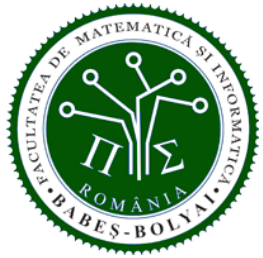


Software Systems Verification and Validation

Assoc. Prof. Andreea Vescan

Babeș-Bolyai University
Cluj-Napoca
2020-2021

Outline

- Verification and Validation
- Software development life cycle Model
  - V-Model
  - Extended V-Model [CB03]
- Quality
  - Quality control vs. Quality Assurance
  - Quality definitions
- What is a bug?
  - First bug
  - Terms for software failures
  - Software error (or bug)
  - When? Why? Cost?
  - Famous Software bugs

Verification and Validation (SEI and NASA) [NT05],[PY08]

Software Engineering Institute

- **Verification**
  - assures the product is developed according to requirements, specifications and standards.
  - building the product correctly.
  - Are we building the product right?
- **Validation**
  - assures that the product will be usable on the market.
  - building the correct product.
  - Are we building the right product?

NASA - Software Assurance Guidebook and Standard [NAS]

- Verification and Validation
  - the process that assures that the software product:
    - will satisfy the requirement (functional and others) = **validation**.
    - every step in the product development is resulting in a correct (sub)product = **verification**.

Verification and Validation - comparison

Verification

- evaluates if the product of a given development phase satisfies the requirements of that phase;
- reviews products to ensure their quality (consistency, completeness, correctness);
- static and dynamic analysis techniques.

Validation

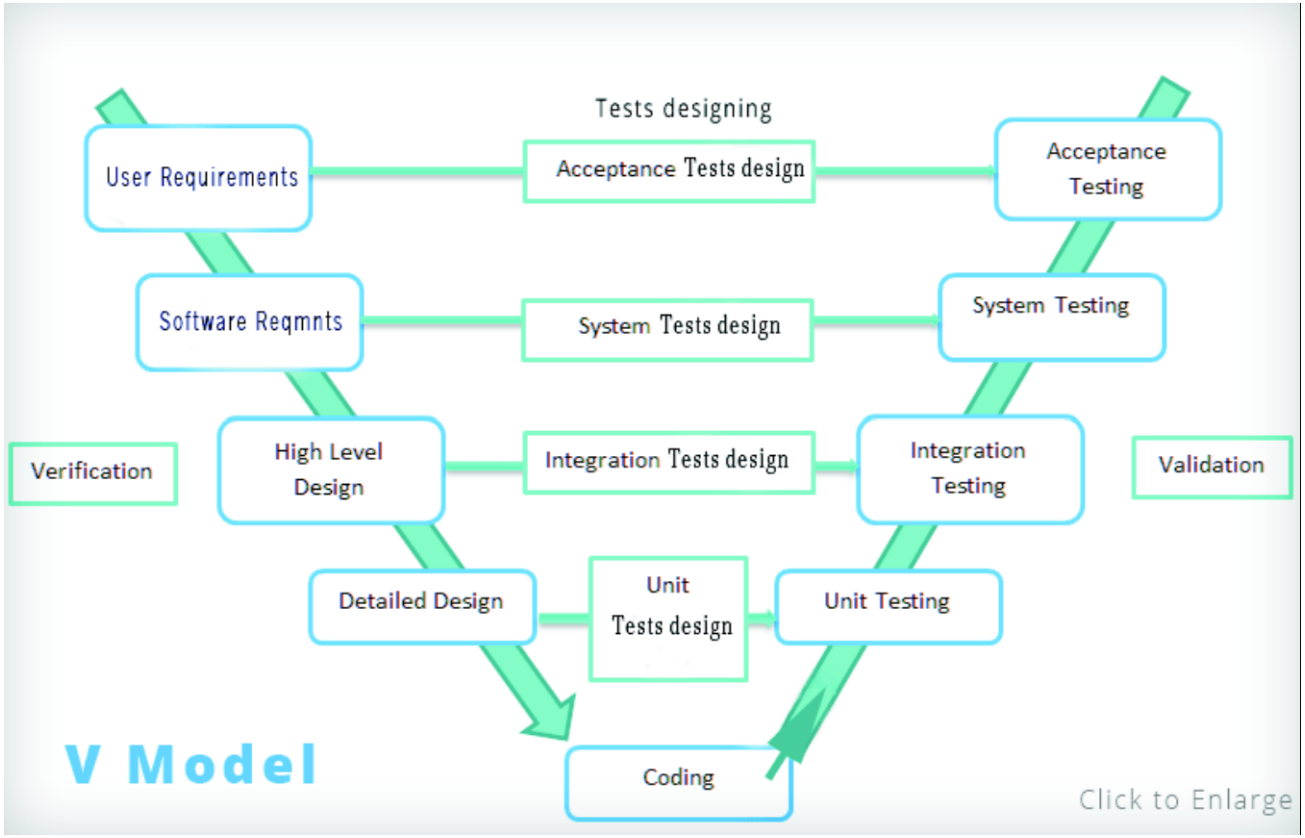
- helps us at confirming that a product meets its intended use.
- is performed toward the end of the system development to determine if the entire system meets the customer's needs and expectations;
- is performed on the entire system by actually running the system in its real environment and using a variety of tests.

