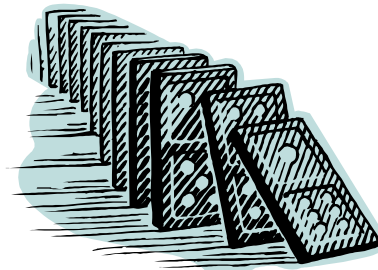


Experimentally Investigating the Effects of Defects in UML Models



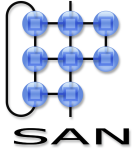
Christian Lange

C.F.J.Lange@TUE.nl

www.win.tue.nl/~clang

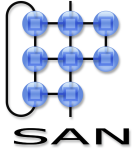
www.win.tue.nl/empanada

SAN meeting – February 11th, 2005



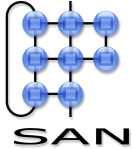
- Introduction and Motivation
- Experimental Design
- Results
- Analysis
- Conclusion





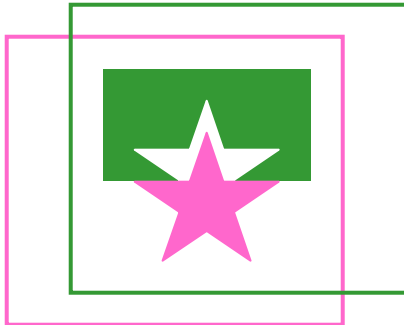
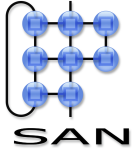
- ❑ EmpAnADa project
 - Empirical Analysis of Architecture and Design Quality
 - ❑ Early evaluation of quality attributes
 - Metrics for Architecture and Design models
 - ❑ Improvement by Refactoring
 - ❑ **Completeness and Consistency** checking
 - Development of methods
 - Method validation by Empirical Studies
 - ❑ Direct feedback
 - ❑ Exploring problems in practice





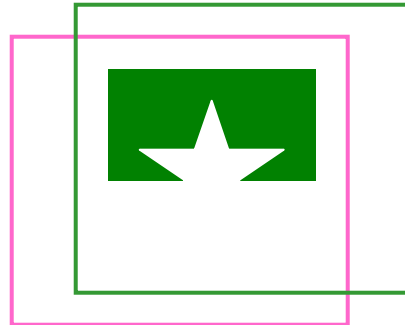
- ❑ Unified Modeling Language
 - No formal semantics
 - Offers large degree of freedom
 - ❑ Extension Mechanisms
 - ❑ 9 Diagram types (UML 2 has even 13!!)
 - Diagrams are overlapping!
 - UML is used in many different ways
 - UML Defects:
 - ❑ Completeness
 - ❑ Consistency





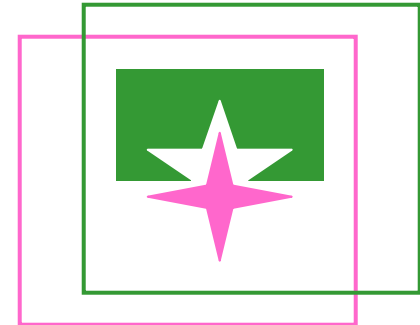
Correct

- Matching elements in overlapping parts of diagrams



Completeness defect

- Missing element

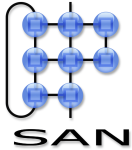


Consistency defect

- Mismatch
- Conflicting elements

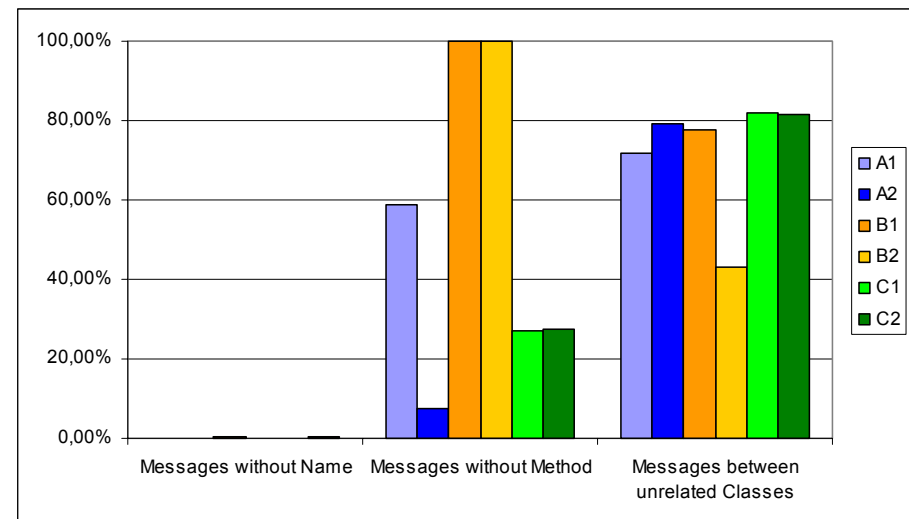
Detection by checking rules

- SAAT
- RPA

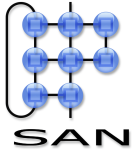


□ Our case studies have shown large numbers of defects

➤ How serious are defects?



