Software Quality Management

Dr. Stefan Wagner
Technische Universität München

Garching
23 April 2010

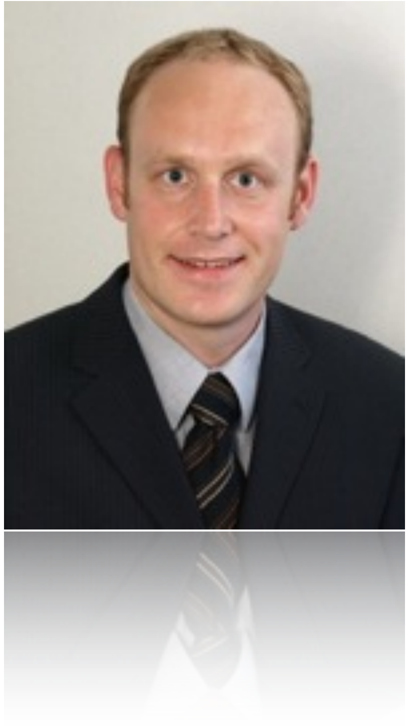
Lecturer

Stefan Wagner

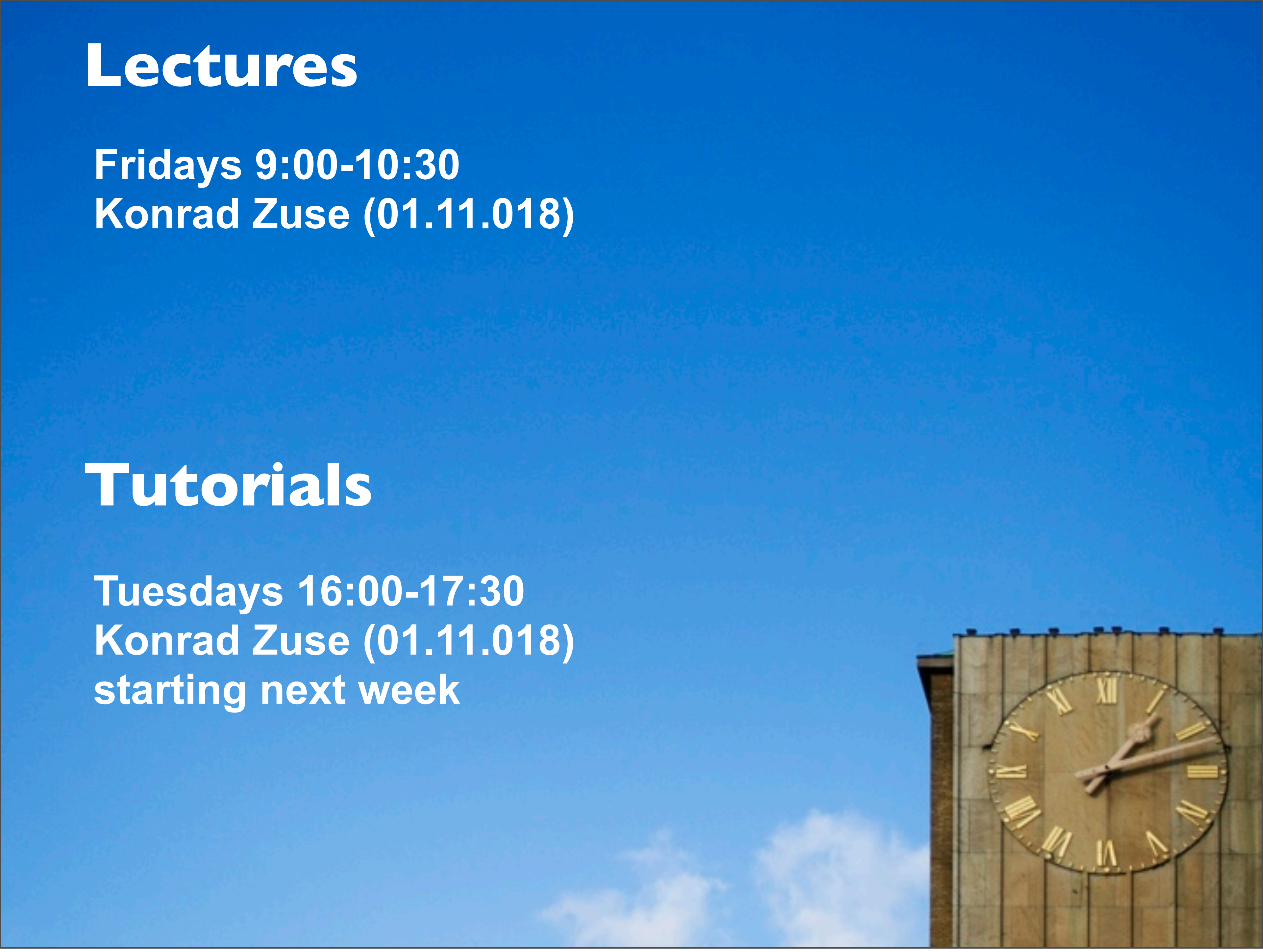
wagnerst@in.tum.de

Teaching Assistants

Lars
Heinemann



Shareeful
Islam



Lectures

Fridays 9:00-10:30
Konrad Zuse (01.11.018)

Tutorials

Tuesdays 16:00-17:30
Konrad Zuse (01.11.018)
starting next week

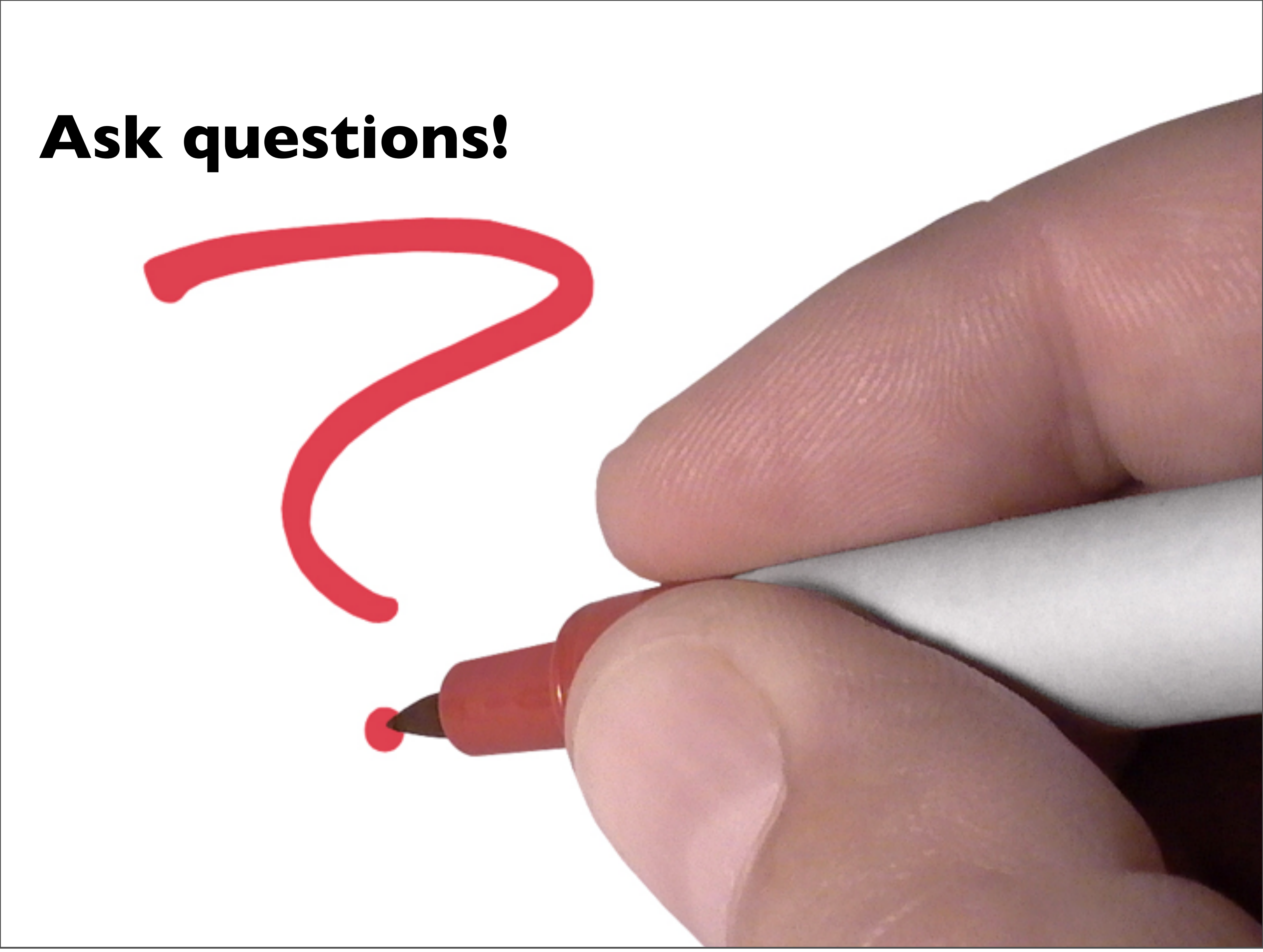
Times are possible to change if the majority of the students prefer another time and we find a room.

Exam

27 July 2010

6 ETCS

The credits are completely based on the exam.
Open book exam (you can use all kinds of written notes)
Duration: 90 minutes
The exam will cover the lectures and the tutorials.
The question style will be similar as in the tutorials.
We might change to an oral exam if only a small number of students need credits.



Ask questions!