Outline

- Verification and Validation
- Software development life cycle Model
  - V-Model
  - Extended V-Model [CB03]
- Quality
  - Quality control vs. Quality Assurance
  - Quality definitions
- What is a bug?
  - First bug
  - Terms for software failures
  - Software error (or bug)
  - When? Why? Cost?
  - Famous Software bugs

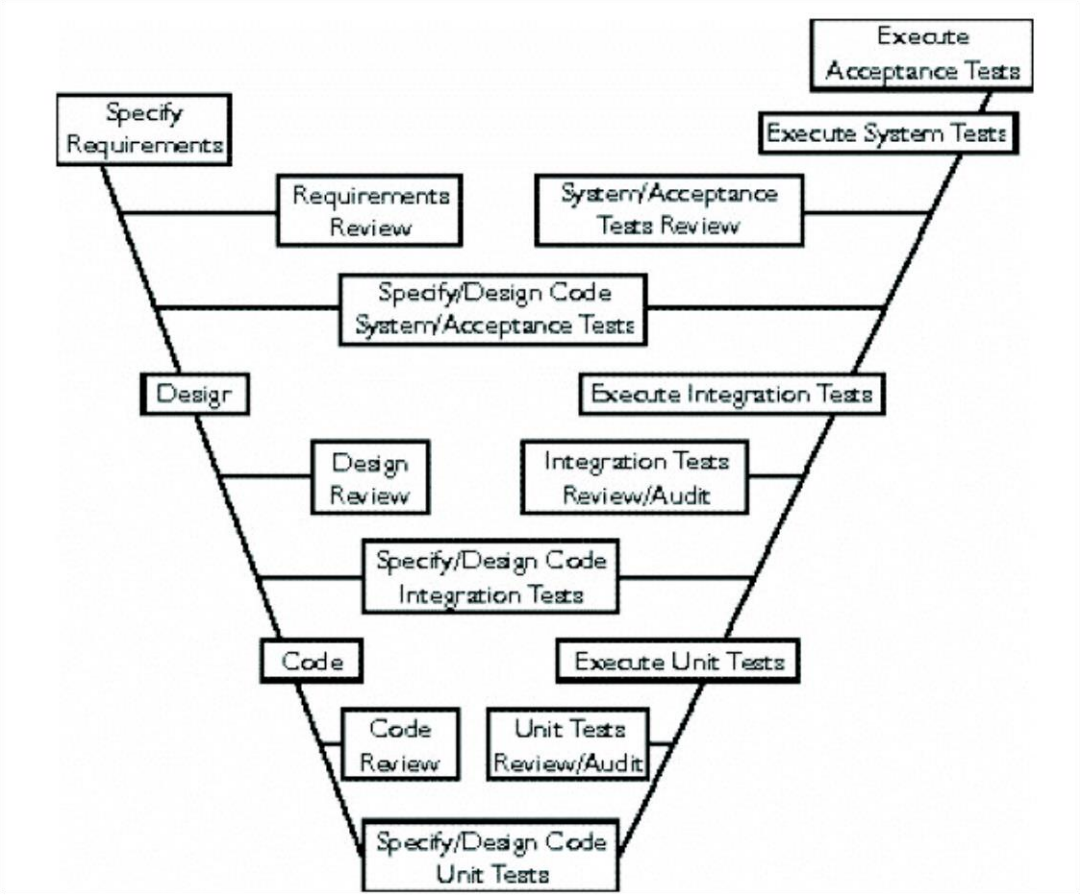
Software development life cycle Model

- V-Model



Software development life cycle Model

- Extended V-Model



Outline

- Verification and Validation
- Software development life cycle Model
  - V-Model
  - Extended V-Model [CB03]
- Quality
  - Quality control vs. Quality Assurance
  - Quality definitions
- What is a bug?
  - First bug
  - Terms for software failures
  - Software error (or bug)
  - When? Why? Cost?
  - Famous Software bugs

Quality control vs. Quality Assurance

- Quality control
  - QC = Quality of products
  - How do you control the quality of the work you have done?
  - Goal: detect problems in the work products

