CURS 02A. TESTARE

Verificarea şi validarea sistemelor soft
[03 Martie 2020]

Lector dr. Camelia Chisăliţă-Creţu
Universitatea Babeş-Bolyai

Conţinut

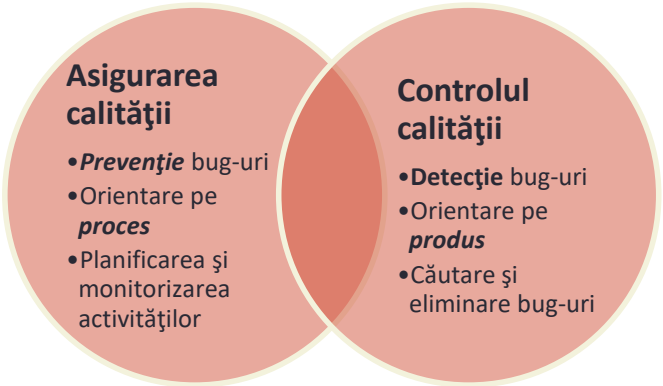
- Evaluarea calității unui produs soft
  - Activităţi asociate calităţii
  - Controlul calităţii. Activităţi asociate
  - Metode de verificare și validare
- Testare
  - Program. Program testat
  - False definiții ale testării. Definiții ale testării
  - Caz de testare. Definiţii. Caracteristici
  - Tipuri de testare
  - Principii de testare. Axiome 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

Activităţi asociate calităţii
Controlul calităţii. Activităţi asociate
Metode de verificare și validare

Activităţi asociate calităţii

- ***în procesul de dezvoltare, calitatea este abordată din perspectiva:***
  - **procesului** ===> **asigurarea calităţii** (*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 calităţii** (*engl.* **quality control**):
    - **Obiectiv**: identifică deficienţele în produsul obținut;
    - **Întrebare**: Cum se controlează calitatea rezultatelor obținute (e.g., work products) în urma activităților desfășurate?



Controlul calității. Activităţi 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, funcţională, non-funcţională), *niveluri de testare* (testare unitară, testare de integrare, testare de sistem, testare funcţională, testare de acceptare);
- se bazează întotdeauna pe execuţia programului.

Controlul calității. Activităţi asociate (2)

Analiză statică (static testing)

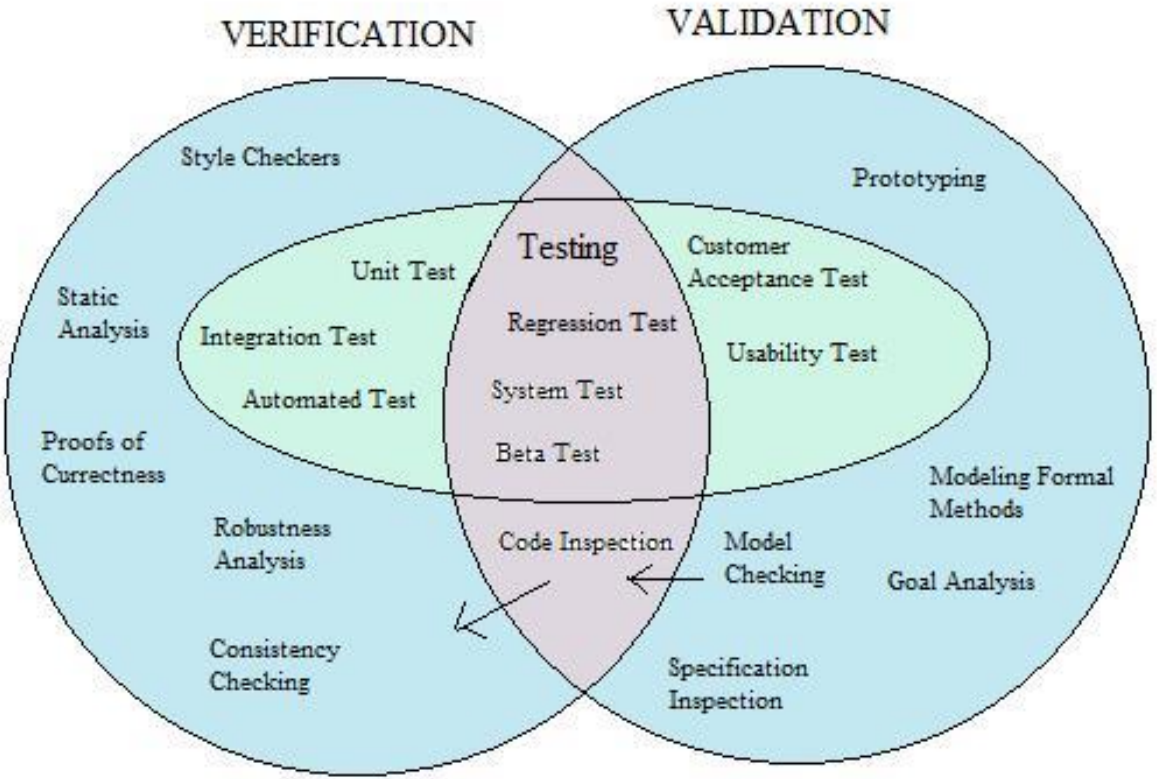
- **permit identificarea mai multor erori (greşeli) care pot fi corectate simultan;**
- **NU presupune execuția propriu-zisă a programului dezvoltat;**

