INF5181: Process Improvement and Agile Methods in Systems Development

Lecture 09:

SPI & Empirical Research Methods

Dr. Dietmar Pfahl

email: dietmarp@ifi.uio.no

Fall 2011

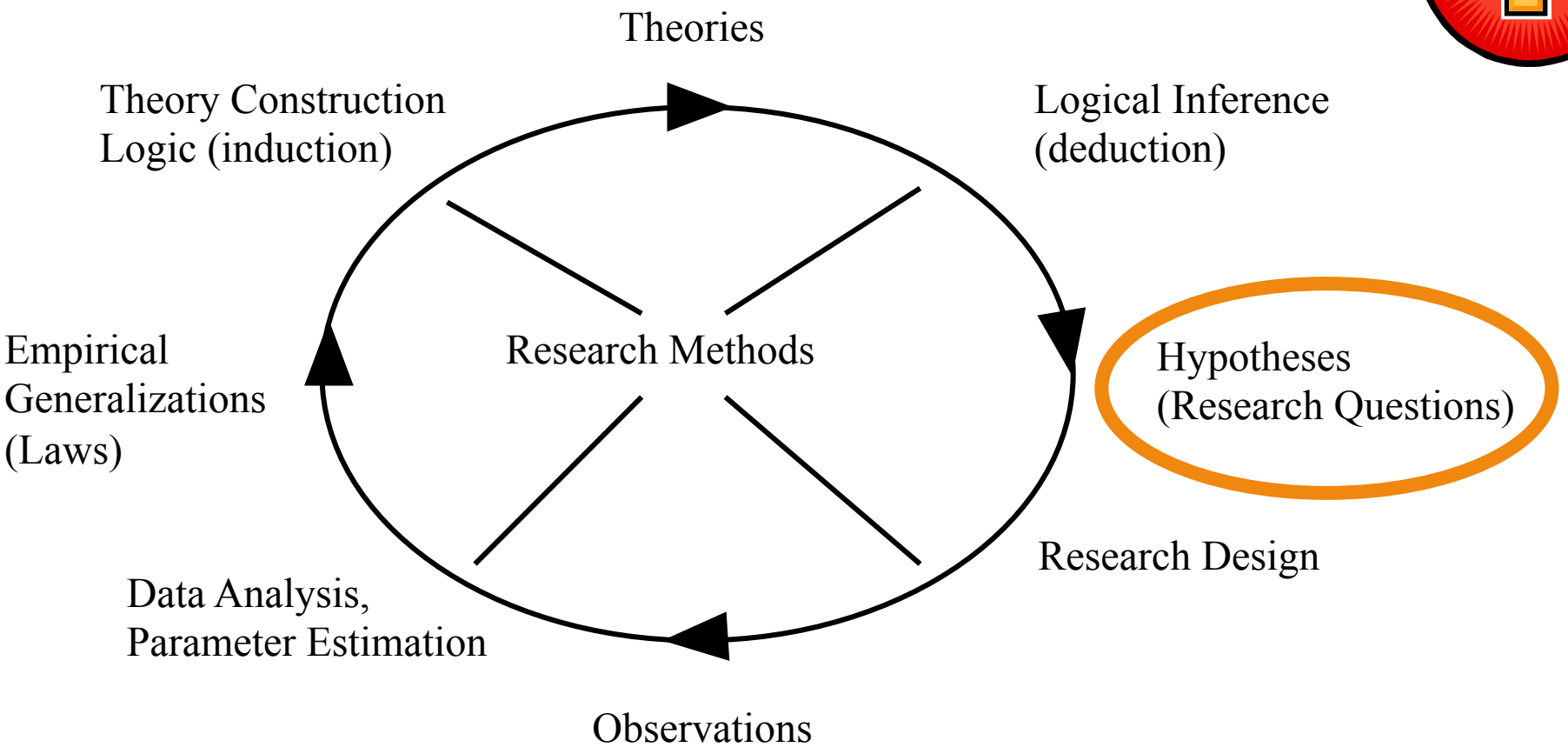
Structure of Lecture 09

- Hour 1:
  - Motivation & Basic Terminology
  - Research Methods
- Hour 2:
  - Example Studies
  - Argumentation
- Hour 3:
  - Lecture 10

How is Research related to SPI?

- SPI may require the gathering of knowledge through research for:
  - Understanding what is going on
  - Identification of root-causes of perceived problems
  - Decision-support, e.g.,
    - if there are alternative solutions to problems (which one is best?)
    - if a given solution needs to be adapted (how to adapt?)

The Wallace Model



Wallace, Walter L. (1971) The Logic of Science in Sociology. New York: Aldine

Research Question – Examples

- How many defects are detected in the field?
- How effective is testing?
- What skills do our architects need?
- What agile practices do our software engineers apply?
- What types of defects do our newly hired programmers typically make? (and why?)
- What requirements analysis method works best for us? (and why?)

Exercise

- *Hanna is a young researcher in an industrial research lab. She would like to understand how developers in the business units of her company use (or not) UML diagrams during software development. This is because, as a student, her professors recommended UML diagrams be used during software design, but her recent exposure to industrial practices used by the developers in the company to which her lab belongs indicates that UML is rarely used. Her goal is to explore how widely UML diagrams are used within her company (and in industry in general), and more specifically how these diagrams are used as collaborative shared artifacts during software development.*
- **What could be Hanna's research question?**

Exercise (cont'd)

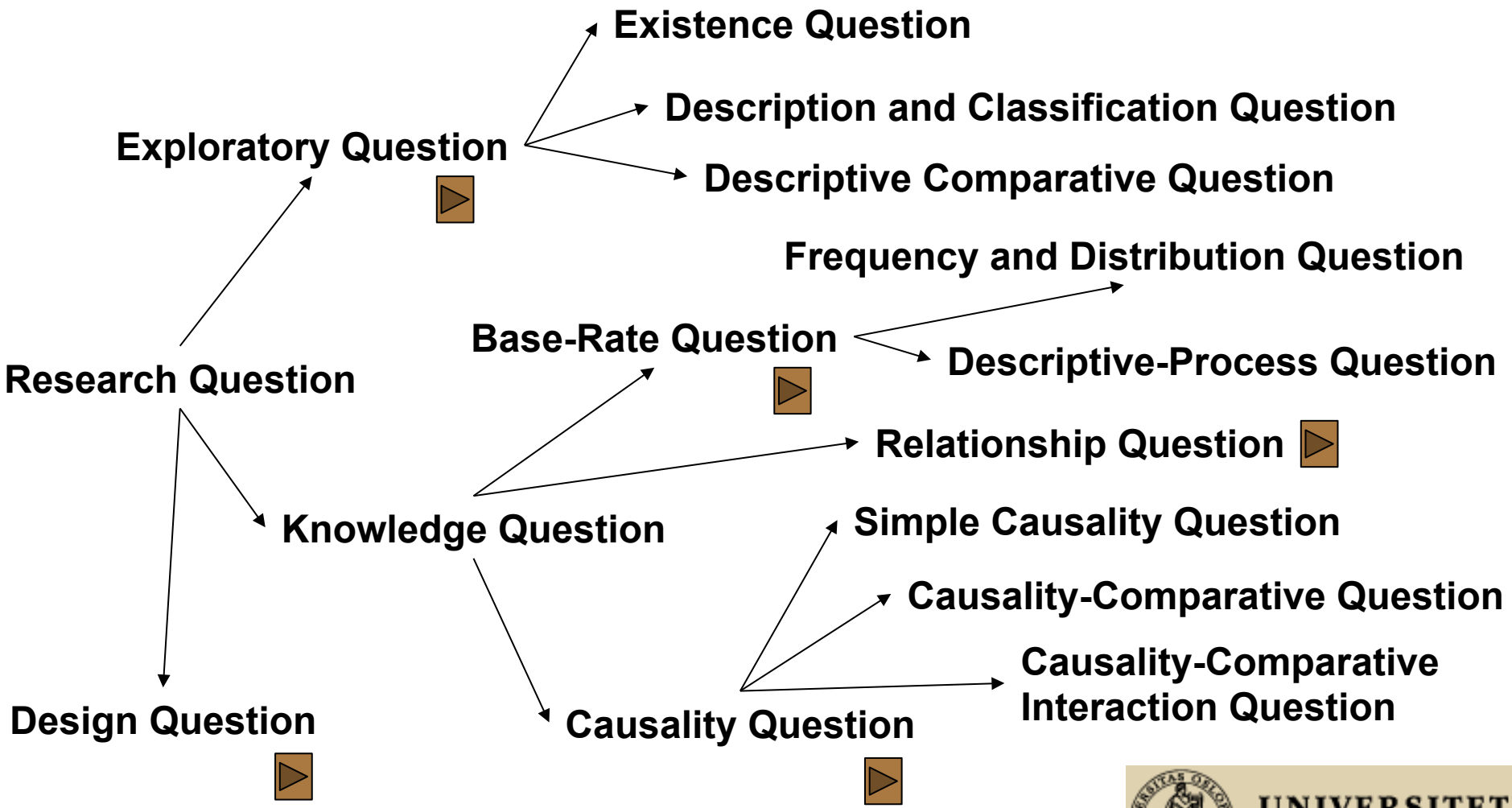
Possible Research Question:

- *How widely are UML diagrams used as collaborative shared artifacts during software development?*

This question pre-supposes:

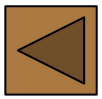
- We know what a "shared collaborative artifact" is
- We can reliably identify "shared collaborative artifacts"
- We can reliably say what "UML diagrams" are
- It is clear what we mean by "software development"

Research Questions – Taxonomy



Exploratory Questions

- Existence questions → Does X exist?
  - Example: *Do collaborative shared artifacts actually exist?*
- Description and classification questions → What is X like? / What are its properties? / How can it be categorized? / How can we measure it? / What is its purpose? / What are its components? / How do the components relate to each other?
  - Example: *What are all the types of shared collaborative artifacts?*
- Descriptive comparative questions → How does X differ from Y?
  - Example: *How do UML diagrams differ from other representations of, say, design representations?*



Other Types of Research Questions

- **Knowledge Questions:** focusing on the way the world is
  - Questions about the normal pattern of occurance of a phenomenon (Base-rate Questions)
  - Questions about relationships between two different phenomena (Relationship Questions)
  - Questions about causality between two phenomena (Causality Questions)

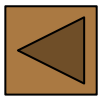
- **Design Questions:** concerned with how to do things better

Knowledge Questions

- Base-rate:
  - Frequency and Distribution Questions → How often does X occur? / What is an average amount of X?