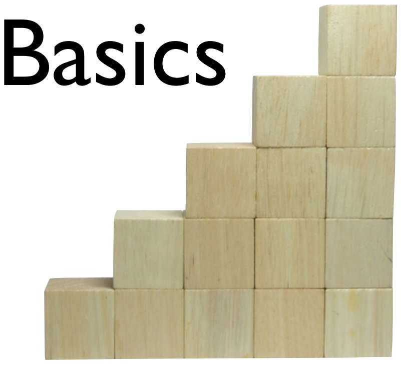
Ask students for background and expectations -> collection on white board -> take picture and recheck

Rules of the course:

* Be on time!
* Don't disturb the others with unrelated activities!
* Be active and ask questions!
* Give us feedback if we cover something too quickly, not deep enough or if there are any other problems!
* Feedback for me: Written answer at the end of the course to one question I pose at the beginning.

    Today: What is the most important quality attribute and why?

Basics



Product

Quality

Metrics and

Measurement

Process

Quality

Quality

Management

Certification

The overview of the topics we are going to cover.

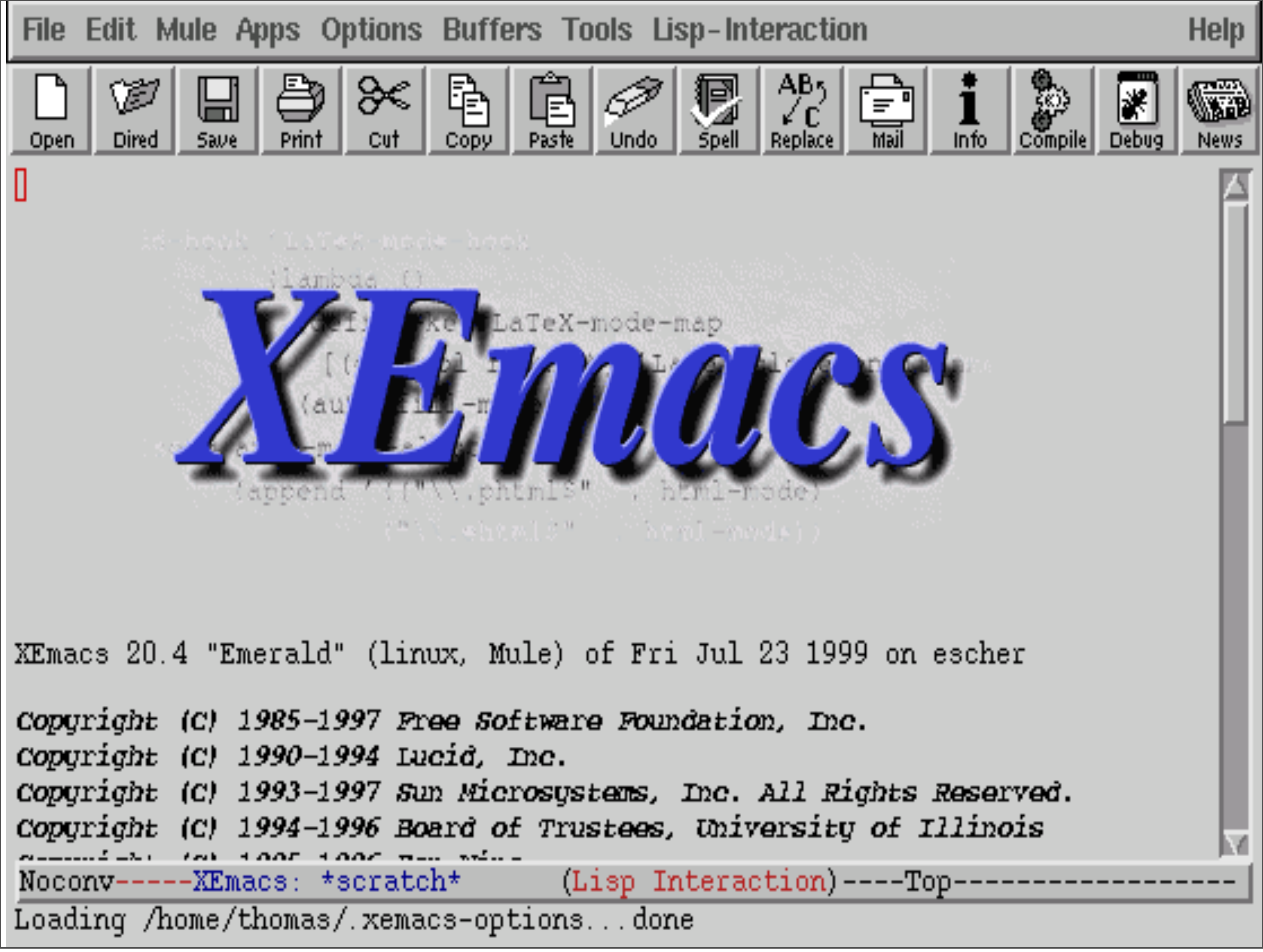
Product quality is a complex concept.
What does it mean outside of software?
What is product quality in cars?
Long-lived?
Doesn't brake down often?
Low maintenance costs?

Product quality in watches?
Very exact?

Product quality in consumer electronics?
Easy to understand?
Intuitive?

Only two discovered remote holes in the default install in more than 10 years

How is product quality different in software?
Some consider OpenBSD of high quality, because it is comparably hard to break the operating system.
Is good support against attackers high quality?

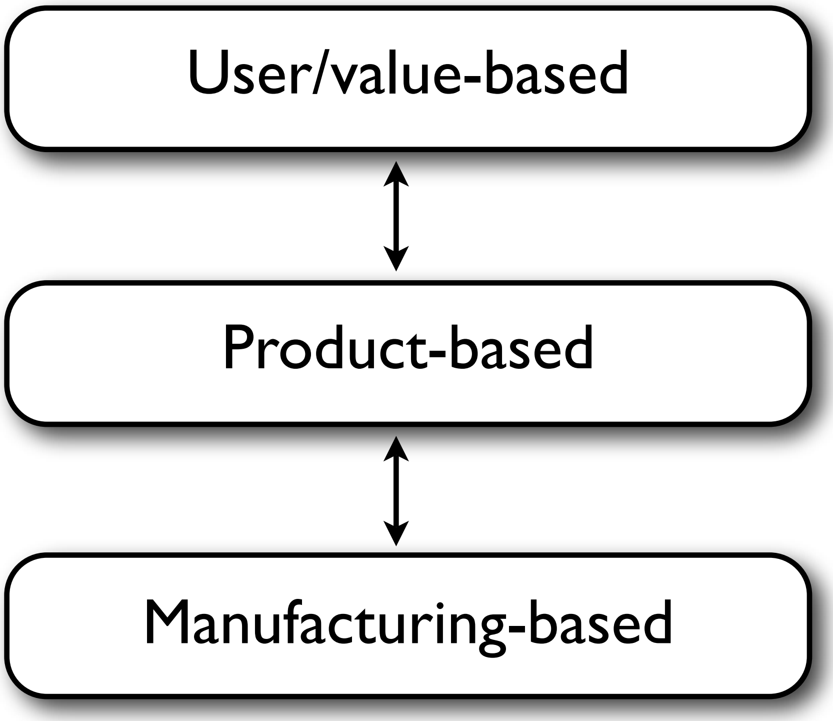


```
XEmacs 20.4 "Emerald" (linux, Mule) of Fri Jul 23 1999 on escher

Copyright (C) 1985-1997 Free Software Foundation, Inc.
Copyright (C) 1990-1994 Lucid, Inc.
Copyright (C) 1993-1997 Sun Microsystems, Inc. All Rights Reserved.
Copyright (C) 1994-1996 Board of Trustees, University of Illinois
Noconv-----XEmacs: *scratch*      (Lisp Interaction)----Top-----------------
Loading /home/thomas/.xemacs-options...done
```

Also Emacs is considered by many of high quality.
It has been used over a long time.
It has many features and it is well suited to be extended.
An expert can work extremely quickly with it.
Others argue that it is very hard to learn and is therefore of low quality.

Garvin's
quality approaches



User/value-based

Product-based

Manufacturing-based

Garvin, What does product quality really mean?, 1984

Garvin wrote a highly cited paper about product quality in general.
He argues that that there is not only "one quality", but there are different approaches or views.
He starts with the transcendent approach: "I know it when I see it"
And moves to a product-based and measureable view.
Furthermore, there is the view on the "manufacturing", which in software is more the process. The process could ensure product quality.
The conclusion is that different view are important at different points in the development process.

1. Start with the user or value view to discuss with stakeholders and define requirements.
2. Transfer this to the product view and specify measureable properties of the product.
3. Take the manufacturing (or process) view and execute a process that ensures that the product has these properties.

-> Write approaches on the whiteboard

"Quality means
conformance to requirements."

Crosby (1979)

-> Question to audience: Which approach?
Mostly product-view

"Quality consists of those product features which meet the needs of customers and thereby provide product satisfaction."

"Quality consists of
freedom from deficiencies."

Juran (1988)

-> Question to audience: Which approach?
Combination of value/user view and product view

> "Conformance to
> explicitly stated functional and
> performance requirements,
> explicitly documented
> development standards, and
> implicit characteristics that are expected
> of all professionally developed software."

Pressman (2000)

-> Question to audience: Which approach?
Mostly product view
The "implicit characteristics that are expected" hint also at a user view.
Explicitly documented development standards hint at manufacturing view.

"Software Quality is:
the degree to which a system, component, or process meets specified requirements.

the degree to which a system, component, or process meets customer or user needs or expectations."

