Course "Empirical Evaluation in Informatics"

Controlled Experiments

Prof. Dr. Lutz Prechelt
Freie Universität Berlin, Institut für Informatik
http://www.inf.fu-berlin.de/inst/ag-se/

- Example 1: flow charts
- Control and constancy
- Threats to constancy
- Techniques for achieving constancy
- Example 2: design pattern documentation

"Empirische Bewertung in der Informatik"

Kontrollierte Experimente

Prof. Dr. Lutz Prechelt
Freie Universität Berlin, Institut für Informatik
http://www.inf.fu-berlin.de/inst/ag-se/

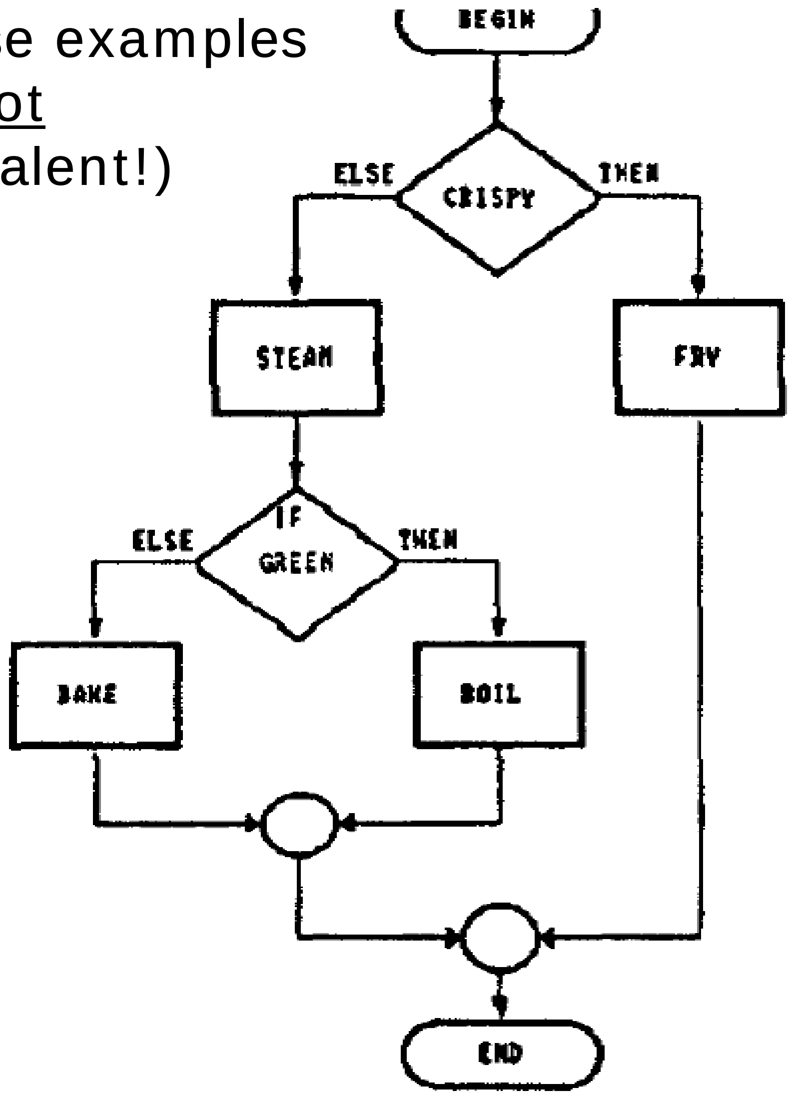
- Beispiel 1: Flussdiagramme
- Kontrolle und Konstanz
- Probleme für Konstanz
- Techniken zum Erreichen von Konstanz
- Beispiel 2: Entwurfsmuster-Dokumentation

Example 1: Flowcharts vs. Pseudocode

- Source: David Scanlan: *"Structured Flowcharts Outperform Pseudocode: An Experimental Comparison"*, IEEE Software, September 1989
- Question: Is an algorithm easier to comprehend if presented as a flow chart or if presented as pseudocode?
- Study format: Controlled experiment

