

Tesztelési típusok

SO, WHAT IS SOFTWARE TESTING



UNIT TESTING

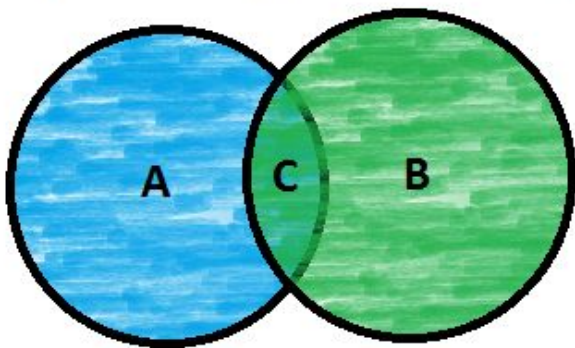
- **Egységvizsgálat**
 - Célja, hogy elkülönítse a kód egy részét és ellenőrizze annak helyességét.
 - Időt és pénzt takarít meg



INTEGRATION TESTING

- Integrációs teszt
- Modulok logikusan integrálva, csoportként

System Integration Testing



Modules

- Miért van rá szükség?
 - Egy modult általában egy egyedi szoftverfejlesztő tervez, amelynek megértése és programozási logikája eltérhet a többi programozótól. Az integrációs tesztelés szükségessé válik annak ellenőrzésére, hogy a szoftvermodulok egységesen működnek-e.
 - A modulfejlesztés idején nagy eséllyel változhatnak az ügyfelek igényei. Ezek az új követelmények nem feltétlenül tesztelhetők, ezért rendszerintegrációt tesz szükségessé.



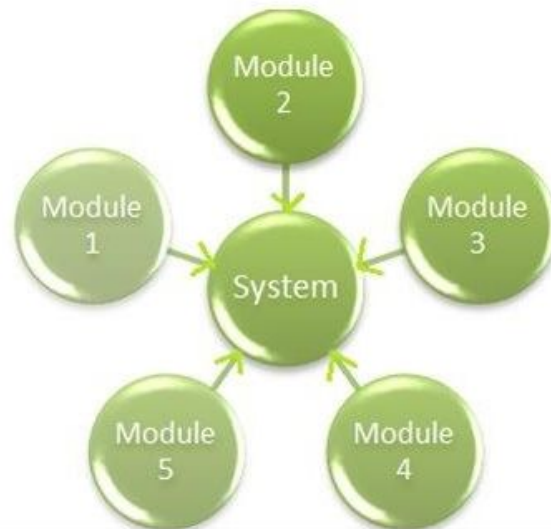
INTEGRATION TESTING - Big Bang megközelítés

- **Előnyök**

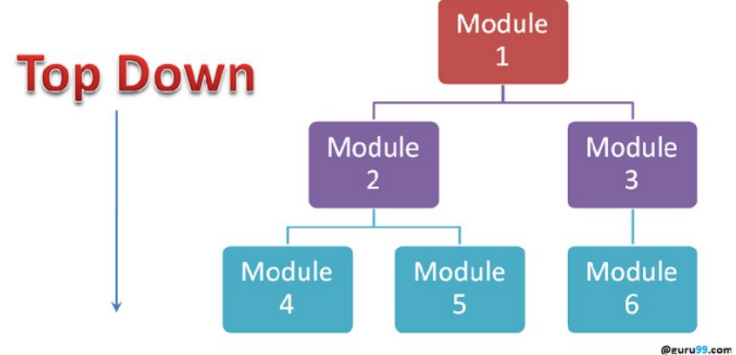
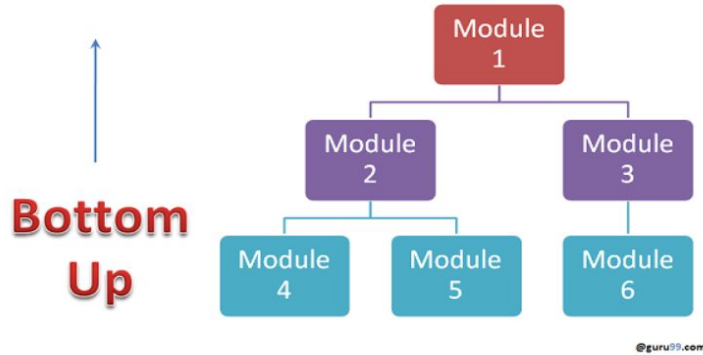
- Kényelmes kis rendszerekhez.

