Designing technical action research, and generalizing from single cases

Roel Wieringa

University of Twente

The Netherlands

1. What is TAR?
2. Logical structure of TAR
3. Generalizing from TAR
4. Summary

1. Wat is Technical Action Research?

What is Technical Action Research?

- Example
  - Researcher develops a technique to assess confidentiality risks in an IT architecture
  - She applies it to a problem that a company has ...
  - producing an advice to the company ...
  - and drawing lessons learned about the method
- She served two goals:
  - The company's goal is to assess confidentiality risks
  - The researcher's goal is to learn something about her method

What is Technical Action Research?

- The researcher plays three roles:
  - **Designer:** Designing a technique
  - **Helper:** Using the technique to help others
  - **Researcher:** Drawing lessons learned about technique

- The key to a proper methodology for TAR is keeping these roles separate

Contrast with observational study

- Example:
  - Researcher observes one or more agile projects to investigate how requirements are prioritized
  - Avoids influencing the projects
  - Observes, analyzes, concludes lessons learned
- No change goal: The company is not influenced
- Researcher’s goal is to learn about prioritization in agile projects as it is currently happening
- (the resulting knowledge **may** be useful to the companies)

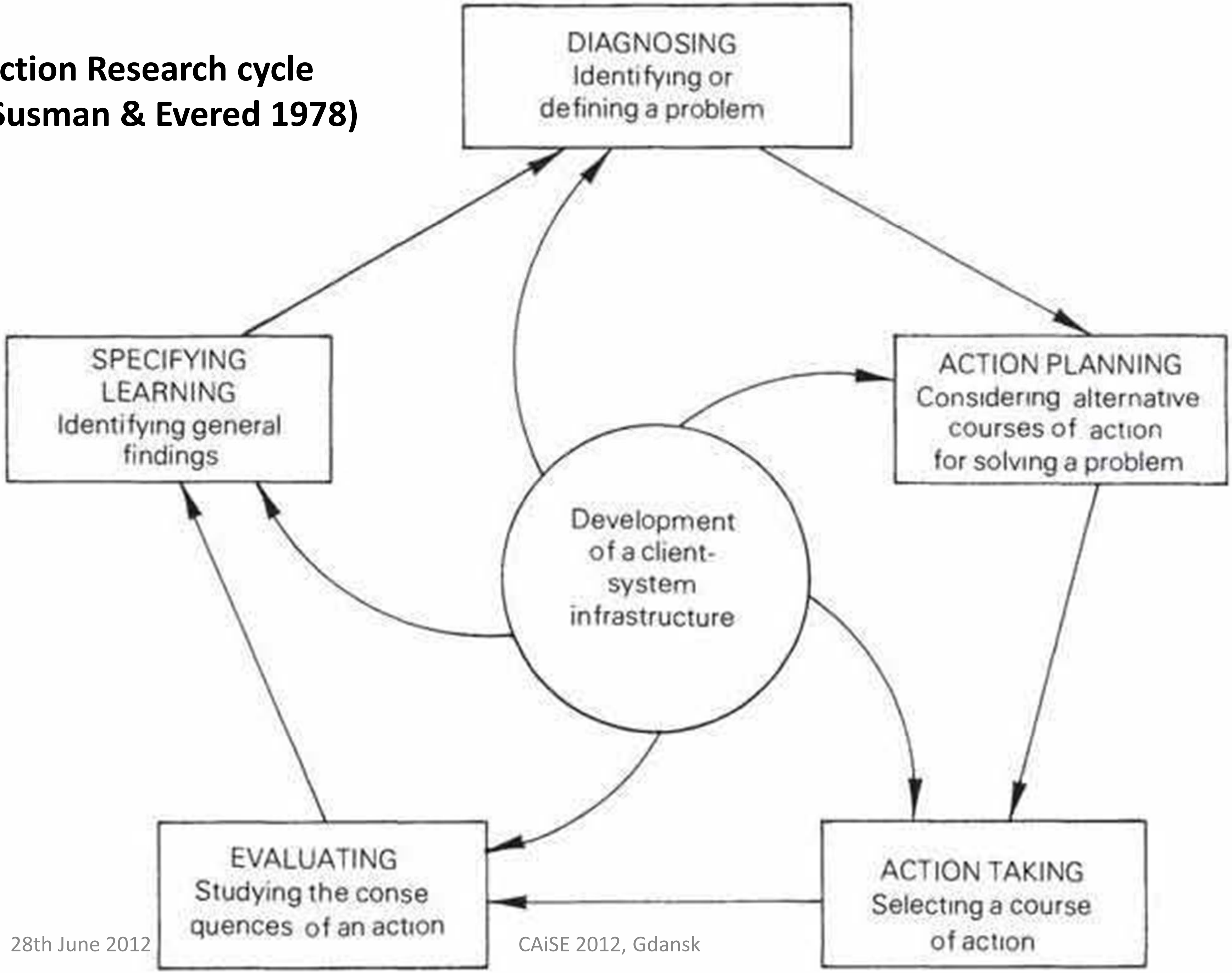
Contrast with consulting

- Consulting
  - Consultant is paid by client
  - Consultant applies known techniques rather than experimental technique
  - Reusable techniques rather than critical evaluation
  - Aims at helping the client and acquiring repeat business, rather than testing a technique
  - Knowledge dissemination (if any) is internal

Contrast with "classical" action research

- In classsical AR, researcher helps client to identify and solve a problem
  - Emancipation of the powerless
  - Learning about their situation
- In TAR, the researcher wants to learn something about a technique by using it to solve a client's problem

Action Research cycle
(Susman & Evered 1978)



Contrast with AR in information systems

- AR in information systems
  - Identify problem in an organization
  - Jointly search for a solution
  - Implement it
  - Evaluate
  - Specify learning
- TAR is technology-driven, not problem driven
  - The technology may be motivated by a desire to solve a class of problems
  - Not a singlular problem