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 evidenţieze o defecţiune doar în anumite situaţii.

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

Metode de Verificare şi Validare



sursa: [Pal2013]

TESTARE

Program. Program testat
False definiții ale testării. Definiții ale testării
Caz de testare. Definiţii. Caracteristici
Tipuri de testare
Principii de testare. Axiome de testare

Program. Definiţie

- **program** (engl. **computer program, software application, software product**):
  - listă de instrucţiuni sau o mulţime de metode sau module care permit execuţia de către un calculator;

- **un program** este
  - o **comunicare**
  - între **persoane** şi **calculatoare**
  - care sunt **separate** în timp şi spaţiu
  - şi conţine **instrucţiuni** care sunt **executate** de către calculator. [BBST2010]

Program testat. Definiţie

- **program testat** (*engl.* **software under test, SUT**):
  - ≈ funcţie matematică;
  - P : D --> R, unde
    - D – mulţimea datelor de intrare;
    - R – mulţimea datelor de ieşire aşteptate.

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 funcţionează corect sau nu doar pentru datele de test utilizate, într-un anumit context şi moment.
- “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 suprinde 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
  - destructiv;
  - se poate finaliza cu succes (passed) sau eşec (failed).

Caz de testare. Definiţie

- **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 informaţional, i.e., **informaţia pe care o descoperim prin testare**, e.g., testul este *passed* sau *failed*, timpul de execuţie asociat testului este foarte mare.
  - **notație:** (i , r), i ∈ D, r ∈ R;
    - pentru intrarea ***i*** se așteaptă să se obțină rezultatul ***r***.

Caz de testare. Caracteristici

- **caracteristici** ale unui caz de testare care are efectul așteptat [BBST2011], 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.

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	

Tipuri de testare. Definiţii

- **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 ϵ 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 ϵ S**, **S⊂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]

1. Defineşte rezultatele aşteptate în urma testării.
2. Evită să testezi programelor proprii.
3. Analizează riguros rezultatele fiecărui test.
4. Scrie cazuri de testare atât pentru condiţii de intrare valide cât şi pentru cele non-valide.
5. Testează dacă programul **nu** face ceea ce se precizează în specificaţie, dar şi dacă ceea ce face programul **nu** este descris în specificaţii.
6. Păstrează întotdeauna cazurile de testare.
7. Organizează şi planifică procesul de testare, considerând că se vor identifica bug-uri.
8. Testarea este o activitate de stimulare a creativităţii.
   - ***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]

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 funcţionalităţi este un indicator al prezenţei altor bug-uri – bug-urile pot fi grupate în anumite funcţionalităţi, nu sunt izolate.
5. Paradoxul pesticidului (în testare): cu cât un program este testat mai mult folosind aceleaşi teste (tehnici de testare), imunitatea la testare creşte (nu se descoperă bug-uri noi).
6. Nu orice bug identificat va fi eliminat.
7. Specificaţia produsului soft se schimbă în permanenţă.
8. Testerii nu sunt cei mai apreciaţi membri ai echipei de dezvoltare.

PROCESUL DE TESTARE

Întrebări fundamentale
Activități ale procesului de testare
Reguli de raportare a unui bug
Ciclul de viață al unui bug

Întrebări fundamentale (1)