Consistency	A1	A2	B1	B2	C1	C2
Messages without Name	0,00%	0,00%	0,28%	0,00%	0,00%	0,46%
Messages without Method	58,73%	7,62%	100,00%	100,00%	27,14%	27,40%
Messages between unrelated Classes	71,94%	79,37%	77,73%	43,14%	81,90%	81,74%

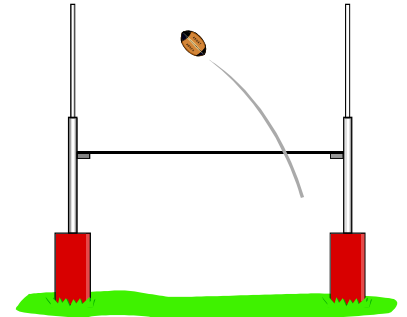


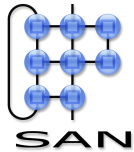
❑ Defects occur in practice, but does it matter?

- Are defects detected in practice?
- If not, do defects lead to misunderstandings?

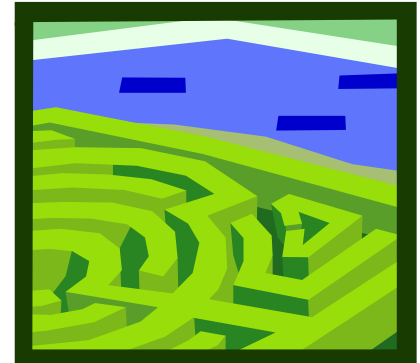
❑ Goal (GQM)

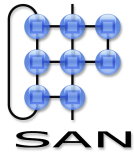
- *Analyze defects in UML models for the purpose of identifying issues and observations with respect to misunderstandings and ambiguities from the perspective of the researcher in the context of TUE grad students and professionals.*





- Give UML models to subjects
- Ask question about UML model
 - Two types
 - Which implementation matches the diagrams?
 - How do you interpret these diagrams?
 - Answer from the perspective of the developer.
- Multiple-choice test
 - 4 options
 - + 1 option (“There’s something wrong, can’t give an answer”)
- Background questions
- Pilot run was performed

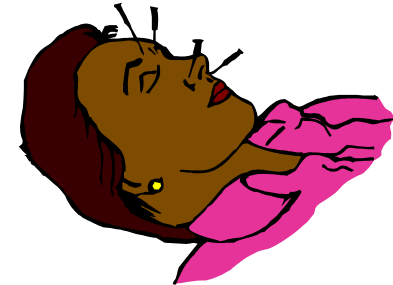


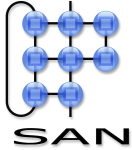


- Treatment
 - one of 9 defects
 - no defect

- Defects are injected in model
 - Each *infected* question is paired with a *control question*

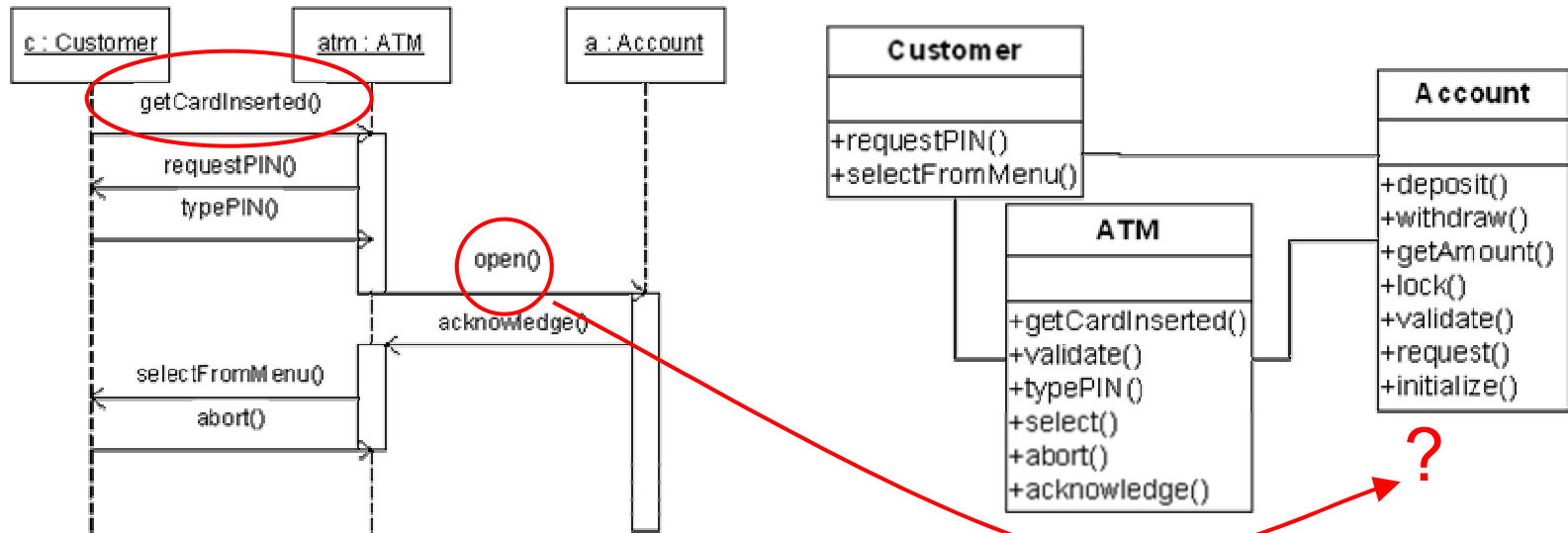
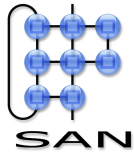
- All subjects receive all treatment levels
 - “same subject design”
 - by definition “balanced”
 - all treatment groups have the same background





