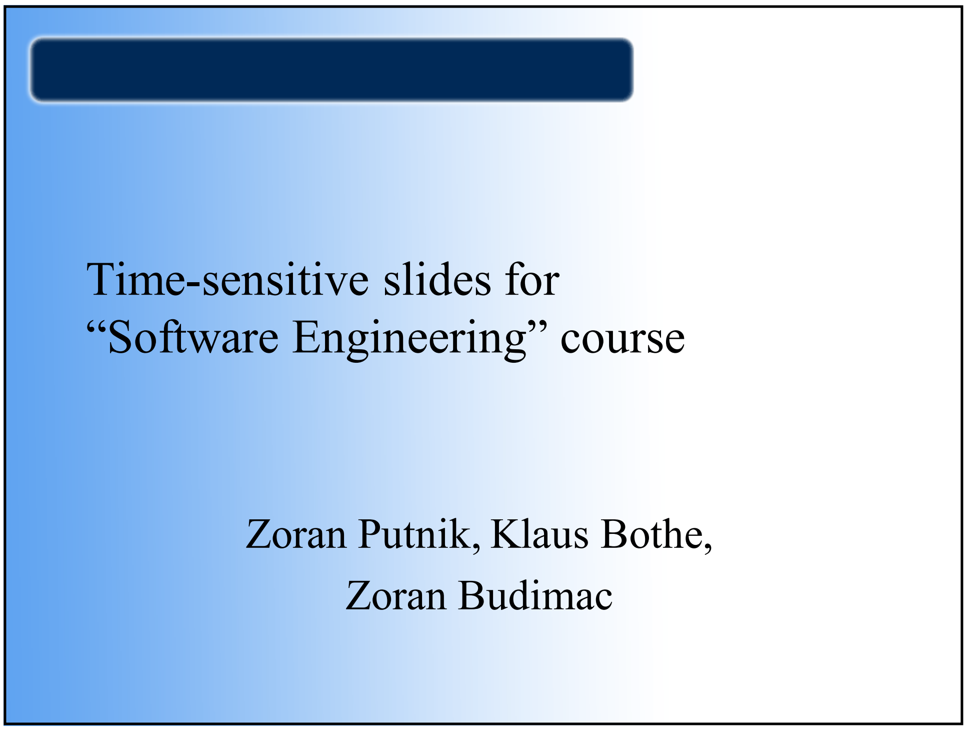
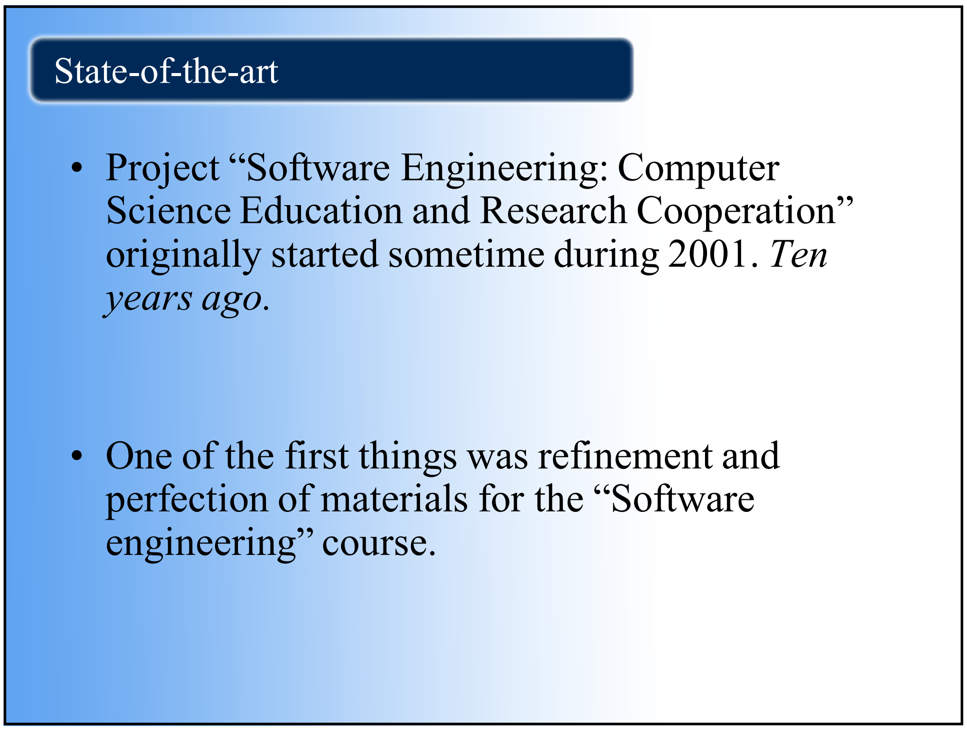


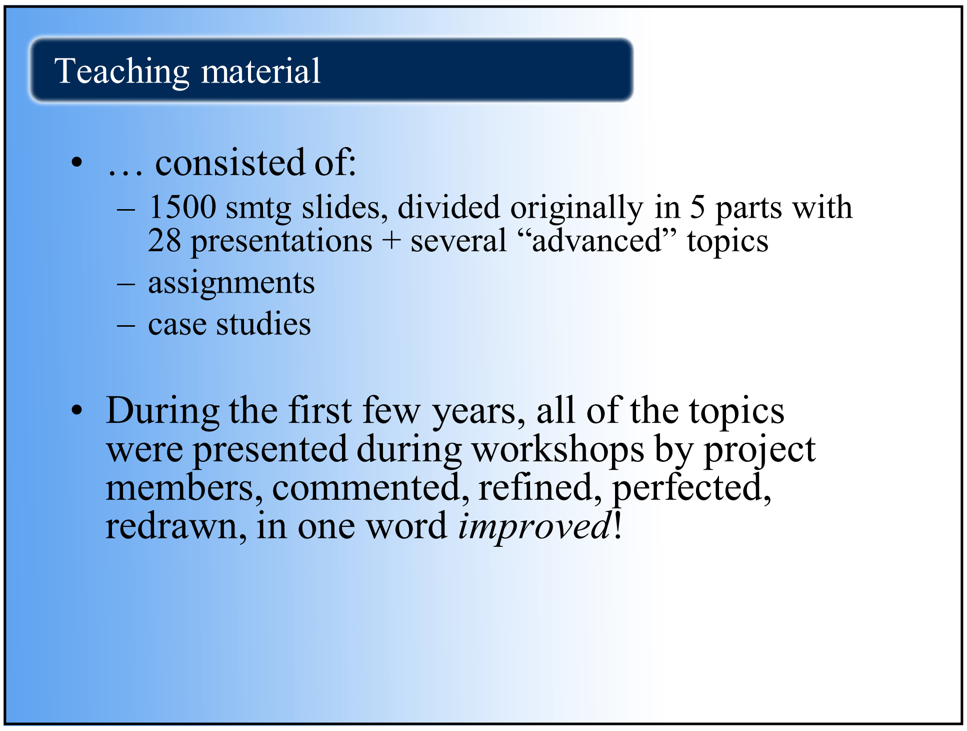
Time-sensitive slides for "Software Engineering" course