- **Hátrányok**

- A hiba lokalizációja nehéz.
- Tesztelés a modulok megtervezése után történik, ezért a tesztelést végző csapatnak kevesebb lesz a végrehajtási ideje a tesztelési fázisban.
- A modulokat együtt tesztelik, ezért a kritikus és a felhasználói interfészekkel foglalkozó perifériás modulokat nem különítik el, modulokat sem tesztelik külön, ez pedig kockázatot jelent.



INTEGRATION TESTING - Inkrementális megközelítés

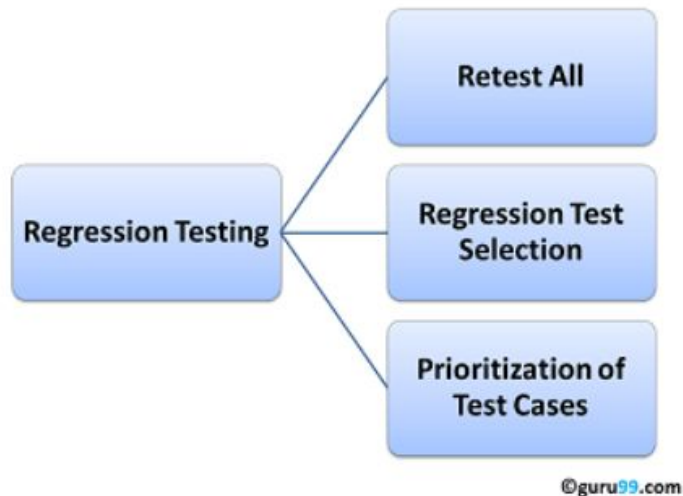


REGRESSION TESTING

- **Regressziós teszt**
- Megerősíti, hogy egy újabb program vagy kódváltozás nem befolyásolja hátrányosan a meglévő szolgáltatásokat.
- A már végrehajtott vizsgálati esetek teljes vagy részleges kiválasztása és újbóli végrehajtása.
- Kódváltozásnak ne legyen mellékhatása a meglévő funkciókra.
- Mikor van rá szükség?
 - követelmény, kódváltozás
 - új funkció
 - meghibásodás
 - teljesítményhiba javítás



REGRESSION TESTING - Tesztelési technikák



- **Mindent újra**
 - Meglévő összes teszt újbóli végrehajtása
- **Regressziós teszt kiválasztása**
 - Tesztcsomag egy részének kiválasztása
 - Újrafelhasználható vizsgálati esetek alkalmazhatók a következő regressziós ciklusokban
 - Elavult tesztoldalak nem használhatóak a következő ciklusokban
- **Vizsgálati esetek prioritása**
 - Üzleti igényre, a kritikus és gyakran használt funkciók függvényében határozza meg a vizsgálati eseteket



REGRESSION TESTING - Kihívások, problémák

- Idő és költségvetési korlátok miatt az egész regressziós tesztcsomag nem hajtható végre.
- Tesztsorozat minimalizálása mellett a tesztlefedettség maximalizálása továbbra is kihívást jelent.
- Gyakoriságának meghatározása
 - Minden módosítás vagy minden frissítés után vagy egy sor hibajavítás után

Regression:
"when you fix one bug, you
introduce several newer bugs."



SMOKE TESTING

- **Füstvizsgálat**
- Annak ellenőrzésére szolgál, hogy a telepített szerkezet stabil-e vagy sem. Megállapítja, hogy a QA csapatnak további vizsgálatra van-e szüksége.
- Minimális számú tesztet végez, fontos funkciók működnek-e.
- Szokás mini és gyors regressziós tesztnek is nevezni.
- Olyan folyamat, ahol a szoftvert a QA környezetre telepítik és az alkalmazás stabilitását biztosítják.
- Fő célja:
 - korai jelentőségű problémák felderítése
 - rendszer stabilitása
 - követelményeknek való megfelelés
- Tartalmaz minden olyan adatot és összetevőt, amely a termékfunkció (k) végrehajtásához szükséges.



