Szoftvertesztelés

Informatikai rendszerüzemeltető
Szoftverfejlesztő, tesztelő
2020.

2023.

Szoftvertesztelés

- Források:
  - Információs rendszerek fejlesztése (PDF)
  - Szoftvertesztelés: Ficsor Lajos, Dr. Kovács László, Krizsán Zoltán, Dr. Kusper Gábor
  - SZOFTVERTESZTELÉS A GYAKORLATBAN: Boda Béla, Bodrogközi László, Gál Zoltán, Illés Péter
  - Programozási alapismeretek: http://progalap.elte.hu/downloads/seged/eTananyag
  - Tesztelési módszerek: https://segedletek.level14.hu
  - Tompa Tamás: Szoftvertesztelés ME. 2019. (PDF/dia)
  - Wikipedia: TDD, XP, Agilis, Scrum...stb.

Szoftvertesztelés

- Témák:
  - Szoftverfejlesztési folyamat
  - Szoftvertesztelés
    - Teszt fogalma
    - Hibák okai
    - Hibák súlyossága, fokozatok
  - Tesztelési módszerek, technikák
  - Tesztelési, fejlesztési stílusok (módszerek)
  - Teszt eszközök, szoftverek

Szoftvertesztelés

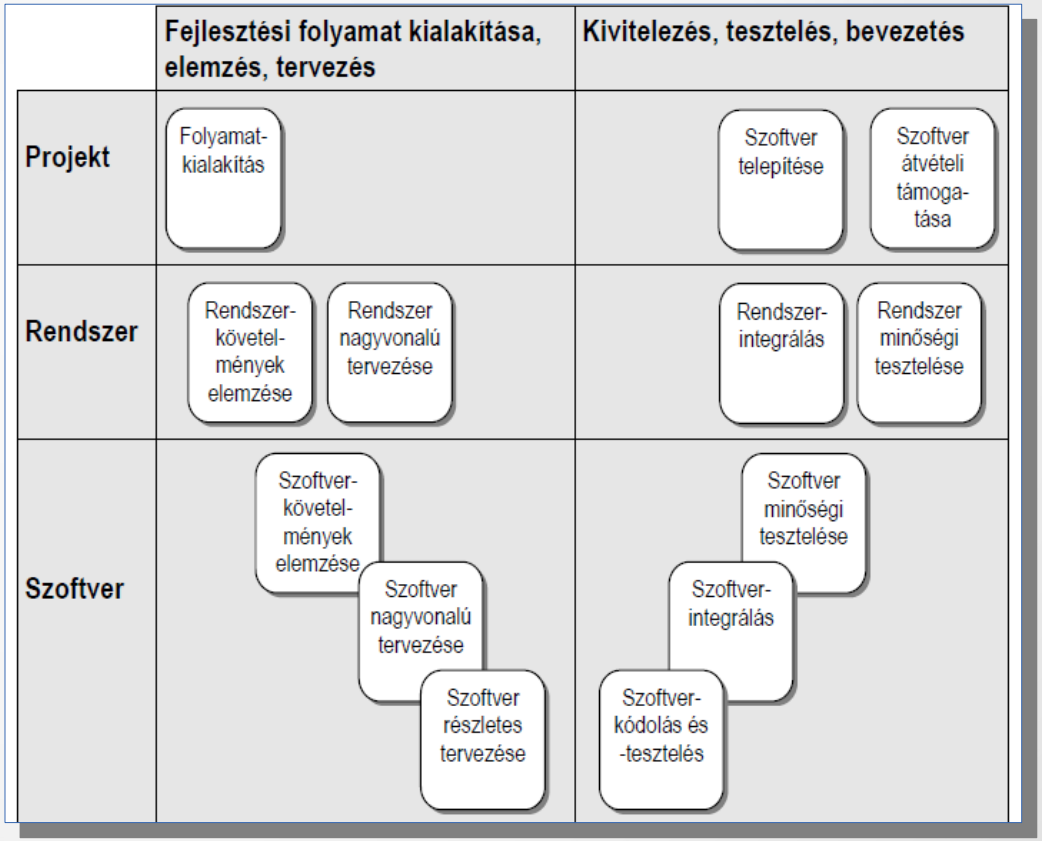
- ***Szoftverfejlesztési folyamat***

Szoftvertesztelés

- ***Szoftverfejlesztési folyamat***
  - ISO 12207 szoftveréletciklus szabvány
  - Tevékenység „V” alakot formál
    - Lsd. következő ábra
  - „V” szárain lefelé, majd felfelé haladva a tevékenységek egy logikai sorrendjét kapjuk
  - A „V” alakú elrendezés azt szimbolizálja, hogy az elemzés és a tervezés felülről az egésztől a részei felé halad, miközben a specifikáció egyre részletesebbé válik; a kivitelezés viszont éppen ellenkező irányú: a legkisebb alkotóelemek megvalósításával kezdődik, majd azokból fokozatosan építi fel az összetettebb szerkezeteket.

Szoftvertesztelés

- ***Szoftverfejlesztési folyamat***
  - V modell:
    - általános modell



Szoftvertesztelés

- ***Szoftvertesztelés***

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Tesztelésre azért van szükség, hogy a szoftver termékben meglévő hibákat még az üzembe helyezés előtt megtaláljuk, ezzel növeljük a termék minőségét, megbízhatóságát. Abban szinte biztosak lehetünk, hogy a szoftverben van hiba, hiszen azt emberek fejlesztik és az emberek hibáznak.

Szoftvertesztelés

- ***Szoftvertesztelés***