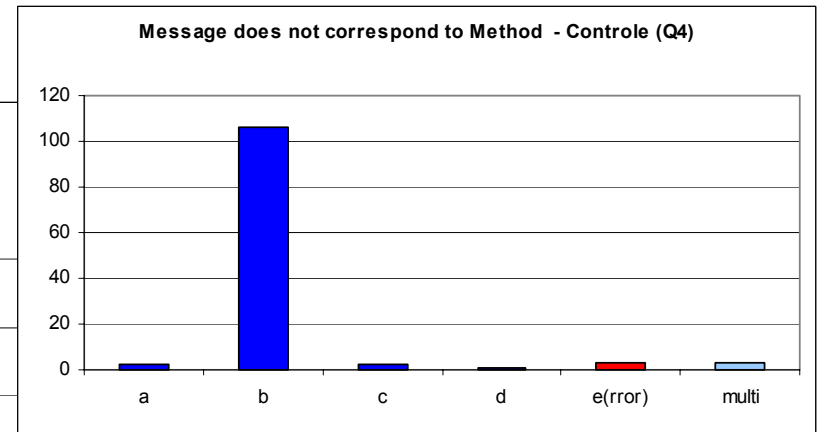
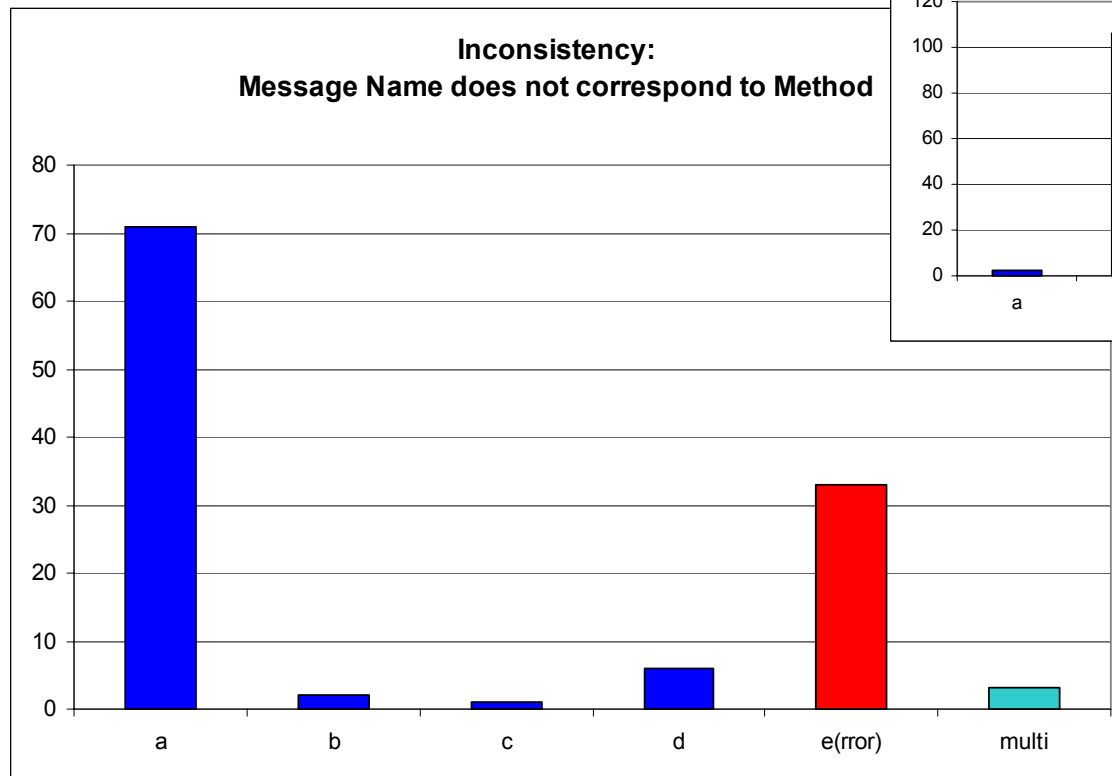
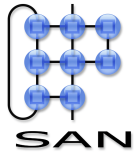
- Message without Name
- Message Name does not correspond to Method
- Message in the wrong direction
- Class from CD not in SD
- Class from SD not in CD
- Use Case without SD
- Multiple definitions of Class with equal Name
- Method from SD not in CD
- Method from CD not in SD

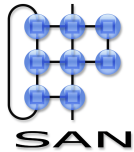
Example Question



Suppose you are developer in this banking software project. It is your task to implement class ATM. Please indicate how you would implement the ATM class given these two UML diagrams?

- | | | | | |
|---|---|---|---|----------------------------|
| <p>A) <i>SD leading</i></p> <pre> Class ATM{ Method getCardInserted(){ c.requestPIN(); dosomething; a.open() } ...} </pre> | <p>B) <i>CD leading</i></p> <pre> Class ATM{ Method getCardInserted(){ c.requestPIN(); dosomething; a.lock() } ...} </pre> | <p>C) <i>wrong</i></p> <pre> Class ATM{ method getCardInserted(){ c.requestPIN(); dosomething; a.acknowledge() } ...} </pre> | <p>D) <i>CD leading</i></p> <pre> Class ATM{ method getCardInserted(){ c.requestPIN(); dosomething; a.validate() } ...} </pre> | <p>Something is wrong!</p> |
|---|---|---|---|----------------------------|





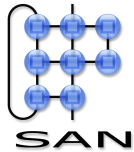
110 Students

- 1st year MSc programme (TUE)
- Course: Software Architecture (2II10)
- Preparation
 - Lecture about UML
 - 5 weeks assignment: designing and evaluating UML