Zoran Putnik, Klaus Bothe,
Zoran Budimac



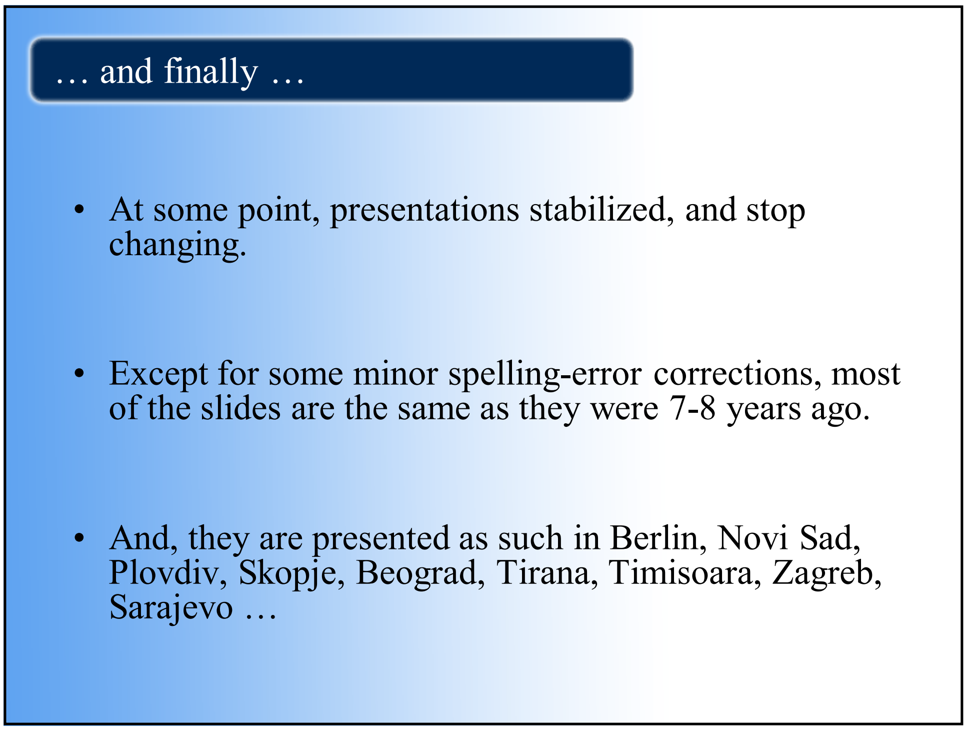
State-of-the-art

- Project "Software Engineering: Computer Science Education and Research Cooperation" originally started sometime during 2001. *Ten years ago.*
- One of the first things was refinement and perfection of materials for the "Software engineering" course.



Teaching material

- … consisted of:
  - 1500 smtg slides, divided originally in 5 parts with 28 presentations + several “advanced” topics
  - assignments
  - case studies
- During the first few years, all of the topics were presented during workshops by project members, commented, refined, perfected, redrawn, in one word *improved*!



… and finally …

- At some point, presentations stabilized, and stop changing.
- Except for some minor spelling-error corrections, most of the slides are the same as they were 7-8 years ago.
- And, they are presented as such in Berlin, Novi Sad, Plovdiv, Skopje, Beograd, Tirana, Timisoara, Zagreb, Sarajevo …

Situation today

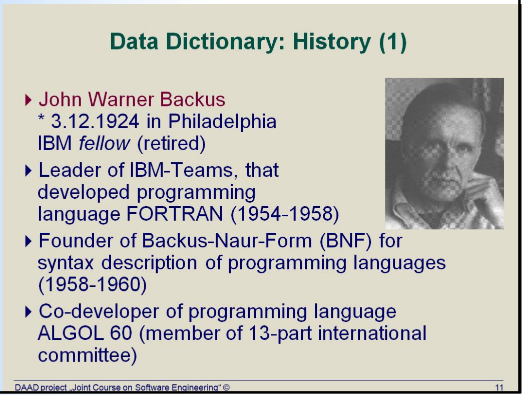
- Those who give mentioned presentations to students, might have encountered the same thing I did.
- Every now-and-then, you come across the slide that raises some innocent question by some of the students. For example:

Data Dictionary: History (1)

- John Warner Backus
  * 3.12.1924 in Philadelphia
  IBM *fellow* (retired)
- Leader of IBM-Teams, that developed programming language FORTRAN (1954-1958)
- Founder of Backus-Naur-Form (BNF) for syntax description of programming languages (1958-1960)
- Co-developer of programming language ALGOL 60 (member of 13-part international committee)

Question?

- "What happened to him in-the-meantime?"
- "Born 1924? Isn't that picture a little bit old then?"
- "Is he still alive?"



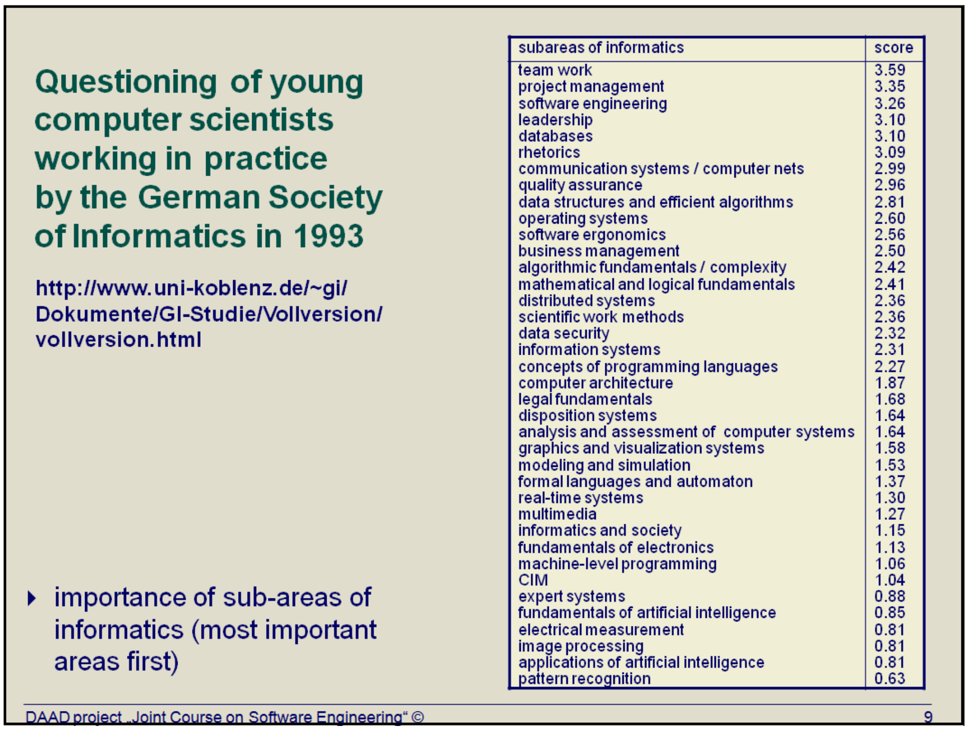
Data Dictionary: History (1)

- John Warner Backus
  * 3.12.1924 in Philadelphia
  IBM *fellow* (retired)
- Leader of IBM-Teams, that developed programming language FORTRAN (1954-1958)
- Founder of Backus-Naur-Form (BNF) for syntax description of programming languages (1958-1960)
- Co-developer of programming language ALGOL 60 (member of 13-part international committee)

DAAD project „Joint Course on Software Engineering" © 11

Questions of another type:

- Of course, some of the questions were not that innocent. Mean? Sarcastic? Or simply truthful?
- For example:



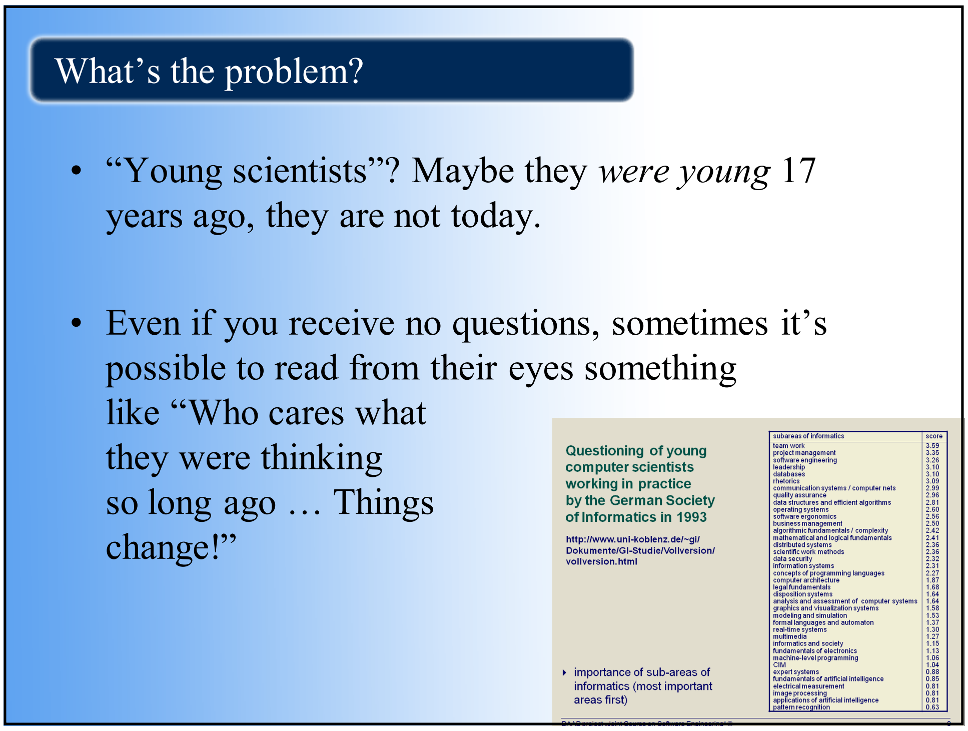
Questioning of young computer scientists working in practice by the German Society of Informatics in 1993

http://www.uni-koblenz.de/~gi/Dokumente/GI-Studie/Vollversion/vollversion.html

- importance of sub-areas of informatics (most important areas first)