    Example: *How many distinct UML diagrams are created in software development projects in large software companies?*
  - Descriptive-Process Questions → How does X normally work? / What is the process by which X happens? / In what sequence do the events of X occur?

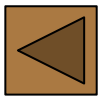
    Example: *How do software developers use UML diagrams?*



Knowledge Questions (cont'd)

- Relationship:
  - Relationship Questions → Are X and Y related? / Do occurrences of X correlate with occurrences of Y?

    Example: *Do project managers' claims about how often their teams use UML correlate with the actual use of UML?*

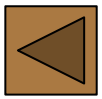


Knowledge Questions (cont'd)

- Causality:
  - Simple Causality Questions → Does X cause Y? / Does X prevent Y? / What causes Y? / What are all the factors that cause Y? / What effect does X have on Y?

    Example: *Does the use of UML diagrams improve the quality of the design?*
  - Causality-Comparative Questions → Does X cause more Y than does Z? / Is X better at preventing Y than Z?

    Example: *Does the use of UML diagrams improve the quality of the design more than other graphical design notations?*
  - Causality-Comparative Interaction Questions



Knowledge Questions (cont'd)

- Causality:
  - Causality-Comparative Interaction Questions → Does X or Z cause more Y under one condition but not others?

    Example: *Does the use of UML diagrams improve the quality of the design more than other graphical design notations in large projects, but not otherwise?*

Design Questions

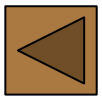
→ ”What is an effective way to achieve X?” / What strategies help to achieve X?”

Examples:

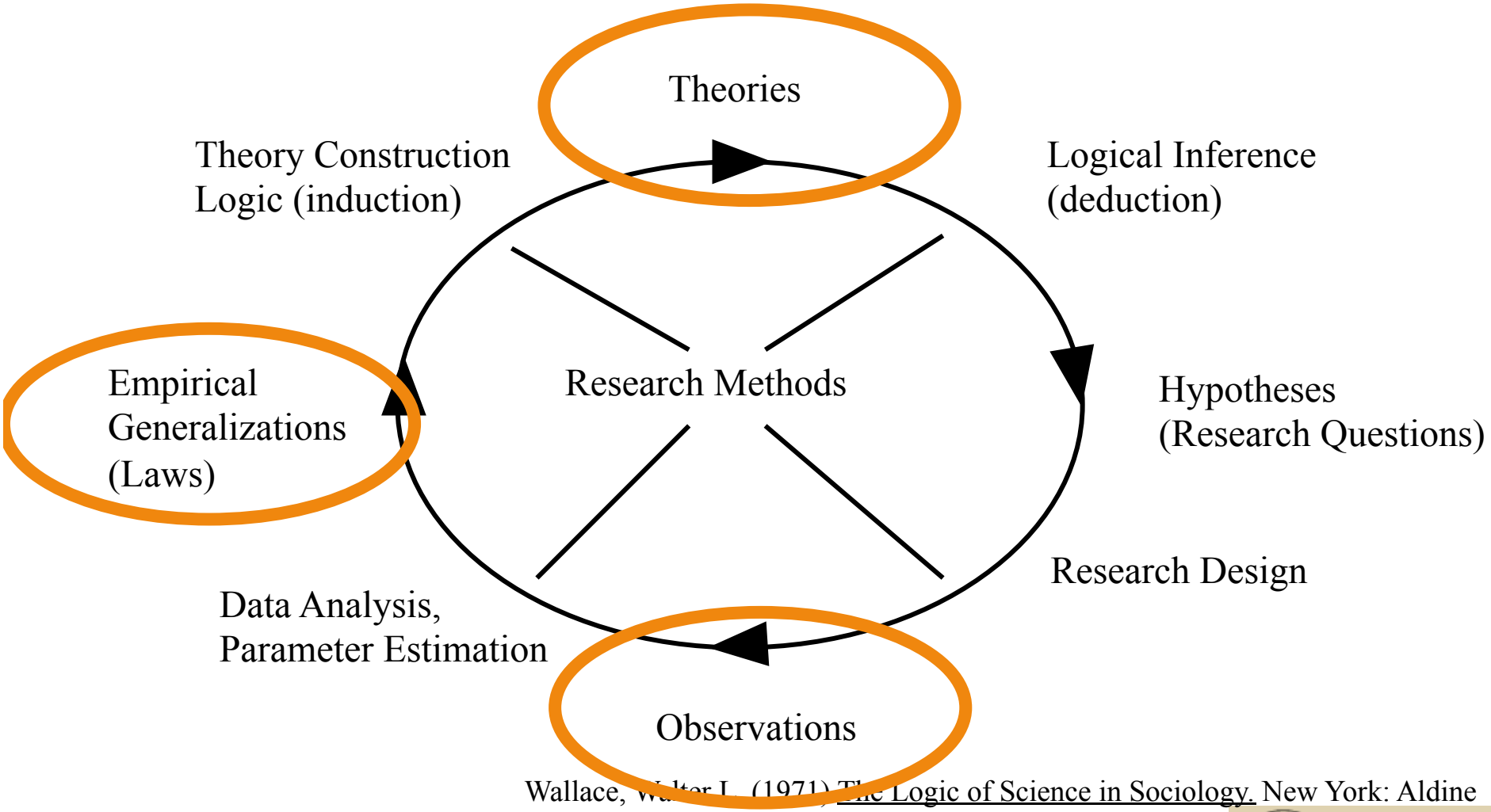
What is an effective way for teams to represent design knowledge in order to improve coordination?

or

What is an effective way for teams to represent design knowledge in order to improve design quality?

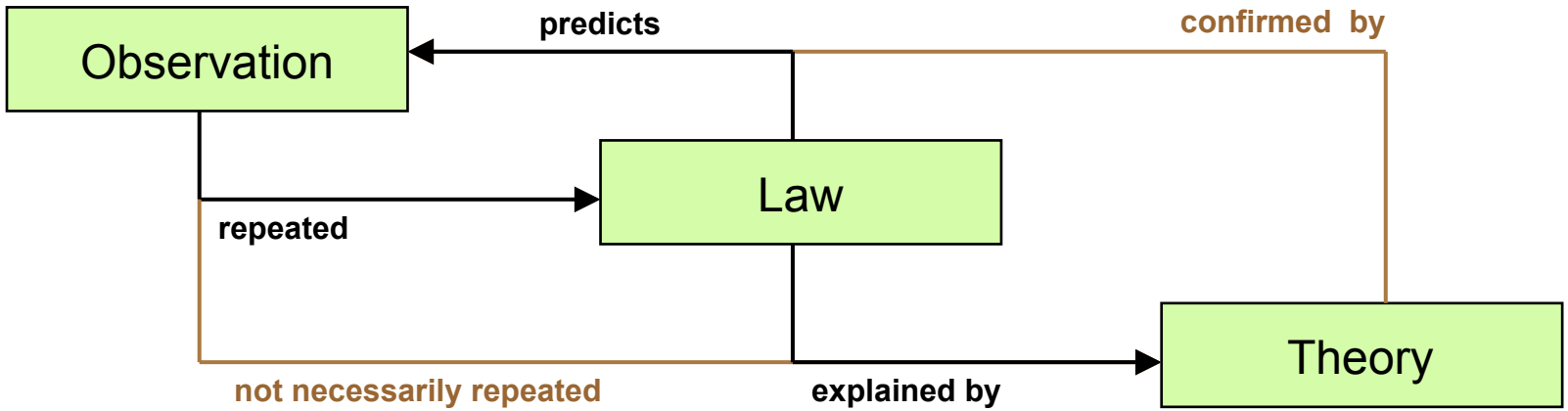


The Wallace Model



Wallace, Walter L. (1971) The Logic of Science in Sociology. New York: Aldine

Observations, Laws, and Theories



Laws explain *how* things happen, but not *why*. **Theory** does.

(Source: Endres, Rombach: A Handbook of Software and Systems Engineering)

Definitions

- **Observations** may be facts or simply subjective impressions; typically they are made through using our senses or instruments

- **Laws** are generalizations of repeatable observations; they tell us how something occurs, but not why; thus laws can be used for predictions, but not for explanation.
  - Example: Kepler's laws of planet movement

- **Theories** explain and classify observations; they explain why laws are true.

The Central Role of Observation

- **Theories** can come “from anywhere” and be based on logical inference (deduction) or **singular observations** (induction)
- A **law** is based on **repeated observations** under similar condition
- In order to **test (refute, check) theories**, **observations** must be made in well-designed **empirical studies**

How to get a Theory?

