Modulul 2:

Strategii de testare.

Termenul de strategie:

- **STRATEGIE** - Parte componentă a artei militare, care se ocupă cu problemele pregătirii, planificării și ducerii războiului și operațiilor militare. (DEX)
- **STRATEGIA DE TESTARE -** descrie modul în care riscurile sunt atenuate la nivel de testare, care tipuri de teste trebuie să fie efectuate, care sunt criteriile de incepere si finisare a testarii. Acestea sunt create pe baza unor documente de proiectare. Pentru fiecare etapă de proiectare se creaza o strategie de testare corespunzătoare pentru a testa noile seturi de caracteristici.

Clasificari ale strategiilor:

- Dupa obiectul testarii;
- Dupa cunoasterea structurii interne a produsului;
- Dupa nivelul de automatizare;
- Dupa gradul de izolare a componentelor;
- Dupa perioada de testare;
- Dupa caracterul pozitiv al scenariilor;
- Dupa nivelul de pregatire pentru testare;

Dupa obiectul testarii:

- Testarea functionalitatii (functional testing)
- Testarea performantei (performance testing)
  - Testarea incarcarii - lansarii(load testing)
  - Testarea de stres (stress testing)
  - Testarea stabilitatii (stability / endurance / soak testing)
- Testarea utilizabilităţii (usability testing)
- Testarea interfetei (UI testing)
- Testarea securitatii/sigurantei (security testing)
- Testarea localizarii (localization testing)
- Testarea compatibilitatii (compatibility testing)

Testarea functionalitatii (functional testing):

- Verificarea functionalitatii unui produs se poate face atât prin metode statice cât si dinamice. Cele statice se refera la verificarea documentelor de cerinte , de analiza si respectiv documentele tehnice.
- Dupa verificarea tuturor documentelor se poate spune ca ceea ce a intrat ca *input* în etapa de dezvoltare este corect, si se trece efectiv la metoda dinamic a de testare a functionalitatii. Scopul final al test arii functionalitatii este de a confirma încrederea ca sistemul software este exact ceea ce s-a cerut (*fit for purpose*). Aceasta presupune ca sistemul sa fie complet fara defecte, cu alte cuvinte, sistemul trebuie sa fie bun de utilizat.
- În timpul verificarii si validarii, defectele descoperite trebuie rezolvate, ceea ce conduce la modificarea programului iar sistemul trebuie reverificat si revalidat. Toate acestea se fac într-un ciclu repetitiv numit *regression test.*

Testarea functionalitatii (functional testing):

- Testarea functionalitatii este cunoscuta si sub denumirea de *black - box testing* deoarece se testeaza comportamentul cu datele de intrare si rezultatul lor la iesire. Pe lânga incorectitudinea sistemului sau lipsa unor functionalitati, în timpul testarii operationalitatii se pot descoperi si erori de interfata, erori de date sau de performanta, care vor fi contorizate si plasate la testele corespunzatoare atributelor operationalitatii.
- Erorile descoperite pe parcursul testarii pot fi cauza diferitelor faze ale dezvoltarii unui produs si pot fi descoperite în:
  - specificatie - erori în documentele de specificatie primite.
  - proiectare - erori generale sau particulare în solutiile de proiectare adoptate;
  - dezvoltare - erori în scrierea efectiva a codului;
  - altele (configurare, etc.)

Testarea functionalitatii (functional testing):

- O mare parte a erorilor sunt în specificatie , deoarece aceasta poate fi insuficienta (nu s-au atins anumite puncte), se schimba continuu sau se datoreaza unei comunicari insuficiente cu echipa de dezvoltare. Dupa erorile de specificatie pot urma, din punct de vedere al numarului, atât scrierea de cod cât si de proiectare, aceasta depinzând de complexitatea proiectului, de tipul de proiect si de echipa cu care se realizeaza produsul.
- Validarea este terminata când software-ul functioneaza într-o maniera rezonabila acceptata de client. Acceptarile rezonabile sunt definite în documentul ce cuprinde *specificatia cerintelor,* în care se descriu toate atributele cerute de client. Specificatiile contin o sectiune numita *criterii de validare* care reprezinta baza testului de validitate.

