

# LOG6306 : Patrons pour la compréhension de programme

Foutse Khomh

[foutse.khomh@polymtl.ca](mailto:foutse.khomh@polymtl.ca)

Local M-4123

Typical software developers?

We offer three kinds of service:

**GOOD - CHEAP - FAST**

You can pick any two

**GOOD** service **CHEAP** won't be **FAST**

**GOOD** service **FAST** won't be **CHEAP**

**FAST** service **CHEAP** won't be **GOOD**

LINEAR SIGNS

# Software costs?



# What is a Design pattern?

A pattern is a general reusable **solution** to a commonly occurring problem within a given context in software development, operation, and maintenance

- The solution must describe steps to solve the problem
  - Architecture
  - Design
  - Implementation

# What is a Design pattern?

A pattern is a **general reusable** solution to a commonly occurring problem within a given context in software development, operation, and maintenance

- The solution must not be particular
- The solution can be adapted
- The solution must be adapted

# What is a Design pattern?

A pattern is a general reusable solution to a commonly occurring problem within a given context in **software development, operation, and maintenance**

- Patterns have been identified for
  - Different phases of software development
  - Different levels of abstraction
  - Different technologies
  - Different paradigms ...

# What is a Design pattern?

*“Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such way that you can use this solution a million times over, without ever doing it the same way twice.”*

—Christopher Alexander, 1977

*“Each pattern is a three part rule, which express a relation between a context, a problem, and a solution.”*

—Christopher Alexander, 1977



# What is a Design pattern?

*“The strict modeling of the real world leads to reflect today’s realities but not necessarily tomorrow’s. The abstractions that emerge during design are key to making a design flexible.”*

—Erich Gamma, 1994

# Design pattern : History

- 1977 et 1979: architecture
  - Christopher Alexander
  - *A Pattern Language: Towns, Buildings, Construction* and the idea of generative patterns
  - *The Timeless Way of Building* and the idea of perfection in architecture
- 1990: object-oriented design
  - Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides
  - Design Patterns are drawn from experience



## Design Patterns

Elements of Reusable  
Object-Oriented Software

Erich Gamma  
Richard Helm  
Ralph Johnson  
John Vlissides



Foreword by Grady Booch

# Structure of a Design Pattern

A design pattern is defined by the following elements:

- a **name** for the Design Pattern;
- a **description** or a model to be applied to solve a problem that can occur in different situations during the design process;
- a **problem**, or the description of the situation you can apply the pattern to;
- a **solution** describing the elements of the project with the relations and their consequences;
- the **consequences**, results and constraints resulting from the application of the pattern.

# **In this class!**

**Examine empirical studies that investigate  
the impact of patterns on code quality**

# In Software Engineering!

- Human component is an essential part of the development task
- The usefulness of a method/tool depends on who is going to use it
- We have many commonalities with social sciences
- Experimentation is essential...**however it can be very complex!**

# Empirical studies in Software Engineering

- Different kinds of study
- Experiment definition and planning
- Experiment design
- Analysis of threats to validity
- Experiment operation

# Kinds of empirical studies

- **Survey:** Retrospective (post mortem), e.g. about a technology/tool being adopted for a period of time
- **Case study:** monitoring an ongoing (real) project
- **Experiment:** performed in a laboratory setting, with a high degree of control
  - Objective: manipulate some variables (e.g. method **A** vs. method **B**) and control others (e.g. ability, experience, experimental objects)
  - Quasi-experiments: you could not really control all variables



# Survey

---

- Collecting opinions, market analysis
  - Example: whether a development process is becoming popular in industry
- Use of questionnaires to collect data
- Characteristics:
  - Intended to understand the entire population, not just the sample
  - Often we can really observe a limited number of variables
  - Of course we can collect data from many variables and observe some of them only





# Various kinds of surveys

---

- **Descriptive:** analyze the distribution of some attributes on a population
  - Distribution of Java knowledge among software developers
- **Explanatory:** try to explain some phenomenon
  - Why developers prefer a technique or the other
- **Exploratory:** preliminary to further studies
  - Understand the developers' characteristics before an experiment



# Case study

---

- Investigate a phenomenon in a specific time frame
  - E.g. evaluate the use of a technique on a real project
  - Study the application of SE techniques in industry settings
  - Differences from experiments
    - Experiments sample on manipulated variables
    - Case studies look at a real situation
  - Pros:
    - Easy to design
    - Often more realistic setting than experiments
  - Cons:
    - Results not generalizable



# (Controlled) Experiments

---

- Need to apply more treatments to evaluate results
  - Compare two different testing techniques
  - Use of UML models vs. stereotyped models
- For each variable involved, I need more measures
  - The same technique should be applied by more subjects
- Could be performed on line
  - High level of control
  - Limited time, thus need for easy tasks
- ...or off-line
  - Lower level of control
  - Could involve more complex tasks



# Experiments are useful for:

---

- Confirm known theories
- Confirm (or sometimes contradict) a common wisdom
- Explore relations existing among variables
- Evaluate the performances of a method
- Many times you believe something is true
  - ...but then you might discover many nice surprises



# Kinds of empirical studies

---

- **Quantitative:** to get numerical relations among variables
  - Are programmers more productive with Java than with C#?
  - Are defects correlated with Chidamber-Kemerer metrics?
- **Qualitative:** to interpret a phenomenon just observing it in its context
  - E.g. by using explanations obtained by interviewing developers
  - I interview developers to know why a given method improves their productivity
    - Live interview
    - Survey questionnaires
- Often quantitative studies should be combined with qualitative ones to better interpret them



# In vitro or in vivo

---

- **In vitro:**
  - Performed in laboratory
  - Controlled conditions
  - Reasonable costs, low risks
    - Experiments carried out with students to evaluate the effectiveness of a testing technique
  - Reality could be different
- I can use the experiment to prepare further studies in a more realistic context



# In vitro vs in vivo studies

---

- **In vivo:**
  - Real projects
  - Cannot control the experimental conditions
  - More realistic settings and subjects
  - Results may be different
  - Higher costs
  - Possibly unacceptable risks
- Can do it when we are sure that the study is mature
  - In vitro experiments provide encouraging results



# Running example

---





# Running example

---

- Use of UML Stereotypes in comprehension and maintenance tasks
  - Filippo Ricca, Massimiliano Di Penta, Marco Torchiano, Paolo Tonella, Mariano Ceccato: How Developers' Experience and Ability Influence Web Application Comprehension Tasks Supported by UML Stereotypes: A Series of Four Experiments. *IEEE Trans. Software Eng.* 36(1): 96-118 (2010)
  - Filippo Ricca, Massimiliano Di Penta, Marco Torchiano, Paolo Tonella, Mariano Ceccato: The Role of Experience and Ability in Comprehension Tasks Supported by UML Stereotypes. *ICSE 2007*: 375-384
- In the following briefly referred as “Conallen”



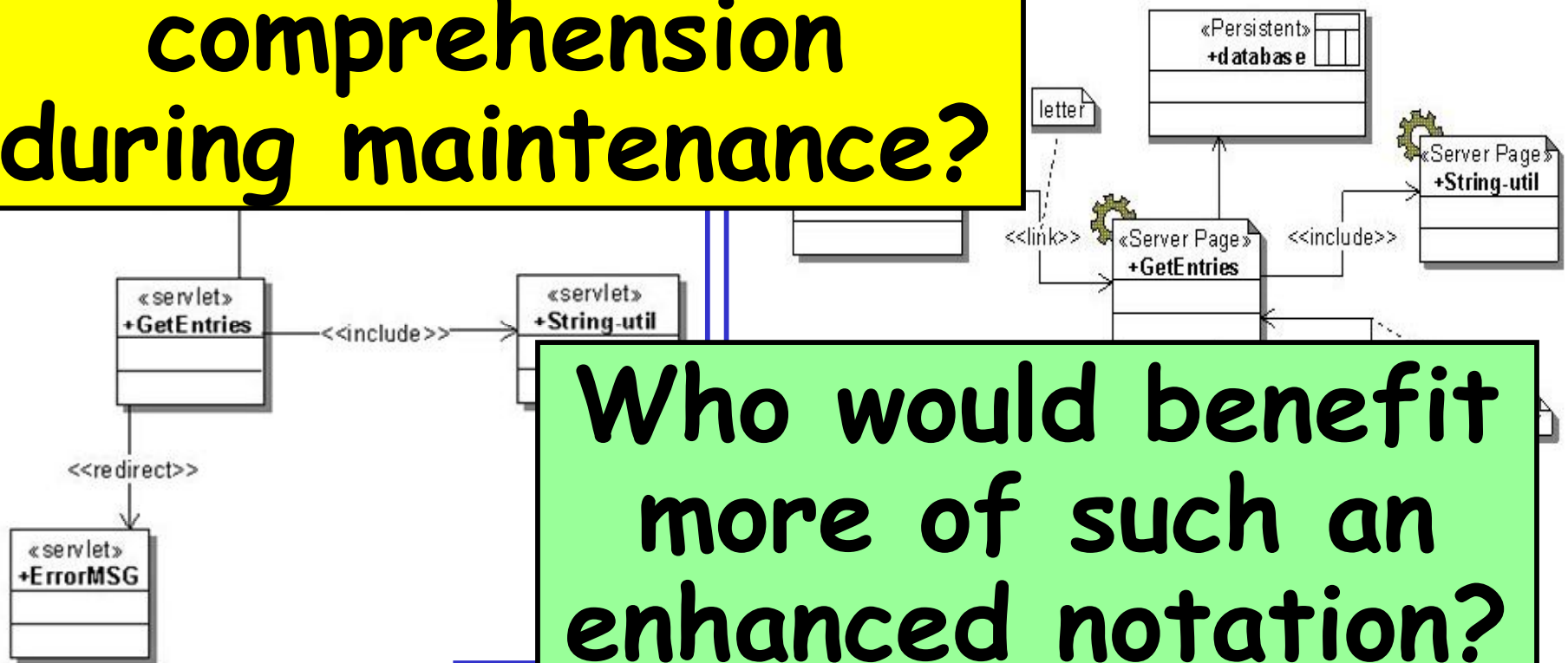
# Motivations

---

- General-purpose notations not adequate
  - **Solution: domain-specific languages**
- Example: Web applications
  - Several notation have been proposed
  - WebML, WSDM, OOHDM, or...
  - **WAE (Conallen's UML stereotypes)**
    - Extends basic UML with stereotypes that model Web application pages and their relationships

# Basic UML vs. Stereotypes

Does it enhance comprehension during maintenance?



Who would benefit more of such an enhanced notation?

