Design science research in information systems and software systems engineering

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

© R.J. Wieringa

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.

© R.J. Wieringa

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

© R.J. Wieringa

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

© R.J. Wieringa

Outline

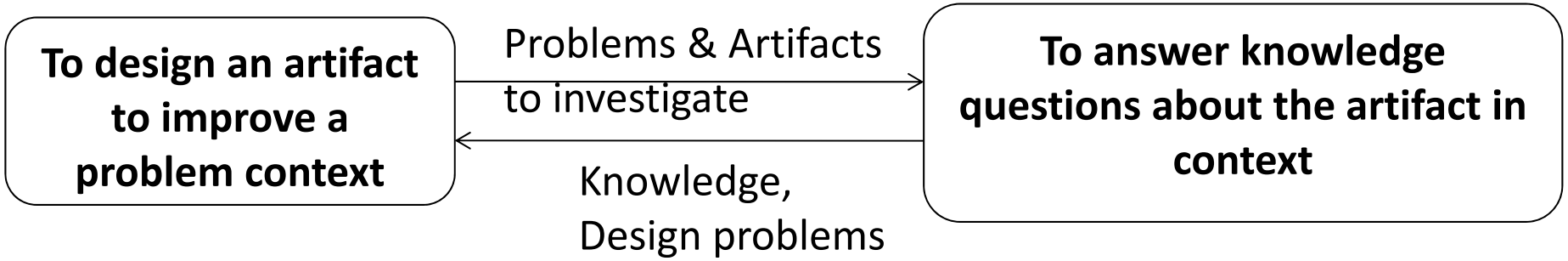
1. **What is design science**
   - Research goals and problems
   - The design and engineering cycles
2. Theories
   - Scientific inference
   - Research design

What is design science?

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

© R.J. Wieringa

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*
- *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**?

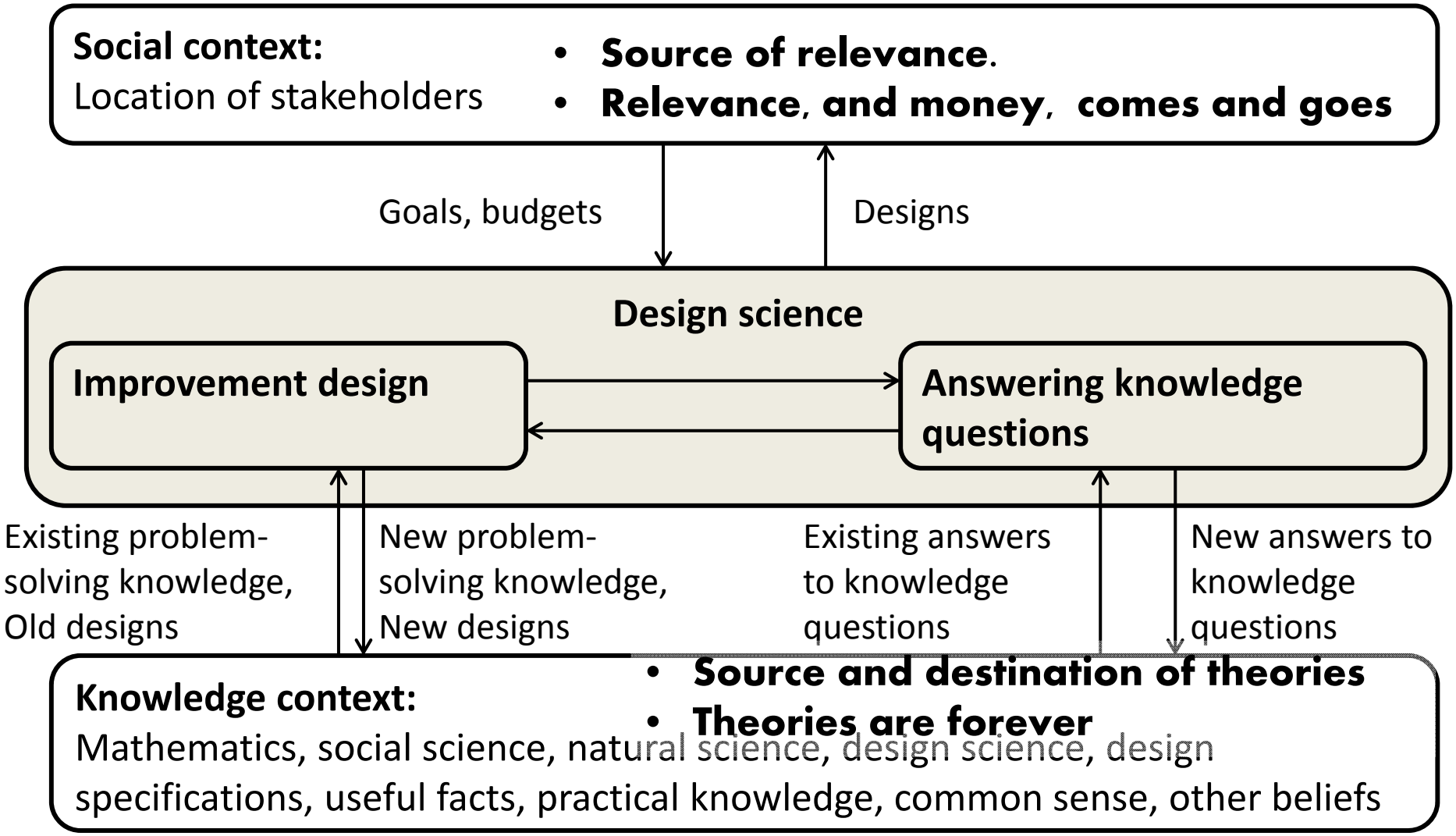
© R.J. Wieringa

Reality check

- What research problem(s) are you investigating?
  - Artifact and context
- NB
  - The title of your thesis is the shortest summary of your research project.
  - Often, it mentions the artifact and the context.

