

A SZOFTVER LENNE TÖKÉLETES



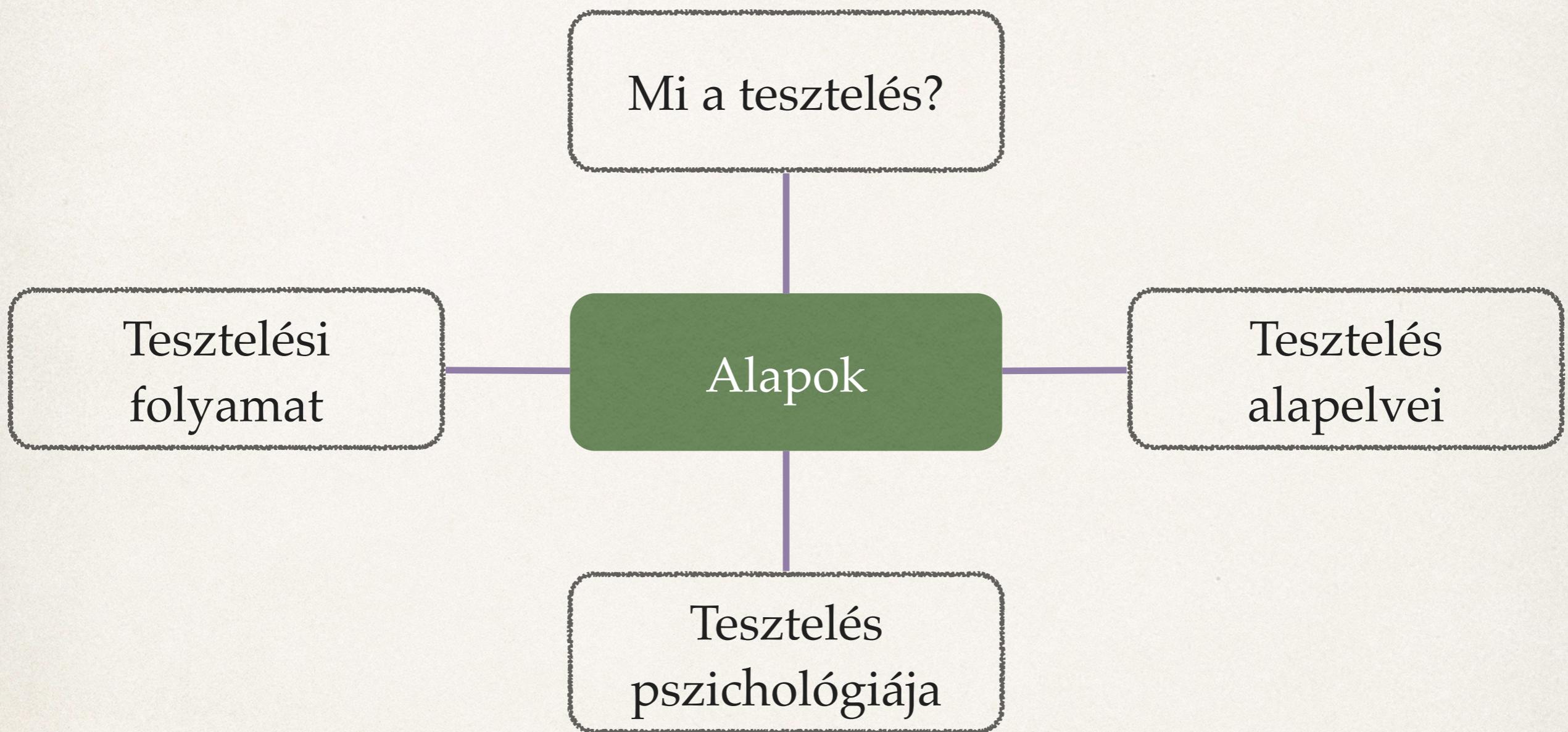
VAGY A TESZTELÉS
VOLT GYENGE?

imgflip.com

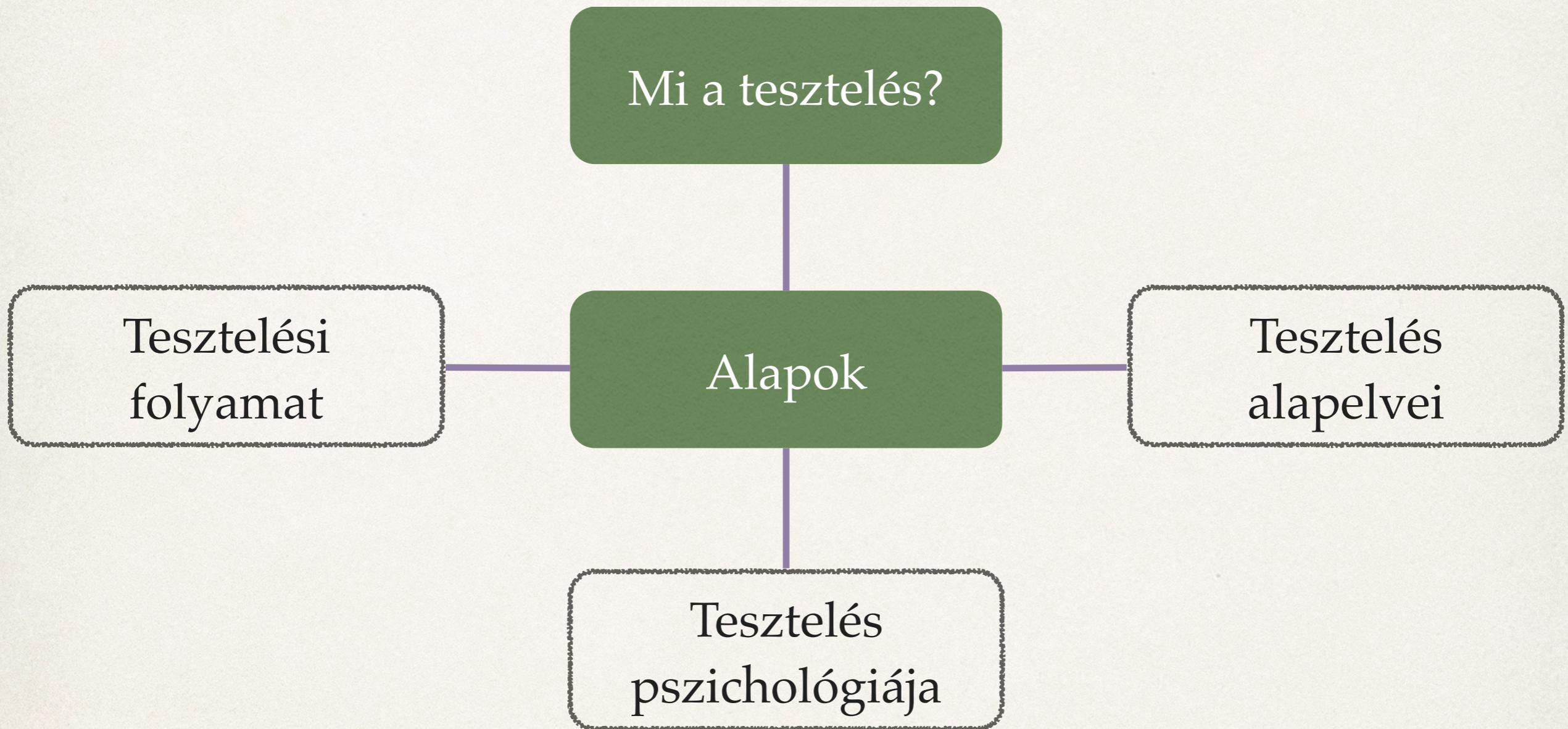
Tesztelés alapjai

Szoftvertesztelés alapjai

Alapok



Alapok::Definíció



Szoftverhibák

- ❖ ESA – Ariane 5
- ❖ Online adóbevallás (előző bevalló adatai láthatóak)
- ❖ „top 10 criminals” weboldal (szerver túlterhelés)
- ❖ Online könyváruház: negatív mennyiség rendelése
- ❖ NASA – Mars Climate Orbiter (metrikus és angol mértékegységek)
- ❖ Mercedes delta-time calculator, 2018-as Ausztrál nagydíj
- ❖ Nagy elektromos szolgáltató számla Y2K probléma miatt
- ❖ Majdnem atomháború (téves riasztás egy Szovjet kilövőálláson, 1983)
- ❖ Bizalmas adat - Ügyfélkapu (hibás szoftver élesítve, üzemeltetés hibája)
- ❖ Útdíjrendszer (összeomlott az élesítéskor)
- ❖ BKV elektronikus bérlet (hibás koncepció)
- ❖ Boeing 737 MAX (két katastrófa a szoftver „biztonsági” funkciója miatt)

Szoftverhibák

Áldozatok:

- ❖ Az ember
- ❖ Egy cég
- ❖ A környezet

☞ *A hibák keresését minél előbb el kell kezdeni!*

Hiba eredménye:

- ❖ Anyagi veszteség
- ❖ Időveszteség
- ❖ Jó hírnév csorbulása
- ❖ Sérülés
- ❖ Halál

Szoftverhibák

\$100 000	1 óra nem tervezett leállás költsége
100-3000 ×	Éles környezet javításának költsége
80%	-a a nem tervezett leállásoknak megelőzhető lenne megfelelő teszteléssel
75%	-a az alkalmazásoknak tesztelés nélkül kerül bevezetésre

Hiba, hiba, hiba

HIBA¹

- Emberi tévedés
- “Error / Mistake”



ARIANE 5 Flight 501 (ESA, 1996)

HIBA²

- Hiba a rendszerben
- “Defect / Bug”

HIBA³

- Manifesztálódott hiba, meghibásodás
- “Failure”

Harvard Mark II (1947)

Hibák gyakori okai

- ✿ Szoros határidők
- ✿ Tapasztalatlan vagy nem megfelelően képzett résztvevők
- ✿ Nem megfelelő kommunikáció
- ✿ Félreértések
- ✿ Kód, tervezek, architektúra komplexitása
- ✿ Új, ismeretlen technológia
- ✿ Emberi tökéletlenség
- ✿ Környezeti tényezők
- ✿ ...