Basic UML

Conallen's Stereotypes



# The experimental process

---

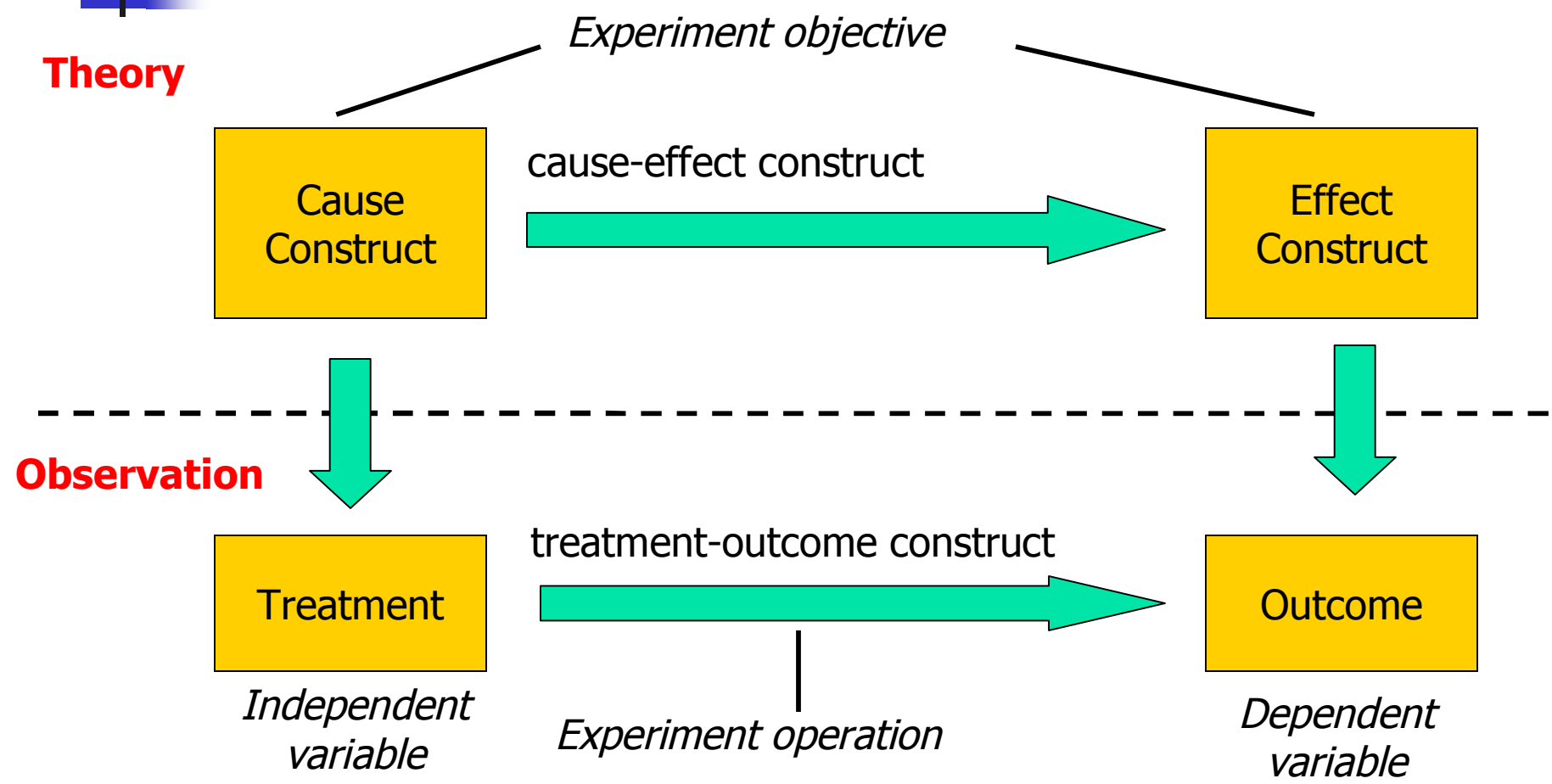


# Where do we start?

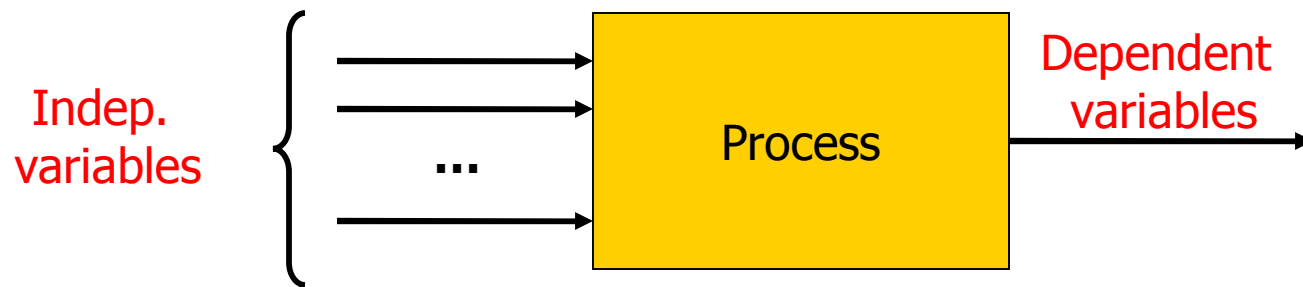
---

- We have an idea/conjecture about a cause effect relation
- We have a theory
- Thus we can formulate a hypothesis
- And to test it.. we run an experiment!

# Experiment principles



# Terminology - I



- **Dependent (or response) variables:** variables we are interested to study
- **Independent variables:** variables we control  
Example: evaluate productivity (**dependent variable**) based on development method, skills, tool (**independent variables**)



# Terminology – II

---

- The experiment studies how changes occurring on **independent variables** (**factors**) influence a **dependent variable**
- A **treatment** is a particular value for a factor
  - e.g. I want to study how effective is a new development method
  - (main) factor: development method
  - Treatments (2): the old method and the new one



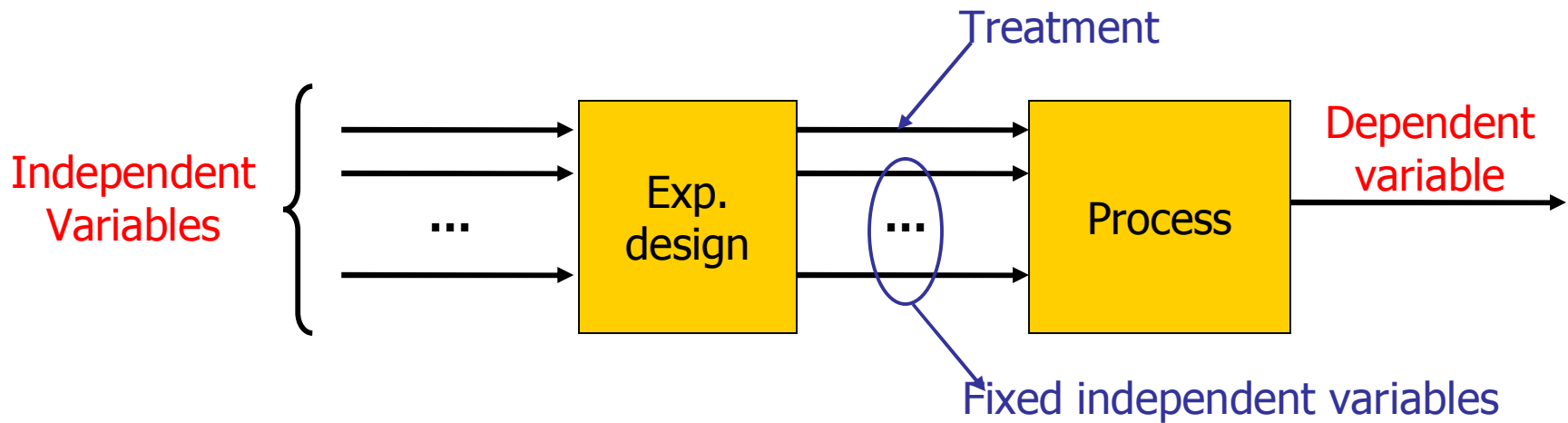


# Terminology – III

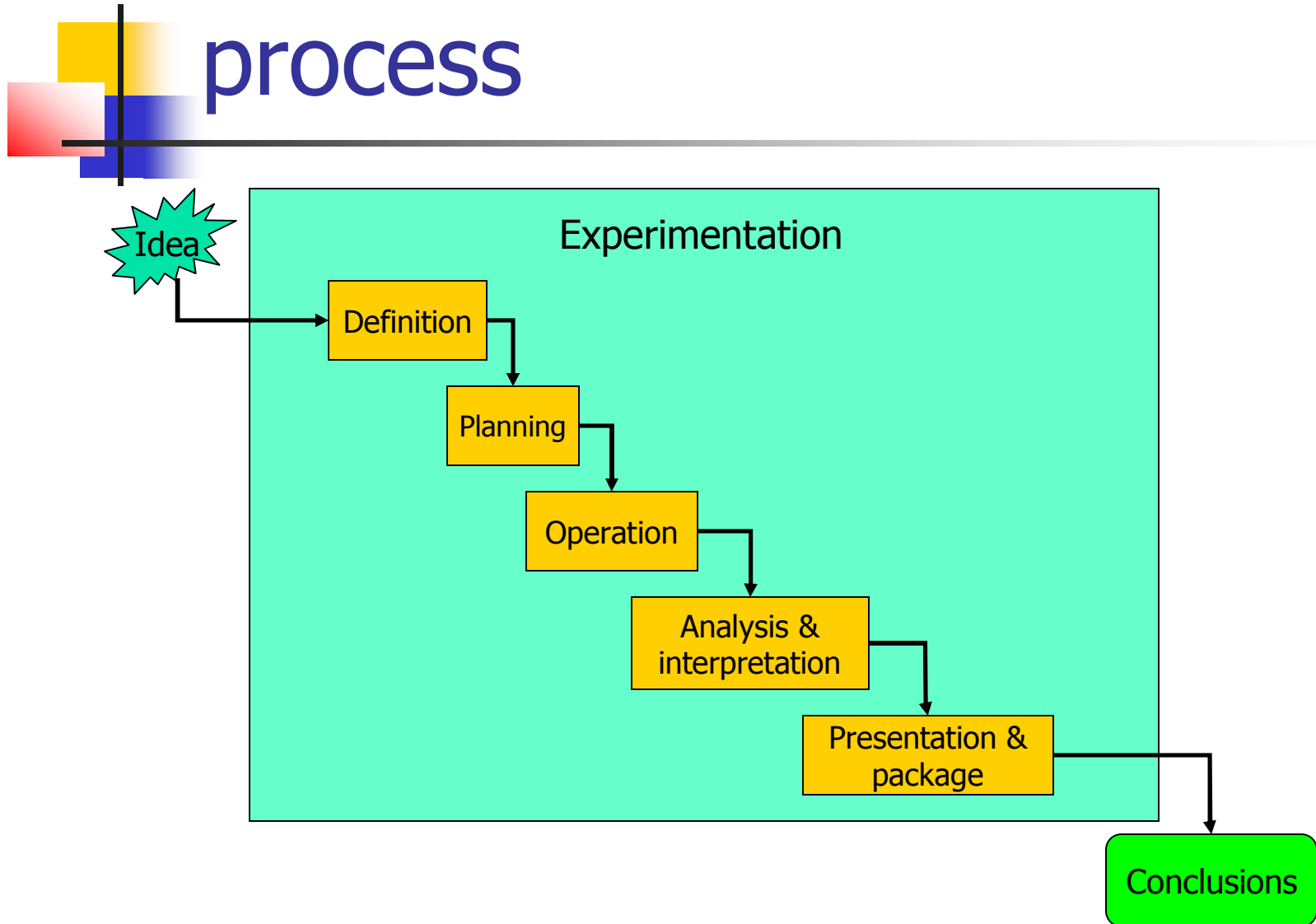
---

- A treatment is applied on a combination of **subjects** and **objects**
- An experiment is a set of **tests** (o **trials**) defined as combinations of treatments, subjects and objects
  - Joe (**subject**) uses the new development method (**treatment**) to develop the program A (**object**)
  - The number of tests influences the ability of making statistically significant conclusions

# Controlling the variables



# Steps of an experimental process





# Experiment Definition

---



# Definition Phase

---

- Based on Goal-Question Metrics  
[Basili, 93]
- Poses the basis for the experimentation
- Wrong definition → useless results





# Goal Definition Template

---

Analyze *<Object(s) of study>*  
for the purpose of *<Purpose>*  
with respect to their *<Quality focus>*  
from the point of view of the *<Perspective>*  
in the context of *<Context>*



# Goal Definition Template - II

---

- **Object of study:** Entity to study
  - Products, processes, theories, tools
- **Purpose:** intent of the experiment
  - Compare two techniques, characterize a learning process
- **Quality focus:** Effect to study
  - Effectiveness, cost, efficiency, precision...
- **Perspective:** from what point of view should I interpret the results?
  - Researcher, project manager, developer,...