  - A tesztelés feladata az, hogy a szoftver használata során fellépő hibák előfordulását csökkentse, a szoftver megbízhatóságát növelje és a szabványoknak, előírásoknak való megfelelőségét biztosítsa. Tehát az ügyfél elégedettségét növelje!

Szoftvertesztelés

- ***Szoftvertesztelés***
  - A teszteléshez kapcsolódó kulcsfogalmak:
    - Minőség
    - Megbízhatóság
    - Elégedettség

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Validáció – Verifikáció fogalma:
    - Validáció: A jó szoftvert készítjük-e, megoldja-e a felhasználó problémáját? Ez a tervezési fázisban van.
    - Verifikáció: Amit elkészítünk, az helyesen működik-e? Ez a kódolás (megvalósítás) közben ill. azt követően van.

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Hibák okai:
    - Hardver hiba
    - Hálózati hiba
    - Konfigurációs hiba
    - Kezelői hiba
    - Nem megfelelő környezet
    - Adatsérülés (működés közbeni adatsérülés)
    - Szoftver készítése során indukált hibák

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Hibák okai:
    - *Szoftver készítése során indukált hibák*:
      - Emberi tévedés, elírás
      - Ismerethiány
      - Segédprogramok hibái (komponensek hibái)
      - Rohammunka és szervezési hibák (kapkodás)

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Hibák súlyossága:
    - A hibák súlyosságát (severity) fokozatokban mérhetjük, ami lehet:
      - 1-5 fokozatú,
      - de elterjedtebb az 1-8 fokozatú.

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Hibák súlyossága:
    - Példa súlyossági szintekre, fokozatokra:
      - ***1 – Wish***: A hiba nincs hatással az üzleti folyamatokra, vagy az általa okozott működésbeli változás nem releváns
      - ***3 – Minor***: A hiba javításra szorul, de vagy nem érinti az üzleti folyamatokat, vagy egy nagyon kis ügyfélkört érint (a napi adatfeldolgozás kevesebb mint 1%-át érinti)

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Hibák súlyossága:
    - Példa súlyossági szintekre, fokozatokra:
      - ***5 – Normal***: Üzleti funkcionalitást érintő folyamatok hibája, nagyobb ügyfélkört érint (a napi adatfeldolgozás kevesebb mint 30%-át, de több mint 1%-át érinti)

