

# What if ...

---

... than Qa/Testers

More time

Less pressure

More proactive

Less defects

Less frustration



... unless  
Devs are  
perfect!

Developers

More power

More  
responsibility

More focused

Less compromise

Better quality



# Testing is a creative process

---

Software development is a **creative** process

Software testing is a **creative** process



I am **original**  
I have **imagination**  
I have good and bad days  
I am an “**artist**” 😊

I have **experience**  
I can **improve on speed**



# Good & Bad practices

---



- “Go test and later we will teach you how to code.”



Test productivity == # bugs



QA/Tester should not Unit Test (devs should)

# Developers vs QA/Testers

---



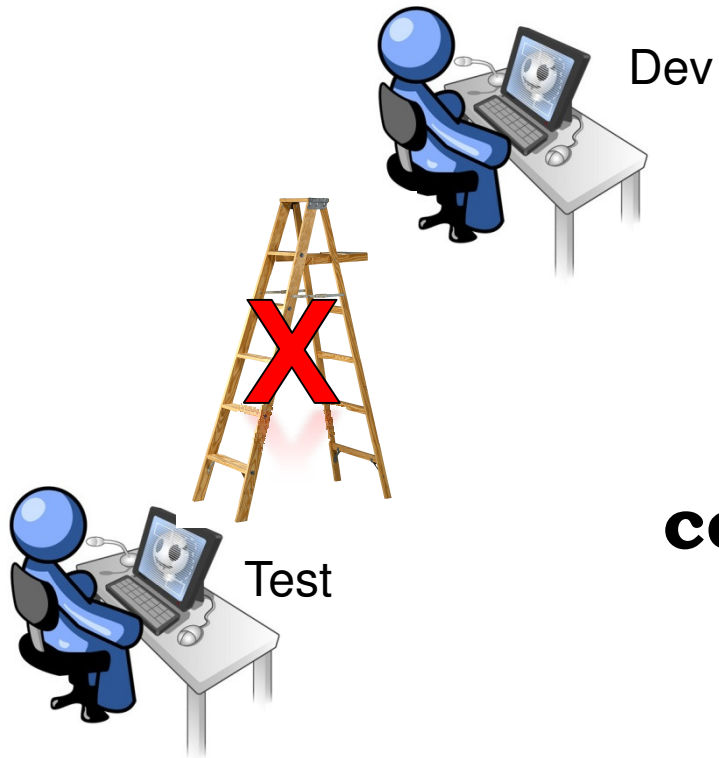