# Goal Definition Template - III

---

- **Context:** environment where the study is carried out
  - **Subjects:** experience, specific skills, etc.
  - **Objects:** complexity, application domain, etc.





# Definition Framework

Object of study	Purpose	Quality focus	Perspective	Context
Product Process Model Metric Theory	Characterize Monitor Evaluate Predict Control Change	Effectiveness Cost Reliability Maintainability Portability	Developer Maintainer Project manager Corporate manager Customer User Researcher	Subjects Objects



# Example - I

---

- **Goal:** Analyze the use of stereotyped UML diagrams with the **purpose** of evaluating their usefulness in Web application comprehension **for different categories of users**
- **Quality focus:** high comprehensibility and maintainability
- **Perspective:** researchers, project managers
- **Context:**
  - Two Web apps: WfMS and Claros
  - Undergrad and Graduate students from Trento and Unisannio, researchers from both Trento and Unisannio



# Experiment Planning

---

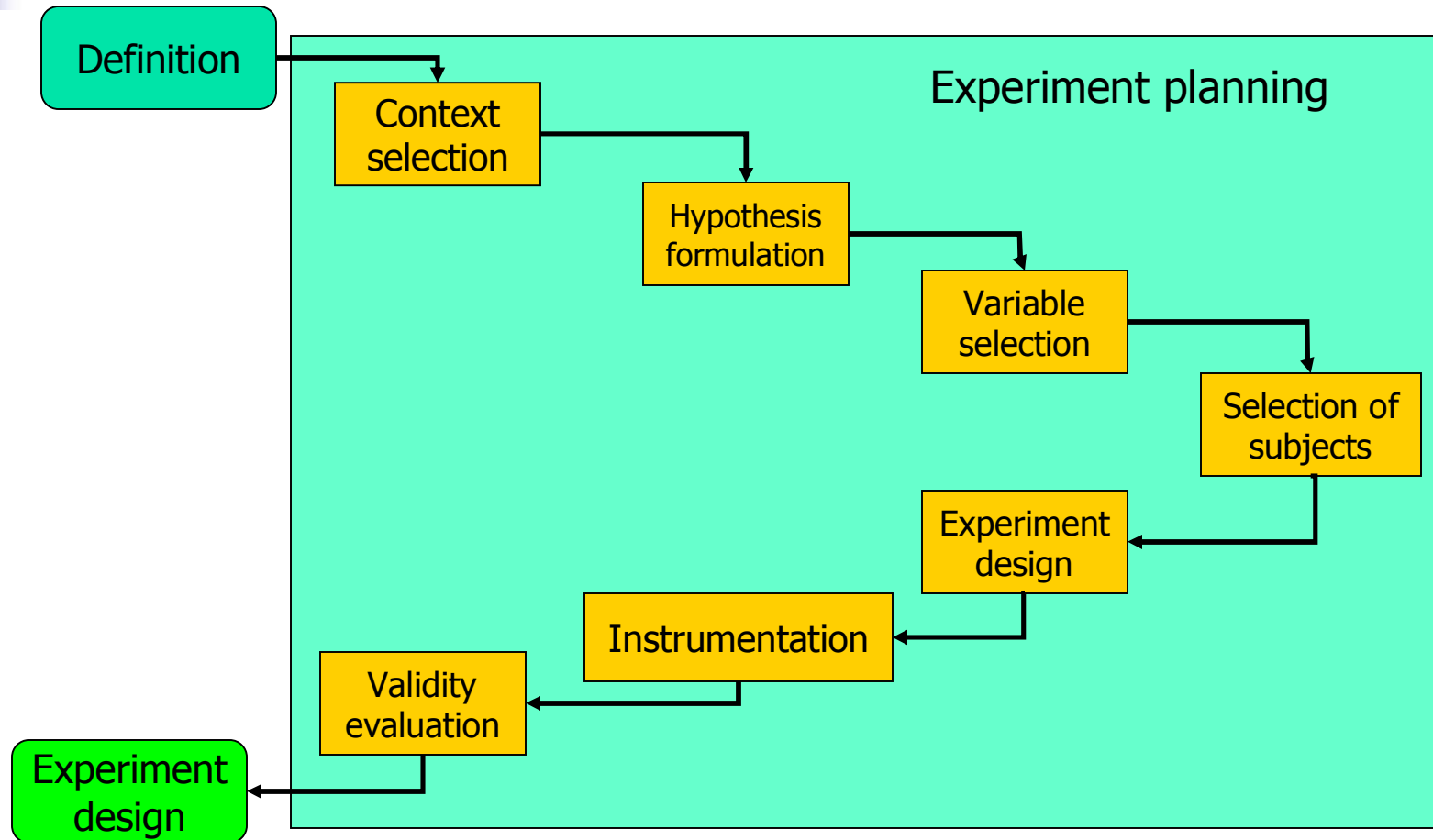


# Planning

---

- The Definition describes **why** we run an experiment
- The **planning** determines **how** the experiment will be executed
- Indispensable for any engineering task

# Planning: steps





# Context selection

---

- Set of **objects** and **subjects** involved in the experiment
- 4 dimensions
  - Off-line vs. on-line
  - Students vs. professionals
  - Toy problems vs. real problems
  - Specific vs. general



# Selection of objects

---

- In many experimental designs you might need more than one object
- Good to vary among domains
- But their complexity should not be too different
- The objects should be simple enough to allow performing the task in a limited time frame
  - But if possible avoid toy examples
  - (sub) systems of small-medium OSS
- Sometimes you have to prepare them
  - E.g. inject faults etc.
  - Be careful to avoid biasing the experiment
  - Check against mistakes in the objects, a major cause of experiment failures!



# Objects: Conallen study

- Two Web-based systems developed in Java
  - **WfMS** (Workflow management system)
  - **Claros** (Web mail system)

Claros			WfMS		
	Files	LOC		Files	LOC
Java	44	6288	Java	85	2378
JSP	34	1996	JSP	7	431
<b>Total</b>	78	8284	<b>Total</b>	92	2809





# Selection of Subjects

---

- Influences the possibility of generalizing our results
- Need to sample the population
- **Probabilistic sampling**
  - Simple random sampling, systematic sampling, stratified random sampling
- **Convenience sampling**
  - Just select the available subjects
  - ..or those more appropriate
- Often convenience sampling is the only way to proceed



# Experiments with students

---

- Very often the easiest way to run your experiments is to do it within a course
  - Need to go through an Ethical committee
  - If done well, it could be a nice exercise and students will appreciate!
- Hints:
  - Don't make it compulsory
  - Don't evaluate students on their performance in the experiment
  - Provide them a **little** reward for their participation



# Experiments with professionals

---

- More realistic
- ... but more difficult to achieve
- Suppose you want to do an experiment with 30 subjects, lasting 6 hours in total
  - How much does it cost?
- Hint
  - First, do experiments with students
  - Then you could do small replications with professionals
  - ... or case studies



# Experiments with students: the good and the bad

---

- Many students could be more expert and better trained on some techniques you want to experiment than professionals
- They have the same experience of junior developers
- The setting is different, students don't have the pressure industrial developers have when completing a project and meeting deadlines
- The experience is different from senior developers able to face off tough problems



# Assessing subjects

---

- Important to:
  - Influence the sampling (whenever possible)
  - Assign subjects with different experience and ability uniformly across experimental groups
    - i.e. avoid that one treatment is performed mainly by high ability or low ability subjects
- Ability could be assessed a-priori
  - Bachelor/laurea degree, previous exams grades
- Same for experience
- Better to discretize
  - Divide ability and experience in macro-categories
    - High, Low



# Pre-test

---

- Better way to assess ability
- **Option 1:** ask subjects to self-assess themselves
  - Easy, but could be subjective
- **Option 2:** ask subjects to perform a task related to what they will do in the experiment
  - E.g. understanding source code
  - Expensive to evaluate ☹️



# Example of pre-test

---

Please rate your knowledge of the following subjects:  
(Answers: 1. Very poor; 2. Poor; 3. Satisfactory; 4. Good; 5. Very good.)

- |   |   |                            |                            |                            |                            |                            |
|---|---|----------------------------|----------------------------|----------------------------|----------------------------|----------------------------|
| 1 | English                                 | 1 <input type="checkbox"/> | 2 <input type="checkbox"/> | 3 <input type="checkbox"/> | 4 <input type="checkbox"/> | 5 <input type="checkbox"/> |
| 2 | Java programming                        | 1 <input type="checkbox"/> | 2 <input type="checkbox"/> | 3 <input type="checkbox"/> | 4 <input type="checkbox"/> | 5 <input type="checkbox"/> |
| 3 | Eclipse IDE for Java                    | 1 <input type="checkbox"/> | 2 <input type="checkbox"/> | 3 <input type="checkbox"/> | 4 <input type="checkbox"/> | 5 <input type="checkbox"/> |
| 4 | Understanding/evolving existing systems | 1 <input type="checkbox"/> | 2 <input type="checkbox"/> | 3 <input type="checkbox"/> | 4 <input type="checkbox"/> | 5 <input type="checkbox"/> |

Please indicate the number of years you have been practicing the following activities:

- 5 Programming (any programming language): \_\_\_\_\_ years
- 6 Java programming: \_\_\_\_\_ years
- 7 Performing maintenance on an existing code base: \_\_\_\_\_ years



# Conallen study: subjects

---

- 74 among students and researchers
  - Exp I - Trento (13 Master students)
  - Exp II - Trento (28 Bachelor students)
  - Exp III - Benevento (15 Master students)
  - Exp IV – Trento/Benevento (8 Researchers)