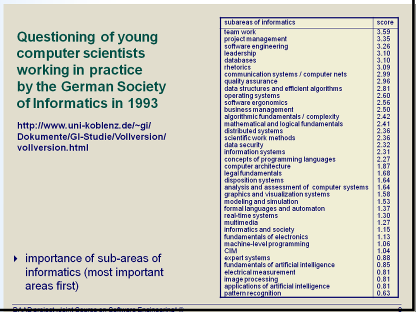
subareas of informatics	score
team work	3.59
project management	3.35
software engineering	3.26
leadership	3.10
databases	3.10
rhetorics	3.09
communication systems / computer nets	2.99
quality assurance	2.96
data structures and efficient algorithms	2.81
operating systems	2.60
software ergonomics	2.56
business management	2.50
algorithmic fundamentals / complexity	2.42
mathematical and logical fundamentals	2.41
distributed systems	2.36
scientific work methods	2.36
data security	2.32
information systems	2.31
concepts of programming languages	2.27
computer architecture	1.87
legal fundamentals	1.68
disposition systems	1.64
analysis and assessment of computer systems	1.64
graphics and visualization systems	1.58
modeling and simulation	1.53
formal languages and automaton	1.37
real-time systems	1.30
multimedia	1.27
informatics and society	1.15
fundamentals of electronics	1.13
machine-level programming	1.06
CIM	1.04
expert systems	0.88
fundamentals of artificial intelligence	0.85
electrical measurement	0.81
image processing	0.81
applications of artificial intelligence	0.81
pattern recognition	0.63

DAAD project „Joint Course on Software Engineering" © 9



What's the problem?

- "Young scientists"? Maybe they *were young* 17 years ago, they are not today.
- Even if you receive no questions, sometimes it's possible to read from their eyes something like "Who cares what they were thinking so long ago … Things change!"



Questioning of young computer scientists working in practice by the German Society of Informatics in 1993

http://www.uni-koblenz.de/~gi/Dokumente/GI-Studie/Vollversion/vollversion.html

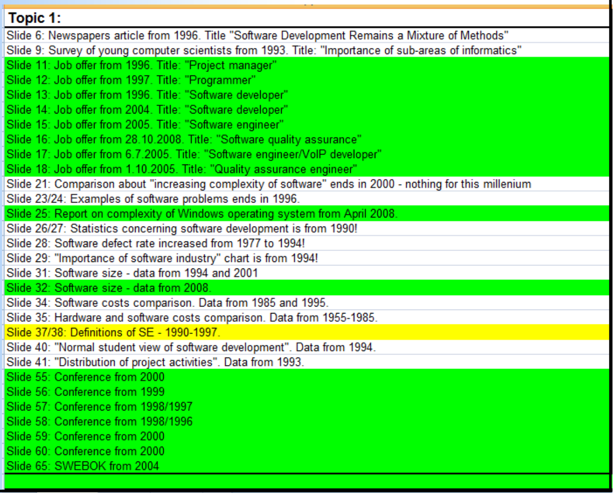
- importance of sub-areas of informatics (most important areas first)

Conclusion

- Either way, the field of computer science *is* fast-changing, so some checking is necessary.
- In agreement with prof. Bothe and prof Budimac, I tried to analyze the situation within our presentations, and check how “time sensitive” they are.

Analysis

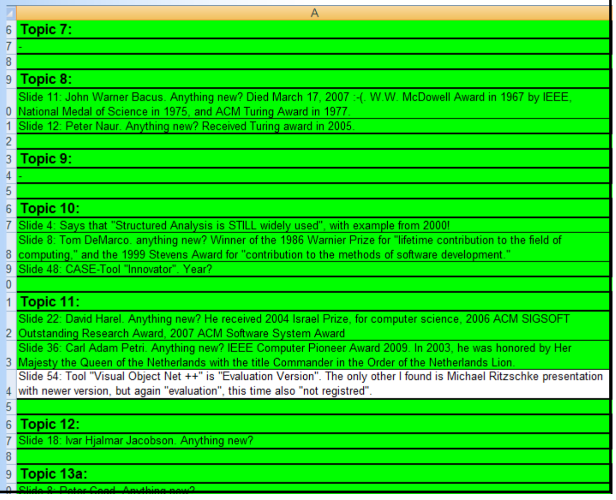
- I started with reading of slides from all presentations and making notes about slides that has some connection with time or date.
- Large number of slides, as you can see.



Topic 1:
Slide 6: Newspapers article from 1996. Title "Software Development Remains a Mixture of Methods"
Slide 9: Survey of young computer scientists from 1993. Title: "Importance of sub-areas of informatics"
Slide 11: Job offer from 1996. Title: "Project manager"
Slide 12: Job offer from 1997. Title: "Programmer"
Slide 13: Job offer from 1996. Title: "Software developer"
Slide 14: Job offer from 2004. Title: "Software developer"
Slide 15: Job offer from 2005. Title: "Software engineer"
Slide 16: Job offer from 28.10.2008. Title: "Software quality assurance"
Slide 17: Job offer from 6.7.2005. Title: "Software engineer/VoIP developer"
Slide 18: Job offer from 1.10.2005. Title: "Quality assurance engineer"
Slide 21: Comparison about "increasing complexity of software" ends in 2000 - nothing for this millenium
Slide 23/24: Examples of software problems ends in 1996.
Slide 25: Report on complexity of Windows operating system from April 2008.
Slide 26/27: Statistics concerning software development is from 1990!
Slide 28: Software defect rate increased from 1977 to 1994!
Slide 29: "Importance of software industry" chart is from 1994!
Slide 31: Software size - data from 1994 and 2001
Slide 32: Software size - data from 2008.
Slide 34: Software costs comparison. Data from 1985 and 1995.
Slide 35: Hardware and software costs comparison. Data from 1955-1985.
Slide 37/38: Definitions of SE - 1990-1997.
Slide 40: "Normal student view of software development". Data from 1994.
Slide 41: "Distribution of project activities". Data from 1993.
Slide 55: Conference from 2000
Slide 56: Conference from 1999
Slide 57: Conference from 1998/1997
Slide 58: Conference from 1998/1996
Slide 59: Conference from 2000
Slide 60: Conference from 2000
Slide 65: SWEBOK from 2004

Not that large, after all …

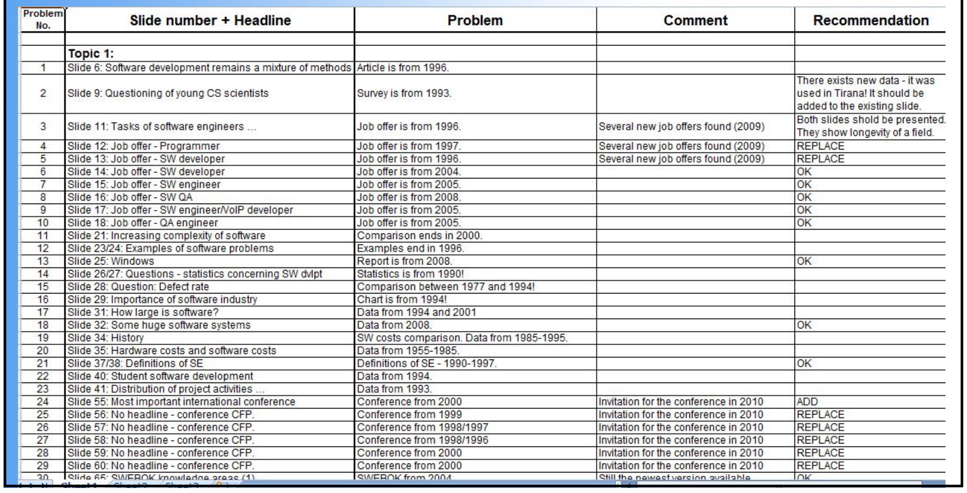
- The biggest problem was the first presentation, with all of the definitions, examples, notions, introductions …
- Later on, it was usually only a couple of slides per presentation.