sometimes

some other times



# Interdependență

---



**comunicare**

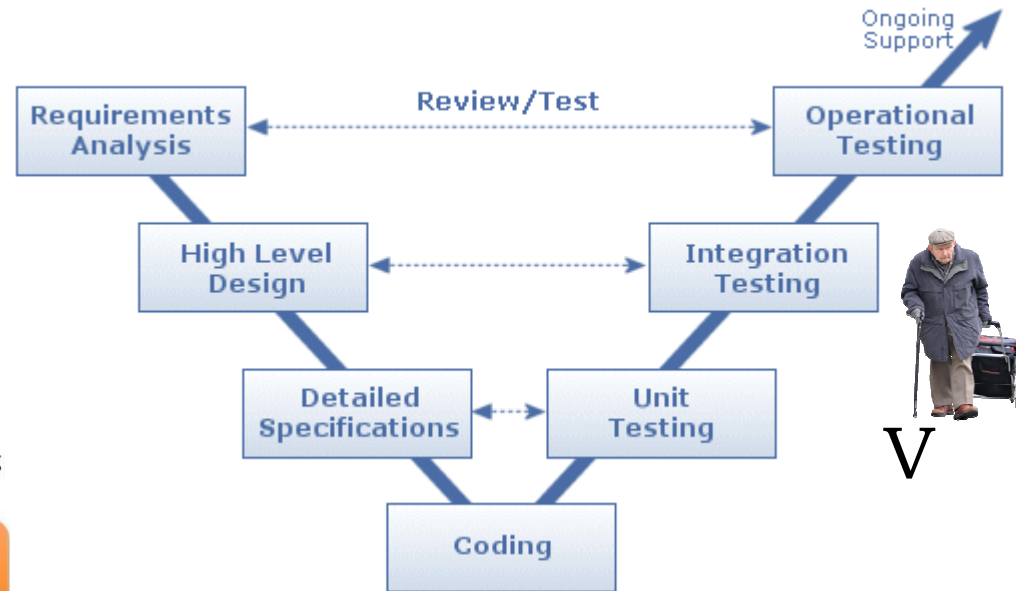
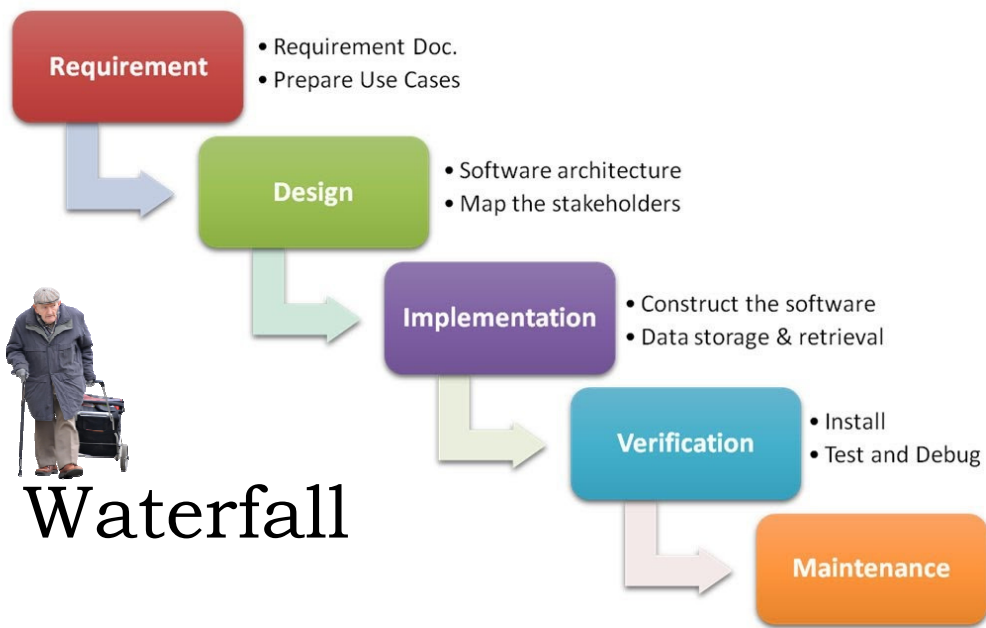


colaborare

echipa

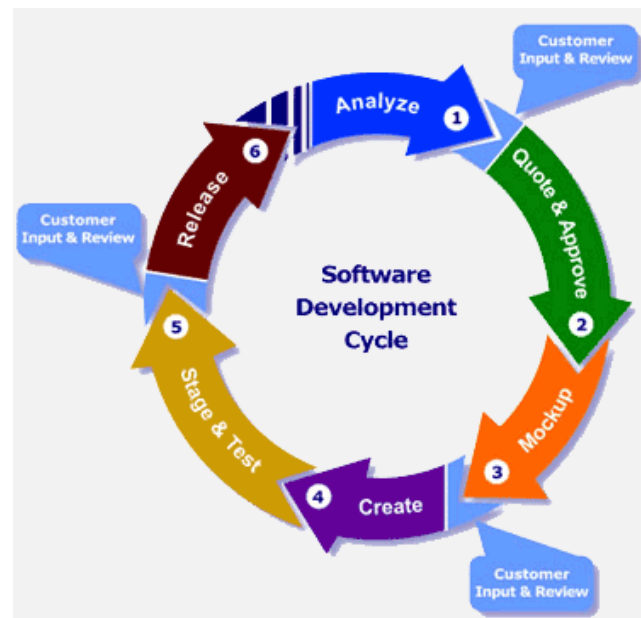


# Software life-cycles



## Agile

Test First vs. Test Last



# Câteva citate

---

*“Given enough eyeballs, all bugs are shallow.”*

Linus Torvalds

*“Testing can be used very efectively to show the presence of bugs but never to show their absence.”*

Dijkstra

*„Software testing is any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results.”*

Hetzel, 1988

*“Software Testing is an empirical investigation conducted to provide stakeholders with information about the quality of the product or service under test.”*