# Hypothesis formulation

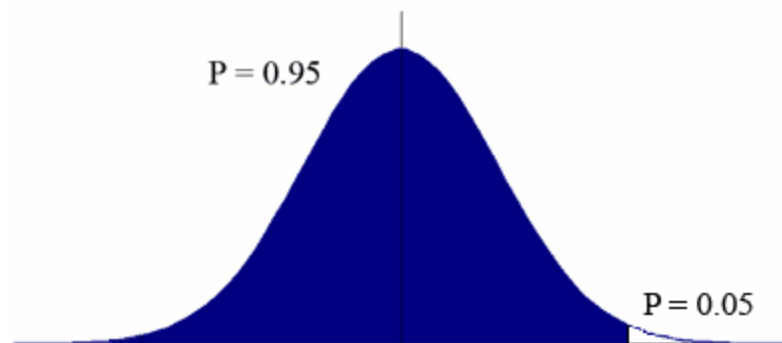
---

- The experiment aims at rejecting a null hypothesis
- We can reject the null hypothesis  
→ we can draw conclusions
- 2 hypotheses:
  - **Null hypothesis  $H_0$** : there do not exist trend/patterns in the experimental setting: the occurred differences are due to chances
    - **Example**: there is no difference in code comprehension with the new technique and the old one  $H_0 \mu_{\text{Nold}} = \mu_{\text{Nnew}}$
  - **Alternative hypothesis  $H_a$** : in favor of which the null hypothesis is rejected
    - **Example**: the new technique allows a better level of code comprehension than the old one  $H_0 \mu_{\text{Nold}} < \mu_{\text{Nnew}}$

# One-tailed vs. two-tailed

## One tailed:

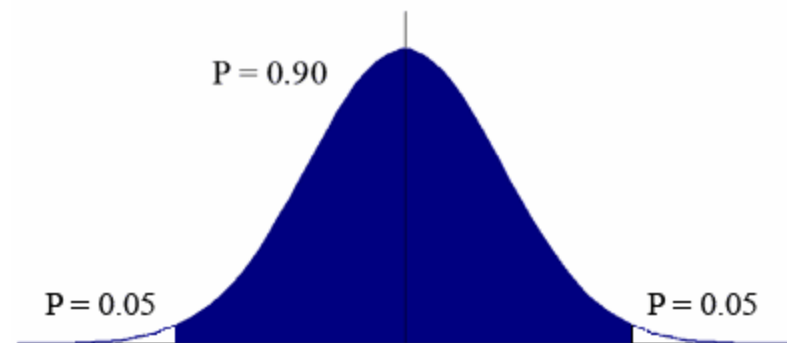
- We are interested to see whether **one mean was higher than the other**
  - Not interested in whether the first mean was lower than the other
  - One side of the probability distribution



## Two-tailed:

to see if **two means are different** from each other

We don't know a priori the direction of the difference  
Both sides of the probability distribution





# Examples

---

## One-tailed:

- We would like to see if additional documentation improves the software comprehension level
  - We don't care to test if this decreases the comprehension level
- We would like to see if complementing testing technique A with technique B would increase the number of faults discovered
  - We are testing the significance of the increment

## Two-tailed:

We would compare the effort/time needed to perform a task with two techniques

We don't know which one requires more time

Number of faults discovered with two different testing techniques

We don't know which one is better



# Example: Conallen study

---

- ◆ Null Hypotheses:
  - $H_0$ : use of stereotypes does not influence comprehension
    - One tailed
  - $H_{0e}$ : subjects' ability does not interact with the main factor
  - $H_{0a}$ : subjects' experience does not interact with main factor
  - $H_{0ea}$ : no interaction ability, experience, and main factor



# IMPORTANT!

---

- An experiment does not prove any theory, it can only fail to reject an hypothesis
- The logic of scientific discovery [Popper, 1959]
  - Any statement made in a scientific field is true until anybody can contradict it
- Thus...
  - Our experiments can only say something if they reject null hypotheses
  - If we don't reject our  $H_0$  we cannot really say that we reject  $H_a$
  - Well.. In practice we could do it after several replications...



# Variable selection

---



# Dependent variables...

---

- Used to measure the effect of treatments
- Derived from hypotheses
- Sometimes not directly measured → need for indirect measures
  - Validation needed, possible threats
- Need for specifying the measure scale
  - Nominal, ordinal, interval, ratio, absolute
- Need for specifying the range
  - If for different systems the variable assumes too different levels, need to normalize

$$M_{norm} = \frac{M - \min}{\max - \min}$$



# Nominal Scale

---

- Labeling/Classification
  - Any numerical representation of classes is acceptable
  - No order relation
  - Symbols are not associated to particular values





# Nominal Scale: Example

---

- Localize where a fault is located
  - requirement, design, code

$$M(x) = \begin{cases} 1 & \text{if } x \text{ is a specification fault} \\ 2 & \text{if } x \text{ is a design fault} \\ 3 & \text{if } x \text{ is a code fault} \end{cases}$$



# Ordinal Scale

---

- Order relation among categories
- Classes ordered wrt. an attribute
  - Any mapping preserving ordering is acceptable
  - E.g. numbers where higher numbers correspond to higher classes
- Numbers just represent rankings
  - Additions, subtractions and other operations are not applicable



# Ordinal Scale: Example

---

- Capture the subjective complexity of a method using terms “trivial”, “simple”, “moderate”, “complex” e “incomprehensible”
- Implicit “less than” relation
  - “trivial” less complex than “simple” etc.

$$M(x) = \begin{cases} 1 & \text{if } x \text{ is trivial} \\ 2 & \text{if } x \text{ is simple} \\ 3 & \text{if } x \text{ is moderate} \\ 4 & \text{if } x \text{ is complex} \\ 5 & \text{if } x \text{ is incomprehensible} \end{cases}$$



# Interval Scale

---

- Captures information about the size of intervals separating classes
- Preserves ordering
- Preserves the difference operation but does not allow comparisons
  - I can compute the difference between two classes but not the ratio
- Addition and subtraction allowed, multiplication and division not possible
- Examples: calendar, temperature scales
- Given two mappings  $M$  e  $M'$ , it is always possible to find 2 numbers  $a > 0$  e  $b$  such that:

$$M' = aM + b$$



# Interval Scale: Example I

---

- Temperature can be represented using Celsius or Fahrenheit scales
  - Same interval
  - The temperature in Rome increases from 20°C to 21°C
  - The temperature in Washington increases from 30°F to 31°F
  - Washington is not 50% warmer than Rome!
  - Transformation from C to F:
    - $F = \frac{9}{5} C + 32$



# Interval Scale: Example II

---

$$M_1(x) = \begin{cases} 1 & \text{if } x \text{ is trivial} \\ 2 & \text{if } x \text{ is simple} \\ 3 & \text{if } x \text{ is moderate} \\ 4 & \text{if } x \text{ is complex} \\ 5 & \text{if } x \text{ is incomprehensible} \end{cases}$$
$$M_2(x) = \begin{cases} 0 & \text{if } x \text{ is trivial} \\ 2 & \text{if } x \text{ is simple} \\ 4 & \text{if } x \text{ is moderate} \\ 6 & \text{if } x \text{ is complex} \\ 8 & \text{if } x \text{ is incomprehensible} \end{cases}$$
$$M_3(x) = \begin{cases} 3.1 & \text{if } x \text{ is trivial} \\ 5.1 & \text{if } x \text{ is simple} \\ 7.1 & \text{if } x \text{ is moderate} \\ 9.1 & \text{if } x \text{ is complex} \\ 11.1 & \text{if } x \text{ is incomprehensible} \end{cases}$$

- I can transform M1 into M3 using the formula
  - $M_3 = 2M_1 + 1.1$



# Ratio Scale

---

- Preserves ordering, size of intervals, and **ratio** between entities
- There is a null element (**zero attribute**) indicating the absence of a value
- The mapping starts from the zero value and increases with equal intervals (**units**)
- **Any arithmetic operation** makes sense
- Transformations are in the form

$$M = aM'$$

where **a** is a positive scalar



# Ratio Scale: Examples

---

- Length of an object in cm
  - An object is twice another
- Length of a program in LOC
  - A program is twice longer than another





# Absolute Scale

---

- Given two measures,  $M$  e  $M'$  only the identity transformation is possible
  - Measures performed just counting elements
  - Any arithmetic operation possible



# Absolute Scale: Examples

---

- Failures detected during integration testing
- Developers working on a project
- What about LOC?
  - If LOC measure the size of a program, they are in ratio scale
    - I could measure size differently (statements, kbytes...)
  - If they are just lines of code, then the scale is absolute



# Conallen study

---

- Dependent Variable: **comprehension level**
  - Assessed through a questionnaire
  - 12 questions per task
  - Covering both system specific and generic changes
- Subjects had to answer by listing items
  - Measured by means of Precision, Recall and F-Measure
  - Standard information retrieval metrics
  - Comprehension level is the mean across questions

# Questions

## Sample question:

**Q2:** Suppose that you have to substitute, in the entire application, the form-based communication mechanism between pages with another mechanism (i.e. Applet, ActiveX, ...).

*Which classes/pages does this change impact?*

**CORRECT ANSWER:** main.jsp, login.jsp, start.jsp

### Sample answer:

2	login.jsp, main.jsp, start.jsp, <del>complete.jsp, history.jsp, noenv.jsp</del>
---	---

$$\text{precision}_{s,i} = \frac{|C_i \cap A_{s,i}|}{|A_{s,i}|} = \frac{3}{6} = 0.5$$

$$\text{recall}_{s,i} = \frac{|C_i \cap A_{s,i}|}{|C_i|} = \frac{3}{3} = 1$$

$$F\text{-Measure}_{s,i} = \frac{2 \cdot \text{precision}_{s,i} \cdot \text{recall}_{s,i}}{\text{precision}_{s,i} + \text{recall}_{s,i}} = \frac{2 \cdot 0.5 \cdot 1}{1 + 0.5} = 0.67$$



# Independent variables

---

- Variables we can control and modify
  - Of course a lot depends on the experimental design!
- The choice depends on the domain knowledge
- As usual we need to specify scale and range
- One independent variable is the **main factor** of our experiment
  - Often one level for the **control group**
    - E.g. use of old/traditional technique/tool
  - One or more levels for **experimental groups**
    - E.g. use of new technique(s) tool(s)
- Other independent variables are the **co-factors**



# Co-factors

---

- Our main (experimented) factor is of course not the only variable influencing the dependent variable(s)
- There are other factors
  - Co-factors or sometimes confounding factors
- In a good experiment
  - limit their effect through a good experimental design
  - able to separate their effect from main factors
  - analyze the interaction with main factors
- Of course we would never account for all possible co-factors



# Conallen Study: Co-factors

---

- Main factor treatments: Pure UML vs stereotyped (Conallen)
- Co-Factors:
  - Lab {Lab1, Lab2}
  - Ability {High, Low}
  - Experience {Grad, Undergrad}
  - System {Claros, WfMS}



# Experiment design

---





# Experiment Design

---

- Is the set of treatment tests
  - Combinations of treatments, subjects and objects
- Defines how tests are organized and executed
- Influences the statistical analyses we can do
- Based on the formulated hypotheses
- Influences the ability of performing replications
  - And combining results



# Basic Principles - I

---

- Experimental design is based on three principles
  1. Randomization
  2. Blocking
  3. Balancing
- **Randomization:** observation must be made on random variables
  - Influences the allocation of objects, subjects and the ordering in which tests are performed
  - Useful to mitigate confounding effects
    - E.g. influence of objects, learning effect



## Basic Principles - II

---

- **Blocking:** sometimes some factors influence our results but we want to mitigate their effects
  - I can split my population in blocks with same (or similar) level of this factor
  - e.g. subjects' experience
- **Balancing:** I should try to have the same (or similar) number of subjects for each treatment
  - Simplifies the statistical analysis
  - Not strictly needed and sometimes we cannot achieve a perfect balancing