- Quality Assurance
  - QA = Quality of processes
  - How do you assure the quality of the work you are going to do?
  - Goal: Ensure adherence to processes, standards and plans

Quality definitions [BBST]

- “Quality is conformance”
  - ”Software quality: Conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software.” [Pre00]
- ”Quality is free.” by Phil Crosby [Cro80]
  - conformance with requirements, i.e. conformance to the user’s actual requirements which may or may not be written down in a specification.
  - Quality - conformance to the needs and not to documents.
- “Quality - fitness for use.” by Joseph Juran [JJ98]
  - satisfiers - anything that makes you like the application
  - dissatisfiers - anything that makes you like the application less.
  - Quality – according to who? (Program manager, Programmer, Tech writer, Tester)
- “Quality - is value to some person.” by Jerry Weinberg [Wei92]
  - quality is subjective - what’s valuable for you may not be so valuable for me.

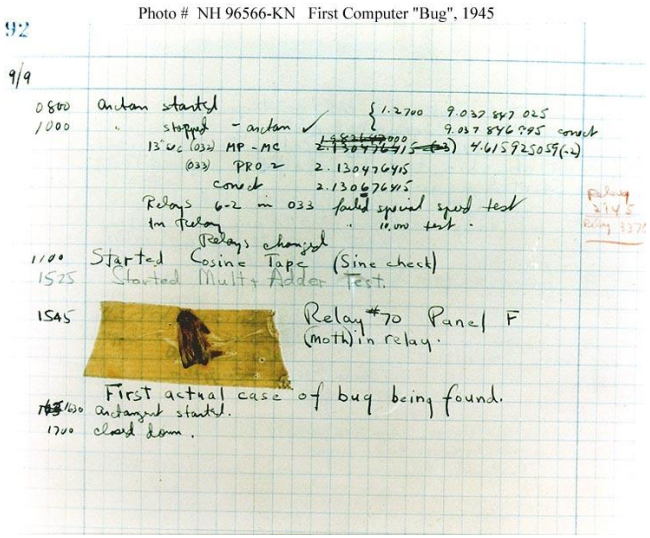


Outline

- Verification and Validation
- Software development life cycle Model
  - V-Model
  - Extended V-Model [CB03]
- Quality
  - Quality control vs. Quality Assurance
  - Quality definitions
- What is a bug?
  - First bug
  - Terms for software failures
  - Software error (or bug)
  - When? Why? Cost?
  - Famous Software bugs

What is a bug?

- First bug
  - Grace Hopper - About first software bug: (2’10’’)
    - https://www.youtube.com/watch?v=lQS0hDqpVLE
  - 1947 - Harvard University - Mark II



  - The first computer bug was born!
    - Well, okay, it died!

Terms for software failures [Pat05]

- **Failure**
  - A failure is said to occur whenever the external behavior of a system does not conform to that prescribed in the system specification.
- **Error**
  - An error is a state of the system. In the absence of any corrective action by the system, an error state could lead to a failure which would not be attributed to any event subsequent to the error.
- **Fault**
  - A fault is the adjudged cause of an error.
- Process of failure manifestation - represented as a behavior chain:

  fault → error → failure.

Software error (or bug)

- A bug is an aspect of the product that causes an unnecessary or unreasonable reduction in the quality of the product.
  - design weaknesses, documentation error, usability annoyances

Remark. Some aspects of a product do limit its quality but are not bugs!

- A bug is anything about the product that threatens its value.

