

# CURS 02A. TESTARE

---

**Verificarea și validarea sistemelor soft**  
**[06 Martie 2018]**

Lector dr. Camelia Chisăliță-Crețu  
Universitatea Babeș-Bolyai

# Conținut

- Evaluarea calității unui produs soft
  - Analiza statică vs Analiza dinamică
  - Metode de verificare și validare
- Testare
  - False definiții. Definiții ale testării. Caracteristici
  - Principii de testare. Axiome de testare
  - Program. Caz de testare. Tipuri de testare
- Procesul de testare
  - Întrebări fundamentale
  - Activități ale procesului de testare
  - Reguli de raportare a unui bug
  - Ciclul de viață al unui bug
- Bibliografie

# EVALUAREA CALITĂȚII UNUI PRODUS SOFT

---

Analiza statică vs Analiza dinamică

Metode de verificare și validare

# Evaluarea calității unui produs soft

- asigurarea calității vs. controlul calității (Curs 01A);
- evaluarea calității = controlul calității (Curs 01B);

# Activități asociate controlului calității

## Analiza statică

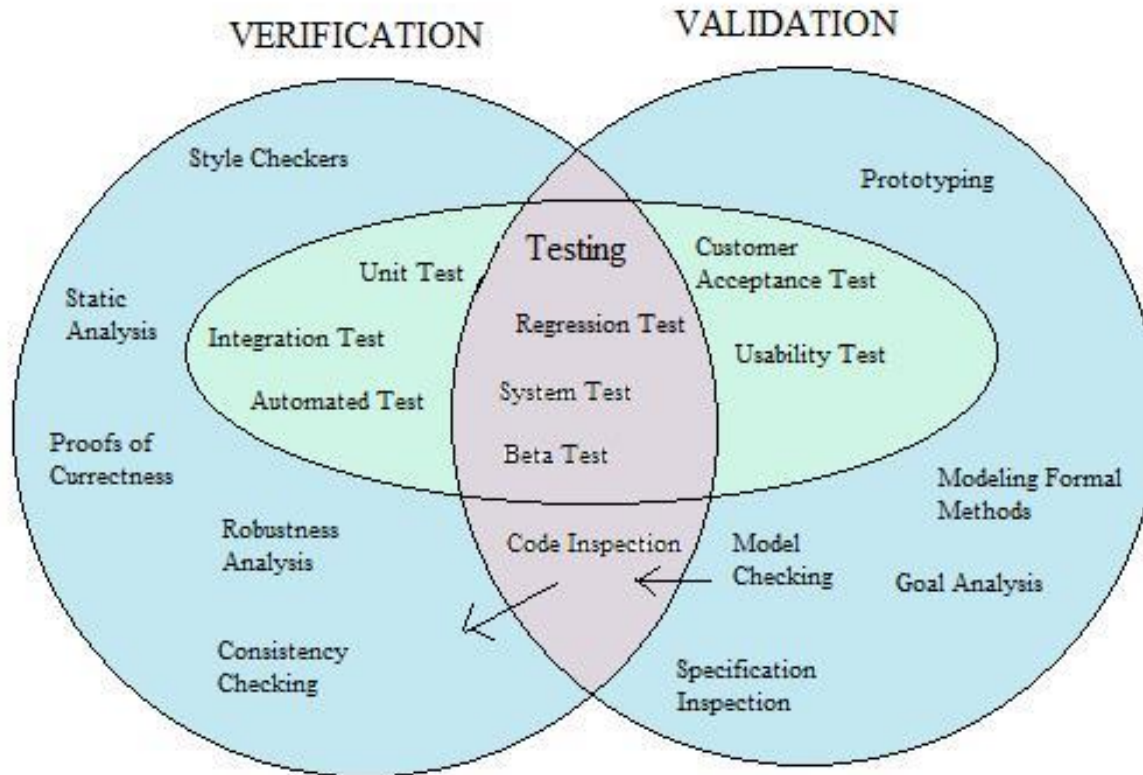
- examinarea unor documente (specificații, modele conceptuale, diagrame de clase, cod sursă, planuri de testare, documentații de utilizare);
- include activități de inspectare a codului, analiza algoritmului, demonstrarea corectitudinii;
- **nu presupune execuția propriu-zisă a programului dezvoltat;**

