Course "Empirical Evaluation in Informatics"

Surveys

Lutz Prechelt
Freie Universität Berlin, Institut für Informatik

- Example: SE education
- Method:
  - Set study goals
  - Select target population
  - Design questionnaire
  - Conduct survey
  - Evaluate results
- Example: Peer review

"Empirische Bewertung in der Informatik"
Umfragen

Lutz Prechelt
Freie Universität Berlin, Institut für Informatik

- Beispiel: Relevanz der Informatik-Ausbildung
- Methode:
  - Auswahl der Ziele
  - Auswahl der Zielgruppe
  - Fragebogenentwurf
  - Durchführung
  - Auswertung
- Beispiel: Peer Review

Example 1: Relevance of CS and SE education

- Source: T. Lethbridge: "*[What Knowledge Is Important to a Software Professional?](#)*", IEEE Computer, May 2000
  - see also http://www.site.uottawa.ca/~tcl/edrel/

- Research questions:
  Which parts of their education are considered how relevant by software engineering practitioners?
  Do they perceive their education as misaligned?

- Study format: Survey

T. Lethbridge

Approach

- Uses a list of 75 topics from Computer Science (CS) and Software Engineering (SE) education
  - e.g. data structures, physics, project mgmt., HW architecture
- For each topic, asks 4 questions:
  - (1) how much was learned in education,
  - (2) how much was learned (or forgotten) since ,
  - (3) how useful the knowledge on the topic has been, and
  - (4) how influential on one's thinking the topic has been
- Determines the topics that are
  - deemed important but not taught widely ("knowledge gap")
  - deemed unimportant but taught widely

Survey population

- Web-based survey
  - employees of various companies (approached via mgmt)
  - postal mailing lists (e.g. university alumni)
  - email lists, Usenet newsgroups
- Over 200 participants
  - 186 participants were selected to form a balanced sample
- 54% from USA, 23% from Canada; 24 countries overall
  - 42% from software companies
- Education of participants:
  - 15% high school or college level (without degree); 48% bachelor; 37% postgraduate
  - >60% CS, SE, or IS degrees; 50% other science or engineering; 20% other disciplines
    - Many had more than one degree

Question 1

- How much did you learn about this in your formal education (e.g. **University or College**)?
  - 0=<u>Learned nothing</u> at all
  - 1=Became <u>vaguely familiar</u>
  - 2=Learned the <u>basics</u>
  - 3=Became <u>functional</u> (moderate working knowledge)
  - 4=Learned <u>a lot</u>
  - 5=Learned <u>in depth</u>; became <u>expert</u> (Learned almost everything)

Question 2

- What is your **current knowledge** about this, considering what you have learned on the job as well as forgotten?
    - 0=Know nothing
    - 1=Am vaguely familiar
    - 2=Know the basics
    - 3=Am functional (moderate working knowledge)
    - 4=Know a lot
    - 5=Know in depth / am expert (Know almost everything)

Question 3

- How useful have the details of this **specific material** been to you in your career as a software developer or software manager?

  Please leave blank if you know little about the material.

  - 0=Completely Useless
  - 1=Almost never useful
  - 2=Occasionally useful
  - 3=Moderately useful, but perhaps only in certain activities
  - 4=Very useful
  - 5=Essential

Question 4

- How much **influence** has learning the material had **on your thinking** (i.e. your approach to problems and your general intellectual maturity), **whether or not you have directly used the details of the material?**

  Please consider influence on both your career and other aspects of your life.

  Please leave blank if you know little about the material.