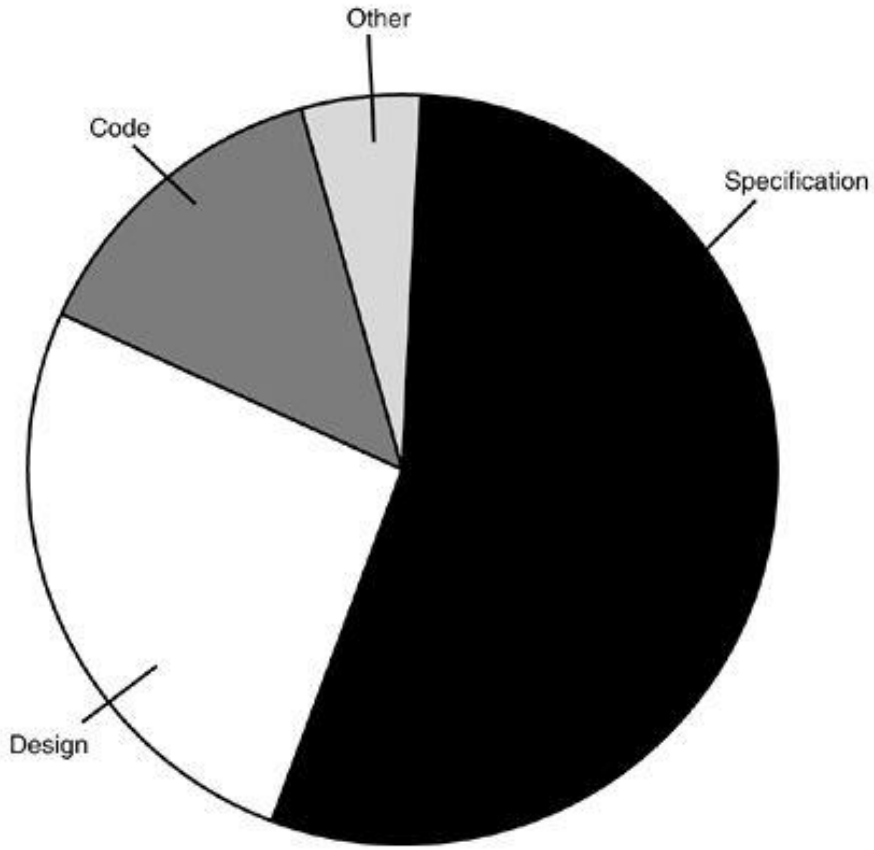
  [James Bach, Michal Bolton] [BBST]

  - In this course, all software problems will be called bugs.

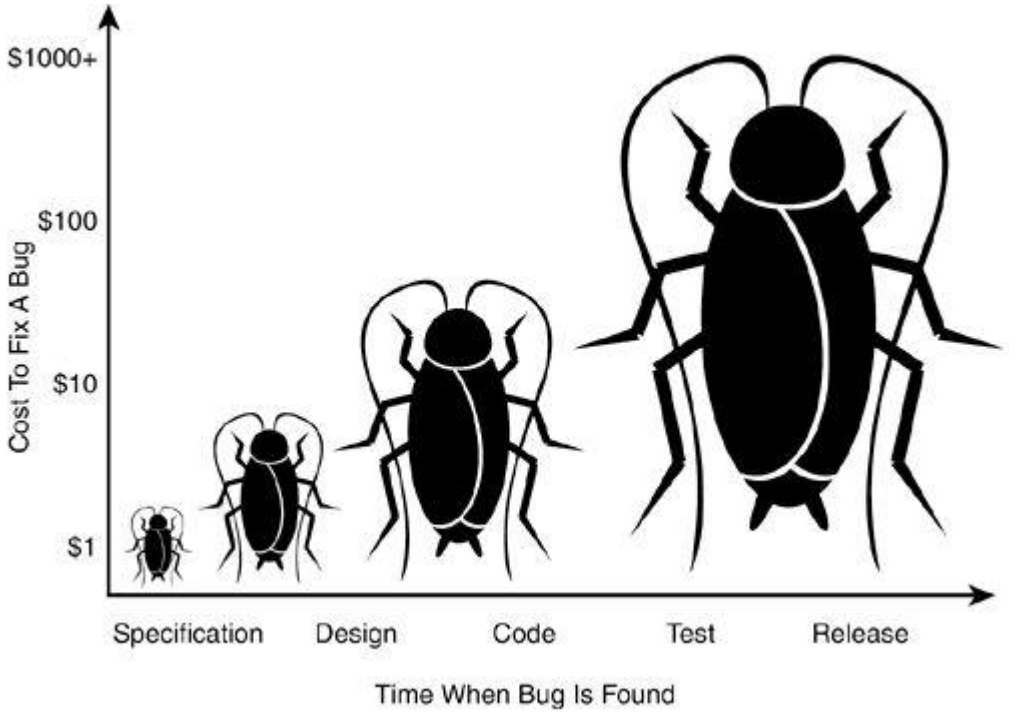
When a software bug occurs?

- A software bug occurs when one or more of the following rules is true [Pat05]:
  - The software doesn't do something that the product specification says it should do.
  - The software does something that the product specification says it shouldn't do.
  - The software does something that the product specification doesn't mention.
  - The software doesn't do something that the product specification doesn't mention but should.
  - The software is difficult to understand, hard to use, slow, or in the software tester's eyes will be viewed by the end user as just plain not right.