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Hibák súlyossága:
    - Példa súlyossági szintekre, fokozatokra:
      - ***7 – Important***: Működést hátráltató, részfunkciókat ellehetetlenítő, bizonyos adatkörökre teljes funkciókat érintő, fennmaradása súlyos üzleti veszteséget okozhat, nagyobb ügyfélkört érint (a napi adatfeldolgozás több mint 30%-át érinti)

Szoftvertesztelés

- ***Szoftvertesztelés***
  - Hibák súlyossága:
    - Példa súlyossági szintekre, fokozatokra:
      - ***8 – Blocker***: A rendszer leállt, használhatatlan. Üzletkritikus, működést nagymértékben befolyásoló, minimális funkcionalitást is ellehetetlenítő hiba, esemény
      - ***9 – Security***: Minden olyan hiba, ami az ügyféladatok szempontjából biztonsági kockázatot jelent

Szoftvertesztelés

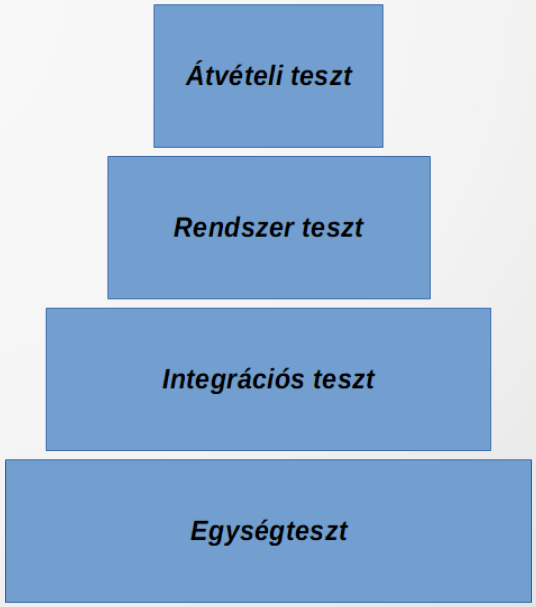
- ***Tesztelési módszerek, technikák***

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztszintek:
    1) Egységtesztek (unit test)
    2) Integrációs teszt
    3) Rendszerteszt
    4) Átvételi teszt
    5) xUnit

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztszintek:
    - egymásra épülnek
    - piramisként szokták ábrázolni
      - többféle ábrázolás lehet



TESZT PIRAMIS

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztszintek:
    - Egységtesztek (unit test): Az egységteszt a rendszer legkisebb önállóan működő egységeit teszteli. Ezek az egységek OOP környezetben az osztályok és azok metódusai, eljárásorientált környezetben az eljárások és a függvények.

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztszintek:
    - Integrációs teszt: Az integrációs teszt (vagy modulintegrációs teszt) programegységeket, modulokat, azok egymással és a környezettel történő együttműködését teszteli. Osztályok, szolgáltatások és függőségeik valós működését teszteljük, ideértve az esetleges adatelérést is.

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztszintek:
    - Integrációs teszt:
      - A *mocking* technika - vagy sokkal inkább egy-egy mocking keretrendszer - a *stubbing* technikával újra és újra megvalósított tesztelési részfeladatokat fogja össze és ad egy egységes keretet a tesztesetek megvalósításához.
        - Mockolás: a tesztelendő funkció helyett egy „dummy”, nem valós funkciót hívunk meg

(pl. email küldő funkciót egy ténylegesen email-t nem küldő funkcióval helyettesítjük)

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztszintek:
    - Rendszerteszt: a teljes integrált rendszer tesztelése, a rendszert olyan vagy ahhoz hasonló környezetben teszteljük, ahogy majd éles környezetben is működni fog. A funkcionális rendszerteszt legfőbb jellemzője, hogy oly módon teszteljük a rendszert, mint ahogy azt a felhasználó használni fogja.
      - Speciális funkcionális rendszertesztek is lehetnek.