Okok feltárása



- ❖ Hibák okainak felkutatása rendkívül fontos!
 - ❖ Megfelelő és igazságos kezelési lépések
 - ❖ Későbbi problémák megelőzése
- ❖ **Eredendő ok:** legkorábbi körülmények vagy tevékenységek, amik hozzájárultak a hibához (root cause)
- ❖ A felhasználó általában csak a hatásokat látja

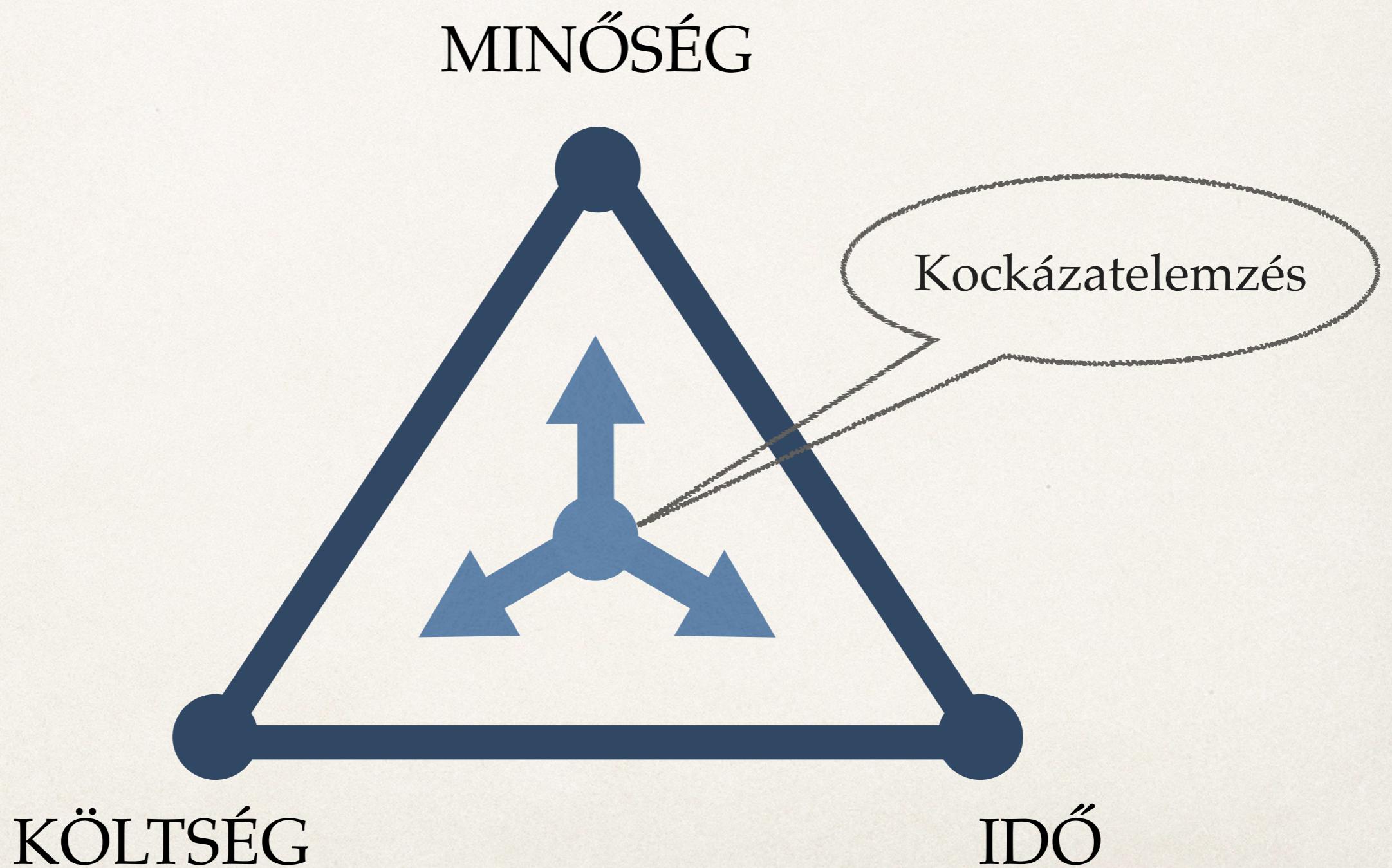
Minőség ára



- ❖ Miért maradhatnak defektusok a rendszerben???
- ❖ Mert **rosszul teszteltünk!**
 - ❖ Keveset teszteltünk
 - ❖ Eleve rossz típusú teszteket használtunk
- ❖ Hibátlan fejlesztés nem létezik
 - ❖ A fejlesztés emberi kreatív tevékenység, ami hibalehetőséget hordoz
 - ❖ A több és jobb teszt így indokolt célnak tűnik, de ez nem olyan egyszerű!