Kaner, 2006

*“Testing is questioning a product in order to evaluate it.”*

Bach

# Dezastre/Accidente pornind de la defecte sw.

---



## **NASA Mariner 1, 1962**

- un amplificator a cedat în timpul lansării -> distrugerea navei
- codificarea incorectă a unei formule FORTRAN,
- citirea neatență a specificațiilor

# Dezastre/Accidente pornind de la defecte sw.

---

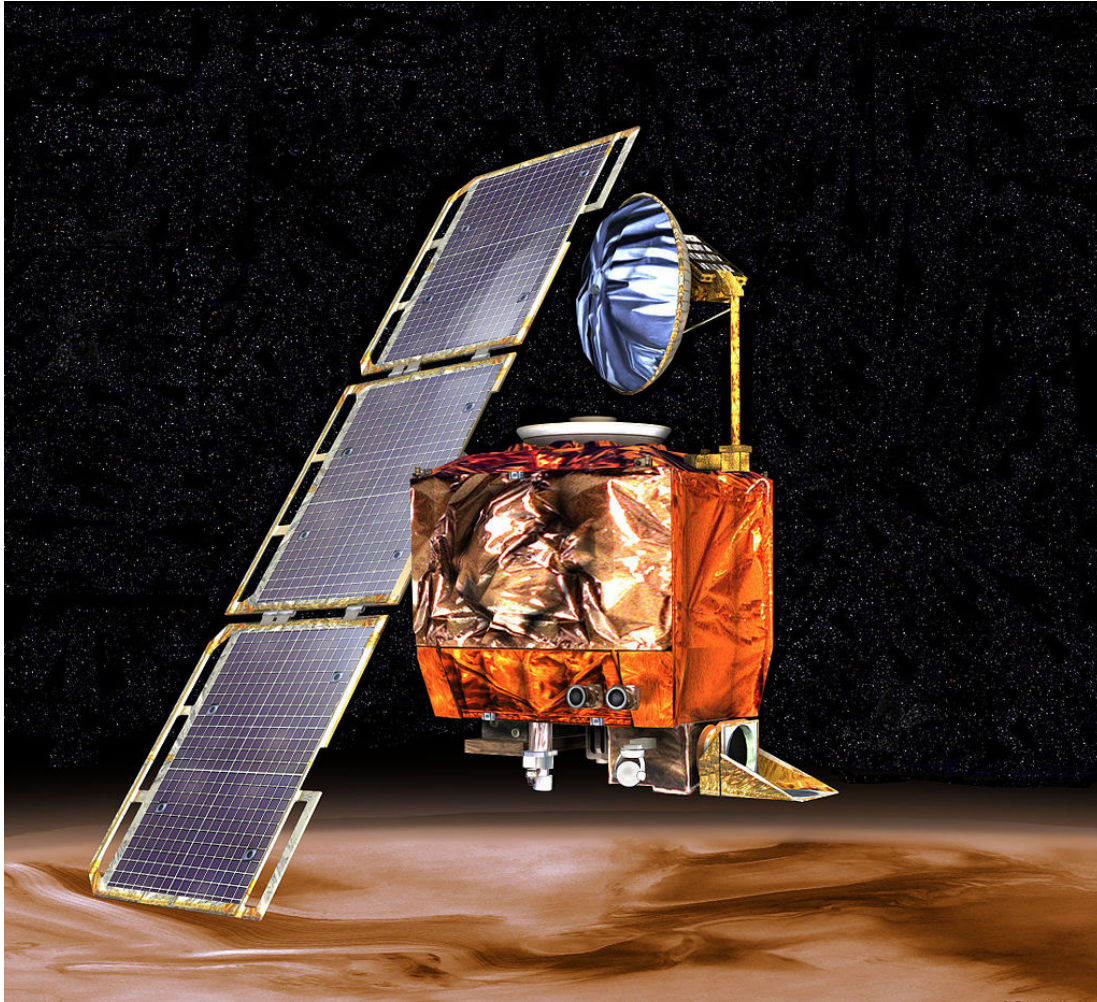


## **Ariane 5, 1996**

- Conversia de la 64-bit float la 16-bit int generează o excepție de depășire netratată -> distrugere
- reutilizarea neatență a codului preluat de la Ariane 4
- cost 7 miliarde \$