# Different kinds of design

---

- One factor and two treatments
- one factor and  $>2$  treatments
- Two factors and two treatments
- $>2$  factors, each one with two treatments



# One factor and two treatments

---

- **Notation**

- $\mu_i$ : dependent variable mean for treatment  $i$
- $y_{ij}$ :  $j$ -th measure of the dependent variable for treatment  $i$
- Example:
  - I'd like to experiment whether a new design produces less fault-prone code than the old design
    - Factor: design method
    - Treatments:
      1. New method
      2. Old method
  - Dependent variable: number of faults detected



# Completely randomized design

Subjects	Treatment 1	Treatment 2
1	X	
2		X
3		X
4	X	
5		X
6	X	

- Examples of hypotheses:
  - $H_0: \mu_1 = \mu_2$
  - $H_a: \mu_1 \neq \mu_2, \mu_1 < \mu_2$  or  $\mu_1 > \mu_2$
- Analyses
  - t-test (unpaired)
  - Mann-Whitney test



# Paired comparison design

Subjects	Treatment 1	Treatment 2
1	2	1
2	1	2
3	2	1
4	2	1
5	1	2
6	1	2

- Each subject applies different treatments
  - Need to have different objects
- Need to minimize the ordering effect
- Examples of hypotheses:
  - Given  $d_j = y_{1j} - y_{2j}$
  - Given  $\mu_d$  the mean of differences
  - $H_0: \mu_d = 0$
  - $H_a: \mu_d \neq 0, \mu_d < 0$  or  $\mu_d > 0$
- Analyses:
  - Paired t-test
  - Sign test
  - Wilcoxon



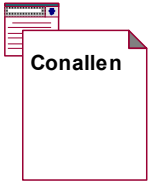
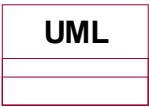
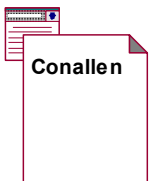
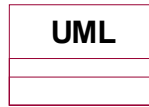
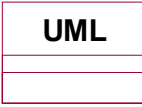
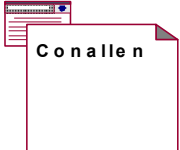
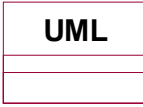
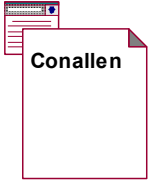
# How to instantiate it

---

- Subjects should work on different systems/objects on different labs
  - To avoid learning effects
- Different possible orderings of main factor treatments between labs
- Different possible orderings of systems/objects



# Example: Conallen

	Group 1	Group 2	Group 3	Group 4
Lab 1	 Claros	 Claros	 WfMS	 WfMS
Lab 2	 WfMS	 WfMS	 Claros	 Claros

- Subjects received:
  - Short description of the application
  - Diagrams
  - Source code



# One factor and $>2$ treatments

---

- Example:
  - Fault proneness wrt. Programming language adopted
    - C, C++, Java



# Completely randomized design

Subjects	Treat-ment 1	Treat-ment 2	Treat-ment 3
1		X	
2			X
3	X		
4	X		
5		X	
6			X

- Example of hypotheses:
  - $H_0: \mu_1 = \mu_2 = \mu_3 = \mu_a$
  - $H_a: \mu_i \neq \mu_j$  for at least one pair (i, j)
- Analyses:
  - ANOVA (ANalysis Of VAriance)
  - Kruskal-Wallis

# Randomized complete block design

Subjects	Treat-ment 1	Treat-ment 2	Treat-ment 3
1	1	3	2
2	3	1	2
3	2	3	1
4	2	1	3
5	3	2	1
6	1	2	3

- Example of hypotheses:
  - $H_0: \mu_1 = \mu_2 = \mu_3 = \mu_a$ 
    - $H_a: \mu_i \neq \mu_j$  for at least one pair (i, j)
- Analyses:
  - ANOVA (ANalysis Of VAriance)
  - Kruskal-Wallis
  - Repeated Measures ANOVA



# Two Factors

---

- The experiment becomes more complex
- The hypothesis need to be split into three hypotheses
  - Effect of the first factor
  - Effect of the second factor
  - Effect of the interaction between the two factors
- Notation:
  - $\tau_i$ : effect of treatment  $i$  on factor  $A$
  - $\beta_j$ : effect of treatment  $j$  on factor  $B$
  - $(\tau\beta)_{ij}$ : effect of interaction between  $\tau_i$  and  $\beta_j$
- Example:
  - Investigate the comprehensibility of design documents
    - Structured vs. OO design (factor A)
    - Well-structured vs. poorly structured documents (factor B)



# 2\*2 factorial design

		Factor A	
		Treatment A1	Treatment A2
Factor B	Treatment B1	Subjects 4, 6	Subjects 1, 7
	Treatment B2	Subjects 2, 3	Subjects 5, 8

- Examples of hypotheses:
  - $H_0: \tau_1 = \tau_2 = 0$
  - $H_a$ : at least one  $\tau_i \neq 0$
  
  - $H_0: \beta_1 = \beta_2 = 0$
  - $H_a$ : at least one  $\beta_j \neq 0$
  
  - $H_0: (\tau\beta)_{ij} = 0$  for each  $i, j$
  - $H_a$ : for each  $(\tau\beta)_{ij} \neq 0$
  
- Analysis:
  - ANOVA (ANALYSIS OF VARIANCE)



# Two-stage nested design - I

---

- Hierarchical design
- Useful when a factor is similar, **but not identical** for different treatments of the other factor
- Example:
  - Evaluate the effectiveness of a unit testing strategy
    - OO Programs and procedural programs (factor A)
    - Presence of defects (factor B)
    - Factor B is slightly different for OO and procedural code



# Two-stage nested design - II

---

	Factor A		
	Treatment A1	Treatment A2	
	Factor B		Factor B
Treatment B1'	Treatment B2'	Treatment B1''	Treatment B2''
Subjects: 1,3	Subjects: 6,2	Subjects: 7,8	Subjects: 5,4





# More than two factors

---

- Need to evaluate the impact of the dependent variable of different interacting co-factors
- Factorial design
- In the following we will consider examples limited to two treatments only



# $2^k$ factorial design

Factor A	Factor B	Factor C	Subjects
A1	B1	C1	2, 3
A2	B1	C1	1, 13
A1	B2	C1	5, 6
A2	B2	C1	10, 16
A1	B1	C2	7, 15
A2	B2	C2	8, 11
A1	B1	C2	4, 9
A2	B2	C2	12, 14

- Generalizes the  $2^2$  (# of factors  $k=2$ )
- $2^k$  treatment combinations



# $2^k$ fractional factorial design

---

- Disadvantage
  - Number of combinations increasing with the # of factors
- Therefore:
  - Some interactions could be useless to be analyzed
  - We could analyze only some combinations

# One-half fractional factorial design

Factor A	Factor B	Factor C	Subjects
A1	B1	C2	2, 3
A2	B1	C1	1, 8
A1	B2	C1	5, 6
A2	B2	C2	4, 7

Factor A	Factor B	Factor C	Subjects
A1	B1	C1	2, 3
A2	B1	C2	1, 8
A1	B2	C2	5, 6
A2	B2	C1	4, 7

- Considers half of the  $2^k$  design combinations
- Selection performed such that if a factor is removed, the remaining design is a  $2^{k-1}$  factorial design
- Two alternative fractions
  - Performed in sequence (replications) you will obtain a  $2^k$  factorial design



# One-quarter fractional factorial design

---

- One quarter of the  $2^k$  combinations
- If you remove 2 factors the remaining design is a  $2^{k-2}$  factorial design
- Dependences between factors
- Four alternatives
  - In sequence (replications) allow to obtain a  $2^k$  factorial design

# One-quarter fractional factorial design: Example

Factor A	Factor B	Factor C	Factor D	Factor E	Subj.
A1	B1	C1	D2	E2	3, 16
A2	B1	C1	D1	E1	7, 9
A1	B2	C1	D1	E2	1, 4
A2	B2	C1	D2	E1	8, 10
A1	B1	C2	D2	E1	5, 12
A2	B1	C2	D1	E2	2, 6
A1	B2	C2	D1	E1	11, 15
A2	B2	C2	D2	E2	13, 14

- D depends on a combination of A and B
  - We have D2 for each combination of A1 B1 or A2 B2
- Similarly, E depends on a combination of A and C

# One-quarter fractional factorial design: Example

Factor A	Factor B	Factor C	Factor D	Factor E	Subj.
A1	B1		D2		3, 16
A2	B1		D1		7, 9
A1	B2		D1		1, 4
A2	B2		D2		8, 10
A1	B1		D2		5, 12
A2	B1		D1		2, 6
A1	B2		D1		11, 15
A2	B2		D2		13, 14

- If I remove factors C and E (or B and D), it becomes a double replication of a  $2^{3-1}$  factorial design

# One-quarter fractional factorial design: Example

Factor A	Factor B	Factor C	Factor D	Factor E	Subj.
A1	B1	C1			3, 16
A2	B1	C1			7, 9
A1	B2	C1			1, 4
A2	B2	C1			8, 10
A1	B1	C2			5, 12
A2	B1	C2			2, 6
A1	B2	C2			11, 15
A2	B2	C2			13, 14

- If I remove D and E, it becomes a  $2^3$  full factorial design with factors A, B, C