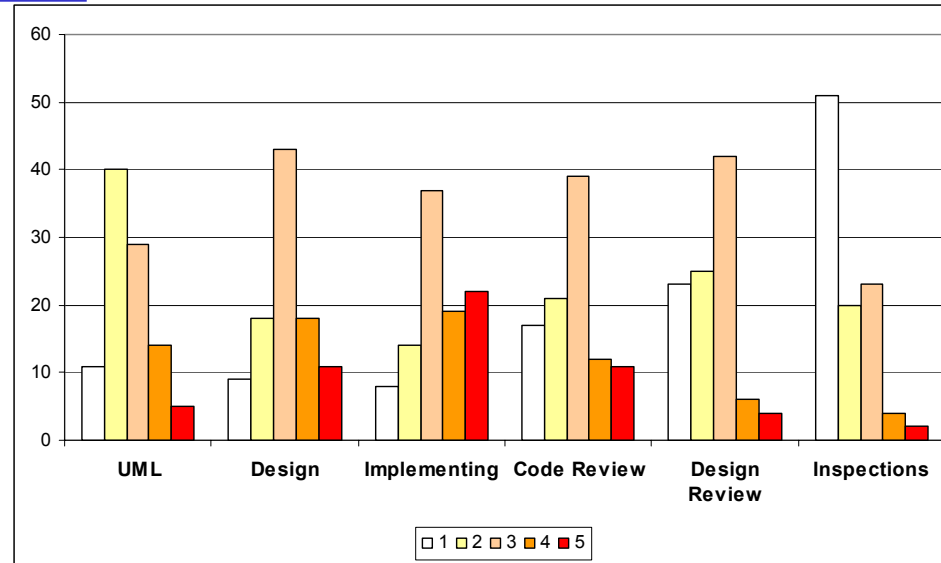


Professionals

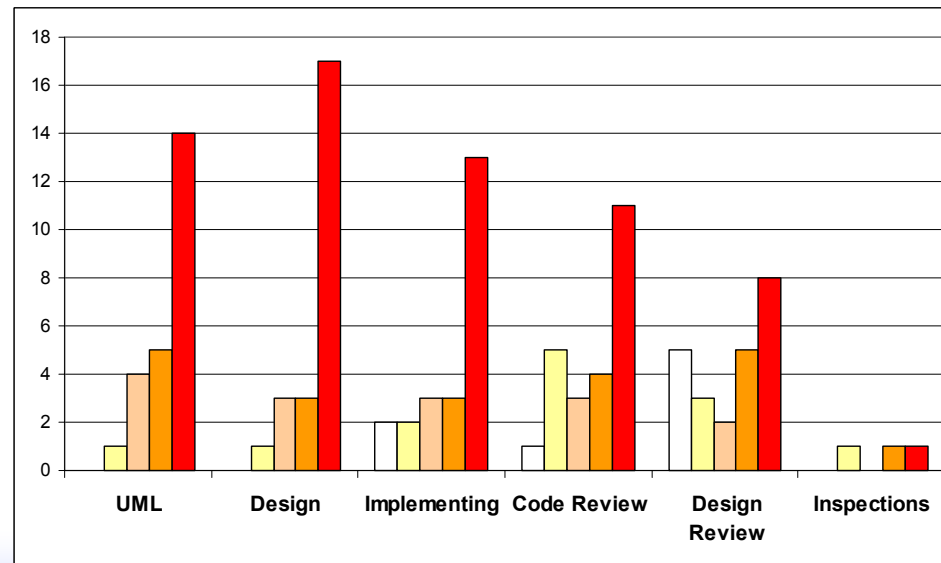
- Completed online questionnaire
- Emailed URL to contacts and newsgroups
- 45 answered Q1 - 24 answered Q10



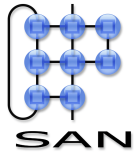
Students



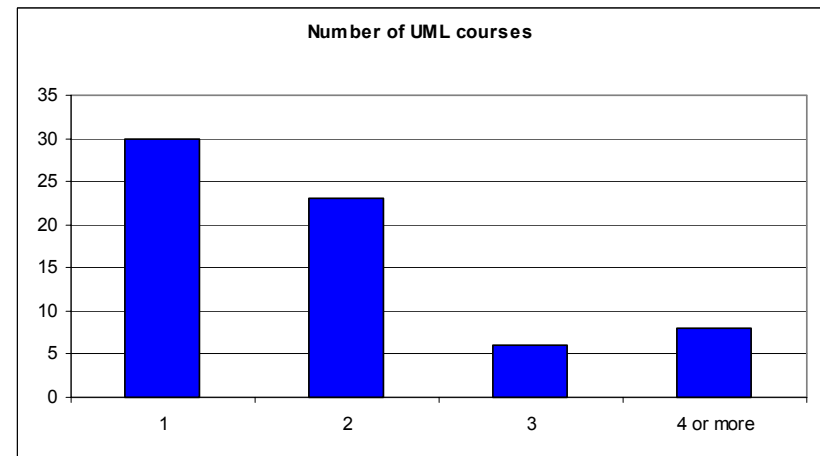
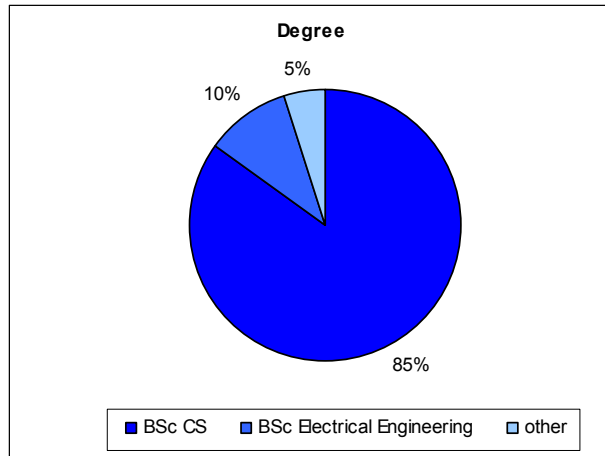
Professionals