Testarea performantei (performance testing):

- efectuate pentru a determina cât de repede un sistem sau o parte din el realizeaza o anumită sarcină in careva conditii.

Se cunosc 3 directii ale testarii performantei:

- Testarea incarcarii – lansarii – supraincarcarii - presupune testarea in careva cazuri speciale: numar mare useri, tranzactii cu frecventa mare.
- Testarea de stres -Presupune execuţia sistemului într-o manieră anormală. Adică se testează confruntarea software cu situaţi anormale (multiple tranzacţii, memorie insuficientă, spaţiu liber mic pe disk, blocarea perifericelor cu care lucrează aplicaţia)
- Testarea stabilitatii - presupune verificarea timpului in care sistemul va rezista intr-o careva incarcatura.

Testarea utilizabilităţii (usability testing):

- Studiu care are ca scop verificarea utilizabilitatii interfetei unei careva aplicatii.
- Rezultatele sunt deduse din parerile utilizatorilor ce au rolul testerilor.
- **Cum anume???????**
  - Se filmeaza mimica utilizatorului;
  - Se filmeaza ecranul ;
  - Se diferite actiuni legate de utilizator:
    - Miscarile mouse-ului;
    - Apasarea pe taste;

Testarea interfetei (UI testing):

- Presupune verificarea aplicatiei daca coincide cu standartele cerute, daca coincid stilurile, etc.
  - Verificarea proportiei;
  - Verificarea in cazul redimensionarii ferestrei;
  - Compatibilitatea cu diferite browsere;

Testarea securitatii/sigurantei (security testing):

- Siguranta unui sistem presupune atât captarea, transferul si retinerea datelor în conditii de siguranta cât si securitatea sistemului în ansamblu. Securitatea sistemului este data în special de domeniul de activitate al produsului. Astfel, securitatea poate fi de la „foarte mic a” pâna la „foarte mare”.
- Domeniul bancar este un domeniu unde securitatea trebuie sa fie spre nivelul maxim. Trebuie tinut cont ca sporirea securitatii este proportionala cu costurile necesare implementarii acestei securitati. Daca crestem costurile de securitate pentru a fi convinsi ca sistemul final o sa fie sigur, s-ar putea ca sistemul sa nu mai îndeplineasca alt atribut al operationalitatii (cel al eficientei).
- Nivelul la care un sistem este sigur poate fi stabilit si din alt punct de vedere: un sistem este sigur atunci când efortul pentru spargerea sistemului este mai mare decât rezultatele obtinute prin spargere.

Testarea securitatii/sigurantei (security testing):

- O particularitate importanta pentru aceasta testare o au *sistemele critice* , ale caror erori duc la o pierdere economica semnificativa, o deteriorare fizica sau afecteaza sanatatea umana.
- Testarea sigurantei la aceste sisteme este foarte importanta din doua motive:
  - accidentele sunt evenimente rare în sistemele critice si ele practic sunt imposibil de simulat în timpul testarii sistemului;
  - exista cerinte care sunt nesigure si acestea sunt imposibil de demonstrat prin testare sau alte operatii de validare.

Dupa cunoasterea structurii interne a produsului:

- *Testarea bazata pe specificatii* – black box(stabilirea claselor de valori, stabilirea limitelor claselor, alcatuirea tabelelor de decizii, diagramei de stari, cazurilor de testare/utilizare)
- *Testarea bazata pe structura* – white box;
- *Testarea mixta* – grey box;

Dupa nivelul de automatizare:

- *Teste automatizate* – presupune utilizarea instrumentelor pentru rularea aplicatiei si verificarea rezultatelor.
  - La nivel de cod – testarea modulara
  - La nivel de interfata – imitarea actiunilor utilizatorului.
- *Teste manuale* – realizarea si verificarea testelor se executa repetat de oameni.
- *Teste semiautomate*

Dupa gradul de izolare a componentelor:

- *Testarea componentelor (component/unit testing)* – verifica fiecare modul aparte cu scopul de a verifica daca modificarile efectuate nu au creat defecte in partea existenta.

