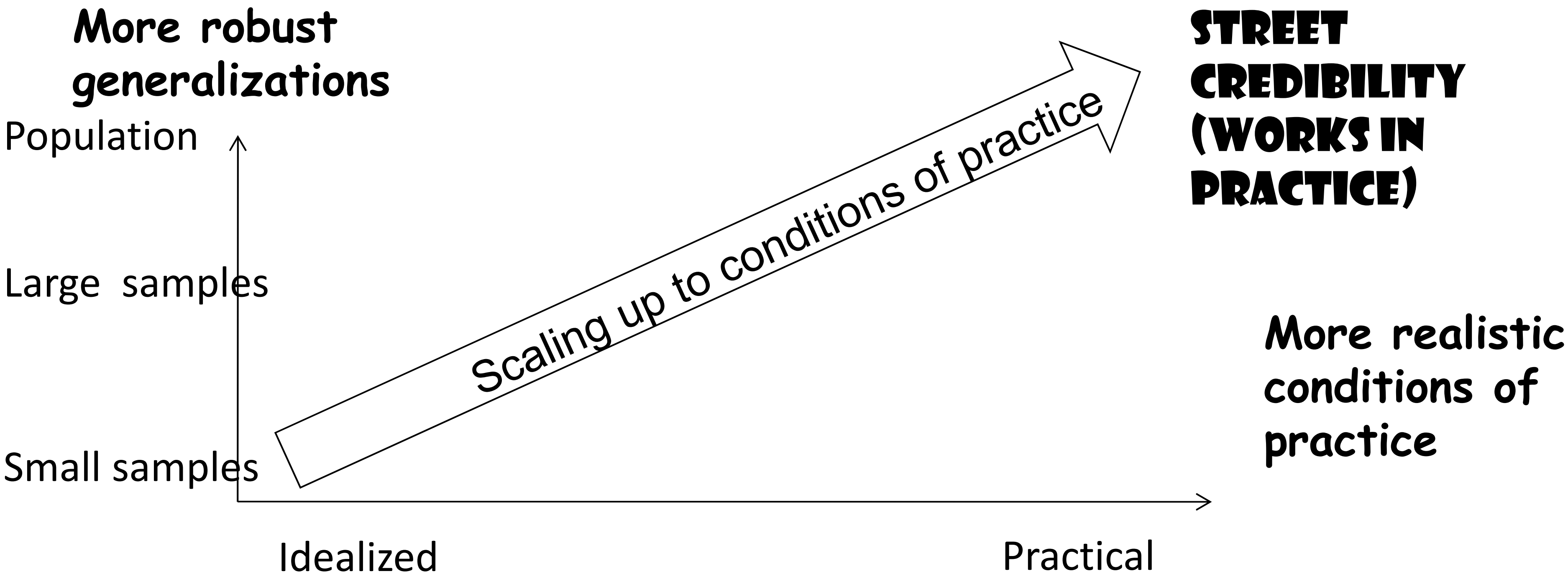
	Observational study (no treatment)	Experimental study (treatment)
Case-based: investigate single cases, look at architecture and mechanisms. Inference: Architectural explanation, generalization by analogy	**Observational case study**	• **Expert opinion** (mental simulation by experts), • **Mechanism experiments** (simulations, prototyping), • **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. Inference: Statistical inference, causal explanation, possible architectural explanation and analogy	**Survey**	• **Statistical difference-making experiment** (treatment group – control group experiments) **Validation methods (depends on budget)**