# Experimental design: conclusions

---

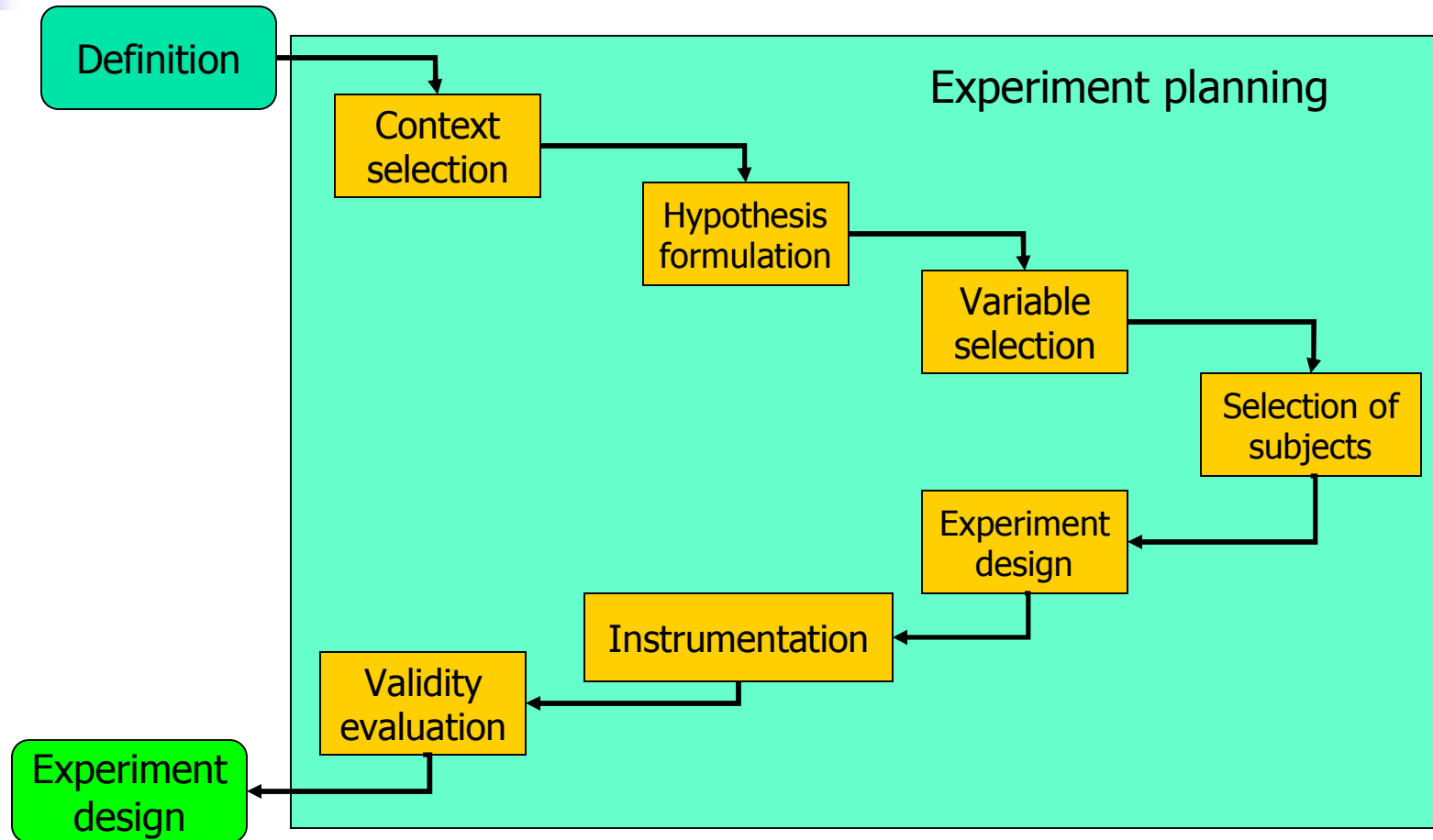
- Essential choice when doing an experiment
- Conclusions we may make depend on the kind of design we choose
- Constraints on statistical methods
- If possible, use a simple design 😊
- Maximize the usage of the available subjects
  - Often not many subjects available ☹️



# Validity evaluation

---

# Planning





# Validity evaluation

---

- Crucial questions in the analysis of experiment results are
  - To what extent are our results valid?
  - They should be at least valid for the population of interest
  - Then, if we could generalize...
- **Be careful: having limited threats to validity does not mean ability to generalize your results**
- Threats to validity [Campbell and Stanley, 63]
  1. Conclusion validity (C)
  2. Internal validity (I)
  3. Construct validity (S)
  4. External validity (E)

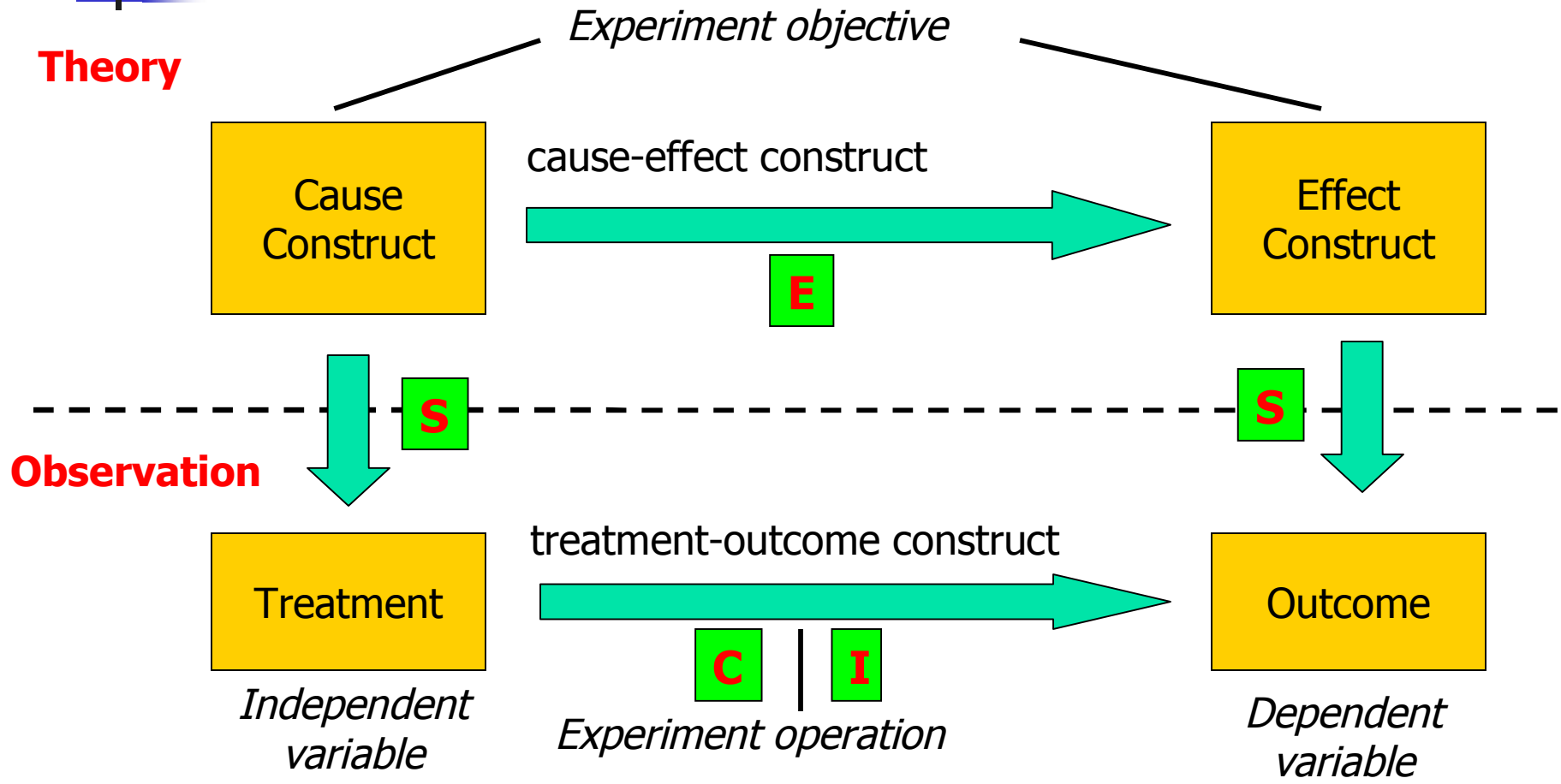


# Threats to validity

---

- **Conclusion validity (C):** concerns the relation between treatment and outcome
  - There must be a statistically significant relation
- **Internal validity (I):** concerns factors that can affect our results
  - We don't control nor measure it
- **Construct validity (S):** relation between theory and observation
  - The treatment should reflect the construct of the cause
  - The outcome reflects the effect construct
- **External validity (E):** concerns the generalization of results
  - If there is a causal relation between construct and effect, could this relation be generalized?

# Mapping Experiment Principles





# Conclusion validity - I

---

- **Low statistical power:**

- Results not statistically significant
- There is a significant difference but the statistical test does not reveal it due to the low number of data points

- **Violated assumptions of statistical tests**

- I use a test when I could not → Erroneous conclusions
- Many tests (e.g. t-test) assume normally distributed and linearly independent samples

- **Fishing and the error rate**

- I look at a particular result and use it to make conclusions
- Error rate: I do 3 tests on the same data sets with significance level 0.05 the actual significance level would be  $(1-0.05)^3=0.14$ 
  - When doing multiple mean comparisons with a two-means or two median test, I need to correct the p-value!



# Conclusion validity - II

---

- **Reliability of measures**
  - I repeat a measure under the same conditions and should obtain the same results
  - Could be due to wrong instrumentation
  - Prefer objective measures over subjective ones
- **Reliability of treatment implementation**
  - The treatment implementation could vary when applied to different subjects or in different context
  - e.g. highly experienced developers would prefer command line tools over graphical ones
  - Some tools can create issues on slow computers
  - I would limit the use of complex tools in experiments if possible!!!
- **Random irrelevancies in experimental setting**
  - External elements disturbing the experiment
  - Even noise outside your room 😊





# Conclusion validity - III

---

- **Random heterogeneity of subjects**
  - The effect of such an heterogeneity could hide the effect of the main factor treatment
  - Students are often more homogeneous than professionals
    - But I could have external validity problems



# Internal validity – Single group threats - I

---

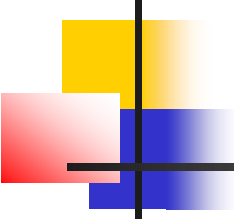
- **No control group**
  - Same group of subjects work with the old method and the new
- Not sure if the effect was caused by the treatment or by a confounding factor
- Single group threats:
  - **History:** experiment performed in particular time frames (after holidays, before exams...)
  - **Maturation:** as time passes subjects react differently
    - Tiresome effect, boredom effect, learning effect
  - **Testing:** if I perform a test twice with the same subjects, the second time they already know something about the task
    - Even if the task is slightly different



# Internal validity – Single group threats - II

---

- **Instrumentation:**
  - Unclear forms, imprecision in measurement instruments
- **Statistical regression:** suppose I do subjects' blocking based on a previous experiment
  - I assume a subject would not perform well based on a previous experiment, but it may not be the case for the new experiment
- **Selection:** due to the performance variability of subjects
  - Experiment with volunteers → better motivated persons than average, thus may not be representative



# Internal validity – Single group threats - III

---

- **Mortality:** some subjects may abandon the experiment
  - Would this affect the representativeness of our sample?
    - E.g. imagine I lose all high ability subjects
  - If a subject does not show up in multiple labs this limits the data points for paired analysis
- **Ambiguity about direction of casual influence:**
  - A causes B, B causes A, or X causes A and B?
  - e.g. correlation between complexity and fault-proneness
    - Complexity causes fault-proneness... (A)
    - Could it be that fault-prone code (B) tend to be on average more complex (A)?
    - Or else problem-specific factors (X) make code more complex (A) and fault-prone (B)



# Internal validity – Multiple group threats

---

- **Arise when studying different groups**
- A **control group** (on which I apply the old method) and an **experimental group** (on which I apply the new method) are influenced by single group threats
- Threats:
  - **Interactions with selection:** different group react in different way
    - The **maturation** effect influences more a group than the other
      - e.g. a group learns faster than the other
    - The **history** effect influences more a group than the other
      - e.g. I've experimented with two groups in different dates



# Internal validity – Social threats - I

---

- Applicable to single and multiple group experiments
- Threats:
  - **Diffusion or imitation of treatments:** the control group imitates the experimental group
    - e.g. the control group uses (even unconsciously) the new method assuming this would help to perform better
  - **Compensatory equalization of treatments:**
    - The control group should not be rewarded for using the old method
    - Nor we should let them use a third, alternative method



# Internal validity – Social threats - II

---

- **Compensatory rivalry:** subjects applying the less desirable treatment could be motivated to perform better
  - Thus you might see better results for the old method than for the new one
- **Resentful demoralization:** opposite of the previous problem
  - The most boring treatment could decrease performances or even cause abandonment
  - Sometimes subjects are better motivated with the new stuffs...