  - 0=<u>No influence</u> at all
  - 1=<u>Almost no</u> influence
  - 2=<u>Occasional</u> influence
  - 3=<u>Moderate</u> influence in some activities
  - 4=<u>Significant</u> influence in many activities
  - 5=<u>Profound</u> influence on almost everything I do

Results: SE topics

		Top (■) and bottom (⬓) quartile and top (●) and bottom (◒) four topics, in terms of:		
Category	**Topic**	**Overall importance (Q3 + Q4)**	**Learned in education (Q1)**	**Learned on the job (or forgotten since education) (Q2 – Q1)**
General software design	Data structures	●	■	
	Algorithm design	■	■	
	Software design and patterns	●		■
	Software architecture	●		
	Object-oriented concepts and technology	■		■
	Specific programming languages	●	●	
Software engineering methods	Requirements gathering and analysis	●		■
	Formal specification methods			
	Analysis and design methods	■		■
	Performance measurement and analysis			
	Testing, verification, and quality assurance	■		●
	Software reliability and fault tolerance			■
	Maintenance, reengineering, and reverse engineering		⬓	●

Results: SE topics (2)

Software management	Project management	■		●
	Software metrics		⬓	
	Software cost estimation		⬓	■
	Configuration and release management	■	◒	●
	Process standards such as CMM, ISO9000		◒	■
Essential subsystem design	Human-computer interaction/user interfaces	■		■
	Databases	■		■
	File management			
Specialized application techniques	Computational methods for numerical problems		■	⬓
	Simulation			
	Artificial intelligence	⬓		
	Pattern recognition and image processing	⬓	⬓	
	Computer graphics	⬓		
	Parsing and compiler design			
	Information retrieval			
	Security and cryptography		⬓	

Results: CS and science topics

Real-time and systems programming	Operating systems	■	■	
	Systems programming			
	Data transmission and networks			
	Parallel and distributed processing			
	Real-time system design		▄	
Computer hardware	Digital electronics and digital logic		■	▄
	Microprocessor architecture			
	Computer system architecture		■	
	Network architecture and data transmission			
	Telephony and telecommunications		▄	
Other electrical and computer engineering	Analog electronics	▄		▄
	Digital signal processing	▄	▄	
	Data acquisition		▄	
	Robotics	◒	▄	
	VLSI	◒	▄	
Computer science theory	Programming language theory		■	
	Formal languages		■	▄
	Computational complexity and algorithm analysis		■	
	Information theory	▄		

Results: CS and science topics (2)

Discrete mathematics	Predicate logic		■	▄
	Set theory		■	▄
	Graph theory	▄		▄
	Automata theory			▄
	Queuing theory	▄		
	Combinatorics	▄		▄
Probability and statistics			●	▄
Linear algebra and matrices			●	◒
Continuous mathematics	Differential and integral calculus	▄	●	◒
	Differential equations	◒	■	◒
	Control theory	▄		▄
	Laplace and Fourier transforms	▄		▄
Natural science	Physics	▄	■	▄
	Chemistry	◒	■	◒

Results: Other topics

Business	Economics			
	Accounting		▄	
	Marketing		▄	
	Management			
	Entrepreneurship		◒	
Psychology and philosophy	Psychology			
	Philosophy			
	Ethics and professionalism	■		■
Technical writing		■		■
People skills	Giving presentations to an audience	■		■
	Leadership	■	▄	■
	Negotiation		◒	■
Second language other than English		▄		▄

Results summary

- Some topics appear to be much over-emphasized in the formal education compared to perceived later usefulness
  - e.g. calculus (dt.: Analysis)
- Others appear much more important in practice than the education reflects, in particular
  - software management
  - people skills
  - requirements gathering
  - quality assurance

Validity problems: Internal and construct

- Self-selection bias
  - Maybe many participants just wanted to lament about their education?
- Subjective answers
  - *Real* amount of knowledge or usefulness is unknown
- 15% answers from respondents without formal degree
  - Validity unclear
- Scale violations:
  - Taking the difference Q2-Q1 (knowledge now minus originally learned) requires equal-sized difference scales
  - Forming the sum Q3+Q4 (usefulness plus influence) requires equal-sized ratio scales
    - What does the sum mean anyway?

How good is the credibility overall?

External validity problems

- Some of the answers strongly reflect north-american curricula
- Importance of topics is continually shifting over time!
  - some parts of the snapshot are obsolete
    - e.g. AI is now much more important
- What mix of software industry branches is represented in the sample?