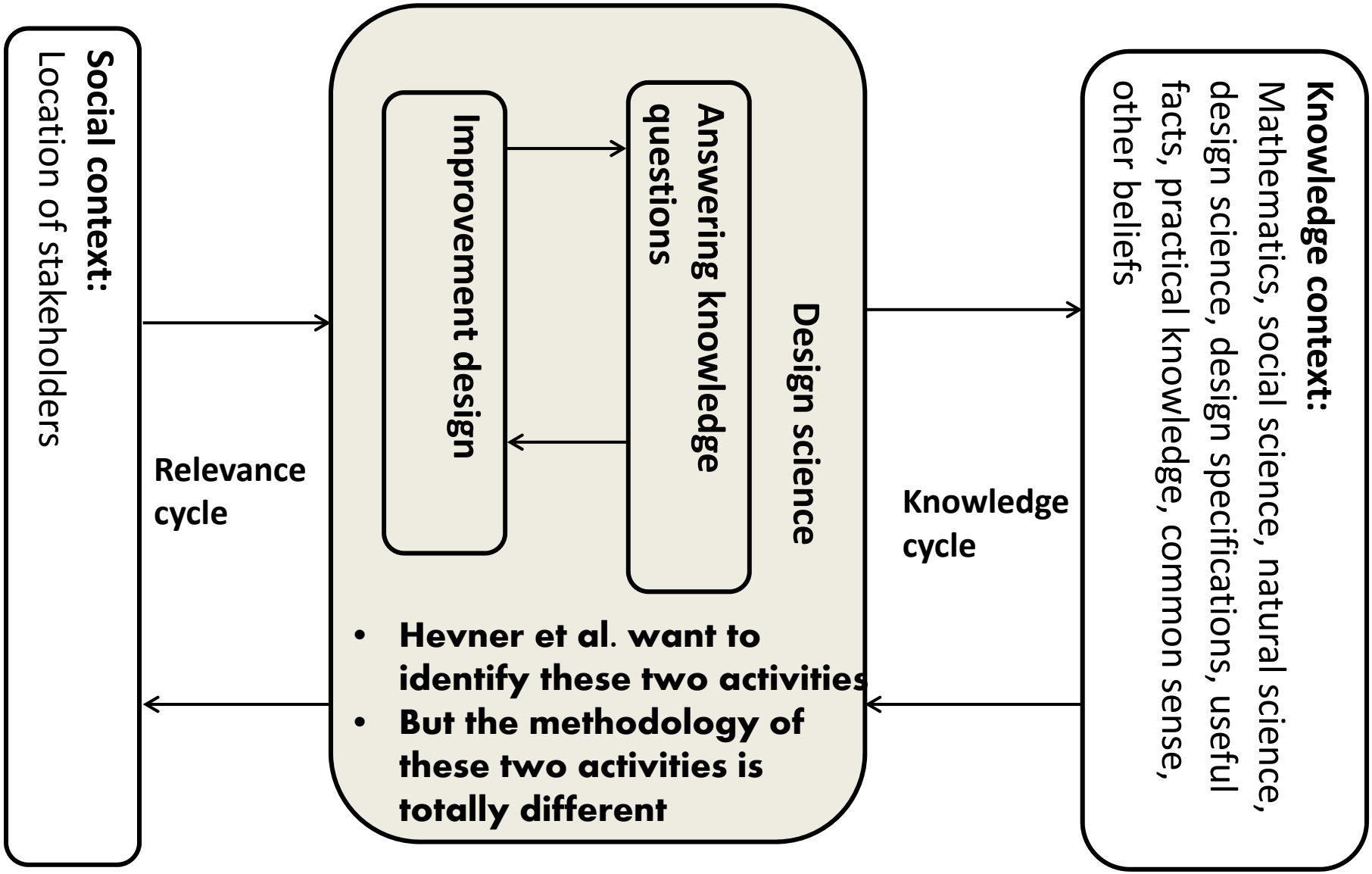
© R.J. Wieringa

Framework for design science



© R.J. Wieringa

(Dis)similarity to Hevner et al. framework

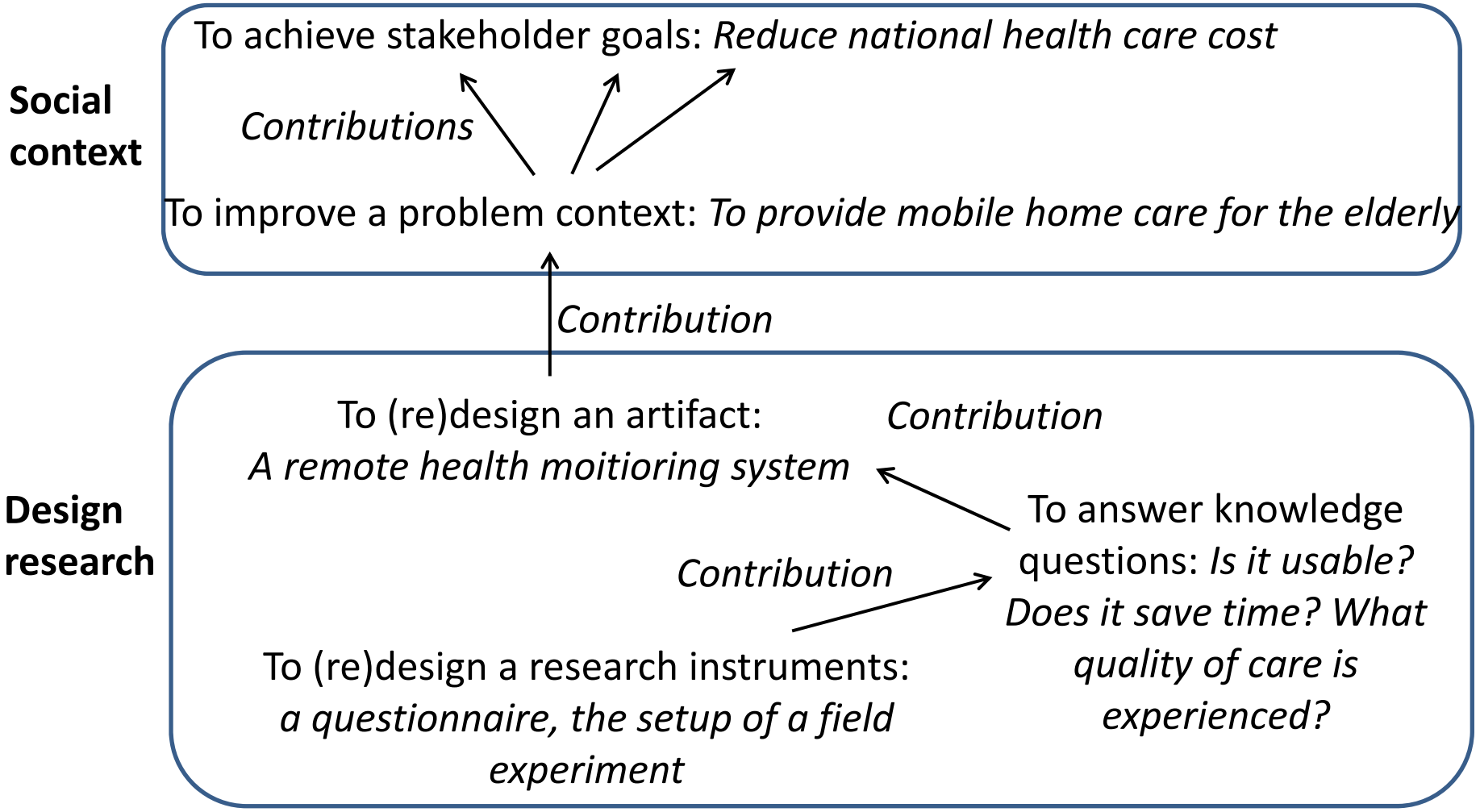


© R.J. Wieringa

Outline

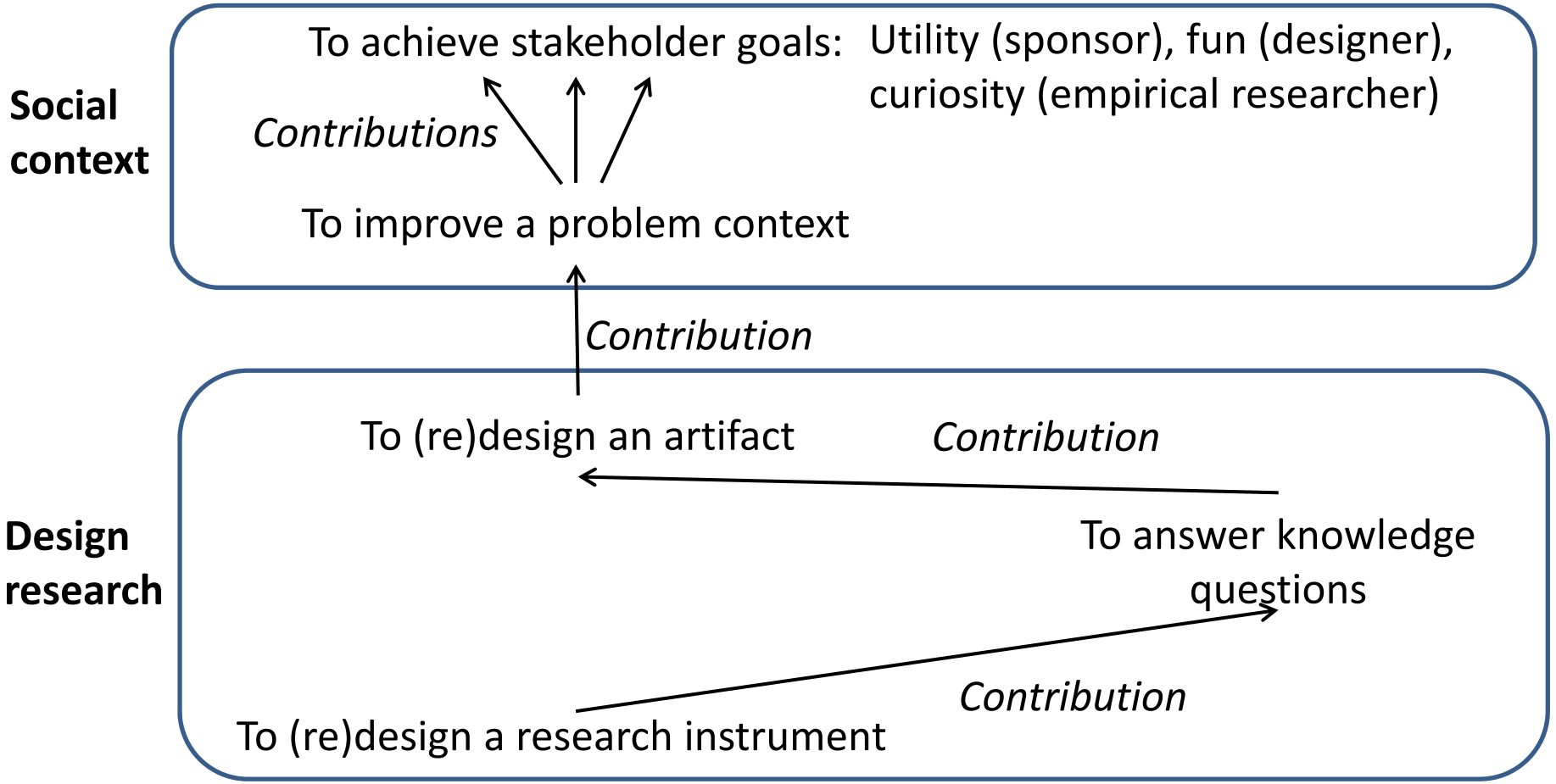
1. What is design science
   - **Research goals and problems**
   - The design and engineering cycles
2. Theories
   - Scientific inference
   - Research design

