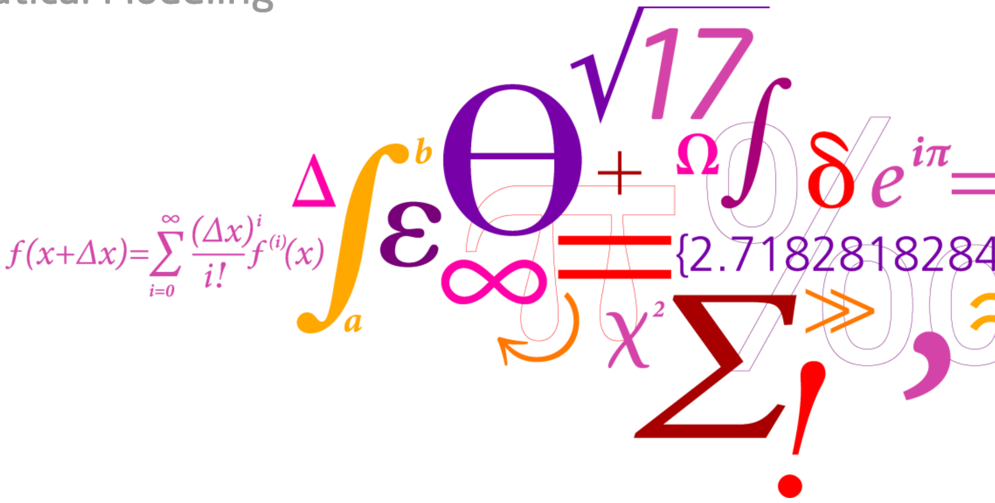
Software Engineering 2

A practical course in software engineering

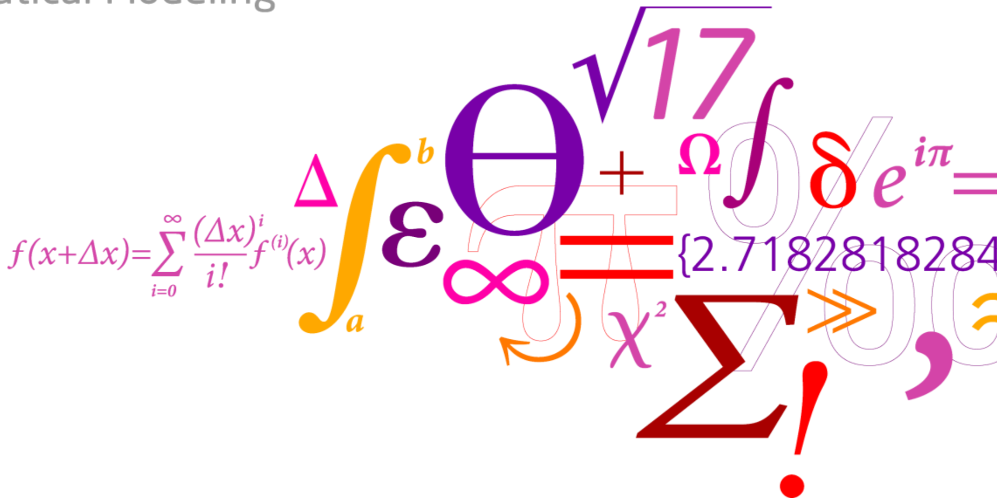
Ekkart Kindler

DTU Informatics
Department of Informatics and Mathematical Modeling

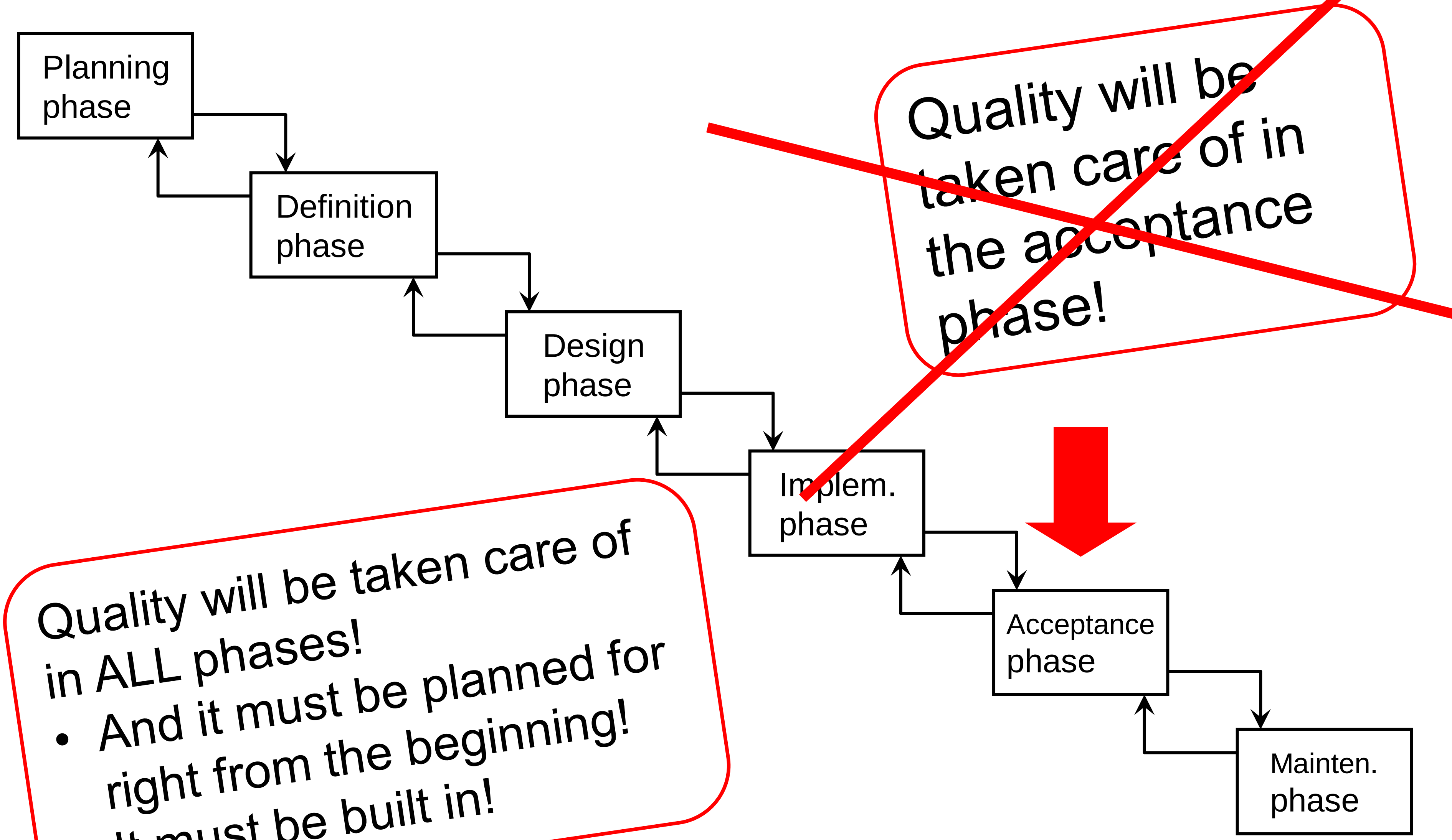


Quality Management

DTU Informatics
Department of Informatics and Mathematical Modeling


$f(x+\Delta x)=\sum_{i=0}^{\infty}\frac{(\Delta x)^i}{i!}f^{(i)}(x)$
$\sqrt{17}$
$e^{i\pi}$
{2.7182818284

Main Message



~~Quality will be taken care of in the acceptance phase!~~

Quality will be taken care of in ALL phases!

- And it must be planned for right from the beginning!
- It must be built in!

1. Overview

Questions

- What is quality?
- How can we measure it?
- How can we “produce” it?

Quality

The totality of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs
(ISO 8402, ISO 9126)

This also applies to documents of the development process; their quality must be assessed too. In this sense, documents are considered to be „products“ too.

Quality Characteristics (ISO 9126)

- A set of attributes of a product by which its quality is described and evaluated
- A quality characteristic may be refined into sub-characteristics (over several levels)

Quality characteristics (ISO 9126)

- Functionality
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

Sub-characteristics

- Functionality
  - Suitability
  - Accuracy
  - Interoperability
  - Compliance
  - Security

Sub-characteristics

- Reliability
  - Maturity
  - Fault-tolerance
  - Recoverability

Sub-characteristics

- Usability
  - Understandability
  - Learnability
  - Operability

Sub-characteristics

- …

Features & Metrics

- **Feature**: property for assessing the quality (sub-)characteristics of a product

  Comments, structure of the code, number of methods or classes, …

- **Quality metric**: a quantitative scale and method which can be used to determine the value a feature takes for a specific software product