IEEE (1991)

-> Question to audience: Which approach?
Explicitly product view as well as user/value view

„Like beauty,
everyone may have his or her idea
of what quality is“

ISO 9000:2000

-> Question to audience: Which approach?
Transcendent view

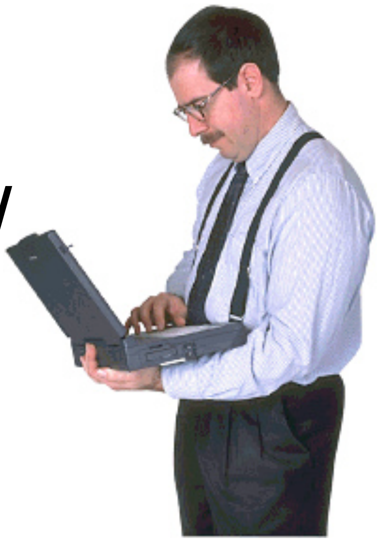
Quality

I know it
when I see it

People have difficulty with specifying quality: "I can say what quality is, when I see it."
-> Similar to the way an American judge defined pornography
Sometimes this is sufficient. Highly iterative or agile development can help to mitigate this problem.
In general this is not enough: How does the developer now what and how to achieve quality?

Customer

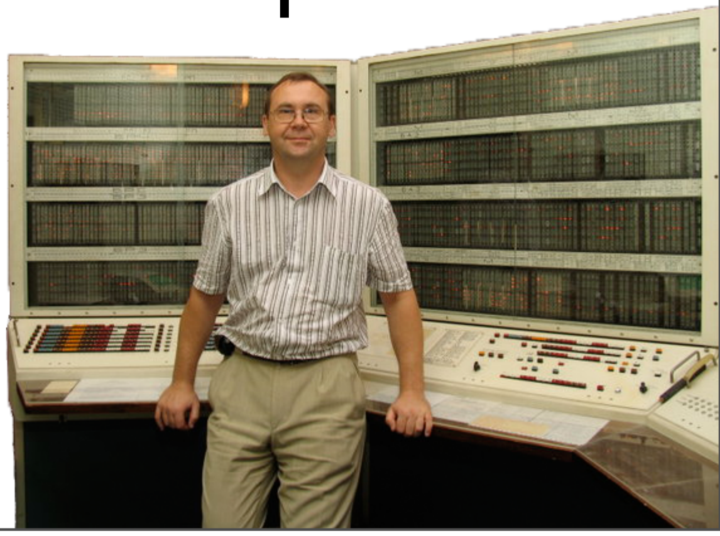
Developer/
Maintainer



Stakeholder

User

Operator



In development as well as maintenance, different stakeholders are involved.
Very common are the user, the developer, the operator, and the customer. But there are more.
All have their own, sometimes conflicting, and often implicit needs and goals.
If and how these needs are fulfilled is closely related to the quality of the software.

What is **software quality**?

So again: What is software quality?
This is very specific for the stakeholder!
The customer wants a good relation of costs and benefits.
The user wants to perform his tasks with ease and satisfyingly.
The developer wants well-written, easy-to-understand code.
The operator is concerned about needed file space or processing power.

Quality profiles

Quality, however, is not only specific for the stakeholder.
Also the domain and the purpose of the software has a high influence on its quality profile.
-> Question to students: What's a quality profile
A quality profile defines what kinds of qualities (often called quality attributes or quality characteristics) are more or less important.
A web retailer like Amazon is dependent on the availability of its systems.
Porsche has its major focus on safety.
Deutsche Bank probably needs a high availability, but they also have a high responsibility for the privacy of their customer data.
-> Visualise quality profile as comb on whiteboard.

Quality
needs to be specified

Quality Requirements

REQ 1372: The system shall be easy to maintain.
REQ 1373: Sensitive data shall be secured from unauthorised access.
REQ 1374: The system shall not crash.

Because of these many different views, stakeholders, domains, and purposes, it is necessary to specify the needed quality.
This is usually part of the software requirements specification (SRS).
They are handled extremely diverse in practice.
A lot of the quality requirements are – despite the differences – quite similar over different products.

"The system shall be secure."

If quality is specified, the suitable level of abstraction is often not clear.
Often requirements are specified on a very abstract level: "The system shall be secure." Again, how should the developer know how to implement this? It has to be specified that the castle needs walls, and towers, and surrounding water.
Sometimes, however, there are also long lists of very specific, technical requirements without a clear rationale.
Therefore, many base their requirements on quality models as a reference.

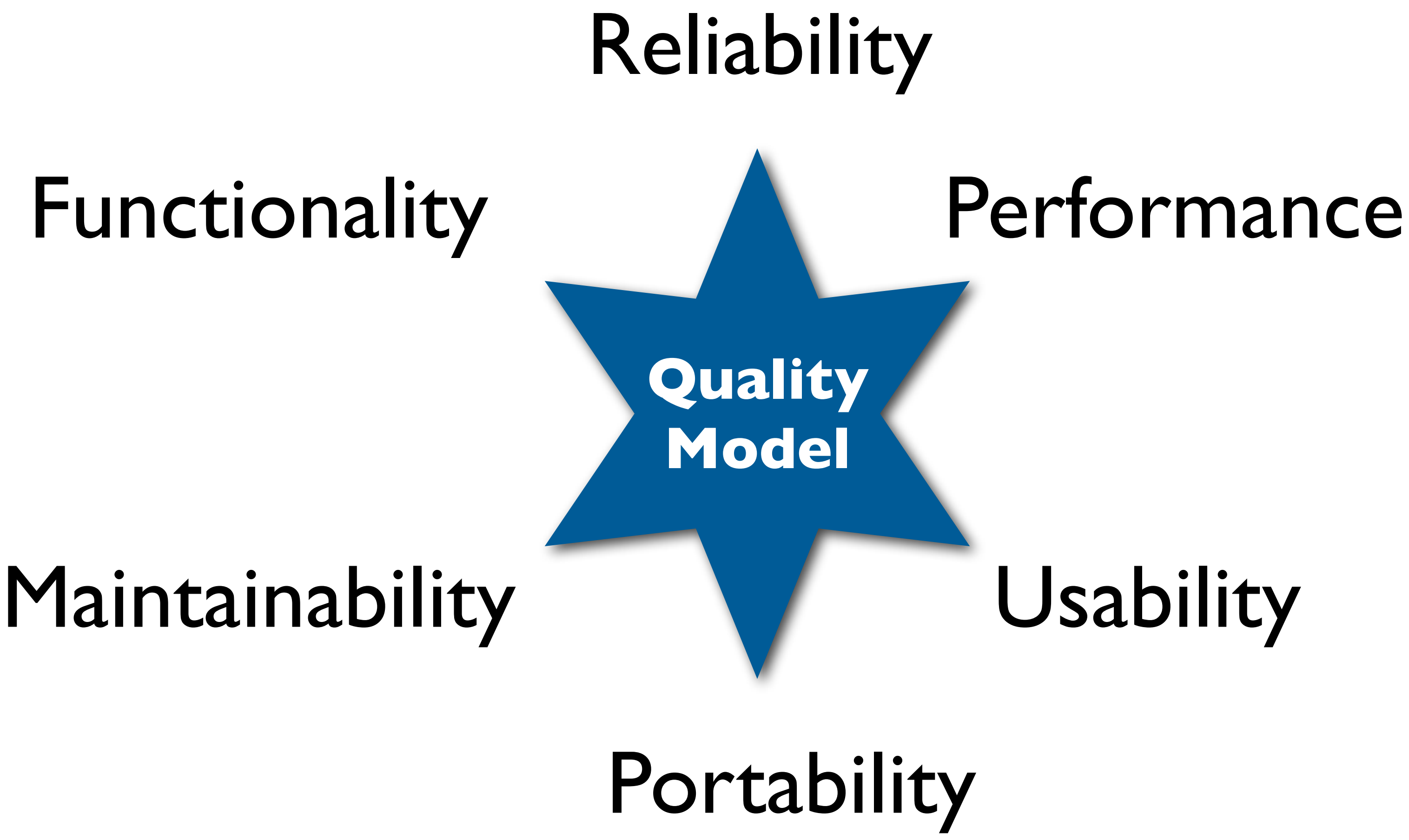
Quality models
for reuse

SAP Product
Standards

Common Criteria
BSI-Grundschutz

ISO 9126/25010

Therefore, already starting 1970, the first quality models for software appeared.
The should describe knowledge about software quality in a general way.
Early examples are from Boehm et al. and McCall.
This quality knowledge should have been reused for specifying requirements.
These kinds of models resulted in the ISO 9126 and finally in the standard ISO 25010 that is developed at present.
There are more specific standards, such as the Common Criteria and BSI-Grundschutz for security.
There are also company-specific models, such as the SAP Produkt Standards.