Goal structure: example



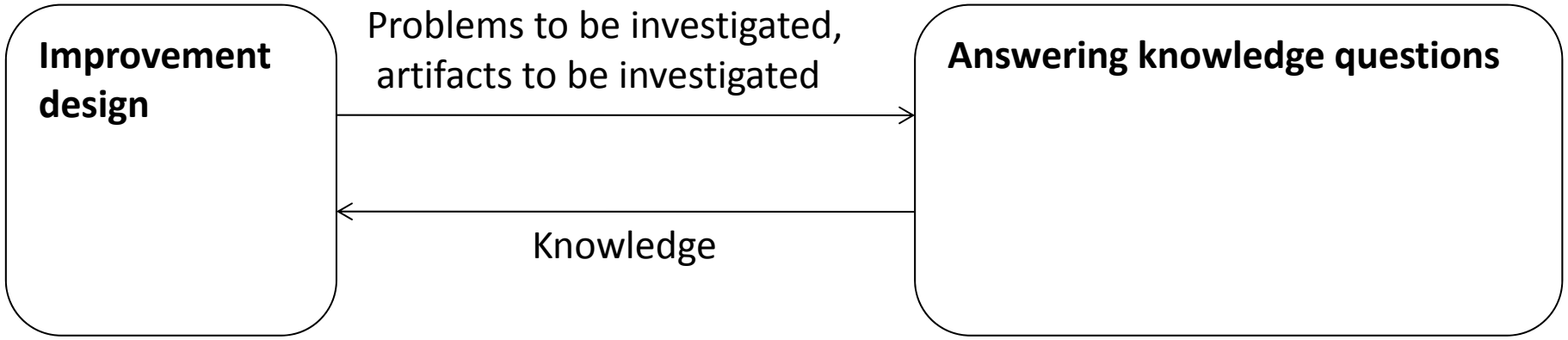
© R.J. Wieringa

Goal structure



© R.J. Wieringa

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

© R.J. Wieringa

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*

© R.J. Wieringa

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

© R.J. Wieringa

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

© R.J. Wieringa

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>

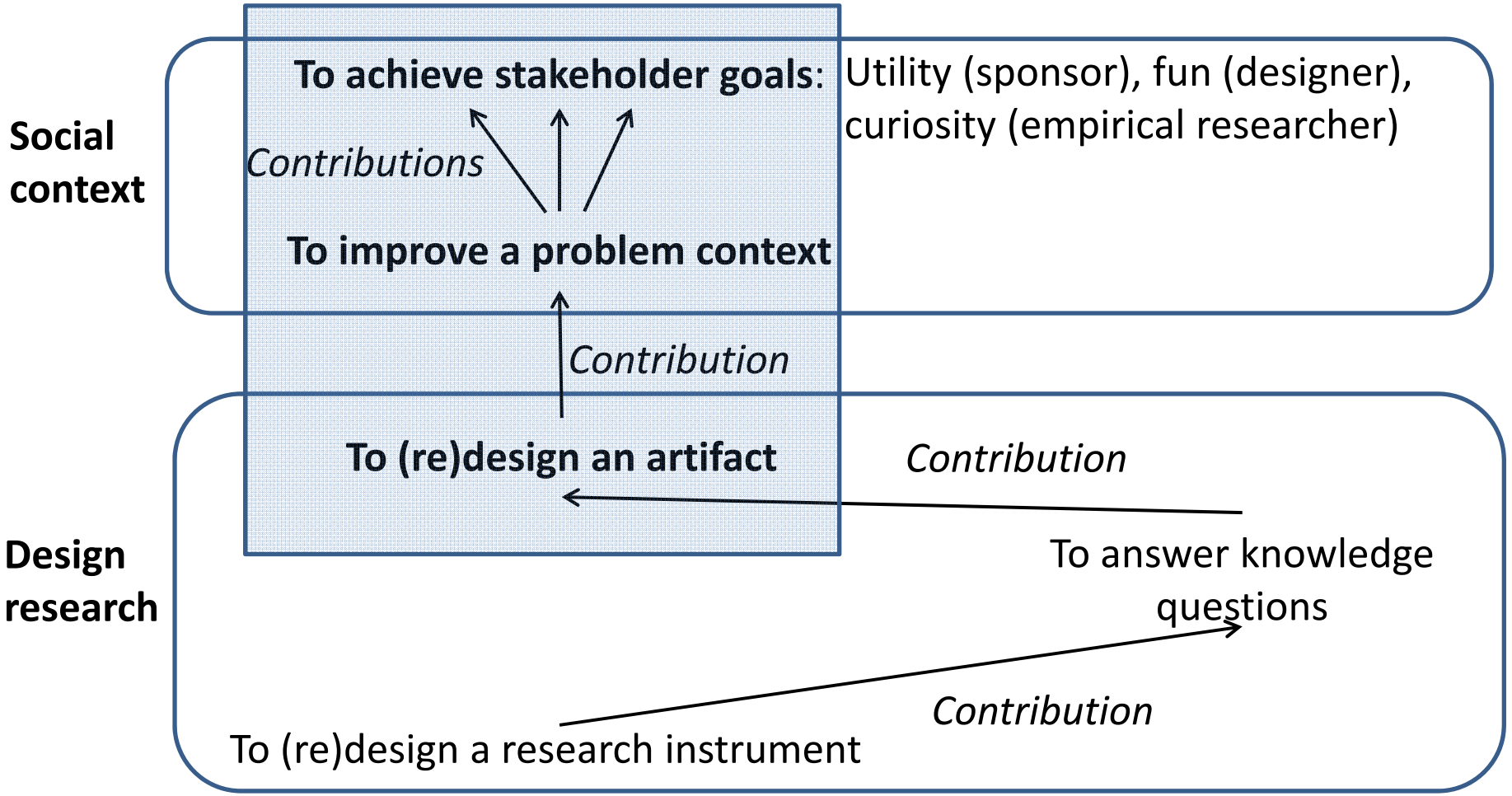
- *Reduce patients' headaches*
- *by treating it with a medicine*
- *that reduces pain fast and is safe*
- *in order for them to function as they wish*

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

© R.J. Wieringa

Goal structure again

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



© R.J. Wieringa

Discussion

- Who are the stakeholders of your project?
  - Real or hypothetical: Stakeholders may not know they are stakeholders
- 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>**

- 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>**

© R.J. Wieringa

- *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?

Context

Artifact

- **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>**

© R.J. Wieringa

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.

© R.J. Wieringa

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.

© R.J. Wieringa

Discussion

- What descriptive and explanatory knowledge are you searching for in your project?

- *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)

Design problem

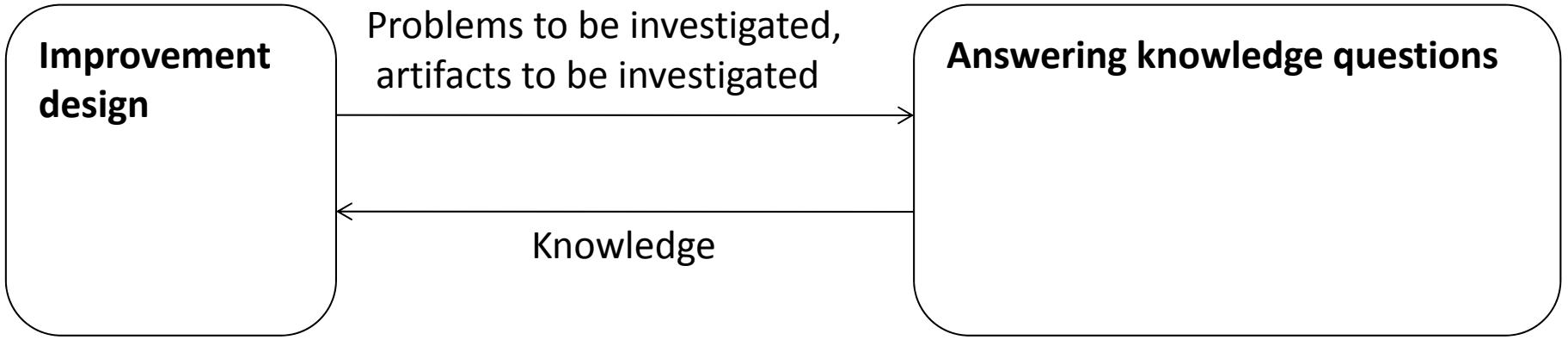
- *Automated generation of attack trees by unfolding graph transformation systems.*
  - RQ1: Can graph transformation be used as a modeling paradigm to specify systems and organizations as input models for the attack tree generation approach?
  - RQ2: Can partial-order reduction, and specifically the 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?

© R.J. Wieringa

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.

© R.J. Wieringa

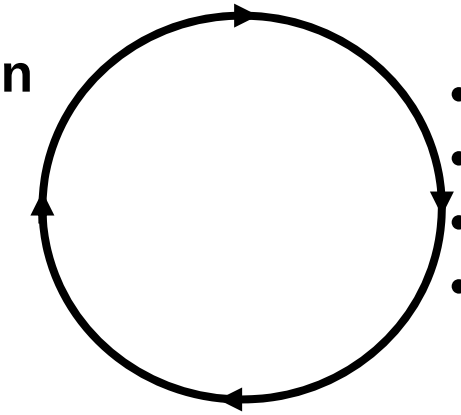
Outline

1. What is design science
   - Research goals and problems
   - **The design and engineering cycles**
2. Theories
   - Scientific inference
   - Research design

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!

© R.J. Wieringa

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

© R.J. Wieringa

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**.

© R.J. Wieringa

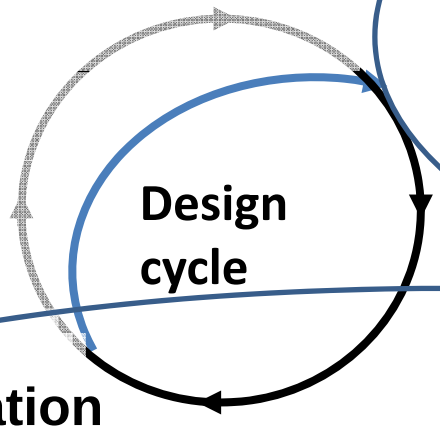
Design cycle

Real-world problem-oriented research

Real-world design implementation

Real-world implementation 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

© R.J. Wieringa

Two kinds of design science research projects

- Problem-oriented: 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

- *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?

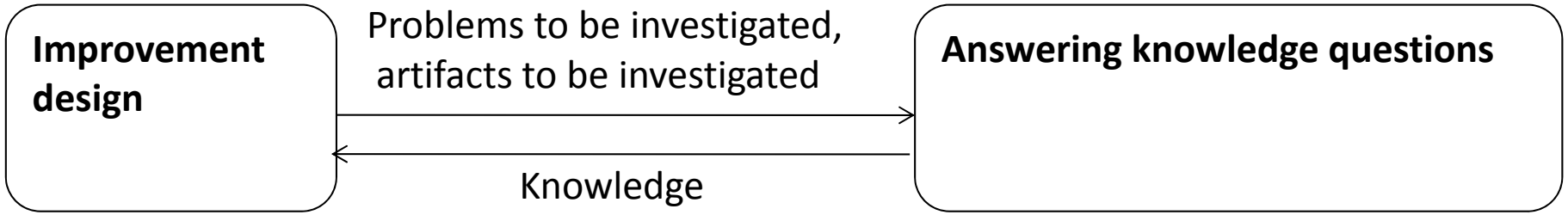
Problem *Implied, no further details.*

- 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?

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

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

© R.J. Wieringa

Questions?

Outline

1. What is design science
   - Research goals and problems
   - The design and engineering cycles
2. **Theories**
   - Scientific inference
   - Research design

© R.J. Wieringa

Facts, 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?

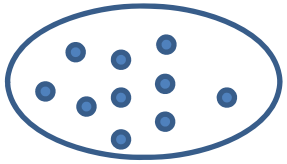
- Yield **facts** about cases or samples.
- May be **generalized** beyond the facts, to **descriptive theories** about a population

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

© R.J. Wieringa

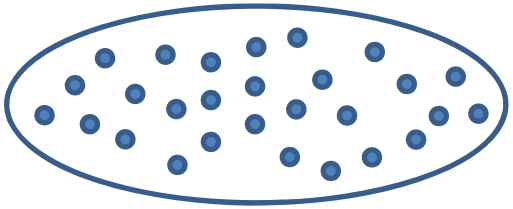
Two ways to go beyond the facts

Observed sample of cases



Generalize

Unobserved population



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

Explain

- Why?