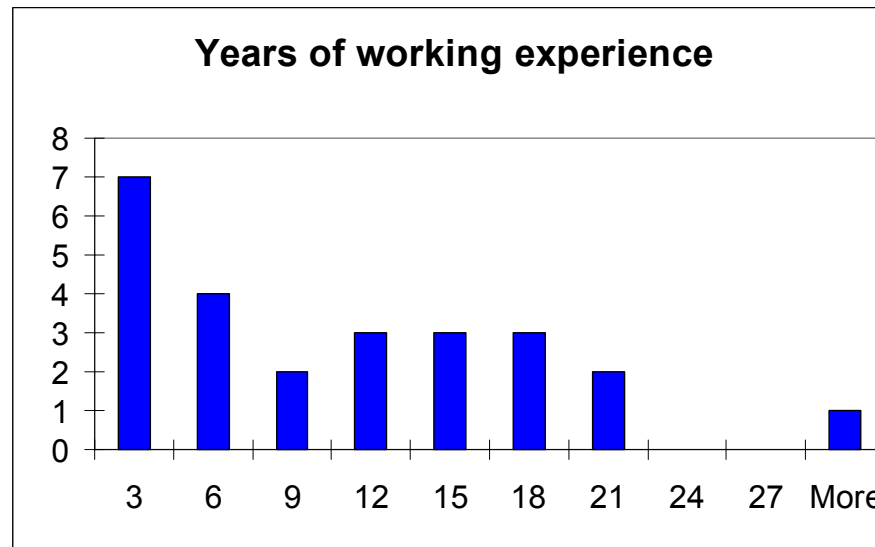
Subjects' Experience I

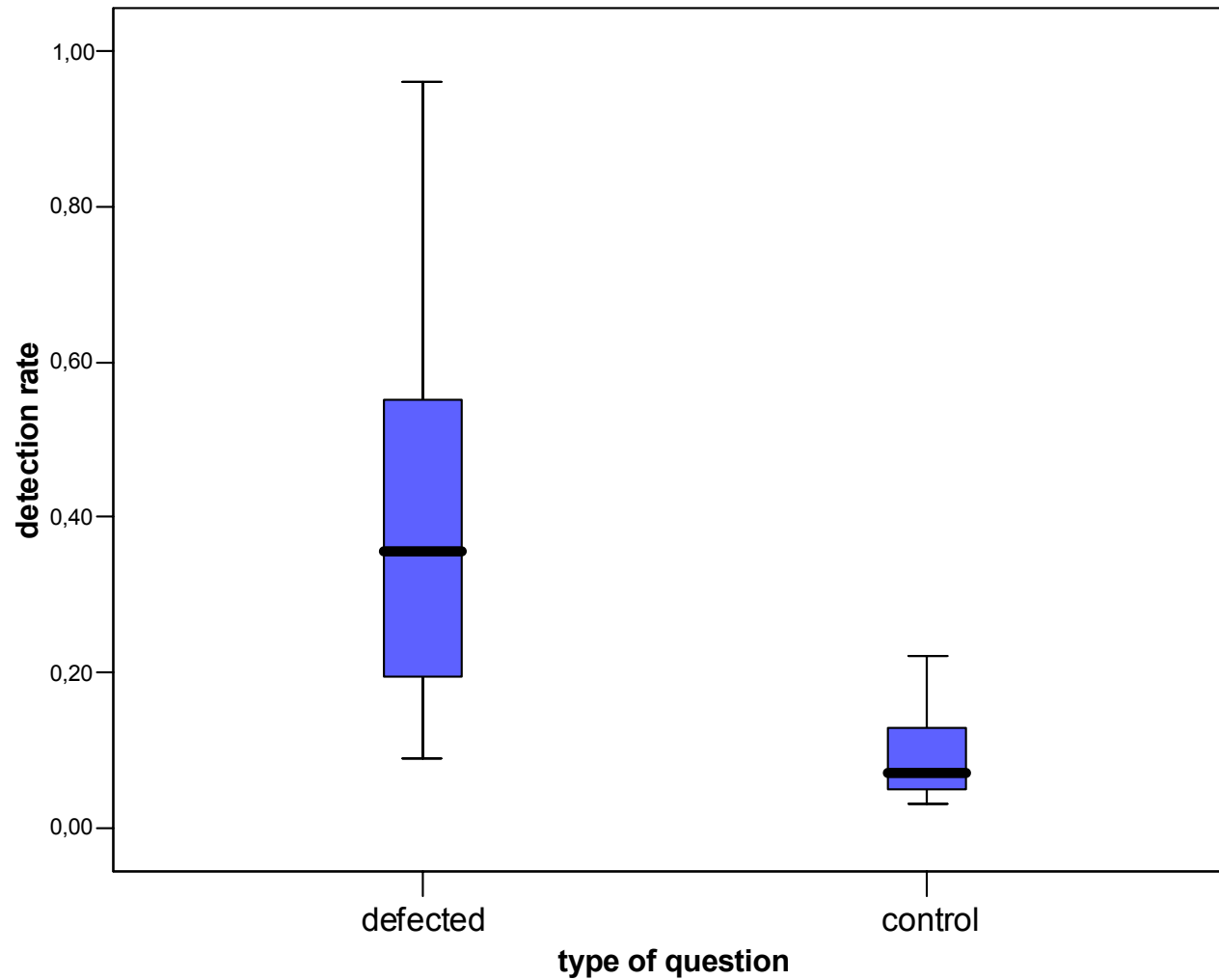
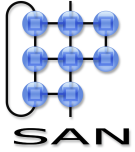