Reliability

Functionality

Performance

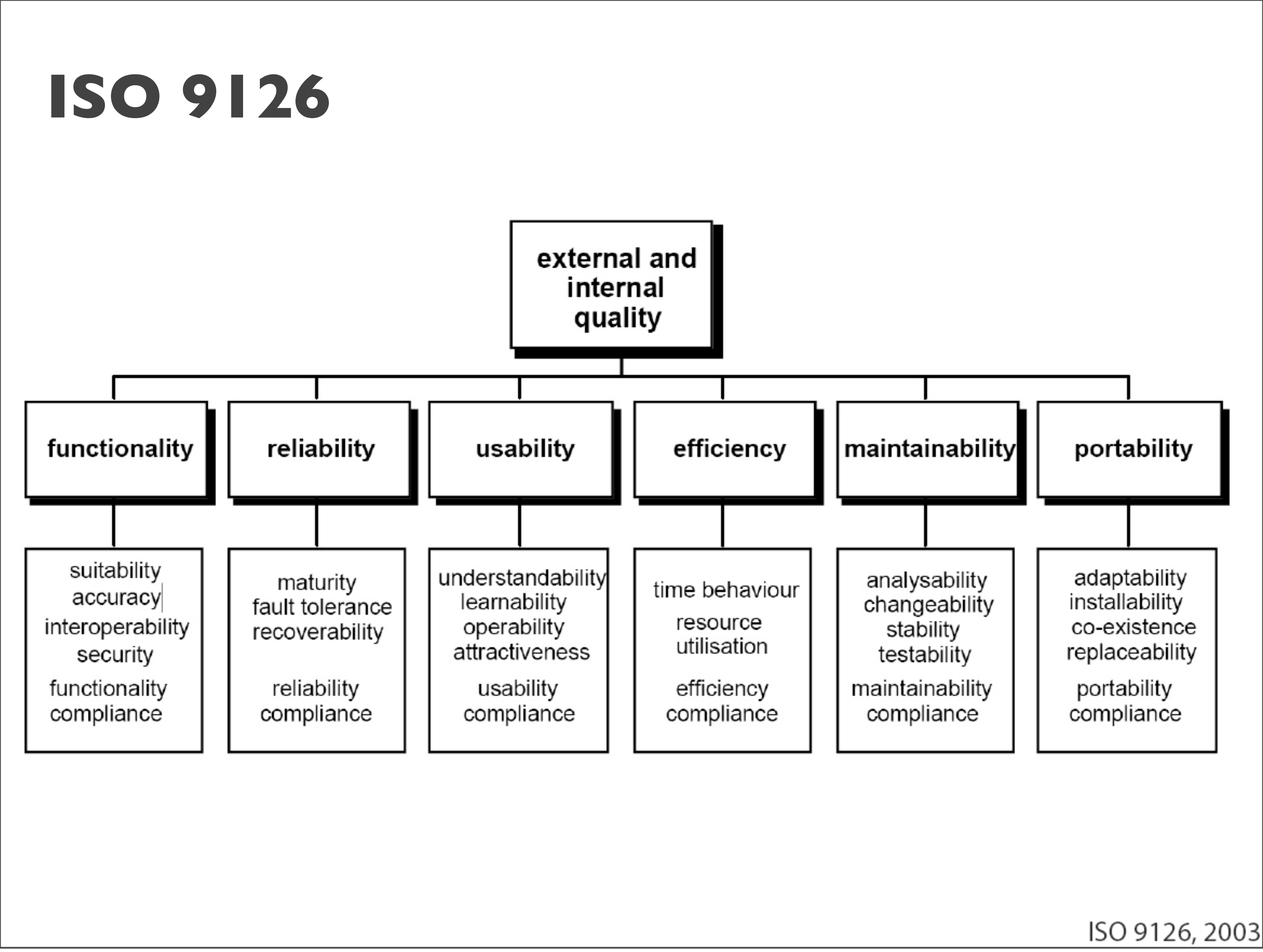
Quality Model

Maintainability

Usability

Portability

- Quality model: abstract definition of the important attributes for quality
- Basis for the definition of quality requirements
- Structured quality assessments
- Typically adapted to organisation, project, domain, …
- Standard: ISO 9126



ISO 9126

external and internal quality

functionality

suitability
accuracy
interoperability
security
functionality compliance

reliability

maturity
fault tolerance
recoverability
reliability compliance

usability

understandability
learnability
operability
attractiveness
usability compliance

efficiency

time behaviour
resource utilisation
efficiency compliance

maintainability

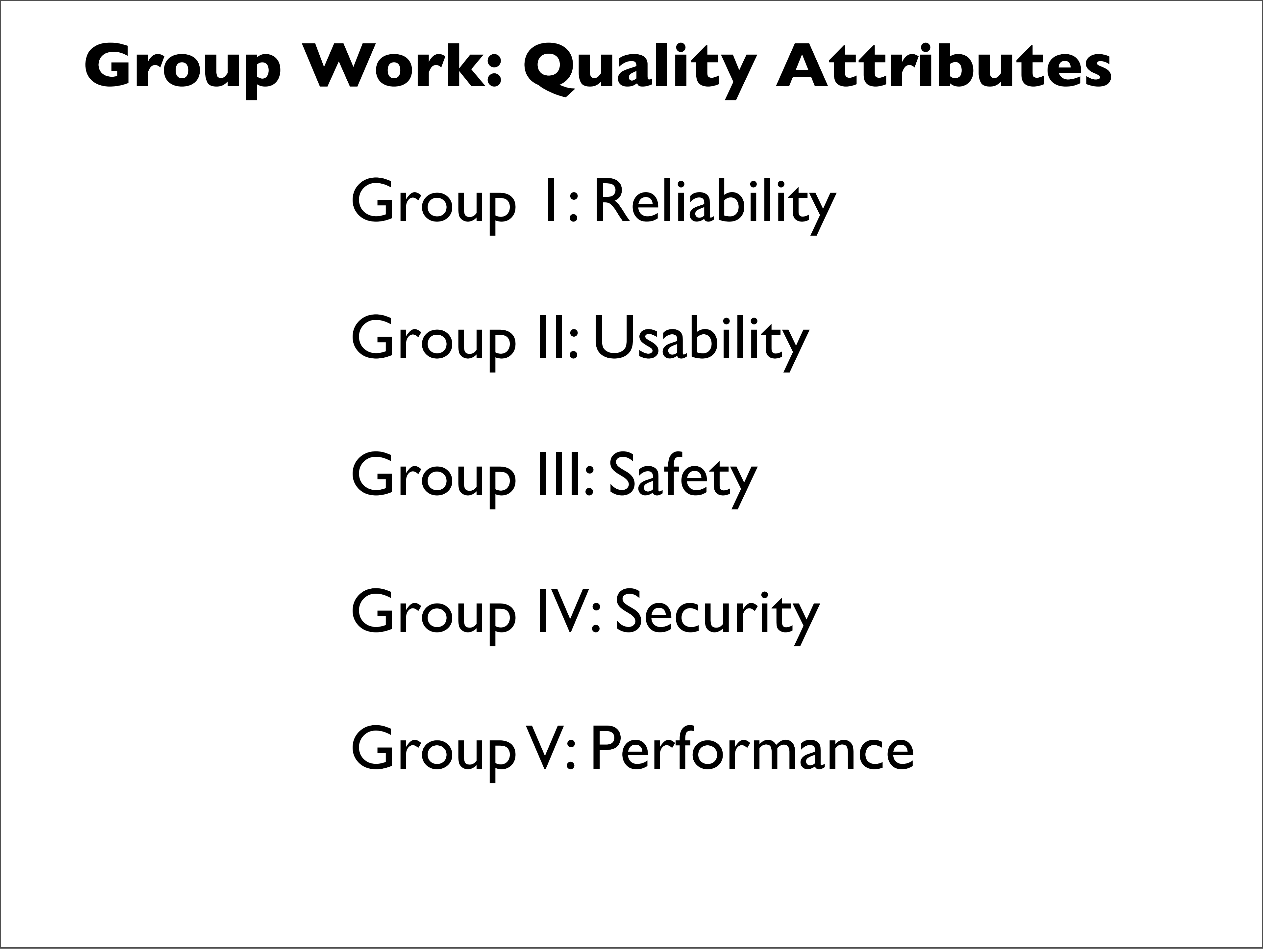
analysability
changeability
stability
testability
maintainability compliance

portability

adaptability
installability
co-existence
replaceability
portability compliance

ISO 9126, 2003

- The current standard for "external and internal quality".
- The standard also contains "quality in use" that is considered with safety, effectiveness, efficiency, and satisfaction of the usage of the system.
- The successor 25010 keeps this in principle
  - "external and internal quality" -> product quality
  - Some reorderings, e.g., security is a high-level attribute



Group Work: Quality Attributes

Group I: Reliability

Group II: Usability

Group III: Safety

Group IV: Security

Group V: Performance

-> Write on whiteboard:
1. Find a definition/description!
2. Write down two examples of problems/defects w.r.t. this attribute!
3. How can you avoid or find the problems/defects?