Why do bugs occur? [Pat05]



The cost of bugs [Pat05]



Famous Software bugs

- **Mariner 1 rochet – 1962 - diverted from its intended flight path shortly after launch.**
  - **Cause**: A programmer incorrectly transcribed a handwritten formula into computer code.
  - **Cost**: $ 18.5 million
- **World War III – almost – 1983 - The Soviet early warning system falsely indicated the United States had launched five ballistic missiles.**
  - **Cause**: A bug in the Soviet software failed to filter out false missile detections caused by sunlight reflecting off cloud-tops.
  - **Cost**: Nearly all of humanity
- **Therac-25 radiation therapy machine – 1985 - Canada Therac-25 radiation therapy machine malfunctioned and delivered lethal radiation doses to patients.**
  - **Cause**: Because of a subtle bug called a race condition, a technician could accidentally configure Therac-25 so the electron beam would fire in high-power mode without the proper patient shielding.
  - **Cost**: Three people dead, three people critically injured
- **Pentium Fails Long Division – 1993 - Intel highly-promoted Pentium chip occasionally made mistakes when dividing floating-point numbers within a specific range.**
  - **Cause**: The divider in the Pentium floating point unit had a flawed division table, missing about five of a thousand entries and resulting in these rounding errors.
  - **Cost**: $475 million, corporate credibility.
- **Disney's Lion King – 1995 - The Disney company released its first multimedia cd-rom game for children, The Lion King Animated Storybook. Several parents couldn't get the software to work.**
  - **Cause**: Disney failed to test the software on a broad representation of the many different PC models available on the market.
  - **Cost**: cd-rom replacements, corporate credibility.
- **Mars Climate Crasher – 1998 - After a 286-day journey from Earth, the Mars Climate Orbiter fired its engines to push into orbit around Mars. The engines fired, but the spacecraft fell too far into the planet atmosphere, likely causing it to crash on Mars.**
  - **Cause**: The software that controlled the Orbiter thrusters used imperial units (pounds of force), rather than metric units (Newtons) as specified by NASA.
  - **Cost**: $125 million.
- **Cancer Treatment -2000 - Radiation therapy software by Multidata Systems International miscalculated the proper dosage, exposing patients to harmful and in some cases fatal levels of radiation.**
  - **Cause**: The software calculated radiation dosage based on the order in which data was entered, sometimes delivering a double dose of radiation.
  - **Cost**: Eight people dead, 20 critically injured.

Next Lecture (Still today!)

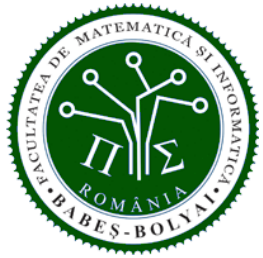
- Inspection

Questions

- Thank You For Your Attention!

References

- [CB03] Jean-Francois Collard and Ilene Burnstein. *Practical Software Testing*. Springer-Verlag New York, Inc., 2003.
- [Cro80] Philip B. Crosby. *Quality Is Free*. Signet Shakespeare, 1980.
- [JJ98] A. Blanton Godfrey Joseph Juran. *JURANS QUALITY HANDBOOK*. McGraw-Hill, 1998.
- [NAS] Nasa - standard for software assurance. http://www.hq.nasa.gov/office/codeq/doctree/87398.htm.
- [NT05] K. Naik and P. Tripathy. *Software Testing and Quality Assurance*. Wiley Publishing, 2005.
- [Pat05] R. Patton. *Software Testing*. Sams Publishing, 2005.
- [Pre00] Roger S. Pressman. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, Inc., 2000.
- [PY08] M. Pezzand and M. Young. *Software Testing and Analysis: Process, Principles and Techniques*. John Wiley and Sons, 2008.
- [Wei92] Gerald Weinberg. *Quality Software Management Vol. 1: Systems Thinking*. DORSET HOUSE PUBLISHING, 1992.
- [BBST] BBST Testing course, http://testingeducation.org/BBST/ (http://testingeducation.org/BBST/bugadvocacy/BugAdvocacy2008.pdf, slides 26-32 for quality definitions, Lecture 1 video, min 27-35)