SMOKE TESTING

Mikor alkalmazzuk?

- Szoftverhez új funkciót fejlesztenek a már meglévő rendszerhez.
- Ellenőrizzük, hogy minden kritikus funkció megfelelően működik.
- Bármilyen hiba fellépésekor a rendszert vissza kell vinni a fejlesztő csapathoz.
- Minden stabilitás érintő változás esetén füstvizsgálatot kell elvégezni.

Miért használjuk?

- Biztosítja a rendszer helyességét a kezdeti szakaszokban.
- Megtakaríthatjuk a tesztelési erőfeszítéseket.
- Csak sikeres füstvizsgálat után kezdjük el a funkcionális tesztelést.
- Egyszerűsíti a főbb hibák észlelését és korrigálását.
- A leg súlyosabb hibákat találja meg.



SMOKE TESTING - Ki végzi?



VERIFICATION & VALIDATION TESTING

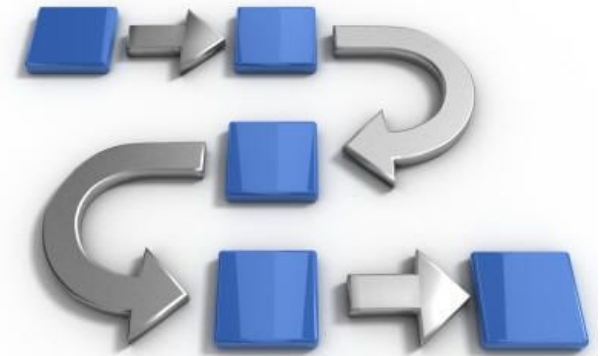
- A “verification and validation” (V&V) az az eljárás, melynek során ellenőrizzük, hogy a szoftver-rendszer megfelel-e a specifikációknak és betölti-e az elvárt szerepét. Másként nevezik szoftver-minőség ellenőrzésnek is.
- Általában a tesztelők végzik az életciklus részeként.

Verifikáció	Validáció
Statikus eljárás, melynek során jóváhagyjuk a dizájnokat, programokat, dokumentumokat és a kódot	Dinamikus eljárás, melynek során ellenőrizzük magát a terméket
Nem kell hozzá futtatni a kódot	Futtatni kell hozzá a kódot
Ember végzi a dokumentumok vizsgálatát	Számítógép végzi a kód futtatását
A szoftver megfelel-e a követelményeknek	A szoftver megfelel-e a felhasználói elvárásoknak és követelményeknek



END-TO-END TESTING

- Az End-to-End tesztelés egy olyan tesztelési típus, mely nem csak a tesztelés alatt álló szoftver-rendszert validálja, hanem teszteli a külső interfészekkel való integrációját is. Ezért hívják “End-to-End”-nek.
- Igazi termelés-szerű adatokat használ és tesztkörnyezetet, hogy real-time beállításokat szimuláljon. Más néven lánc tesztelésnek nevezik.
- Az End to End Testing általában a funkcionális és rendszertesztek után következik.



End to End Testing



END-TO-END TESTING

- Előnyök:
 - Igazolja az alkalmazás megfelelőségét
 - Bővíti a teszt-lefedettséget
 - Detektálja a hibákat és növeli az alkalmazás produktivitását
 - Mérsékli a tesztelési erőfeszítéseket és költségeket
- 2 eljárás:
 - vertikális End to End tesztelés
 - horizontális End to End tesztelés

“Irreproducible bugs become highly reproducible right after delivery to the customer.”

Michael Stahl's derivative of Murphy's Law



BLACK BOX TESTING

- **Fekete dobozos tesztelés**
- Teszteli a szoftver funkcionalitását anélkül, hogy megvizsgálná a belső kódszerkezetet, a végrehajtási részleteket, valamint, hogy ismerné a belső útvonalakat.
- Teljes mértékben a szoftver követelményei és specifikációi alapján készül.
- A szoftver rendszer bemeneteire és kimeneteire koncentrálnunk.