Ariane 5

- First flight in June 1996
- 40 Sec. after take-off left course because of a software defect
- Detonation
- Reuse of software of Ariane 4 with not enough tests
- Range Checking was switched off during runtime because of processor limitations
- The redundant, second system failed because of the same defect

Reliability

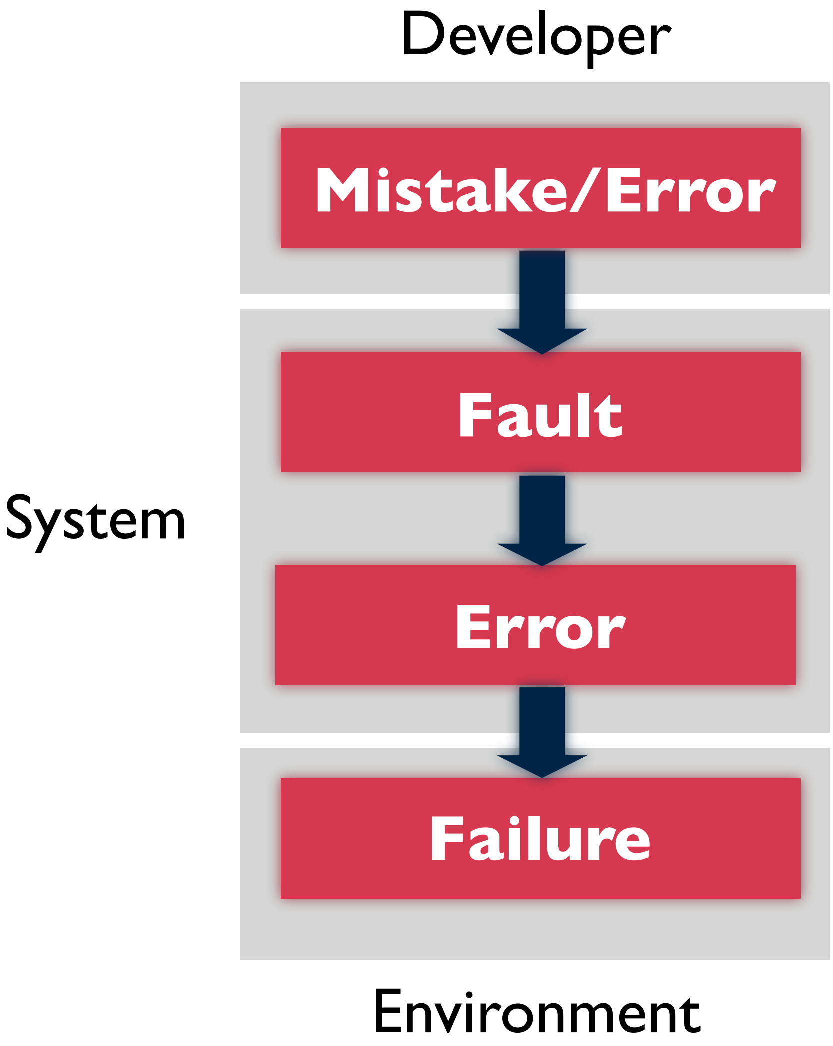
John D. Musa

Reliability is the probability of failure-free operation of a software for a specific time period in a specific environment.

J.D. Musa. Software Reliablity Engineering. AuthorHouse, 2004.

- Frequency of failure occurences
- Often synonym for quality
- Reciprocal is availability
- Methods
  - Reliability models for estimation and prediction
  - Fault tolerance at run–time
  - But: simple software redundancy is useless (compare Ariane 5)!

Defect terminology

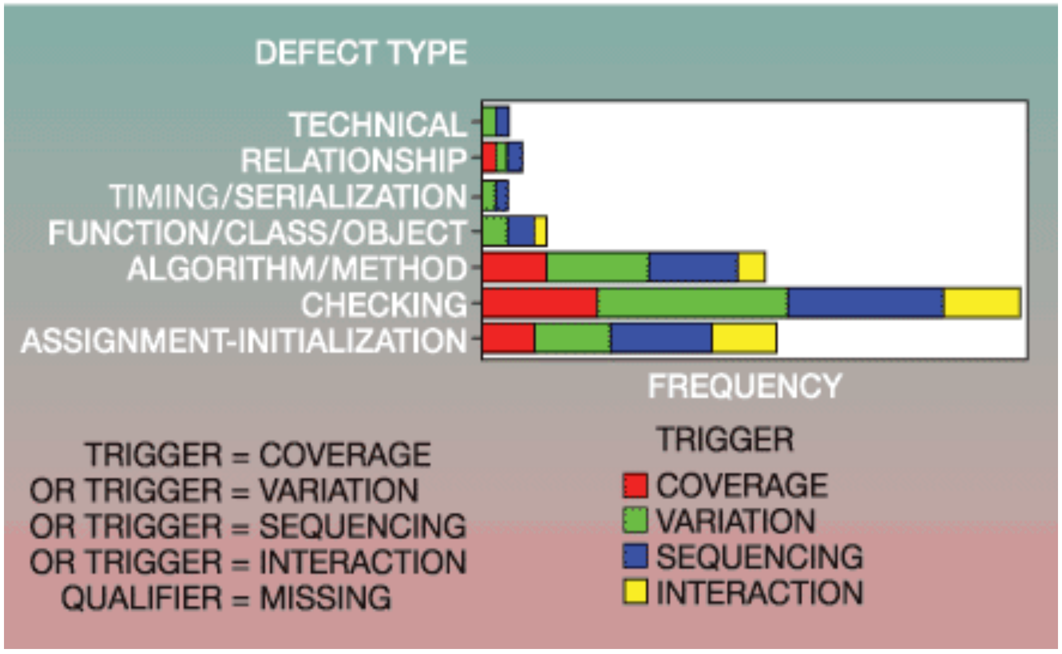


- Failure: The user notices that the system stopped its service
- Fault, bug: The cause of of a failure (usually in the code)
- Defect: Often a generic term for failure and fault
- Mistake, Error: The act of a person that causes a fault.
- Error: The state after a fault was executed but not necessarily a failure occurred (this is where fault tolerance avoids failures)

Defect types

Orthogonal Defect Classificiation (ODC)

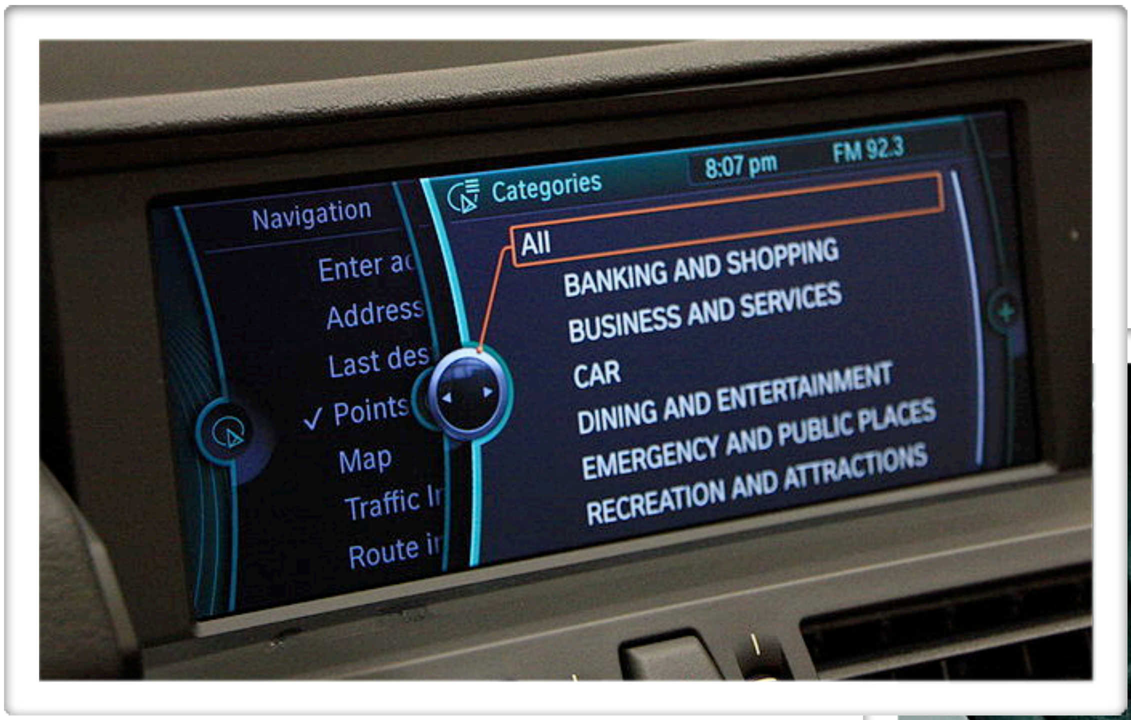
- Defect Removal Activity
- Trigger
- Impact
- Defect Type
- Qualifier

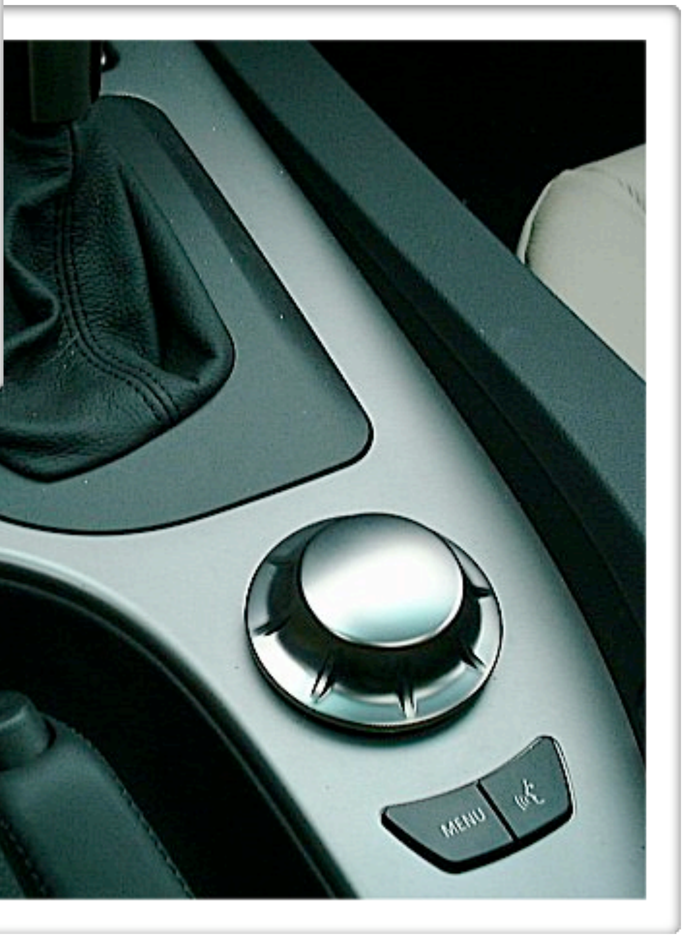


M. Butcher, H. Munro, and T. Kratschmer. Improving software testing via ODC: Three case studies. IBM Systems Journal, 2002.