- Theories can come “from anywhere” and be based on logical inference (deduction) or singular observations (induction)

- **Grounded Theory** is a technique for developing theory iteratively from qualitative data (e.g., from interviews).
  - Initial analysis of the data begins without any preconceived categories.
  - As interesting patterns emerge, the researcher repeatedly compares these with existing data, and collects more data to support or emerge the supporting theory.

Theory – Characterisation

- A scientific theory
  - identifies and defines a set of phenomena, and makes assumptions about those phenomena and the relationships between them.
  - precisely defines the theoretical terms, so that a community of scientists can observe and measure them.
  - explains why certain relationships occur.

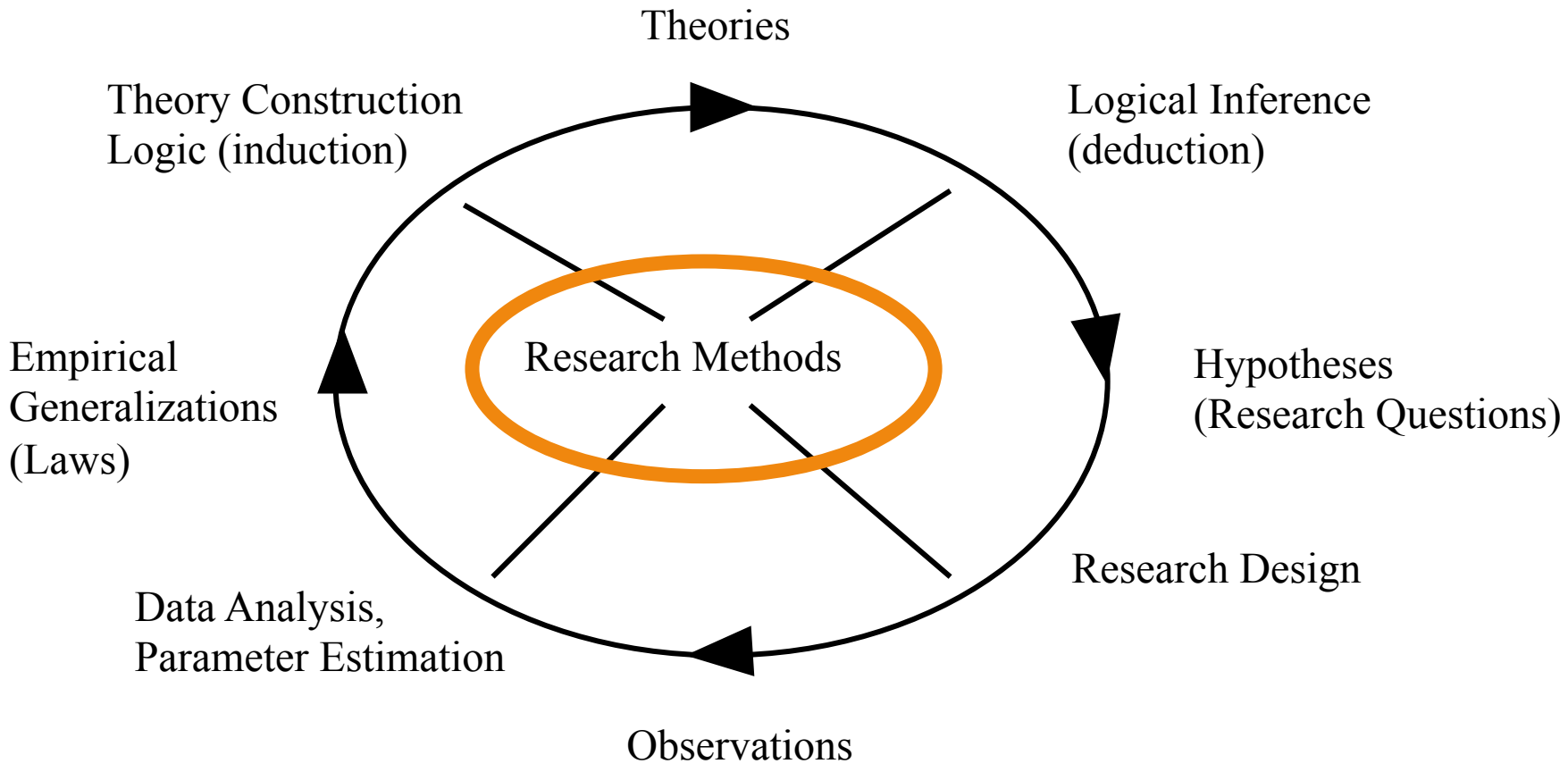
- *Positivists* expect their theories to have strong predictive power, and so look for generalised models of cause-and-effect as the basis for theories.

- *Constructivists* expect theories to strengthen their understanding of complex situations, and so tend to make more use of categorisations and analogies.

Theory – Example

- Hanna might develop a theory around the use of UML diagrams as standardised forms of external memory.
- According to this theory, UML diagrams are used to summarize the results of meetings and discussions, to remind participants of a shared understanding that they have already developed.
- The theory
  - should precisely define the meanings of terms such as ”diagram”, ”participants”, ”discussions”, in order to identify them in any studies performed.
  - should explain why developers chose to use UML diagrams in some circumstances but not in others, and why they include certain things in their diagrams and exclude others.
  - should be able to predict qualities of the diagrams that a team produce based on certain factors, and/or predict the quality of the software produced based on the use of UML diagrams.

The Wallace Model



Wallace, Walter L. (1971) The Logic of Science in Sociology. New York: Aldine

Structure of Lecture 09

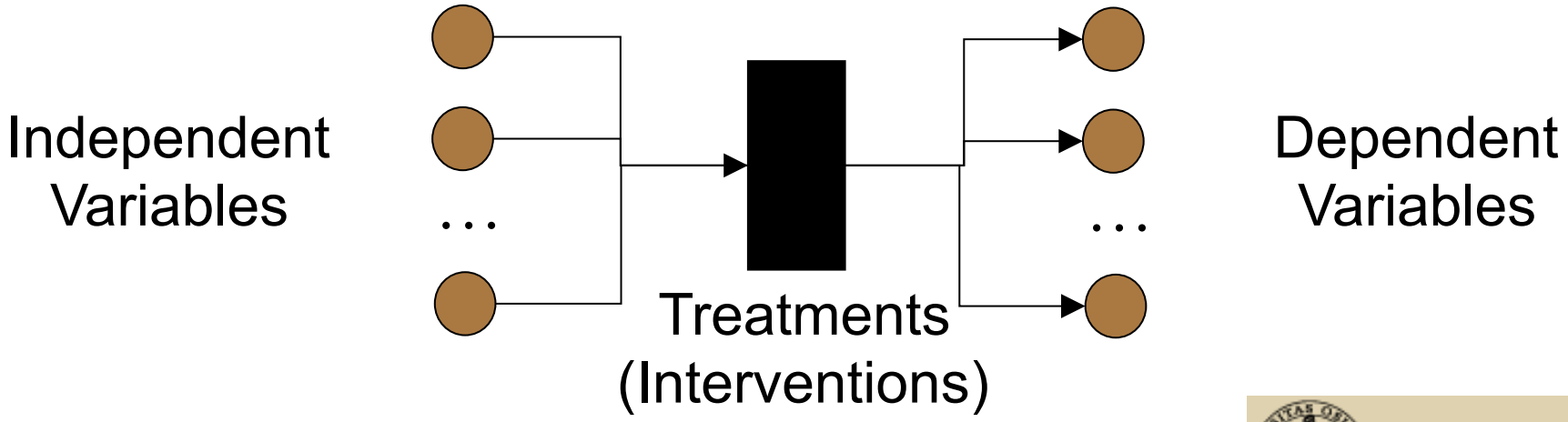
- Hour 1:
  - Motivation & Basic Terminology
  - Research Methods
- Hour 2:
  - Example Studies
  - Argumentation
- Hour 3:
  - Lecture 10

(Empirical) Research Methods

- Controlled Experiment
  - Randomized Experiment
  - Quasi-Experiment
  - Time-Series Experiment
- Case Study
  - Descriptive Case Study
  - Exploratory Case Study
  - Confirmatory Case Study
- Survey
  - Questionnaire-based (primary study)
  - Literature-based (secondary or tertiary study)
- Ethnography
- Action Research

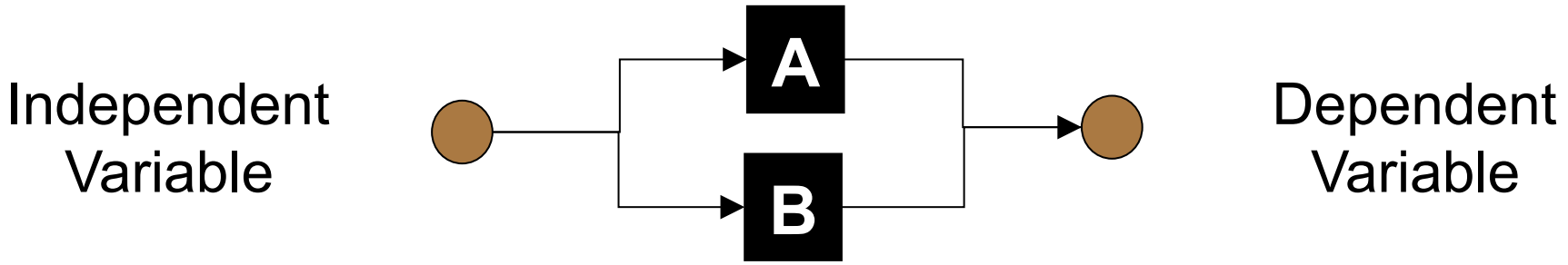
Controlled Experiment – Characterisation

- An investigation of a testable hypothesis where one or more *independent variables* are manipulated to measure their effect on one or more *dependent variables*. Each combination of values of the independent variable is a *treatment*.
- In Software Engineering, typically, experiments require human subjects to perform some task.