Kimerítő tesztelés:
Minden lehetséges
végrehajtás ellenőrzése

Minőség ára



Alapok::Definíció::Kockázatok



GOOD



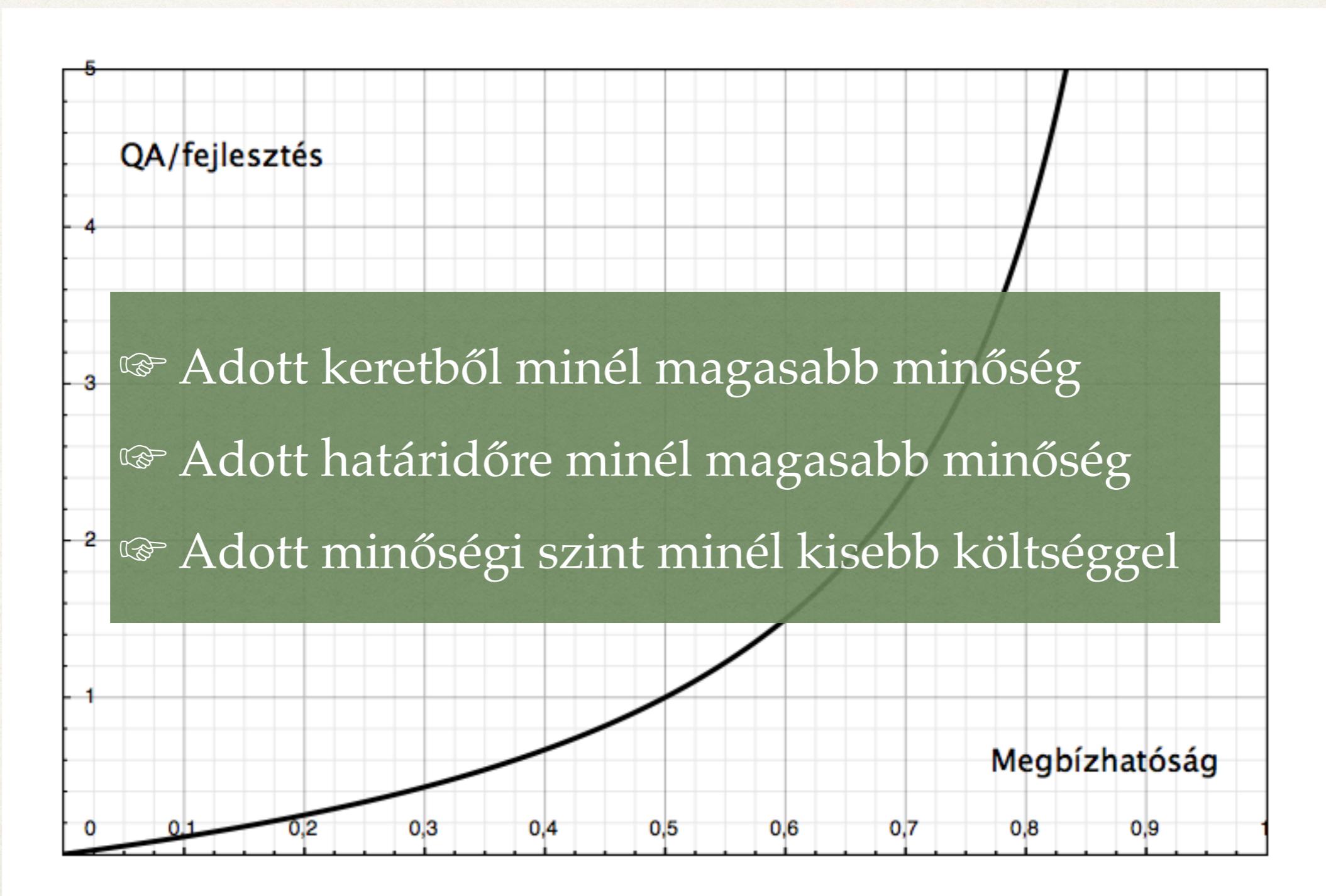
CHEAP



FAST

ANIMATION BY
MISSINGCLOUD.COM - © 2017

Tesztelés = kockázatcsökkentés



Kritikus rendszerek

- ✿ Elérhetőség
- ✿ Megbízhatóság
- ✿ Biztonságosság
- ✿ Védettség



Kritikus rendszerek fajtái

Fajtái:

1. Biztonságosság-kritikus
atomerőmű

2. Küldetéskritikus
űrhajó

3. Üzletkritikus
számlavezető rendszer

Károk:



*Közvetlen
költségek*



*Közvetett
költségek*

Mi a tesztelés és mit csinál?

- ✿ A tesztelés nem csak tesztek futtatása, hanem egy nagyobb folyamat, többféle részfeladattal
- ✿ A tesztelés:
 - ✿ *Verifikáció* (“jól készítettük el a rendszert?”)
 - ✿ *Validáció* (“jó rendszert készítettünk el?”)

Mi a tesztelés és mit csinál?

*A tesztelés
a defektusok felfedezésével:*

- ✓ képes felmérni a szoftver aktuális minőségét,
- ✓ hozzájárul a kockázatok csökkentéséhez és ezáltal alapot ad a szoftver minőségének javításához

Mit és mennyit teszteljünk?

- ✿ Nem tesztelhetünk minden
- ✿ minden rendszernek (és projektnek) megvannak a saját kockázati tényezői és minőségi követelményei
- ✿ Végesek az erőforrásaink
- ✿ DE! A cél: kockázat csökkentése (és nem a tökély elérése)

Tesztelés prioritálása

- ❖ Először a **fontos teszteket futtassuk**, így bármikor hagyjuk is abba, ez a konzervatív
- ❖ Teljesítési / **kilépési kritérium**: előre határozzuk meg, milyen feltételekkel tekintjük befejezettnék a tesztelést, például:
 - ❖ minden tervezett teszt lefutott
 - ❖ minden funkcionális követelményt legalább egyvalaki felülvizsgálta
 - ❖ minden funkcionális követelményt legalább egyszer leteszteltünk
 - ❖ minden ismert, kritikus hibát kijavítottunk
 - ❖ minden eljárást legalább egyszer futtattunk
 - ❖ az összes elágazás 80%-át futtattuk

Tesztelés gyakori céljai

- ✿ Munkatermékek kiértékelése
- ✿ Követelmények teljesülésének ellenőrzése
- ✿ Validáció
- ✿ A teszt tárgyába vetett bizalom kiépítése
- ✿ Hibák és meghibásodások felfedezése
- ✿ Döntés-előkészítés
- ✿ Kockázati szint csökkentése
- ✿ Jogszabályoknak, szerződéseknek, szabályozásnak való megfelelőség ellenőrzése

Tesztelés és hibakeresés

Debugging

- ✿ Két különböző tevékenység
- ✿ Mindkettő fontos és magasabb minőséghez vezet, de
 - ✿ még a debugging semmit nem mond a szoftver egészéről,
 - ✿ addig egy alapos tesztelés (és az azt követő javítások) után mondhatjuk, hogy a szoftver megfelel a követelményeknek

A tesztelés:

- ✿ a fejlesztőtől független
- ✿ defektusok felfedezésére és bejelentésére irányul
- ✿ szisztematikus tevékenység

A hibakeresés, nyomkövetés, hibajavítás (debugging):

- ✿ a fejlesztő végzi
- ✿ célja a defektusok okainak felderítése, kijavítása
- ✿ kevésbé szisztematikus, leginkább ad-hoc

Tesztelés mint szakterület

A tesztelés mint folyamat:

- ❖ A tesztelés sokkal több, mint szimplán a teszt végrehajtása
- ❖ Előkészítés
 - ❖ tervezés
 - ❖ környezet felállítása, stb.
- ❖ „Utómunkák”
 - ❖ jegyzőkönyv vezetése
 - ❖ teszt kiértékelése
 - ❖ teljesítési kritérium kiértékelése
 - ❖ jelentéskészítés, stb.

A teszt céljának meghatározása:

- ❖ Specifikációhelyesség igazolása
- ❖ A lehető legtöbb defektus felfedezése

Statikus és dinamikus tesztelés:

- ❖ Statikus: a kód működését nem vizsgáljuk: review-k
 - ❖ nem csak kódon
 - ❖ minél előbb
- ❖ Dinamikus: teszt futtatás tesztadatokon

Tesztelés mint szakterület

A tesztelés mint módszerek halmaza:

- ❖ Fontos, hogy a teszt effektív legyen, azaz megtalálja a szoftverhibákat
- ❖ Amelyik teszt nem talált hibát, annak nincs a szoftverhez hozzáadott értéke, csak fogyasztotta az erőforrásainkat
- ❖ Ne örüljünk, nem a szoftver hibátlan...
- ❖ Hogyan lehet jó teszteket készíteni?
 - ❖ Gyakorlatban bizonyított technikák alkalmazásával
 - ❖ Számos alapelv van, amelyekre rengeteg technika épül

Teszt típusok (példák)

- ❖ Funktionális / nem-funkcionális
- ❖ Statikus / dinamikus
- ❖ Black-box / white-box
- ❖ Ellenőrző tesztelés (megismételt #1)
- ❖ Regressziós tesztelés (megismételt #2)
- ❖ *Terhelés-teszt (stressz-)*
- ❖ *Használhatósági teszt*
- ❖ *Biztonsági teszt*
- ❖ *Alfa / Béta teszt*
- ❖ *“Csimpánz” teszt*
- ❖ *Stb.*

Tesztelés hozzájárulása a sikerhez

- ❖ Megfelelő teszteléssel csökkennhető a hibásan átadott, üzemeltetett rendszerek gyakorisága
- ❖ De, csakis megfelelő szakemberekkel, megfelelő időben és módon!
- ❖ Példák:
 - ❖ Követelmény review: használhatóbb termék
 - ❖ Részvétel tervezéskor: jobb megértés miatt relevánsabb tesztek
 - ❖ Részvétel kódoláskor: jobb minőségű kód
 - ❖ Tesztelés átadás előtt: hibák megtalálása “házon belül”

Tesztelés és minőségbiztosítás

- ❖ Kockázat folyamatos menedzsmentje
- ❖ Elkerülés, felismerés, eltávolítás
- ❖ QA: prevenció
- ❖ Tesztelés: detekció
- ❖ Nem csak tesztelés!
- ❖ Verifikáció: “jól csináltuk-e?”
- ❖ Validáció: “jót csináltunk-e?”