Flowchart, Pseudocode

- (These examples are <u>not</u> equivalent!)




```
PROC
  IF GREEN
   THEN
     BAKE
   ELSE
     BOIL
     IF CRISPY
      THEN
        STEAM
      ELSE
        FRY
     ENDIF
  ENDIF
END PROC
```

Experiment rationale

- Earlier experiments by Shneiderman et al. on the same question had not found any differences

- Scanlan criticizes these experiments:
  - Have measured only correctness, not work time
  - Some questions could not be answered from flowchart alone
  - Program was too simple

- Scanlan attempts to create experiments without these flaws

Experiment setup

- Subjects: 82 MIS majors (junior to graduate)
- Independent variables (inputs):
  - program complexity/length: simple, medium, complex
  - presentation type: flowchart, pseudocode
  - therefore, there are 3*2 = 6 experiment groups
- Subjects study an algorithm and answer a fixed set of comprehension questions
  - 6*2, 9*4, 10*6 questions for simple, medium, complex alg.
  - Example questions:
    - *"What are the values (true/false/unknown) at all decisions in the algorithm when the vegetable is boiled?"*
    - *"What are the values at all decisions in the algorithm when the vegetable is both boiled and steamed?"*
    - (all questions are of this type)
- Experiment is run fully automatically
  - by a computer with speech output

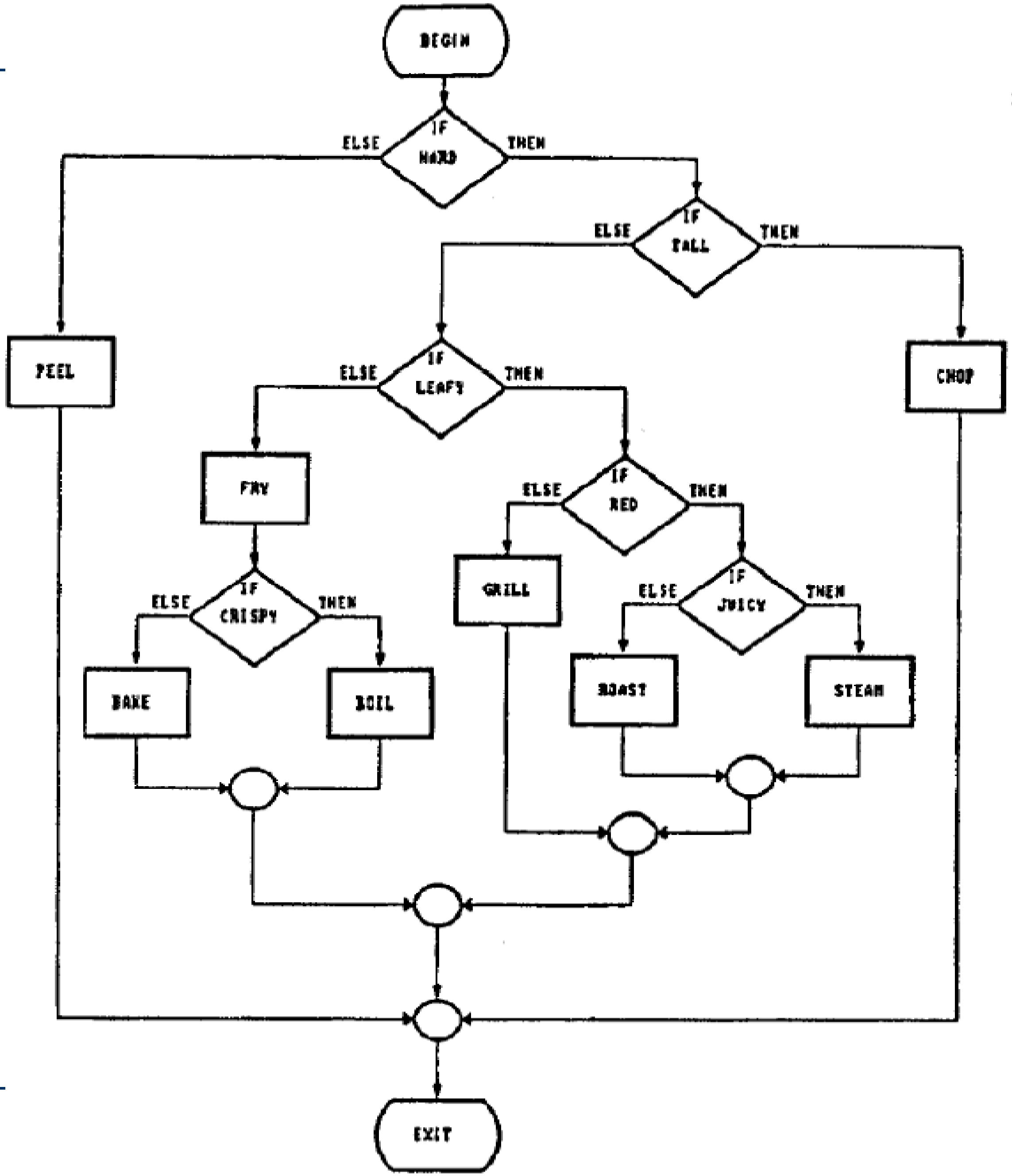
Experiment setup (2)