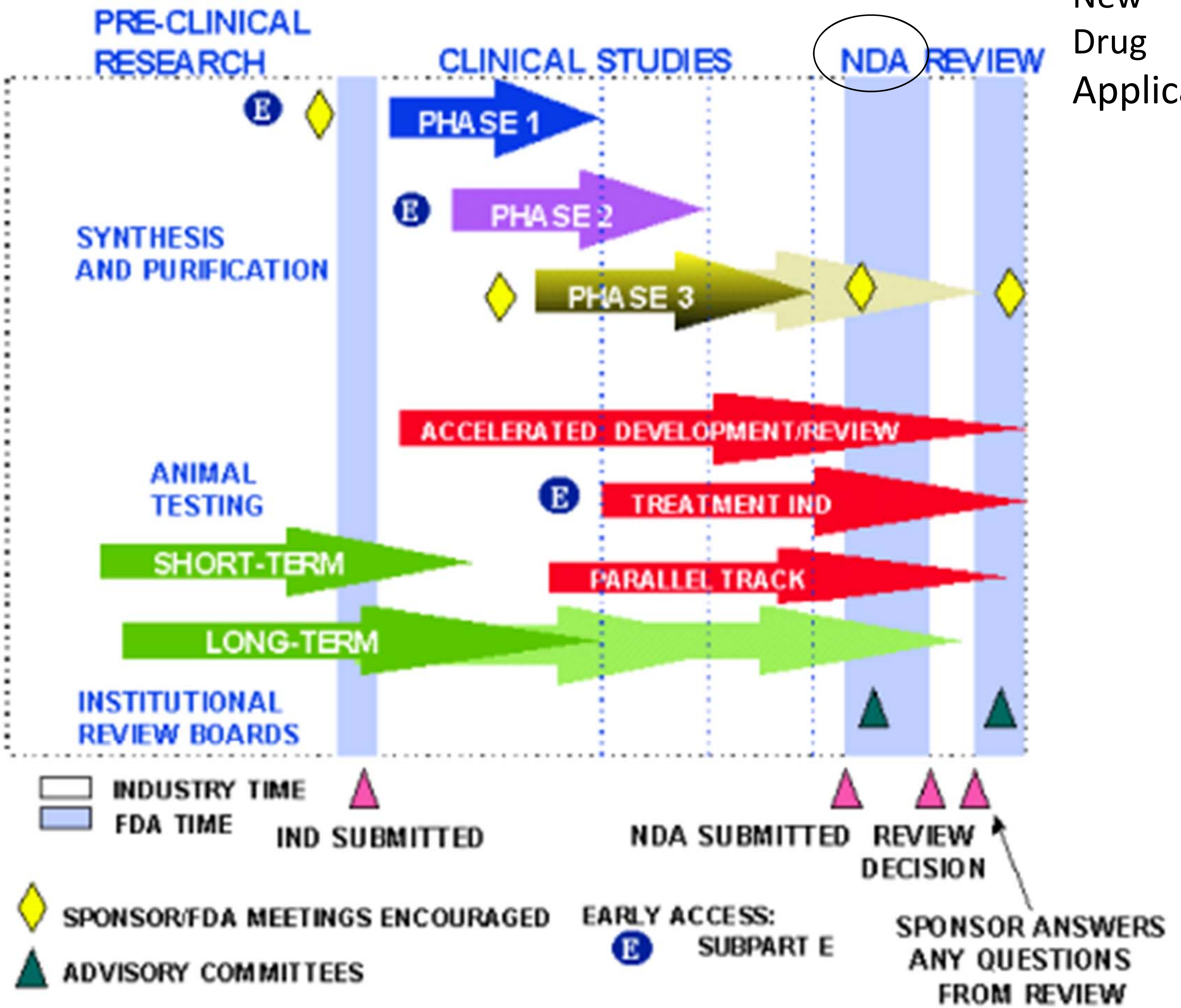
Why TAR for the client

- Risky project with large chance of non-result
- What is in it for the client?
  - Free consult
  - Potentially useful result
  - Advance knowledge of and experience with new techniques
  - Good relationships with university (PR, HRM)

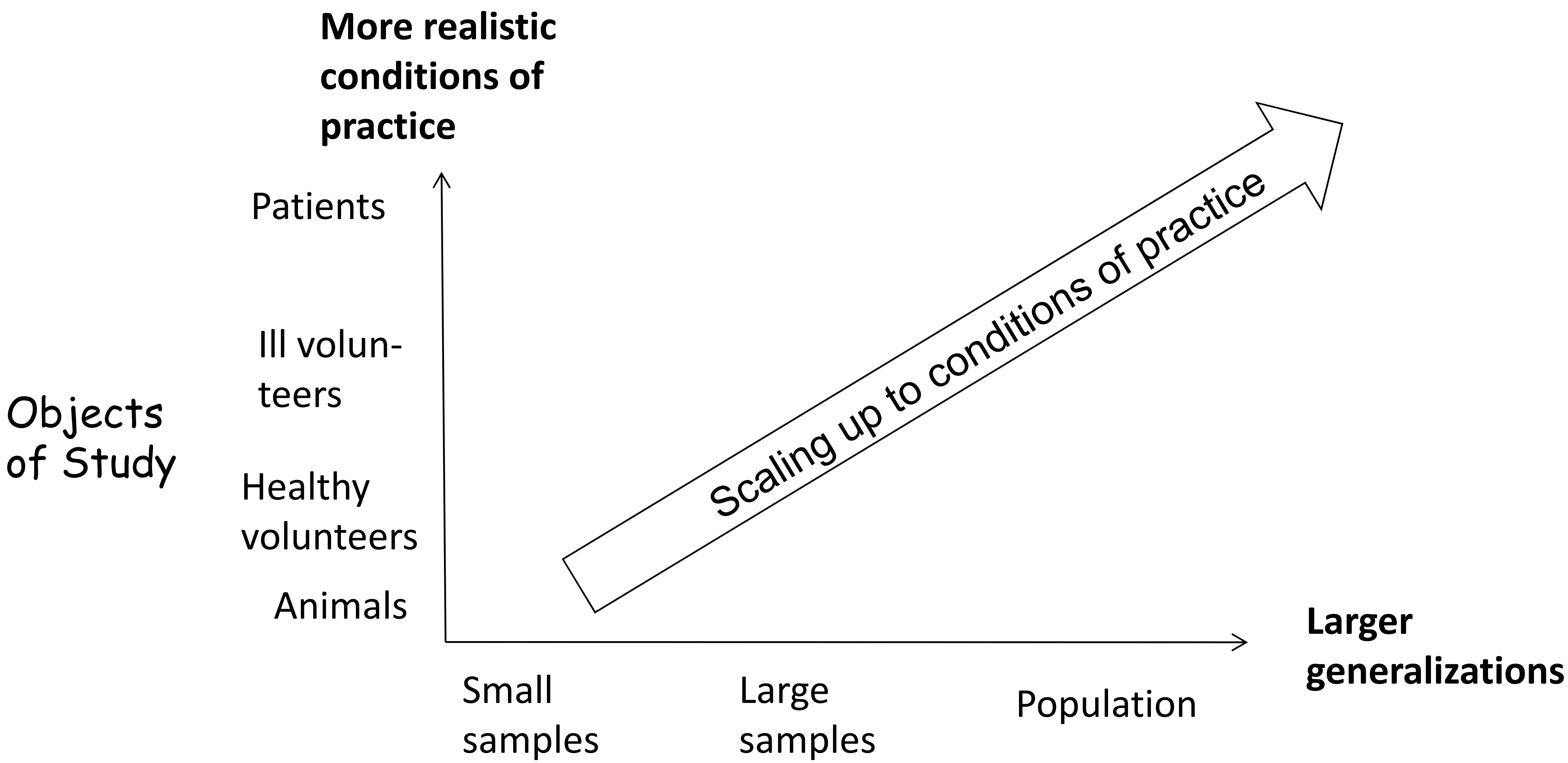
Why TAR for the researcher

- Researcher developed a technique behind her desk
- Applied it to first to small and then to realistic examples
- Compared with other proposals
- Then what?
  - Students will do as teacher tells: no realistic validation
  - Best way to learn about the technique is to apply it yourself
- Important to scale up from desk to practice

