CURS 01B. INSPECTARE

Verificare şi validarea sistemelor soft
[27 Februarie 2018]

Lector dr. Camelia Chisăliţă-Creţu
Universitatea Babeş-Bolyai

Conţinut

- Calitatea produselor soft
  - Evaluarea calității unui produs soft
  - Metode de verificare
- Metode bazate pe factorul uman
  - Definiție. Motivație. Caracteristici
  - Inspectare
  - Walkthroughs
  - Pair-Programming
- Pentru examen...
- Bibliografie

CALITATEA PRODUSELOR SOFT

Evaluarea calității unui produs soft

Metode de verificare

Calitatea produselor soft

- 4 definiții
  - ***"produsul soft este conform cu cerințele documentate"*** **[Pressman2005]:**
    - conformitatea cu cerințele funcţionale și de performanţă precizate şi documentate explicit;
  - ***"produsul soft este conform cu cerințele reale ale utilizatorului"*** **[Crosby1980]:**
    - conformitate cu cerințele reale, nu doar cu cerinţele documentate;
  - ***"produsul soft este adecvat pentru a fi utilizat"*** **[Juran1998]:**
    - orice aspect care îl mulțumește sau nu îl mulţumeşte pe utilizator, i.e., *satisfiers*, *dissatisifiers*;
  - ***"produsul soft este relevant/important pentru o persoană"*** **[Weinberg1992]:**
    - calitatea este subiectivă;

Evaluarea calităţii unui produs soft

- asigurarea calităţii vs. controlul calităţii (Curs 01A);
- evaluarea calităţii = controlul calităţii
  - aplicarea unor *metode de verificare* prin care se stabileşte modul în care produsul soft satisface anumite criterii (atribute) de calitate necesare care indică nevoile beneficiarului;
- beneficiar
  - client, utilizator final;
  - programator, tester, manager de proiect;

Activități asociate controlului calității

Analiza statică

- examinarea unor documente (specificații, modele conceptuale, diagrame de clase, cod sursă, planuri de testare, documentații de utilizare);
- **exemple:** activități de inspectare a codului, analiza algoritmului, demonstrarea corectitudinii;
- **NU presupune execuția propriu-zisă a programului dezvoltat;**

Analiza dinamică

- examinarea comportamentului programului cu scopul de a evidenția defecțiuni posibile;
- **exemple:** *tipuri de testare* (de regresie, funcţională, non-funcţională), *niveluri de testare* (testare unitară, testare de integrare, testare de sistem, testare funcţională, testare de accpetare);
- **include activitatea de execuție propriu-zisă a programului (testare);**

- **metode de analiză complementare;**
- **dezvoltatorii aplică metode hibride, care folosesc avantajele celor două abordări.**

Metode de verificare

- preconcepţie (anii ’60) –
  - *„singura modalitate de a verificare a unui program este execuţia pe calculator”* [Myers2004, Cap.3];
  - se presupunea că un program este scris doar pentru execuția de către calculator și nu este util și necesar să fie citit și înțeles de o persoană, e.g., programator, tester;

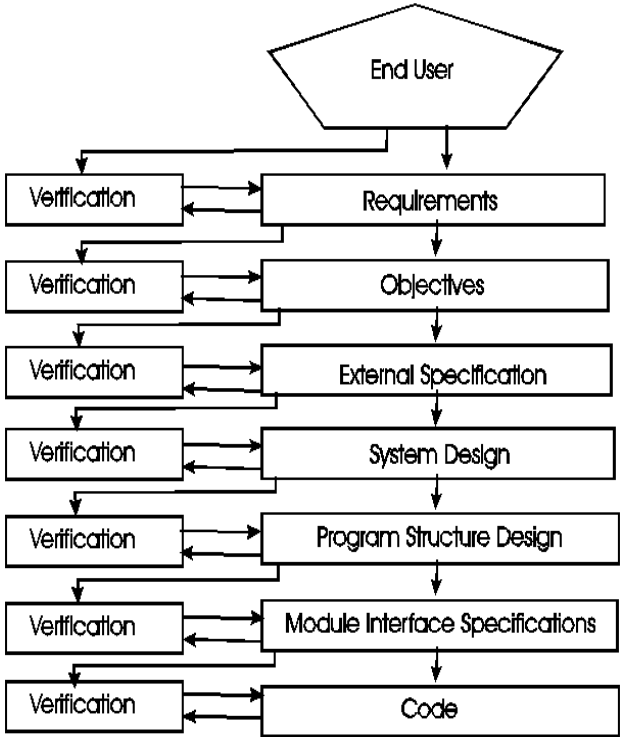
- metode de verificare bazate pe:
  - *factorul uman* (*engl.* **human-based testing**, HbT);
  - *calculator* (*engl.* **computer-based testing**, CbT);