- Flowcharts and pseudocodes are each printed on a single sheet of paper
- A mechanical machine switches between algorithm sheet and question/answer sheet
  - only one is visible at any time
  - subject can switch as s/he pleases
- Dependent variables (outputs):
  - algorithm view time
  - question answering time
  - number of algorithm views
  - percentage of correct answers
  - subjective confidence in the answers

Experiment setup (3)

- Each subject is part of all six groups
  - leads to 6*82 = 492 data points overall
  - This is possible because the algorithms use randomized combinations of verbs and adjectives
    - (What would be the problem otherwise?)

Complex algorithm




```
PROC
  IF RED
    THEN
      IF TALL
        THEN
          FRY
        ELSE
          IF HARD
            THEN
              IF CRISPY
                THEN
                  IF JUICY
                    THEN
                      PEEL
                    ELSE
                      GRILL
                  ENDIF
                ELSE
                  BOIL
              ENDIF
            ELSE
              ROAST
              IF LEAFY
                THEN
                  CHOP
                ELSE
                  BAKE
              ENDIF
          ENDIF
      ENDIF
    ELSE
      STEAM
  ENDIF
END PROC
```

Results

1. The subjects in the flowchart groups
   1. require less algorithm view time
   2. require much fewer algorithm views
   3. provide more correct answers
   4. have higher confidence in their answers

2. The differences tend to become more pronounced with increasing algorithm complexity

Table A.
Percentage of correct answers to all question parts.

Complexity level	IV	Total parts	% correct (means)	*s*	*t*	*df*	*p*
Simple	FC	12	97.97	8.50			
	PC	12	93.80	10.90	2.77	81	.0035
Medium	FC	36	98.81	3.40			
	PC	36	94.92	10.30	4.05	81	.0000
Complex	FC	60	98.68	3.50			
	PC	60	91.71	14.40	4.82	81	.0000

FC = flowchart; PC = pseudocode; IV = independent variable; *s* = standard deviation (in seconds); *t* = correlated *t*-test result; *df* = degrees of freedom; *p* = probability.

Discussion: Internal validity / credibility

- The internal validity of this experiment is very high
  - We can be confident to find similar results if we repeated the experiment
- Problems avoided by this experiment setup:
  - accidental group differences
    - by using large groups and an intra-subject design
  - measurement errors
    - by fully automatic measurement mechanism
  - accidental experimenter influence on subject motivation
    - by fully automatic experiment guidance (speech output etc.)
  - and more
    - e.g. by using a shielded room, by having practice sessions
- The only remaining question:
  - Are the subjects equally well trained in both notations?

Discussion: External validity / credibil.+relevance

- The external validity of this experiment is very problematic:
  - Issues with the structure of the algorithms
  - Issues with the meaning of the algorithms
  - Issues with the size of the algorithms
  - Issues with the number of questions (in relation to algorithm size)
  - Issues with the type/content of questions

External validity: Task too simple

Table 1.
Number of seconds subjects looked at algorithm when answering each question part.