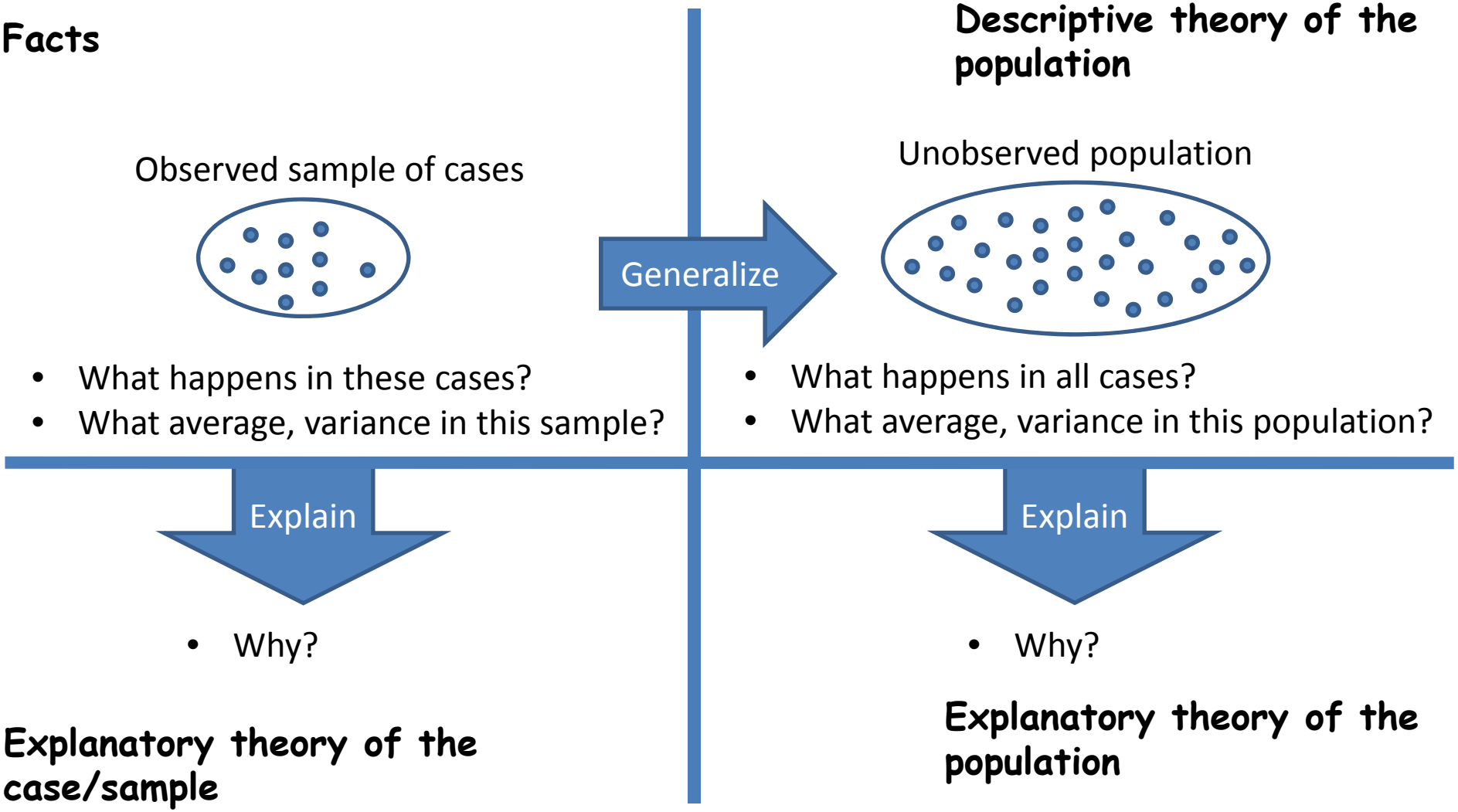
- What happens in all cases?
- What average, variance in this population?

Explain

- Why?

© R.J. Wieringa

Facts versus theories



© R.J. Wieringa

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*

© R.J. Wieringa

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*

© R.J. Wieringa

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.

© R.J. Wieringa

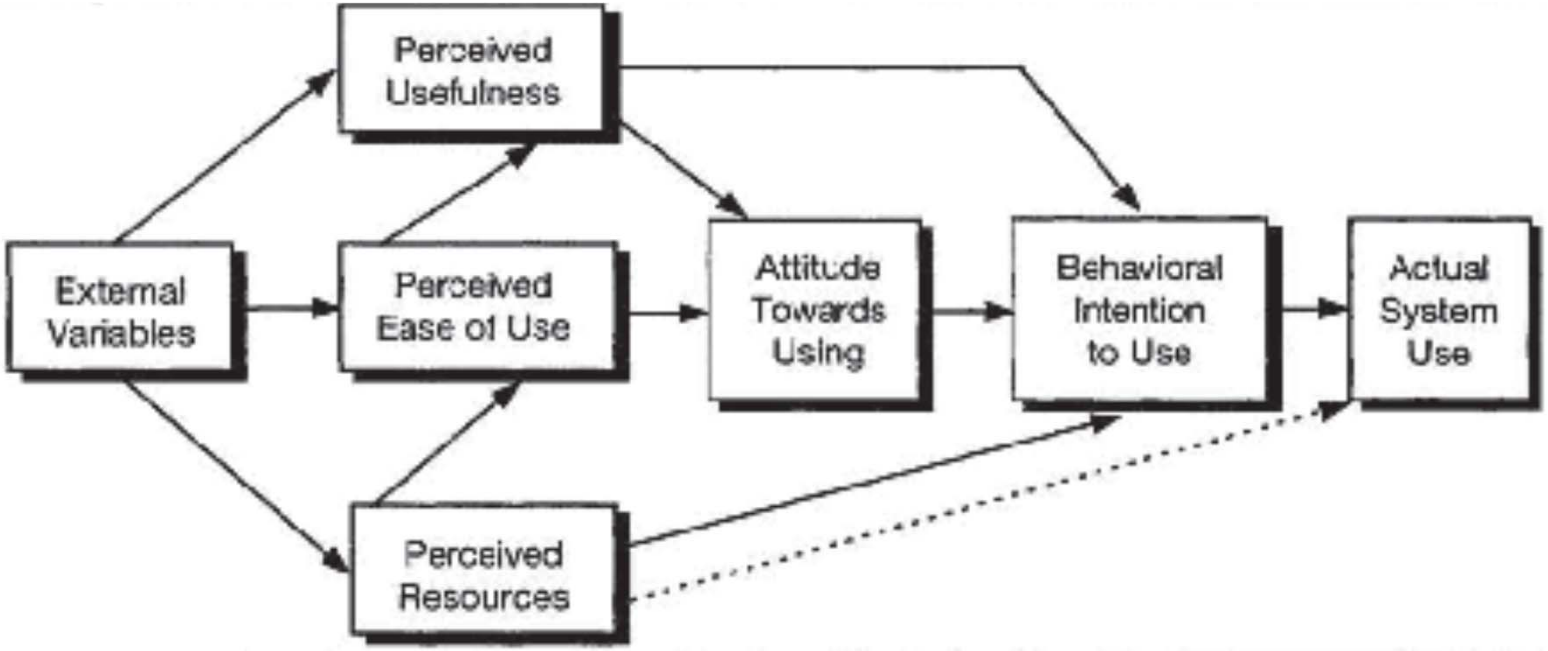
Theory of electromagnetism

- *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*

© R.J. Wieringa

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.

- Example design theory
  - *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*
  - *Generalization: (Algorithm MUSIC) x (antenna array, plane waves, white noise) → (execution time less than 7.2 ms.)*

- *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)*

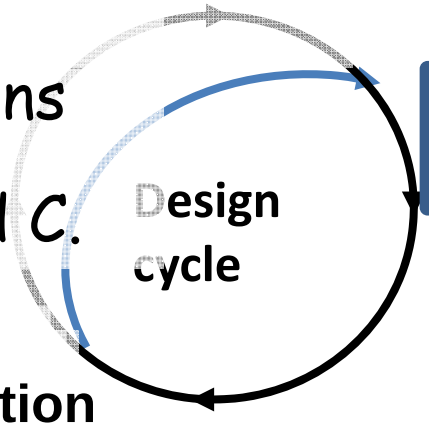
- *Explanatory UML theory:*
  - *Concepts: definition of concept of domain, understandability*
  - *Explanatory generalizations:*
    - *UML models resemble the domain more than other kinds of models;*
    - *they are easier to understand for software engineers;*
    - *So they they make less errors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

The use of theories in the design cycle

Design implementation

Implementation evaluation = Problem investigation

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



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

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

Treatment validation

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

Treatment design

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

© R.J. Wieringa

All theories can be used to make predictions

- **Problem theory** describes and explains the problem. Symptoms and diagnosis.
- **Design theory** describes and possibly explains interaction between A and C.

- Both theories may be used to predict
  - What will happen if the problem is untreated
  - What will happen if the treatment is applied

© R.J. Wieringa

Discussion

- Which theory do you hope to produce?
  - (Theory about real-world problems/implementations, or theory about newly designed artifact in a context.)
  - Explanations, generalizations provided by the theory?
- What evidence do you have, and what do you still intend to produce?

Outline

1. What is design science
   - Research goals and problems
   - The design and engineering cycles
2. Theories
   - **Scientific inference**
   - Research design

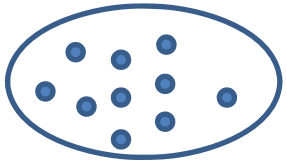
© R.J. Wieringa

Facts versus theories

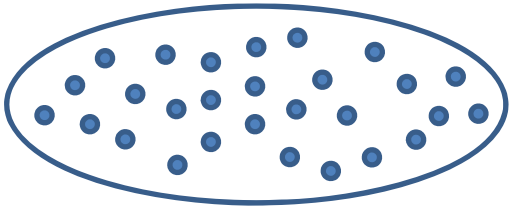
Facts

Descriptive theory of the population

Observed sample of cases



Unobserved population



Generalize

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

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

Explain

Explain

- Why?