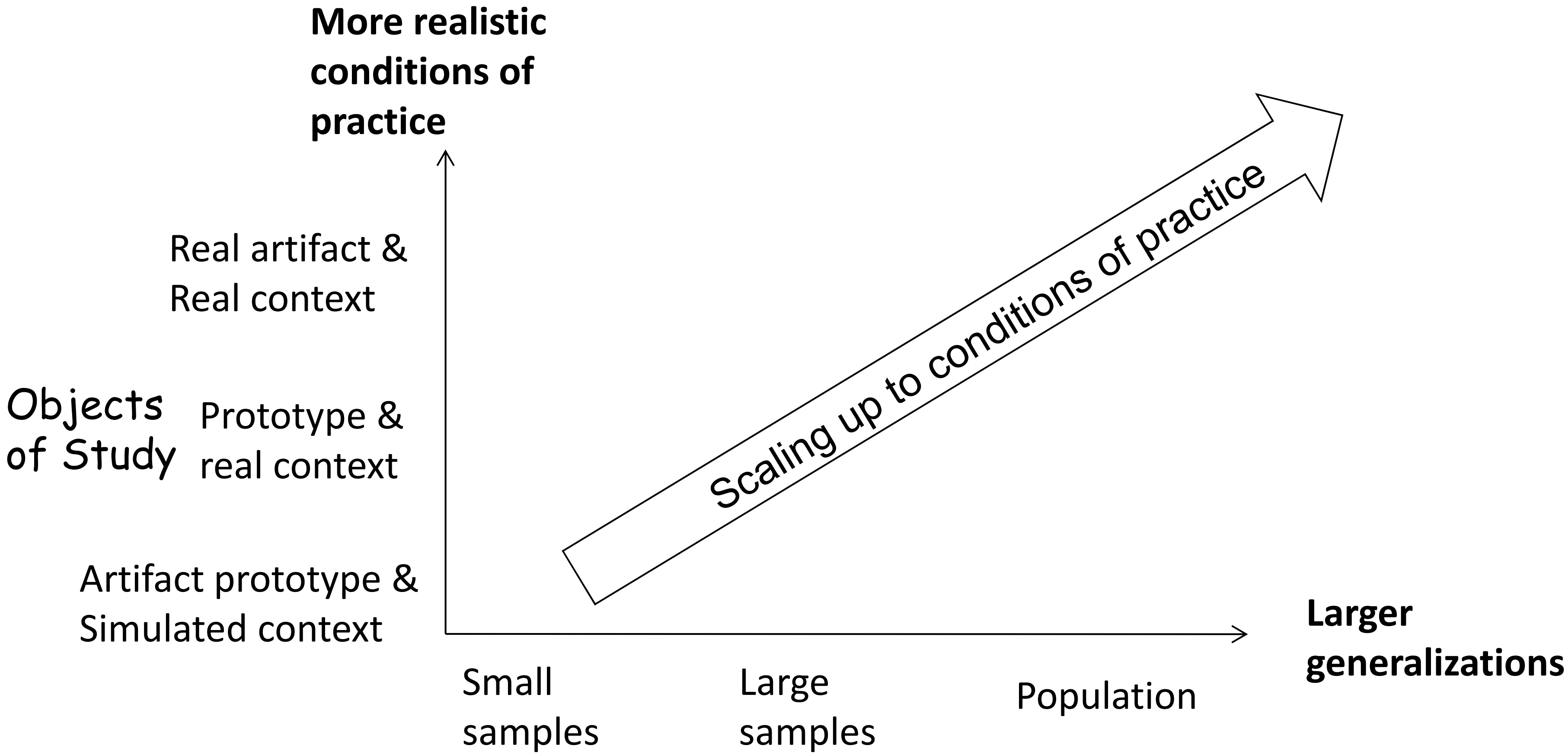
2. Scaling up to practice



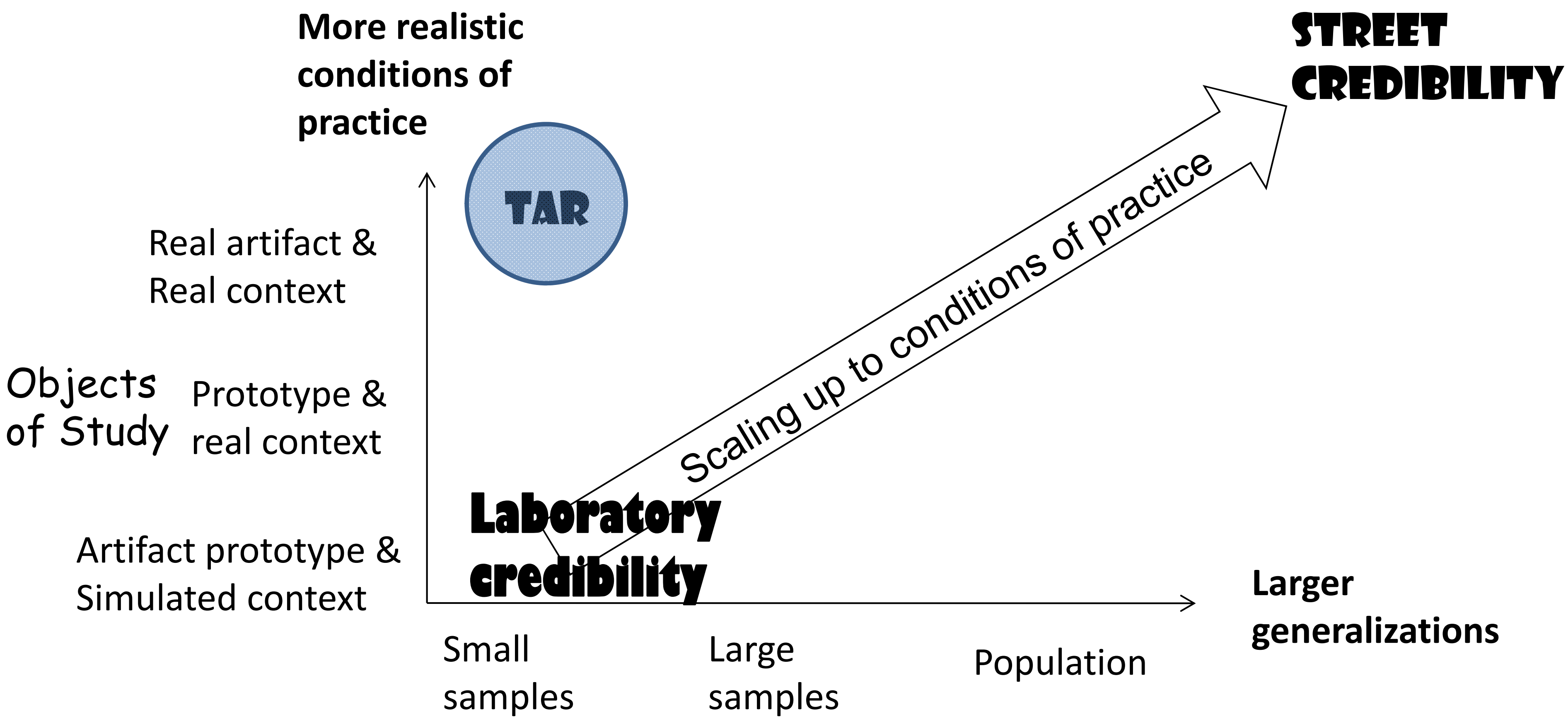
New
Drug
Application



- Animals, healthy volunteers, and ill volunteers are used as **models** of arbitrary patients
- Conclusions about the models are generalized to arbitrary patients



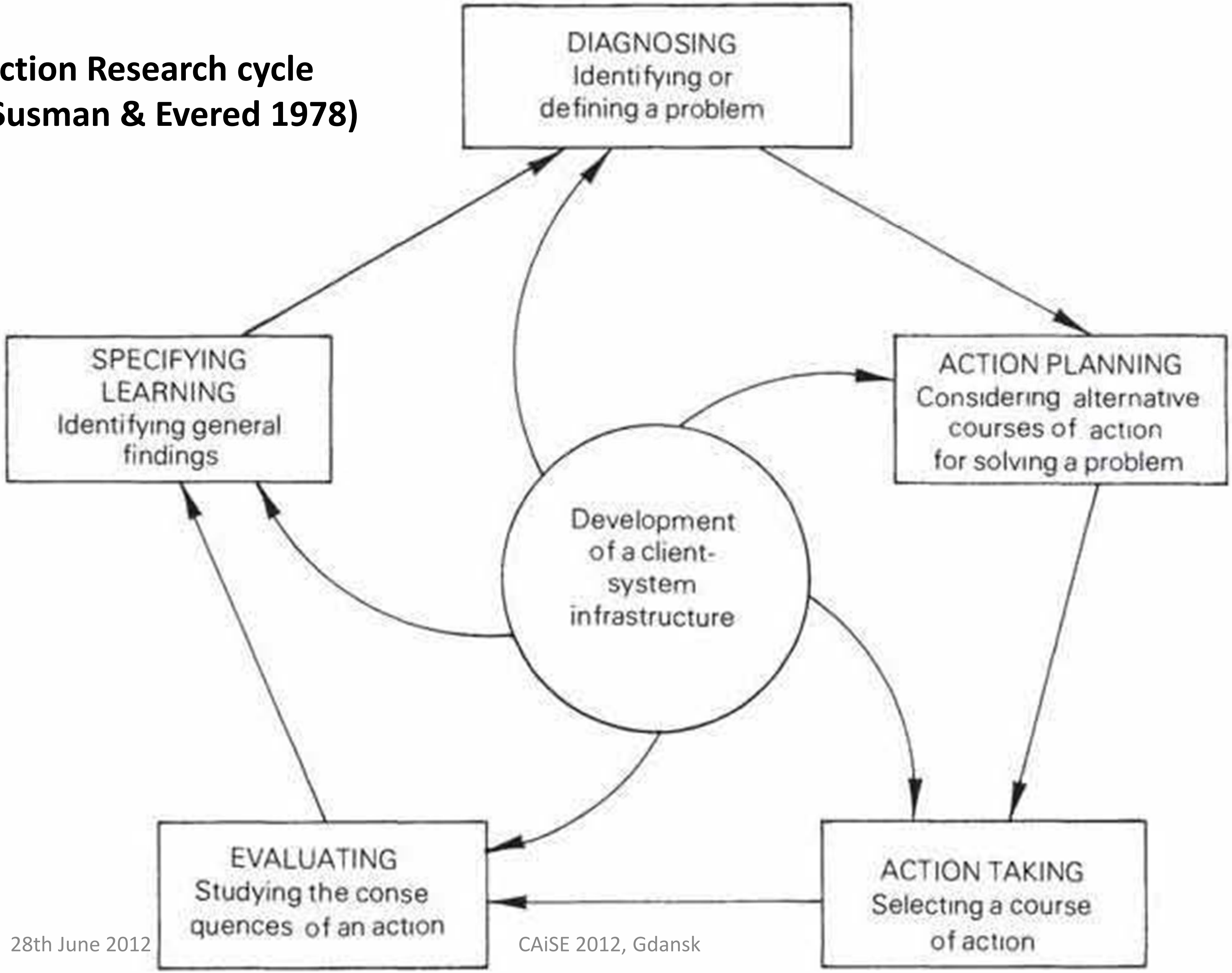
- Start with testing of prototype in the lab
- End up with using the artifact in practice
- Start with small samples of comparison, end up with large