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztszintek:
    - Átvételi teszt: az ügyfél vagy annak megbízottja végzi, tehát a fejlesztői csapattól független tesztelő. Az átvételi tesztek célja nem az, hogy az ügyféllel hibát kerestessünk, hanem az, hogy kiépítsük a bizalmat a rendszerrel és annak a való életben történő alkalmazásával kapcsolatban.

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztszintek:
    - xUnit: Az programozói szintű teszttechnikák eszközeit és az azokhoz kapcsolódó filozófiai, módszertani elgondolásokat fogja össze.
      - szinte minden nyelven, platformon megvalósították azokat a konkrét eszközöket, melyekkel hatékonyan tudunk alacsony szintű teszteket írni.
      - xUnit Java megvalósítása: JUnit

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Teszttípusok:
    - Funkcionális tesztek
      - (funkciók tesztje)
    - Nem funkcionális tesztek
      - (nem konkrét funkció teszt, pl. terhelhetőségi teszt)

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Teszttípusok:
    - Funkcionális tesztek:
      - ALFA-teszt: Azokat a teszteket, melyeket egy szűk, esetlegesen belső felhasználói kör segítségével hajtunk végre, alfa-teszteknek hívunk. Leginkább az úgynevezett dobozos alkalmazások esetén elterjedt ez a fajtaterminológia. Ezeket a teszteket még azelőtt hajtjuk végre, hogy külső felhasználók számára elérhetővé tennék, azaz bétatesztre engednék az alkalmazást.

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Teszttípusok:
    - Funkcionális tesztek:
      - BÉTA-teszt: Ezeket a teszteket még azelőtt hajtjuk végre, hogy külső felhasználók számára elérhetővé tennék, azaz további bétatesztre engednék az alkalmazást. A béta verzióval együtt illik egy ún. changelog-ot is kiadni, mely tartalmazza a változásokat, azok leírását. A béta-teszteket az átvételi teszt szintbe soroljuk.

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Teszttípusok:
    - Nem funkcionális tesztek:
      - Teljesítmény teszt (performance test)
        - pl. terhelési-, stresszteszt, kitartási teszt
      - Csúcs teszt
      - Mennyiségi teszt
      - Biztonsági teszt
        - ...stb.

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztelési technikák:
    - Statikus
    - Dinamikus

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztelési technikák:
    - Statikus technikák:
      - nem igényelnek futtatást, hiszen elemző módszerek, ezért jól alkalmazhatók nem csak programkódon, de különböző dokumentációkon, specifikációkon is. Segítségükkel korán találhatunk eltéréseket az előírásoktól (requirements), az előzetes tervektől, így kis költséggel sokat javíthatunk a termék minőségén (hiszen egy hiba javítása annál olcsóbb, minél korábban sikerül észrevennünk) – átvizsgálás, átnézés

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztelési technikák:
    - Dinamikus technikák:
      - futó rendszert elemezzük működés közben
      - Két alapvető típusa lehet:
        - ***Fekete doboz*** technika (black box techniques)
        - ***Fehér doboz*** technika (white box techniques)

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztelési technikák:
    - Dinamikus technikák:
      - ***Fekete doboz technika (black box techniques):***
        - Nem áll rendelkezésre forráskód
        - csak a bemenetét és a kimenetét tudjuk kezelni, olvasni
        - az ismert rendszerspecifikációk (system specification) alapján történő modellalkotás, majd e modell segítségével a szükséges bementi adatok és a belőlük fakadó kimeneti adatok összességének meghatározása

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Tesztelési technikák:
    - Dinamikus technikák:
      - ***Fehér doboz technika (white box techniques):***
        - Rendelkezésre áll a forráskód
        - olyan eseteket állapít meg (tesztel), melyek különböző megközelítések alapján be tudják járni a kód egészét (egész kód tesztelhető).