## Analiza dinamică

- examinarea comportamentului programului cu scopul de a evidenția defecțiuni posibile;
- exemple: tipuri de testare (de regresie, funcțională, non-funcțională), niveluri de testare (testare unitară, testare de integrare, testare de sistem, testare funcțională, testare de acceptare);
- **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 și Validare



sursa: [Pal2013]

# TESTARE

---

False definiții ale testării. Definiții ale testării

Principii de testare. Axiome de testare

Program. Caz de testare. Tipuri de testare

# False definiții ale testării

- “Testarea este procesul prin care se demonstrează că nu există bug-uri.”
  - *“Testing can only reveal the presence of errors, never their absence.”* [[Dijkstra1969](#)]
- “Scopul testării este să arate că programul funcționează corect.”
  - Testarea arată că programul funcționează corect sau nu doar pentru datele de test utilizate.
- “Procesul prin care se asigură certitudinea că un program are funcționalitatea pentru care a fost dezvoltat.”
  - Testarea nu poate garanta un anumit comportament al programului. Testarea surprinde comportamentul programului într-un anumit context.



# 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. *investigare tehnică și empirică realizată cu scopul de a oferi beneficiarilor testării informații referitoare la programul testat* [[BBST2010](#)].
- Testarea este un proces
    - distructiv;
    - care se finalizează cu succes (**passed**) sau eșec (**failed**).

# Principii de testare [\[Myers2004\]](#)

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.

# Axiome ale testării [\[Patton2005\]](#)

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. Cu cât sunt descoperite mai multe bug-uri, cu atât mai multe au rămas de identificat.
5. Paradoxul testării cu cât un program este testat mai mult, cu atât crește imunitatea la testare.
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.

# 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.

# Caz de testare. Definiție. Caracteristici

- **caz de testare** (*engl. test case*) –
  - **notație:**  $(i, r)$ ,  $i \in D$ ,  $r \in R$ ;
    - pentru intrarea  $i$  se așteaptă să se obțină rezultatul  $r$ ;
  - **definiție:**
    - 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](#)];
  - **caracteristici** ale unui caz de testare care are efectul așteptat, i.e., **caz de testare eficace**:
    - probabilitate mare de a identifica bug-uri;
    - nu este redundant;
    - relevant în cadrul categoriei din care face parte;
    - nu este prea simplu;
    - nu este prea complex.

# Tipuri de testare. Definiții

- **testare exhaustivă** (*engl. exhaustive testing, complete testing*; **testare completă**):
  - 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 aleg 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.