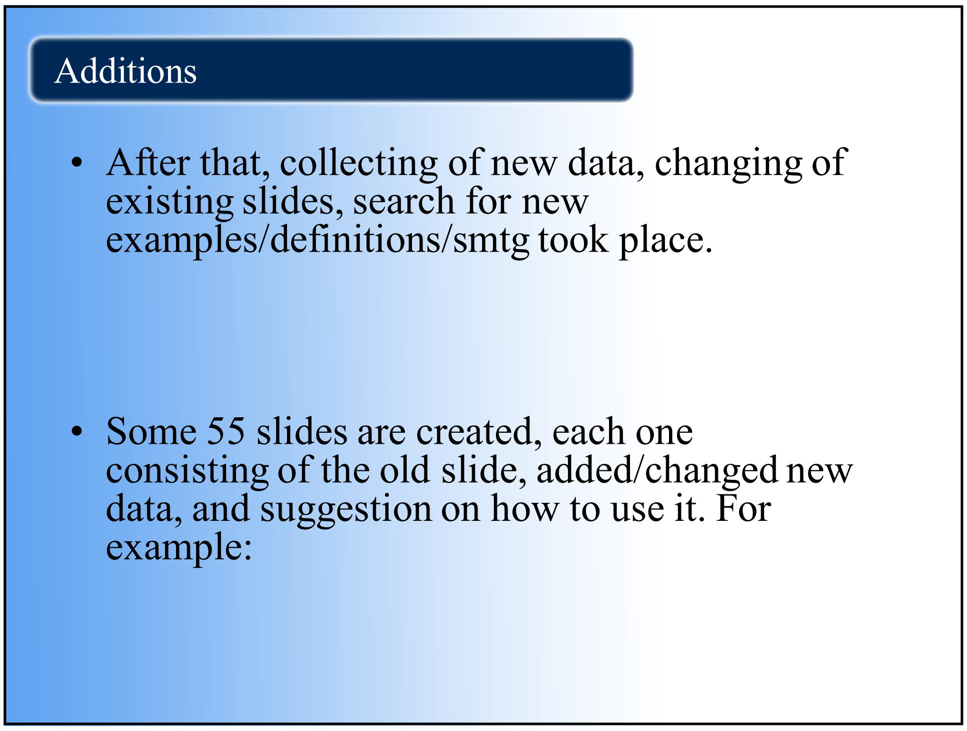
	A
6	**Topic 7:**
7	-
8	
9	**Topic 8:**
0	Slide 11: John Warner Bacus. Anything new? Died March 17, 2007 :-(. W.W. McDowell Award in 1967 by IEEE, National Medal of Science in 1975, and ACM Turing Award in 1977.
1	Slide 12: Peter Naur. Anything new? Received Turing award in 2005.
2	
3	**Topic 9:**
4	-
5	
6	**Topic 10:**
7	Slide 4: Says that "Structured Analysis is STILL widely used", with example from 2000!
8	Slide 8: Tom DeMarco. anything new? Winner of the 1986 Warnier Prize for "lifetime contribution to the field of computing," and the 1999 Stevens Award for "contribution to the methods of software development."
9	Slide 48: CASE-Tool "Innovator". Year?
0	
1	**Topic 11:**
2	Slide 22: David Harel. Anything new? He received 2004 Israel Prize, for computer science, 2006 ACM SIGSOFT Outstanding Research Award, 2007 ACM Software System Award
3	Slide 36: Carl Adam Petri. Anything new? IEEE Computer Pioneer Award 2009. In 2003, he was honored by Her Majesty the Queen of the Netherlands with the title Commander in the Order of the Netherlands Lion.
4	Slide 54: Tool "Visual Object Net ++" is "Evaluation Version". The only other I found is Michael Ritzschke presentation with newer version, but again "evaluation", this time also "not registred".
5	
6	**Topic 12:**
7	Slide 18: Ivar Hjalmar Jacobson. Anything new?
8	
9	**Topic 13a:**
0	Slide 8: Peter Coad. Anything new?

Analysis

- Than, the detailed analysis was considered.
- Topic/Slide number, what is the problem with it, what has been done so far, what should be done with the slide.

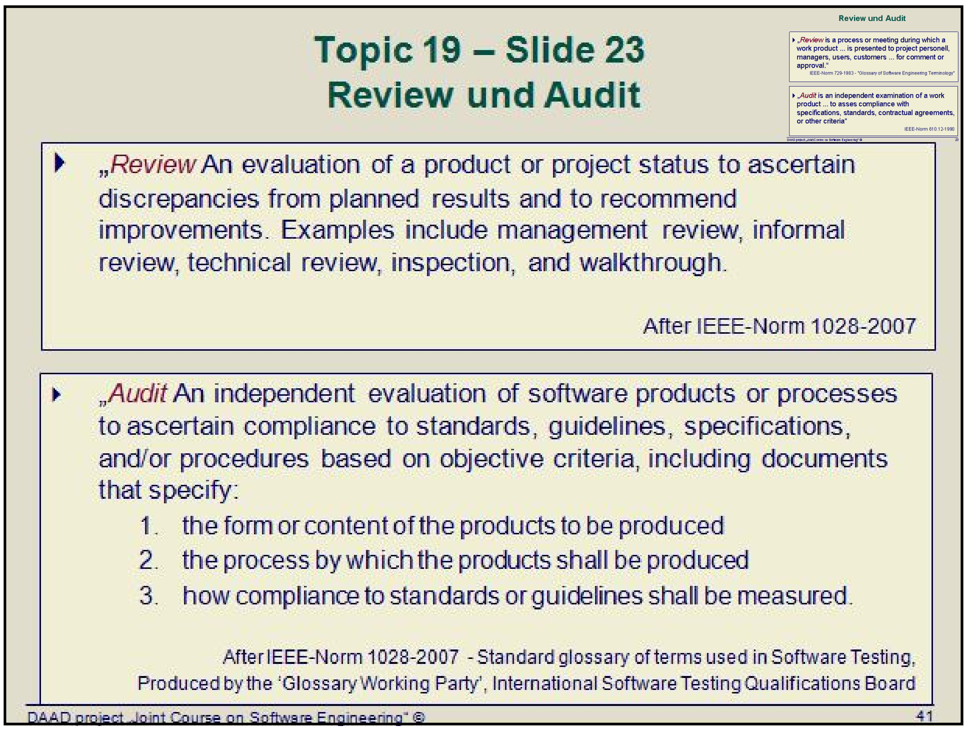


Problem No.	Slide number + Headline	Problem	Comment	Recommendation
	Topic 1:			
1	Slide 6: Software development remains a mixture of methods	Article is from 1996.		
2	Slide 9: Questioning of young CS scientists	Survey is from 1993.		There exists new data - it was used in Tirana! It should be added to the existing slide.
3	Slide 11: Tasks of software engineers …	Job offer is from 1996.	Several new job offers found (2009)	Both slides shold be presented. They show longevity of a field.
4	Slide 12: Job offer - Programmer	Job offer is from 1997.	Several new job offers found (2009)	REPLACE
5	Slide 13: Job offer - SW developer	Job offer is from 1996.	Several new job offers found (2009)	REPLACE
6	Slide 14: Job offer - SW developer	Job offer is from 2004.		OK
7	Slide 15: Job offer - SW engineer	Job offer is from 2005.		OK
8	Slide 16: Job offer - SW QA	Job offer is from 2008.		OK
9	Slide 17: Job offer - SW engineer/VoIP developer	Job offer is from 2005.		OK
10	Slide 18: Job offer - QA engineer	Job offer is from 2005.		OK
11	Slide 21: Increasing complexity of software	Comparison ends in 2000.		
12	Slide 23/24: Examples of software problems	Examples end in 1996.		
13	Slide 25: Windows	Report is from 2008.		OK
14	Slide 26/27: Questions - statistics concerning SW dvlpt	Statistics is from 1990!		
15	Slide 28: Question: Defect rate	Comparison between 1977 and 1994!		
16	Slide 29: Importance of software industry	Chart is from 1994!		
17	Slide 31: How large is software?	Data from 1994 and 2001		
18	Slide 32: Some huge software systems	Data from 2008.		OK
19	Slide 34: History	SW costs comparison. Data from 1985-1995.		
20	Slide 35: Hardware costs and software costs	Data from 1955-1985.		
21	Slide 37/38: Definitions of SE	Definitions of SE - 1990-1997.		OK
22	Slide 40: Student software development	Data from 1994.		
23	Slide 41: Distribution of project activities …	Data from 1993.		
24	Slide 55: Most important international conference	Conference from 2000	Invitation for the conference in 2010	ADD
25	Slide 56: No headline - conference CFP.	Conference from 1999	Invitation for the conference in 2010	REPLACE
26	Slide 57: No headline - conference CFP.	Conference from 1998/1997	Invitation for the conference in 2010	REPLACE
27	Slide 58: No headline - conference CFP.	Conference from 1998/1996	Invitation for the conference in 2010	REPLACE
28	Slide 59: No headline - conference CFP.	Conference from 2000	Invitation for the conference in 2010	REPLACE
29	Slide 60: No headline - conference CFP.	Conference from 2000	Invitation for the conference in 2010	REPLACE
30	Slide 65: SWEBOK knowledge areas (1)	SWEBOK from 2004	Still the newest version available	OK