METODE BAZATE PE FACTORUL UMAN

Definiție. Motivaţie. Obiective
Avantaje și dezavantaje. Hbt vs CbT
Inspectare
Walhthroughs
Pair-Programming

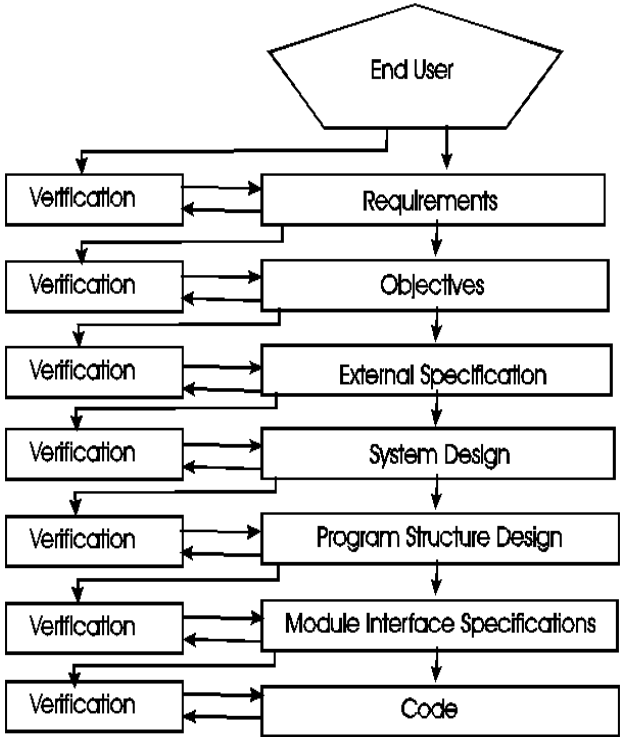
Metode bazate pe factorul uman

- **metodă HbT** [Young2008, Frentiu2010]
  - verificare efectuată de o persoană sau un grup persoane la sfârșitul fiecărei etape a procesului de dezvoltare a softului și înainte de a demara umătoarea fază de dezvoltare;
- **tipuri de metode HbT:**
  - inspectare, walkthroughs, pair-programing;
- **exemplu:**
  - activitate: inspectarea codului sursă;
  - se efectuează după etapa de implementare și înainte de începerea testării.



Metode HbT. Obiective. Motivație

- **obiective**
  - *identificarea defectelor;*
- **motivaţie**
  - *utilizarea metodelor HbT contribuie la creșterea productivităţii şi a gradului de încredere că rezultatul obţinut îndeplineşte cerinţele:*
    - costul de corectare (eliminare) al defectelor creşte odată cu parcurgerea etapelor de dezvoltare a softului;
    - modificarea comportamentului programatorilor la demararea CbT, i.e., la depanare se introduc mai multe bug-uri;



Metode HbT. Avantaje și dezavataje

Avantaje

- sunt implicate în proces şi **alte persoane** pe lângă autorul documentului verificat;
- permite **localizarea** defectelor;
- **identifică între 30% şi 70% din bug-urile de proiectare şi implementare ale produselor soft;**

Dezavantaje

- nu sunt eficiente la identificarea **erorilor majore** de proiectare;
- nu pot evidenţia situaţii excepţionale care apar în utilizarea propriu-zisă a softului;