Controlled Experiment – Simple Example

- Independent Variable: UML diagram usage (yes or no)
- Dependent Variable: Design Quality
- Treatments: A = use UML / B = don't use UML



Treatments
(1 Factor / 2 Levels)

NB: Design can be *within-subject* or *between-subject*

Controlled Experiment – Variants

- Randomised Experiment
  - Random assignment of subjects to treatments
- Quasi-Experiment
  - No random assignment of subjects to treatments
- Time-Series Experiment
  - The effect of a treatment is measured in discrete time-steps over a period of time
- NB: The most powerful controlled experiments are those that are randomised!

Case Study – Characterisation

- An empirical enquiry that investigates a contemporary phenomenon *within its real-life context*, especially when the boundaries between phenomenon and context are not clearly evident. (Yin, 2002)
  - Example: Experienced practitioners communicate about software design in many different ways (and other than, say, undergraduate students). Thus, Hanna conducts a case study in a local company.
- Important: Proper case selection / clearly stated research question / clearly defined framework for interpreting the observations

Case Study – Variants

- Descriptive Case Study
  - Purely observational / Focus on “What happens?”
- Explorative Case Study
  - Initial investigation of some phenomena to derive new hypotheses and build theories / Focus on “What and Why?”
- Confirmatory Case Study
  - Start out with a given theory and try to refute it, ideally with a series of case studies covering various contexts

Survey – Characterisation

- A type of research that is used to identify the characteristics of a broad population, aiming to answer base-rate questions.
- The defining characteristic is the selection of a *representative* sample from a well-defined population with the aim to generalise from the sample to the population.
- Usually conducted with questionnaires, but can also involve structured interviews or data logging techniques
- Example:
  - Investigate to what extent, for which purpose, by which companies, and by whom within the companies, UML diagrams are used.

Exercise

- Discuss in groups (pairs) the strengths and weaknesses of
  - *controlled experiment,*
  - *case study, and*
  - *survey research*.

Ethnography – Characterisation

- Generally, the goal is to study a community of people to understand how the members of that community make sense of their social interaction. (Robinson, 2007)
- For Software Engineering, ethnography can help to understand how technical communities build a culture of practices and communication strategies that enables them to perform technical work collaboratively.

- Example:
  - UML users in general, or in a specific company / project / team

Action Research – Characterisation

- Action researchers attempt to solve a real-world problem while simultaneously studying the experience of solving the problem.

Or:

- Action researchers aim to intervene in the studied situation for the explicit purpose of improving the situation.

- NB: This approach requires a lot of discipline and critical reflection about the role of the researcher in the studied situation.
- Example:
  - Hanna participates in a software project using UML diagrams as means for cooperation. She is an actor and at the same time observes.

Selecting the Research Method

- The choice of method depends among other things on:
  - Suitable study subject (e.g., do participants have enough experience?)
  - Possibility to control the environment
  - The size/scale/cost of the study
  - The need for generality in the results
  - Availability of information/data and other resources
  - What is the purpose of the study? (exploration, prediction, understanding of cause-effect relations, applicability of results in industry, ...)

- Difficult to provide general recommendation with respect to choice of method

Exercise

- Discuss how the points from the previous slide effect the choice of the research method, e.g., if the aim is to *study the productivity of programmers*.

Experiment or Case Study or Survey?

- **Experiments** give the researchers freedom to isolate a defined effect and to hold other things constant (→ research in-depth)
  - Often difficult to avoid that people respond differently than they would have done in a natural environment
  - Also, you might have a too restricted setting and omit important influencing factors that play a role in a natural environment
- **Case-studies** have the advantage that you observe people doing what they are actually doing in their natural environment (→ research in-the-typical)
  - It is limited what you can control without interfering with natural activity
- **Surveys** provide researchers with information about what many people (think they) are doing (→ research in-breadth)
  - No control whatsoever (issues relate to data validity and representativeness)

Exercise

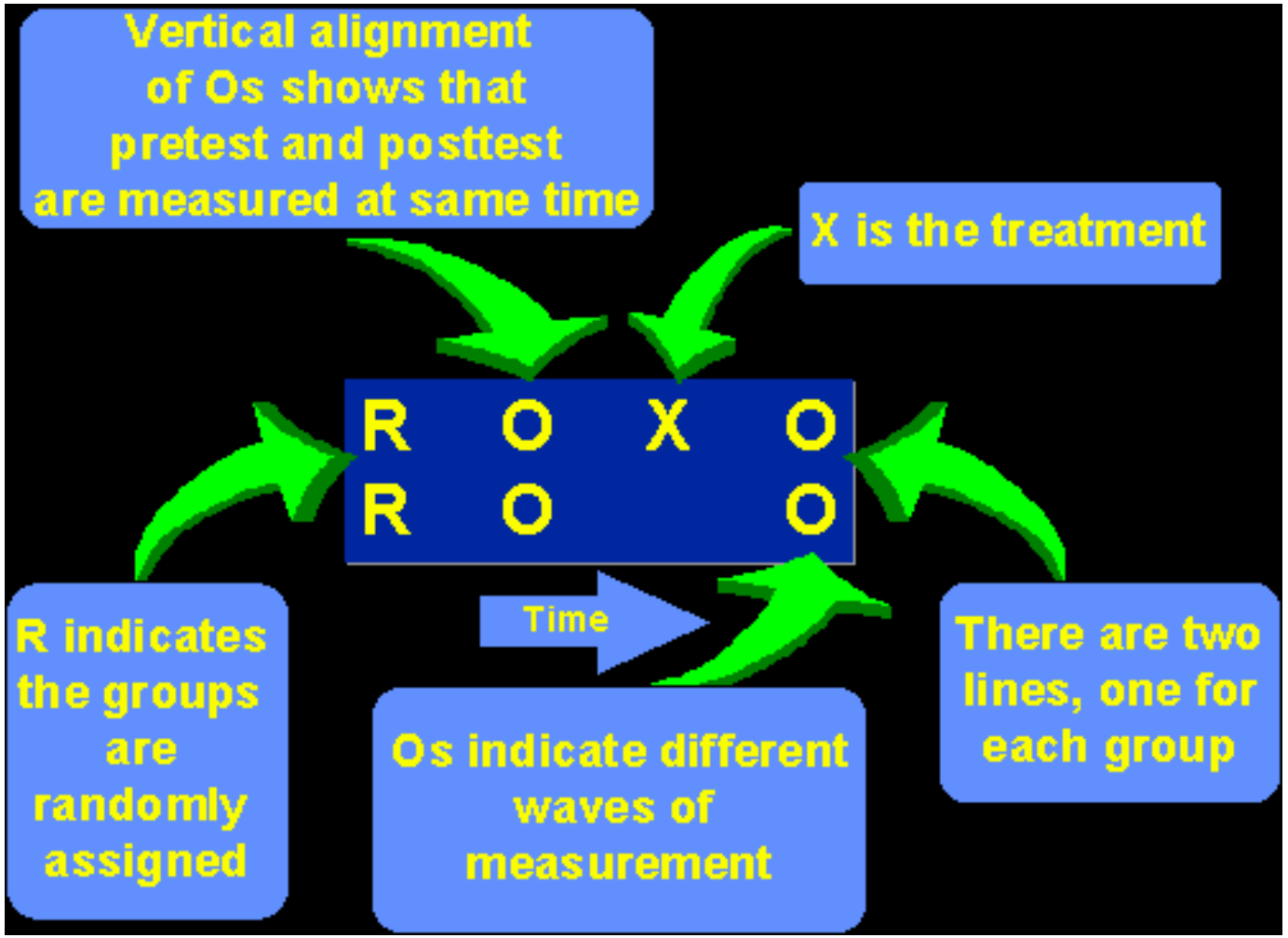
Outline a study to find out whether JAVA or C++ should be the programming language for an introductory course on programming.

Important:

- Provide assumptions/details concerning context, objectives, subjects, etc.
- Give reasons for your choice of research method (and associated design)

http://www.socialresearchmethods.net/kb/design.php

Experimental Designs



Group
=
Set of
“experimental units”
(subjects)

UNIVERSITETET I OSLO

Experimental Designs (cont'd)

- One-Group designs (within-group):
  - Post-Test

    X O
  - Pre-Test and Post-Test

    O X O
  - Interrupted time-series

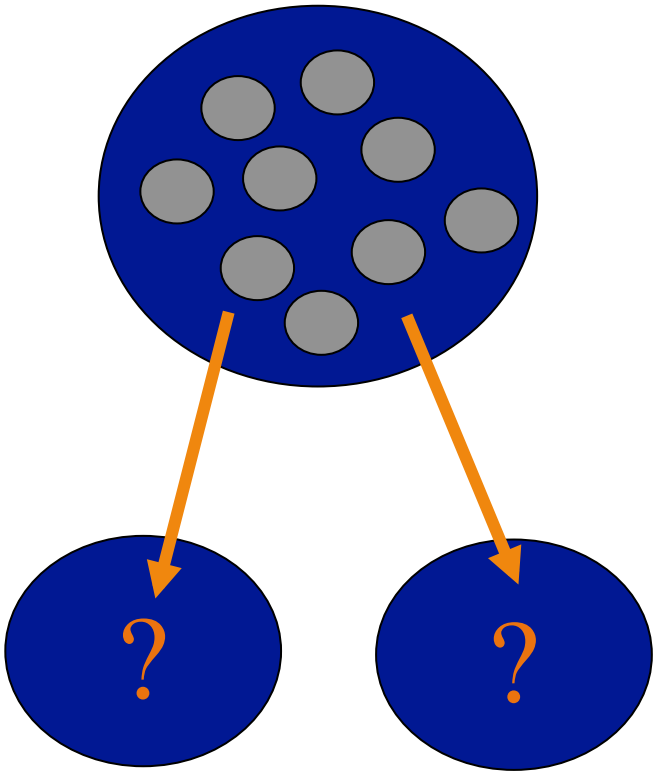
    O O X O O O X O X O …

    With:

    O = observation (measurement)

    X = treatment (intervention)
- Multiple-Group designs (between-groups):
  - With or without random sampling / assignment
  - With or without blocking
  - Balanced or unbalanced
  - Factorial Designs:
    - nested vs. crossed
    - interaction between factors

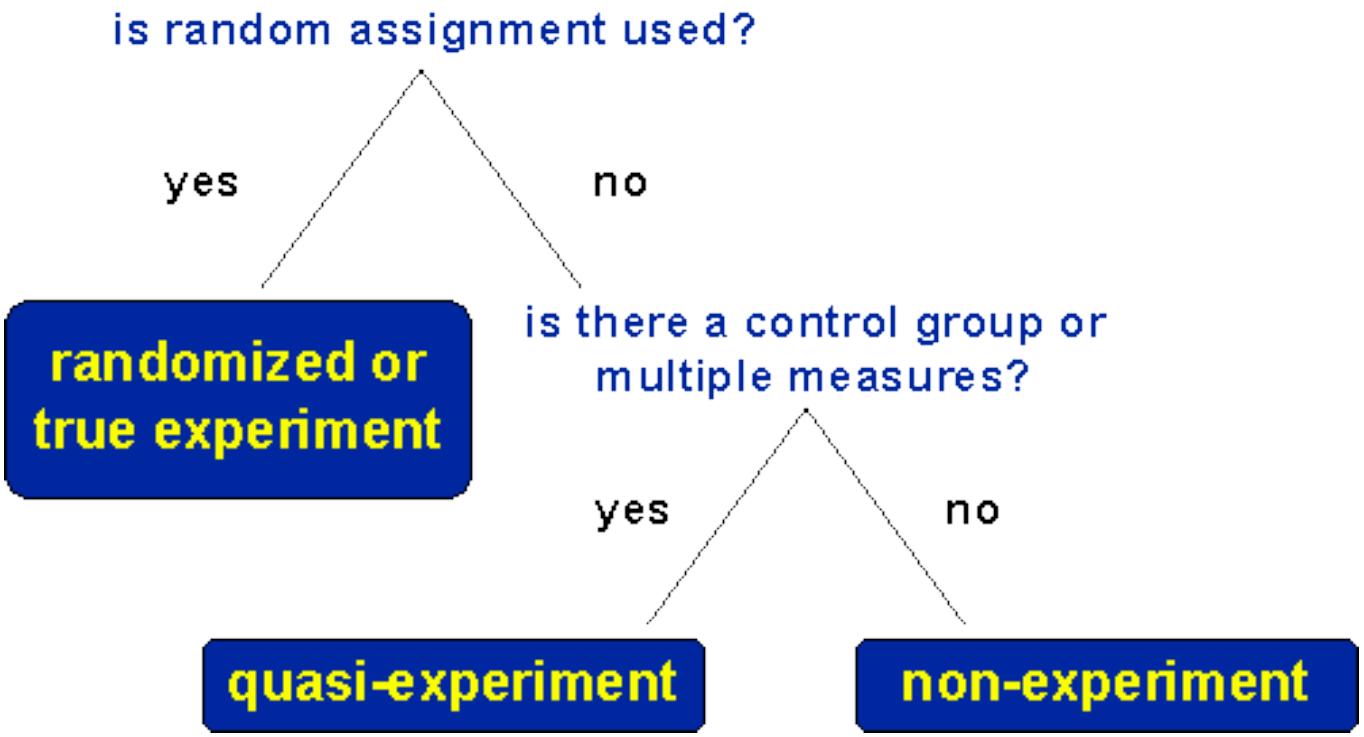
Experimental Designs: Random Assignment /1



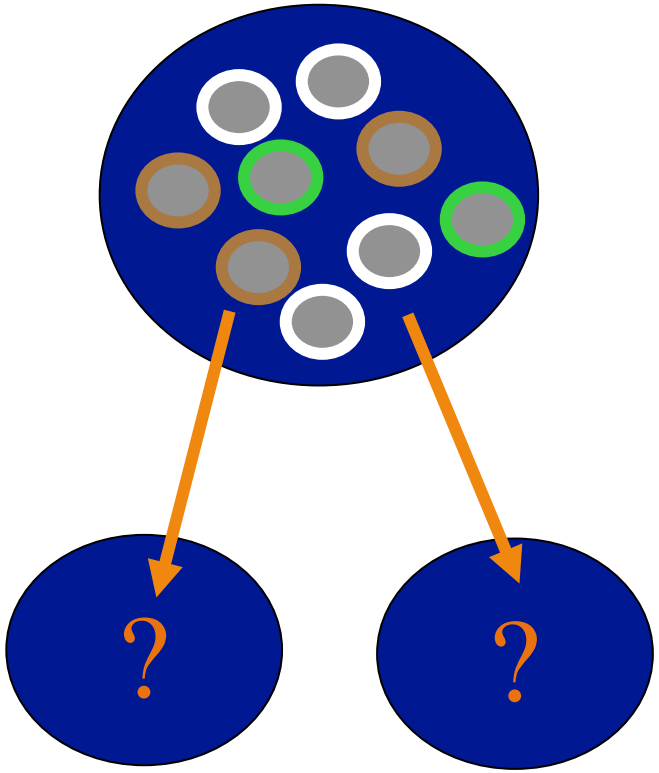
- **Definition [Pfl94]:**
  - Randomization is the random assignment of subjects to groups or of treatments to experimental units, so that we can assume independence (and thus validity) of results.
- **Rationale for Randomization [Pfl94]:**
  - Sometimes the results of an experimental trial can be affected by the time, the place or unknown characteristics of the participants (= experimental units / subjects)
  - These uncontrollable factors can have effects that hide or skew the results of the controllable variables.
  - To spread and diffuse the effects of these uncontrollable or unknown factors, you can
    - assign the order of trials randomly,
    - assign the participants to each trial randomly, or
    - assign the location of each trial random[y, whenever possible.

Experimental Designs: Random Assignment /2

Randomization is a prerequisite for a controlled experiment!

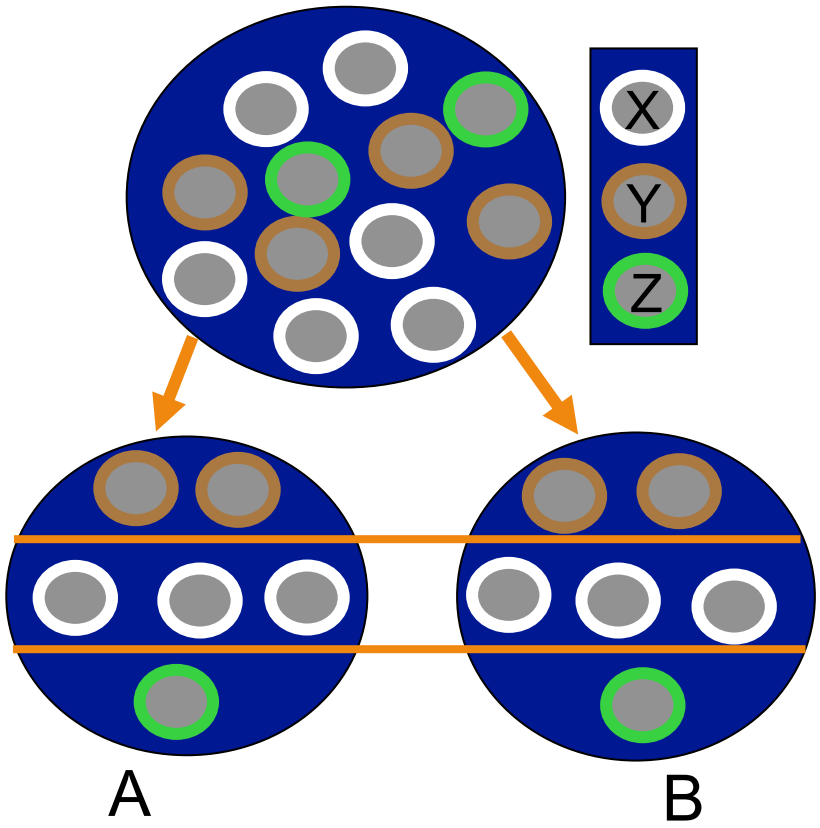


Experimental Designs: Blocking /1



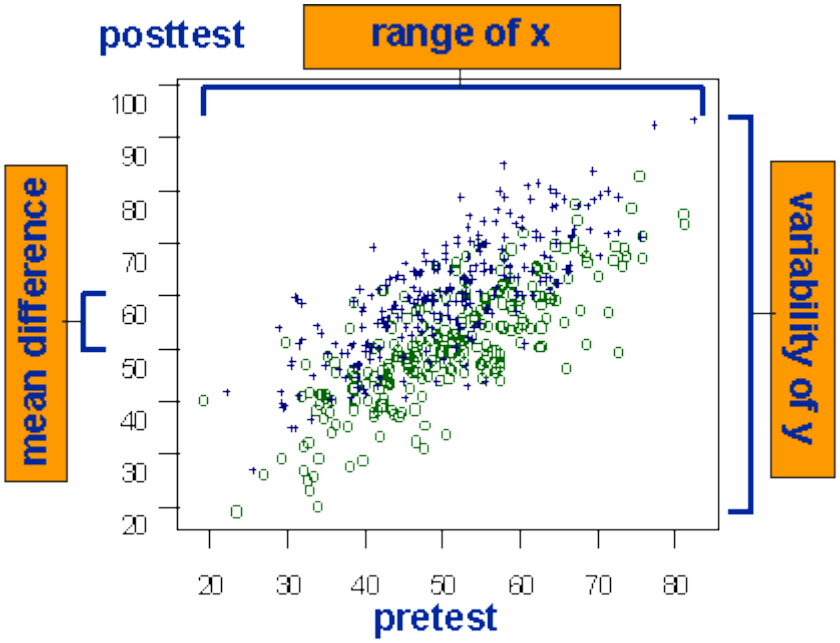
- **Definition [Pfl94]:**
  - Blocking (Stratification) means allocating experimental units to blocks (strata) or groups so the units within a block are relatively homogeneous.
- **Rationale for Blocking [Pfl94]:**
  - The blocked design captures the anticipated variation in the blocks by grouping like varieties, so that the variation does not contribute to the experimental error.