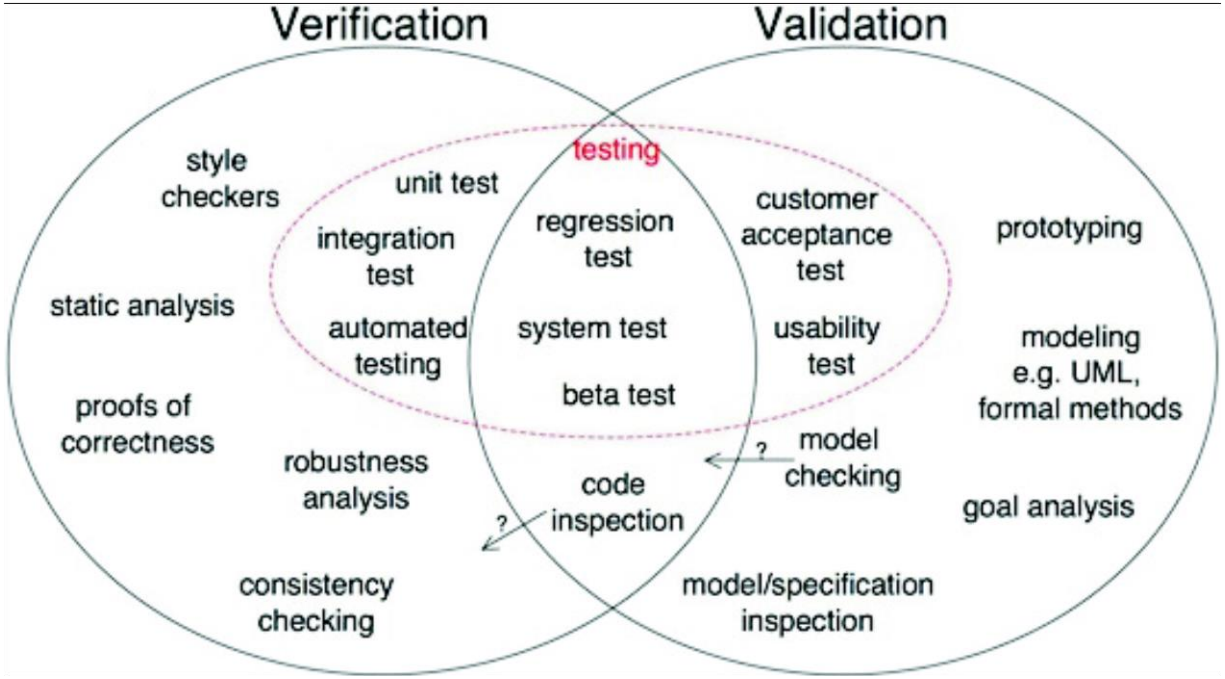
Software Systems Verification and Validation

Assoc. Prof. Andreea Vescan

Babeș-Bolyai University
Cluj-Napoca
2020-2021

Sales paradigm - SSVV

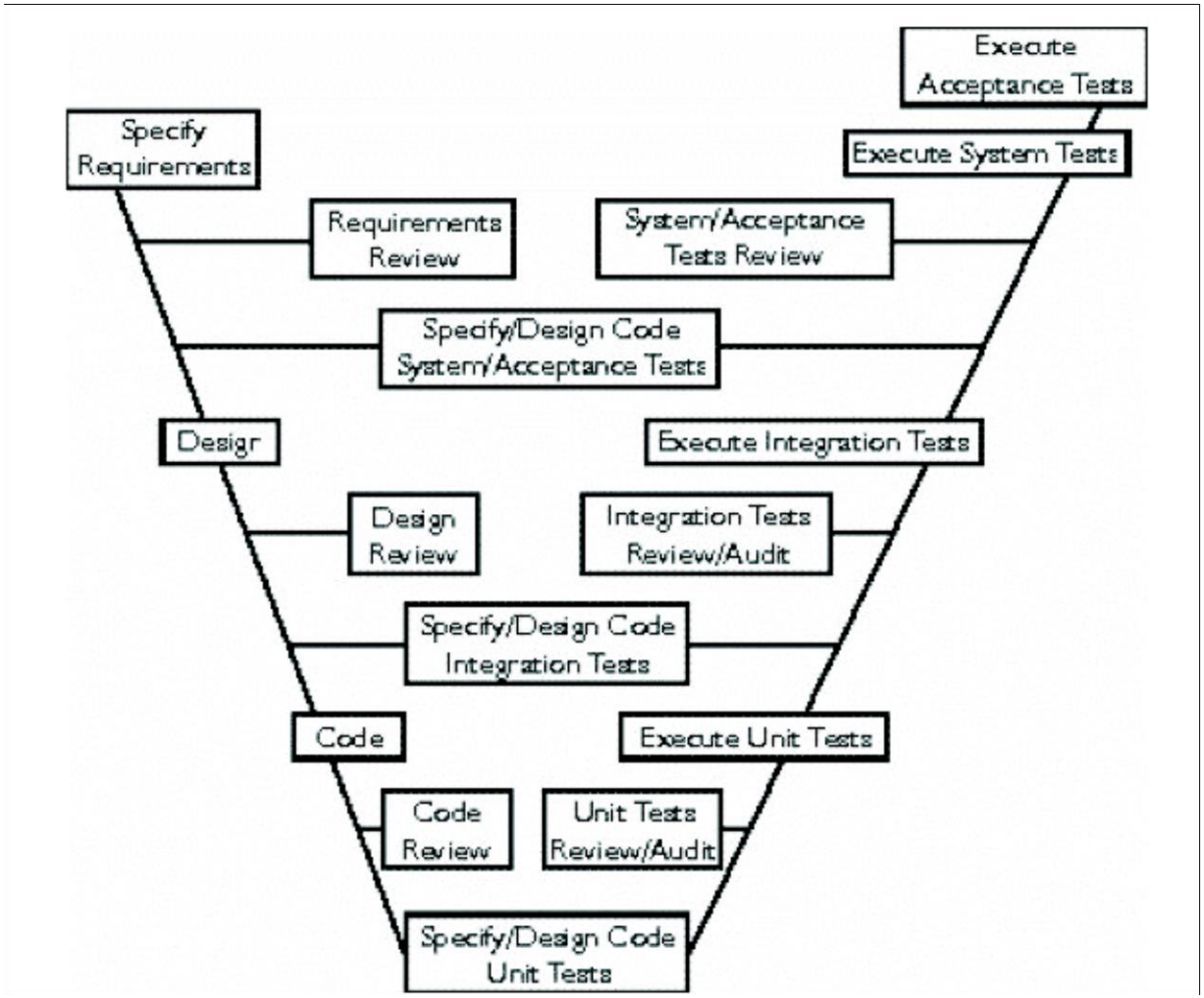
What you will learn!