# Dezastre/Accidente pornind de la defecte sw.

---

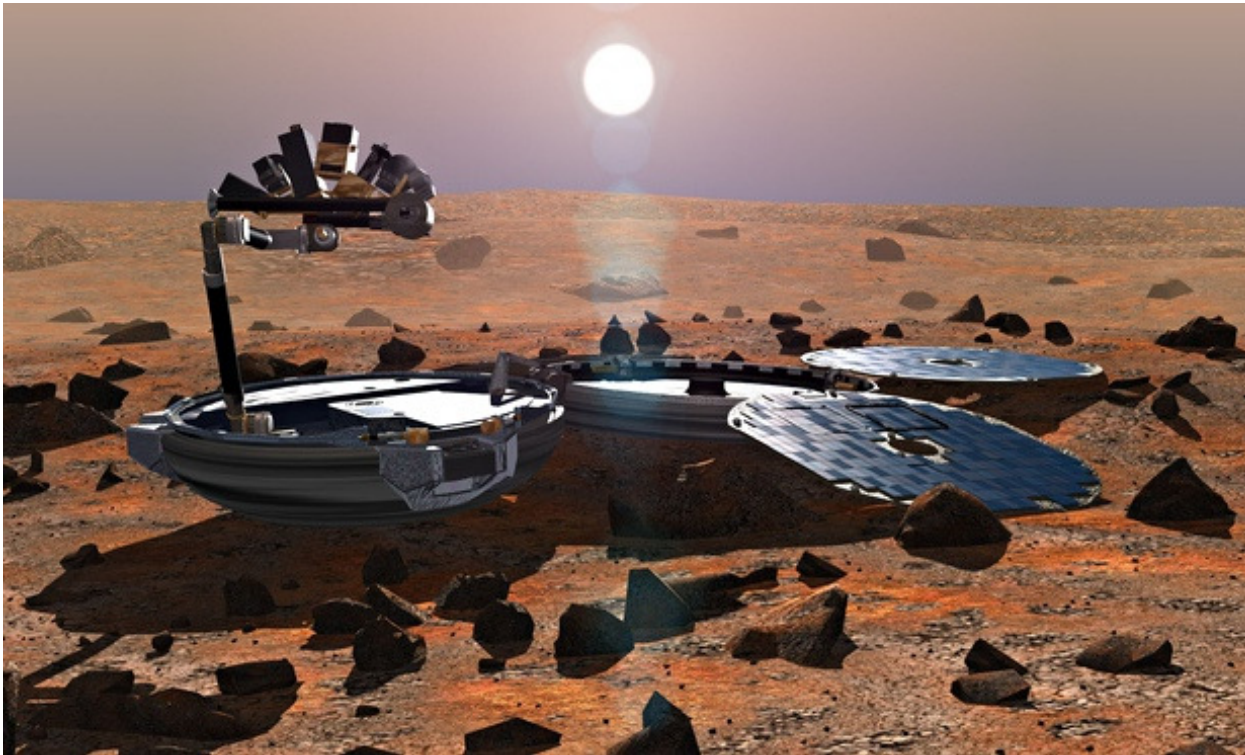


## **Mars Climate Orbiter, 1998**

- discrepanță între unități de măsură în sistemele anglo-american și metric -> dezintegrare
- cost 500 milioane \$

# Dezastre/Accidente pornind de la defecte sw.

---



## **Misunea Beagle 2, 2003**

- testele unitare și de integrare omise în totalitate sau realizate parțial
- lansare fără testarea finală -> s-a pierdut comunicarea datorita unor panouri solaref

# Objective

---

- Descoperirea defectelor
  - Verificarea îndeplinirii specificațiilor
  - Verificarea și validarea produsului software
  - Asigurarea calității
  - Luarea deciziilor de a da drumul pe piață produsului software
- 
- Minimizarea riscurilor de comercializare
  - Reducerea pierderilor ulterioare
  - Predicție asupra costurilor de suport tehnic
  - Procesul de testare oferă siguranță clientului