Experimental Designs: Blocking /2

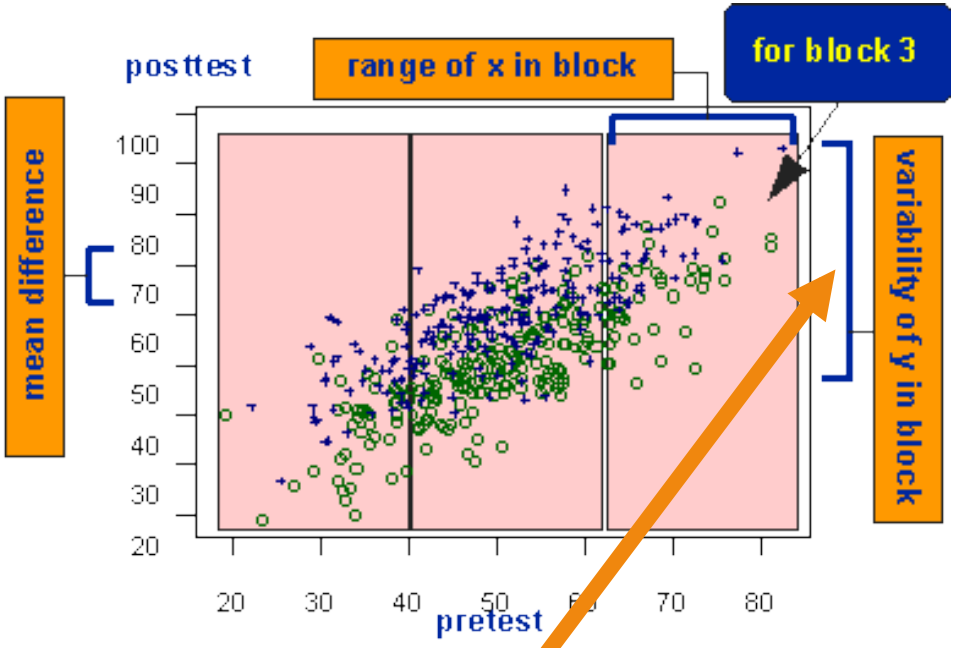


- **Example [Pfl94]:**
  - Suppose you are investigating the comparative effects of two design techniques A and B on the quality of the resulting code.
  - The experiment involves teaching the techniques to twelve developers and measuring the number of defects found per thousand lines of code to assess the code quality.
  - It may be the case that the twelve developers graduated from three universities. It is possible that the universities trained the developers in very different ways, so that the effect of being from a particular university can affect the way in which the design technique is understood or used.
  - To eliminate this possibility, three blocks can be defined so that the first block contains all developers from university X, the second block from university Y, and the third block from university Z. Then, the treatments are assigned at random to the developers from each block. If the first block has six developers, you would expect three to be assigned to design method A, and three to method B, for instance.

Experimental Designs: Blocking /3



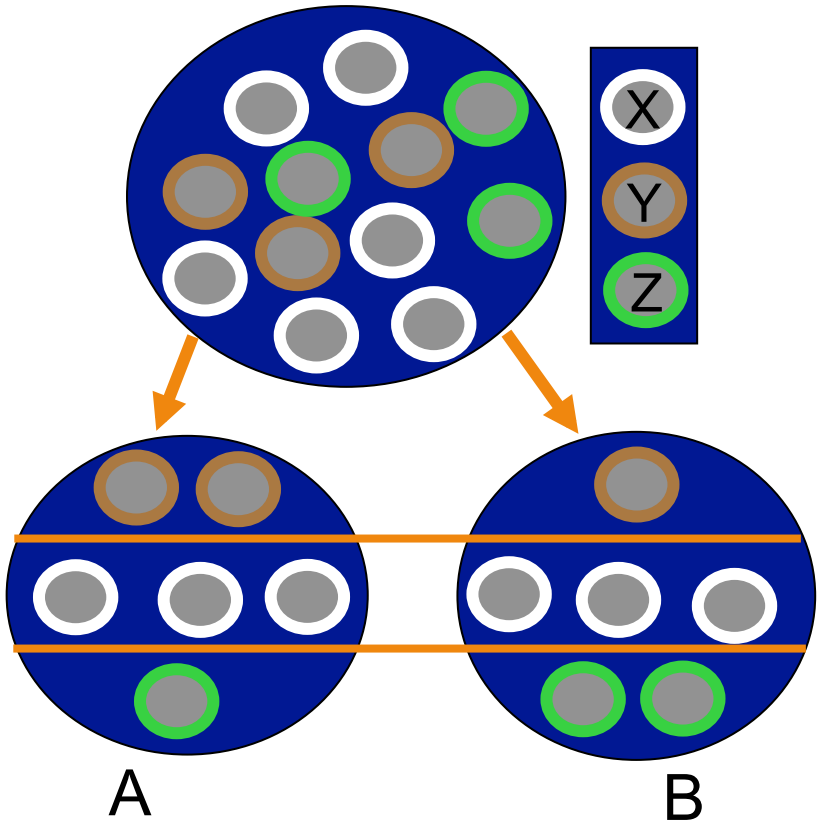
without blocking



with blocking

Less variance increases statistical power
(for the same mean difference)

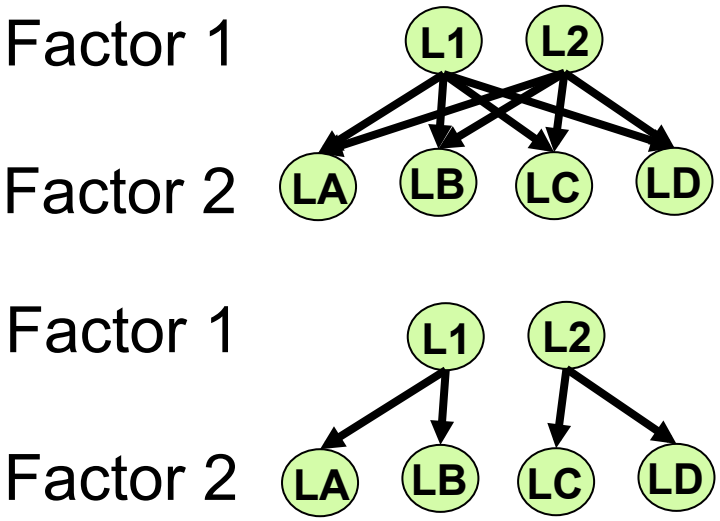
Experimental Designs: Balancing



- **Definition [Pfl94]:**
  - Balancing is the blocking and assigning of treatments so that an equal number of subjects is assigned to each treatment, wherever possible.
- **Rationale for Balancing [Pfl94]:**
  - Balancing is desirable because it simplifies the statistical analysis, but it is not necessary.
  - Designs can range from being completely balanced to having little or no balance.

⬅ unbalanced

Experimental Designs: Factorial Designs



- **Definition of "Factorial Design:**
  - The design of an experiment can be expressed by explicitly stating the number of factors and how they relate to the different treatments.
  - Expressing the design in terms of factors, tells you how many different treatment combinations are required.
- **Crossed Design:**
  - Two factors, F1 and F2, in a design are said to be crossed if each level of each factor appears with each level of the other factor.
- **Nested Design:**
  - Factor F2 is nested within factor F1 if each meaningful level of F2 occurs in conjunction with only one level of factor F1.

Experimental Designs: Crossed vs. Nested

Useful for looking at two factors, each with two or more conditions.