BLACK BOX TESTING - Általános lépések

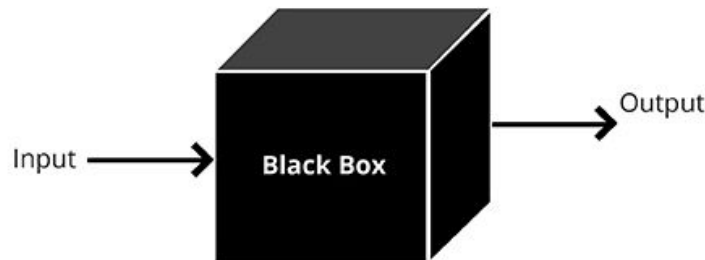
- Kezdetben megvizsgáljuk a rendszer követelményeit és specifikációit.
- Érvényes és nem érvényes bemenetek kiválasztása és ellenőrzése.
- A Tesztelő választja meg az összes bemenetet és várható kimenetet.
- Kiválasztott bemenetekkel tesztesetek létrehozása.
- Meghatározott tesztek végrehajtása.
- Aktuális tesztek kimenetelének összehasonlítása a várt kimenettel.
- Hiba esetén azok rögzítése, majd javítást követően újratesztelés.



BLACK BOX TESTING - Tesztelés típusai

- **Funkcionális tesztelés**
 - A rendszer funkcionális működése kerül megvizsgálásra.
- **Nem funkcionális tesztelés**
 - Olyan nem funkcionális részek kerülnek tesztelésre, mint pl. teljesítmény, skálázhatóság, használhatóság.
- **Regressziós tesztelés**
 - Kódjavítás, frissítés vagy bármilyen más rendszer karbantartás után történik, hogy ellenőrizze az új kód nem befolyásolja a már meglévőt.

BLACK BOX TESTING APPROACH



WHITE BOX TESTING

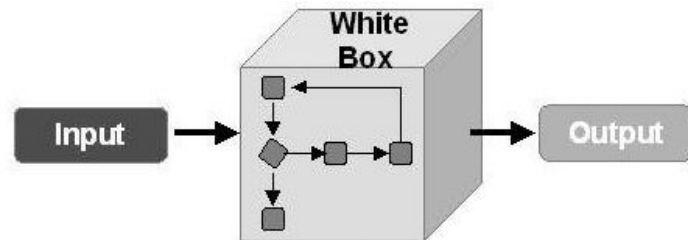
- **Fehérdobozos tesztelés**
- Egy alkalmazás belső működésén alapul, belső tesztelés körül forog.
- Biztonság erősítése, inputok és kimentek áramlása az alkalmazáson keresztül, javítja a tervezést és használhatóságot.
- Clear Box Test, Open Box Test, Glass Box Test, kód alapú tesztelés

WHITE BOX TESTING



WHITE BOX TESTING - Mit tesztelünk?

- Szoftver kódjának tesztelése:
 - Belső biztonsági lyukak
 - Rosszul strukturált útvonalak a kódolási folyamatokban
 - Specifikus bemenetek áramlása a kódon keresztül
 - Várható kimenet
 - Egyes objektumok és funkciók egyéni vizsgálata
- Célja: alkalmazás működésének ellenőrzése.



WHITE BOX TESTING

- **Előnyök:**

- Kódoptimalizálás rejtett hibák elérésével
- Könnyen automatizálható
- Alapos, mivel minden kódútvonalat lefed

- **Hátrányok:**

- Meglehetősen bonyolult és költséges
- Szakmai erőforrásra van szükség a programozás és a megvalósítás részletes megértésével
- Időigényes



AUTOMATED DATA-STRESS TESTING

- Ez a tesztelési típus is a performancia teszt egyik fajtája. A lényege, hogy megpróbáljuk “elárasztani” a rendszert és tesztelni az erre adott válaszát, viselkedését.
- Pl. egyazon időben
 - több felhasználó próbál több fájlt feltölteni
 - több felhasználó próbálja elérni ugyanazt az aloldalt vagy fájlt a rendszerben
 - a felhasználók különböző hostokról töltenek ki egy formot/formokat és elküldik