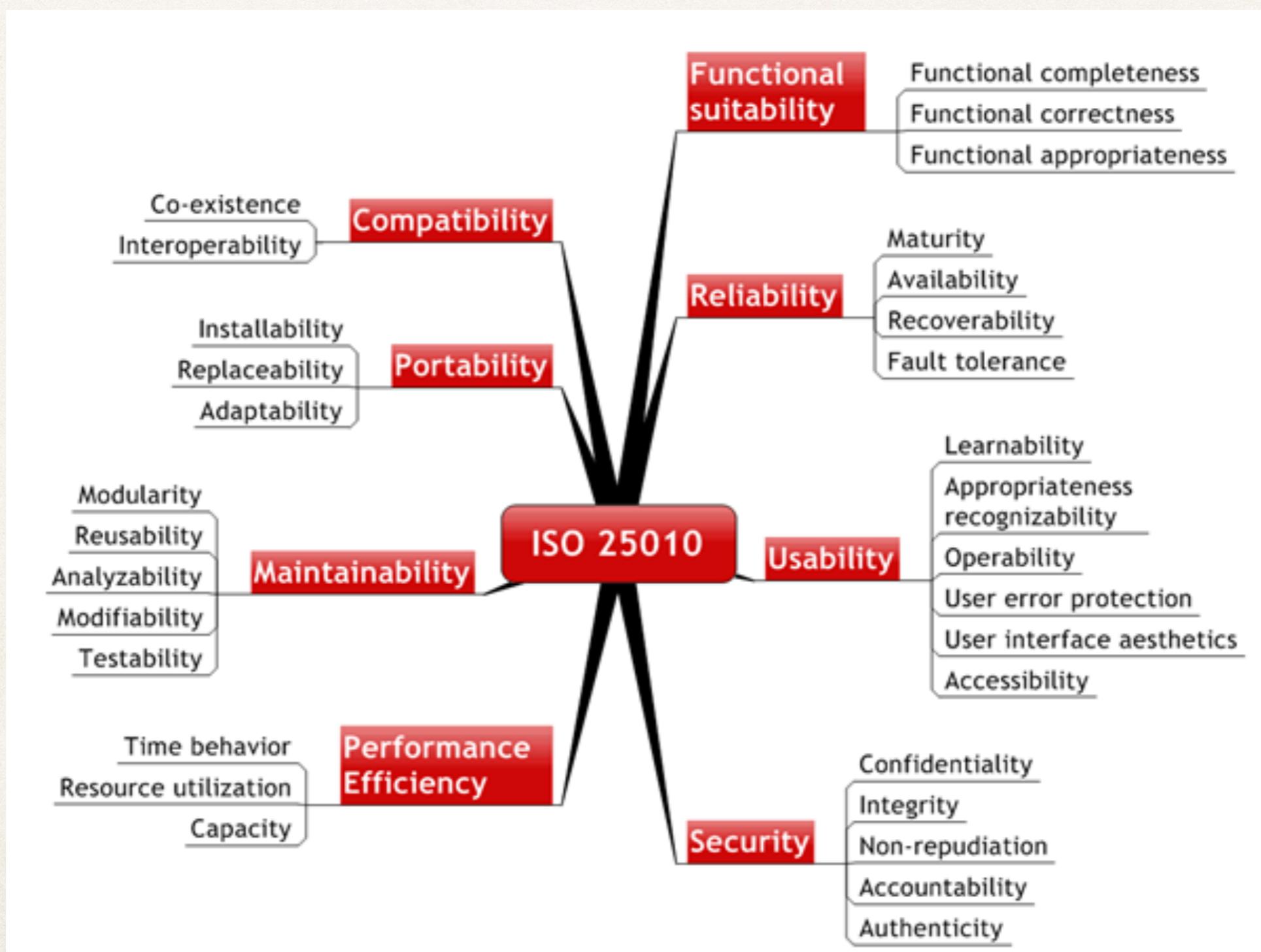


Folyamat-alapú QA

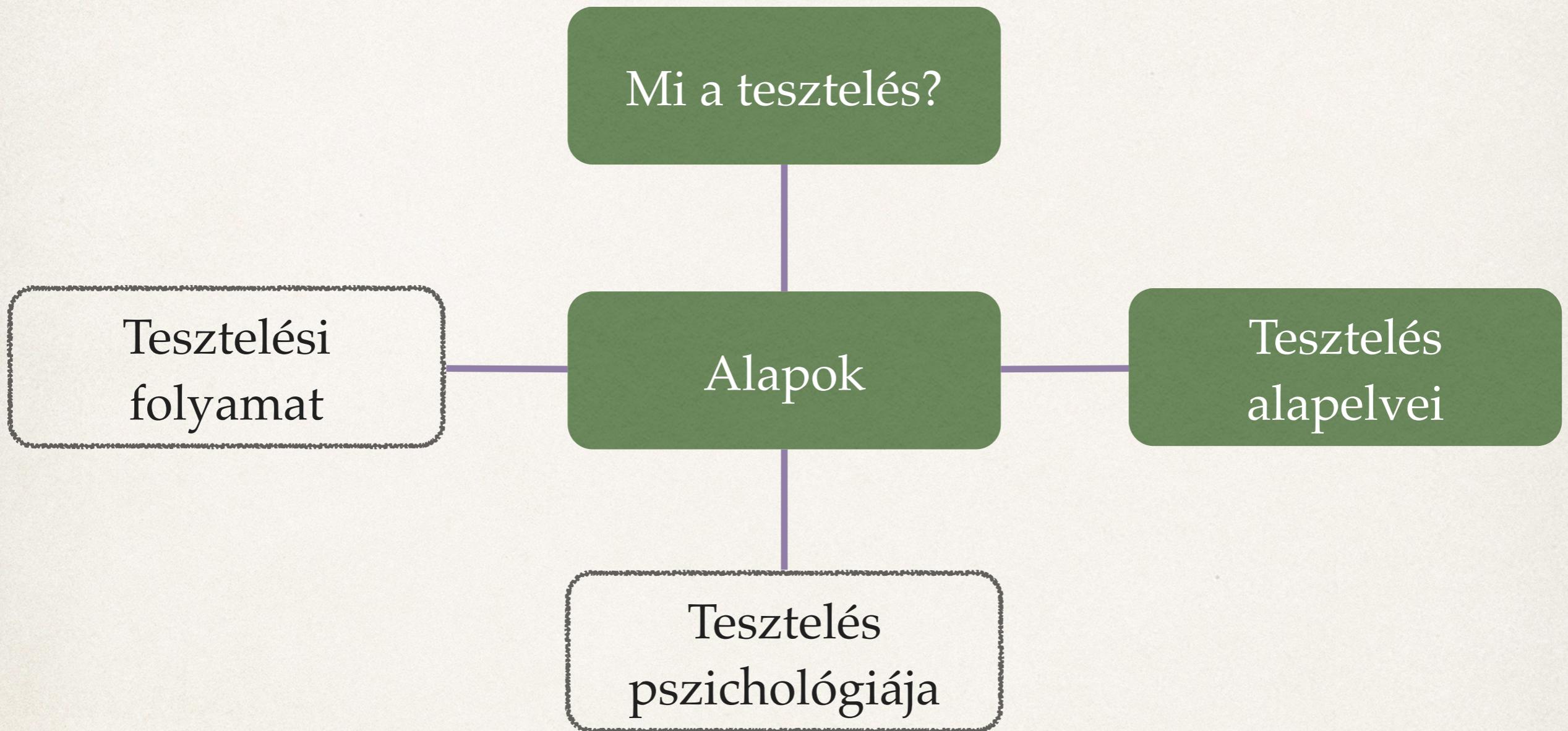
- ✿ Prevenció!
 - ✿ Alapfeltevés:
 - ✿ Gyártás / fejlesztés minőségbiztosítása garancia a termék minőségére
 - ✿ ld. ISO 9001, CMMI
- Tevékenységek:**
1. Minőségbiztosítás (szabályok)
 2. Minőségtervezés (alkalmazás)
 3. Minőség ellenőrzés

Minőség ezer arca

ISO/IEC 9126 (25010)