- Classification of similar defects in classes or types
- Defect profiles of software and quality assurance/defect-detection techniques
- Common method: Orthogonal Defect Classification (ODC) of IBM
  - Defect Removal Activity (Design Review, Unit Test, ...)
  - Trigger (Simple Path, Workload/Stress, ...)
  - Impact (Reliability, Performance, ...)
  - Defect Type (Assign/Init, Checking, Alg/Method, Func/Class/Object, Timing/Serial, Interface/OO-Messages, Relationship)
  - Qualifier (Missing, Incorrect, Extraneous)

BMW iDrive





- 2002 new usage concept in BMW 7 Series
- Usage via controller know and display
- Bad acceptance in public: [J. G. Cobb. Menus Behaving Badly. NY Times, 2002.]
  - „In the 745i, tuning the radio is an interactive experience at 75 m.p.h.“
  - „IDrive is capable of managing more than 700 functions, but I can't imagine more than a few dozen things I'd want a car to do.“

Usability

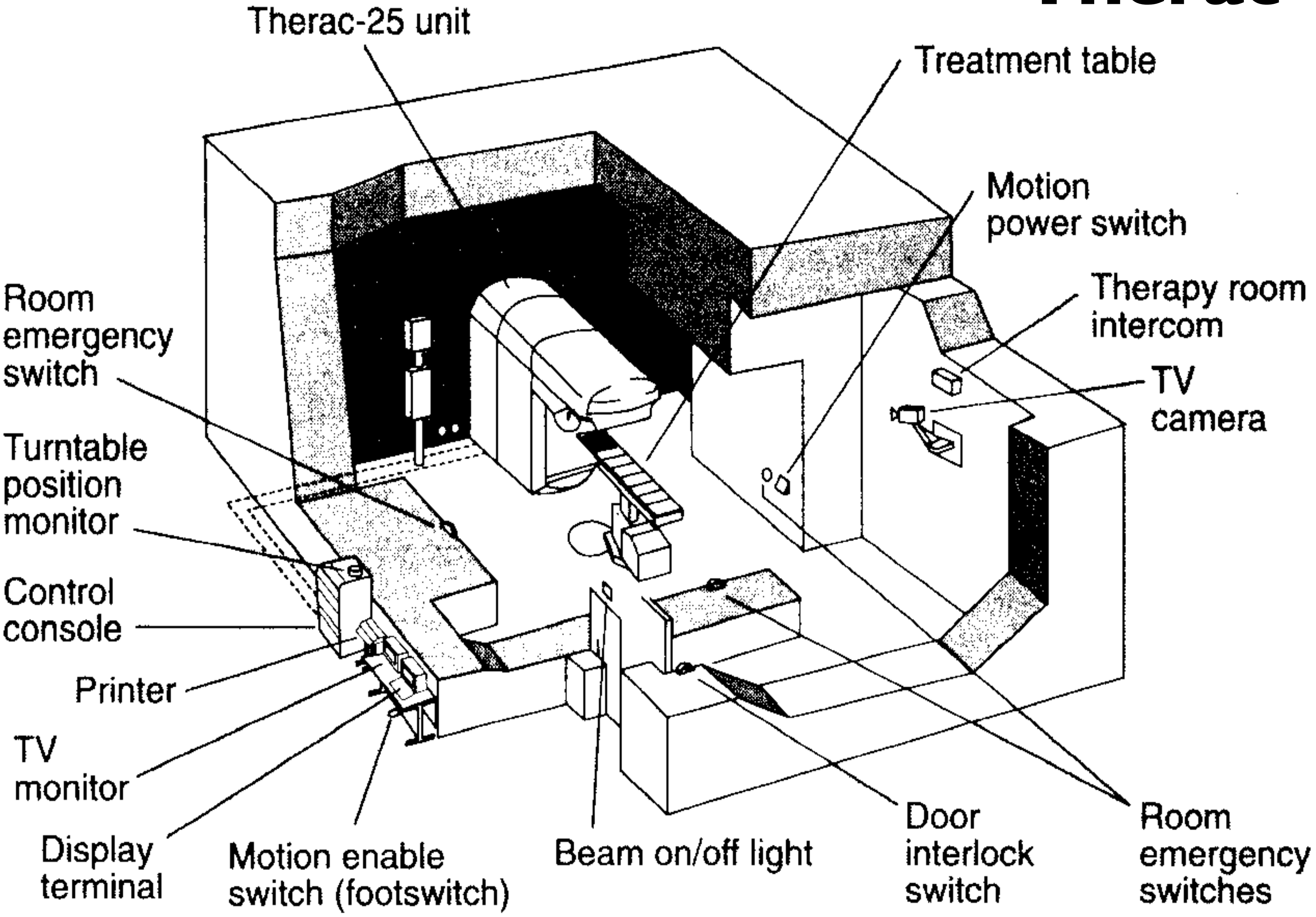
Usability is

- Effectiveness in solving a problem with the system
- Efficiency in using the system
- Satisfaction of the user of the system

ISO 9241-11

- Basics of interaction design
  - Suitable to needed tasks
  - Self-descriptiveness
  - Controllability
  - Conformity to expectations
  - Fault tolerance
  - Individual for different users
  - Easy to learn

Therac-25



N. G. Leveson, C. S. Turner. An Investigation of the Therac-25 Accidents. Computer, 1993.

- Medical device to destroy tumours by radiation
- X-Ray or electrons
- Several accidents with fatal consequences and severe damages to humans
- Wrong usage led to accidental usage of electrons instead of X-rays
- Typing fault in a control component

Safety

A system is safe
if it cannot reach failure states
that are harmful to its environment
(especially humans).

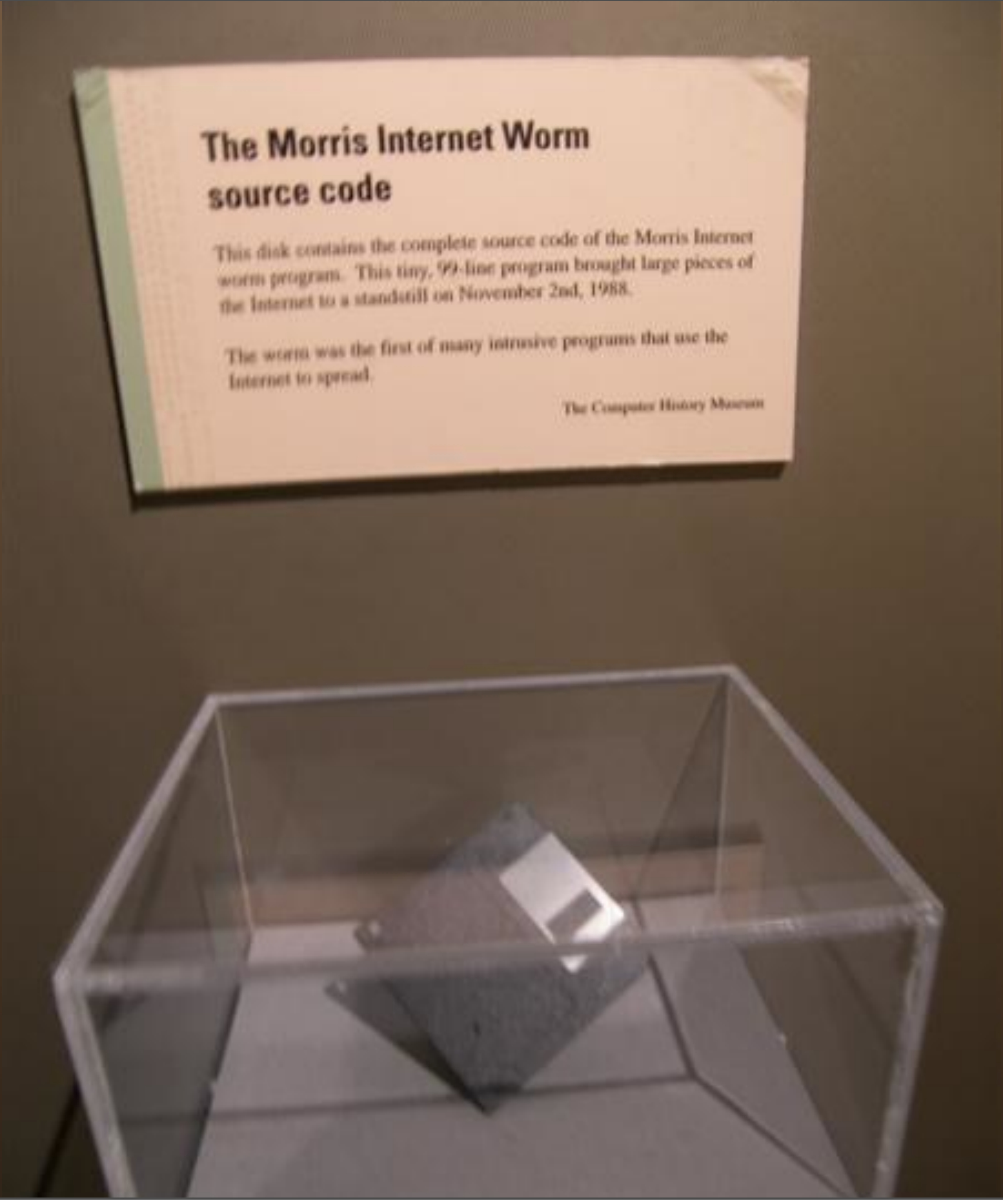
John C. Knight

Nancy Leveson

N. Leveson. Safeware. Addison-Wesley, 1995.

J. Knight, N. Leveson. An Experimental Evaluation of the Assumption of Independence in Multi-Version Programming. TSE, 1986.