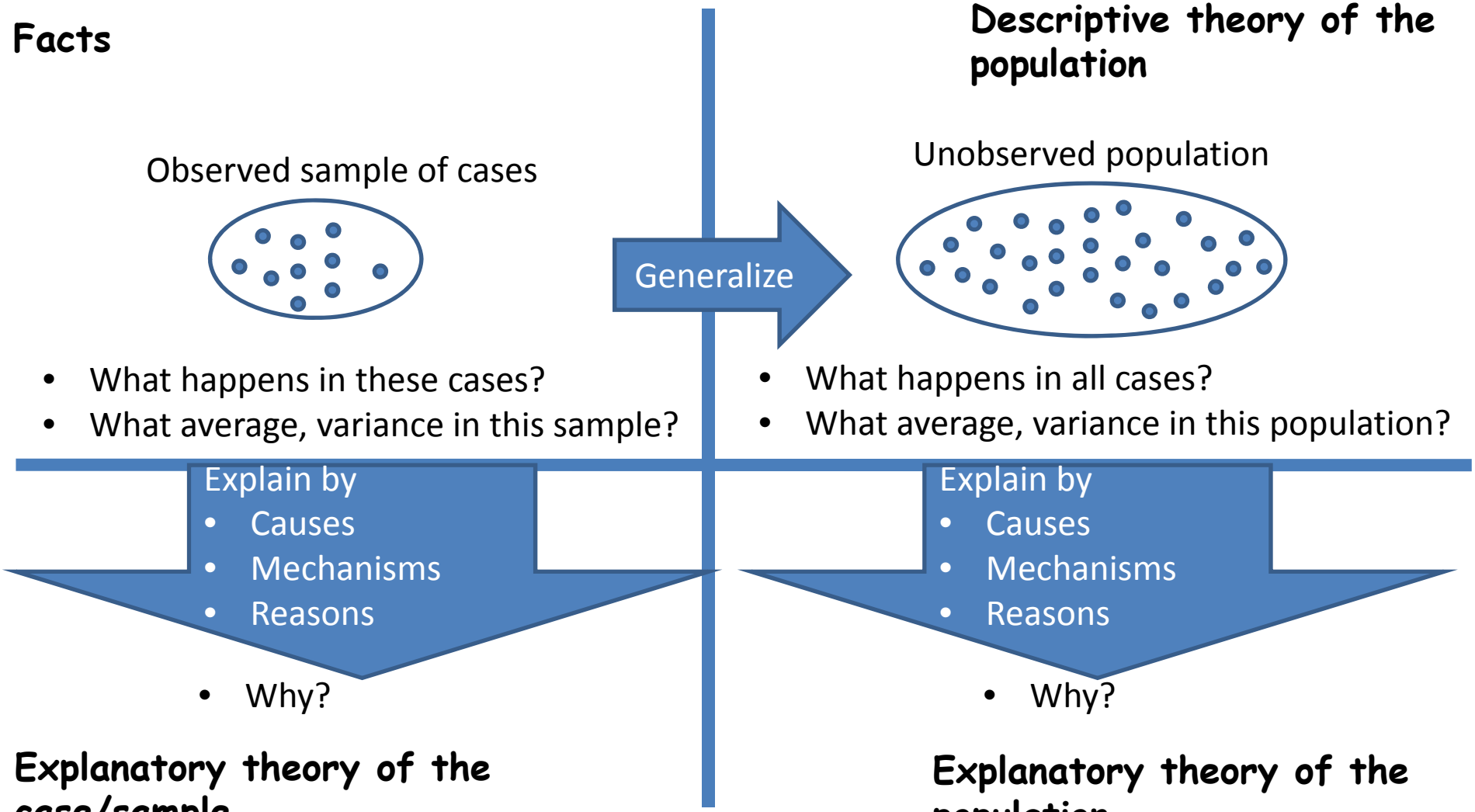
- Why?

Explanatory theory of the case/sample

Explanatory theory of the population

© R.J. Wieringa

Three kinds of explanation



© R.J. Wieringa

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*

© R.J. Wieringa

Keywords in the three kinds of explanations

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

© R.J. Wieringa

One more example

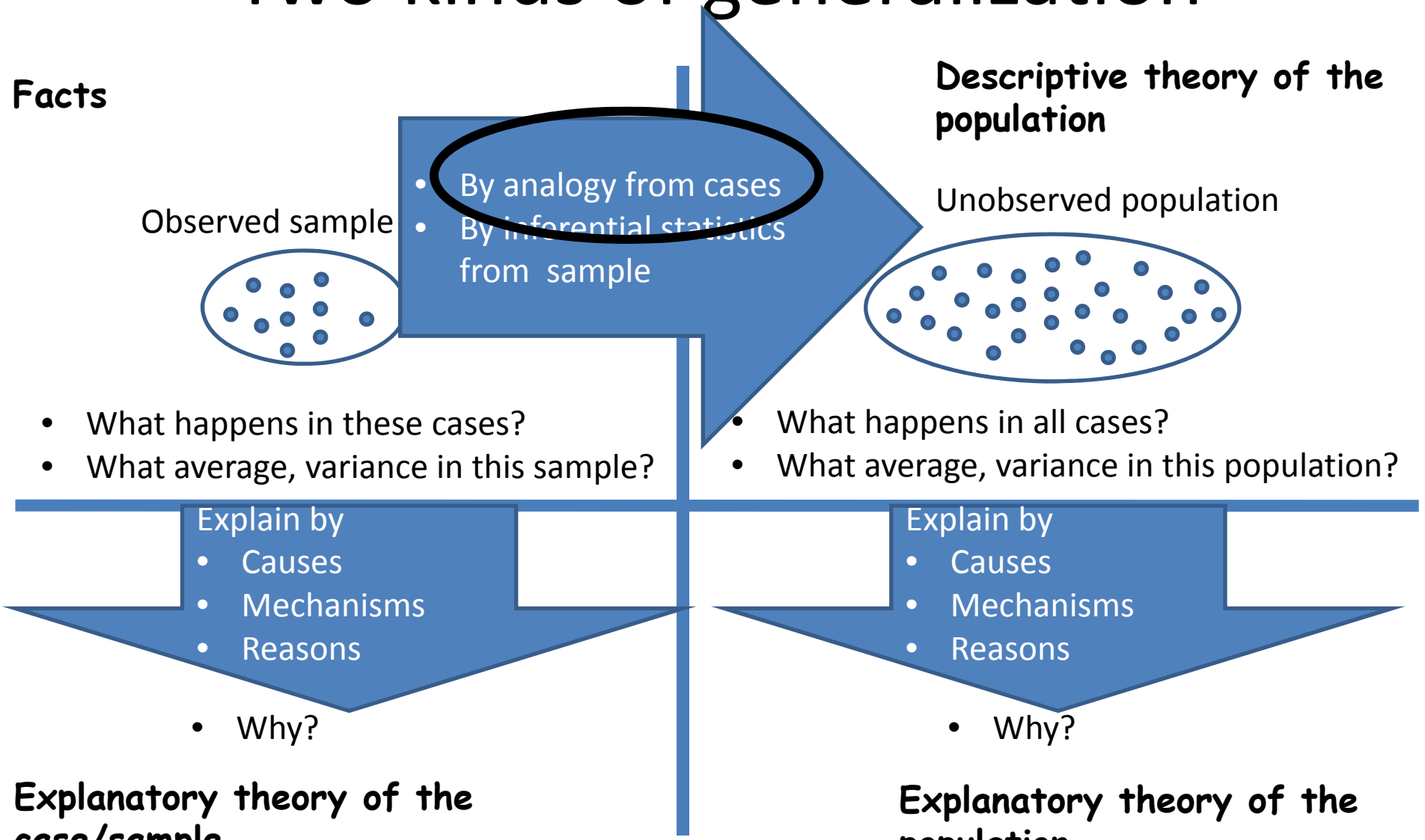
- **Causal explanation:** effect attributed to a cause. Explain difference in outcomes by difference in interventions. Causation is difference-making.
  - *The coffee made me stay awake late.*
- **Architectural explanation:** Outcome produced by interaction among components. Explain capability of system 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 contributes 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 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?

© R.J. Wieringa

Two kinds of generalization



© R.J. Wieringa

Generalization by analogy

- *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.*
- *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.

Generalization by analogy

- *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*
  - *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*

Generalization by analogy

- Must be based on architectural similarity
  - Similar components, with similar capabilities
  - Similar mechanisms involving these components

- Analogy based in similarirty of superficial features, without knowledge of underlying mechanisms, is too weak a basis for generalization.