- From: “It works in theory” before simulation
  - To “It works in the lab” ….
    - … via increasingly realistic simulations …
      - To “It works in practice”

3. Logical structure of TAR

Action Research cycle (Susman & Evered 1978)



- This conflates **two** action cycles:
  - Action cycle of client
  - Action cycle of researcher
- Each has a different goal and justification

The engineering cycle

- The logical structure of a rational action is that of the engineering cycle
  - Problem investigation
  - Treatment design
  - Design validation
  - Treatment implementation
  - Implementation evaluation

The rationality of the engineer

- Separating solutions ("treatments") from problems
  - Don't define the problem as absence of (your) solution
- Acknowledging that there are many solutions
  - Your view is not the only one
- Specifying your action before you act
  - Think before you act
- Justifying your choice of action before you act
  - Comparison, trade-offs
- Evaluating your action after you act
  - You could have been wrong ...
  - Learn from the effects of your action

- Problem investigation → Stakeholders, goals, Phenomena, diagnosis, evaluation
- Treatment design
- Design validation
- Treatment implementation
- Implementation evaluation

Treatment =
interaction between
artifact and context.

- Problem investigation
- Treatment design → Requirements? Contribution to goals? Available treatments? Design a treatment.
- Design validation
- Treatment implementation
- Implementation evaluation

Requirements?
Contribution to goals?
Available treatments?
Design a treatment.

•Interaction between pill and patient
•Interaction between Software and its Context
•Interaction between method and its context of use

- Problem investigation
- Treatment design
- Design validation
- Treatment implementation
- Implementation evaluation

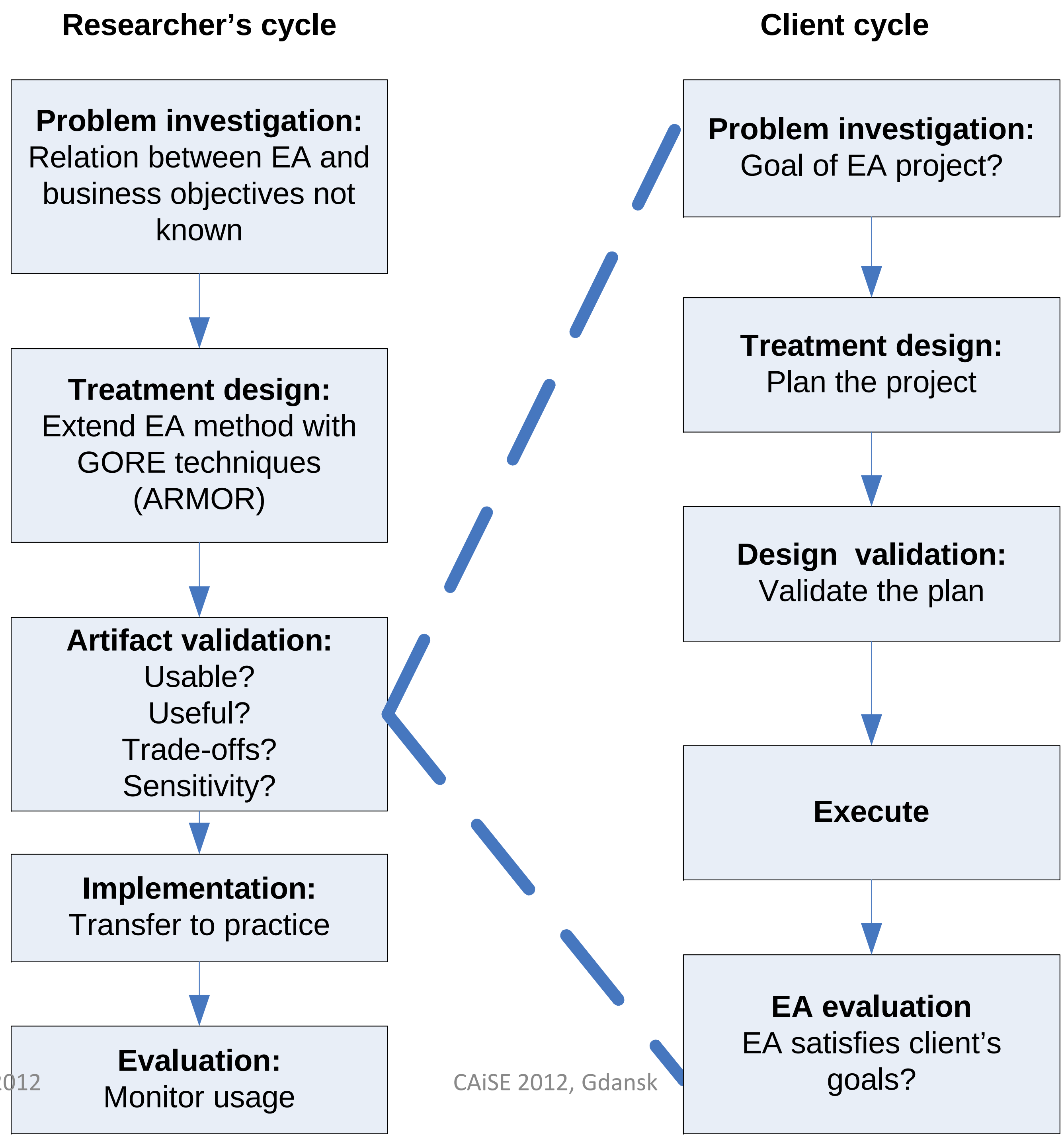
Artifact & Context → Effects?
Trade-off: Changes in artifact
Sensitivity: Changes in context
Effects satisfy Requirements?

- Problem investigation
- Treatment design
- Design validation
- Treatment implementation → Transfer to practice!
- Implementation evaluation

- Problem investigation
- Treatment design
- Design validation
- Treatment implementation
- Implementation evaluation →

Stakeholders, goals, requirements?
Phenomena: Artifact & Context → Effects?
Evaluation: Effects satisfy Requirements?

- Example: Extending an enterprise architecture (EA) method with goal-oriented requirements engineering (GORE) manage links to business goals


Researcher's cycle
Client cycle
Problem investigation:
Relation between EA and business objectives not known
Treatment design:
Extend EA method with GORE techniques (ARMOR)
Artifact validation:
Usable?
Useful?
Trade-offs?
Sensitivity?
Implementation:
Transfer to practice
Evaluation:
Monitor usage
Problem investigation:
Goal of EA project?
Treatment design:
Plan the project
Design validation:
Validate the plan
Execute
EA evaluation
EA satisfies client's goals?

- Two goals
  - The client evaluates its redesigned EA against its goals
  - The researcher validates ARMOR against **his** goal

- Three roles for the researcher
  - Designing a technique
  - Using it to help a client
  - Learning from it
    - How do we use the client cycle to answer these validation questions?

The empirical research cycle

- This is the engineering cycle applied to one specific goal: Answering knowledge questions
  - Knowledge problem investigation
  - Research design
  - Design validation
  - Research execution
  - Results evaluation