→ Software metrics

Quality requirement

- Defines, for every feature, the minimum quality metrics to be reached (quality level)

Quality Assurance (QA)

- All actions to provide confidence (assure) that the product meets the quality requirements

Example: Action

- Tests are an action for QA

  Problem:

  - Tests can assert quality
    (if it is there)
  - But test do not “generate quality”
    (they can only sort out products with bad quality)

Quality Management (QM)

- is much more than just quality assurance!
- Quality management comprises all measures and actions to “generate” and “assure” quality

Quality needs to be **planned**, **controlled** and **assured**!

see slide 3

Quality Planning

Which product (part) needs to be checked

- when
- by whom, and
- with respect to which quality requirements!

2. Principles

- Product centred QM:

  Quality will be assured directly at the product (→ QA).

- Process centred QM:

  “Quality of the process” assures that the produced product has the required quality (→ ISO 900x, CMM, …)

Experience shows:

Purely product oriented QM turned out to be impractical for software development!

- Constructive measures

  during the development take care that, at the end, quality requirements are met

- Analytical measures

  check, at the end, whether the product meets the quality requirements

Constructive Measures

- **Notations**
- **Methods**
- **Tools**
- …

→MDA
→Code generation

Can enforce syntactical conventions!

- Predefine schemas or templates
- Conventions
- Standards
- Check lists
- …

Analytical Measures

- Testing procedures

  run the product (program) for checking the quality

- Analytical procedures

  asses the quality without executing it
  (in particular for documents)

Testing Procedures

- Dynamic test
- Simulation
- Symbolic tests
- …

Analytical procedures

- Program analysis (static analysis)
- Program verification
- Review (→ Section 3.1)
- …

The transition from testing to analysing procedures is continuous!

Ex.: Model checking, slicing, symbolic execution …

Principles

- Explicit definition of the quality requirements and quality planning
- Constructive Measures
- Early and continuous
- Independent
- Quantitative (metrics/measurable)

Early measures

- The earlier errors are found, the less follow-up costs it will cause

→ Errors should be detected as early as possible

Independent QA

- Nobody likes to invalidate his own product (“psychology of testing”)!
- If you forget a special case while programming, you are likely to forget to test exactly this case too!

→ QA actions should not be taken by the developer or programmer himself

3. Assessment Methods

- “Review”
  - Audit
  - Inspection
  - **Review**
  - Walkthrough
  - …

- Testing
  - Coverability
  - Unit tests
  - Integration tests
  - Acceptance tests
  - …

3.1 “Review“

- More or less formal process, with the goal to identify errors, inconsistencies, ambiguities (weaknesses in general) of a document
- To this end, the document is inspected and discussed in a **systematic** way (with the authors present)
- The result is a review report (recording the tracked problems) or the release of the document (possibly after several iterations)

Psychology of “Reviews”

Problem:

- Authors “are in the line of fire” or are “grilled”

- Psychological aspects need to be considered: e.g.
  - No superiors present
  - No evaluation of persons based on reviews
  - …

Inspection

- Very formal form of a „review“

Participants:

- Moderator
- Author
- Reviewer
- Recorder

Inspection

Procedure:

- Initial check
  (Moderator can refuse inspection)
- Planning
- Individual review
  (by reviewers)
- Inspection session
  (all participants; result: records)
- Revision
- Final check
- Release

iteration
(if necessary)

Review

- Simple form of a “review“

Participants:

- Moderator
- Authors
- Reviewer
- Recorder

Review

Procedure:

- Individual review
  (result: comments on document)
- Inspection session
  (result: record)
- Revision
- Final check

Iteration

Walkthrough

- Informal version of “reviews”

Participants:

- Author
- Reviewer

Walkthrough

Procedure:

- Maybe, individual review
- Inspections session
  (author moderates; result: record)

3.2 Testing

„Testing is the execution of a program in order to find (as many as possible) errors“

Test

- A test should find deviations between the actual and the expected behaviour of a program
- A **test** consist of a set of **input data** along with the **expected result**
- Running a **test** means **executing the program** with the input data and comparing the actual result with the expected result