Example of an unsound analogic generalization

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

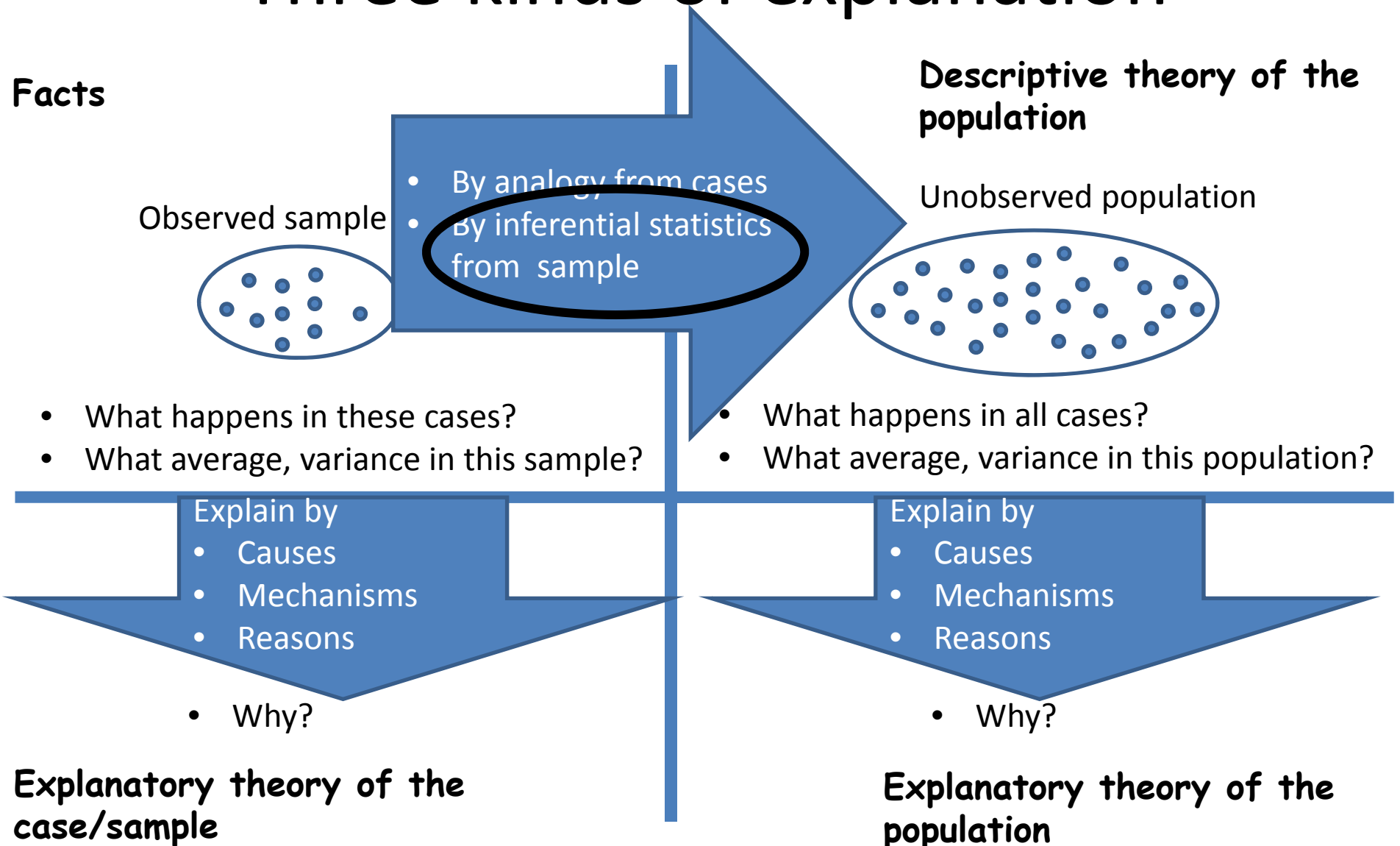
- This is only superficial similarity
- There is no mechanism that produces thinking in brains and wallnuts!

© R.J. Wieringa

External validity

- Degree of support for generalization by analogy
- Threats that decrease support:
  - Cases that look superficially similar may not be architecturally similar.
  - Analogic generalization is not universal: it may be falsified by interfering mechanisms.
- Mitigate this by analytic induction: Cases are studied one by one, theory updated 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

Three kinds of explanation



© R.J. Wieringa

Generalization by statistical 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*

Example continued: Further generalization by analogy

- *We may want to **generalize by analogy** to similar populations, e.g. the population of all OS software, or of all Web Applications.*
  - *This would be justified if the mechanisms and reasons that produced the phenomenon in the study population are the same in these other populations.*
  - *But we do not really know what these mechanisms are….*

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

© R.J. Wieringa

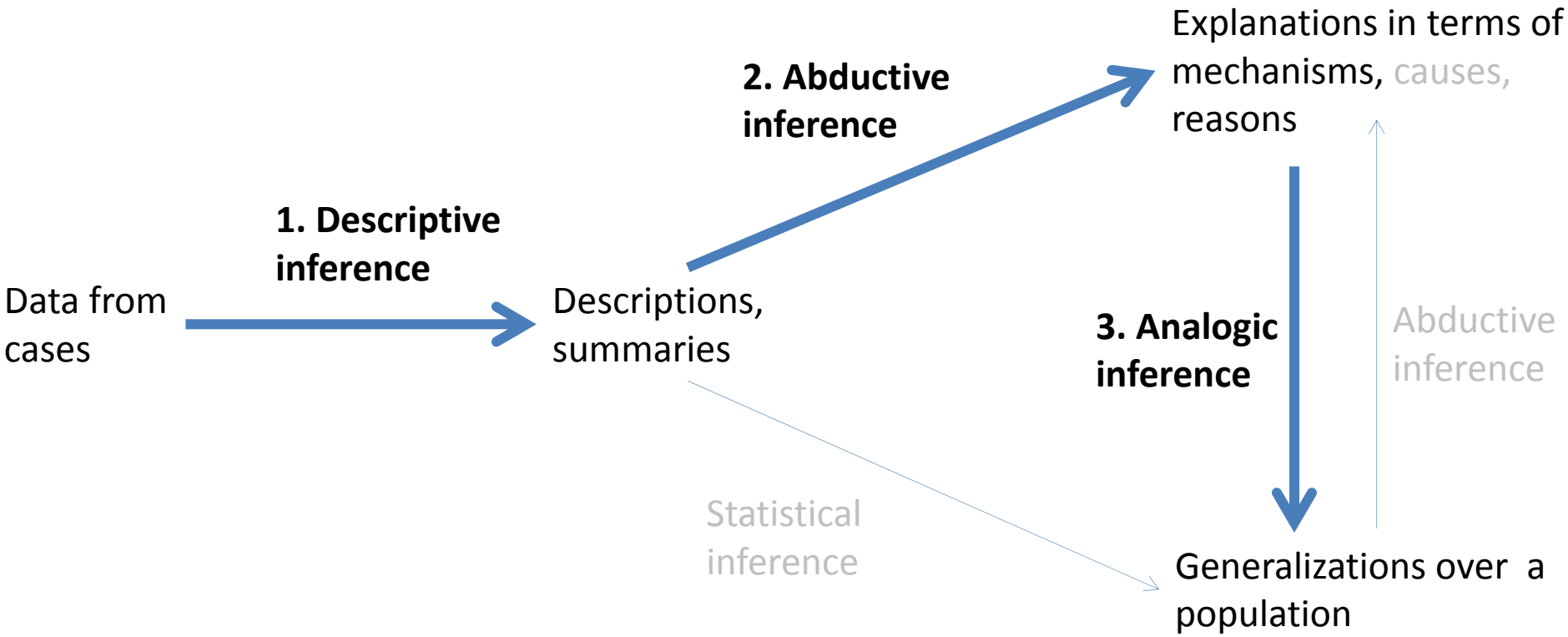
Big data

- If the sample is almost the size of the population, and the unobserved part of the population is **similar** to the observed part, statistical inference may be omitted.
  - descriptive statistics of the sample may allow us to generalize to the population based on similarity.
  - This is often called statistical learning (e.g. regression or classification)

- *Based on an analysis of data about 95% 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*

How to use all of these kinds of arguments in your research?

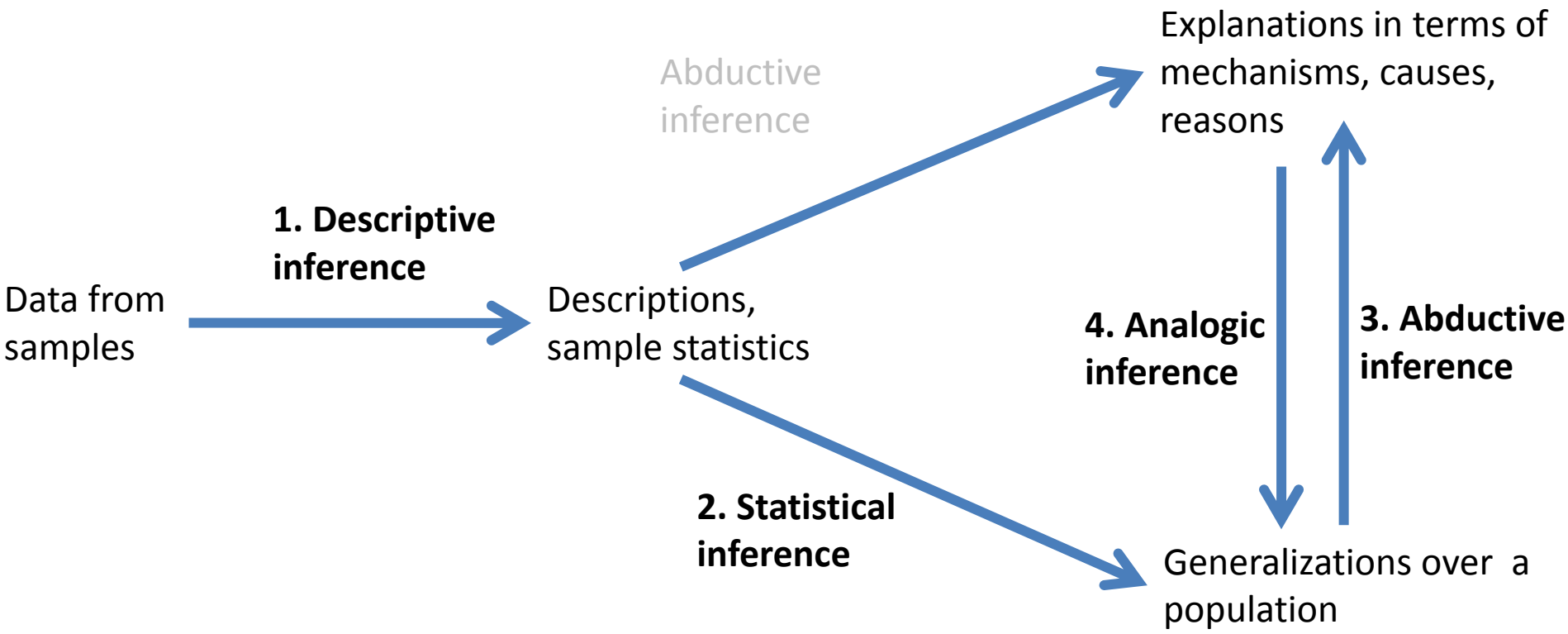
Case-based inference



- Analogic inference to similar cases must be based on architectural explanations (in terms of mechanisms or reasons)
- Even if the sample is very big!