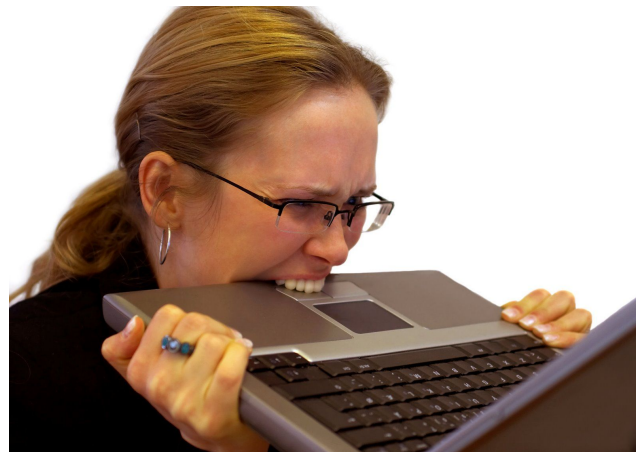
AUTOMATED DATA-STRESS TESTING

- Ha automatizált teszteléshez használjuk, ennek a tesztnek a célja, hogy felhasználja a dinamikus adat-generálást és egy meghatározott időintervallum alatt többször is végrehajtsa. A dinamikus adat-generálás jelentése az adatbevitel randomizálása különböző beviteli mező megszorítások alkalmazásával.
- Mivel az automatizált tesztek bármikor végre lehet hajtani, be lehet állítani, hogy bizonyos időintervallum alatt többször is végrehajtsódjanak, ez a fajta tesztelés lefedettséget tud kínálni széles spektrumú input-adat variációkra.



AUTOMATED DATA-STRESS TESTING

- Fő célja az, hogy biztos legyen, hogy hiba esetén is megjavul a rendszer, vagyis a 'recoverability'.
- Miért hasznos?
 - Teszteli, hogy viselkedik a rendszer extrém terhelés esetén
 - Megfelelő hibaüzenetek legyenek extrém terhelés esetén
 - Ne legyen rendszerleállás ilyen esetekben



AUTOMATED DATA-STRESS TESTING

- Típusai:
 - **Distributed Stress Testing:** elosztott szerver-kliens rendszereknél a szervertől az összes kliens irányába kell tesztelni
 - **Application Stress Testing:** adat-elérhetetlenséghez és blokkoláshoz kapcsolódó, valamint hálózati hibákat keres egy alkalmazásban
 - **Transactional Stress Testing:** kettő vagy több alkalmazás között teszteli a tranzakciókat
 - **Systemic Stress Testing:** a szerveren futó több rendszer tesztelésére alkalmas. Segít megtalálni, ha valamilyen adat-blokk miatt egyik akadályozza a másikat.
 - **Exploratory Stress Testing:** valószínűtlen paraméterek és feltételek használatával kereshetünk vele ritkán előforduló hibákat



SECURITY TESTING

- Security testing, vagyis a biztonsági teszt ellenőrzi, hogy az alkalmazás vagy termék biztonságos-e. A célja, hogy tesztelje az adatok védelmét és a rendszer sebezhetőségét támadások esetén.
- 7 alapelvet kell lefednie a biztonsági teszteknek:
 - Titoktartás
 - Egység
 - Autentikáció
 - Meghatalmazás
 - Elérhetőség
 - Nem-megtagadás
 - OWASP ajánlások



SECURITY TESTING

- A teszt tervezésnél szükséges meghatározni:
 - a biztonsághoz kapcsolódó teszteseteket
 - a teszteléshez szükséges eszközöket
 - a teszteléshez szükséges adatokat
 - a különböző biztonsági eszközök teszt outputjainak az analízisét
- A fejlesztési életciklus elején szükséges elvégezni a Security tesztek, különben magasabb költséggel jár az eljárás.

“The only system which is truly secure is one which is switched off and unplugged, locked in a titanium lined safe, buried in a concrete bunker, and is surrounded by nerve gas and very highly paid armed guards. Even then, I wouldn’t stake my life on it.”