# Construct validity – Design threats - I

---

- Related to the experimental design and its possible influence on the study construct
- Threats:
  - **Inadequate preoperational explication of constructs:** construct not well defined before being translated into measures
    - Theory unclear
    - Comparing two methods, but not clear what does mean that a method is better than another
  - **Mono-operation bias:** I have one independent variable only, one single object or treatment
    - the experiment could not represent the theory
      - E.g. inspection conducted on a single document not representative of the set of documents on which the technique is often applied





# Construct validity – Design threats - II

---

- **Mono-method bias:** I use a single type of measure
  - If the measure is biased, it influences the results
  - e.g. I should use different clone detector, different measures for complexity...
- **Confounding constructs and levels of constructs:**
  - e.g. the (lack of) knowledge may or may not be a meaningful variable for the experiment
  - ...while the years of experience could be a meaningful variable!
- **Interaction of different treatments:** if I apply different treatments A and B to the same subject, the outcome could be due to treatment A, to treatment B, or to their interaction



# Construct validity – Design threats - III

---

- **Interaction of testing and treatment:**
  - If subjects are aware that I'm measuring their mistakes, they do their very best to avoid making any mistake
- **Restricted generalizability across constructs:** a treatment could act positively on a construct and negatively on others → difficult to generalize our results
  - The new method improves the productivity
  - However it reduces the maintainability, that I'm not measuring
  - What can I conclude? Does it make sense to propose the new method?



# Construct validity – Social threats - I

---

- **Hypothesis guessing:** in this case subjects could act
  - Positively, influencing the results towards the expected one
  - Negatively, contradicting the expected results
  - **Note: in experiments we should not really expect any result!!!**
- **Evaluation apprehension:** If I evaluate subjects based on how they perform in the experiment, this could bias the results
  - I'm experimenting a testing strategy, subjects might try to identify more faults without following the strategy



# Construct validity – Social threats - II

---

- **Experimenter expectancies:** the scientist could influence the experiment based on her expectancies
  - Questionnaire that “guides” the subjects towards answers aimed at validating her own theory
  - Someone else (not aware about the theory) should prepare the questionnaire



# External validity - I

---

- Threats:

- **Interaction of selection and treatment:** the population of subjects is not representative of the one for which I would like to generalize my results
  - Performing experiments with students to use results in industry
  - I do a code inspection experiment with programmers, while testers would behave differently than programmers
- **Interaction of setting and treatment:** the experimental setting or the material are not representative
  - E.g. I let the subjects using tools that they don't use in the reality
  - E.g. Web development using textual editors
  - I use toy objects



# External validity - II

---

- **Interaction of history and treatment:** I perform the experiment in particular days and this could influence the results
  - Questionnaire about the system reliability compiled the day after a major crash
- **In conclusion:** we should make the experimental environment as much realistic as possible



# Prioritize threats to validity

---

- Many threats are conflictual
  - When I reduce one threat, the other could increase
  - Quite an optimization problem
- Experiments with students
  - Larger samples, higher homogeneity  
→ good conclusion validity
  - Low external validity
- Use of different measures to make sure that treatment and outcome well represent the construct
  - Good construct validity
  - Conclusion validity problems  
→ errors due to multiple measures



# Conallen study: Threats to validity

---

- **Conclusion validity**
  - Statistical tests properly used
  - F-Measure as aggregate measure → precision and recall also checked separately
- **Construct validity**
  - Questionnaires as a measure of comprehension
  - Ability measure
- **Internal validity**
  - Abandonment → Paired tests on few subjects (enough), unpaired on all subjects
  - Learning effect → balanced by the design
- **External validity**
  - Different categories of students (and Universities) but...
  - Results need to be extended to professionals





# Operation

---

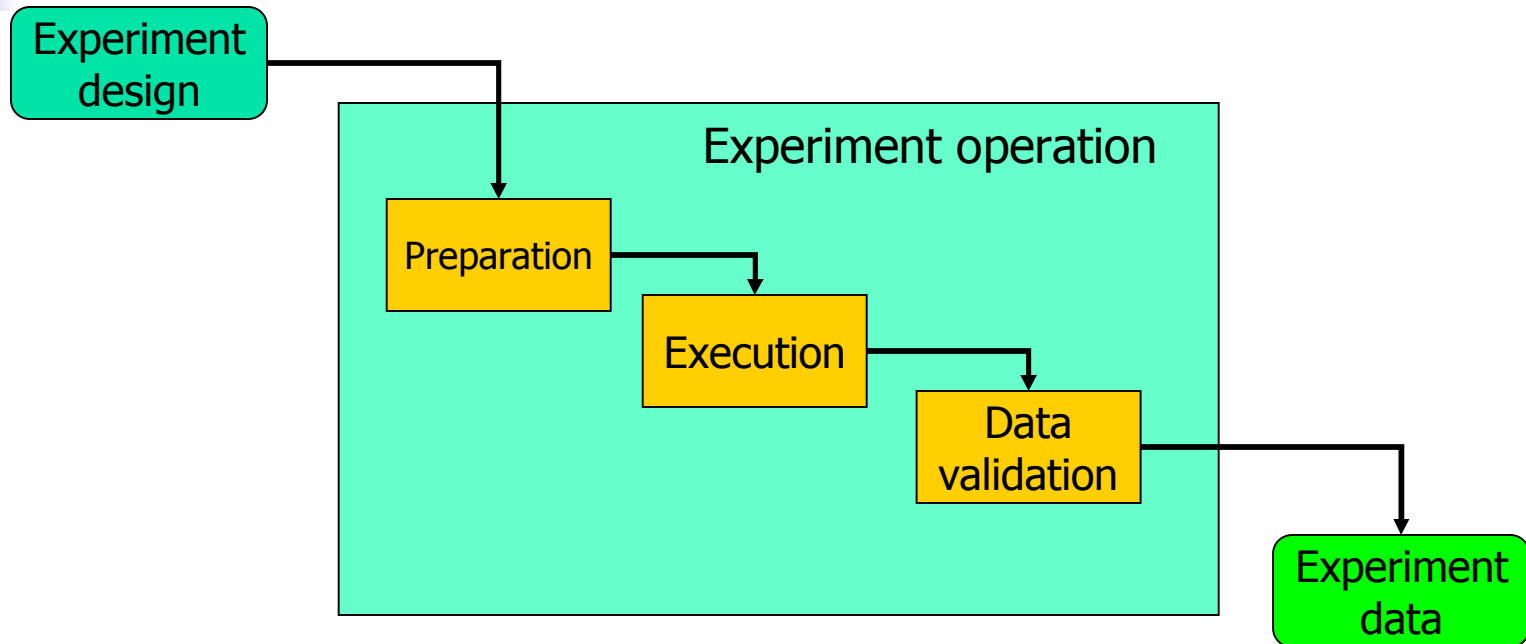


# Operation

---

- After having designed an experiment we need to execute it
- We are in touch with subjects for the first time
  - Besides the pre-experiment briefing and training
- Although the design and plan are perfect, everything depends on the operation
  - If something goes wrong in a couple of hours we could waste months of work... ☹️

# Experiment operation: steps





# Preparation

---

- **Obtain consent:**
  - Participants agree with the research objectives
    - But should not be aware of the hypotheses!
  - Explain how we will use the experiment results
  - Participation should not be compulsory
- **Confidentiality:** do not disseminate sensitive data
  - E.g. participants' productivity
- **Reward** in some way participation



# Preparation - Briefing

---

- Before the experiment, it is advisable to show a short presentation for:
  - Explaining the experiment and its objectives
    - Be careful to introduce any bias
    - Be careful to hypothesis guessing
  - Introduce the objects (briefly describe them, show class diagrams, etc.)
  - Introduce the instrumentation (form, tool, etc.)
    - Indicate where subjects can get forms/tools
    - Where they can get documentation...
    - Explain how to use tools if needed
  - Describe in detail the steps of the experiment
    - Be careful with tiny details: how to name files to be sent back, etc.



# Instrumentation

---

- Strongly influences the experiment outcome
- Tipi di instrumentation:
  - **Objects:** specification, code, documentation,...
  - **Guidelines:**
    - Description of the experimental process
    - Checklists
    - Guidelines need to be complemented with a proper training
  - **Measurement instruments:**
    - Paper-based forms, interviews, web-based tools
    - **Prepare questionnaires suitable for subjects' skills**
    - **Avoid intrusive questionnaires!**
    - **Results should not depend on the measurement instrument**



# Preparation: instrumentation

---

- Before the experiment the instrumentation must be ready
  - Objects, guidelines, tools, measurement instruments
    - Put data in a homogeneous format  
→ easier to analyze
    - Anonymous form if you don't need to identify the participant
      - But then you might not contact her/him anymore
  - If interviews are planned, prepare the questions before the experiment



# Conallen: Material

---

- Description of the application
- URL of the “running” Web application
- Source code of the application (URL)
- Paper Diagrams (Conallen/Pure-UML)
- Visual UML Diagrams (Conallen/Pure-UML)
- Questionnaire

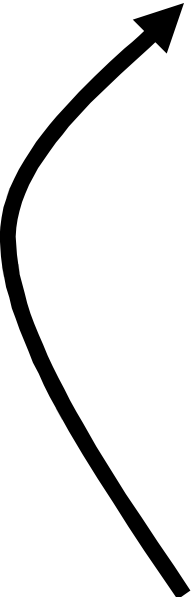


# Conallen:

## Experimental Procedure

---

1. Read the description of the application
  2. Read a question of the questionnaire
  3. Understand it
  4. Infer the answer using:
    - Diagrams
    - Code

(If you want you can execute the application)
  5. Compile the questionnaire
- 



# Execution

---

- Different ways to execute an experiment
  - Online
    - You can actually monitor the experiment
  - Offline
    - Distribute the task via email and wait for results



# Data collection

---

- **Manual:** analyze forms, artifacts produced by the subjects
  - Form vs. interviews
    - Forms do not require you to actively take part to the experiment execution
    - However interviews may reveal things that form could not reveal
- **Automatic:** web based forms, automated analysis
  - e.g. test case execution to analyze the correctness of the produced code
    - As in the Fit experiment



# Post-Experiment Questionnaire

---

- Used to understand:
  - Whether anything went wrong with clarity of objectives material, time available, tasks
  - How much time (approx) subjects spent on particular artifacts
  - Whether they “felt” a particular method easier/better than another
- Qualitative information
  - Used to explain quantitative results not to replace them!



# Building your questionnaire

---

- Responses using a Likert scale

1. Strongly disagree

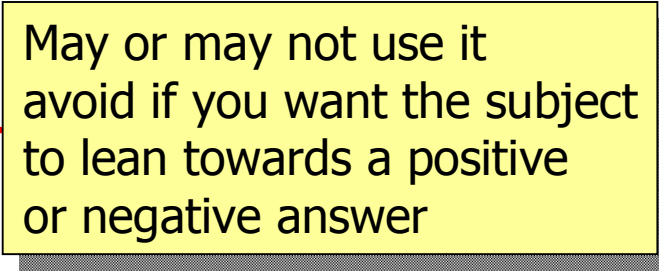
2. Weakly disagree

3. Uncertain

4. Weakly Agree

5. Strongly Agree

- NA Not applicable



May or may not use it  
avoid if you want the subject  
to lean towards a positive  
or negative answer



# Example: Conallen

---

Asked the subjects to assess:

- Clarity of
  - task objectives and
  - individual questions
- Difficulty in reading
  - diagrams
  - code
- Time spent on
  - diagrams
  - code
- For task with Conallen's notation
  - understandability of stereotypes
  - usefulness of stereotypes



# Example: Conallen

Issue	ID	Question
Objectives clarity	Q1	I had enough time to perform the lab tasks (1–5).
	Q2	The objectives of the lab were perfectly clear to me (1–5).
	Q3	The questions were clear to me (1–5).
Artifacts comprehension	Q4	I experienced no difficulty in reading the diagrams (1–5).
	Q5	I experienced no difficulty in reading the source code (1–5).
	Q8	I understood the meaning of Conallen’s stereotypes (1–5).
Cognitive effects	Q6	How much time (as a percentage) did you spend looking at class diagrams?
	Q7	How much time (as a percentage) did you spend for source code browsing?
	Q9	I found Conallen’s stereotyped diagrams useful (1–5).