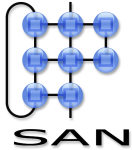
Students



Professionals





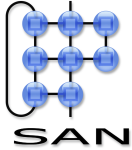


□ Defects

- Sorted by detection rate (students)

Defect	Stud.	Prof.
Class not in SD (symb.)	0,95	0,96
Message without Name	0,69	0,60
Message in the wrong Direction	0,60	0,58
UC without SD	0,50	0,52
Method not in CD	0,49	na
Message without Method (symb.)	0,49	0,33
Class not in SD	0,47	0,68
Message without Method	0,39	0,38
Class not in CD	0,18	0,11
Method not in SD	0,14	na
Multiple Class defs.	0,10	0,33
	<i>Average</i>	<i>0,46</i> <i>0,50</i>
	<i>Std Dev</i>	<i>0,26</i> <i>0,25</i>
	<i>Max</i>	<i>0,95</i> <i>0,96</i>
	<i>Min</i>	<i>0,10</i> <i>0,11</i>

Variability Measure (Entropy)

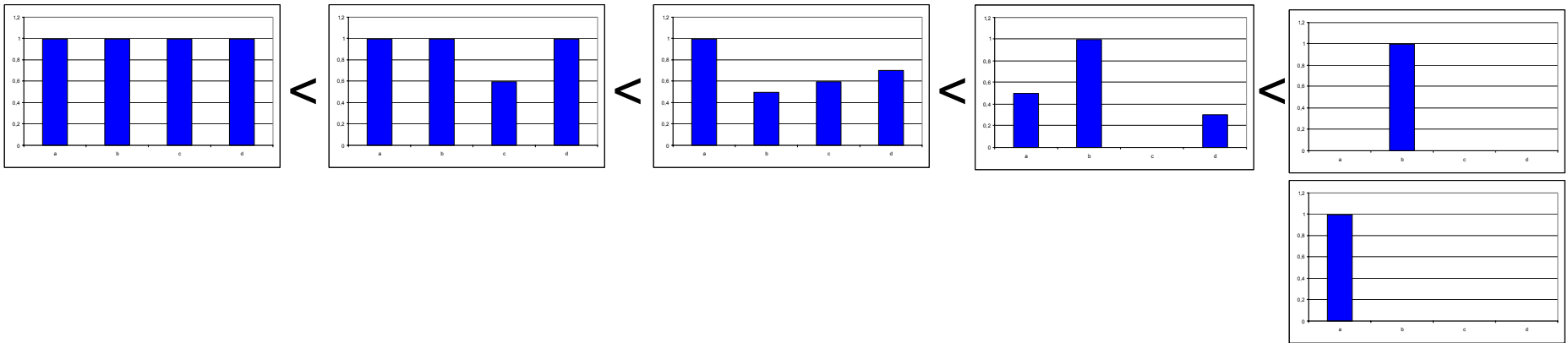


- If a defect is not detected, does it lead to misinterpretation?