Testing and QA

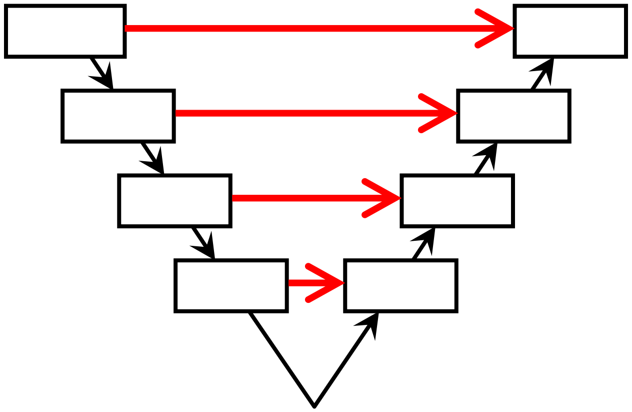
- Tests can **increase the confidence** in the **absence of errors**; this provides a measure for
  - Accuracy (functionality) and the
  - Fault-tolerance and maturity (reliability)

  of the software

Test Levels

There are different levels of test:

- Acceptance tests (by/with client)
- System tests
- Integration tests
- Unit tests



Automation

Regression tests should guarantee that the quality of the software does not “regress” over time.

- Test can be executed automatically (→ JUnits)
- Regression test will be executed automatically whenever some part of the system was changed (in order to ensure that the quality of the software does not regress).

  Then, a system will be released only when all regression test were passed successfully; this guarantees that supposed corrections did not inject new errors (→ JUnits)
- Concurrent (multi-threaded) programs and GUIs are still problematic

The Limitations of Testing

- Tests can only show the presence of errors
- Tests cannot guarantee the absence of errors

→ Absence of errors can be shown by verification; but, complete verification of software is impractical today (but sometimes required for parts)

Questions

- What are good test?
- How do I test properly?
- When did I test enough?
- How can I make sure to find as many errors as possible?

→ Systematic testing (see below)

→ Principles of testing (see below)

Principles of Testing

- Author does not (exclusively) test
  (Psychology of testing)
- Expected result should be defined before executing the test
  (Define tests early → XP/agile: before implementation)
- Rigorous testing
  - Check result carefully
    (at best automatically)
  - Check “everything”
    (→ Systematic testing)
  - “over and over again” testing
    (Regression tests: executed after every change for the complete system)

Systematic Testing

Systematic construction of tests

- **Black-box Test** from specification (without knowing the implementation):
  - Normal cases from specification
  - Special cases from specification
  - Illegal input from specification
- **Glass-box Test** from implementation:
  - Normal- and special cases from program conditions (alternatives and loops)
  - Coverage criteria (→ next slides)

Coverage

```
if (x == 0)
  z = x-y;
else
  x = x/y;

if (y > 0)
  z = 27;
else
  z = y/x;
```

Statement coverage:
Number of statement executed by at least one test divided by the number of all statements of the program.

100%: Every statement was executed at least once

Coverability

```
if (x == 0)
  z = x-y;
else
  x = x/y;

if (y > 0)
  z = 27;
else
  z = y/x;
```

100% Statement coverage:
Every statement was at least executed once

Necessary tests:

- x=0, y=0
- x=1, y=1

Coverage

100% Path coverage:
Every path of the program was executed at least once.

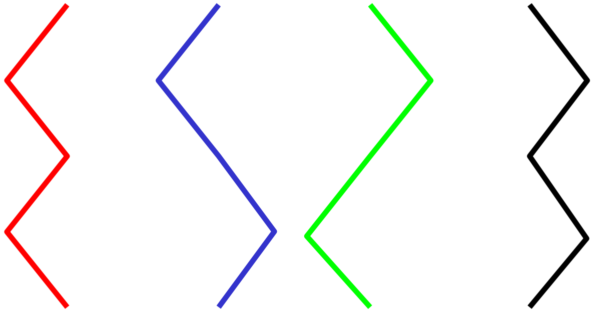
```
if (x == 0)
   z = x-y;
else
   x = x/y;

if (y > 0)
   z = 27;
else
   z = y/x;
```

Possible tests:

- x=0, y=1
- x=0, y=0
- x=1, y=1
- x=1, y=0

Problem: Loops give rise to infinitely many paths! We don't go into details here.



Construction vs. Counting

- For glass-box test, we can construct test in such a way that a specified coverage will be reached
- By instrumenting our software, we can also "count" which coverage is reached (if it is not high enough, we can add further tests to the test set; some test suits do this automatically)
- Both approaches can be combined

Warning

- Even if you have achieved 100% path coverage, this does not guarantee that all errors are found!!!!

- Nevertheless this is a quality metric for the sub-characteristics “Accuracy” of the product and increases the confidence in the quality of the product