The investigator’s rationality

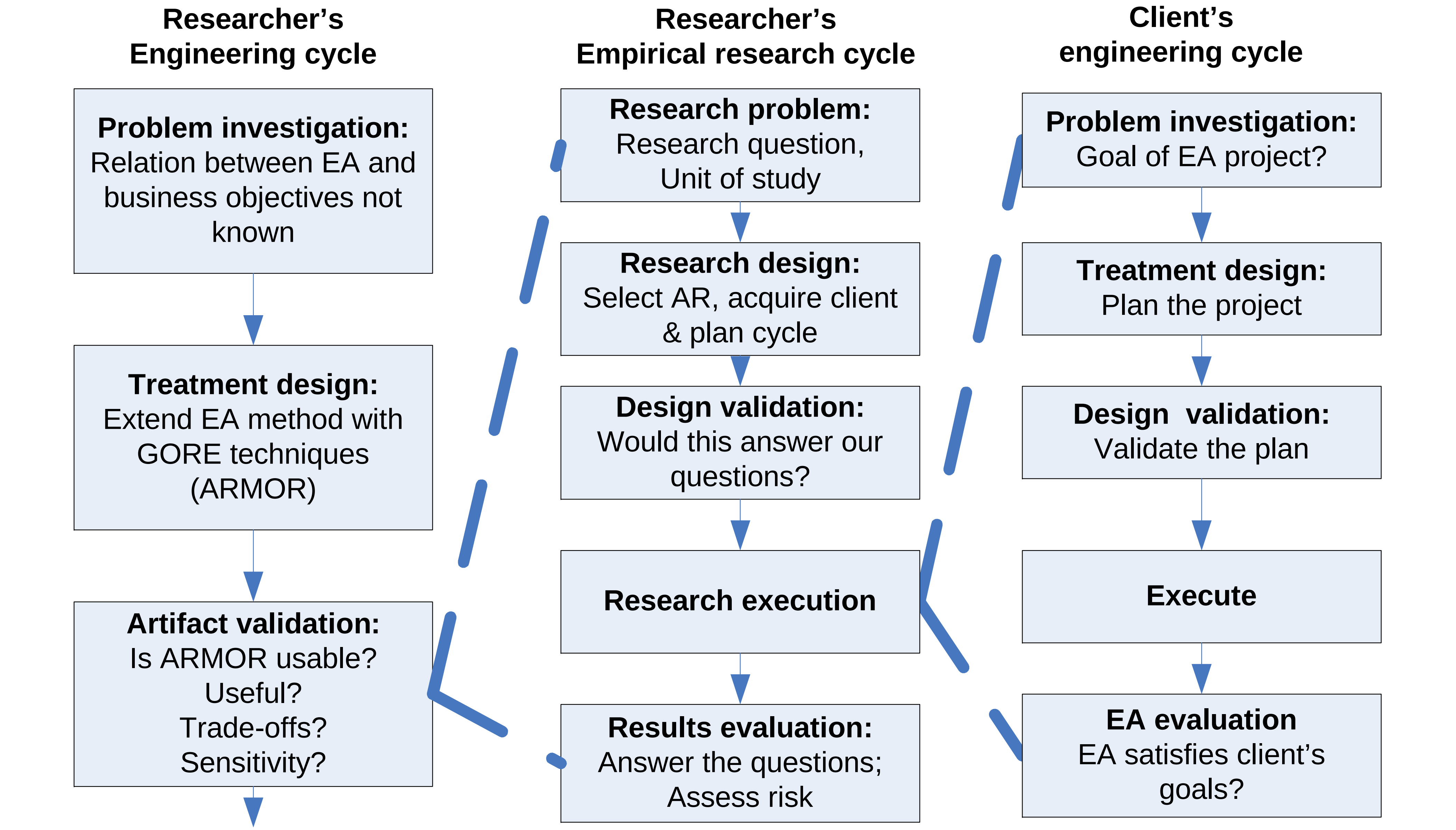
- Adopted from the engineer
- Applied to knowledge acquisition
  - Ask your questions before answering them
  - Do something (i.e. confront with reality) when answering them
  - Be honest about your uncertainty (“in which ways could I be wrong?”)
  - Justify your answers

- Knowledge problem investigation → Research questions, Unit of study
- Research design
- Design validation
- Research execution
- Results evaluation

- Knowledge problem investigation
- Research design → Survey, observational case, Experiment, Action case, Simulation, …
- Design validation
- Research execution
- Results evaluation

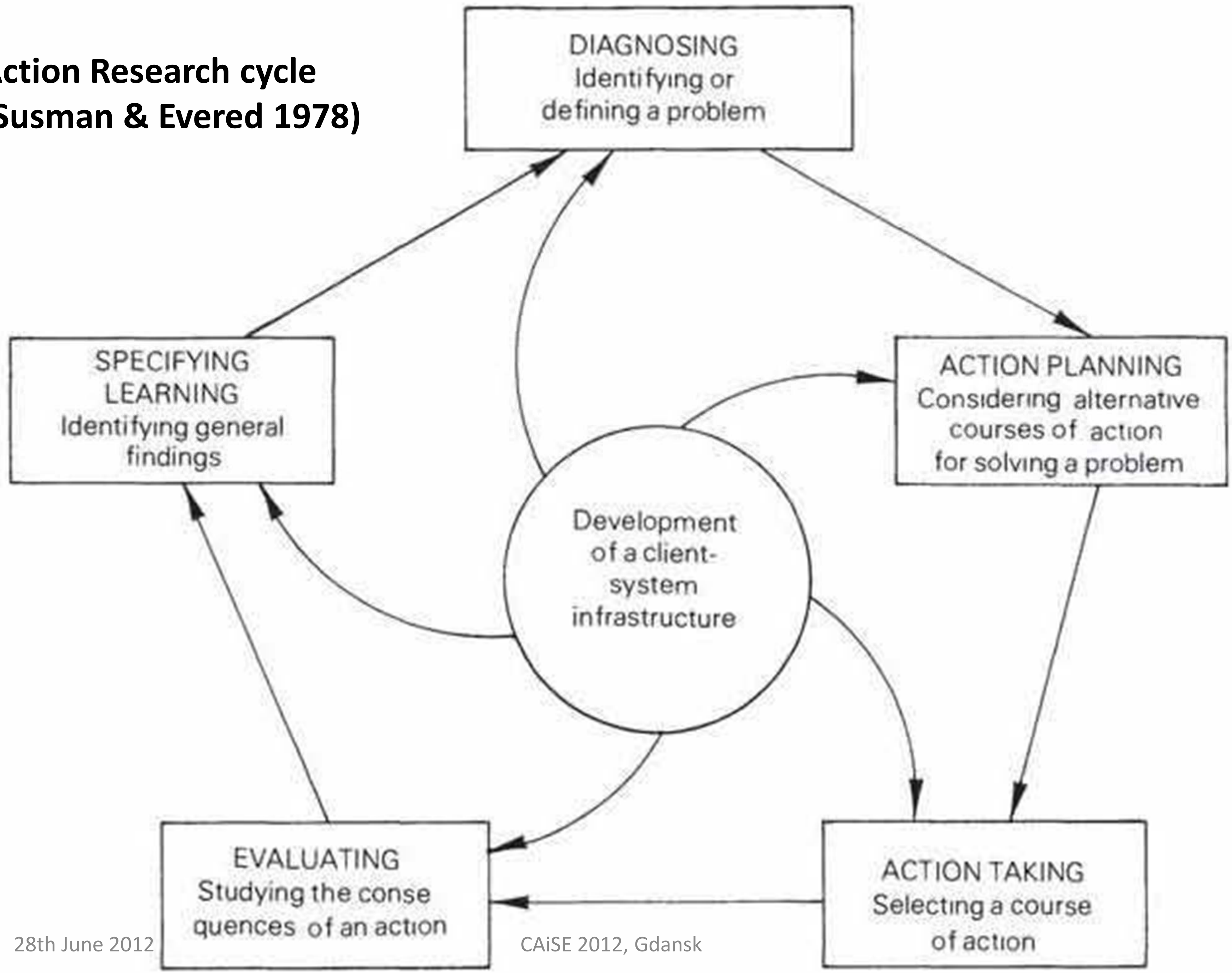
- Knowledge problem investigation
- Research design
- Design validation → Would this really answer our questions? Risk assessment of doing the wrong thing to answer the questions
- Research execution
- Results evaluation