# Provocări

---

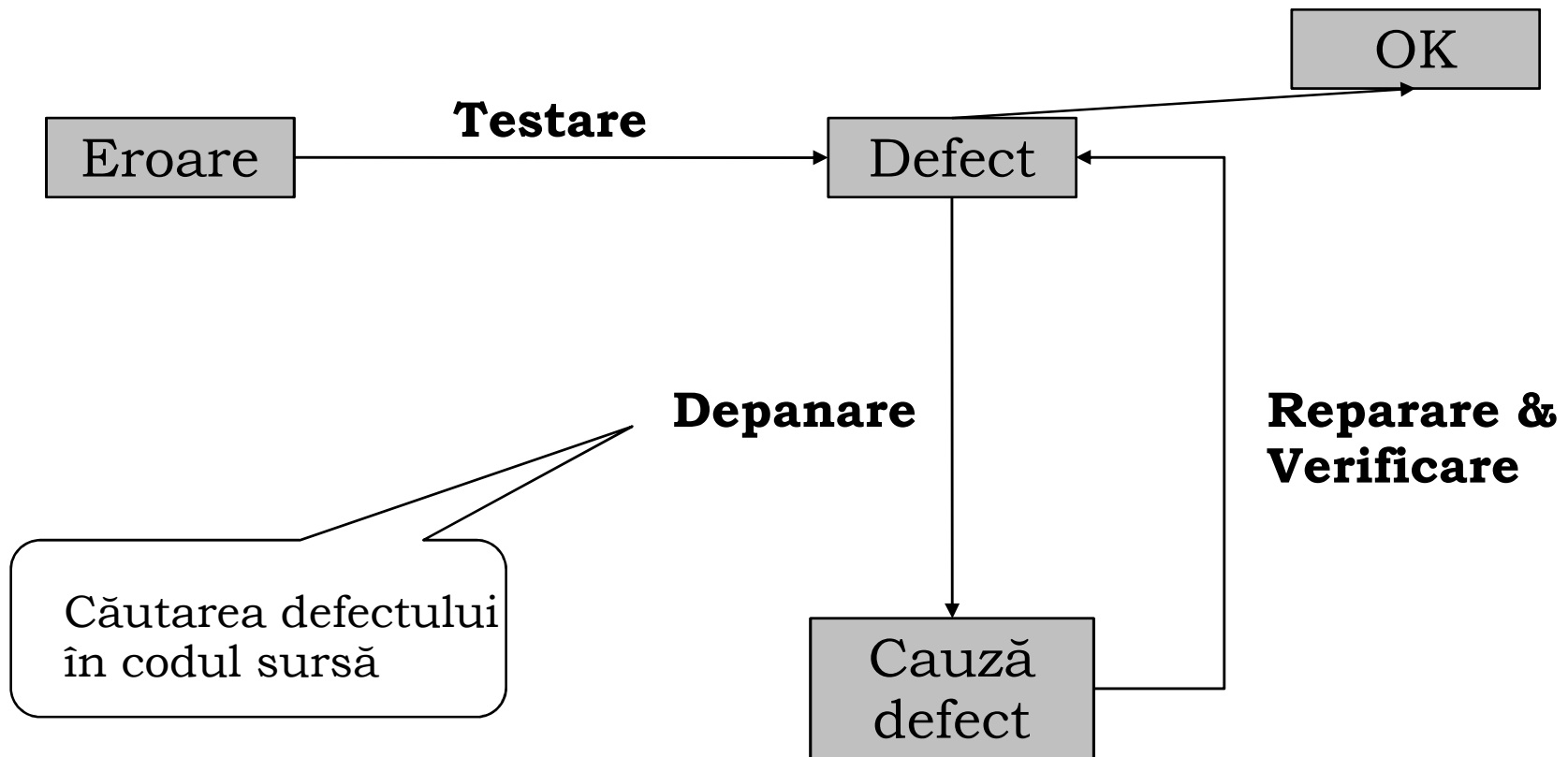
- Timp insuficient pentru procesul de testare
- Prea multe combinații de testat
- Lipsa cursurilor specializate
- Lipsa unui proces de testare
- Lipsa instrumentelor software dedicate
- Lipsa specificațiilor clare/ modificarea acestora
- Managementul nu înțelege necesitatea procesului de testare – nu alocă resurse
- Testarea = factor necesar (nu și suficient) în procesul de asigurare a calității  
(realizarea specificațiilor, comunicarea cu clientul, managementul proiectului, ...)

calitate = „valoare pentru o anumită persoană” [Gerald Weinberg]



# Testare vs. Depanare

---



- › unit testing – reduce timpul de depanare
- › test first

# Întrebări esențiale [Kaner]

---

Cine

Ce

Strategie



Cum

De ce

Când se consideră FINALIZAT procesul de testare?