Additions

- After that, collecting of new data, changing of existing slides, search for new examples/definitions/smtg took place.
- Some 55 slides are created, each one consisting of the old slide, added/changed new data, and suggestion on how to use it. For example:



Review und Audit

- „*Review* is a process or meeting during which a work product ... is presented to project personell, managers, users, customers ... for comment or approval.“

  IEEE-Norm 729-1983 - "Glossary of Software Engineering Terminology"

- „*Audit* is an independent examination of a work product ... to asses compliance with specifications, standards, contractual agreements, or other criteria“

  IEEE-Norm 610.12-1990

DAAD project „Joint Course on Software Engineering“ © 23

Topic 19 – Slide 23
Review und Audit

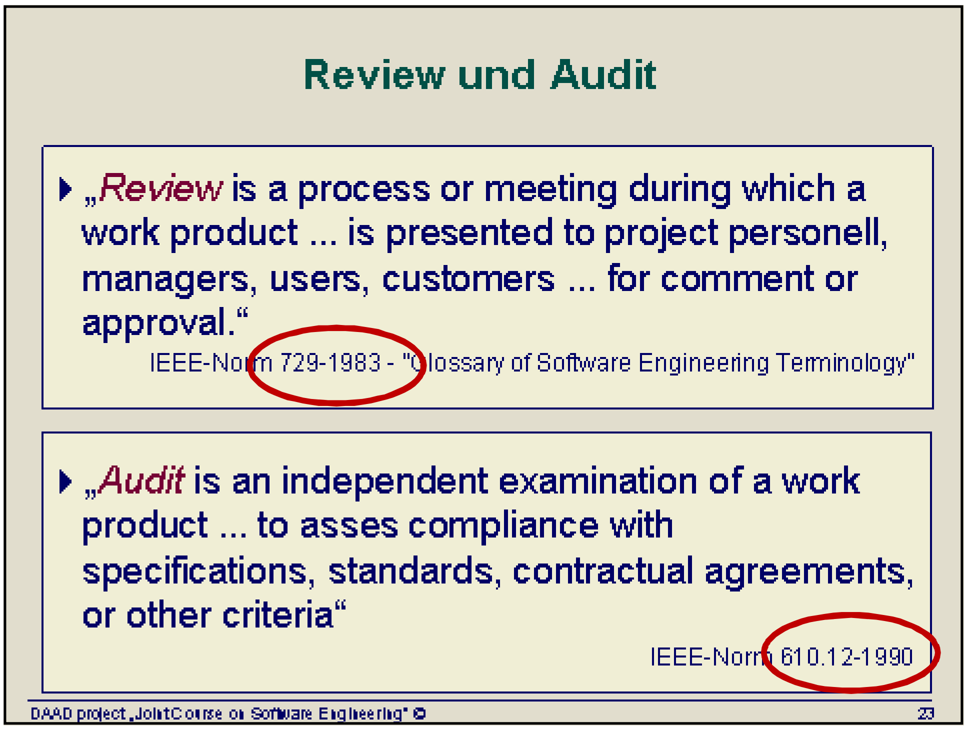
- „*Review* An evaluation of a product or project status to ascertain discrepancies from planned results and to recommend improvements. Examples include management review, informal review, technical review, inspection, and walkthrough.

  After IEEE-Norm 1028-2007

- „*Audit* An independent evaluation of software products or processes to ascertain compliance to standards, guidelines, specifications, and/or procedures based on objective criteria, including documents that specify:
  1. the form or content of the products to be produced
  2. the process by which the products shall be produced
  3. how compliance to standards or guidelines shall be measured.

  After IEEE-Norm 1028-2007 - Standard glossary of terms used in Software Testing, Produced by the 'Glossary Working Party', International Software Testing Qualifications Board

DAAD project „Joint Course on Software Engineering“ © 41



Review und Audit

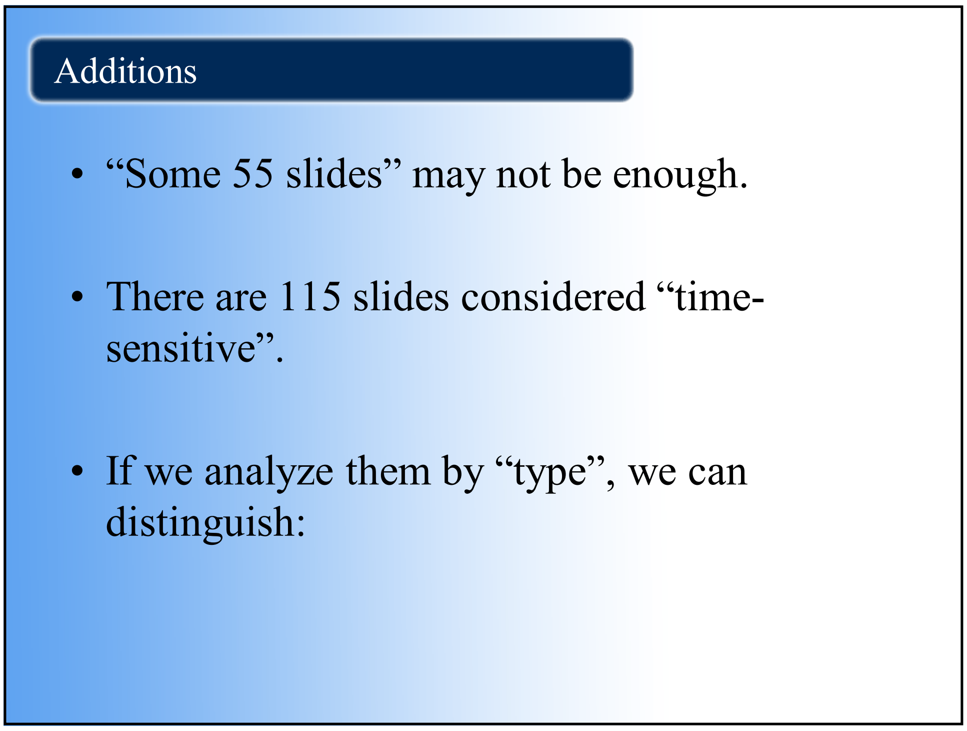
- „*Review* is a process or meeting during which a work product ... is presented to project personell, managers, users, customers ... for comment or approval.“

  IEEE-Norm 729-1983 - "Glossary of Software Engineering Terminology"

- „*Audit* is an independent examination of a work product ... to asses compliance with specifications, standards, contractual agreements, or other criteria“

  IEEE-Norm 610.12-1990

DAAD project „Joint Course on Software Engineering“ © 23



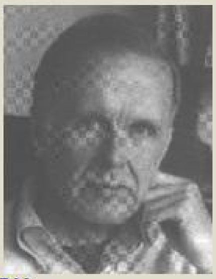
Additions

- “Some 55 slides” may not be enough.
- There are 115 slides considered “time-sensitive”.
- If we analyze them by “type”, we can distinguish:

Types of Time-sensitivity:

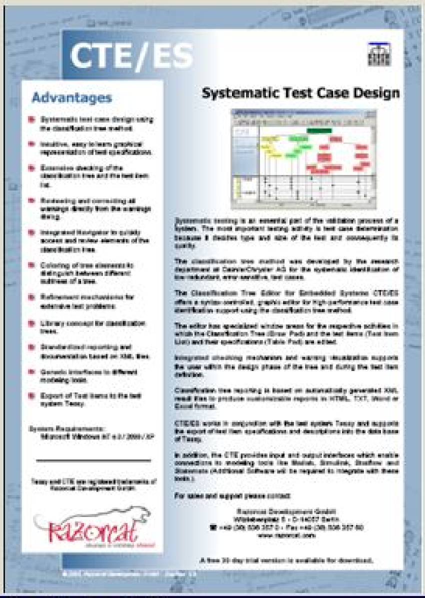
- Easily-solved
  - "Obsolete", "outdated" slides:
    - CFP for conferences from 1995-2000
    - Job offers from 20 years ago
    - Diploma/Master/PhD thesis from last century
    - Presentations of famous inventors/scientists
    - Extracts from various standards
  - Easily solved by replacing them with the new slides, or by adding data to old slides.
  - For example:

Topic 8 – Slide 11: Data Dictionary: History (1)



- John Warner Backus
  Born 3.12.1924 in Philadelphia,
  Died 17.3.2007., IBM *fellow* (retired)
- Leader of IBM-Teams, that developed programming language FORTRAN (1954-1958)
- Founder of Backus-Naur-Form (BNF) for syntax description of programming languages (1958-1960)
- Co-developer of programming language ALGOL 60 (member of 13-part international committee)
- Received:
  - W.W. McDowell Award by IEEE in 1967,
  - National Medal of Science in 1975,
  - ACM Turing Award in 1977

Topic 20 – Slide 47a


CTE/ES

Advantages

- Systematic test case design using the classification tree method.
- Intuitive, easy to learn graphical representation of test specifications.
- Extensive checking of the classification tree and the test item list.
- Reviewing and correcting all warnings directly from the warnings dialog.
- Integrated Navigator to quickly access and review elements of the classification tree.
- Coloring of tree elements to distinguish between different subtrees of a tree.
- Refinement mechanisms for extensive test problems.
- Library concept for classification trees.
- Standardized reporting and documentation based on XML files.
- Generic interfaces to different modeling tools.
- Export of Test Items to the test system Tessy.

System Requirements:
Microsoft Windows NT 4.0 / 2000 / XP

Tessy and CTE are registered trademarks of Razorcat Development GmbH.

Razorcat

Systematic Test Case Design

Systematic testing is an essential part of the validation process of a system. The most important testing activity is test case determination because it decides type and size of the test and consequently its quality.

The classification tree method was developed by the research department at DaimlerChrysler AG for the systematic identification of low-redundant, error-sensitive, test cases.

The Classification Tree Editor for Embedded Systems CTE/ES offers a syntax-controlled, graphic editor for high-performance test case identification support using the classification tree method.

The editor has specialized window areas for the respective activities in which the Classification Tree (Draw Pad) and the test items (Test Item List) and their specifications (Table Pad) are edited.

Integrated checking mechanism and warning visualization supports the user within the design phase of the tree and during the test item definition.

Classification tree reporting is based on automatically generated XML result files to produce customizable reports in HTML, TXT, Word or Excel format.

CTE/ES works in conjunction with the test system Tessy and supports the export of test item specifications and descriptions into the data base of Tessy.

In addition, the CTE provides input and output interfaces which enable connections to modeling tools like Matlab, Simulink, Stateflow and Statemate (Additional Software will be required to integrate with these tools.)

For sales and support please contact:

Razorcat Development GmbH
Witzlebenplatz 5 · D-14057 Berlin
☎ +49 (30) 536 357 0 · Fax +49 (30) 536 357 60
www.razorcat.com

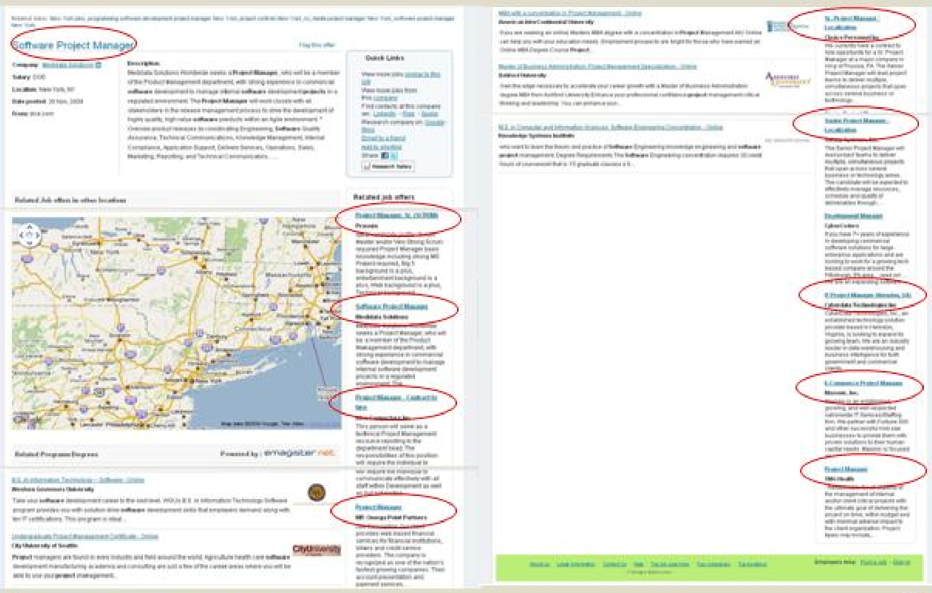
A free 30-day trial version is available for download.

© 2003 Razorcat Development GmbH [illegible]

Types of Time-sensitivity:

- Solved by addition
  - Slides from 10-20 years ago:
    - Invitation for the conference about some sub-field of software engineering
    - Job offer
    - Literature
    - Numerical data about something
  - All of the mentioned have their value:
    - they show longevity of the field,
    - they show trends in data
    - they show new developments
  - Suggestion – leave those AND add the new data. For example:

Topic 1 – Slide 11



Software Project Manager

Quick Links

Related job offers

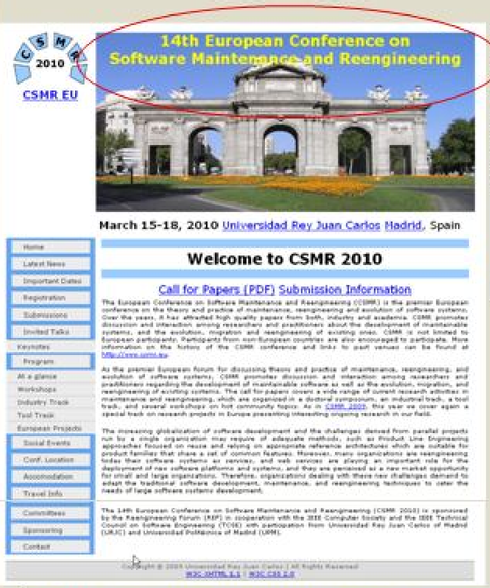
Related Job offers in other locations

Related Programs Degrees

Powered by: emagister.net

DAAD project „Joint Course on Software Engineering" © 2

Topic 1 – Slide 58



CSMR 2010

CSMR EU

14th European Conference on Software Maintenance and Reengineering

March 15-18, 2010 Universidad Rey Juan Carlos Madrid, Spain

Home
Latest News
Important Dates
Registration
Submissions
Invited Talks
Keynotes
Program
At a glance
Workshops
Industry Track
Tool Track
European Projects
Social Events
Conf. Location
Accomodation
Travel Info
Committees
Sponsoring
Contact

Welcome to CSMR 2010

Call for Papers (PDF) Submission Information

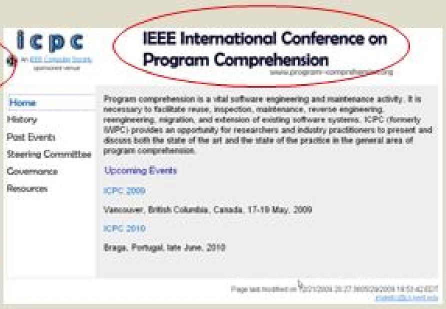
The European Conference on Software Maintenance and Reengineering (CSMR) is the premier European conference on the theory and practice of maintenance, reengineering and evolution of software systems. Over the years, it has attracted high quality papers from both, industry and academia. CSMR promotes discussion and interaction among researchers and practitioners about the development of maintainable systems, and the evolution, migration and reengineering of existing ones. CSMR is not limited to European participants. Participants from non-European countries are also encouraged to participate. More information on the history of the CSMR conference and links to past venues can be found at http://www.csmr.eu

As the premier European forum for discussing theory and practice of maintenance, reengineering, and evolution of software systems, CSMR promotes discussion and interaction among researchers and practitioners regarding the development of maintainable software as well as the evolution, migration, and reengineering of existing systems. The call for papers covers a wide range of current research activities in maintenance and reengineering, which are organized in a doctoral symposium, an industrial track, a tool track, and several workshops on hot community topics. As in CSMR 2009, this year we cover again a special track on research projects in Europe presenting interesting ongoing research in our field.

