

CURS 02A. TESTARE

Verificarea și validarea sistemelor soft
[04 Martie 2025]

Lector dr. Camelia Chisăliță-Crețu
Universitatea Babeș-Bolyai

Conținut

- Evaluarea calității unui produs soft
 - Activități asociate calității
 - Controlul calității. Activități asociate
 - Metode de verificare și validare
- Testare
 - Program. Program testat
 - Definiții ale testării
 - Caz de testare. Definiții. Caracteristici
 - Tipuri de testare
 - Principii de testare. Axiome de testare
- Procesul de testare
 - Întrebări fundamentale
 - Activități ale procesului de testare
- Bibliografie

EVALUAREA CALITĂȚII UNUI PRODUS SOFT

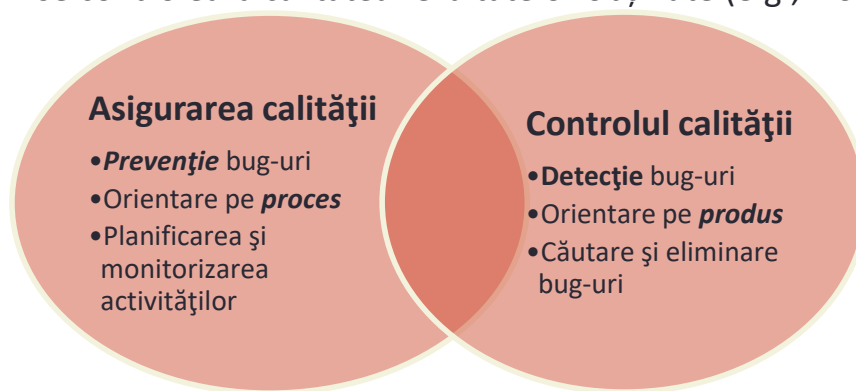
Activități asociate calității

Controlul calității. Activități asociate

Metode de verificare și validare

Activități asociate calității

- *în procesul de dezvoltare, calitatea este abordată din perspectiva:*
 - **procesului** ==> **asigurarea calității** (*engl. quality assurance*):
 - **Obiectiv:** asigură respectarea standardelor, planurilor și etapelor proceselor de dezvoltare necesare elaborării adecvate a produsului cerut;
 - **Întrebare:** Cum se asigură calitatea activităților desfășurate în procesul dezvoltare?
 - **produsului** ==> **controlul calității** (*engl. quality control*):
 - **Obiectiv:** identifică deficiențele în produsul obținut;
 - **Întrebare:** Cum se controlează calitatea rezultatelor obținute (e.g., work products) în urma activităților desfășurate?



Controlul calității. Activități asociate (1)

Analiză statică (static testing)

- 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;
- se pot baza pe factorul uman (reviews) sau utilizarea tool-urilor (analiza statică).

Analiză dinamică (dynamic testing)

- examinarea comportamentului programului cu scopul de a evidenția defecțiuni posibile;
- **exemple:** *tipuri de testare* (de regresie, funcțională, non-funcțională), *niveluri de testare* (testare unitară, testare de integrare, testare de sistem, testare funcțională, testare de acceptare);
- se bazează întotdeauna pe execuția programului.

Controlul calității. Activități asociate (2)

Analiză statică (static testing)