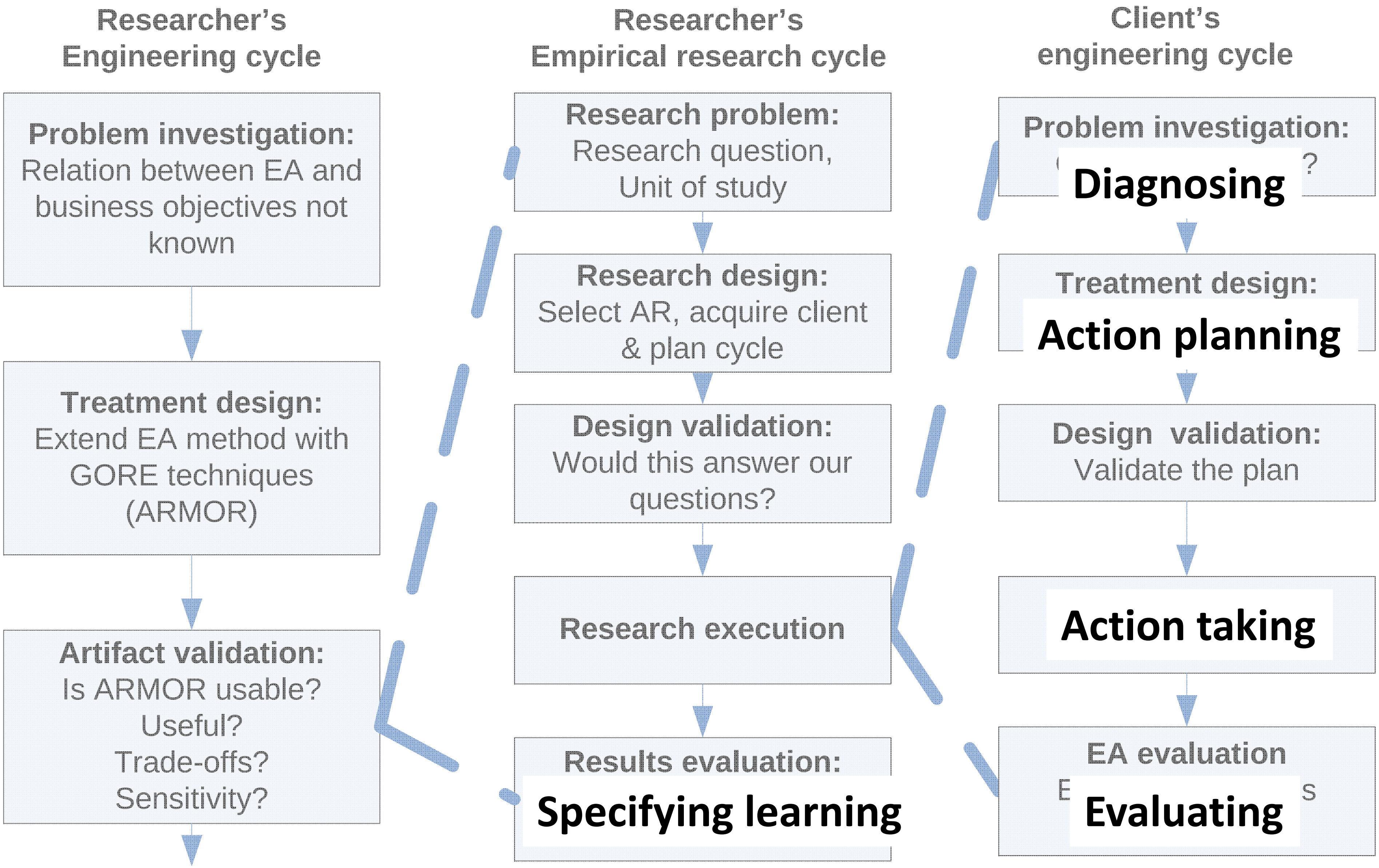
- Knowledge problem investigation
- Research design
- Design validation
- Research execution
- Results evaluation → Did this really answer our questions?
  Risk assessment of answering the questions incorrectly



Corresponds to the three roles of the researcher:
Designer, researcher, helper

Action Research cycle (Susman & Evered 1978)





Now we can see what is ignored in classical AR

Practical problem:
Specify confidentiality control requirements of an outsourcing client in an SLA.

Problem investigation
(Section I) Stakeholders, Goals, Problems, Diagnosis, Criteria C0-C6, Existing solutions

Treatment design
(Section III) CRAC++ = CRAC + confidentiality requirements specification

Treatment validation
Q1 Would this work if implemented?
Q2 Trade-offs?
Q3 Sensitivity?

Treatment implementation
Transfer CRAC++ to practice

Implementation evaluation
Evaluate practical experience with CRAC++

Research question:
Is the proposed method valid?

Research question investigation
(RQ1) Does CRAC++ satisfy criteria?
(RQ2) How does CRAC++ compare to alternative treatments?
(RQ3) In which contexts is CRAC++ usable?

Research design
Acquire a case

Validate the research design
(Section VIII)
- Internal validity
- External validity

Execute the research

Analyze results
Answers to research questions? Explanations?
(*Section VII-A*) RQ1: Goal achievement
(*Section VII-B*) RQ2: Comparison
(*Section VII-C*) RQ3: Generalizability
(*Section VIII*) Validity of answers?

Practical problem:
Specify confidentiality requirements of X in a particular outsourcing relation.

Problem investigation
(Section IV)
- stakeholders involved,
- organization architecture,
- IT architecture
- goals/problems of the stakeholders,
- criteria to measure goal-achievement

Treatment design
Agree on a treatment plan

Treatment validation
Would this achieve stakeholder goals?

Treatment implementation
(Section V)
Perform the plan

Implementation evaluation
(Section VI)
Evaluate whether stakeholder goals have been achieved

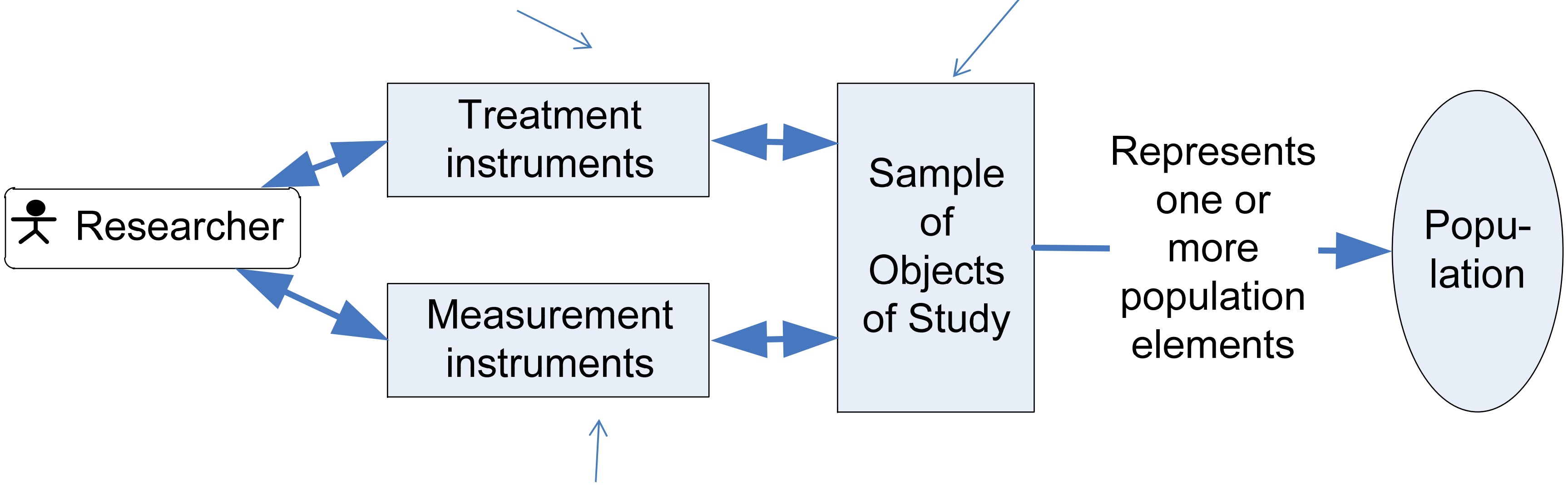
4. Generalizing from TAR

Discussion

General model of empirical scientific research

Entity actually studied by a researcher:
A set of one or more population elements or surrogates for population elements

Instruments to influence the OoS in a particular way (and no other way)



Instruments to observe the OoS (and avoid influence on OoS)

Generalization

- Inference from observations of the OoS to the population
- Like all non-deductive inferences, it is fallible.
  - Ampliative inference: there is more information in the conclusion than in the premisses
  - The researcher needs to give arguments in favor of conclusion
  - And discuss any reasons why the conclusion could be false (threats to external validity)

Kinds of generalization

- **Statistical inference** is reasoning about samples
  - Make an assumption about population distribution and parameters
  - Predict sampe statistic
  - Observations confirm or disconfirm the assumption
- **Case-based inference** is reasoning about cases
  - Observe phenomena in a case
  - Explain in terms of architecture
  - Predict that cases with similar architecture will exhibit similar phenomena