$$\text{VarM}(k_0..k_{i-1}) = 1 - 2 \frac{\sum_{i=0}^K k_i i}{N(K-1)}$$

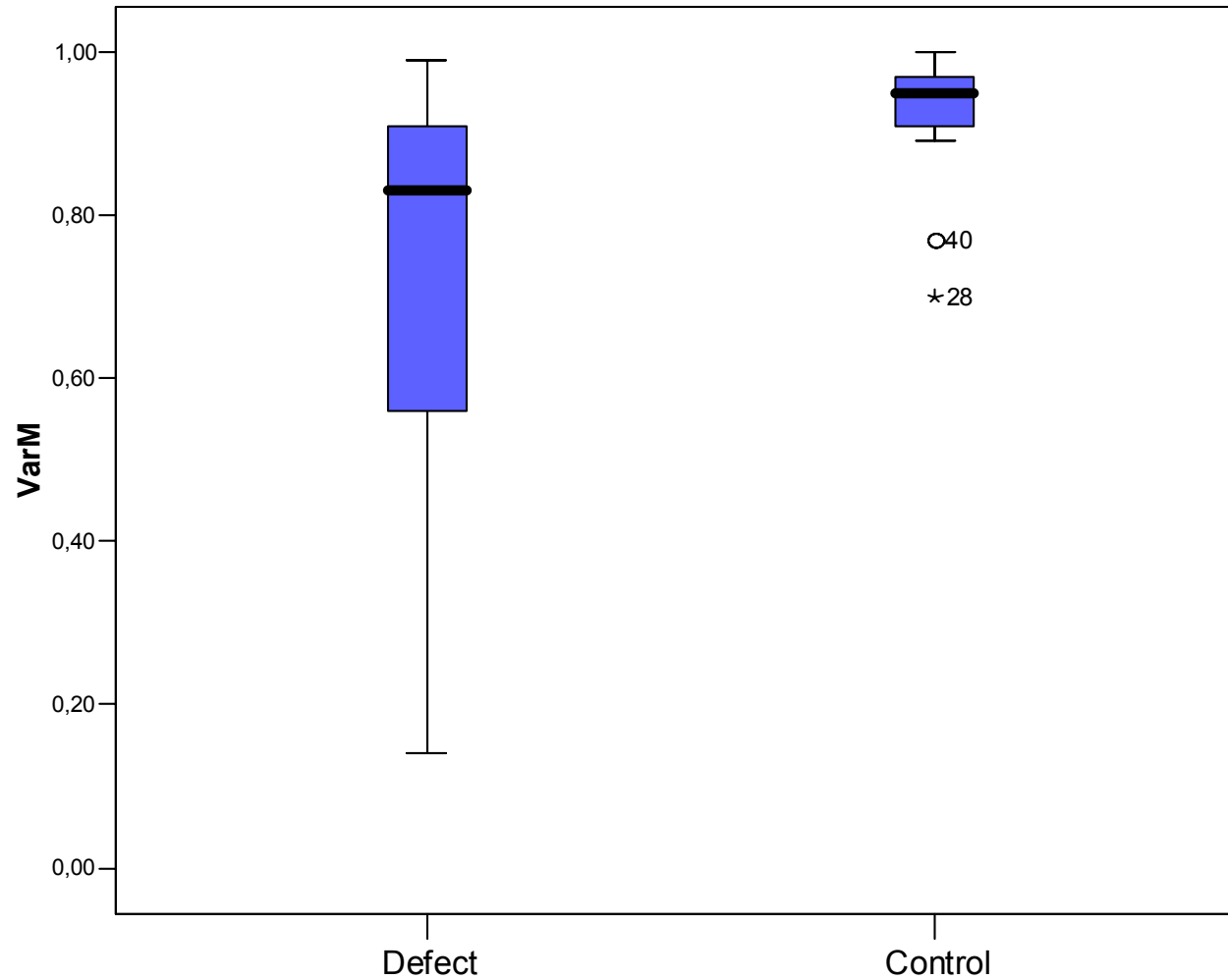
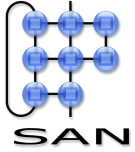
- K = Number of options, N = Sum of all answers,
- k_i = # of answers of option i (k_i 's are ordered: $k_i \leq k_{i+1}$)

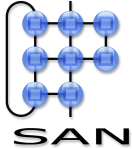
- Intuition: Measuring how discriminating a distribution is



VarM=0

VarM=1

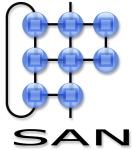




□ Defects

➤ Sorted according to VarM

Defect	Stud.	Prof.
Multiple Class defs. (meth.)	0,92	0,68
Message without Method (symb.)	0,86	0,94
Message without Method	0,84	0,90
UC without SD	0,83	0,44
Class not in CD (meth.)	0,83	0,93
Method not in CD	0,69	n/a
Method not in SD	0,67	n/a
Class not in SD	0,49	0,64
Message in the wrong Direction	0,47	0,95
Message without Name	0,47	0,44
Class not in SD (symb.)	0,34	0,14
	<i>Average</i>	<i>0,71</i> <i>0,80</i>
	<i>Std Dev</i>	<i>0,16</i> <i>0,21</i>
	<i>Max</i>	<i>0,86</i> <i>0,95</i>
	<i>Min</i>	<i>0,47</i> <i>0,44</i>



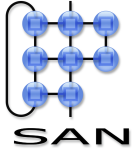
☐ Product of

➤ Detection rate

➤ VarM

☐ Combination of low detection rate and many different interpretations (low VarM)
→ causes most misunderstandings

Defect	Stud.	Prof.
Class not in SD (symb.)	1,29	0,14
Message without Name	0,37	0,26
Message in the wrong Direction	0,32	0,55
Class not in SD	0,24	0,44
Method not in CD	0,15	n/a
UC without SD	0,09	0,23
Message without Method (symb.)	0,07	0,31
Message without Method	0,06	0,34
Method not in SD	0,05	n/a
Class not in CD	0,03	0,21
Multiple Class defs.	0,01	0,23



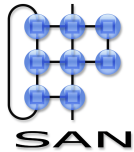
“There are no defects in UML models that SAN people create!”

- Can defects be corrected by using context knowledge (domain knowledge) ?

- We made equal questions
 - One version with “context” (Traincrossing, Sensor)
 - One version without (Class A, Method 3)

- (Cultural background was taken into account)

