Conclusions

Automation of Testing
Minimum Requirements

- Necessary, minimum requirements of testing
- A simple practical testing strategy
- Generation of a testing strategy in practice
- Summary
- Literature

© Prof. Dr. Liggesmeyer, 2

Necessary, Minimum Requirements of Dynamic Testing

- Absolutely necessary according to all authoritative standards:
  - Function-oriented test planning for all test phases
- Reproducibility of test results => automatic regression test after software modification
- Large consensus:
  - Supplementary structure-oriented coverage (minimum: branch coverage test)
  - In critical application areas – e.g., avionics – more thorough structure-oriented tests are explicitly required by the standards
  - Execution preferably during the first test phase after finishing the code (module test)
  - Additional performance and stress testing, especially in technical application areas

© Prof. Dr. Liggesmeyer, 3

A Simple, Practical, Dynamic Test Strategy

Module Test

- Function-oriented module test using a branch coverage testing tool
  - Function-oriented test case generation (e.g. generation of functional equivalence classes)
  - Preparation – viz. instrumentation – of the modules to be tested for controlling the branch coverage achieved
  - Complete execution of the function-oriented testing
  - Controlling of the branch coverage achieved in this way (according to experience, approx. 70% - 80%)
- Structure-oriented module test using the branch coverage testing tool
  - Cause analysis for the non-execution of branches
  - Generation of test cases for the branches not yet executed

Integration and System Test

- Function-oriented test

Measurement

- Software is nowadays often used in application areas in which quantitative statements are common or necessary:
  - Contract design: “We stipulate that the system’s minimum availability shall be 99.8%!“
  - Safety proof of a railway system at the Federal Railway Authority: “How high is the remaining risk posed by software failures?“
  - Is the expected number of the remaining failures sufficiently low for the release?
  - Is the probability adequately small that software failures in control units will cause malfunctions of our luxury sedans?
  - We need a failure-free mission time of 4 weeks. Can this be attained?
- Many enterprises have installed defined processes: The next step is to quantitatively control these.

Measurement

- First steps in measurement:
  - Measurement of the test coverage => integrated into dynamic test tools
  - Measurement of code features => separate measurement tools

Extension of Tests

- Using tools in support of …
  - … regression tests
  - … load and stress tests
  - … GUI tests

Statistic Analyses

- Prior to test execution:
  - Checking compliance with programming conventions => Tool
  - Data flow anomalies analysis => Tool / Compiler
  - Code Inspection / Review => WITHOUT Tool
- Dynamic test
  - As described
  - Beforehand: Switching on the assurances
  - At the same time:
    - Recording of the test cases (regression test) and
    - recording of the coverage
    - If applicable: load and stress tests / GUI test

In Addition

- If applicable: measurement of the achieved reliability (evaluation of the test observations) => Tool
- If applicable: monitoring of special requirements from the standards: e.g., RTCA DO 178 B demands advanced dynamic tests (avionics)
- If applicable: early tool-supported safety analysis (FMECA, RBD, FT)

Literature

/Liggesmeyer 02/: Liggesmeyer P., Software-Qualität, Heidelberg: Spektrum Akademischer Verlag 2002