Metode HbT vs Metode CbT

Metode HbT

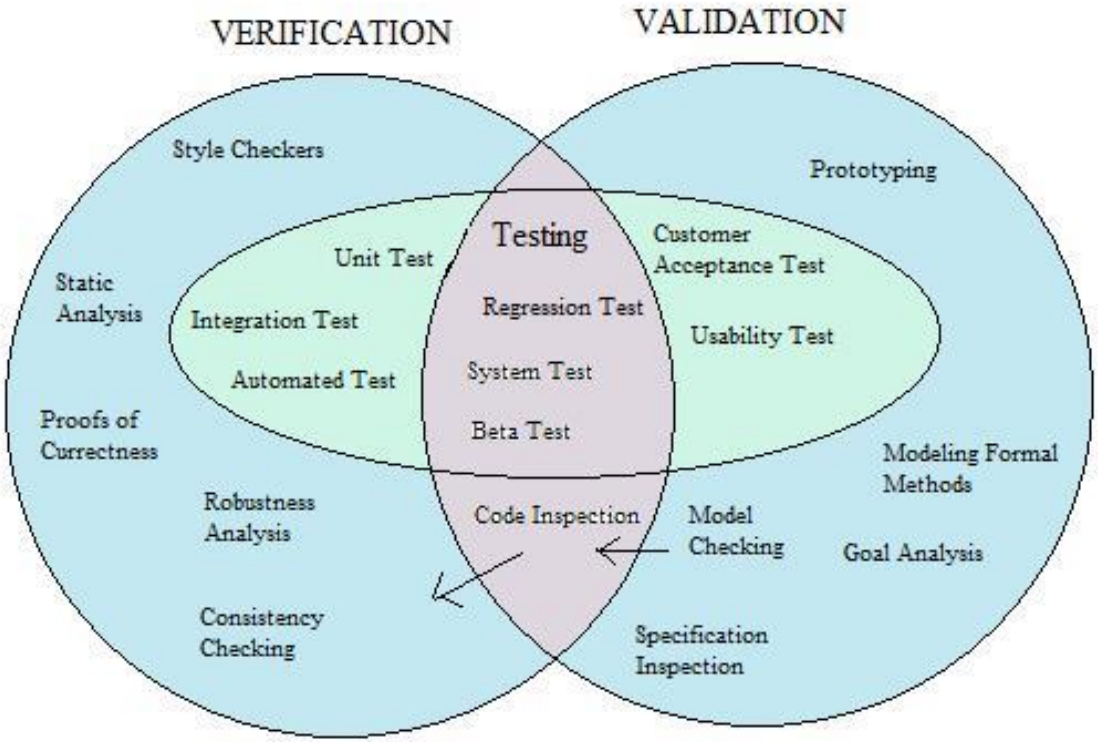
- permit identificarea mai multor erori care pot fi corectate simultan;
- nu presupun execuţia programului;

Metode CbT

- sugerează doar un simptom, fiecare eroare identificată fiind eliminată individual;
- este necesar ca programul să fie executat;
- poate să evidenţieze o defecţiune doar în anumite situaţii;

Metodele HbT (e.g., inspectare) şi metodele CbT (e.g., testare) sunt metode complementare.

Metode HbT şi metode CbT în Verificare şi Validare



- **metodele HbT şi metodele CbT sunt activităţi de verificare şi validare**

- sursa: [Pal2013]

INSPECTARE

Definiție. Caracteristici
Echipa de inspectare. Atribuţiile membrilor
Activităţi de inspectare
Checklists. Definiţie. Motivaţie. Tipuri de checklists
Avantaje

Inspectare Fagan. Definiție. Caracteristici

- 1976 – Fagan [Fagan1976] introduce la IBM procesul de inspectare;
- **Inspectare**
  - proces structurat prin care se încearcă **identificarea defectelor** din **documentele elaborate** pe parcursul etapelor de dezvoltare a softului, pe baza unor **criterii prestabilite**;
- **Caracteristici**
  - **echipa de inspectare (4 membri):**
    - moderator, autor, secretar, prezentator;
  - **activităţi de inspectare (6 etape):**
    - planificarea, prezentarea, pregătirea, şedinţa de analiză, corectarea, reinspectarea;
  - **tipuri de erori căutate:** *checklists*, adaptate tipului de document inspectat;
  - **timp de desfăşurare:** 90-120 minute.

Echipa de inspectare. Atribuţiile membrilor

- **moderator**
  - distribuie materialele şi planifică sesiunile de inspectare; conduce sesiunea de inspectare;
  - urmăreşte modul în care sunt corectate erorile;
- **autorul** documentului inspectat (analist, proiectant, programator, tester);
  - răspunde la întrebările adresate de membrii echipei, clarifică nelămuririle semnalate de către aceştia;
  - participă la discuţiile purtate în timpul şedinţei de analiză; remediază defecţiunile constatate;
- **secretar**
  - redactează concluziile şedinţei de analiză;
  - înregistrează defectele semnalate şi problemele discutate într-un document (raport de inspectare);
- **prezentator** (reader)
  - citeşte în cadrul şedinţei de analiză părţi ale documentului inspectat;
- **inspectori** – cu excepţia autorului, toţi ceilalţi sunt consideraţi inspectori;
  - analizează documentul primit cu scopul de a identifica cât mai multe defecte (bug-uri).