© R.J. Wieringa

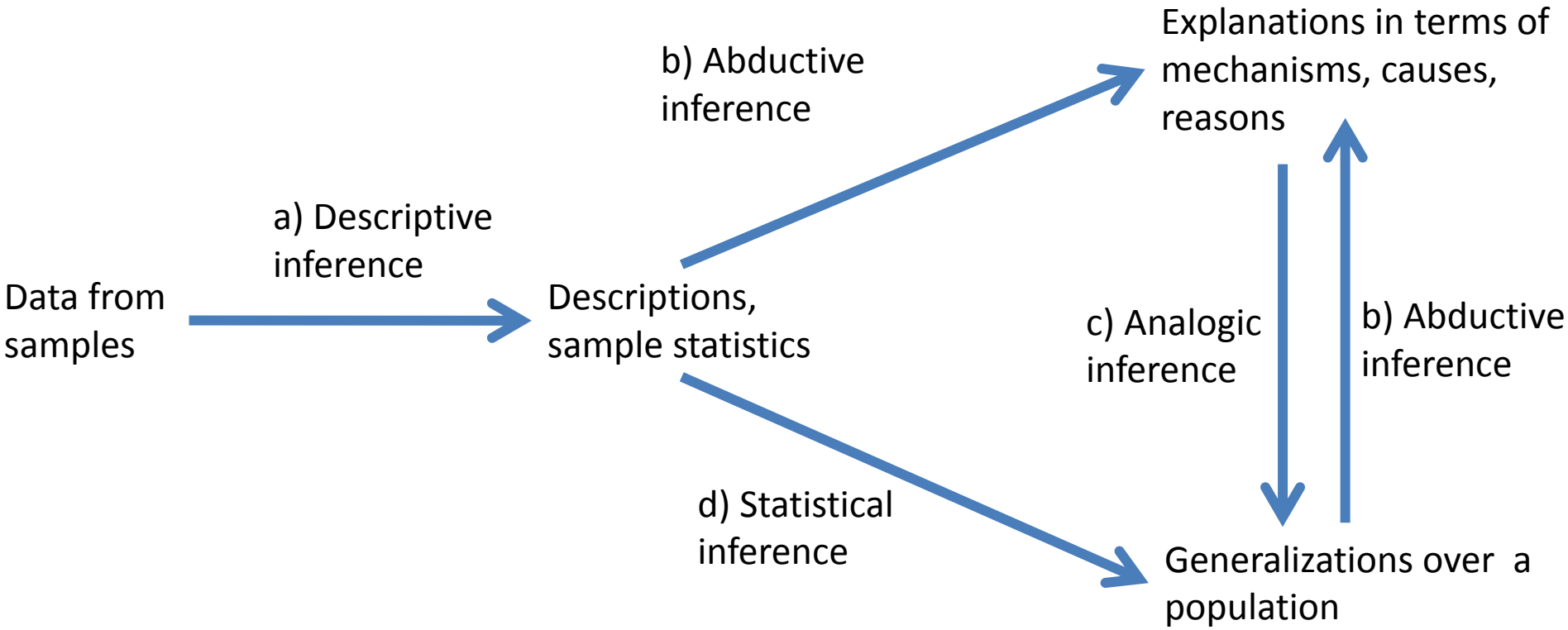
Sample-based inference



- 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.

© R.J. Wieringa

Validity of inferences



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

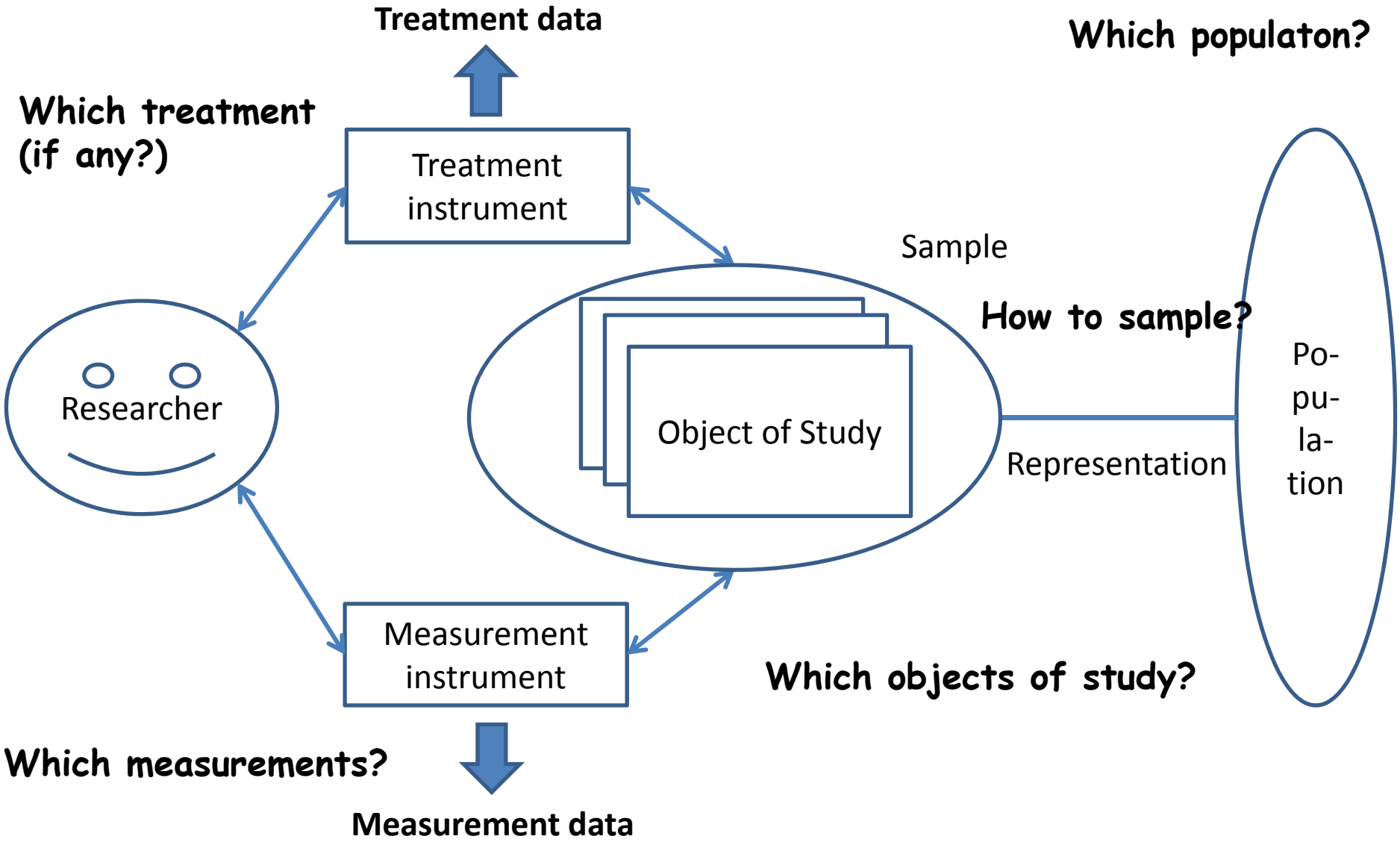
© R.J. Wieringa

Outline

1. What is design science
   - Research goals and problems
   - The design and engineering cycles
2. Theories
   - Scientific inference
   - **Research design**

How can you set up our empirical research so that you can support the desired inferences

Design decisions for research setup



© R.J. Wieringa

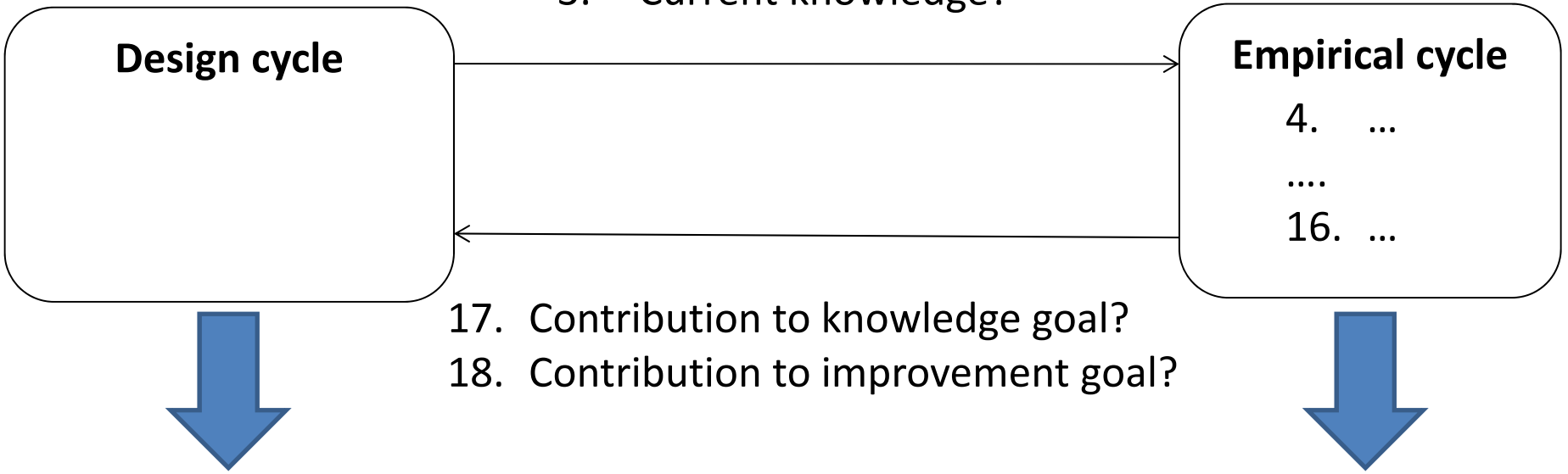
Research designs

	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)

© R.J. Wieringa

Checklist for the empirical cycle: context

1. Improvement goal?
2. Knowledge goal?
3. Current knowledge?



© R.J. Wieringa

Data analysis

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

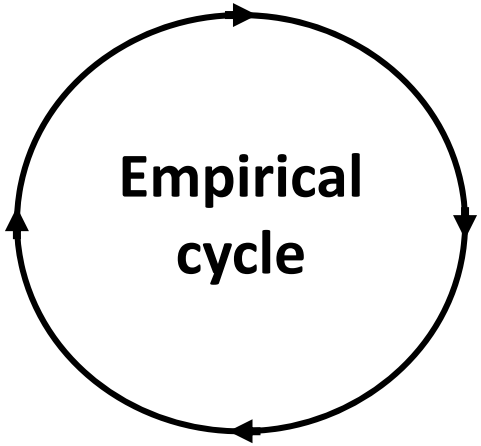
This is a checklist for

- **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	**Research & inference design**	
7. Object of study validity?	7. Object of study?	Research setup
8. Treatment specification validity?	8. Treatment specification?	
9. Measurement specification validity?	9. Measurement specification?	
10. Inference validity?	10. Inference?	Inference

© R.J. Wieringa

Validity of research design: Does research setup match inference?