Generic survey method

1. Decide on objectives
2. Select a target population
3. Design the survey instrument (questionnaire)
4. Administer the survey
   - supervised, unsupervised, or semi-supervised
5. Collect, validate and analyze the data
6. Answer the research questions

1. Decide on objectives

- A good understanding of the survey goals is required to select a compact set of questions
  - Too-long questionnaire will reduce number of respondents and may reduce the quality of the answers
- Surveys are suitable for measuring attitudes, much less suitable for determining factual situations

Basic types of objectives:

- Cross-sectional: snapshot ← most common type
  - What is the status now?
- Longitudinal: cohort observation
  - How does the status change over time?
    - Requires multiple rounds of surveying with the *same* participants
- Retrospective: explanation
  - What are the reasons for the status ?

2. Select a target population

Check questions:

- What kind of respondents do I need to reach my goals?
  - Need to use language/terminology appropriate for them
- How many such respondents do I need?
  - Is that realistic with acceptable effort?
- How can I reach these people?
- How can I motivate them to participate?
  - What response rate should I expect?

- What kinds of irrelevant or distortive participants should I expect?
  - Can I recognize these from their answers and sort them out?
- So where and how should I advertise my study?

3. Design the survey instrument (questionnaire)

Freie Universität Berlin

- Search for similar, previously used questionnaires
  - Psychometricians call them "instrument": development is difficult
  - Analyze them (and the experience made) and adapt them
    - Piece your questionnaire together from multiple sources

If you need to design your own:

- Minimize the number of questions
  - Standardize the response format where possible
    - e.g. strongly agree, agree, disagree, strongly disagree
- Design each question carefully (see next slide)
- Put the demographic questions at the <u>end</u>
  - So people already know what information they have provided
  - So they are less likely to drop out near the end
- Ask for global <u>comments</u> on both the topic of the survey and the survey itself

Design the instrument: questions

For each question, make sure you have:

- Clear purpose
  - Respondent (and you!) must understand the role of the question in the context of the survey and for the goals of the survey
- Single purpose
  - A question must not mix two issues
- Complete, precise, unambiguous formulation
  - Use simple and complete sentences
    - What is simple depends on the population
    - Avoid jargon and specialized terminology
      - e.g.: *"How would you rate your training/education experiences regarding co-occurring disorder clients to date?"*
    - Avoid negations
  - What exactly does the question refer to?
    - time, context, entity/attribute
  - What not?

Design the instrument: question types

- Open questions:
  - Respondents formulate their own answer
  - Advantages:
    - wider spectrum of possible insights
    - less dependent on prior knowledge of questionnaire designers

- Closed questions:
  - Respondents choose among fixed answer categories
    - e.g. single choice, multiple choice, numeric, date
    - Never forget the category
      "don't know"/"none of these"/"does not apply"
  - Advantages:
    - easy quantitative evaluation
    - reduced ambiguity, less danger of irrelevant answers

(A good mix is usually the best idea.)

Design the instrument: validation

Any questionnaire <u>must</u> be pilot-tested (e.g. by exit interviews):

  - (in particular for unsupervised surveys; make sure you *shut up*)

- Find out whether overall purpose is clear
  - and why the participant should be motivated
- Find out whether purpose and formulation of all questions are clear
  - Detect ambiguities
  - Detect obscure terminology etc.
- Find out if time to complete is acceptable
- Find out whether layout and user interface are acceptable

Pilot-test again after the very last 'correction'

Design the instrument: validity, reliability

- Validity
  - The degree to which the instrument really measures what it was designed to measure (construct validity)
  - Assessing validity is methodically quite difficult and is beyond the scope of this course
    - See a textbook on social science research methods

- Reliability
  - The degree to which the instrument will give the same results when used in the same circumstances
    - if reliability is low, validity will be limited
  - Assessing it requires a number of respondents answering the questionnaire again after some time (e.g. a few weeks)
    - questions that are difficult to decide for the respondents typically lead to low reliability

4. Administer the survey

Basic types of administration:

- Supervised
  - An interviewer asks questions, answers clarification questions, and records answers (one-on-one, e.g. telephone)
- Unsupervised
  - The participant is completely on his/her own with the questionnaire (e.g. web-based)
    - see http://en.wikipedia.org/wiki/Comparison_of_survey_software, live answering, but only isolated questions: http://onlineTED.de
  - Issues: multiple participation, question misunderstandings, joke answers, random answers
- Semi-supervised
  - An interviewer gives some introduction to a group of participants and answers questions, but the filling-in is unsupervised

5. Collect, validate and analyze the data

Tasks:

- Collect data into machine-readable form
  - Avoid/detect mistakes when typing in paper questionnaires
- Validate data:
  - Detect and remove duplicates (e.g. sent by email or http)
  - Detect invalid responses (needs consistency check questions)
  - Detect ambiguous questions (by incoherent answer structure)
- Perhaps balance the respondent set:
  - If some subgroups are over-represented,
    - either sample a subset from each of these (if you have enough data)
    - or weight subsets differently during analysis
- Analyze

6. Answer the research questions

When drawing conclusions from a survey

- keep in mind the limitations of your sample
  - in particular non-representativeness
- keep in mind possible validity problems such as
  - bias in the questions,
  - bias in the answers,
  - ambiguities and misunderstandings

Example 2 [futreview]: Perceptions of peer review

- Source: L. Prechelt, D. Graziotin, D. Mendez:
  "*On the Status and Future of Peer Review in Software Engineering*", Information and Software Technology, 2018
  - Peer Review is a quality assurance mechanism:
    Experts (colleagues, peers) review a draft research article and recommend whether it should appear (at a given journal or conference) and what changes are needed before.
    - Question and results relevant? Sufficiently new?
    - Method and conclusions valid? (credibility)
  - (Does not work well, if reviewers
    - are not knowledgable,
    - work sloppily, or
    - make exaggerated requirements.)
- Study format:
  mixed quantitative/qualitative survey

Watch out:
Very dense slides!