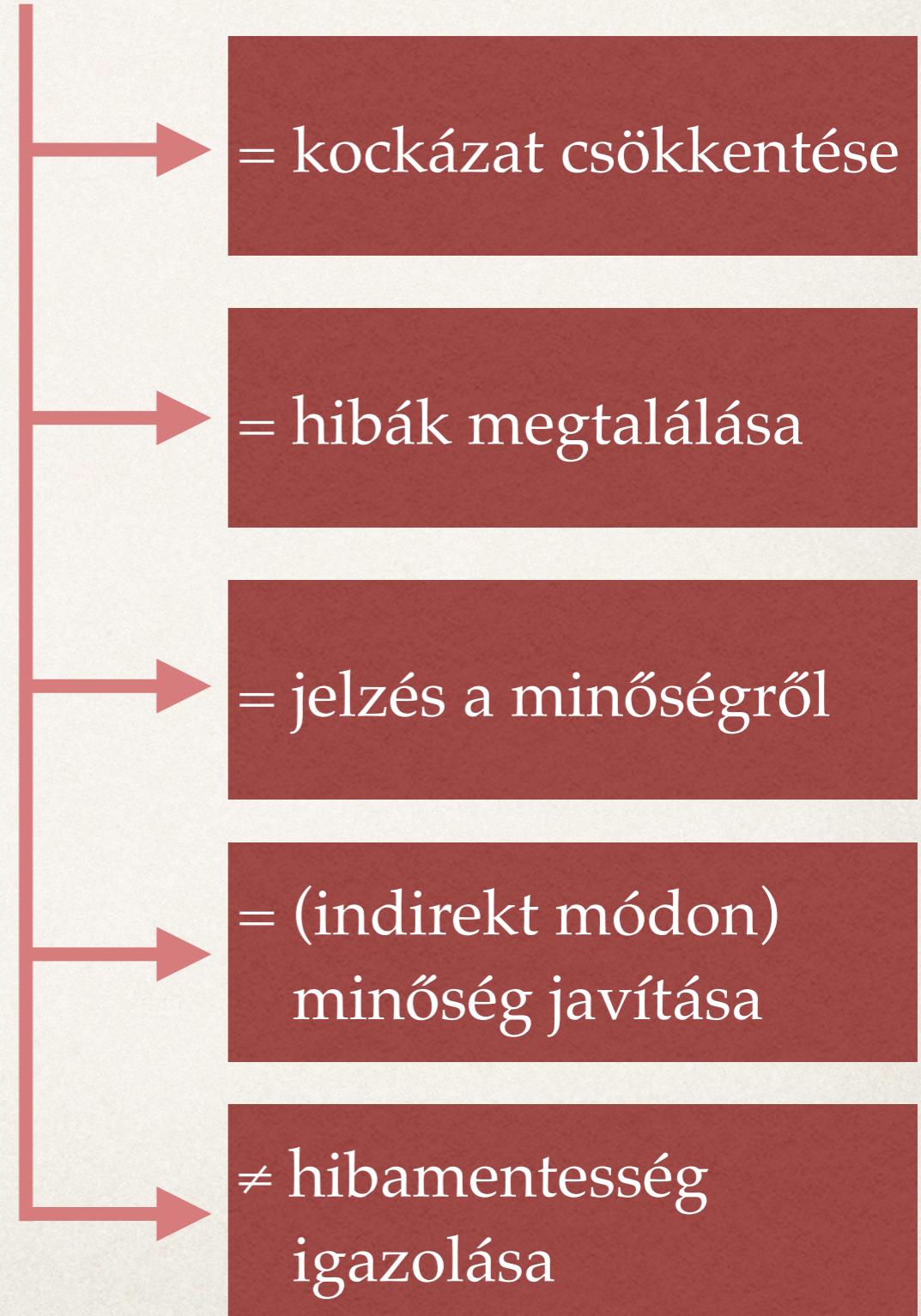
Alapok::Alapelvezek



Alapelvezek

-
- 1. Hibák jelenlétét mutatja ki
 - 2. Kimerítő tesztelés lehetetlen
 - 3. Korai tesztelés
 - 4. Defektus csoportosulás
 - 5. Rovarirtó jelenség
 - 6. Tesztelés környezetfüggő
 - 7. Megtévesztő hibamentesség
 - 8. Tesztelés függetlensége
 - 9. Tesztelés pszichológiája

Tesztelés



A tesztelés a hibák jelenlétét mutatja ki

“A tesztelés a szoftverhibák meglétét képes kimutatni, és nem a hiányukat” [Dijkstra, 1972]

- ✿ A megtalált hibák hiánya még nem jelenti azt, hogy a szoftver megfelelő
 - ✿ A tesztelés csökkenti a felfedezetlen hibák valószínűségét, de még a hibák teljes hiánya sem bizonyítja a szoftver helyességét.
- ✿ „Hibák léteznek” / „A jó tesztelő pessimista”
- ✿ A teszteket úgy kell tervezni, hogy a lehető legtöbb defektust találják meg

A kimerítő tesztelés lehetetlen

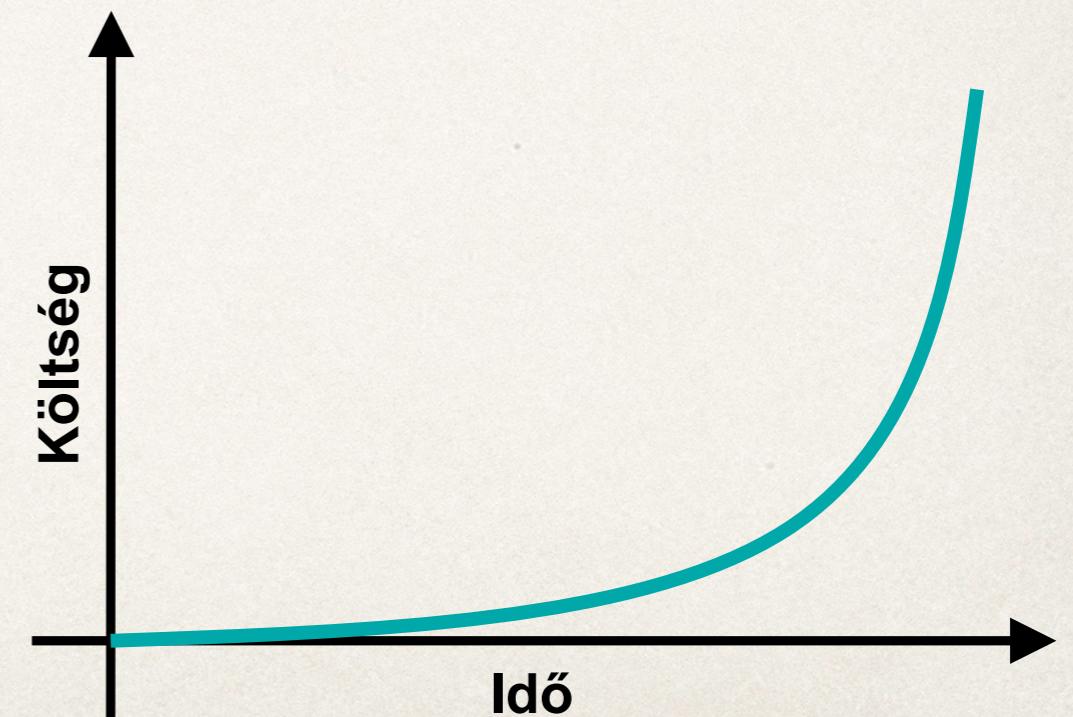
- ❖ Mindent (az összes adat és előfeltétel kombinációt) tesztelni a triviális esetek kivételével lehetetlen
- ❖ Például 4 karakter begépelésénél ez $256^4 = 2^{32}$ lehetőség (plusz a leütések közötti idők kérdése)
- ❖ Kockázatkezelés és prioritás segítségével kiválaszthatók a teszt fontosabb szempontjai

Kimerítő tesztelés (exhaustive testing):

- ❖ Egy olyan teszt, amelyben minden lehetséges adat-kombináció használva van
- ❖ Ez magában foglalja a szoftver állapotaiban/adataiban kódolt implicit adat-kombinációkat is.

Korai tesztelés

- ✿ Minél később kezdünk tesztelni, annál kevesebb időnk marad rá
- ✿ Amint a fejlesztési modell egyik fázisát befejeztük, a hozzá tartozó teszteket elkezdhetjük elkészíteni, és ezzel együtt a fázis eredményét is átnézzük
- ✿ Az adott fázisban elkövetett hibát még azelőtt észre kell venni, hogy az a következő fázisban is újabb defektusokat okozhatna
- ✿ Minél korábban el kell kezdeni a tesztelési tevékenységeket jól definiált célok mentén!



Hibafürtök megjelenése

- ✿ A defektusok eloszlása nem egyenletes a modulok között, mivel pl.:
 - ✿ komplexitás
 - ✿ gyakran változó kód
 - ✿ fejlesztők tapasztaltsága / tapasztalatlansága
- ✿ **Pareto-elv:** a problémák kb. 80%-a a modulok kb. 20%-ában található
 - ✿ A modulok kis aránya tartalmazza az átadás előtti tesztek során vagy az üzemeltetéskor fellépő meghibásodások nagy részét
- ✿ Várható hiba gyakoriság alapján koncentráljuk a ráfordításokat



Rovarirtó jelenség

- ✿ Ugyanannak a tesztnak a futtatása nem fog újabb hibákat találni, „immúnissá” válik a rendszer
- ✿ Két jellemző ok:
 1. A változatlan/ellenőrizetlen tesztek elavulhatnak (pl. a specifikáció változása miatt)
 2. Ha a tesztelési módszerekben nincs változás, a fejlesztők “hozzászoknak” a tesztekhez, és az adott hibákat elkerülik, de attól még más hibákat ugyanúgy elkövethetnek
- ✿ A féregirtó jelenség elkerüléséhez tehát a teszteseteket folyamatosan frissíteni, újítani kell, újabb és különböző típusú teszteket kell készíteni a szoftver vagy rendszer különböző részeihez