The increasing globalization of software development and the challenges derived from parallel projects run by a single organization may require of adequate methods, such as Product Line Engineering approaches focused on reuse and relying on appropriate reference architectures which are suitable for product families that share a set of common features. Moreover, many organizations are reengineering today their software systems as services, and web services are playing an important role for the deployment of new software platforms and systems, and they are perceived as a new market opportunity for small and large organizations. Therefore, organizations dealing with these new challenges demand to adapt the traditional software development, maintenance, and reengineering techniques to cater the needs of large software systems development.

The 14th European Conference on Software Maintenance and Reengineering (CSMR 2010) is sponsored by the Reengineering Forum (REF) in cooperation with the IEEE Computer Society and the IEEE Technical Council on Software Engineering (TCSE) with participation from Universidad Rey Juan Carlos of Madrid (URJC) and Universidad Politécnica of Madrid (UPM).

Copyright © 2009 Universidad Rey Juan Carlos | All Rights Reserved
W3C XHTML 1.1 | W3C CSS 2.0



icpc

An IEEE Computer Society sponsored venue

IEEE International Conference on Program Comprehension

www.program-comprehension.org

Home
History
Past Events
Steering Committee
Governance
Resources

Program comprehension is a vital software engineering and maintenance activity. It is necessary to facilitate reuse, inspection, maintenance, reverse engineering, reengineering, migration, and extension of existing software systems. ICPC (formerly IWPC) provides an opportunity for researchers and industry practitioners to present and discuss both the state of the art and the state of the practice in the general area of program comprehension.

Upcoming Events

ICPC 2009

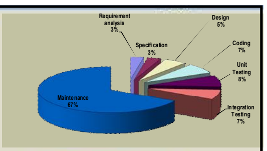
Vancouver, British Columbia, Canada, 17-19 May, 2009

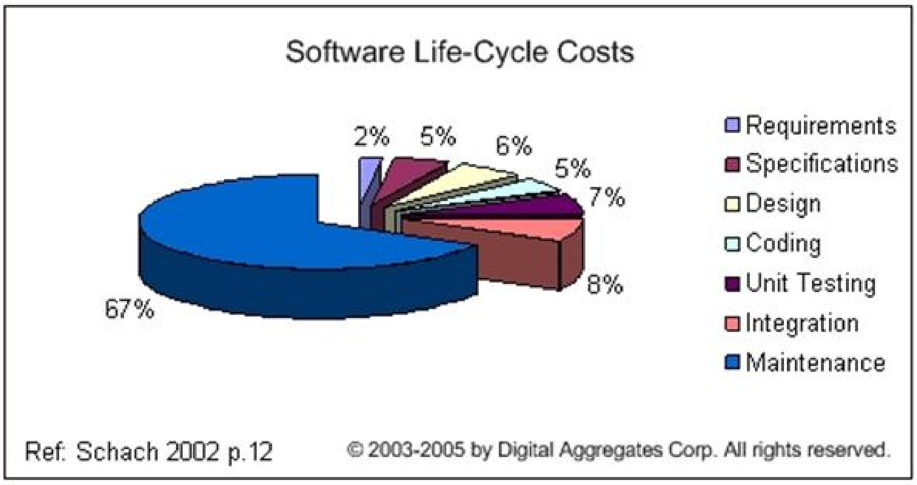
ICPC 2010

Braga, Portugal, late June, 2010

DAAD project „Joint Course on Software Engineering" © 9

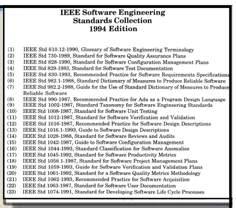
Topic 3 – Slide 20





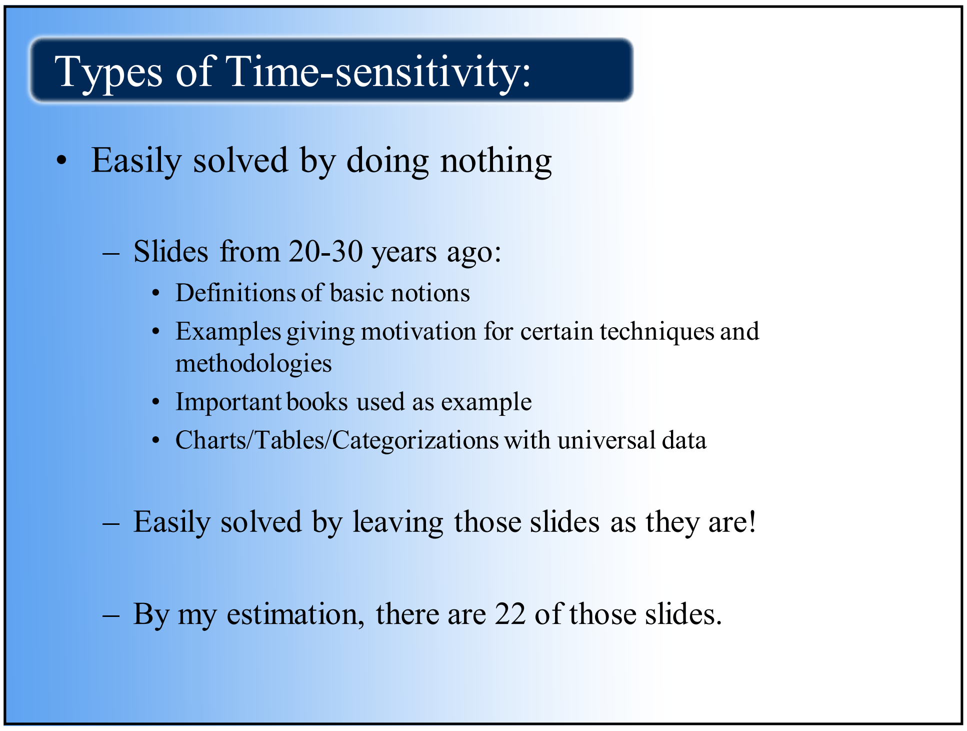
Stephen Schach, "*Object-Oriented and Classical Software Engineering*" [McGraw-Hill, 2002]

Topic 5 – Slide 23



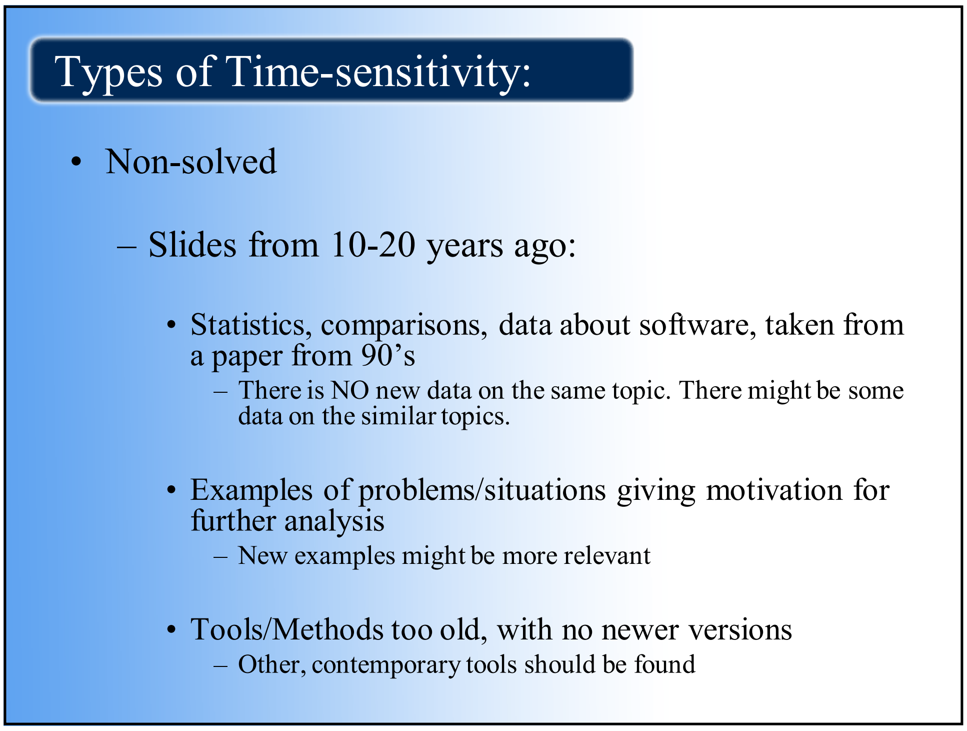
IEEE Software Engineering Standards Collection 1994 Edition