Observational study may confirm or falsify generalizations

Experimental studies may support or falsify causal explanations of the differences betyween treatment and non-treatment

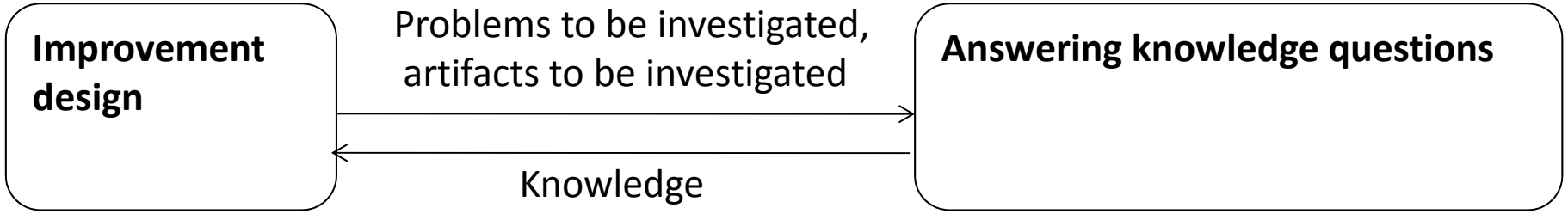
Case-based research may support architectural explanations in terms of mechanisms or reasons

Sample-based research may support descriptive generalizations about a population

	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)

© R.J. Wieringa

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

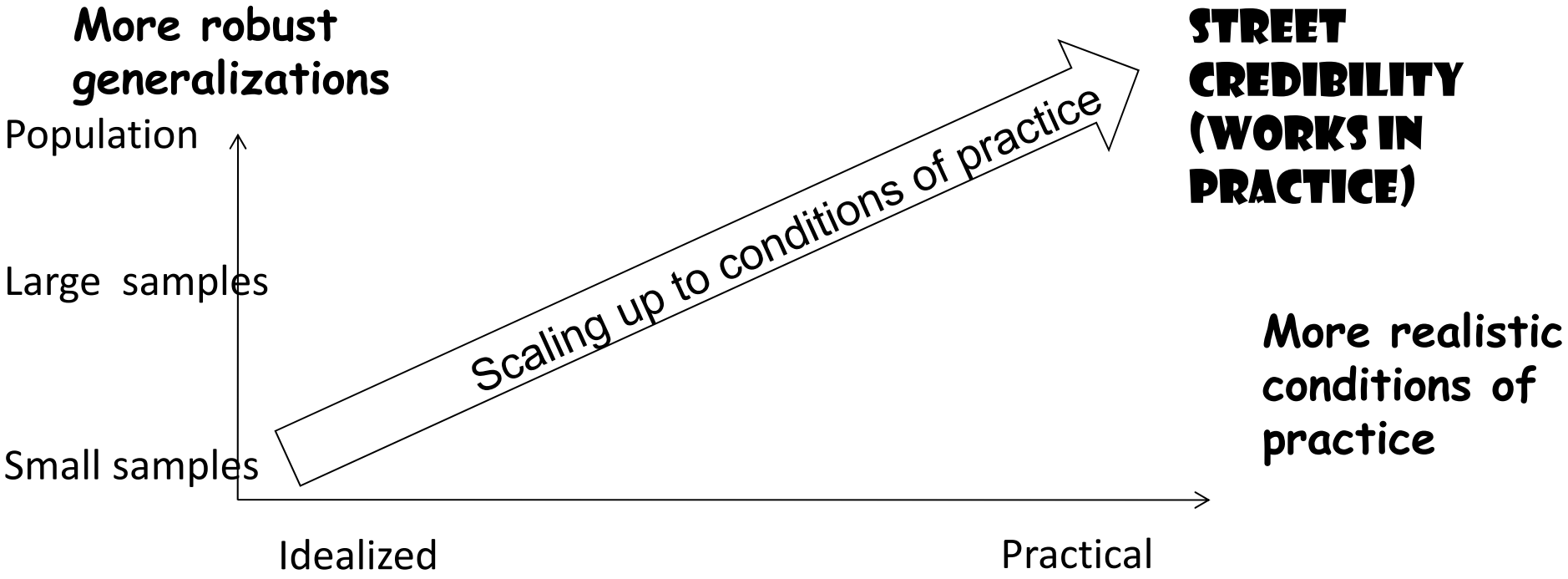
- Research problem analysis
- Research design & validation
- Research execution
- Data analysis

Theories → Empirical cycle → Theories

Analytical knowledge questions

© R.J. Wieringa

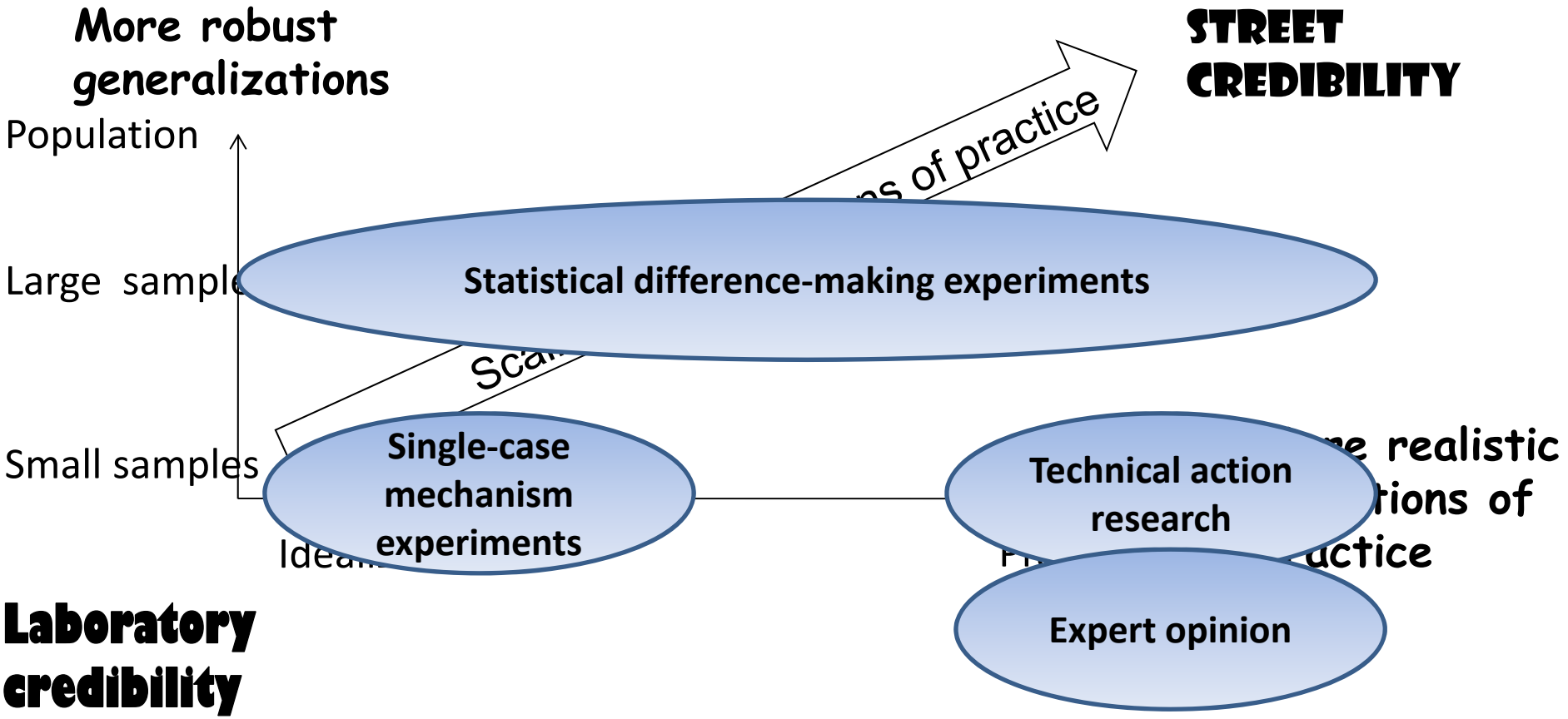
When to use these methods in design science research?



Laboratory credibility (works in theory)

- Just like New Drug Research

© R.J. Wieringa



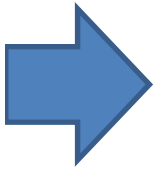
- Scaling up:
  - Single-case mechanism experiment (laboratory simulation)
  - Expert opinion
  - Single-case mechanism experiment (field simulation)
  - TAR (apply technique in a real-world project)

© R.J. Wieringa

Take-home

- Design science
  - Design problems with the design cycle
  - Empirical knowledge questions with the empirical cycle
- Design theories
  - Effects of artifact in context
- Validation research methods
  - Case-based and sample-based research
  - Architectural inference from cases (mechanisms, reasons)
  - Statistical inference from samples (averages, variance)
  - Causal reasoning from experiments
- Scaling up from lab to practice

- 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.