# PROCESUL DE TESTARE

---

Întrebări fundamentale

Activități ale procesului de testare

Reguli de raportare a unui bug

Ciclul de viață al unui bug

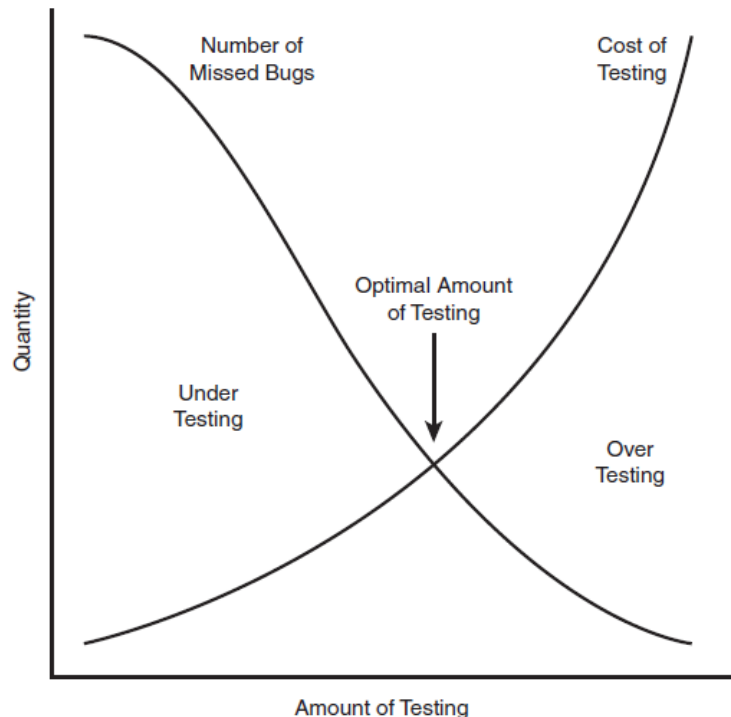
# Întrebări fundamentale

- **De ce este necesar sa testăm un produs soft ? Care este scopul testării ?**
  - evaluarea unor caracteristicile sau attribute are calității produsului soft;
  - cunoașterea unor informații referitoare la produsul soft [[BBST2010](#)];
- **Cum se organizează procesul de testare ?**
  - **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** în care se desfășoară testarea;
  - **tehnică de testare** = metodă de proiectare, implementare și interpretare a rezultatelor unui test;
  - **context** = particularități de realizare a testării, e.g., componenta testată, beneficiarul testării, diverse constrângeri, etc;
- **Cum considerăm că testarea efectuată este suficientă ? Cât timp testăm, câte teste executăm ?**
  - criterii de terminare a testării (*engl. exit criteria*);



# Întrebări fundamentale

- **Exemplu. Criteriu de terminare a testării bazate pe risc:**
- Relația dintre numărul de teste executate, i.e., **amount of testing**, și numărul de bug-uri identificate, i.e., **quantity**;
- Dacă se testează tot: costurile cresc, numărul de bug-uri scade
  - raportul  $\frac{\text{bugs found}}{\text{testing costs}}$  devine mic ==> **eficiența testării scade**;
- 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 cost de testare optim.**



# Întrebări fundamentale

- **Cum se alege tehnica de testare potrivită ?**

- Fiecare tehnică de testare permite proiectarea un teste trebuie să aibă câteva caracteristici esențiale [[BBST2010](#)], printre care:

power	representative	performable	easy to evaluate	affordable
valid	non-redundant	maintainable	support troubleshooting	opportunity cost
value	motivating	information value	appropriately complex	
credible	reusable	coverage	accountable	

- **Tehnica de testare aleasă va permite atingerea obiectivelor de testare propuse.**

# 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*);

## 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:** 1 .. 4;
- **activități QA:** 1, 5;

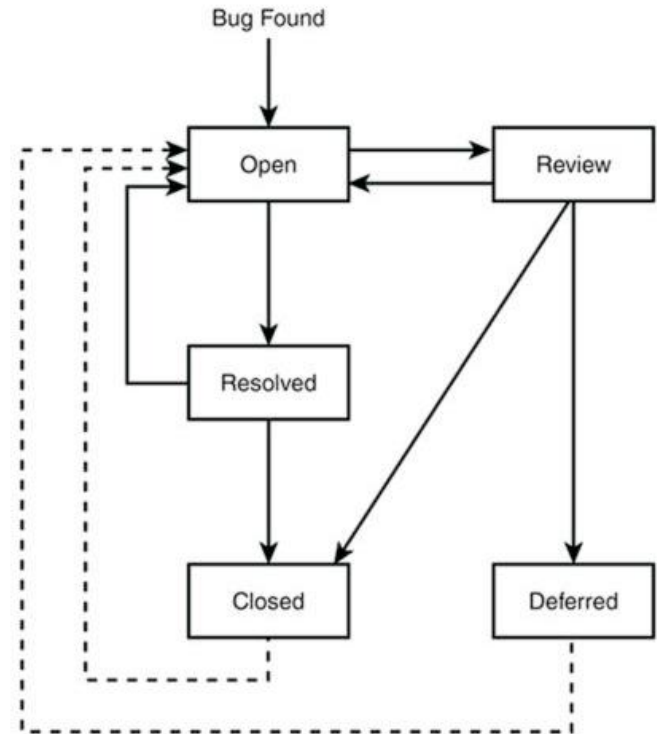
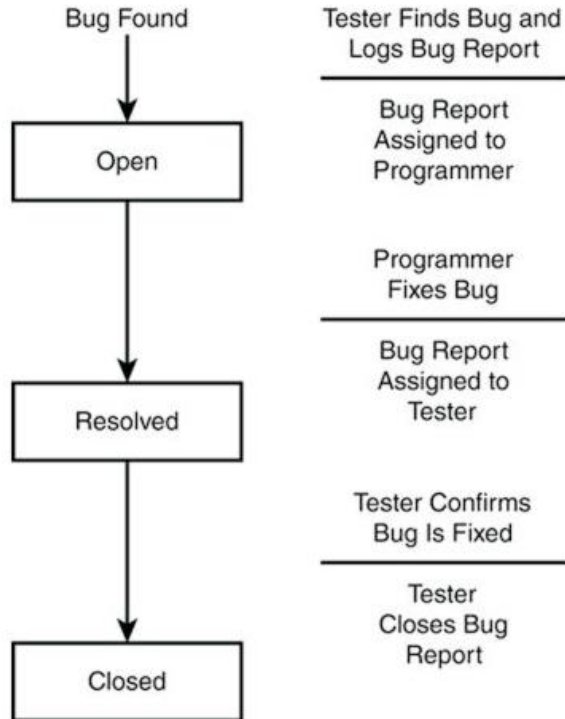
# Reguli de raportare a unui bug (1)