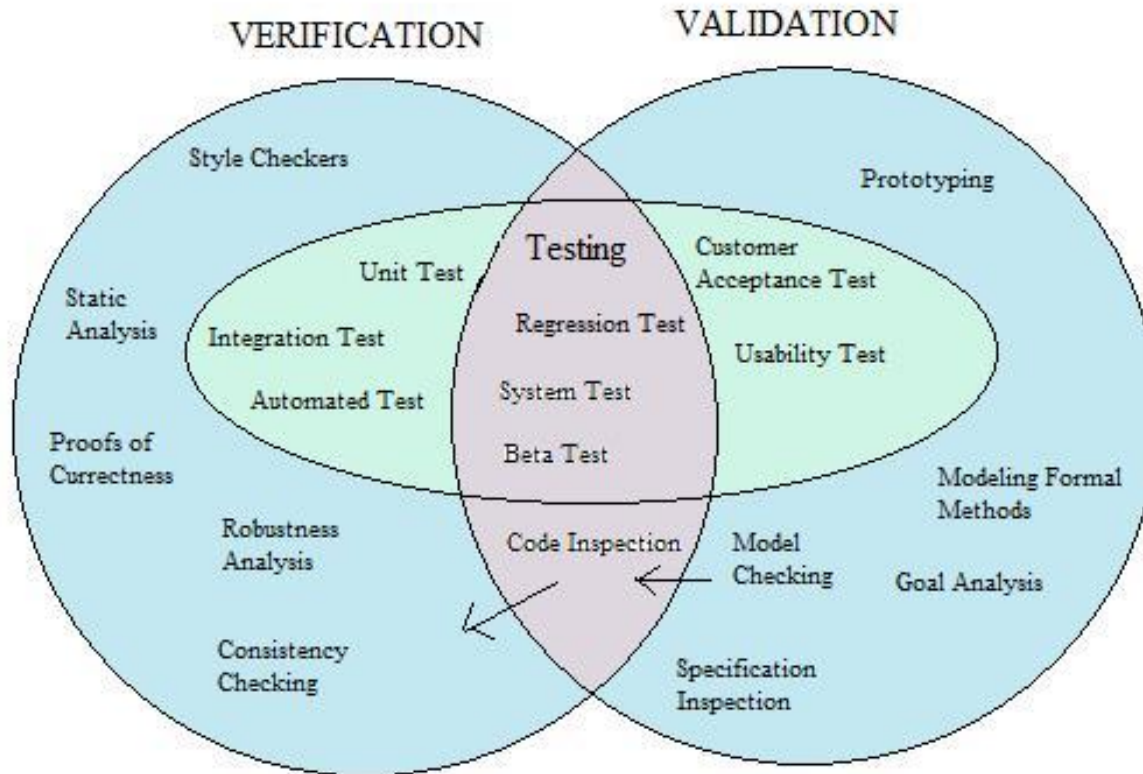
- permit identificarea mai multor erori (greșeli) care pot fi corectate simultan;
- **NU** presupune execuția propriu-zisă a programului dezvoltat;

Analiză dinamică (dynamic testing)

- sugerează doar un simptom, fiecare eroare identificată fiind eliminată individual;
- include activitatea de execuție propriu-zisă a programului (testare);
- poate să evidențieze o defecțiune doar în anumite situații.

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

Metode de Verificare și Validare



sursa: [Pal2013]

TESTARE

Program. Program testat

Testare. Definiții

Caz de testare. Definiții. Caracteristici

Tipuri de testare

Principii de testare. Axiome de testare

Program. Definiție

- **program** (engl. **computer program, software application, software product**):
 - listă de instrucțiuni sau o mulțime de metode sau module care permit execuția de către un calculator;
- **un program** este
 - o **comunicare**
 - între **persoane și calculatoare**
 - care sunt **separate** în timp și spațiu
 - și conține **instrucțiuni** care sunt **executate** de către calculator. [\[BBST2010\]](#)

Program testat. Definiție

- **program testat** (*engl. software under test, SUT*):
 - \approx funcție matematică;
 - $P : D \rightarrow R$, unde
 - D – mulțimea datelor de intrare;
 - R – mulțimea datelor de ieșire așteptate.

Testare. Definiții. Caracteristici

1. *semnalează prezența defectelor unui program, fără a garanta absența acestora* [[Dijkstra1969](#)].
 2. *procesul de execuție al unui program cu scopul de a identifica erori* [[Myers2004](#)].
 3. *observarea comportării unui program în mai multe execuții* [[Frențiu2010](#)].
 4. *investigație tehnică și empirică realizată cu scopul de a oferi beneficiarilor testării informații referitoare la programul testat* [[BBST2010](#)].
- Testarea este un proces
 - distructiv;
 - se poate finaliza cu succes (**passed**) sau eșec (**failed**).

Caz de testare. Definiție

- **caz de testare** (*engl. test case*) –
 - mulțime de **date de intrare, condiții de execuție și rezultate așteptate**, proiectate cu un anumit scop (e.g., cum ar fi parcurgerea unui drum particular în execuția programului sau pentru a verifica respectarea unei cerințe specifice) [[IEEE1990](#)];
 - **o interogare** adresată de tester programului testat [[BBST2010](#)];
 - este relevant obiectivul informațional, i.e., **informația pe care o descoperim prin testare**, e.g., testul este *passed* sau *failed*, timpul de execuție asociat testului este foarte mare.
 - **notație:** (i, r) , $i \in D$, $r \in R$;
 - pentru intrarea i se așteaptă să se obțină rezultatul r .