Activităţi de inspectare (1)

1. **planificarea** (*engl.* **planning**)
   - moderatorul alege membrii echipei de inspectare;
   - distribuie materialele tuturor membrilor echipei şi atribuie sarcini de inspectare;
   - verifică dacă documentul care trebuie inspectat este complet şi acceptabil pentru a fi inspectat;
2. **prezentarea** (*engl.* **overview**) **– nu este obligatorie**
   - se prezentă detaliile materialului inspectat tuturor membrilor echipei de inspectare;
   - moderatorul poate decide dacă este necesară etapa de prezentare sau se trece direct la pregătirea individuală;
3. **pregătirea individuală** (*engl.* **preparation**)
   - citirea atentă şi înţelegerea documentului primit pentru inspectare;
   - inspectorii reţin toate observaţiile critice şi formulează întrebări referioare la aspectele care nu sunt clare.

Activităţi de inspectare (2)

4. **şedinţa de inspectare** (*engl.* **inspection meeting**)
   - se discută observaţiile critice ale fiecărui inspector;
   - secretarul notează observaţiile considerate prin consens ca fiind defecte şi ulterior redactează concluziile inspectării;
   - concluziile inspectării sunt predate autorului documentului inspectat pentru a corecta greşelile;
5. **corectarea** (*engl.* **rework**)
   - autorul efectuează modificările necesare şi corectează erorile;
6. **reinspectarea** (*engl.* **follow-up**)
   - se verifică dacă modificările efectuate au eliminat erorile;
   - se poate reduce la o întâlnire între autor şi moderator.

Checklists. Definiţie. Motivaţie. Tipuri de checklists

- **checklist** = listă cu defecte frecvent întâlnite într-un anumit tip de document;
- **motivaţie**
  - obiectivul inspectării: identificarea defectelor;
  - în raport cu documentul analizat, se urmăreşte identificarea unor bug-uri specifice;
- **tipuri de checklists** pentru inspectarea:
  - documentaţiei de specificare;
  - documentaţiei de analiză;
  - documentaţiei de codificare;
  - documentaţiei de testare.
- Fiecare checklist conţine aspecte particulare documentelor inspectate şi sunt rezultatul experienţei acumulate în identificarea greşelilor întâlnite frecvent în desfăşurarea unor etape de dezvoltare software.

Tipuri de checklists (1)

- checklist utilizat la inspectarea **documentaţiei de specificare**:
  1. specificaţia respectă cerinţele beneficiarului?
  2. există ambiguităţi în specificare?
  3. datele de intrare şi de ieşire, cât şi condiţiile de intrare şi de ieşire asociate sunt specificate corect?
  4. există cerinţe care nu sunt specificate în document?
  5. există cerinţe de precizie a datelor? sunt clar exprimate?
  6. există cerinţe de performanţă?
- checklist utilizat la inspectarea **documentaţiei de analiză**:
  1. se respectă specificaţia?
  2. toate funcţionalităţile din specificare au fost descrise?

Tipuri de checklists (2)

- checklist utilizat la inspectarea **documentaţiei de codificare**:
  1. codul sursă respectă cerinţele de proiectare, specificaţiile şi cerinţele utilizatorului?
  2. sunt apelate toate metodele?
  3. sunt iniţializate toate variabilele?
  4. aspecte analizate în mod special: cicluri infinite, accesarea unui index non-valid, alocarea şi accesarea memoriei.
- checklist utilizat la inspectarea **documentaţiei de testare**:
  1. toate cazurile de testare au fost documentate complet?
  2. cazurile de testare sunt relevante?
  3. datele de testare satisfac criteriul de acoperire ales?
  4. la testarea de integrare este clară ordinea de integrare?

Inspectare Fagan. Avantaje

- **avantaje**
  - permite descoperirea defectelor devreme;
  - reducere costul şi timpul de dezvoltare;
  - metodă de grup – membrii echipei conlucrează;
  - modalitate de învăţare la nivelului echipei;
  - **stabileşte sursa defecţiunii, nu oferă doar indicii** (e.g., testarea) referitoare la existenţa lor;
  - elimină stresul depanării într-un timp foarte scurt.

- **Inspectare vs. Testare** [Collard2003]
  - identificarea, localizarea şi eliminarea defectului;
  - abordare aplicată în două etape (individual şi apoi de grup);
  - checklists se focalizează pe anumite părţi ale documentului care sunt predispuse la introducerea de defecte pe parcursul dezvoltării softului;

WALKTHROUGHS

Definiţie. Caracteristici

Walkthroughs vs Inspectare

Walkthroughs. Definiție. Caracteristici

- **walkthroughs** [Yourdon1979]
  - procesul prin care se încearcă identifcarea defectelor din documentele elaborate pe parcursul etapelor de dezvoltare a softului;
- **caracteristici** [Yourdon1979 , Collard2003]
  - **echipa de realizare (3-5 membri) :**
    - secretar, inspectori şi moderator (autorul documentului inspectat, i.e., analist, proiectant, programator, tester);
  - **activităţile de walkthrough (4 etape):**
    - planning, meeting, rework, follow-up;
  - aplică tehnici de identificare a erorilor diferite de inspectarea Fagan, i.e., nu se folosesc checklists;
  - **timp de realizare:** 90-120 minute.