[futreview]: Survey overview

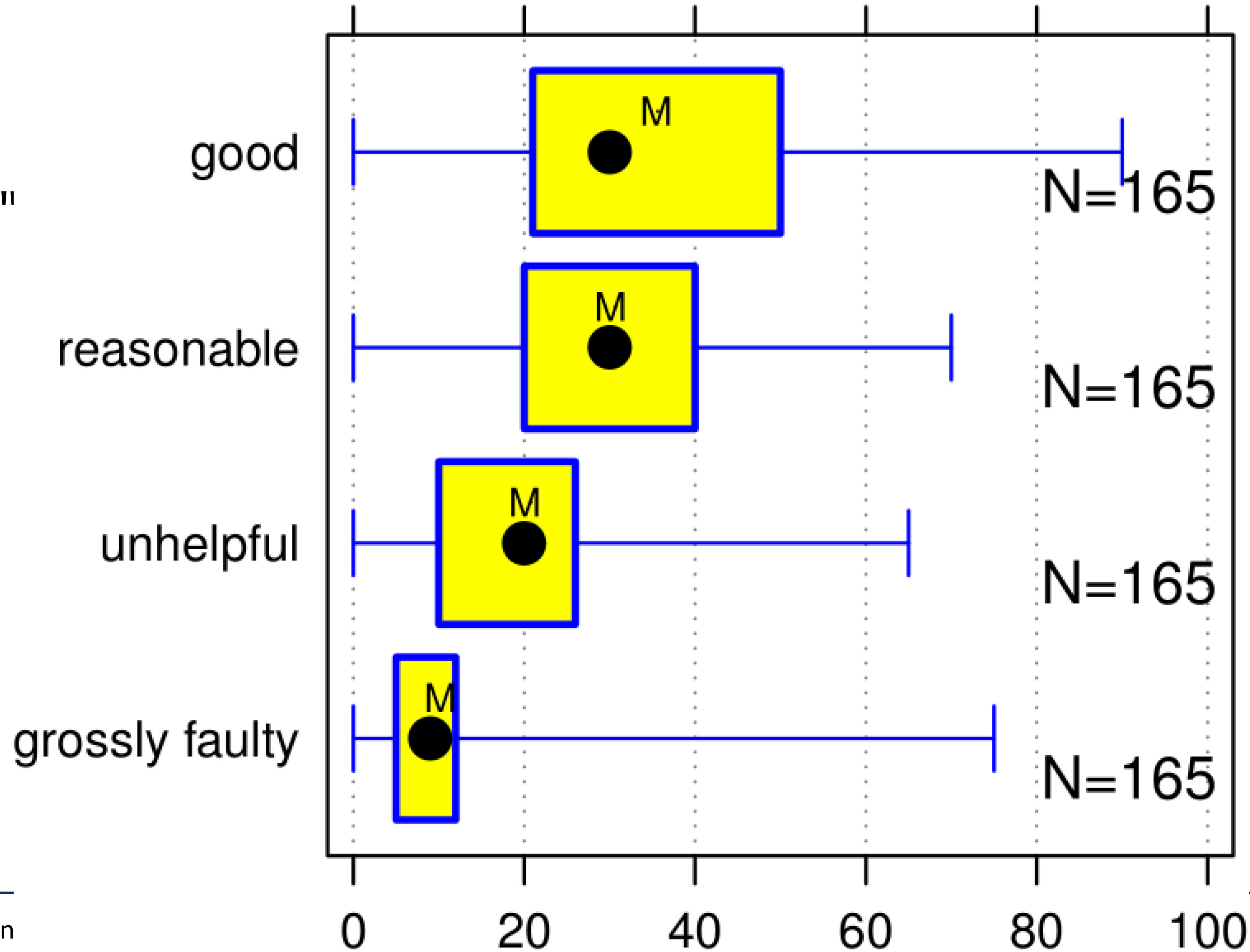
- 19 questions:
  - types: yes/no, 11-point scale, percentage, text
- Topics:
  - purposes of peer review and their relative importance
  - perceived quality of peer reviews (quantitative)
    - and why/how the bad ones are bad (qualitative)
  - anonymity in peer review (double-blind, …)
  - publicness of reviews
  - compensation for reviewers ((quasi-)monetary, non-monetary)
  - age, professional status, author experience, reviewer experience
  - [computed: how the above attitudes might evolve in the future]
- Base population: the 932 authors and reviewers of the ICSE 2014, 2015, 2016 conferences
  - 29% response rate (32 countries), fewer junior people

Too many results to explain here

[futreview]: current peer review quality

Freie Universität Berlin

- "As an author, what percentage of the reviews that you receive is good, reasonable, unhelpful or grossly faulty?"
  - …then elaborates on these four terms