Complexity level	IV	Total parts to questions	Means (sec./part)	*s*	*t*	*df*	*p*
Simple	FC	12	7.83	5.09			
	PC	12	13.44	7.75	6.47	81	.0000
Medium	FC	36	6.19	3.02			
	PC	36	11.71	6.50	9.43	81	.0000
Complex	FC	60	6.33	2.37			
	PC	60	15.80	10.98	8.45	81	.0000

FC = flowchart; PC = pseudocode; IV = independent variable; *s* = standard deviation (in seconds); *t* = correlated *t*-test result; *df* = degrees of freedom; *p* = probability.

External validity: Too many questions (2)

Table D.
Mean number of times the algorithm was viewed when answering each part of all questions.

Complexity level	IV	Total parts	Times/ part	*s*	*t*	*df*	*p*
Simple	FC	12	1.30	.275			
	PC	12	1.41	.344	3.25	81	.0008
Medium	FC	36	.86	.239			
	PC	36	.92	.289	2.84	81	.0030
Complex	FC	60	.72	.229			
	PC	60	.82	.296	4.55	81	.0000

Methodology of controlled experiments

- "Experiment": Latin 'experimentum' (attempt, trial, experience)
  - means to try something out, to manipulate the situation

- **Control** refers to the construction of a repeatable situation
  - rather than one that has many arbitrary or even unknown attributes
  - Assume the situation can be fully characterized by N attributes
  - Then we want to experiment with k of them (often k=1)
    - We manipulate them: These are the **independent variables**
  - If we keep the other N-k attributes constant
    - These N-k attributes are called **extraneous variables**
    - The purpose of control is achieving *constancy*
  - we understand the effects of changing the independent variables.
    - The effects are defined by the observed **dependent variables**

Constancy in the natural sciences

- In basic physics or chemistry it is often relatively easy to achieve constancy
  - Although it may be difficult to set the independent variables to the values one wishes to explore
    - e.g. temperature and pressure for nuclear fusion

- The most difficult problem historically is finding out what attributes are relevant
  - i.e. what is not just an unimportant extraneous variable
  - e.g. understanding the nature of infectious diseases

Constancy with human beings

- In contrast, whenever human beings are part of the experiment, constancy becomes extremely difficult:
  - No two human beings are the same
  - No one human being is the same over time (memory!)

- The only known approach to obtain constancy for the human-related attributes of an experiment is averaging:
  - Pick a large number of humans ("subjects") at random
  - Assign each to an experiment condition at random
  - Perform the experiment with each one
  - Use the average results per **group**: differences balance out

- It works, except for one problem:
  - Subject motivation may depend on the value of the experimental variable
    - e.g. design method A is considered more 'sexy' than B

Threats to constancy

- Individual differences
  - The largest and most important effect in most human-related informatics experiments
  - e.g. capability, endurance, motivation
- History
  - Long-running experiments are influenced by outside events
- Maturation
  - Subjects learn and change during an experiment
- Instrumentation
  - Human observers change during an experiment
  - Technical measurement infrastructure may also change
- Mortality
  - Not all subjects stay until the end of the experiment
    - and drop-out probability may be related to the experiment variable

Threats to constancy (2)

- Experimenter influence
  - Experimenter handles subjects of different groups (or the data collected about them) in a biased way
- Sequence effects
  - The influence if the same subject solves more than one task
    - The order can influence the results
  - E.g. learning, tiring, boredom
- Sophistication
  - If subjects understand what the experiment is trying to find out, that can influence the result
  - e.g. unrealistic focus on one aspect of a task

If any of these occur, they must occur equally (on average) in each experiment group

Constancy in medicine: double blind testing

- The averaging method for achieving constancy can be applied to perfection in drug testing
  - We want to compare two medicines A and B
    - Or even A to doing nothing: use a placebo
  - A subject does not know which one s/he receives ("blinding")
  - The doctor does not know which one s/he applies ("blinding")
  - This is called a "double blind" experiment
    - But mortality can still be a big problem

- Unfortunately this approach is almost never applicable in informatics
  - You cannot apply a technique without knowing
  - So we almost always need to consider motivation differences as a threat to constancy and hence to internal validity

Techniques for achieving constancy

- Randomization
  - balances individual differences to achieve constancy
- Matching
  - reduces the impact of individual differences, maximizes constancy
- Counterbalancing
  - compensates sequence effects