Domain Knowledge II

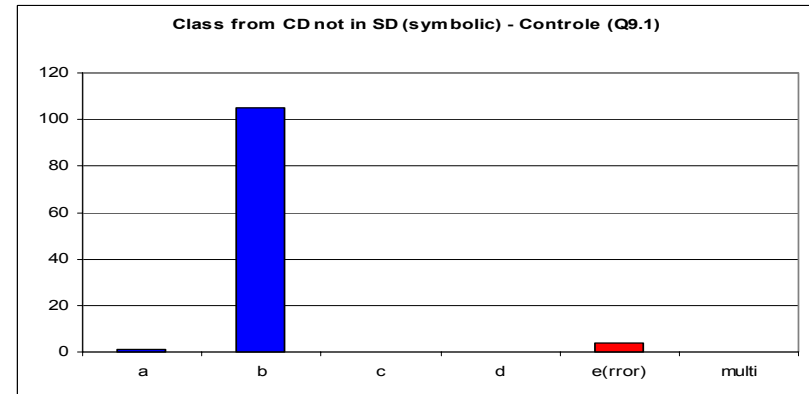
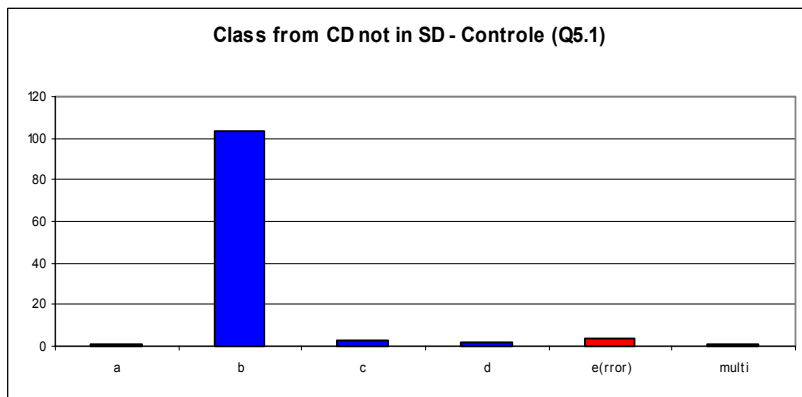
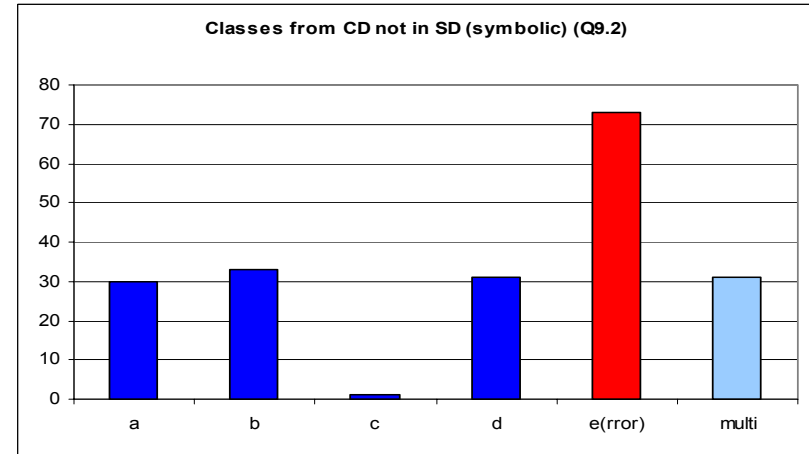
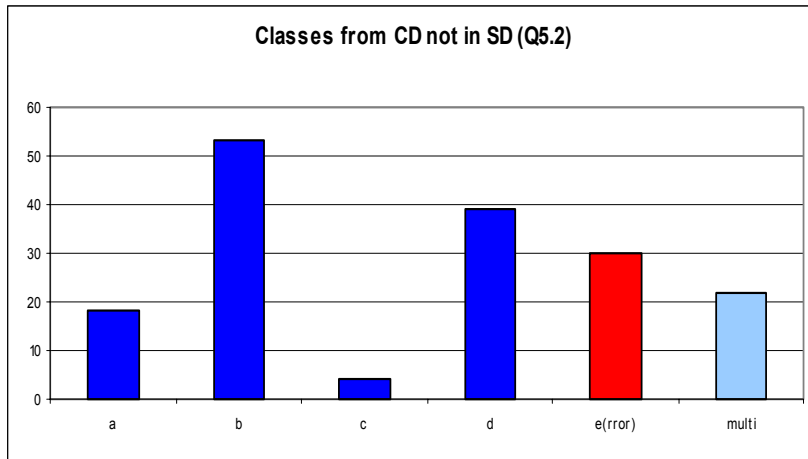


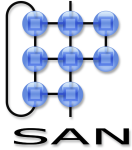
Domain Knowledge

Detection Rate 0.47
 VarM 0.49

No Domain Knowledge

0.95
 0.34





Is it valid to generalize the outcome of the students experiment?

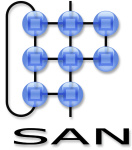
Pearson Correlation

➤ **Between student results and professional results**

➤ **For detection rate: 0.929 (p-value < 0,001)**

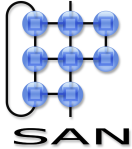
➤ **For VarM: 0.643 (p-value < 0,004)**

➤ **“significant” = p-value < 0,05**



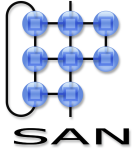
- ❑ In most cases the sequence diagram was regarded as the *leading* diagram
- ❑ To investigate order effects, we presented some questions in a second run with the diagrams in reversed order

- ❑ Pearson Correlation
 - Professionals vs. reversed
 - ❑ Detection rate: 0.824 (p-value: 0.044)
 - ❑ VarM: 0.899 (p-value: 0.015)
 - Students vs. reversed
 - ❑ Detection rate: 0.913 (p-value: 0.011)
 - ❑ VarM: 0.638 (p-value: 0.173)



- It makes sense to detect defects!
- Defects in UML models
 - are detected only to a rather low degree
 - do cause misunderstandings
- We ordered defects according to severity
- Domain knowledge matters!
 - But can cause misunderstandings
- Order of models does not matter

- Fault-injection in UML models



Experimenting is not as simple as it seems

➤ Design

- What is the question? Hypothesis?
- What are the variables?
- How to distribute the treatments over the subjects?

➤ Preparation

- Experiment Material

➤ Carefully instructing the subjects

➤ Analysis

- Using the proper methods