A tesztelés környezetfüggő

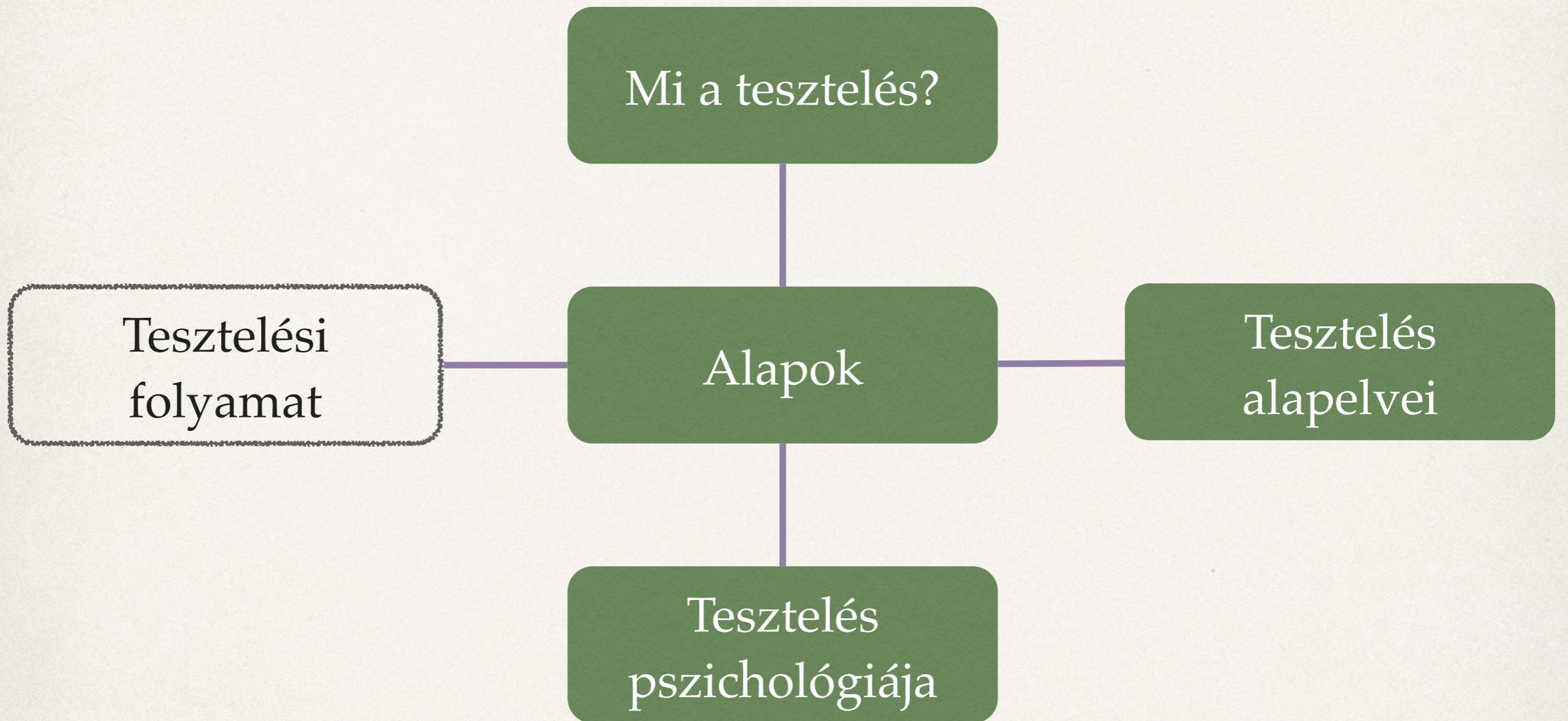
- ❖ Különböző körülmények között különböző tesztelési stratégiákra lehet szükség
 - ❖ MÁshogyan kell tesztelni egy információs weblapot mint egy online áruházat, egy légiforgalom-irányító rendszert mint egy hitelkalkulátort
- ❖ A kockázati tényező egy fontos szempont a teszt típusának meghatározásakor
 - ❖ Minél nagyobb a várható veszteség, annál többet kell költeni a tesztelésre
 - ❖ A szoftver biztonsági teszteléséhez például valószínűleg speciális szakértőre és/vagy eszközre lesz szükség

A hibamentesség téveszme

- ❖ Mi a valószínűbb?!
 - ❖ Rendszer tökéletes és hibátlan
 - ❖ A tesztelés nem volt elégsges
- ❖ A defektusok felfedezése és kijavítása nem segít, ha a rendszer használhatatlan és nem teljesíti a felhasználók elvárásait
- ❖ A verifikáció (~ a termék tesztelése valamely követelményekkel szemben) nem helyettesíti a validációt (~ a követelmények tesztelése)



Alapok::Pszichológia



Tesztelés pszichológiája



Tesztelés pszichológiája

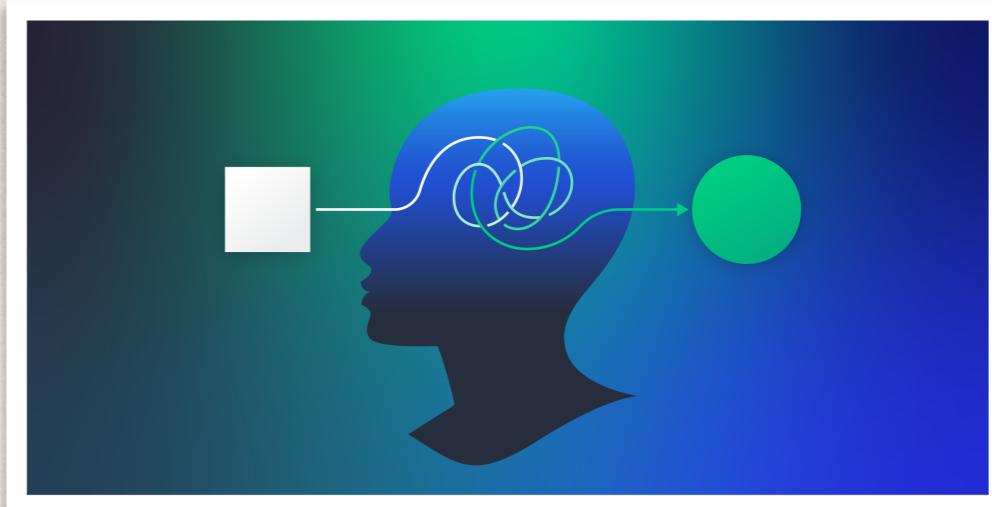
When developer
finds a bug



When tester
finds a bug



Emberi tényezők



- ❖ Hibás teszt futás vagy review comment:
 - ❖ Kritika megfogalmazása
 - ❖ A kérdés csak az, hogy a személy vagy rendszer felé?
- ❖ Pszichológia:
 - ❖ "Megerősítési torzítás"
 - ❖ "Felsőbbrendűségi illúzió"
 - ❖ "Kognitív disszonancia csökkentés"

A jó tesztelő tulajdonságai

- ❖ Alaposság, odafigyelés, kíváncsiság
- ❖ Jó kommunikációs készségek
- ❖ Analitikus és kritikus gondolkodás
- ❖ Tesztelői tudás!
 - ❖ Módszerek ismerete
 - ❖ Technikai tudás
 - ❖ Szakterület ismerete



Kommunikáció

- ✿ Fejlesztők és tesztelők nem egymás ellen, hanem egymás mellett a közös célért:
 - ✿ jobb szoftver!
- ✿ Tesztelés nem destruktív tevékenység
- ✿ Ne személyeskedjünk, a szoftver hibás, nem a fejlesztő
- ✿ Próbáljuk megérteni mások érzéseit; egy problémát úgy is meg lehet beszélni, hogy a beszélgetést mindenki pozitívan értékelje
- ✿ A megbeszélés végén ellenőrizzük, hogy pontosan értettünk minden, és minket is megértettek