		Tool Usage		
		B_1	B_2	no
Design Method	A_1	$a_1\ b_1$	$a_1\ b_2$	$a_1\ b_3$
	A_2	$a_2\ b_1$	$a_2\ b_2$	$a_2\ b_3$

Useful for investigating one factor with two or more conditions,

Design Method				
Method A_1			Method A_2	
Tool B_1 Usage			Tool B_2 Usage	
yes		no	yes	no
$a_1\ b_1$		$a_1\ b_3$	$a_2\ b_2$	$a_2\ b_3$

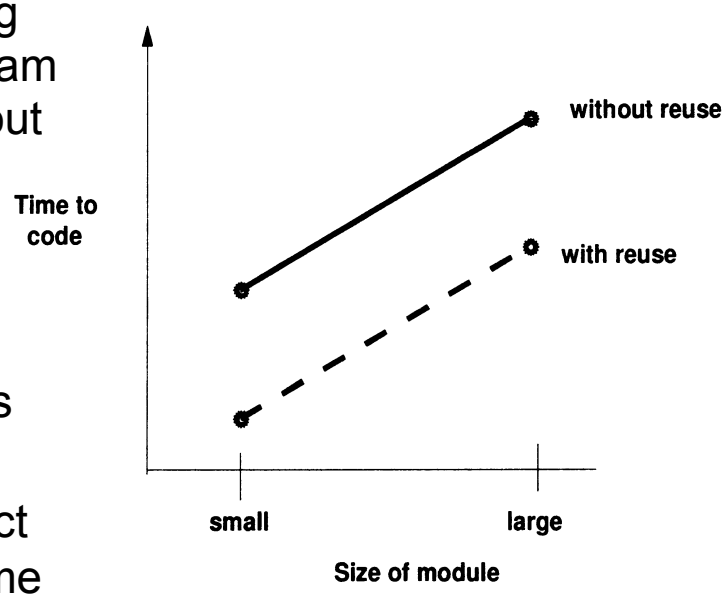
similar, but not necessarily identical Factors

Factorial Design:

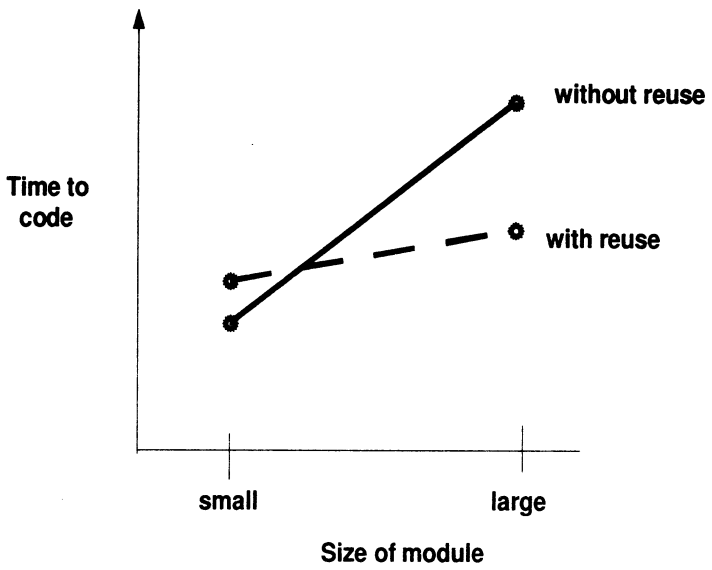
- **Crossing** (each level of each factor appears with each level of the other factor
- **Nesting** (each level of one factor occurs entirely in conjunction with one level of another factor)
- Proper nested or crossed design may reduce the number of cases to be tested.

Experimental Designs: Interaction Effects

- **Example:** Measuring time to code a program module with or without using a reusable repository
  - **Case 1**: No interaction between factors
  - **Case 2**: Interaction effect → Effect on Time to Code (Factor 1) depends (also) on Size of Module (Factor 2)

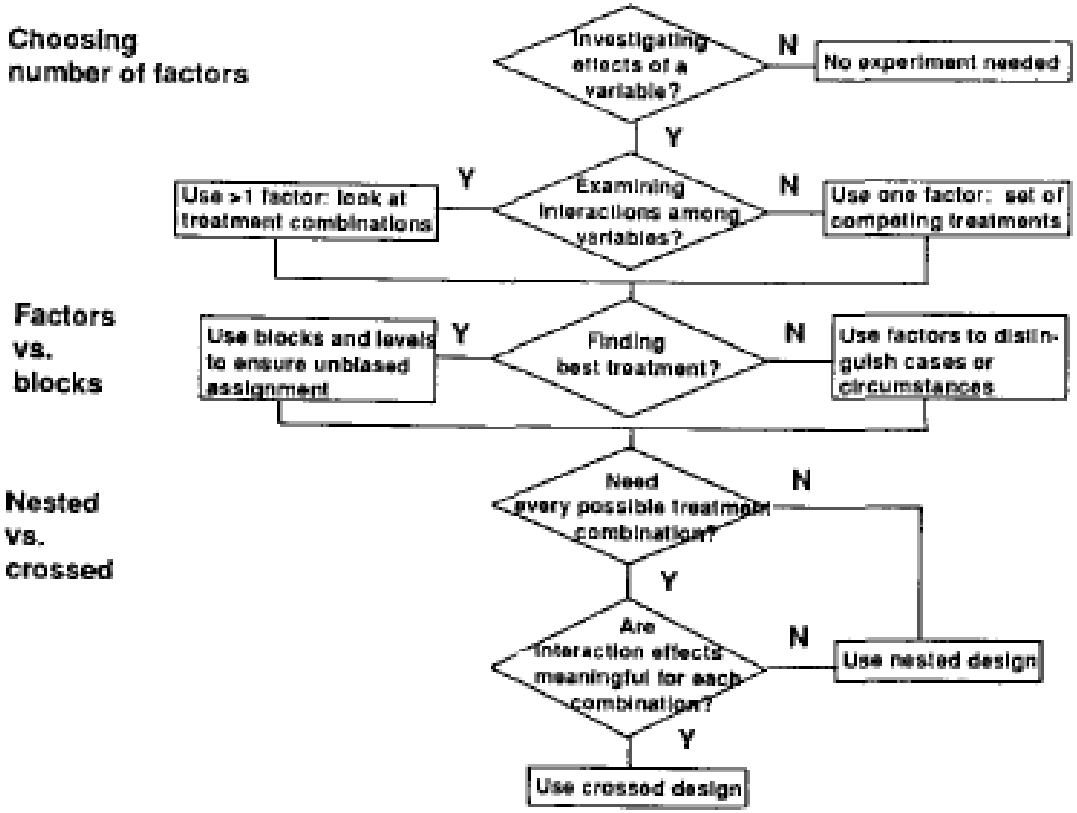


Case 1



Case 2

Experimental Designs: Design Selection



Flow Chart for selecting an Experimental Design [Pfl95]

- [Pfl95] S. L. Pfleeger: Experimental Design and Analysis in Software Engineering. Annals of Software Engineering, vol. 1, pp. 219-253, 1995.
- Also appeared as: S. L. Pfleeger: Experimental design and analysis in software engineering, Parts 1 to 5, Software Engineering Notes, 1995 and 1996.

Validity & Reliability of Empirical Studies

- **Construct Validity**
  - Concepts being studied are operationalised and measured correctly (→ do the measures used actually represent the concepts you want to measure?)
- **Internal Validity**
  - Establish a causal relationship and sort out spurious relationships (→ exclude confounding variables / by: random sampling, blocking, balancing)
- **Conclusion Validity**
  - Do proper statistical inference
- **External Validity**
  - Establish the domain to which a study's findings can be generalized (→ precisely describe the population and experimental conditions)
- **Reliability**
  - The study can be repeated (i.e., by other researchers) and yields the same results
  - The measurement instrument is reliable (→ interrater agreement)

Beware of the following ”effects”:

- **Hawthorne-effect**
  - The attention given to the workers in "Hawthorne Plant of the Western Electric Company" is thought to have had greater effect than the process changes that were studied.
- **Weinberg-effect**
  - Study showed that system developers who knew that they were being assessed, decreased their performance on most parameters which were **not** assessed.
- **Observer effect**
  - Refers to changes that the act of observing will make on the phenomenon being observed
- **Observer-expectancy effect (theory-loaded observation/cognitive bias)**
  - What you (think you) observe, is influenced by what you think you will observe. (→ the man who shot after a tractor while moose hunting)
- **Subject-expectancy effect (cognitive bias)**
  - A cognitive bias that occurs when a subject expects a given result and therefore unconsciously manipulates an experiment or reports the expected result.
- **The effect of the question’s structure (construction, forming)**
  - “Do you think the lecture today was A: good, B: very good, C: brilliant, D: exceptional"

Empirical Research Guidelines

National Research Council Canada
Institute for Information Technology

Conseil national de recherches Canada
Institut de Technologie de l'information

ERB-1082

NRC·CNRC

Preliminary Guidelines for Empirical Research in Software Engineering

Kitchenham, B.A., Pfleeger, S.L., Pickard, L.M.,
Jones, P.W., Hoaglin, D.C., El-Emam, K., and Rosenberg, J.
January 2001

UNIVERSITETET I OSLO

Structure of Lecture 09

- Hour 1:
  - Motivation & Basic Terminology
  - Research Methods
- Hour 2:
  - Example Studies
  - Argumentation
- Hour 3:
  - Lecture 10

Example of a famous but not so good study

"Chaos Report" by The Standish Group (1994):

Cost ($)	Succeeded	Challenged	Failed
< 750K	55%	31%	14%
750K-1.5M	33%	45%	22%
1.5M-3M	25%	47%	28%
3M - 6M	15%	52%	33%
6M-10M	8%	51%	41%

The Standish Group's 1994 CHAOS Report*

- *"The Standish Group research shows a staggering 31.1% of projects will be canceled before they ever get completed. Further results indicate 52.7% of projects will cost 189% of their original estimates."*
- These results are still – despite their age – the most frequently quoted cost estimation figures. They are still used as:
  - Evidence of how bad software projects are
  - Input to governmental reports
  - Excuse for large cost overruns (we are better than average!)
  - Support for the belief that "we have improved a lot since 1994".
  - A means to get research funding

How Large Are Software Cost Overruns? A Review of the 1994 CHAOS Report
By Magne Jørgensen and Kjetil Moløkken Østvold, Simula Research Laboratory

Problem 1: Lack of precise definition of measures

- What does 189% average cost overrun mean?
  - *"… 52.7% of projects (the so-called challenged projects) will overrun their initial cost estimates by 189%"*
  - *"The average cost overruns for combined challenged and cancelled projects is 189%."*
  - *"… 52.7% of projects will cost 189% of their original estimates"*

NB: All formulations mentioned above are used by the Standish Group

A web-search on the use of the numbers revealed the confusion ...