# CE se testează?

---



- Funcționalitatea aplicației/componentei <- specificații
- Comportamentul la rulare concomitentă cu alte aplicații/dependințe
- Comportamentul în diferite configurații hardware
- Orice posibile influențe exterioare (sistem de operare, utilizarea procesorului, procese concurente,...)
- Lucrul cu memoria

# Rolul echipei de testare

---

- Atenție la specificații și la detalii
- Respectarea procedurilor de testare aferente produsului în cauză
- Realizarea cazurilor de test
- Detecția defectelor
- Raportarea defectelor
- Rigurozitate și minuțiozitate pe tot parcursul proiectului



# Rolul echipei de testare

---

- Atenție la specificații și la detalii
- Respectarea procedurilor de testare aferente produsului în cauză
- Realizarea cazurilor de test
- Detecția defectelor
- Raportarea defectelor
- Rigurozitate și minuțiozitate pe tot parcursul proiectului



# Testarea specifică a produselor software

---

- pagini web de informare, liste de discuții
- jocuri pe calculator -> sume mari de bani
- aplicații militare -> securitate națională
- aplicații medicale -> viața pacienților
- aplicații mari (x 100 mil Euro)
- aplicații medii și mici



# CINE face testarea ?

---



- echipa de testare (realizează numai testarea produsului) – uneori face parte din aceeași companie ca și echipa de dezvoltare, alteori este o companie separată
- echipa de dezvoltare (nu este recomandat, deși este o practică actuală) – procesul de testare este combinat cu cel de depanare
- clientul (în funcție de caietul de sarcini) – produsul este testat de către client pe parcursul dezvoltării sau doar la final

# 3 strategii/categorii

---

## Black Box

Testarea domeniilor de valori, testarea funcțională, testarea bazată pe specificații, testarea axată pe riscuri, testarea la limită, testarea de regresie, testarea bazată pe scenarii, ...

## White Box

Testarea acoperirii, testarea API, testarea statică (examinarea codului, verificarea sintaxei), inserarea de defecte

## Gray Box

Asemănător cu testarea Black Box, având însă cunoștințe legate de structura internă a produsului software

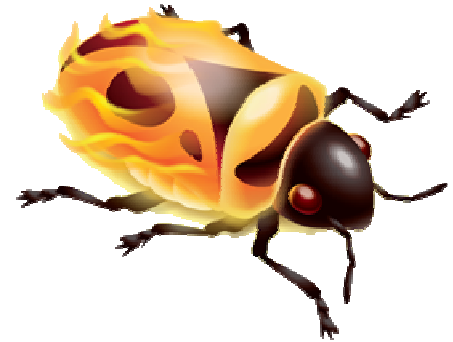


# Ce este un defect (bug)?

---

*“A bug is something that bugs somebody who matters.”*

*James Bach*



*“nerespectarea așteptărilor utilizatorilor”*

*Glenford Myers*

- eroare într-un program software
- abaterea programului software de la specificații
- problemă de funcționalitate