Tipuri de testare. Definiții

- **testare exhaustivă (testare completă, *engl.* exhaustive testing, complete testing):**
 - testare cu toate cazurile de testare posibile, folosind toate datele și scenariile de utilizare posibile;
 - dacă D este finit atunci P se poate executa pentru fiecare $i \in D$;
 - în majoritatea situațiilor D nu este finit, deci testarea exhaustivă nu este posibilă și nici eficace;
- **testare selectivă (*engl.* selective testing):**
 - testare cu o submulțime de cazuri de testare;
 - dacă D nu este finit, atunci se alege o parte din elementele i , unde $i \in S, S \subset D$.
- **depanare (*engl.* debugging, bug fixing):**
 - proces de localizare și eliminare al unui bug care a fost evidențiat prin testarea programului;
 - se formulează ipoteze asupra comportamentului programului, se corectează defectele și apoi se reia procesul de testare.

Principii de testare [\[Myers2004\]](#) [Cap2. secțiunea Software Testing Principles]

1. Definește rezultatele așteptate în urma testării.
2. Evită să testezi programelor proprii.
3. Analizează riguros rezultatele fiecărui test.
4. Scrie cazuri de testare atât pentru condiții de intrare valide cât și pentru cele non-valide.
5. Testează dacă programul **nu** face ceea ce se precizează în specificație, dar și dacă ceea ce face programul **nu** este descris în specificații.
6. Păstrează întotdeauna cazurile de testare.
7. Organizează și planifică procesul de testare, considerând că se vor identifica bug-uri.
8. Testarea este o activitate de stimulare a creativității.
 - *The goal of a software tester is to find bugs, find them as early as possible, and make sure they get fixed.*

Axiome ale testării [\[Patton2005\]](#) [\[Cap.3. secțiunea Testing Axioms\]](#)

1. Este imposibil ca un program să fie complet (exhaustiv) testat.
2. Testarea softului presupune asumarea unui risc.
3. Testarea nu poate demonstra ca bug-urile nu există.
4. Numărul mare de bug-uri asociat unei funcționalități este un indicator al prezenței altor bug-uri – bug-urile pot fi grupate în anumite funcționalități, nu sunt izolate.
5. Paradoxul pesticidului (în testare): cu cât un program este testat mai mult folosind aceleași teste (tehnici de testare), imunitatea la testare crește (nu se descoperă bug-uri noi).
6. Nu orice bug identificat va fi eliminat.
7. Specificația produsului soft se schimbă în permanență.
8. Testerii nu sunt cei mai apreciați membri ai echipei de dezvoltare.

PROCESUL DE TESTARE

Întrebări fundamentale

Activități ale procesului de testare

Întrebări fundamentale (1)

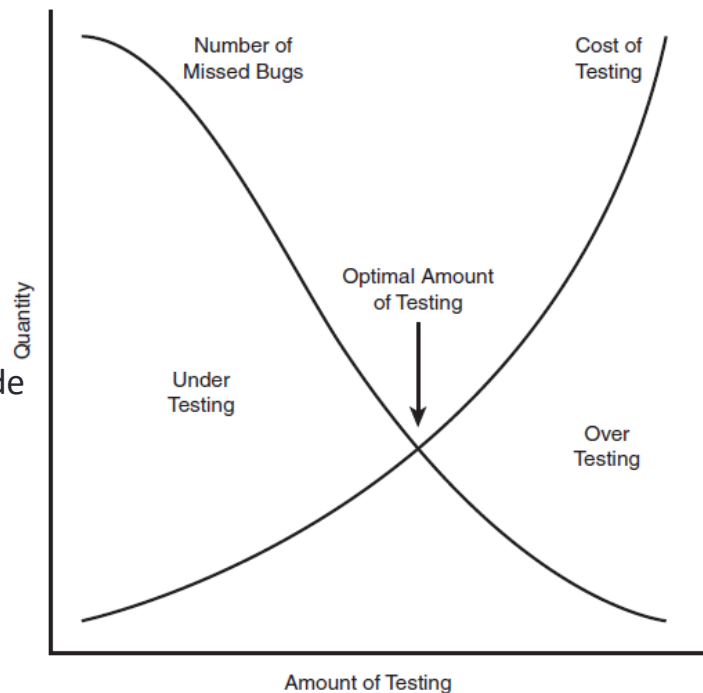
- **De ce este necesar sa testăm un produs soft ? Care este scopul testării ?**
 - evaluarea unor caracteristici sau attribute ale calității produsului soft;
 - descoperirea unor **informații** referitoare la produsul soft [[BBST2010](#)];
 - **obiective ale testării** = aspectele de interes care vor fi investigate (evaluate) în procesul de testare;
- **Cum se organizează procesul de testare ?**
 - **contextul aplicației** = particularități de realizare a testării, e.g., componenta testată, beneficiarul testării (**stakeholder**), diverse **constrângeri**, etc;
 - **misiunea testării** = acțiunea desfășurată prin testare pentru a atinge **obiectivele testării**;
 - **strategie de testare** = cadru general prin care se determină care sunt cele mai potrivite teste care trebuie proiectate (i.e., ce **tehnici de testare** se aplică), astfel încât testarea să își atingă obiectivele informaționale, luând în considerare **contextul aplicației** în care se desfășoară testarea;
 - **tehnică de testare** = *metodă de proiectare, implementare și interpretare a rezultatelor unui test*;
 - **abordare a testării** = modalitate de aplicare a unei tehnici de testare, e.g., *black-box testing, white-box testing, grey-box testing, exploratory testing, scripted testing*.