Validity of inferences: degree to which they are justified



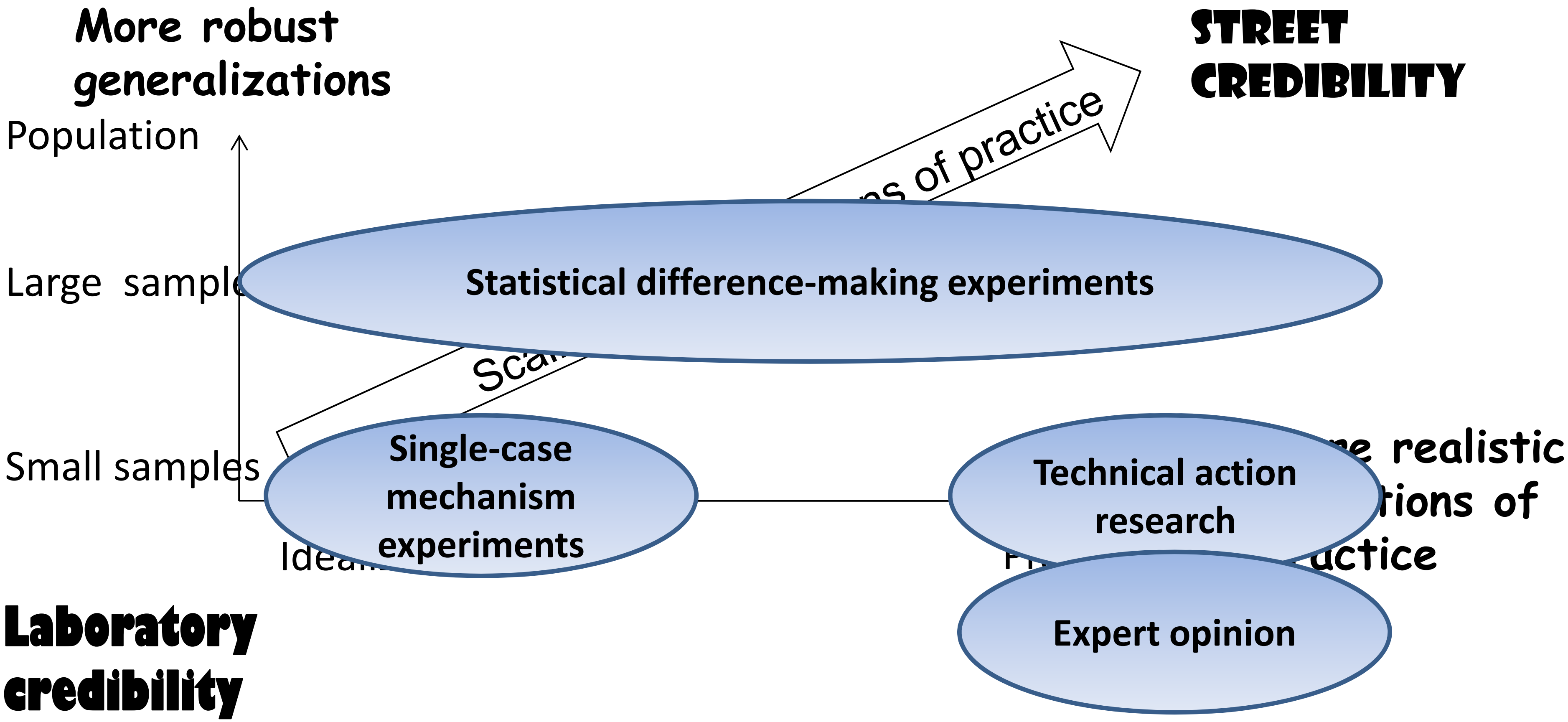
a) Descriptive validity: no information added in the descriptions
b) Internal validity: degree of support for explanations
c) External validity: degree of support for analogic generalizations
d) Statistical conclusion validity: degree of support for statistical inference

Design science research strategy



Laboratory credibility (works in theory)

- 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)

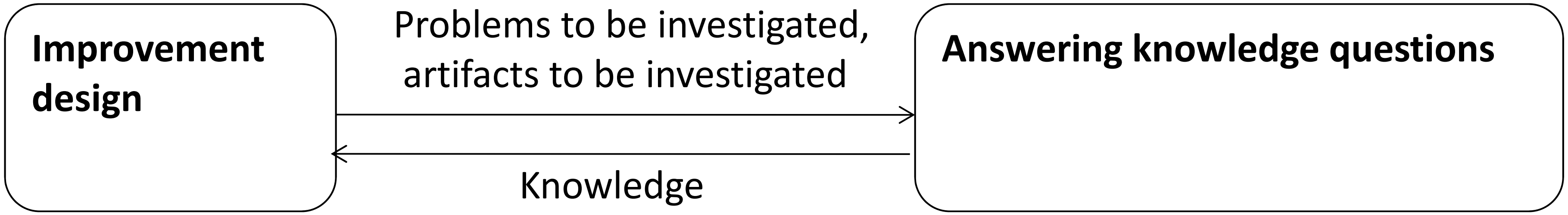
Exercise (design-driven thesis): your table of contents

- Make a poster with the outline of the table of contents of your thesis, following this pattern:
  1. Introduction: Societal improvement problem, stakeholders and their goals, current designs, gap with improvement needs.
  2. Research problem: top-level design problem; decomposition into subproblems; **knowledge questions**
  3. State of the art: existing designs
  4. Requirements for a new design; motivation in terms of stakeholder goals; evaluation of current designs against the requirements
  5. New design
  6. **Validation of new design: prototypes, simulations, field experiments, etc**.
  7. (More designs and validations)
  8. Conclusions, recommendations, and further work

Exercise (knowledge-driven thesis): your table of contents

- Make a poster with the outline of the table of contents of your thesis, following this pattern:
  1. Introduction: Societal improvement problem, stakeholders and their goals, current knowledge, gap with desired knowledge.
  2. **Research problem: Top-level knowledge question; decomposition into sub-questions**
  3. State of the knowledge: existing knowledge
  4. **Research methods followed**
  5. **Study: observational study, experimental, case-based, sample-based, etc.**
  6. (More studies)
  7. Conclusions, recommendations, and further work

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

Artifacts → Design cycle → Artefacts

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

Theories → Empirical cycle → Theories

Analytical knowledge questions

- 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.