# Calitatea unui produs software

---

**Satisfiers** vs. **Disatisfiers**

||

**Programator** vs. **Testor**



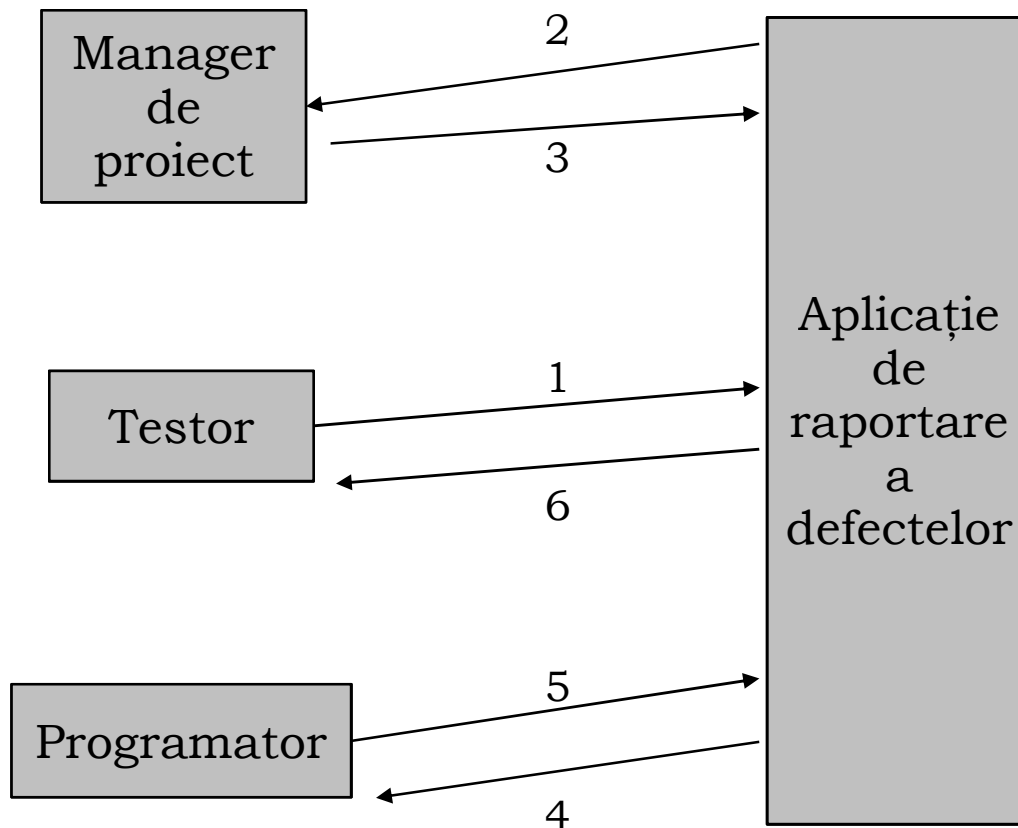
[Joseph Juran]

- respectarea specificațiilor
- funcționalitate
- dezvoltarea rapidă

- greu de utilizat
- lent în rulare
- nefiabil – erori
- nepotrivit cu mediul de lucru al clientului

- diferă de la o companie la alta
- interfață testor -> programator -> testor
- aplicații de raportare
- înregistrarea etapelor de testare pe parcursul dezvoltării proiectului software
- realizarea statisticilor și calcularea metricilor de evaluare a stadiului actual în dezvoltarea proiectului

# Principii de raportare a defectelor



1. deschiderea unui test și raportarea unui defect

2,3. atribuirea defectului către un programator

4,5. rezolvarea defectului pentru versiunea indicată de către managerul de proiect

6. testarea prin regresie

- repetarea pașilor dacă defectul persistă

# Tipuri de defecte

---



- specificații greșite – necesită modificare/completare
- cerințe client greșite (hidden requirements) – necesită completarea/modificarea specificațiilor
- defecte de implementare – nerespectarea specificațiilor
- defecte de proiectare – necesită reproiectarea pe baza specificațiilor
- defecte de documentare
- defecte non-functionale

# Ce se raportează?

---



- Nume defect
- Produsul/Componenta/Subcomponenta
- Versiunea produsului
- Platforma
- Sistemul de operare
- Importanța/gravitatea/severitatea defectului
- Descriere amănunțită

# Utilitare de raportare

---



- JIRA - <http://www.atlassian.com/software/jira>
- GitHub - <http://www.github.com>
- Bugzilla – <http://www.Bugzilla.org>
- Bugaware - <http://www.bugaware.com>
- TrackStudio - <http://www.trackstudio.com>
- TaskComplete - <http://www.taskcomplete.com>

# Bugzilla. Principii de raportare

---

- precizie
- claritate – explicarea pașilor premergători defectului
- un singur defect / raportare
- se raportează orice defect – defect mic -> defect mare
- motivare asupra existenței defectului