- http://www.easterbrook.ca/steve/2010/11/the-difference-between-verification-and-validation/

Software development life cycle Model

- Extended V-Model

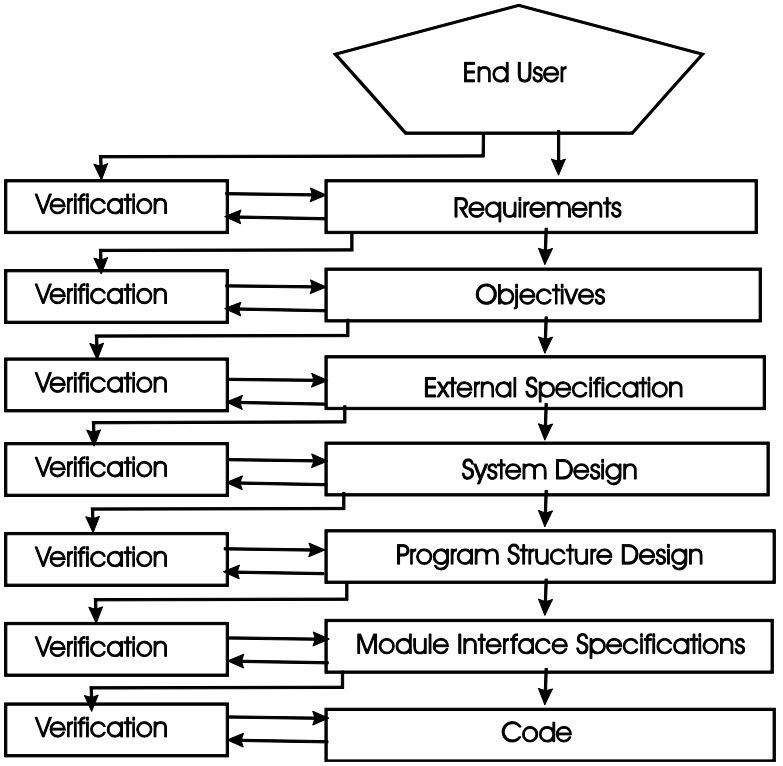


Outline

- Human testing
- Human testing methods
  - Inspections
  - Walkthroughs
  - Pair-programing

Human testing

- Prevent errors
  - introduction of a verification step at the end of each process.

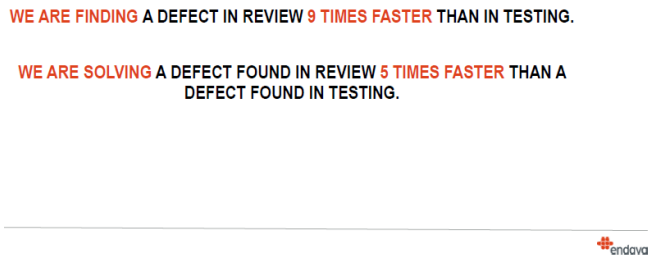


Outline

- Human testing
- Human testing methods
  - Inspections
  - Walkthroughs
  - Pair-programing

Human testing methods
[Mye04] (chapter 3), [PY08] (chapter 18), [Fre10] (chapter 4)

- Is it useful? they contribute to productivity and reliability:
  - The earlier errors are found, the lower the cost of correcting the errors.
  - Psychological change of programmers when computer-based testing commences.
- Human testing methods are:
  - Inspections
  - Walkthroughs
  - Pair-programing



- Objective - to find errors but not to find solutions to the errors.
- Advantage - when an error is found it is usually located.
  - **Finds from 30% to 70% of the logic-design/coding errors in programs (?).**
- Inspection and computer-based testing are complementary.