- Analysis methods
  - Fault tree analysis
  - FME[C]A (Failure Modes and Effects [Criticality] Analysis)
- At present: „safety cases“ as a comprehensive and systematic means to consider all relevant evidences in a safety argument.
- Constructive QA
  - Failsafe
  - Watchdogs
- Software redundancy not effective, because different teams tend to introduce similar faults.



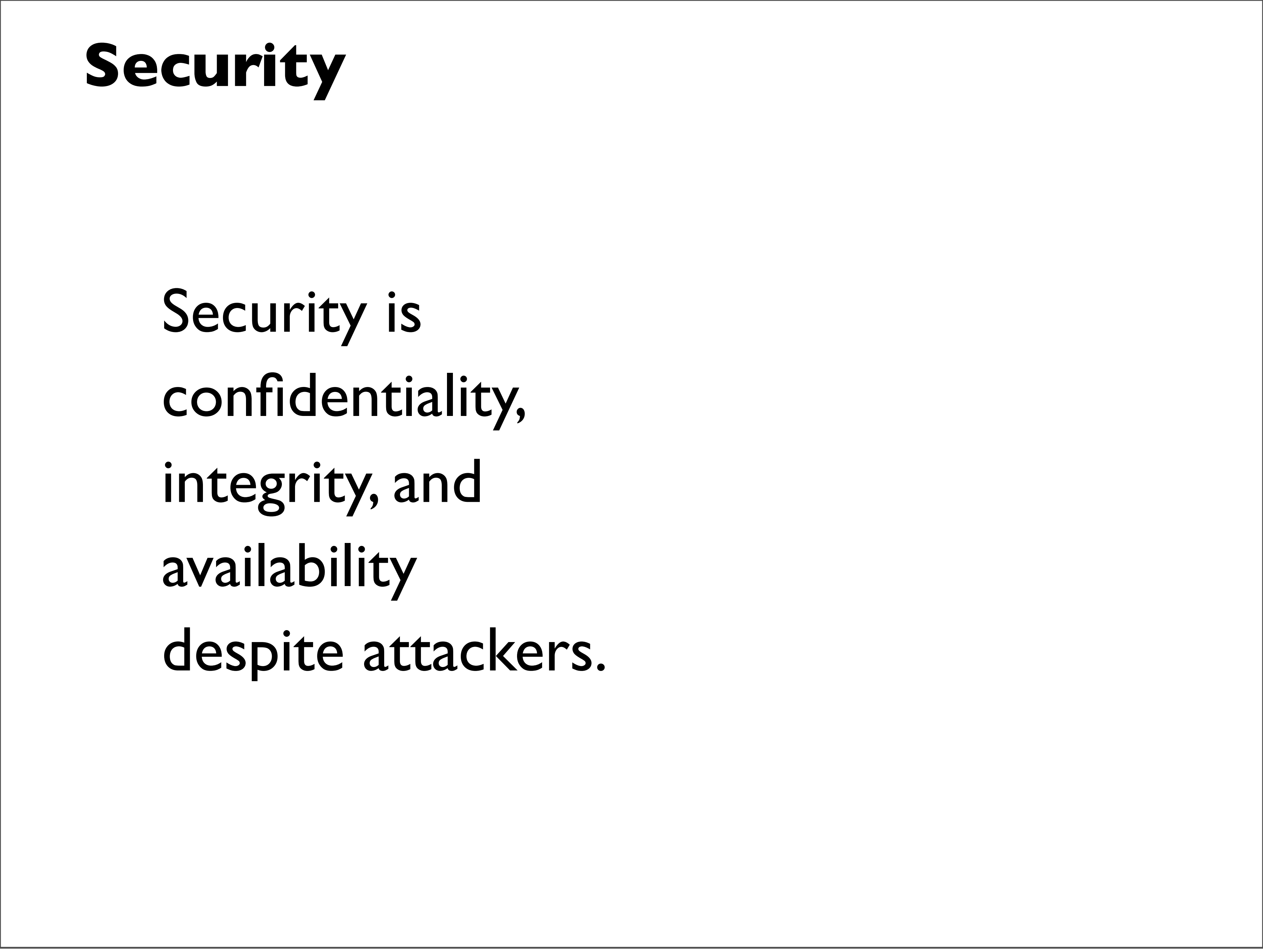
The Morris Internet Worm
source code

This disk contains the complete source code of the Morris Internet worm program. This tiny, 99-line program brought large pieces of the Internet to a standstill on November 2nd, 1988.

The worm was the first of many intrusive programs that use the Internet to spread.

The Computer History Museum

- 3rd November 1988: The black Thursday of the Internets
- System administrators of servers in the Internet noticed a high load on their computers.
- New processes were more quickly started then they could be deleted.
- They hat a worm on their systems.
- A worm is a program that is able to autonomously distribute over a network and thereby uses the resources of on computer to break into other computers.
- Worms use common security holes or vulnerabilities.
- Probably thousands of systems affected.
- Common vulnerabilities
  - Execution of introduced code (finger, sendmail)
  - Simple passwords for remote shells

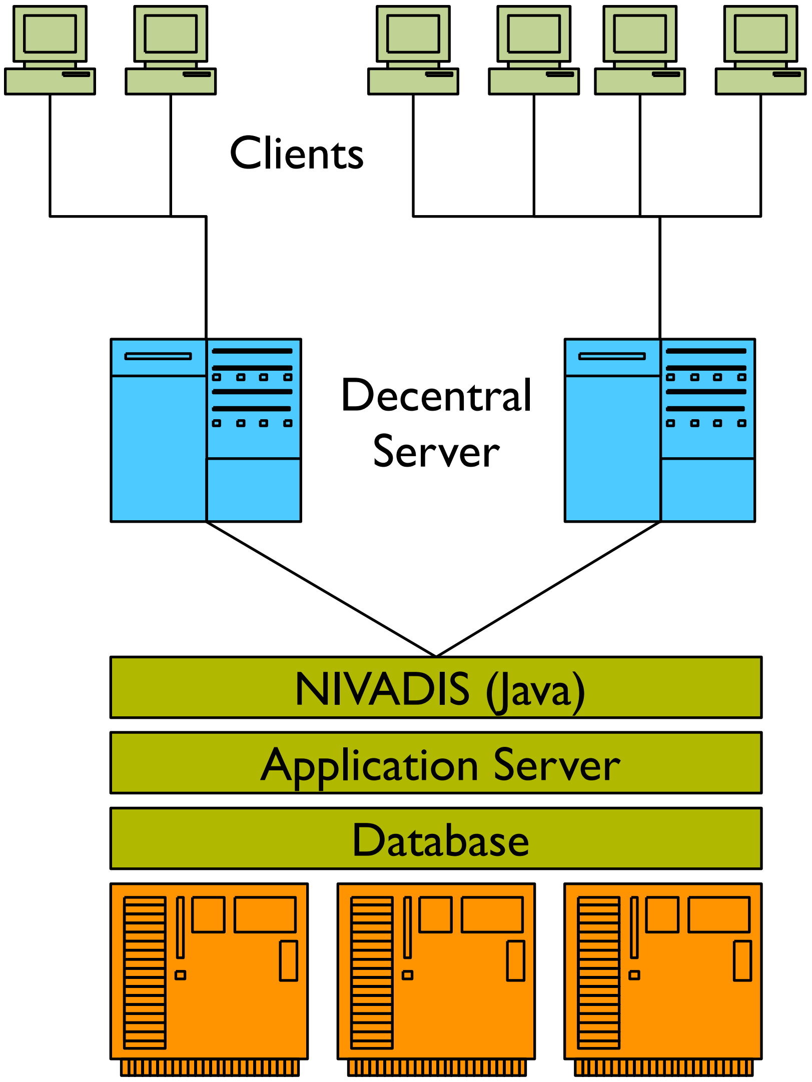


Security

Security is
confidentiality,
integrity, and
availability
despite attackers.

- In contrast to safety not „can the system harm someone?“, but „can the system be harmed by someone?“
- Analytical Qa
  - Security Reviews
  - Audits
  - Static analyses for overflows etc.
  - Formal verification of protocols
- Constructive QA
  - Cryptography
  - Rights management (authentication)
  - Network security (firewalls, secure channels)

NIVADIS



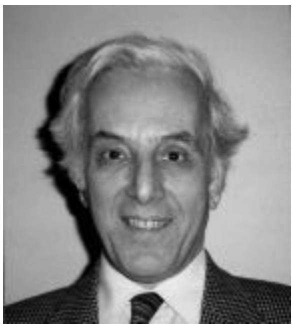
[http://www.computerwoche.de/index.cfm?pid=2440&pk=1059669]

- Modern data system of the police in lower saxony
- After changing from Bea Weblogic version 6.1 to 8.1 there were blockages during high load and normal load
- Request from clients were not delivered to the central application cluster
- Causes
  - Hardware Upgrade
  - Parameter configuration after the upgrade of the application server

Performance

the degree to which
a system or component
accomplishes its designated functions
within given constraints, such as
speed, accuracy, or memory usage

Jackson‘s Rules of Optimisation:
Rule 1: Don‘t do it.
Rule 2 (for experts only): Don‘t do it yet – that is, not until you have a perfectly clear and unoptimised solution.