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Az ***AAA*** minta:
    - Unit tesztek általános felépítése
    - Arrange, Act, Assert
      - Arrange: értékek, adatok beállítása, inicializálása ami kell az adott tesztesethez.
      - Act: tesztelt metódus meghívása
      - Assert: viselkedés ellenőrzése

Szoftvertesztelés

- ***Tesztelési módszerek, technikák***
  - Az ***AAA*** példa – C# programnyelv:

```
// arrange
double r1 = 1.0;
double area1 = r1*r1*Math.PI;
Circle c1 = new Circle(new Point( 0.0, 0.0 ) );

// act
double resultArea = c1.Area();

// assert
Assert.AreEqual( area1, resultArea, 0.001 );     # 0.001 delta/eltérés
```

Szoftvertesztelés

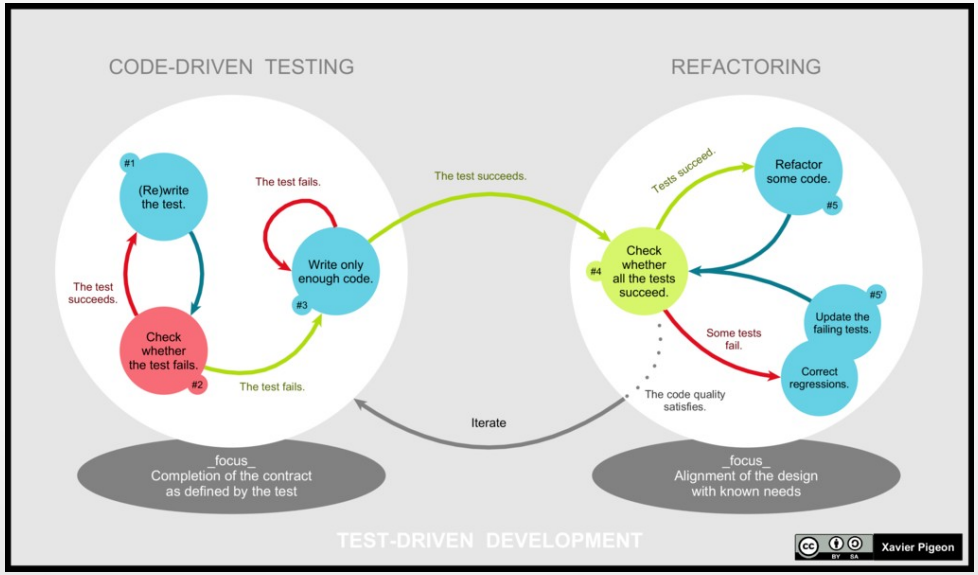
- ***Tesztelési, fejlesztési stílusok (módszerek)***
  - ***TDD***
  - ***XP***
  - ***Agilis***
  - ***Scrum***

Szoftvertesztelés

- ***Tesztelési, fejlesztési stílusok (módszerek)***
  Többféle stílus, módszer létezik. Néhány:
  - ***TDD (test-driven development)***:
    - Tesztvezérelt fejlesztés
    - Folyamata:
      1)Teszt hozzáadás (tesztírással kezdődik)
      2)Minden korábbi teszt futtatása, hogy kiderüljön az új teszt megbukik e
      3)Kódírás
      4)Minden teszt újbóli futtatása
      5)Kódszépítés, -finomítás

Szoftvertesztelés

- ***Tesztelési, fejlesztési stílusok (módszerek)***
  Többféle stílus, módszer létezik. Néhány:
  - ***TDD (test-driven development)***:
    - Használhatnak unit test-eket
    - Teszt keretrendszereket használnak