Randomization

- Subjects must not assign themselves to the experiment conditions based on personal preferences
  - May produce bias
  - e.g. the more capable subjects may be more interested in the design method that appears more 'modern'
    - whereas the less capable ones rather stick with the familiar

- Experimenters also must not assign subjects based on whatever kinds of preferences
  - May produce bias; e.g. may assign the more capable subjects to his/her favorite method – even unconsciously

- Random assignment is the only method for avoiding bias
  - But may be very difficult, e.g. because not all subjects have the required knowledge for all experiment conditions
    - Without random assignment, the study becomes a quasi-experiment

Matching

- Random assignment needs not make each single assignment from the whole pool of remaining subjects
  - Instead, we may pre-group 'similar' subjects into tuples of j (for j experiment conditions) and randomize over one tuple at a time
  - This is called matching
- Matching may increase group similarity and may effectively reduce individual variation across the groups
- Example:
  - Order the subjects by expected design capability
  - Take the next best 2 at each time
  - Assign one to method A and one to B randomly
- Matched samples improve the sensitivity of statistical analysis

Counterbalancing

- Often subjects need to perform more than one task
  - because suitable subjects are rare,
    because instructing them is expensive, etc.
- This will produce sequence effects
  - learning, tiring, etc.
- To compensate these effects:
  - Have the same number of subjects perform the tasks in each of all possible task orders
  - for each of the experiment conditions or orders of experiment conditions
    - usually realistic only for 2 tasks

Counterbalancing: example

A typical experiment plan in informatics is as follows:

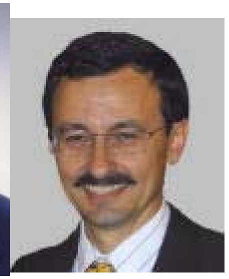
- We want to compare design methods A and B
- We use two different tasks 1 and 2
- Each subject solves both tasks
  - Solving one task twice (once with each method) makes no sense
    - due to learning (sequence effect)
- Experiment groups:
  - (group: first task, second task)
  - **G1: A1, B2**
  - **G2: A2, B1**
  - **G3: B1, A2**
  - **G4: B2, A1**

Example 2: Design pattern documentation

- Prechelt, Unger, Philippsen, Tichy: *"Two Controlled Experiments Assessing the Usefulness of Design Pattern Documentation in Program Maintenance",* IEEE Transactions on Software Engineering, June 2002

- Situation: You have programs that use/contain design patterns. The programs (source code) are well commented, but no separate design documentation exists.
Now the programs must be modified.
- Question: *Does understanding and modifying the programs become easier if the design pattern usage is documented explicitly?*





Experiment variable

- The independent variable of this is whether or not PCLs were added to an already well-documented program
  - PCL: Pattern Comment Line
    A comment section that explicitly describes how a particular program element participates in a pattern
  - Example: lines 484 and 485 are PCLs

```
477  /**
478         NTTupleDisp2 displays NTTuple, where
479         1. Tuples with an empty telephone number are left out and
480         2. Tuples are sorted by (last)name
481         Using Tuple objects of other Tuple types results in
482         ClassCastException.
483         *** DESIGN PATTERN: ***
484         NTTupleDisp2 completes the **Template Method** newTuple()
485         of TupleDispA
486  */
487  final class NTTupleDisp2 extends TupleDispA {
```

Experiment tasks

- The subjects worked on two different programs
  - Phonebook: A trivial phonebook management application with two different views of the data
    - Uses the 'Observer' and 'Template Method' design patterns
  - And/Or tree: A library (plus simple application) for handling AND/OR trees of Strings
    - Uses the 'Composite' and 'Visitor' design patterns

- For each program they solved a set of 4 small comprehension and modification tasks
  - for which the patterns were relevant

Dependent variables

- The observed variables were
  - time: The total time for solving one task
  - quality: A grading (in points) of the submitted solution according to well-defined criteria

Experiment design

- Nomenclature:
  - A: And/Or tree, P: Phonebook,
  - +: with PCL added, -: without

- Counterbalanced design:
  - 4 groups: A+ P- A- P+
    P+ A- P- A+
  - Randomized assignment of subjects to groups
  - No matching