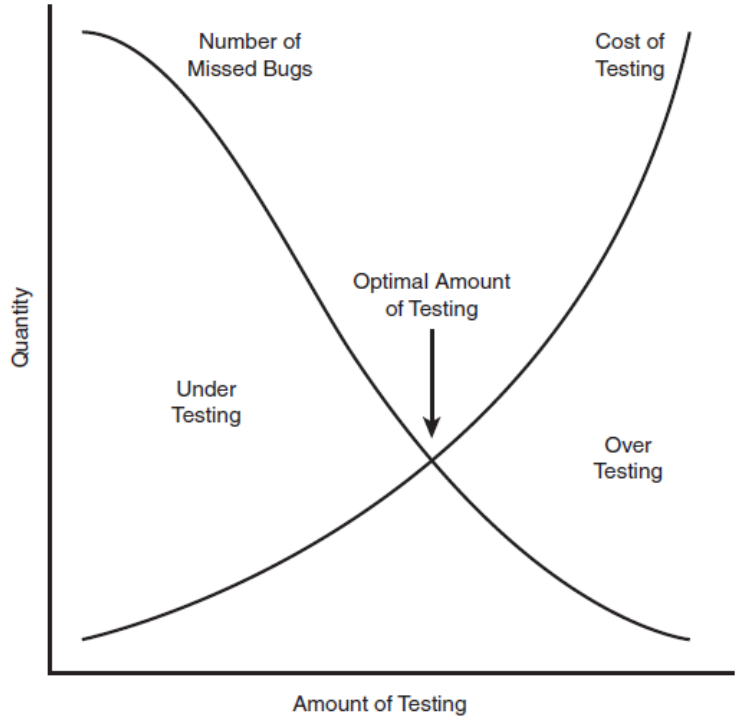
- **De ce este necesar sa testăm un produs soft ? Care este scopul testării ?**
  - evaluarea unor caracteristici sau atribute are calităţii produsului soft;
  - descoperirea unor **informaţii** referitoare la produsul soft [BBST2010];
  - **obiective ale testării** = aspectele de interes care vor fi investigate (evaluate) în procesul de testare;
- **Cum se organizează procesul de testare ?**
  - **contextul aplicaţiei** = particularităţi de realizare a testării, e.g., componenta testată, beneficiarul testării (**stakeholder**), diverse **constrângeri**, etc;
  - **misiunea testării** = acţiunea desfăşurată 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 îşi atingă obiectivele informaţionale, luând în considerare **contextul aplicaţiei** în care se desfăşoară testarea;
  - **tehnică de testare** = *metodă de proiectare, implementare şi 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 condiţiile 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 relaţia dintre numărul de teste executate, i.e., **amount of testing**, şi numărul de bug-uri identificate, i.e., **quantity**;
- **Over testing:**
- Dacă se testează tot: costurile cresc, numărul de bug-uri scade
  - raportul $\frac{bugs\ found}{testing\ costs}$ devine mic ===> **eficienţa testării scade;**
- **Under testing:**
- Dacă se testează puţin sau se iau decizii nepotrivite legate de **CE** se va testa: costurile sunt mici, numărul de bug-uri rămâne ridicat
  - raportul $\frac{bugs\ found}{testing\ costs}$ rămâne mare ===> **calitate redusă;**
- **Fiecare proiect soft are un cost de testare optim.**



Atributele unui caz de testare

- **Cum se alege tehnica de testare potrivită ?**
  - Fiecare tehnică de testare permite proiectarea unor teste care au câteva caracteristici esenţiale [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*** **folosind** ***strategia de testare*** **propusă.**

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;

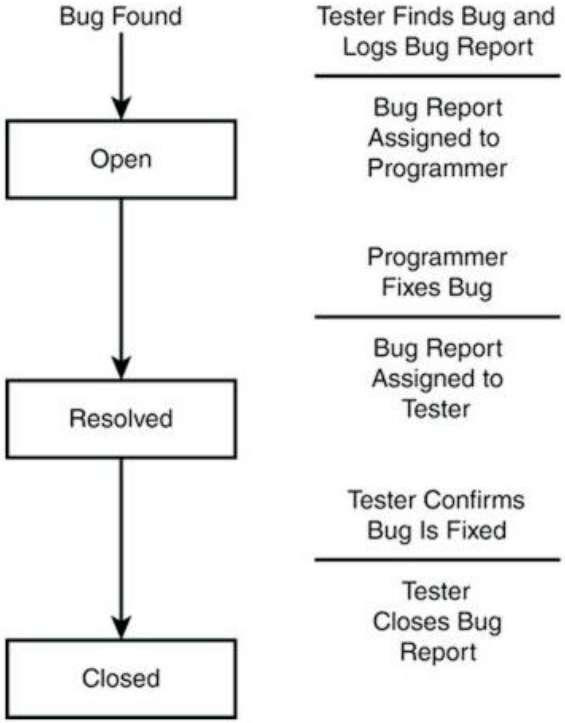
Reguli de raportare a unui bug (1)