- **189% or 89% overrun?**
  - 50% of the documents described the result as “189% cost overrun”,
  - 40% as “189% of original estimate”, and
  - 10% as “89% cost overrun”.

- **All projects, challenged projects, or challenged + cancelled projects?**
  - 70% of the web documents related the result to “53% of the projects”
    - without explicitly pointing out that this 53% referred to challenged projects only
  - 16% related the results to “all projects”
  - 8% related the results to “challenged and cancelled projects”
  - only 6% of the documents explicitly pointed out that the average cost overrun is based on “challenged projects” only.

Problem 2: Lack of correspondence with other surveys

Study	Jenkins [9]	Phan [10]	Bergeron [11]
Year	1984	1988	1992
Respondents	23 software organizations	191 software projects	89 software projects
Country of Respondents	USA	USA	Canada
Average Cost Overrun	34%	33%	33%

Problem 3: Lack of knowledge about research method

- No description of cost measures (!!!), selection process, or analysis method.
- Simula sent several mails to them and received responses like:
  - *”Why would you be surprised that we would not want to give out detailed information about how we conduct our studies? This is how we make a living. There is absolutely no incentive for us to want anyone to replicate our study - that's like giving away our business for free.”*
- After 3 mails they stopped responding.
  - Not even willing to respond on whether the 1994-study results should be interpreted as ”89%” or ”189%” cost overrun.
- They sell their CHAOS reports, but not to Simula ...
  - Simula had to order the report through SINTEF (thanks to Tore Dybå)

Potential reasons for high cost overruns

- Incorrect interpretation of own results.
  - Seems that initial results were interpreted as 189% of initial estimate, i.e., 89% cost overrun.
  - Later, however, this is by the Standish Group described as 189% cost overrun.
- No category for cost underrun in data collection schema
  - Underruns described as 0% overrun?
  - Project challenged on time or functionality may still have effort under-runs.
- Non-random sampling
  - “We then called and mailed a number of confidential surveys to a random sample of top IT executives, asking them to share failure stories [!!!]. During September and October of that year, we collected the majority of the 365 surveys we needed to publish the CHAOS research.”

Structure of Lecture 09

- Hour 1:
  - Motivation & Basic Terminology
  - Research Methods
- Hour 2:
  - Example Studies
  - Argumentation
- Hour 3:
  - Lecture 10

How to use study results properly?

Example: How important is an early introduction of computers to children?

A real story, …

- The company ”Intelligente barn AS” claimed in a newspaper advertisement that *”children who early start to use PCs will get better school results than children who start later”.*
- This claim by ”Intelligente barn AS” was the conclusion from a study conducted at a number of Norwegian schools.
- The study had shown that *”children who used PC at home at least once a week when 6 year old, got better evaluation from the teachers in such subjects as Norwegian and math in their first school years.”*

- **Question:** Suppose that the study is correctly conducted. Is there then a relationship between what ”Intelligente barn AS” claims and the conclusion from the study?

Argumentation versus Persuasion

- **Question 1:** What is the difference between argumentation and persuasion?
  - "Persuasion is an attempt to move an audience to accept or identify with a particular point of view"
  - "Argumentation is a form of instrumental communication relying on reasoning and proof to influence belief or behavior through the use of spoken or written messages."

  [cf. "Advocacy and opposition" by Rybacki & Rybacki]

- **Question 2:** What is the relationship between empirical study methods and argumentation?

What is an Argument?

Toulmin's Model of Argument

- Stephen Toulmin, originally a British logician, is now a professor at USC. He became frustrated with the inability of formal logic to explain everyday arguments, which prompted him to develop his own model of practical reasoning.

  [Source: http://commfaculty.fullerton.edu/rgass/toulmin2.htm]

Toulmin's Model of Argument

- The **first triad** of his model consists of three basic elements:

Toulmin's Model of Argument (1)

Claim:

- A **claim** is the point an arguer is trying to make.
- The claim is the proposition or assertion an arguer wants another to accept.
- The claim answers the question: "So what is your point?"
  - Example: "You should send a birthday card to Mimi, because she sent you one on your birthday."
  - Example: "I drove last time, so this time it is your turn to drive."
- There are three basic types of claims:
  - **fact:** claims which focus on empirically verifiable phenomena
  - **judgment/value:** claims involving opinions, attitudes, and subjective evaluations of things
  - **policy:** claims advocating courses of action that should be undertaken

Toulmin's Model of Argument (2)

Ground:

- **Ground** refers to the proof or evidence an arguer offers.
- Grounds answers the questions: "What is your proof?", "How come?", "Why?"
- Grounds can consist of statistics, quotations, reports, findings, physical evidence, or various forms of reasoning.
  - Example: "It looks like rain. The barometer is falling."
  - Example: "The other Howard Johnson's restaurants I've been in had clean restrooms, so I'll bet this one has clean restrooms too."
- Grounds can be based on:
  - **evidence:** facts, statistics, reports, or physical proof,
  - **source credibility:** authorities, experts, celebrity endorsers, a close friend, or someone's say-so
  - **analysis and reasoning:** reasons may be offered as proof

Toulmin's Model of Argument (3)

Warrant:

- The **warrant** is the inferential leap that connects the claim with the grounds.
- The warrant is typically implicit (unstated) and requires the listener to recognize the underlying reasoning that makes sense of the claim in light of the grounds.
- The warrant performs a "linking" function by establishing a mental connection between the grounds and the claim
  - Example: "Muffin is running a temperature. I'll bet she has an infection." warrant: sign reasoning; a fever is a reliable sign of an infection
  - Example: "That dog is probably friendly. It is a Golden Retriever." warrant: generalization; most or all Golden Retrievers are friendly
- Warrants can be based on:
  - **ethos:** source credibility, authority
  - **logos:** reason-giving, induction, deduction
  - **pathos:** emotional or motivational appeals
  - **shared values:** free speech, right to know, fairness, etc.

Toulmin's Model of Argument (4)

- The second triad of Toulmin's model involves three additional elements:
  - **Backing** provides additional justification for the warrant.
    - Backing usually consists of evidence to support the type of reasoning employed by the warrant.
  - The **qualifier** states the degree of force or probability to be attached to the claim.
    - The qualifier states how sure the arguer is about his/her claim
  - The **rebuttal** acknowledges exceptions or limitations to the argument.
    - The rebuttal admits to those circumstances or situations where the argument would not hold.

Argumentation in Discussions – Recommendations

How to do proper argumentation:

- During preparation:
  - Acquisition of relevant information from different sources
  - Critical assessment of quality of information.
  - Develop a point of view after information has been acquired and analyzed
  - Make sure that irrelevant facts have no/little influence
  - Self-critical insight in one's own evaluation ability and prejudices (→ limitations)
- During discussion (argumentation):
  - Introduce the background of your argumentation (concepts, opinion, motivation, one's own knowledge level, quality of information, objective, ...)
  - Balance of arguments for and against
  - Focused and relevant argumentation
  - Clear relationship between facts (arguments) and conclusion
  - Conclusion (claim/opinion) that is derived from the facts (arguments) and logical inference rules. In contrast: pre-fabricated opinion that tries to find supportive facts (arguments).

Argumentation in Discussions – What to Avoid

- What should be avoided in a good (constructive, goal-oriented) argumentation:
  - Make indefensible generalizations
  - Make indefensible transfers of validity domain
  - Bring up irrelevant arguments
  - Blow-up an irrelevant side-aspect just for the sake of talking
  - Circular argumentation
  - Talk about something off-topic
  - Personal attack
  - Use definitions and concepts to influence the audience in a non-rational way (→ sophism)
  - Suppose that something is true because it is not proven that it is untrue
  - Refer to prejudices
  - Irrelevant reference to feelings
  - Manipulative use of authorities, tradition, humor, ambiguities, charged words, strange words, ...

Argumentation – Ethics

- **Ethical** standards for argumentation:
  - Knowledge
    - Preferably from different viewpoints
  - Good intensions
    - Especially, no ”hidden agendas”
  - Rationality
  - “Argumentation freedom”
    - Respects other’s right to have other arguments
    - No personal attacks

Literature

- T. Dybå, B. A. Kitchenham, M. Jørgensen (2004) “Evidence-based Software Engineering for Practitioners”, *IEEE Software*
- S. Easterbrook et al. (2008) ”Selecting Empirical Methods for Software Engineering Research”, in F. Shull, J. Singer and D. I. K. Sjøberg: Advanced Topics in Empirical Software Engineering, Chapter 11, pp. 285-311, Springer London (ISBN: 13:978-1-84800-043-8)
- A. Endres and D. Rombach (2003) *A Handbook of Software and Systems Engineering – Empirical Observations, Laws and Theories*, Addison-Wesley
- S. L. Pfleeger (1995-96) “Experimental design and analysis in software engineering”, Parts 1 to 5, *Software Engineering Notes*
- H. Robinson, J. Segal, H. Sharp (2007) ”Ethnographically-informed empirical studies of software practice”, in *Information and Software Technology*,49(6), pp. 540-551
- W. L. Wallace (1971) *The Logic of Science in Sociology*, New York: Aldine
- R. K. Yin (2002) *Case Study Research: Design and Methods*, Sage, Thousand Oaks