Subjects

The experiment was performed twice:

- UKA: 74 diploma students of University of Karlsruhe; programs in Java
  - prepared solutions on paper
    - → incorrect answers produce no feedback → harder to detect
- WUSTL: 22 undergraduate students of Washington University, St. Louis; programs in C++
  - implemented solutions on Unix workstations
- All had taken a laboratory course on Java/C++ including design patterns

Results And/Or tree (difficult task)

- UKA: '+' is slower but much more often correct
  - Reason: wrong answers produce no feedback (work is on paper!)
- WUSTL: '+' is much faster

Variable	mean with PCL D^+	mean w/o PCL D^-	means difference (90% confid.) I	significance p
UKA, program *And/Or-tree*:				
1 relevant points	8.5	7.8	$-7.7\% \ldots +23\%$	0.20
2 #corr. solutions	15 of 38	7 of 36		0.077
3 time (minutes)	58.0	52.2	$-3.0\% \ldots +24\%$	0.094
4 — corr. only	52.3	45.4	$-11\% \ldots +41\%$	0.17
WUSTL, program *And/Or-tree*:				
5 relevant points	6.7	6.5	$-12\% \ldots +19\%$	0.28
6 #corr. solutions	4 of 8	3 of 8		1
7 time (minutes)	52.1	67.5	$-43\% \ldots -0.5\%$	0.046

Results phonebook (simple task)

- UKA: '+' is faster
- WUSTL: these results had to be discarded
  - because the subjects lacked knowledge of Observer pattern.
  - Also, the C++ version had no GUI, hence was unintuitive

Variable	mean with PCL D^+	mean w/o PCL D^-	means difference (90% confid.) I	significance p
UKA, program *Phonebook*:				
8 relevant points	16.1	16.3	$-8.0\% \ldots +4.0\%$	0.35
9 #corr. solutions	17 of 36	15 of 38		0.64
10 time (minutes)	51.5	57.9	$-22\% \ldots +0.3\%$	0.055

Discussion of internal validity

- Extraneous variables are controlled well by the counterbalanced design
  - even if groups were unequal, differences contribute equally to the experiment condition and the control condition

Problem:

- Quite some mortality in the WUSTL experiment
  - Very last event of the semester
  - *"I have to catch my plane home"*

- Fortunately, mortality in experiment and control groups is about equal
  - Has therefore probably not distorted the results

Threats to external validity

Differences to professional SW engineering contexts:

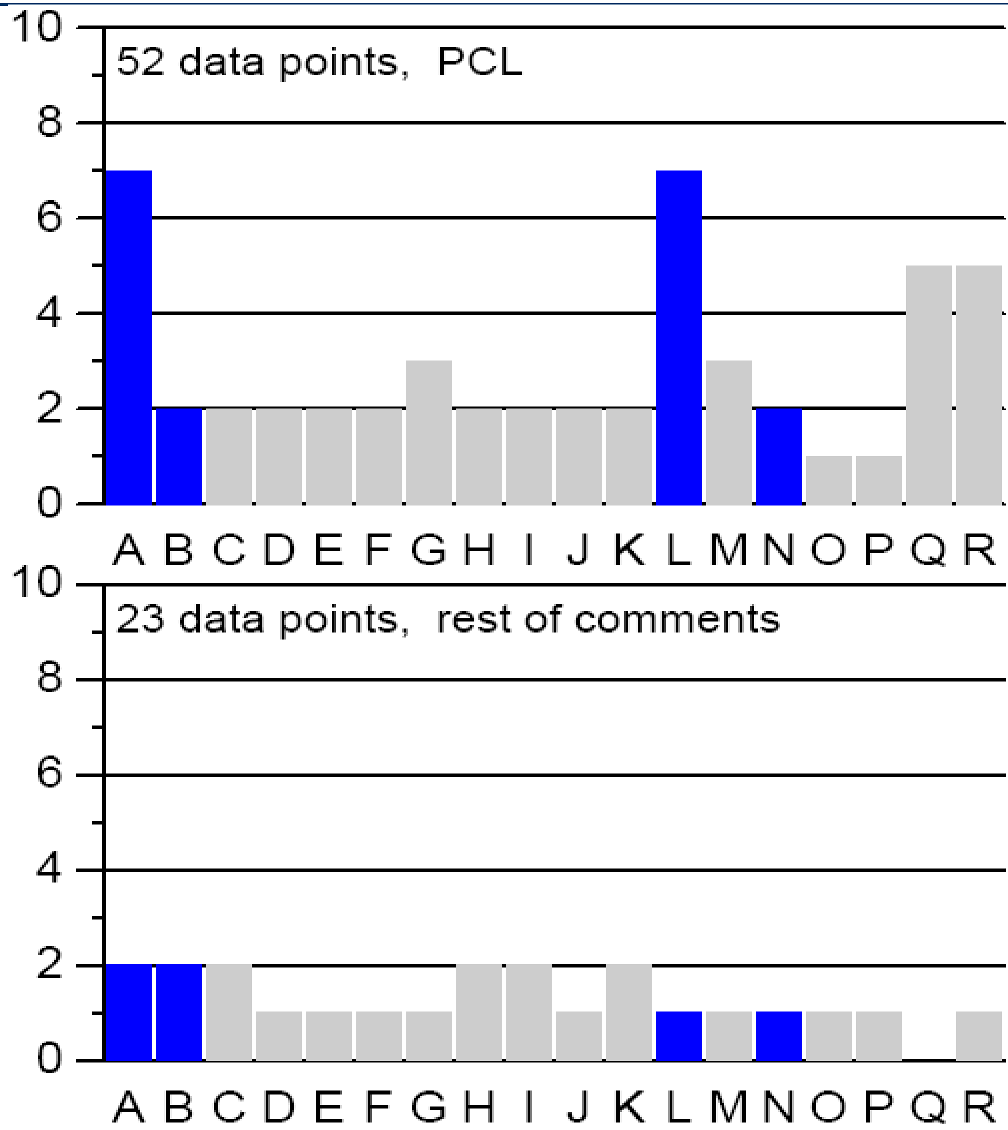
- Subject experience/capabilities:
  - Professionals may
    - have less need for PCL (would decrease effect) or
    - may make better use of PCL information (would increase effect)
- Team work:
  - May increase effect because patterns provide a common terminology; PCL allows for exploiting it
- Program size:
  - Larger programs may show a larger effect, as PCL provides program slicing information
- Program and task representativeness:
  - is unclear

Is 'no PCL' a good control group?

- It is surprisingly unclear what would be a valid experiment design for finding out whether *"having design pattern information is useful"* for maintenance:
  - Giving somebody program structure information (which somebody else does not have) will often help
    - but may have nothing to do with design patterns
  - Can the given comparison be considered fair?

Analysis of documentation content

- Analyzed which pieces of information are present how often in the documentation
  - here: for And/Or tree
- Identified 18 pieces (A-R), 4 of them crucial for solving the given tasks
- PCL is redundant: 17 pieces are present in non-PCL comments
  - incl. the 4 crucial ones A, B, L, M
- Therefore, the comparison is fair:
  - redundant information could also have hurt!



Description of some information pieces

id	design information unit (UKA *And/Or-tree*)
A	There is an *element/container structure*
B	**Element** is the superclass of the *element/container structure*
C	**Element** is abstract
D	**AndElement** is a part of the *element/container structure*
E	**OrElement** is a part of the *element/container structure*
F	**StringElement** is a part of the *element/container structure*
G	There are multiple *container* classes
H	**AndElement** is a *container* class
I	**OrElement** is a *container* class
J	There is only one *element* class
K	**StringElement** is an *element* class*
L	There is an *iterator structure**

Summary

- Controlled experiments apply the scientific method in its purest form:
  - Test whether an effect predicted by some theory is observed
- Control is for achieving constancy in the attributes that are not investigated (extraneous variables)
- Constancy is difficult to obtain with human subjects
  - They just differ so much!
  - The only way is repetition and averaging
- Other threats to constancy are history, maturation, instrumentation or experimenter effects, mortality, sequence effects, and sophistication
- Methods for improving constancy are randomization, matching, and counterbalancing

Thank you!