# Pros and cons of survey questionnaires

---

- Help to better understand the subjects behavior during an experiment
- Can be used (of course) if you have developers available
- Controlled experiments, in vivo case studies
- Not possible for MSR studies
- Risk of bias very high
- Sometimes subjects tend to be overly positive or negative
- It remains a purely qualitative feedback
- Don't try to make strong conclusions based only on that





# Interviewing

---

- Alternative to survey questionnaire
  - Respondent better think about the answer 😊
  - ... but they could feel under pressure 😞
  - The questions can be adapted case by case 😊
  - ...but the risk is to have a too unstructured set of answers 😞
    - Difficult to make comparisons

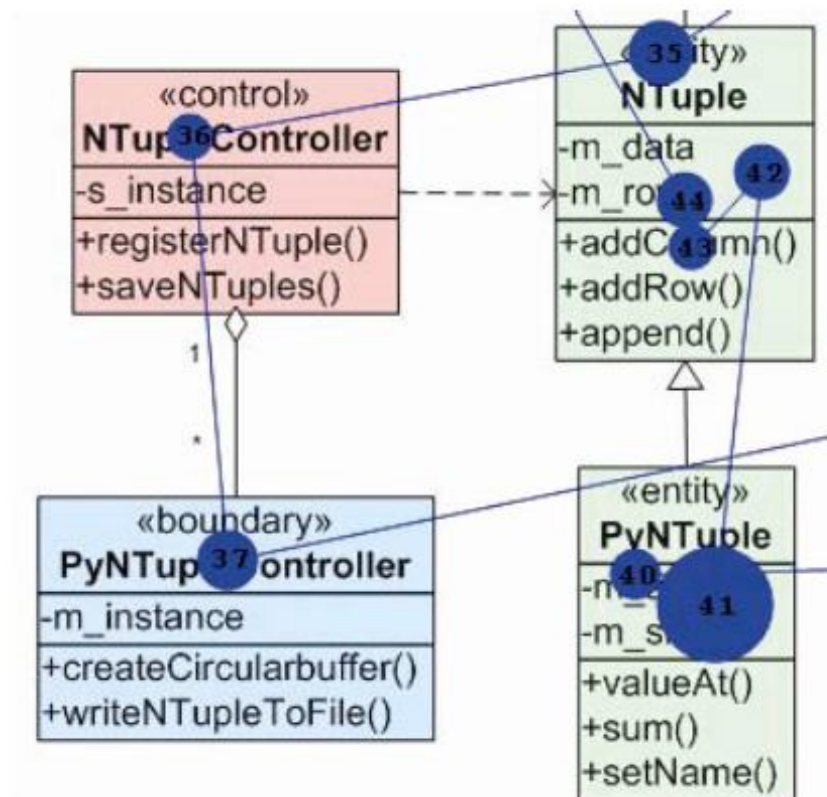


# Contacting team members

---

- Questionnaires/surveys cannot be applied to MSR studies
- However it is worth trying to contact project contributors / core project members
  - Instead of just “guessing”
  - They may or may not respond, but it costs nothing...

# Using Eye Tracking tools



Yusuf, S., Kagdi, H., and Maletic, J. I. 2007. Assessing the Comprehension of UML Class Diagrams via Eye Tracking. In *Proceedings of the 15th IEEE international Conference on Program Comprehension* (June 26 - 29, 2007)



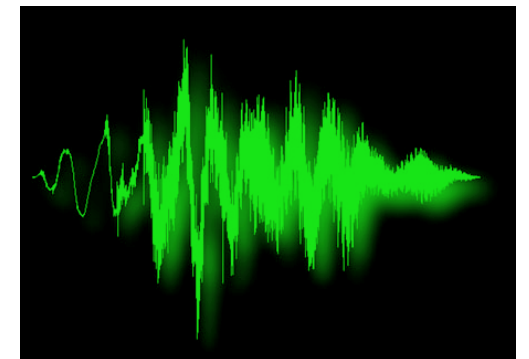
# Eye Tracking: Pros and Cons

---

- You can really record what subjects looked at during the study 😊
- Might be somewhat expensive 😐
- Very likely, you need to perform the study in sequence, with a few subjects only 😞
- Complex tasks difficult to be tracked 😞

# Other monitoring techniques

- Taping the session
- Recording (thinking aloud)
- Intercepting events on the machine
  - Diana Coman, Alberto Sillitti, Giancarlo Succi: A case-study on using an Automated In-process Software Engineering Measurement and Analysis system in an industrial environment. ICSE 2009: 89-99
- Problems:
  - Too invasive
  - Data analysis might require a huge amount of work





# Data validation

---

- Once the experiment has been completed, we need to do a consistency check on the collected data
  - Were treatments correctly applied?
  - Did subjects understand the provided forms?
  - Did subjects correctly fill the forms?
  - Remove subjects that
    - Did not participate to the experiment
    - Exhibited a weird behavior (e.g. did not pay attention to the task)
- Try to have a quick look at data as soon as possible
  - At least using descriptive statistics



# Replication

---

- “We do not take even our own observation quite seriously, or accept them as scientific observations, until we have repeated and tested them” (Popper, 1960)
- Replication is essential in experimentation
  - You should document your experiments so that others can
    - Repeat the experiment with different subjects
    - Replicate your statistical analysis
  - Look at the non-replicable case of the cold fusion!



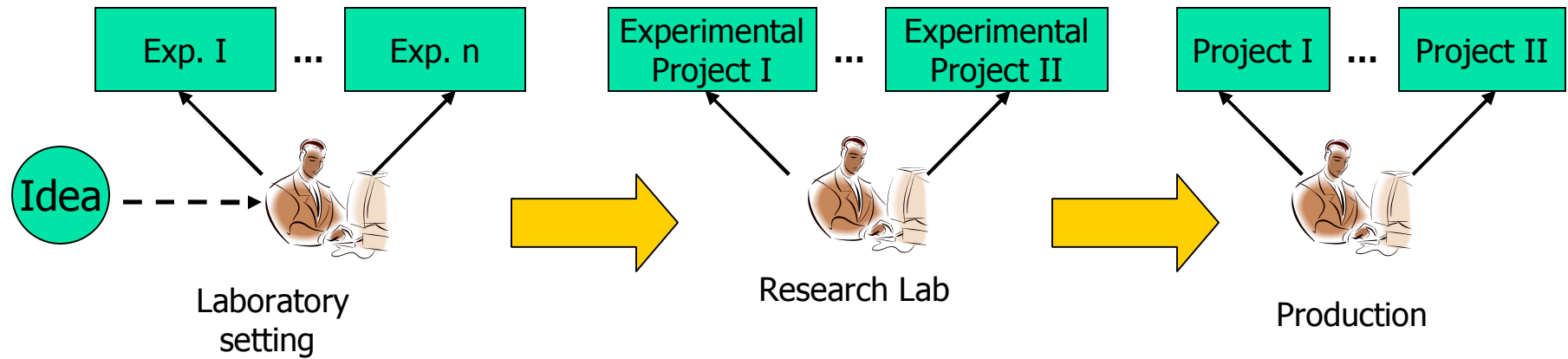
# Advantages of replications

---

- **Fix problems occurred in the first experiments**
  - Training not adequate
  - Tasks too complex
- **Consider a wide variety of subjects**
  - Different subjects in different replications
  - Sometimes different objects also
- **Increase the statistical power**
  - Possibility of analyzing results as a whole or do meta-analysis



# Consolidating ideas



Many replications every time



# Conclusions

---

- Software engineering/development is a human-intensive activity
- Surely you can assess your new approach using various measures but.....
- .... to really show the usefulness/efficiency/\* of a technique, you should see how developers perform with that technique

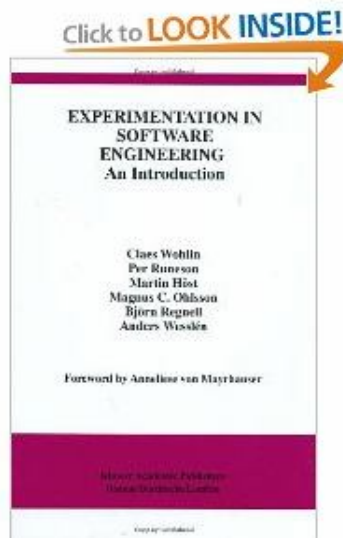


# Conclusions

---

- Experiment definition and planning is crucial
  - Wrong definition → you're studying something irrelevant
  - Design influences the kind of analyses you can do on your data
- Experiment operation needs to be carefully performed
  - You can lose months of work in one shot
- Carefully analyze the threats to validity
  - Don't be afraid of doing that...there's no perfect study

# Suggested Readings - I

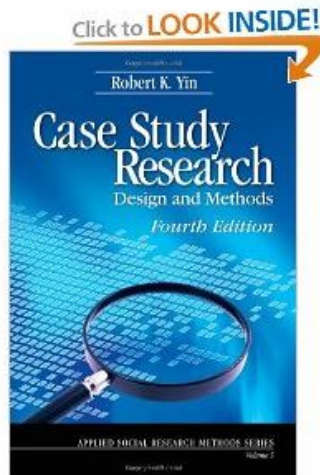


- **Experimentation in Software Engineering: An Introduction**  
Claes Wohlin, Per Runeson, Martin Höst ,  
Springer, 1999

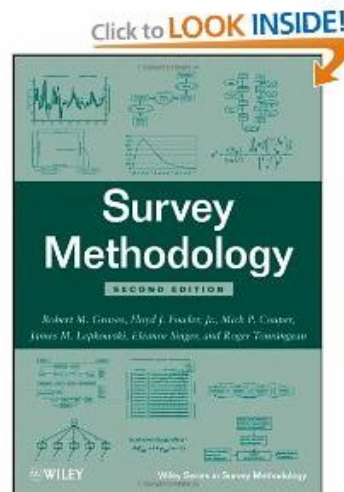


- **Basics of Software Engineering Experimentation**  
Natalia Juristo, Ana M.  
Moreno, Springer, 2010

# Suggested Readings - II



- **Case Study Research: Design and Methods**  
Robert K. Yin, Sage Publications, Inc; 4th edition (October 31, 2008)



- **Survey Methodology**  
Robert M. Groves, Floyd J. Fowler Jr., Mick P. Couper, James M. Lepkowski, Eleanor Singer, Roger Tourangeau, Wiley; 2 edition (July 14, 2009)

# References & Acknowledgment

- Massimiliano Di Penta, Experimentation in Software Engineering: Theory and Practice.