Walkthroughs vs Inspectare

Walkthroughs

- activitate mai puţin riguroasă;
- echipa este formată din 3-5 membri;
- se desfăşoară în 4 etape;
- nu are pretenţia identificării tuturor defectelor;
- autorul conduce echipa de walkthrough;
- folosind scenarii prestabilite;

Inspectare

- activitate riguroasă;
- echipa este formată din 4 membri;
- se desfăşoară în 6 etape;
- identifică defectele des întalnite;
- moderatorul conduce echipa de inspectare;
- foloseşte checklists pentru identificarea defectelor;

PAIR-PROGRAMMING

Definiţie. Caracteristici

Pair-Programming. Definiție. Caracteristici

- **pair-programming**
  - metodă de elaborare a programelor, în care două persoane lucrează împreună;
- **caracteristici**
  - combină activităţile: inspectarea codului şi implementarea (codificarea);
  - programatorii alternează rolurile;
  - **activităţi de inspectare:**
    - nu sunt determinate de checklists;
    - se bazează pe împărtăşirea aceloraşi principii de programare şi a unui stil de programare asemănător;
  - **timp de desfăşurare:** durata unei zile normale de muncă, fără exces de ore suplimentare sau presiunea unui program de lucru strict;
  - nu există mediatori, iar responsabilitatea pentru atmosfera de lucru deschisă şi non-agresivă depinde de programatori.

PENTRU EXAMEN…

Pentru examen...

- **concepte, caracteristici, asemănări şi diferenţe:**
  - verificare, validare; verificare vs. validare;
  - eroare, defect/bug, defecţiune; eroare vs. defect/bug vs. defecţiune;
  - HbT, motivaţie şi aplicare HbT, HbT vs. CbT;
  - inspectare, echipa de membri: enumerare şi descrierea rolurilor, activităţi de inspectare : enumerare şi descrierea lor;
  - avantajele inspectării;
  - walkthroughs, walkthroughs vs. inspectare;
  - pair-programming, caracteristici.

Cursul următor…

- **Testare**
  - modele folosite în testare;
  - planuri de testare;
  - cazuri de testare;
  - raportarea testării;
- **Testare Black-box. Tehnici de alegere a datelor de test:**
  - împărţirea în clase de echivalenţă;
  - analiza valorilor limită;
- **Testing Management Tool – TestLink**
  - prezentare tool.

Referințe bibliografice

- **[Crosby1980]** Philip B. Crosby, *Quality Is Free*, Signet Shakespeare, 1980.
- **[Juran1998]** A. Blanton Godfrey, Joseph Juran, *JURANS QUALITY HANDBOOK*, McGraw-Hill, 1998.
- **[Weinberg1992]** Gerald Weinberg, *Quality Software Management , Vol. 1: Systems Thinking,* Dorset House Publishing, 1992.
- **[Pressman2000]** Roger S. Pressman, *Software Engineering: A Practitioner's Approach*, McGraw-Hill, Inc., 2000.
- **[Pal2013]** Kaushik Pal, *Software Testing: Verification and Validation,* http://mrbool.com/software-testing-verification-and-validation/29609l
- **[Fagan1976]** M. E. Fagan, *Design and code inspections to reduce errors in program development*, IBM Systems Journal, pages 182–211, 1976.
- **[Collard2003]** J. F. Collard, I. Burnstein. *Practical Software Testing*. Springer-Verlag New York, Inc., 2003.
- **[Yourdon1979]** E. Yourdon, *Structured Walkthroughs*, Prentice-Hall,Englewood Cliffs, NJ, 1979.
- **[Myers2004]** Glenford J. Myers, *The Art of Software Testing,* John Wiley & Sons, Inc., 2004
- **[Young2008]** M. Pezzand, M. Young. *Software Testing and Analysis: Process, Principles and Techniques*. John Wiley and Sons, 2008.
- **[Frentiu2010]** M. Frentiu, *Verificarea si validarea sistemelor soft*, Presa Universitara Clujeana, 2010.