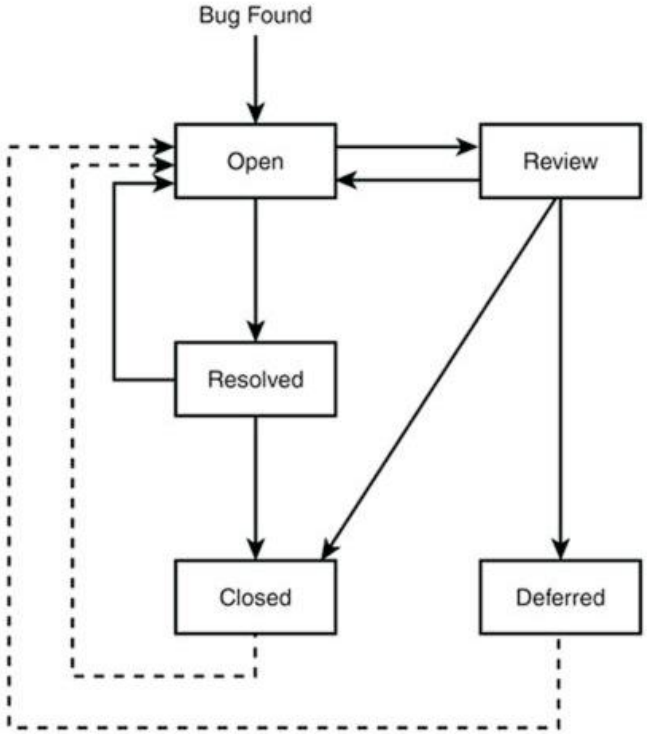
- **Reguli de raportare a unui bug [Patton2005]:**
  - se realizează o descriere a bug-ului;
  - se raportează imediat după identificare;
  - nu se fac aprecieri subiective referitoare la bug-ul raportat;
  - se urmărește starea bug-urilor (corectat sau nu) raportate anterior, i.e., se folosește un sistem de monitorizare a bugurilor (*engl.* **bug tracking system**);
    - atribut gestionat de tester: **severitate** *(critical, non-critical)*;
    - atribut gestionat de programator: **prioritate la depanare** *(low, medium, high, urgent).*

Reguli de raportare a unui bug (2)

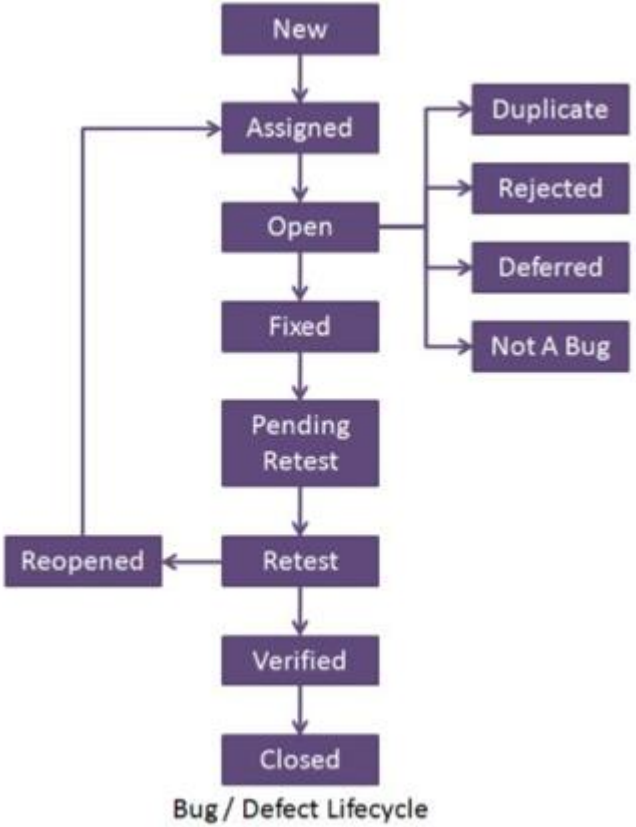
- **Izolarea și reproducerea bug-urilor [Patton2005]. Recomandări:**
  - *fii suspicios* – nu te baza pe ceea ce au făcut sau spus alții, fii consecvent și riguros;
  - *acordă atenție timpului de execuţie* – nu ignora durata de realizare a unei operații (e.g., momentul zilei, utilizarea unui device 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;
  - *urmăreşte secvenţa de paşi 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 dependenţele şi interacţiunile* - dependenţele existente între resurse utilizatele și interacțiunea cu memoria, partajarea rețelei și a componentelor hard pot indica inconsistenţe sau incompatibilităţi;
  - *nu ignora componentele hard* – componentele hard trebuie analizate, acestea se pot degrada și pot funcţiona imprevizibil.

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





Ciclul de viață al unui bug (2) [ISTQBCertification2020]



Bug / Defect Lifecycle

- stări ale unui bug stabilite de **tester**:
  - **New, Pending Retesting, Retest, Reopened, Verified, Closed;**
- stări ale unui bug stabilite de **programator**:
  - **Assigned, Open (Duplicate, Rejected, Deffered, 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.
- **[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