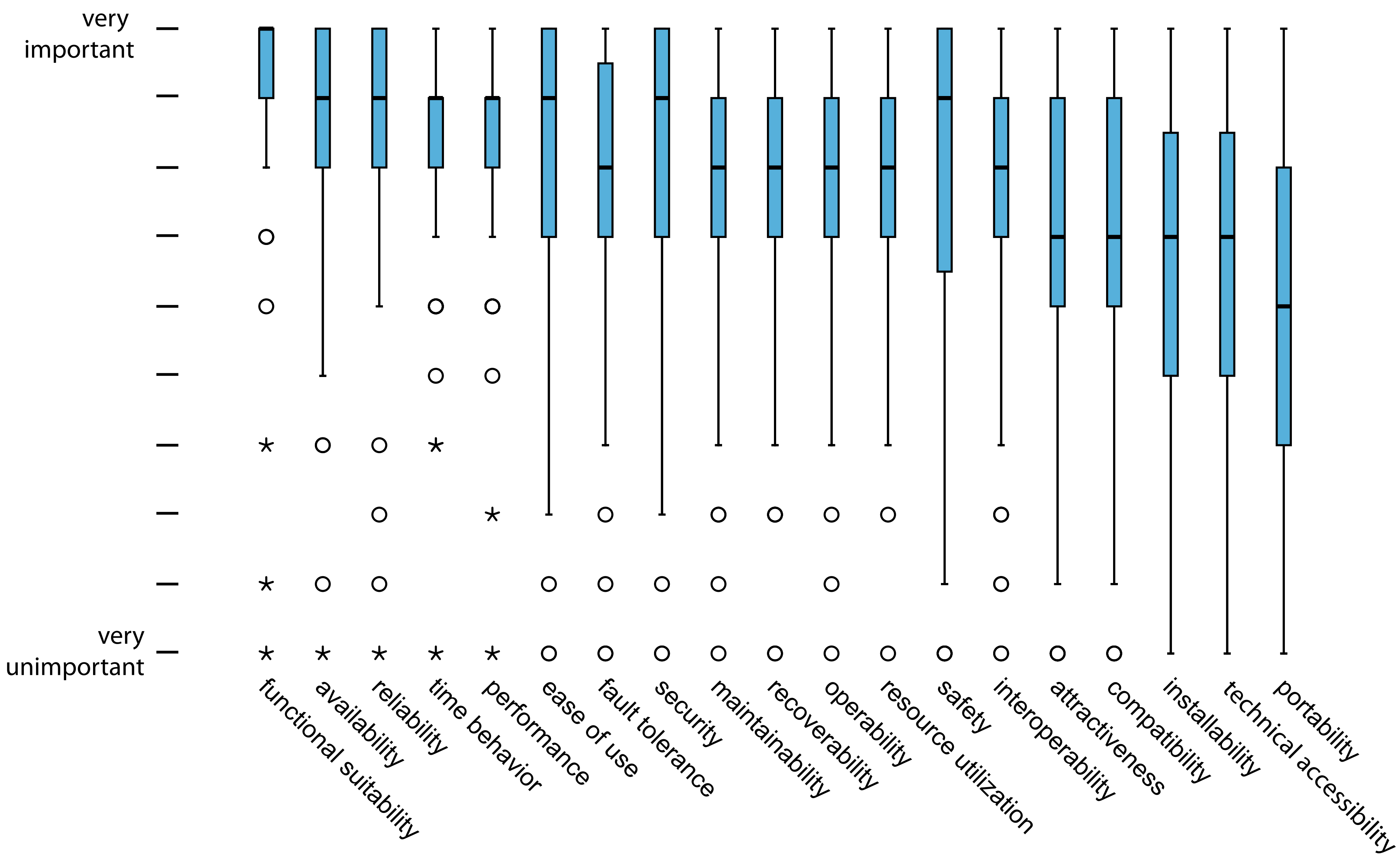


Michael Jackson

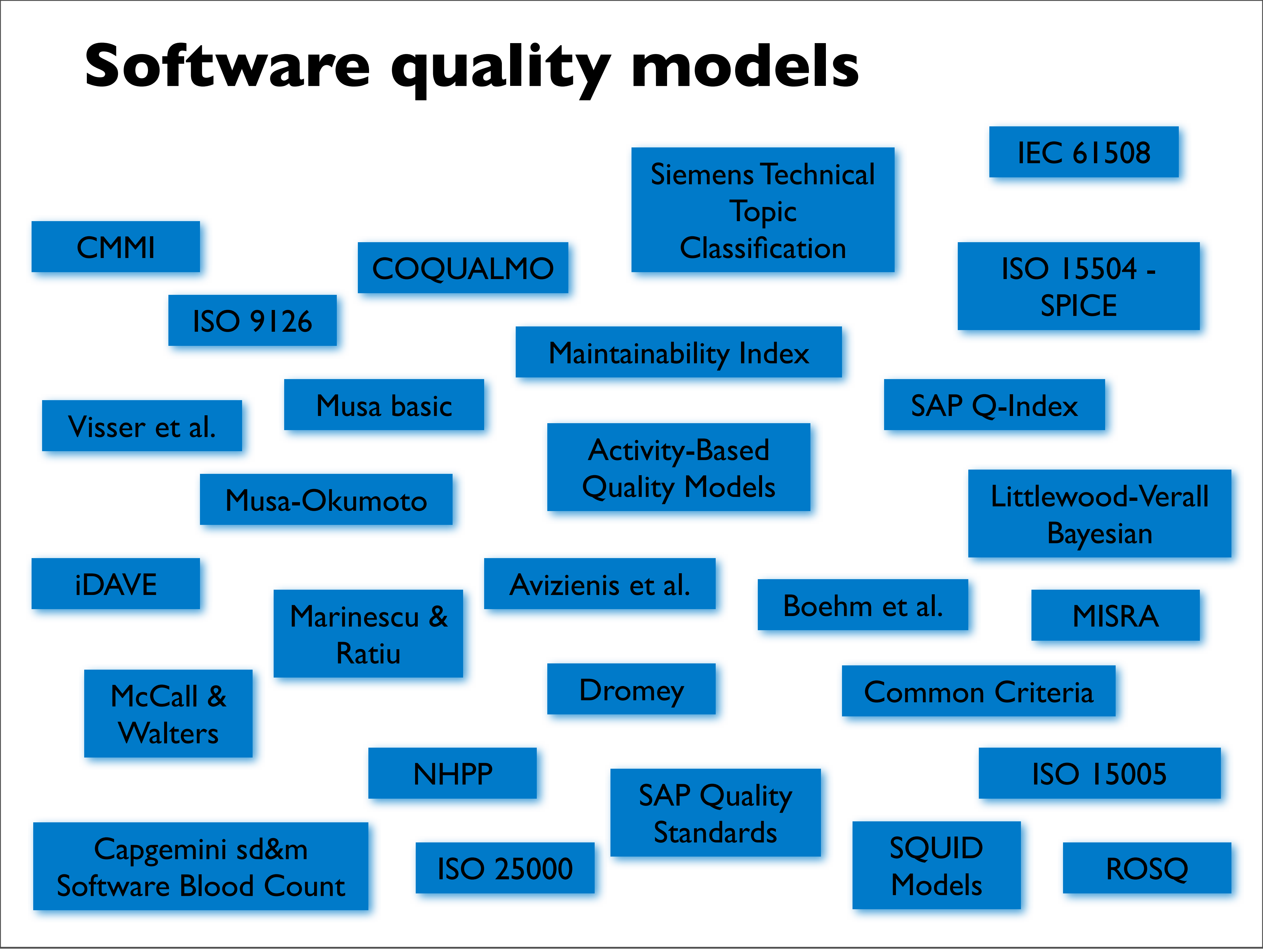
ISO/IEC 24765

- Often in contrast to other quality attributes
- QA methods
    - Models that include timing
    - Architecture (caching, …)
    - Performance and load tests

Importance of quality attributes



How important are quality attributes in general?
There is no clear answer to this question.
Large study from 2009: How important are these quality attributes in your products?
Some trend: functionality tends to be most important, portability least
There is, however, a high variation
Therefore, the importance quality attributes depends on other factors! Compare with quality profile!



Software quality models

IEC 61508

Siemens Technical Topic Classification

CMMI

COQUALMO

ISO 15504 - SPICE

ISO 9126

Maintainability Index

Visser et al.

Musa basic

SAP Q-Index

Activity-Based Quality Models

Musa-Okumoto

Littlewood-Verall Bayesian

iDAVE

Avizienis et al.

Marinescu & Ratiu

Boehm et al.

MISRA

McCall & Walters

Dromey

Common Criteria

NHPP

ISO 15005

SAP Quality Standards

Capgemini sd&m Software Blood Count

ISO 25000

SQUID Models

ROSQ

There is a huge amount of quality models in research as well as practice. They have very different goals and cover different aspects.

Notes:

ISO 15005: Road vehicles – Ergonomic aspects of transport information and control systems – Dialogue management principles and compliance procedures

NHPP: Non-homogeneous Poisson process (reliability growth models)

Tutorial: Tuesday next week
Lecture: Friday next week

No lecture on 7th May!