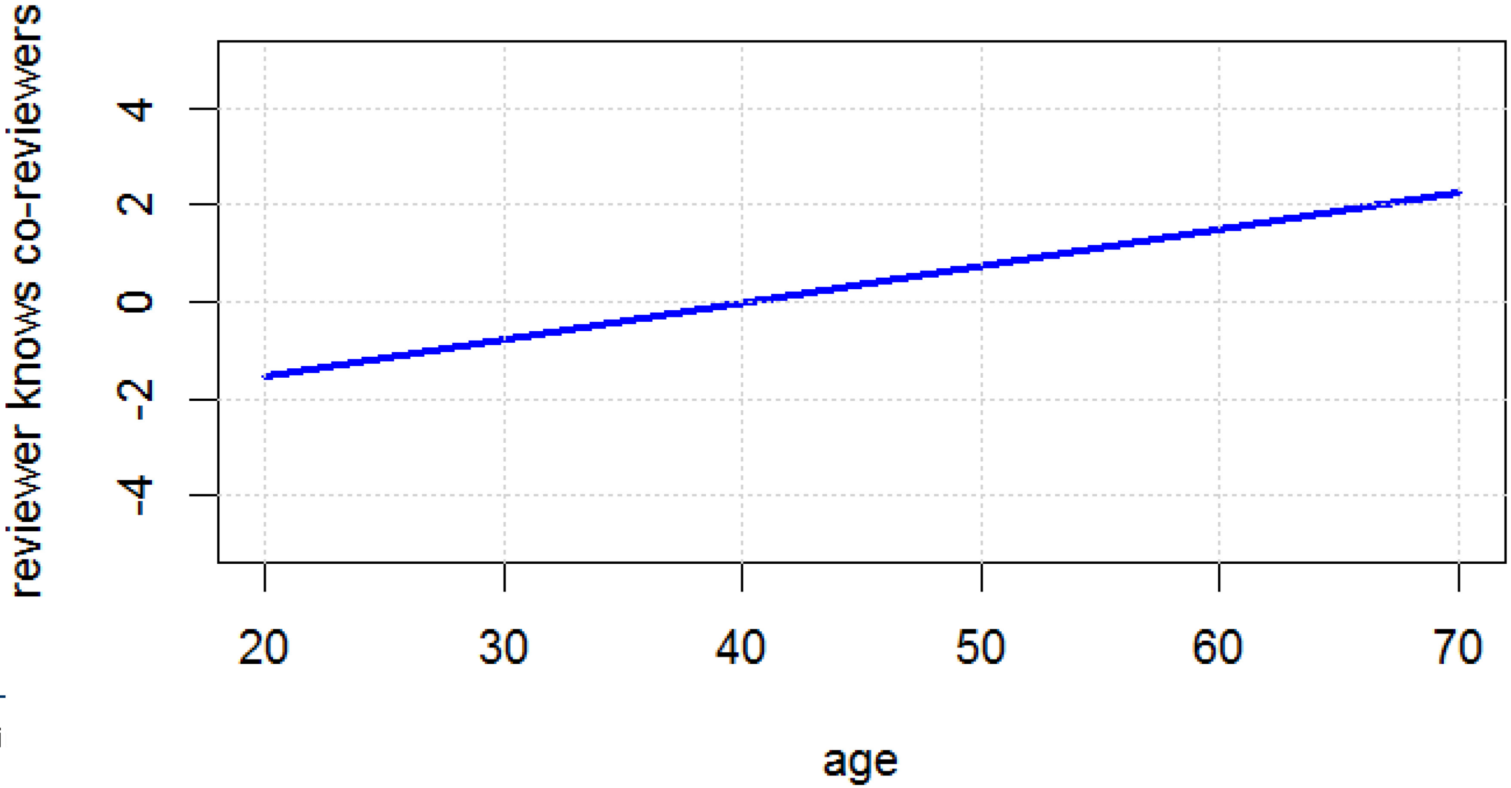
[futreview]: current peer review quality: Why?

- "In your opinion, what were the main reasons for unhelpful and/or grossly faulty reviews (if any)?"
  - answers as free text
  - use open coding to develop a categorization ➔

- Reviewers not allocating enough time (24%)
  - e.g. *"lack of time or effort"*
- Reviewers being insufficiently familiar with the topic of the work (22%)
  - e.g. *"In cases it is simply because the reviewer did not do his/her job, or accepted to referee a paper for which he/she was not qualified. But when you submit to good venues, with good PCs, that happens less frequently."*
- and 23 other (and much less mentioned) reasons

[futreview]: Future trends

- Consider age and professional status as predictors of attitude:
  - e.g. "Should reviewers know who their co-reviewers are?"
    - -5: strongly disagree, 5: strongly agree
  - age sometimes plays a role, professional status rarely does
  - ➔ these are generational trends, not seniority trends
  - ➔ they probably predict future attitudes   Example:



Summing up: Surveys

- Surveys can be a low-cost means of collecting interesting information
  - They can be cross-sectional, longitudinal, or retrospective

- Try to reuse or adapt existing questionnaires where possible
- Consider mixing quantitative and qualitative questions

- Carefully design and validate in any case
  - nice instructions: https://github.com/ds4se/chapters/blob/master/ermurph/survey-chapter.md

- Watch out for sampling bias!
  - You will almost always have some
  - but if you do not understand what it is, your data becomes dubious

Thank you!