Gene Spafford



SECURITY TESTING

- Típusai:
 - **Vulnerability Scanning:** ismert sérülékenységi jeleket keres
 - **Security Scanning:** hálózati és rendszer gyengeségeket keres
 - **Penetration Testing:** rossz szándékú hacker támadás szimulálása, sérülékenységek felderítésére
 - **Risk Assessment:** a vállalatban fellelhető kockázatok analízisének bevonása a tesztelésbe
 - **Security Auditing:** az alkalmazás vagy operációs rendszer belső megfigyelése biztonsági hibák keresésére
 - **Ethical Hacking:** segítő szándékú hackerek
 - **Posture Assessment:** Security Scanning, Ethical Hacking és Risk Assessment kombinációja, átfogó biztonság





COMPUTER SCIENCE

IF YOU PUT ENOUGH MONKEYS IN FRONT OF ENOUGH KEYBOARDS...

MONKEY TESTING

- A Monkey Testing, vagyis majom tesztelés egy olyan technika, amely során a felhasználó úgy teszteli az alkalmazást vagy rendszert, hogy random inputokat ad meg, és figyeli az erre adott választ, viselkedést, hogy lássa, hogy crashel-e az alkalmazás / rendszer.
- A Monkey tesztet általában random, automatizált unit teszt formájában hajtják végre.
- Miért hívják “majom tesztnek”?
 - A Monkey Testing során a tesztelő (vagy fejlesztő) egy majomnak tekinthető.
 - Ha egy majom próbál használni egy számítógépet, random módon fogja használni, mivel nem érti a működését.
 - Mint ahogy a tesztelő is random módon fog teszteseteket kipróbálni a rendszerben, hogy hibákat találjon, anélkül, hogy előzetesen definiálná a teszteseteket.



MONKEY TESTING




- Típusai:
 - Dumb Monkey: A tesztelők semmit se tudnak a rendszer funkcionalitásáról vagy magáról a rendszerről, így fogalmuk sincs, hogy a tesztesetek mennyire validak.
 - Smart Monkey: A tesztelő pontosan tudja, hogy mi a rendszer célja és funkcionalitása. Valid inputokat ad meg teszteléskor.
 - Brilliant Monkey: A tesztelő úgy hajt végre tesztek, hogy modellezi egy felhasználó viselkedését, és meg tudja határozni, hogy milyen valószínűséggel jönnének elő a hibák.



MONKEY TESTING

- Előnyök:
 - Új fajta bugok
 - Könnyű végrehajtás
 - Elég alacsonyabb szintű szakmai felkészültség is
 - Kevésbé költségigényes
 - A szoftver megbízhatóságát jól lehet tesztelni vele
 - “High impact” bugokat térképezhetünk fel
- Hátrányok:
 - A bug-reprodukció nehézkes lehet
 - Kevésbé hatékony
 - Jó technikai tudást követel
 - Nem elvárt (“Not Expected”) működés is felléphet, aminek analízise időigényes és nehéz lehet
 - Kevesebb bug
 - Időigényes



A serene sunset scene over a body of water. In the foreground, a small, weathered wooden boat with a blue-painted hull is positioned, its reflection visible in the calm water. Two long wooden poles are propped up inside the boat. In the background, several fishing nets are strung across the water, and a few distant boats are visible on the horizon. The sky is a mix of soft pinks, purples, and blues, with scattered clouds. The overall mood is peaceful and contemplative.

Right or wrong, it's very
pleasant to break something
from time to time.

Fyodor Dostoyevsky

Alpha Testing	Beta Testing
Belső alkalmazottak	Ügyfelek/Végfelhasználók
Fehér és fekete doboz technika	Fekete doboz technika
Laboratóriumi környezet	Valós idejű környezet
Hosszú végrehajtási ciklus	Néhány hét
Azonnali vizsgálat kritikus kérdések esetén	Visszajelzések jövőbeli verziókba felhasználva
Termék minőségének biztosítása	Biztosítja, hogy a termék készen álljon a valós idejű felhasználásra

“Alpha is simply that you want somebody to share your pain!”

Anonymous





*“We cannot solve our
problems with the same
thinking we used when we
created them.”*

Albert Einstein

GYAKORLATI FELADAT

- Can't unsee -



Köszönjük a figyelmet!



Attrecto Zrt.
Attrecto Next Tech Digital Solutions

H-9024 Győr, Wesselényi str. 6.
info@attrecto.com