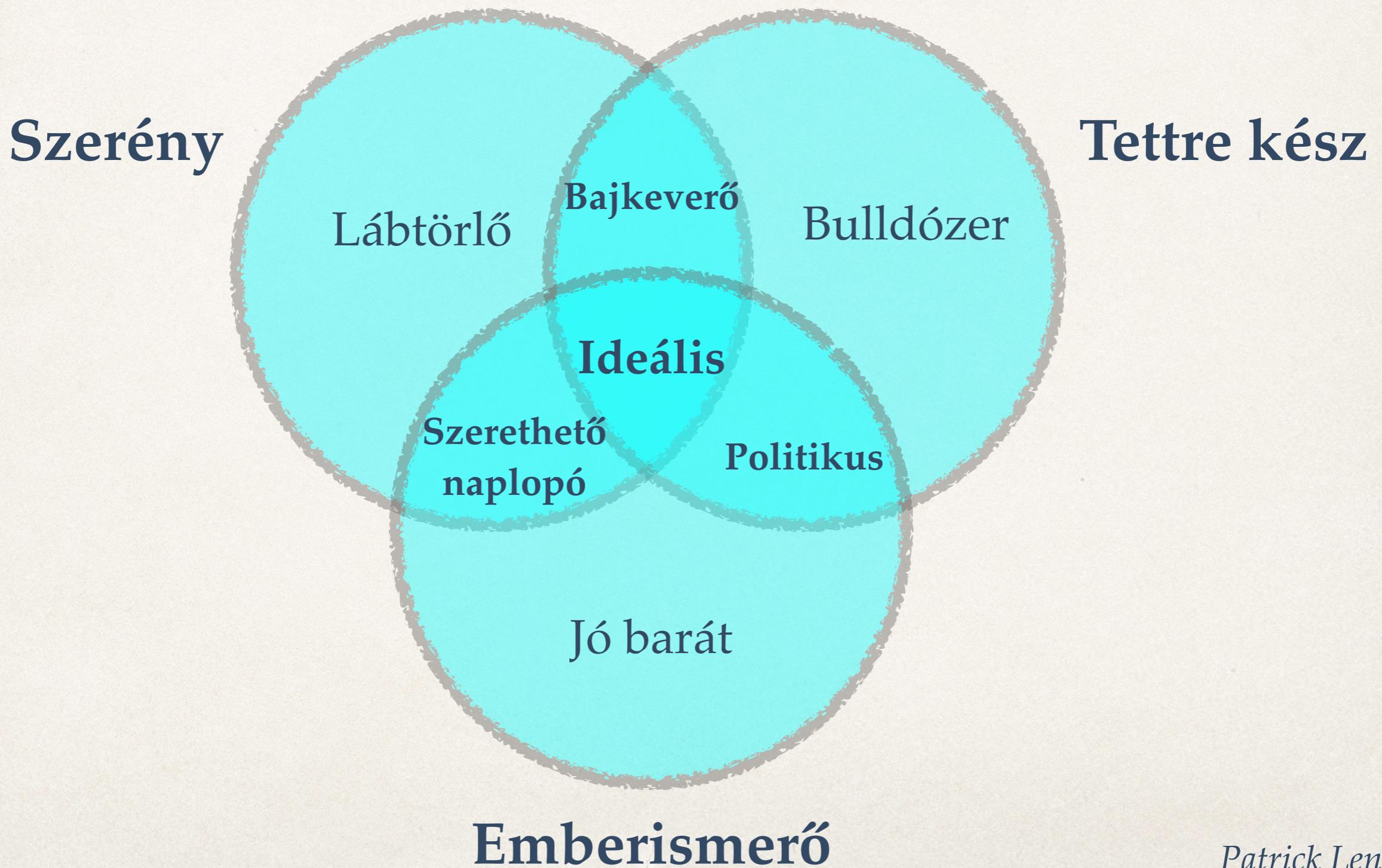
Csapatszemlélet

Whole team approach

- ✿ Keresztfunkcionális
 - ✿ “T” tudás modell
 - ✿ minden kompetencia jelen van
- ✿ Önszerveződő
 - ✿ Belső döntéshozatal, nincs “vezető”
- ✿ Csapat szintű felelősség
 - ✿ csapat célok vs. egyéni célok
- ✿ Szoros együttműködés

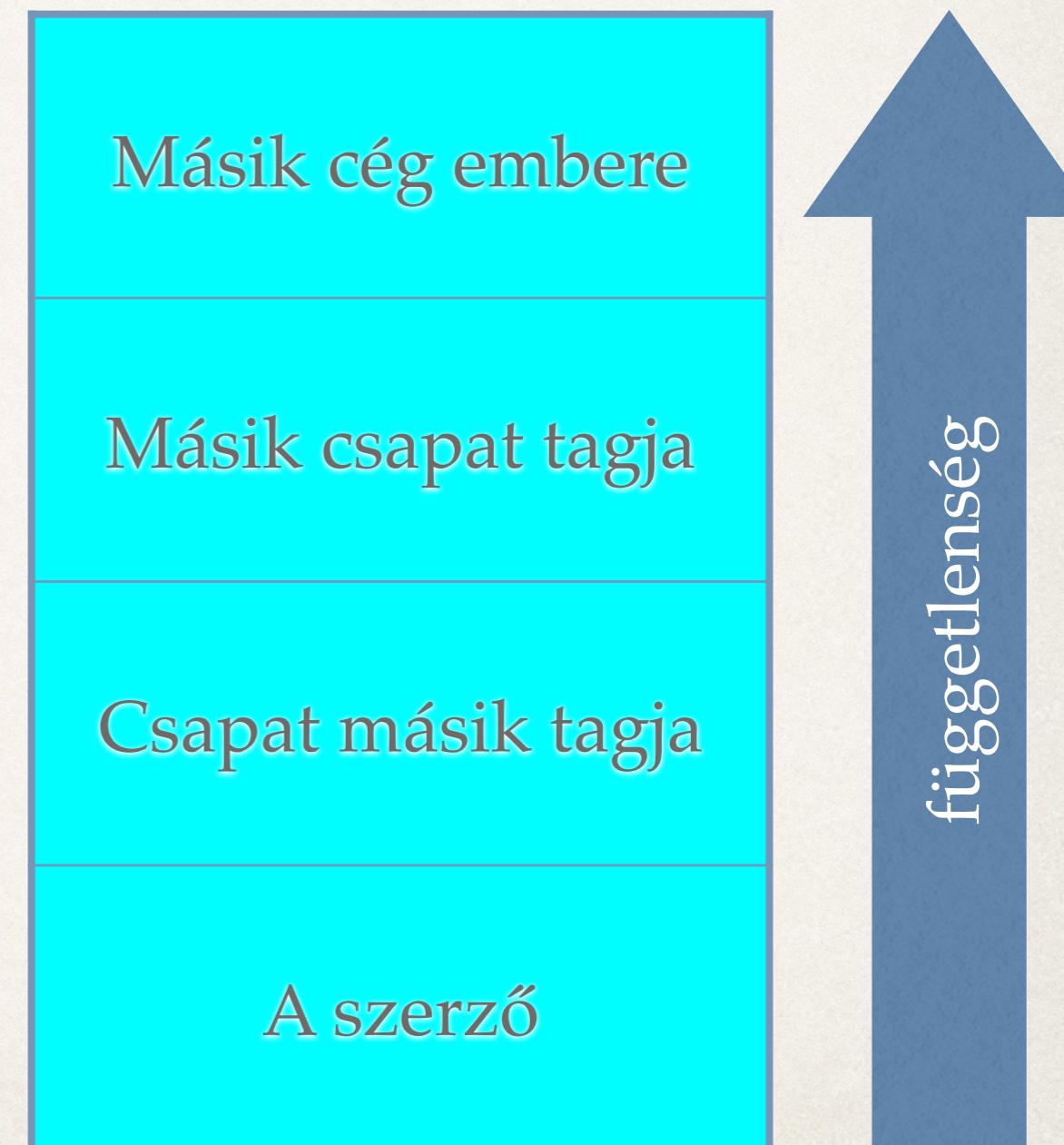


Az ideális csapatjátékos



Független tesztelés

- ❖ A tesztelésbe sokféle ember bevonható
- ❖ Saját kódot tesztelni nem hatékony
 - ❖ A készítő számára a hibák láthatatlanok
 - ❖ A hibakeresés természetesen szükséges
 - ❖ Korai szakaszban vannak előnyei
 - ❖ A függetlenségnek szó szerint ára van



Alapok::Folyamat



Tesztelési folyamat

A tesztfolyamatot befolyásoló környezeti tényezők:

- ❖ Életciklus-modell
- ❖ Projekt-módszertan
- ❖ Kockázatok
- ❖ Üzleti tevékenység
- ❖ Költségvetés, idő, szerződések, szabályok
- ❖ Szervezeti policy-k és gyakorlatok
- ❖ Belső és külső szabványok

Tesztelési folyamat:
tesztelési tevékenységek, melyek a
fejlesztési folyamatba ágyazódnak

Nyomonkölvethetőség

Traceability

- ❖ A tesztek és bármely munkatermék között definiálható (követelmény, kód, adatforrás, ...)
- ❖ Miben segít?
 - ❖ A változások hatásainak elemzésében
 - ❖ A tesztek auditálhatóvá tételeben
 - ❖ A teszt előrehaladási jelentések jobb megértésében
 - ❖ A tesztek technikai jellemzőinek minden résztvevő számára érthetővé tételeben

Szerepkörök a tesztelésben

1. Teszt menedzser

- Szervezői szerep: vezető, koordinátor (Agilis környezetben speciális)

2. Tesztelő

- Mérnöki, technikai szerep: elemző, végrehajtó

3. Egyéb szerepkörök

- Fejlesztő, üzemeltető, marketing, jog, stb.