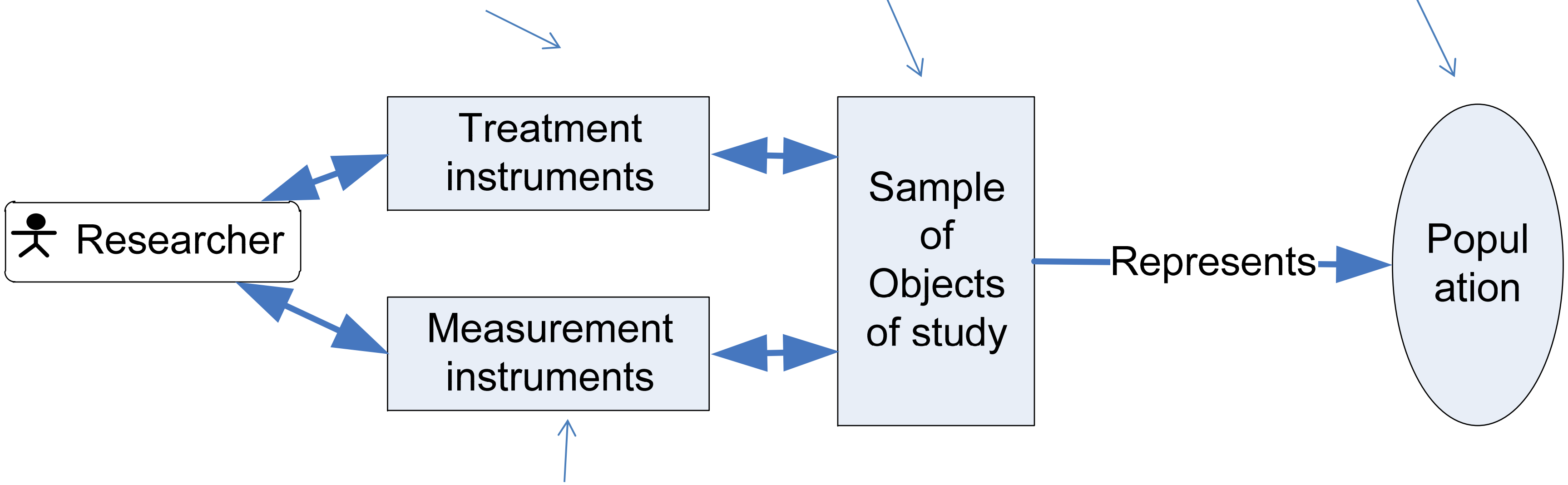
- Statistical inference uses the law of large numbers
  - …. Applied to a population
  - …. Population of what?
  - …. Of similar elements
- Case-based inference uses the similarity
  - …. Similarity of population elements (cases)
  - …. Similarity in what?
  - …. In architecture of population elements (cases)

Model of experimental research

Experimental unit(s) to be treated

Instruments used by researcher to apply a treatment to the object of study

Population of all possible elements similar to object of study

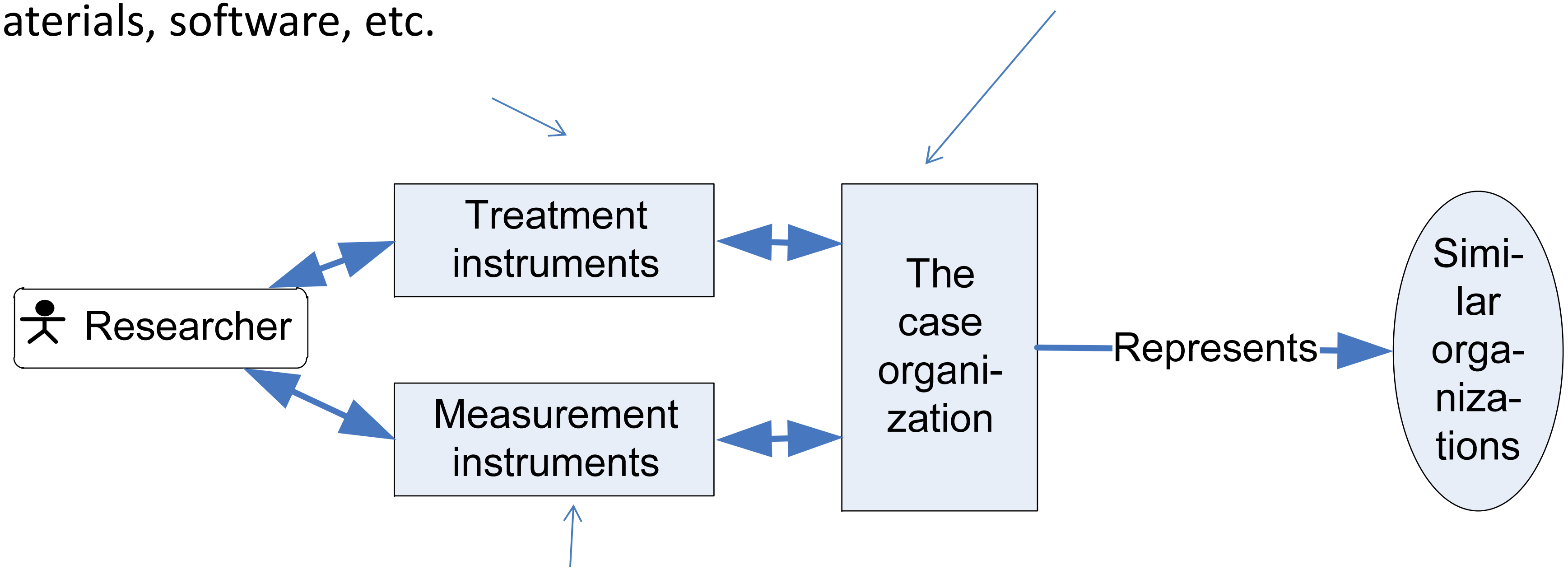


Instruments to observe what happened, e.g. pressure meters, voltmeters, questionnaires, interviews, cameras, a diary, logs, etc.

Model of action research

An individual organization deemed to be representative for a population of unobserved similar organizations

Instruments used by researcher to help the organization, e,.g. teaching materials, software, etc.



Instruments to observe what happened, e.g. a diary, logs, etc.

Case-based reasoning

- Reasoning from an observed case to an unobserved case
- Is based on **similarity** between cases.
- Source in legal reasoning
  - When are two cases “”similar”?
  - What follows from this “similarity”?

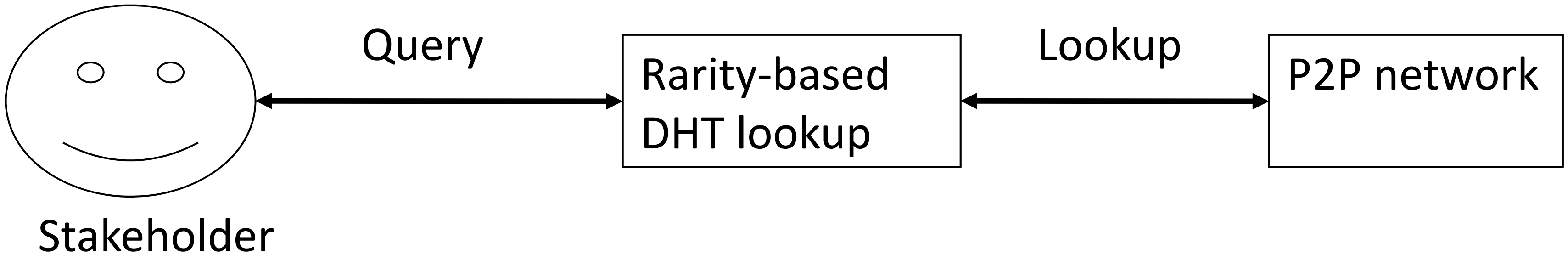
Theory of similarity, Created, defended, attacked and (dis)agreed on in the courtroom

- Also well-known in engineering
  - Test an airfoil in a wind tunnel.
  - Infer how a real airplane with similar shape behaves in the air.
- If cases A and B are “similar” then **some** observations of A can also be expected to occur in B
  - Must be justified by a **theory of similarity**.

Example of case-based reasoning

- Researcher designs a “rarity-based” lookup algorithm for distributed hash tables (DHTs).
- The algorithm should improve ability to store and look up larger numbers of service descriptions
- Service descriptions are relatively small and have many keys.

Simulated context | **Artefact prototype** | **Simulated context**



Simulated context

- Pick number n according to some probability distribution;
- pick random document;
- pick n terms according to uniform distribution;
- use these as query terms

represents

- Eventual set of queries

Artefact prototype

- FreePastry DHT system with 500 nodes
- Java 1.5 lookup implementation;
- Run on DAS-2 distributed supercomputer;
- Limit the number of answers to 50

represents

- Intended implementation

Simulated context

- Random selection of 100 000 from a set of 260 000 documents with on the average 104 terms, created for IR research

represents

- set of resource descriptions. (Both have Zipf distribution.)