- *Testarea integrarii (integration testing)*

- *Testarea de sistem (system/end-to-end testing)* – se realizeaza cind tot sistemul este gata cu scopul de a verifica integritatea si compatibilitatea componentelor. Face parte din testarea specificatiilor si nu cere cunoasterea codului.

Dupa perioada de testare:

- *Testare alfa (alpha testing)* – realizeaza programatorii sau echipa pentru asigurarea calitatii
  - “testarea de fum” (smoke testing) – realizeaza programatorii pina sa fie transmisa pentru testare, se verifica instalarea, conexiunile cu BD, etc.
  - Testarea noii functionalitati (new feature testing) – se verifica functionalitatea noua adaugata.
  - Testarea pentru confirmare (confirmation testing)
  - Testarea regresiva (regression testing) – dupa ce a fost adaugata o noua functionalitate se verifica si cele vechi.
  - Testarea de acceptanta(acceptance testing)
- *Testare beta (beta testing)* – realizeaza un grup de utilizatori in conditii reale.

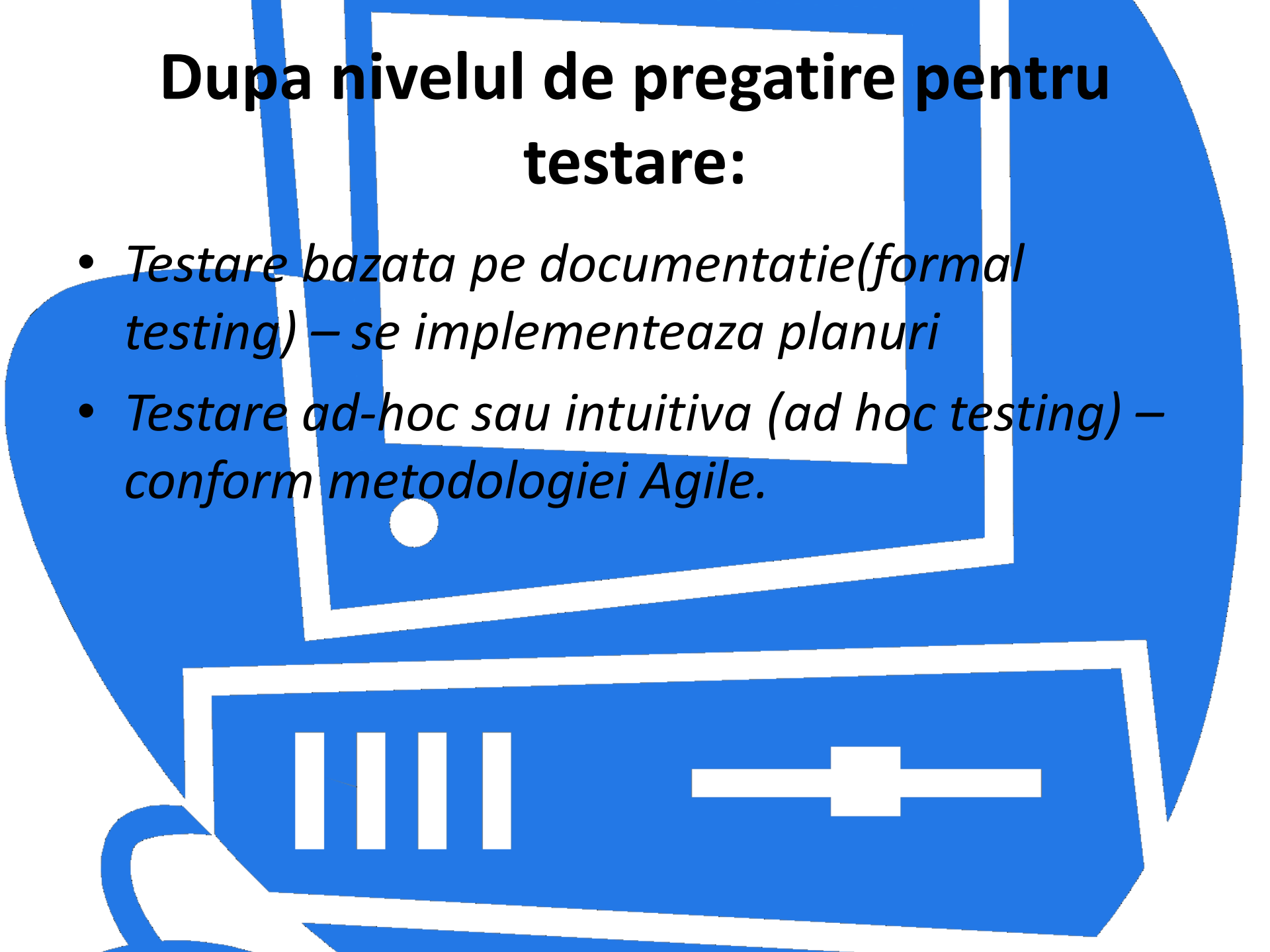
Dupa caracterul pozitiv al scenariilor:

- *Testare pozitiva* – se elaboreaza date si scenarii ce corespund utilizarii corecte a programului. Scopul acestei testari este de a verifica functionalitatea produsului.

- *Testare negativa* – elaborarea de date si scenarii ce nu corespund utilizarii corecte a aplicatiei: provocarea unor mesaje de eroare, situatii extreme etc. Scopul acestei testari este de a vedea rezistenta produsului in asa conditii(date incorecte, tratarea exceptiilor etc)

Dupa nivelul de pregatire pentru testare:

- *Testare bazata pe documentatie(formal testing) – se implementeaza planuri*
- *Testare ad-hoc sau intuitiva (ad hoc testing) – conform metodologiei Agile.*



Indicatori de testare:

- În cadrul etapei de testare se calculează diferiţi indicatori care dau o imagine a procesului de testare cât şi o imagine asupra calităţii sistemului testat.
- Complexitatea testării reprezintă numărul de cazuri de test necesare raportat la volumul aplicaţiei. Complexitatea testării se calculează atât pentru întregul produs dar şi pe o anumită parte sau unitate de produs. Mai exact complexitatea testării **Ct** reprezintă numărul cazurilor de test raportate la unitatea testată dată de relaţia:

$$C_t = \frac{CT}{UT}$$

unde: **CT** – Cazuri de test; **UT** – Unitatea testată;

- Cazurile de test sunt raportate la nu anumit număr de tranzacţii, un modul, un grup de module sau întregul sistem.

Indicatori de testare:

- **Calitatea defectelor raportate** - Raportul dintre defectele efective si totalul defectelor raportate este numită calitatea defectelor raportate Cd dat de relaţia: unde: **TDunice** - totalul defecte unice; **TD** – totalul defectelor raportate.

$$C_d = \frac{TD_{unice}}{TD} * 100$$

- **Rata defectelor** - Numărul de defecte efective descoperite relative la numărul cazurilor de test reprezintă rata defectelor **Rd**.

$$R_d = \frac{TD_{unice}}{CT} * 100$$

- **Rata incidentelor - numărul de incidente în exploatarea sistemului** - Prin incident înţelegem acele erori care apar din exploatarea sistemului cum ar fi: afişarea şi imprimarea rezultatelor, întreruperea programului, blocarea programului, resetări de funcţii şi de stări şi alte comportamente neprevăzute. Acest indicator ne relevă eficienţa măsurilor de întreţinere şi de corectare după implementare a produsului-program. Contorizarea acestor incidente se face pentru a calcula o rată a incidentelor Ri prin raportarea numărului de incidente la numărul de ore de funcţionare sau la numărul de tranzacţii efectuate date de relaţia:

$$R_i = \frac{N_I}{N_T} * 100$$

unde: **NI** – număr de incidente;
NT – număr de tranzacţii;