# Bugzilla. Starea defectelor

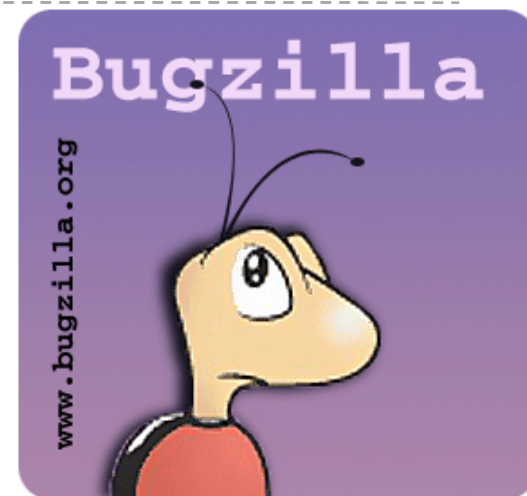
---



- UNCONFIRMED – defectul adăugat în sistem
- NEW – utilizatorii cu drepturi de confirmare (QA/PM) – acceptă defectul
- ASSIGNED – defectul este atribuit unui testor/ grup de testori
- REOPENED – defectul nu a fost reparat conform indicațiilor
- RESOLVED – programatorul repară defectul și consideră procesul terminat
- VERIFIED – procesul de reparare a defectului este verificat
- CLOSED – defectul este considerat inexistent în noua versiune

# Bugzilla. Importanta/Severitatea unui defect

---



## Stabilita de către Testor

- Blocker - importanță maximă
- Critical - nu există soluții de a fi evitat
- Major - există soluții de a fi evitat
- Normal - nu necesită soluții speciale de evitare
- Minor - comportament ciudat
- Trivial - problemă de GUI
- Enhancement - îmbunătățire

# Bugzilla. Prioritatea unui defect

---

## Stabilita de către PM

- P1 – prioritate maximă
- P2
- P3
- P4
- P5 – prioritate minimă



# Bugzilla. Exemplu raportare



Before reporting a bug, please read the [bug writing guidelines](#), please look at the list of [most frequently reported bugs](#)

[Show Advanced Fields](#)

**Product:** C App

**Reporter:** grup\_4@tst.ro

**Component:**

Component Description

login

**Version:**

**Severity:** normal

**Hardware:** PC

**OS:** Windows

We've made a guess at your operating system and platform. Please check them and make any corrections.

**Summary:**

Dimensiunea campului nume

**Description:**

In ecranul de inregistrare a utilizatorilor, dimensiunea campului Nume nu este conforma specificatiilor. Numarul maxim de caractere acceptate este 15. Ar trebui sa fie 20

# Elemente cheie pentru raportare

---



- Descrierea defectului – cât mai amănunțită
- Motivarea programatorului asupra acceptării și rezolvării – prezentarea/documentarea amănunțită a defectului.
- Motivarea importanței/gravității alese (Trivial -> Blocking)
- Atașarea de jurnale (log-uri) și de capturi de ecran – interesant și ușor de urmărit/reprodus pentru programator
- Alegerea unei versiuni actuale – programatorul nu va fi încântat să testeze o versiune învechită și probabil nu va da importanță sau va cere testarea unei versiuni mai noi
- Indicarea cât mai amănunțită a configurației sistemului testat

# Descrierea amănunțită a defectului

---



1. Descriere succintă și la obiect a defectului
2. Enumerarea pașilor realizați pentru obținerea defectului
3. Atașarea de fișiere ajutătoare
4. Explicarea detaliată a defectului și dacă este posibil prezentarea cauzei și a modului de remediere
5. Explicarea posibilelor consecințe ce pot duce la defect blocking

# Raportarea neinteligibilă a defectelor

---



- Programatorul nu reușește să refacă defectul
- Programatorul va renunța să lucreze la rezolvarea defectului
- Pașii necesari reproducerii defectului sunt complecși și ciudați
- Nu este furnizată suficientă informație de reproducere a pașilor
- Programatorul nu înțelege raportul
- Programatorul consideră că este cel mult o cerință nouă

# Obiecțiile programatorilor legate de raportare

---



- Nu este reproductibil
- Nu se înțelege
- Irealist
- Nu este defect – cel mult cerință nouă

# Defecte nereproductibile

---



- Foarte important de a fi raportate
- Programatorul va ști unde să caute
- Programatorul ca să ști cu ce să asocieze dacă defectul este descris amănunțit
- Dacă se atașează salvări de ecrane sau log-uri, programatorul va estima porțiunea de cod care generează defectul
- Verificarea sistemului de raportare dacă nu conține alte defecte asemănătoare
- Descrierea amănunțită și rapidă a condițiilor în care a survenit defectul. În timp se pot uita pașii urmăți