Outline

- Human testing
- Human testing methods
  - Inspections
  - Walkthroughs
  - Pair-programing

Inspection

- **Inspection** - process of trying to find defects in development documents during various phases of the software development process.
- Fagan Inspection team ([4 members])
  - Moderator - duties
    - Distributing materials and scheduling the inspection session.
    - Leading the session
    - Ensuring that the errors are subsequently corrected.
  - Author of the product (analyst, designer, programmer)
  - Secretary
  - Reader
- Checklists
- Time - 90-120 minutes

Inspection activities

- **Planning**
  - the moderator selects the team members
  - distribution of the materials to the members; task assignment
- **Presentation/Overview - not compulsory**
  - used to present details to the members of the inspection team
- **Individual preparation**
  - reading and understanding the received documentation
- **Inspection meeting**
  - critical observations of each individual inspectors - discussed
  - conclusions of the inspection - documented
- **Rework**
  - the author makes the required changes and correct the errors
- **Follow-up**
  - to verify if the modification did eliminate the errors
  - may be only between the author and the moderator

Inspection checklists

- Inspection scope - to find errors
- Depending on the analyzed document - special kind of errors
- **Specification Document**
  - Does the specification conforms to the user's needs?
  - Are there ambiguities in the specification?
  - Do the input/output date are clearly stated? What about input/output conditions?
  - Are there requirements that are not present in the specification?
  - Are there performance conditions? What precise computation conditions?
- **Analysis Document**
  - Does the design conforms to the specification?
  - Are all the functionalities from the specification specified?
  - Is there an analysis documentation about the made decisions?

Inspection checklists (cont.)

- **Code**
  - Does the code conforms to the design?
  - Are all the methods are called?
  - Are all the variables initialized?
  - Problems with: infinite cycles, out of bound indexes, improper allocation of memory.
- **Test Document**
  - The test cases are well documented?
  - The test cases are well chosen?
  - Are the test data sufficient to coverage criterion?
  - For the integration testing, the order of integration is clear?
  - At regression testing is the testing continued?

Inspection advantages [CB03]

- Early error discovery
- Reduce product development time and cost
- Group method
- Mean to education
- The source of error is known (locating defect)
- Eliminates the debugging stress if few day remains until product release

- Inspection – more efficient than testing [CB03]
  - detecting, locating, repairing defect
  - a two pass approach (individuals first and by the group)
  - checklist – calls attention to specific defect prone areas

Outline

- Human testing
- Human testing methods
  - Inspections
  - Walkthroughs
  - Pair-programing

Walkthrough

- **Walkthrough** [You79], [CB03] - process of trying to find defects in development documents during various phases of the software development process.
- Similar to Inspection
- **Team members** ([3-5] members)
  - Moderator ([CB03]- moderator = the producer of the reviewed material
    - ➔ a larger amount of material can be processed by the group)
  - Secretary
  - Tester
- **Procedures** are slightly different
  - Planning
  - Meeting - the participants “play computer” (no checklist)
  - No Individual preparation [CB03]
  - Rework [You79]
  - Follow-up
- Different error-detection technique
- Time - 90-120 minutes

Outline

- Human testing
- Human testing methods
  - Inspections
  - Walkthroughs
  - Pair-programing

Pair-Programming

- Variation of program inspection.
- Merges coding and inspection activities.
- The inspection activities
  - are not driven by checklists
  - are based on shared programming practice and style
- Programmers frequently alternate roles
- Is carried out in normal work days, without excessive overtime and without severe schedule pressure.
- No mediator, so responsibility for open and nondefensive discussion of decisions/alternatives falls to the programmers.

Outline

- Human testing
- Human testing methods
  - Inspections
  - Walkthroughs
  - Pair-programing
- **Lecture 1 - Exit Ticket**
  - **Mentimeter**

Next Lecture (tentative)

- Testing. Test planning.
- Test case design - Black-box testing

- Testing Management Tool - TestLink
- Continuous integration - Jenkins

Questions

- Thank You For Your Attention!

References

- [PY08] M. Pezzand and M. Young. *Software Testing and Analysis: Process, Principles and Techniques*. John Wiley and Sons, 2008.
- [Mye04] Glenford J. Myers, *The Art of Software Testing,* John Wiley & Sons, Inc., 2004
- [You79] E. Yourdon, Structured Walkthroughs, Prentice-Hall,Englewood Cliffs, NJ, 1979
- [CB03] Jean-Francois Collard and Ilene Burnstein. *Practical Software Testing*. Springer-Verlag New York, Inc., 2003.
- [Fre10] M. Frentiu, Verificarea si validarea sistemelor soft, Presa Universitara Clujeana, 2010