Example (continued)

- What theory of similarity is used in this example?
- *Any implementation of my rarity-based lookup procedure*
  - *Running on any P2P network*
  - *Using any distributed hash table*
  - *Looking up any set of small documents containing terms in a Zipf distribution*
  - *According to any query*
- *will have the same performance in terms of*
  - *Recall*
  - *Execution time*
- To provide more support for this we need additional validation
  - on extreme cases (more nodes, more documents, more queries)
  - On different systems (P2P network, DHT)

Architectural inference

- How can this inference be valid?
  - Because it is plausible that the **mechanisms** observed in the observed case will also occur in the unobserved case …
  - … because they have similar **architecture**
- Architectural inference
  - Identify the case **architecture**
  - Identify the **mechanisms** by which the case responds to stimuli
  - **Explain** the observations in terms of these mechanisms
  - Conclude that in cases with **similar** architecture, similar mechanisms will produce similar responses
  - *Provided there are **no** countervailing mechanisms*

Repeatability

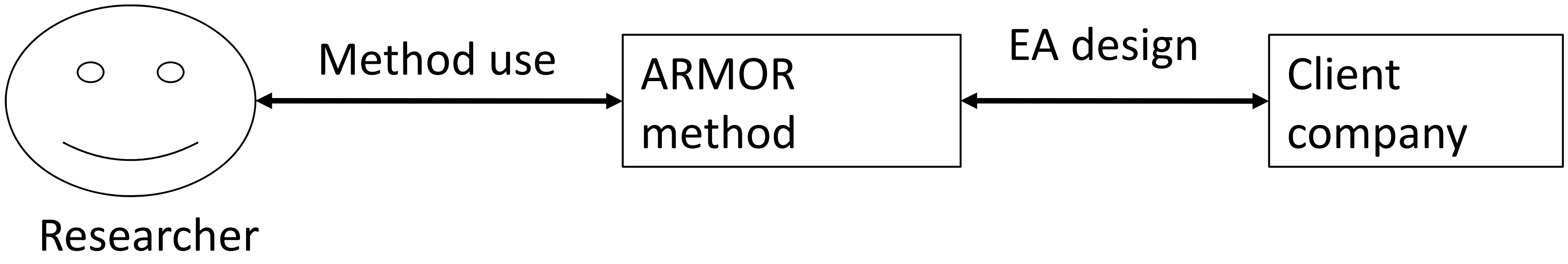
- Like any scientific claim, plausibility must be tested by repeating the research
  - By different researchers
  - Differerent time and place
  - Different objects of study from the same population
- This rules out any of these factors as relevant similarities

Regularities versus mechanisms

- Uses **statistical inference** to show there are regularities without using any knowledge of underlying mechanisms
  - Statistical claims are about samples from a population of similar elements

- Use **case-based inference** to test the presence of mechanisms
  - Case-based claims are about individuals from a population of similar elements

Simulated context | **Artefact prototype** | **Simulated context**



•Designer-researcher	•Limited description	•Company with EA organization
represents	**represents**	**represents**
•Architects who will eventually use method	•Eventual method description	•Future companies where ARMOR will be applied

- Researcher is not representative of intended users
- Client company is representative of similar companies
  - service organization, experienced architects, mature EA process are relevant features that impact the effectiveness of ARMOR

Summary of architectural inference

- Architecture of a case
  - Entities with **capabilities**
  - Relations of **influence**
- Mechanism of an architecture
  - The way entities interact when a system stimulus occurs

- Relevant similarities of cases are architectural
  - The case is a sociotechnical system with an architecture
  - Components have capabilities and influence relations
  - People have competencies, devices have specifications, matter has potential to respond

Architectural inference gives us architectural generalizations

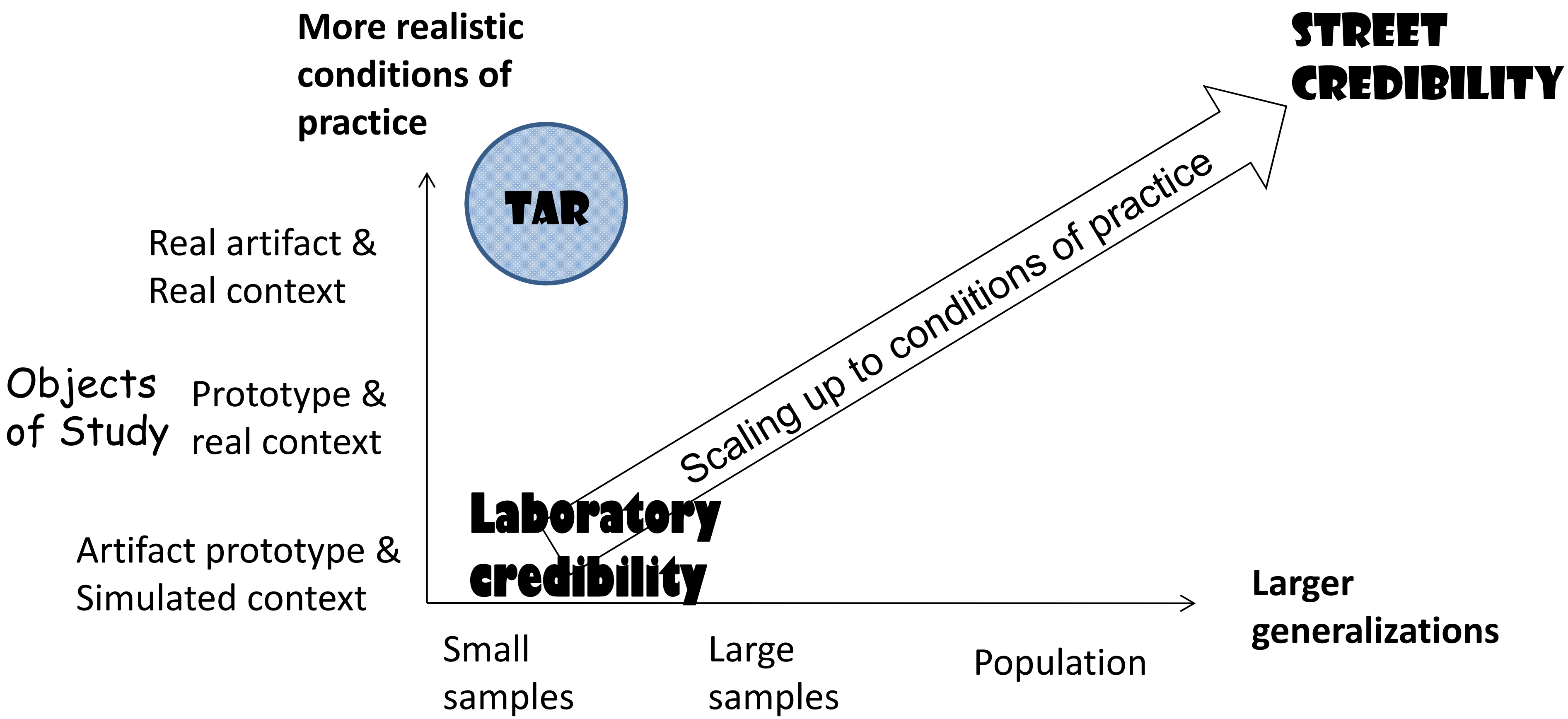
- Generalizations are existential (“for some”, “for many”, “for most”),
- not universal (“for all”)
  - There may be exceptions
  - Individual cases have many architectures
  - Components may have many capabilities
  - A stimulus may trigger many interacting mechanisms

- Universality comes at the price of idealization
  - Laws of nature are about an idealized, non-existing universe
  - Point masses (physics), perfect rationality (economics) and Turing machines (computer science)

4. Summary

TAR and design science

- Design science is designing and investigating artifacts
- Characteristic for design science is scaling up to practice
  - Start at the desk,
  - continue in the lab,
  - end up in the field
  - In the field you do TAR and/or statistical field experiments
  - Similar to scaling up in pharmaceutical research



- From: “It works in theory” before simulation
  - To “It works in the lab” ….
    - … via increasingly realistic simulations …
      - To “It works in practice”

Limitations of TAR

- Not always clear which of the many conditions of the case contribute to the effect of the artifact
  - These conditions must be present in other cases too
  - But we may not know what they are
- Competencies of people in the context may have a major influence on effect of artifact
- Manage these limitations by repeating the research

- Technical action research is the validation of an artifact by applying it in a realistic case

- The technical researcher is
  - a designer
  - a helper
  - a researcher of knowledge questions

- Generalize by identifying architecture and mechanisms