Întrebări fundamentale (2)

- Cum determinăm momentul în care putem realiza testarea? *Care sunt condițiile care trebuie îndeplinite pentru a demara procesul de testare?*
 - criterii de începere a testării (*engl. entry criteria*);
- Cum determinăm momentul în care testarea efectuată este suficientă? *Cât timp testăm, câte teste executăm?*
 - criterii de terminare a testării (*engl. exit criteria*).

Întrebări fundamentale. Exemplu

- **Exemplu. Criteriu de terminare a testării într-o strategie bazată pe risc:**
- Se analizează relația dintre numărul de teste executate, i.e., **amount of testing**, și numărul de bug-uri identificate, i.e., **quantity**;
- **Over testing:**
 - Dacă se testează tot/mult: costurile cresc, numărul de bug-uri scade
 - raportul $\frac{\text{bugs found}}{\text{testing costs}}$ devine mic ==> **eficiența testării scade**;
- **Under testing:**
 - Dacă se testează puțin sau se iau decizii nepotrivite legate de **CE** se va testa: costurile sunt mici, numărul de bug-uri rămâne ridicat
 - raportul $\frac{\text{bugs found}}{\text{testing costs}}$ rămâne mare ==> **calitate redusă**;
- **Fiecare proiect soft are un cost de testare optim.**



Activități ale procesului de testare

1. planificare (*engl. test planning*):

- stabilirea obiectivelor (*de ce testăm*);
- stabilirea terminării testării (*cât testăm*);
- identificarea unității de program care trebuie testată (*ce testăm*);
- elaborarea strategiei de testare (*cum testăm, ce tehnici aplicăm, ce abordare folosim*);

2. proiectare (*engl. test design*):

- stabilirea datelor de intrare;
- stabilirea rezultatului așteptat;
- configurarea mediului de execuție pentru program;

3. testare (*engl. test execution*):

- execuția programului, i.e, rularea testelor;

4. analiza (*engl. test result analysis*):

- analiza rezultatului testului (*evaluarea rezultatului*);
- raportarea bug-urilor;

5. monitorizare (*engl. test control and monitoring*):

- supravegherea procesului de testare;
- evaluarea și îmbunătățirea procesului de testare.

• QA vs QC în procesul de testare:

- activități QC: 2 .. 4;
- activități QA: 1, 5;

Referințe bibliografice

- **[Pal2013]** Kaushik Pal, *Software Testing: Verification and Validation*, <http://mrbool.com/software-testing-verification-and-validation/29609>
- **[Dijkstra1969]** E.W. Dijkstra, *Software engineering techniques*, Report on a conference sponsored by the NATO Science Committee, Rome, Italy, 27-31 October 1969.
- **[Myers2004]** Glenford J. Myers, *The Art of Software Testing*, John Wiley & Sons, Inc., 2004
- **[Frentiu2010]** M. Frentiu, *Verificarea si validarea sistemelor soft*, Presa Universitara Clujeana, 2010.
- **[BBST2010]** Black-Box Software Testing (BBST), Foundations, <http://www.testingeducation.org/BBST/foundations/BBSTFoundationsNov2010.pdf>.
- **[IEEE990]** IEEE, IEEE STD 610, In IEEE Standard Glossary of Software Engineering Terminology, 1990.
- **[Patton2005]** R. Patton, *Software Testing*, Sams Publishing, 2005.
- **[ISTQBCertification2020]** ISTQB Exam Certification, <http://tryqa.com/what-is-a-defect-life-cycle/>.
- **[BBST2011]** BBST – Test Design, Cem Kaner, <http://www.testingeducation.org/BBST/testdesign/BBSTTestDesign2011pfinal.pdf>