(1) IEEE Std 610.12-1990, Glossary of Software Engineering Terminology
(2) IEEE Std 730-1989, Standard for Software Quality Assurance Plans
(3) IEEE Std 828-1990, Standard for Software Configuration Management Plans
(4) IEEE Std 829-1983, Standard for Software Test Documentation
(5) IEEE Std 830-1993, Recommended Practice for Software Requirements Specifications
(6) IEEE Std 982.1-1988, Standard Dictionary of Measures to Produce Reliable Software
(7) IEEE Std 982.2-1988, Guide for the Use of Standard Dictionary of Measures to Produce Reliable Software
(8) IEEE Std 990-1987, Recommended Practice for Ada as a Program Design Language
(9) IEEE Std 1002-1987, Standard Taxonomy for Software Engineering Standards
(10) IEEE Std 1008-1987, Standard for Software Unit Testing
(11) IEEE Std 1012-1987, Standard for Software Verification and Validation
(12) IEEE Std 1016-1987, Recommended Practice for Software Design Descriptions
(13) IEEE Std 1016.1-1993, Guide to Software Design Descriptions
(14) IEEE Std 1028-1988, Standard for Software Reviews and Audits
(15) IEEE Std 1042-1987, Guide to Software Configuration Management
(16) IEEE Std 1044-1993, Standard Classification for Software Anomalies
(17) IEEE Std 1045-1992, Standard for Software Productivity Metrics
(18) IEEE Std 1058.1-1987, Standard for Software Project Management Plans
(19) IEEE Std 1059-1993, Guide for Software Verification and Validation Plans
(20) IEEE Std 1061-1992, Standard for a Software Quality Metrics Methodology
(21) IEEE Std 1062-1993, Recommended Practice for Software Acquisition
(22) IEEE Std 1063-1987, Standard for Software User Documentation
(23) IEEE Std 1074-1991, Standard for Developing Software Life Cycle Processes



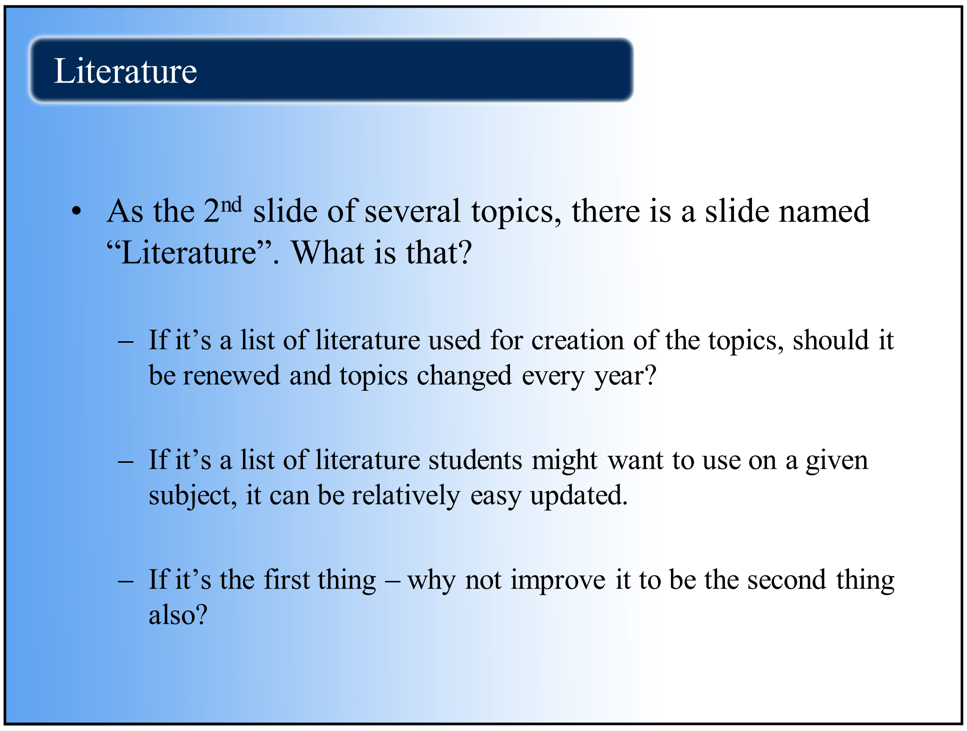
Types of Time-sensitivity:

- Easily solved by doing nothing
  - Slides from 20-30 years ago:
    - Definitions of basic notions
    - Examples giving motivation for certain techniques and methodologies
    - Important books used as example
    - Charts/Tables/Categorizations with universal data
  - Easily solved by leaving those slides as they are!
  - By my estimation, there are 22 of those slides.



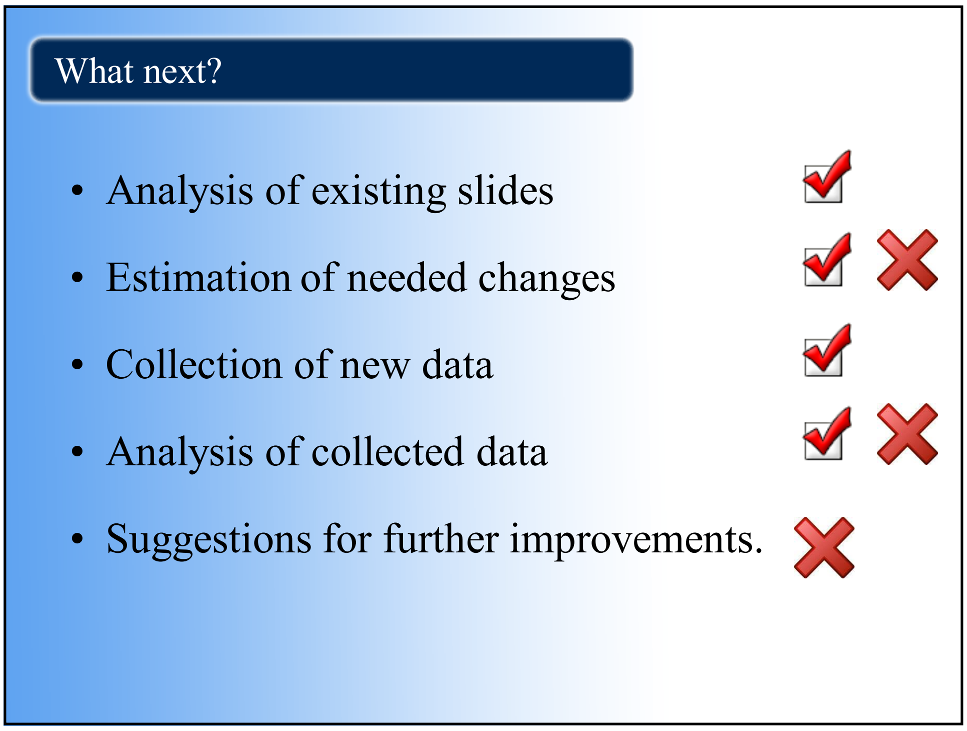
Types of Time-sensitivity:

- Non-solved
  - Slides from 10-20 years ago:
    - Statistics, comparisons, data about software, taken from a paper from 90's
      - There is NO new data on the same topic. There might be some data on the similar topics.
    - Examples of problems/situations giving motivation for further analysis
      - New examples might be more relevant
    - Tools/Methods too old, with no newer versions
      - Other, contemporary tools should be found



Literature

- As the 2nd slide of several topics, there is a slide named “Literature”. What is that?
  - If it’s a list of literature used for creation of the topics, should it be renewed and topics changed every year?
  - If it’s a list of literature students might want to use on a given subject, it can be relatively easy updated.
  - If it’s the first thing – why not improve it to be the second thing also?



What next?

- Analysis of existing slides ✓
- Estimation of needed changes ✓ ✗
- Collection of new data ✓
- Analysis of collected data ✓ ✗
- Suggestions for further improvements. ✗