Szoftvertesztelés

- ***Tesztelési, fejlesztési stílusok (módszerek)***
  Többféle stílus, módszer létezik. Néhány:
  - ***XP (Extreme Programming)***:
    - célja a szoftver minőségének fejlesztése és a változó vevői igényekhez való reagálás javítása
    - rövid fejlesztési ciklusokban támogatja a gyakori "kiadásokat"

Szoftvertesztelés

- ***Tesztelési, fejlesztési stílusok (módszerek)***
  Többféle stílus, módszer létezik. Néhány:
  - ***XP (Extreme Programming)***:
    - Tevékenységek:
      - Kódolás (szoftver készítése)
      - Tesztelés (központi elem, unit test / elfogadási teszt)
      - Meghallgatás (ügyféllel folyamatos egyeztet)
      - Tervezés (teljes folyamat tervezése)

      Minden szinten folyamatos visszajelzés!

Szoftvertesztelés

- ***Tesztelési, fejlesztési stílusok (módszerek)***
  Többféle stílus, módszer létezik. Néhány:
  - ***Agilis szoftverfejlesztés***:
    - a szoftver követelmények és a megoldások együttműködésen keresztül együtt fejlődnek az önszerveződő és multifunkcionális csapatok között
    - korai szállítás, folytonos továbbfejlesztés

Szoftvertesztelés

- ***Tesztelési, fejlesztési stílusok (módszerek)***

  Többféle stílus, módszer létezik. Néhány:

  - ***Agilis szoftverfejlesztés***:
    - Agilis Kiáltvány 12 alapelv:

1) Ügyfél elégedettség fenntartása a használható szoftver gyors szállításaival
2) Változtatás kérések fogadása bármikor még a fejlesztés végén is
3) Működő szoftver gyakori szállítása (hónaponként helyett hetenként)
4) Szoros napi együttműködés az üzleti emberek és a fejlesztők között
5) Projektek motivált, megbízható egyénekből épülnek fel
6) Kommunikáció legjobb formája a szemtől-szembe való információcsere
7) Működő szoftver a haladás legfőbb mérőfoka
8) Fenntartható fejlesztés, állandó mértékben való fenntartás
9) Folyamatos figyelem a technikai kiválóságok és jó tervezés felé
10) Egyszerűség - az el nem végzett munka mennyiségének maximalizálása művészi fokon - elengedhetetlen
11) Önszerveződő csapat
12) Alkalmazkodás szabályos időközönként a változó körülményekhez

Szoftvertesztelés

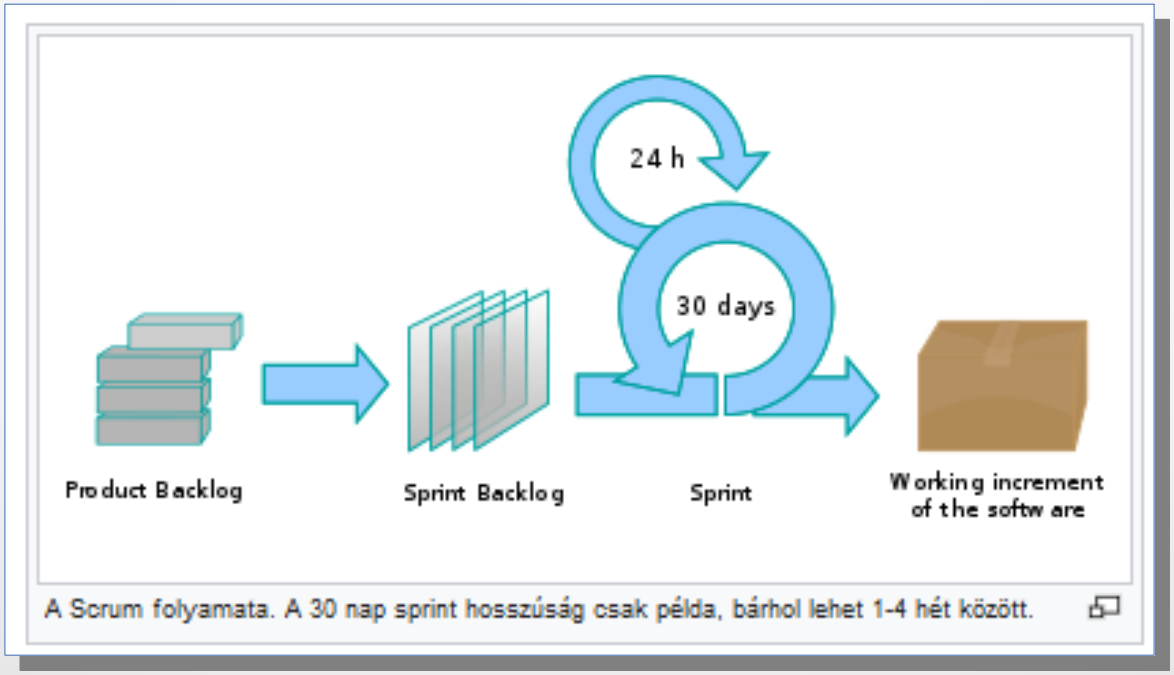
- ***Tesztelési, fejlesztési stílusok (módszerek)***
  Többféle stílus, módszer létezik. Néhány:
  - ***Scrum szoftverfejlesztés***:
    - Agilis fejlesztéshez kapcsolódik, ahhoz hasonló
    - A Scrum keretrendszer a Scrum Csapatokból, valamint a hozzájuk rendelt szerepekből, eseményekből, munkaanyagokból (artifacts) és szabályokból áll.