- Reguli de raportare a unui bug [[Patton2005](#)]:
  - se realizează o descriere a bug-ului;
  - se raportează imediat după identificare;
  - nu se fac aprecieri subiective referitoare la bug-ul raportat;
  - se urmărește starea bug-urilor (corectat sau nu) raportate anterior, i.e., se folosește un sistem de monitorizare a bugurilor (*engl.* **bug tracking system**);

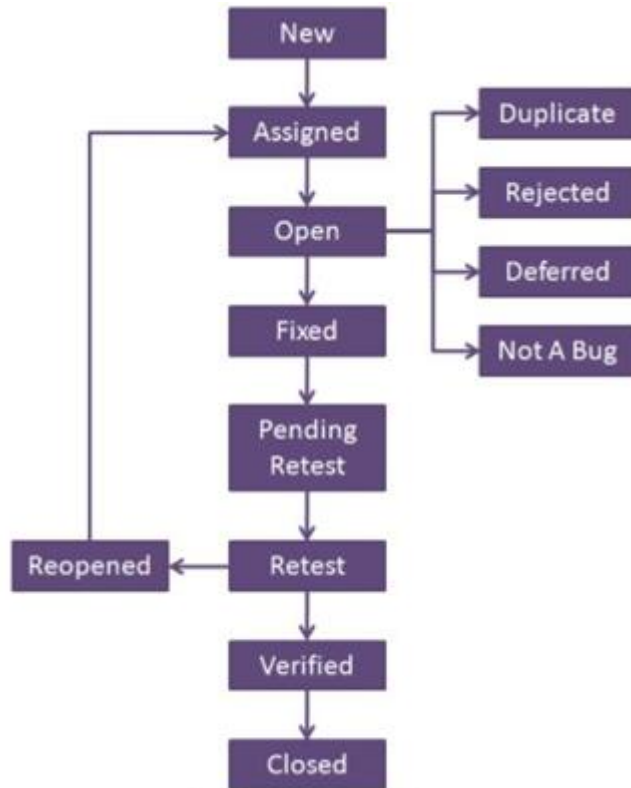
# Reguli de raportare a unui bug (2)

- Izolarea și reproducerea bug-urilor [[Patton2005](#)]. Recomandări:
  - *fii suspicios* – nu te baza pe ceea ce au făcut sau spus alții, fii consecvent și riguros;
  - *acordă atenție timpului* – nu ignora durata de realizare a unei operații (e.g., momentul zilei, utilizarea unui device care lucrează încet, viteza de prelucrare a datelor, etc.);
  - *acordă atenție domeniilor de valori* – valori limită, volum de date mare, alocare și accesare a memoriei;
  - *urmărește secvența de pași executată* – poate fi mascată de încheierea aparent cu succes a unei operații; un bug poate fi evidențiat de execuția într-o anumită ordine a pașilor de execuție și nu de momentul în care a apărut;
  - *identifică dependențele și interacțiunile* - dependențele existente între resurse utilizate și interacțiunea cu memoria, partajarea rețelei și a componentelor hard pot indica inconsistențe sau incompatibilități;
  - *nu ignora componentele hard* – componentele hard trebuie analizate, acestea se pot degrada și pot funcționa imprevizibil.

# Ciclul de viață al unui bug (1) [\[Patton2005\]](#)



# Ciclul de viață al unui bug (2) [\[ISTQBCertification2017\]](#)



Bug / Defect Lifecycle

- stări ale unui bug stabilite de **tester**:
  - **New, Pending Testing, Retest, Reopened, Verified, Closed;**
- stări ale unui bug stabilite de **programator**:
  - **Assigned, Open (Duplicate, Rejected, Deferred, Not a Bug), Fixed;**

# 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.
- **[ISTQBCertification2017]** ISTQB Exam Certification, <http://istqbexamcertification.com/what-is-a-defect-life-cycle/>.