Szoftvertesztelés

- ***Tesztelési, fejlesztési stílusok (módszerek)***
  Többféle stílus, módszer létezik. Néhány:
  - ***Scrum szoftverfejlesztés***:
    - Részei, folyamata:
      - Csapatok
        - Szerepek
      - Megbeszélések
        - Folyamatos visszacsatolás
      - Munkaanyag
        - Backlog-ok:
          - Teendők, követelmények leírása

Szoftvertesztelés

- ***Tesztelési, fejlesztési stílusok (módszerek)***
  Többféle stílus, módszer létezik. Néhány:
  - ***Scrum szoftverfejlesztés***:
    - Folyamata:



A Scrum folyamata. A 30 nap sprint hosszúság csak példa, bárhol lehet 1-4 hét között.

Szoftvertesztelés

- ***Tesztelési eszközök, szoftverek***

Szoftvertesztelés

- ***Tesztelési eszközök, szoftverek***
  - ***Jenkins:***
    - Automatizáló szerver szoftver
    - Tesztelés is automatizálható a megfelelő komponensekkel
    - Fontos szerepe van a Jenkins rendszernek a CD (folyamatos fejlesztés), CI (folyamatos integráció) folyamatokban!
    - https://jenkins.io

Szoftvertesztelés

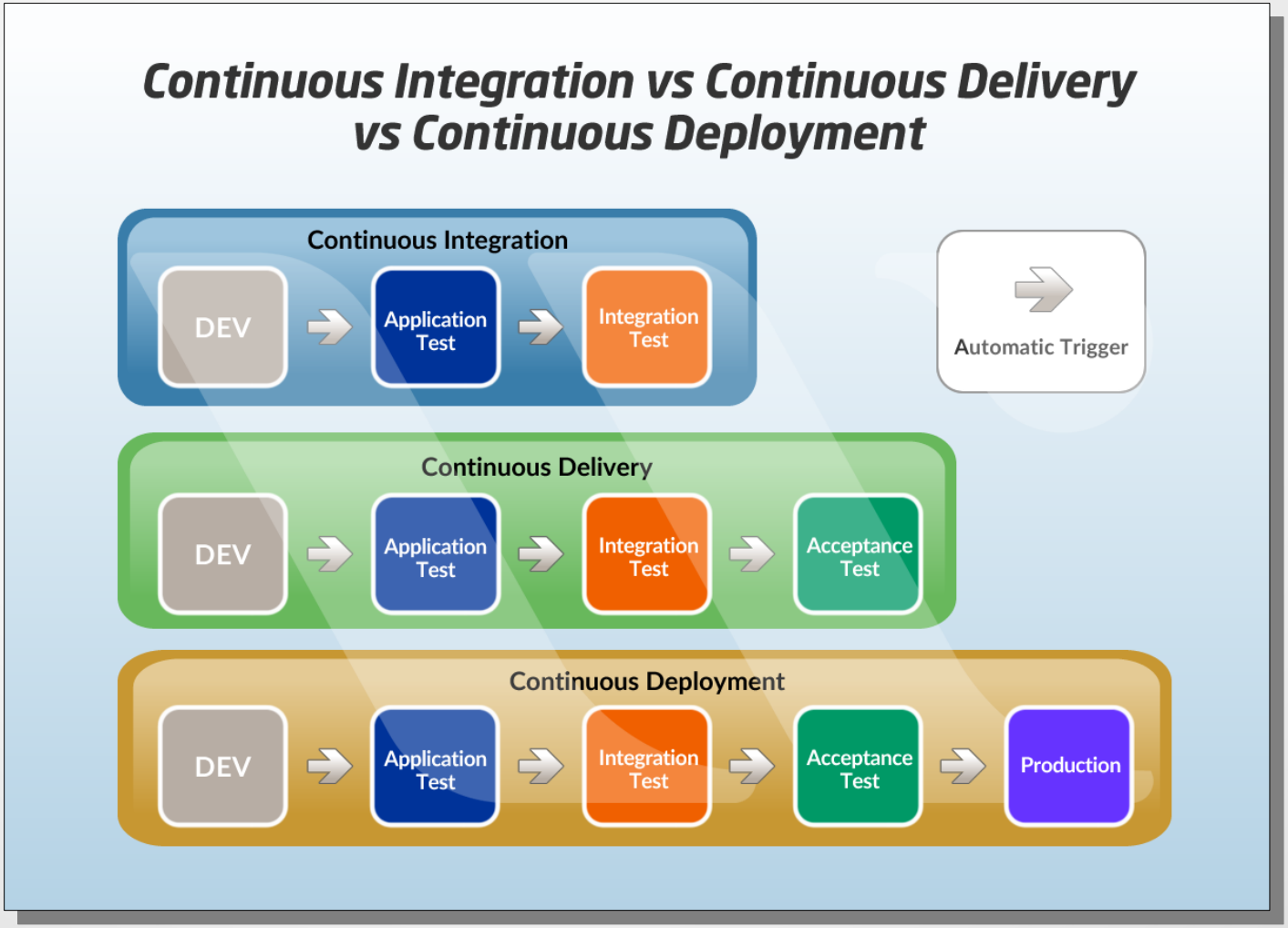
- ***Tesztelési eszközök, szoftverek***
  - ***Apache JMeter:***
    - Terheléses tesztekre használható eszköz
    - Főként web alkalmazások szolgáltatásainak teljesítmény mérésére és analízisre használják
    - használható egységteszt eszközként
    - Konfigurálható monitor eszközként, bár ez nem a tipikus alkalmazása
    - http://jmeter.apache.org

Szoftvertesztelés

- ***Tesztelési eszközök, szoftverek***
  - ***Selenium:***
    - Web alkalmazás tesztelő keretrendszer
    - Funkcionális tesztelésre használható
    - Komponensei:
      - Selenium IDE: ***webes tesztelés böngészőkbe épülő modullal***
      - Selenium Grid: ***központi, elosztott tesztelés***
      - Selenium WebDriver: ***programozható tesztelő eszköz***
        - Java, C#, Python ...stb.
    - https://selenium.dev

Szoftvertesztelés

- ***CI/CD***



Szoftvertesztelés

KÖSZÖNÖM A FIGYELMET!