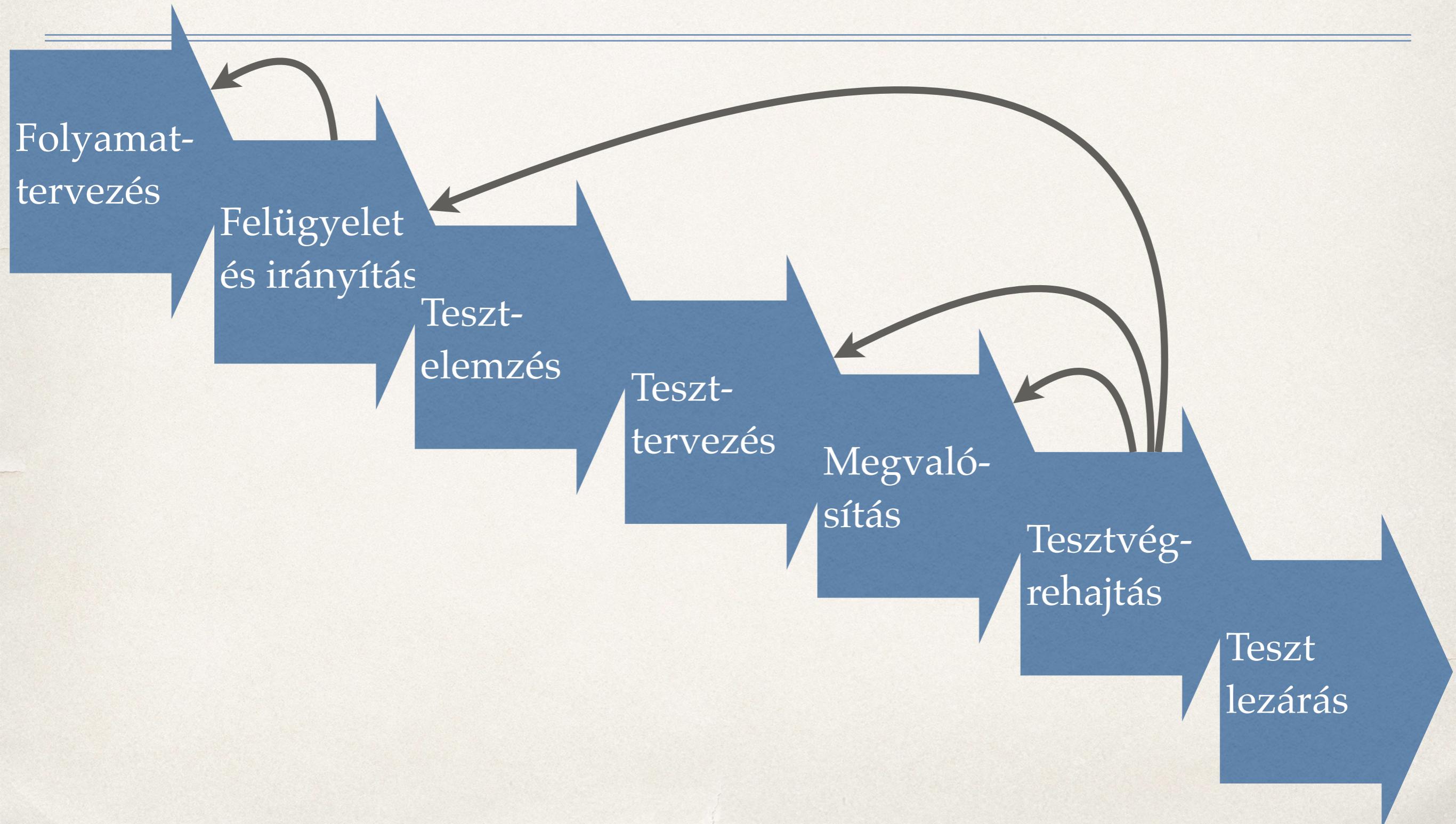
További részletek: 5. fejezetben

Tesztelés eredménytermékei

Testware

- ✿ minden tevékenységnek jól definiált eredménytermékei (munkatermékei) kell, hogy legyenek (ez a "tesztver")
- ✿ Azok dokumentálása, tárolása, stb. esetenként változó
- ✿ Sok eredményterméket dedikált eszköz segítségével állítják elő
- ✿ Nyomon követhetőség fontos a tesztbázis és a munkatermék között

Alapvető tesztelési folyamat



Folyamattervezés

- ❖ Mik a céljaink?
- ❖ Mit, hogyan és mikor fogunk tesztelni?
- ❖ Mik a kockázatok?
- ❖ Mik a feltételek (humán, SW, HW, ...)
- ❖ Mikor tekintjük teljesítettnek?
- ❖ A tesztelés befejezéséig folyamatosan zajlik

Munkatermékek:

- ❖ Teszt tervezek, tesztelési szintenként

Nyomonkövethetőség:

- ❖ Mik az egyes „döntések” okai, forrásai?

Teszt-felügyelet és irányítás

- ❖ A kontroll összehangolja a tervezett és a valós lépéseket, ha eltérnénk a tervtől
 - ❖ Az eredmények mérése és analízise
 - ❖ Az elvárt és valós folyamat összehasonlítása, a kilépési feltétel kiértékelése
 - ❖ Korrekciók végrehajtása, döntések, pl. újabb tesztek
- ❖ A tesztelés befejezéséig folyamatosan zajlik

Munkatermékek:

- ❖ Tesztjelentések (köztes, összefoglaló)

Teszt-elemzés

- ❖ Válaszok a „Mit teszteljünk?” kérdésre
- ❖ Fontosabb feladatok a lépés során:
 - ❖ Tesztbázis átnézése és elemzése
 - ❖ Tesztfeltételek meghatározása
 - ❖ Tesztvázlatokhoz feltételek gyűjtése
 - ❖ Nyomonkövethetőség meghatározása
 - ❖ Hibakeresés a tesztbázisban: kétértelműség, kihagyások, inkonziszencia, pontatlanságok, ellentmondások, felesleges részek

Tesztbázis:

- ❖ ami alapján a teszteket megtervezzük, pl.:
- ❖ specifikáció, terv, forráskód, tapasztalati tudás, stb.

Teszt-elemzés

Munkatermékek:

- ✿ Priorizált tesztfeltételek listája
- ✿ Tesztvázlat-kezdemények
- ✿ Tesztbázisban található hibák listája

Nyomon követhetőség:

- ✿ Az egyes tesztek honnan, mely követelményekből, forrásokból erednek?

(Műszaki) Teszt-tervezés

- ❖ Válaszok a „Hogyan teszteljünk?” kérdésre
- ❖ Magasszintű tesztesetek, teszthalmazok és “tesztverek” meghatározása
- ❖ Fontosabb feladatok a lépés során:
 - ❖ Tesztesetek tervezése és priorizálása
 - ❖ Tesztadatok meghatározása
 - ❖ Teszkörnyezet és infrastruktúra megtervezése

Munkatermékek:

- ❖ Tesztesetek
- ❖ Teszteset-halmazok
- ❖ Tesztadatok

Nyomonkövethetőség:

- ❖ Tesztesetek, tesztfeltételek és tesztbázis közötti kapcsolatok meghatározása

Tesztesetek (előzetes)

Teszteset (Test case) - Előfeltételek, bemenetek, tevékenységek, elvárt eredmények és utófeltételek halmaza, a tesztfeltételek alapján.

1. **Tesztfeltételek** meghatározása
(*mit* fogunk tesztelni?)
2. **Tesztesetek** megtervezése valamely technika segítségével
(*hogyan* fogjuk tesztelni?)
3. **Teszteljárások** (szkriptek) kidolgozása
(implementáció és végrehajtásra való előkészítés)

Teszt megvalósítás

- ❖ Válaszok a „Minden megvan, hogy teszteket futtassunk?” kérdésre

- ❖ Főbb tevékenységek:

- ❖ Teszteljárások fejlesztése és prioritálása
- ❖ Automatizálás, szkriptek írása
- ❖ Automatikus futtatás ütemezése
- ❖ Tesztkészletek létrehozása, prioritálás
- ❖ Tesztkörnyezet kiépítése és beállítása
- ❖ Tesztadatok előkészítése és betöltése
- ❖ Tesztkörnyezet, tesztadatok ellenőrzése

Munkatermékek:

- ❖ Tesztver
- ❖ Teszt eljárások
- ❖ Tesztkörnyezet
- ❖ Szkriptek

Nyomonkövethetőség:

- ❖ Tesztbázis, tesztfeltételek, tesztesetek, teszteljárások, szkriptek közötti kapcsolatok meghatározása

Teszt végrehajtás

- ❖ A tesztek „futtatása”
- ❖ Főbb tevékenységek:
 - ❖ Tesztelemek és tesztek verziójának rögzítése
 - ❖ Tesztek végrehajtása (kézi vagy automatikus)
 - ❖ Elvárt és kapott eredmények összehasonlítása
 - ❖ Rendellenességek elemzése, hibák jelentése
 - ❖ Teszt eredmények naplózása
 - ❖ Tesztek ismétlése

Munkatermékek:

- ❖ Teszt logok
- ❖ Tesztesetek eredményei
- ❖ Hibák

Nyomonkövethetőség:

- ❖ Tesztbázis, tesztfeltételek, tesztesetek, a tesztelt szoftver és a teszteredmények közötti kapcsolatok meghatározása, frissítése

Teszt lezárás

- ❖ Projekt mérföldköveknél (pl. release) hajtjuk végre

- ❖ Főbb tevékenységek:

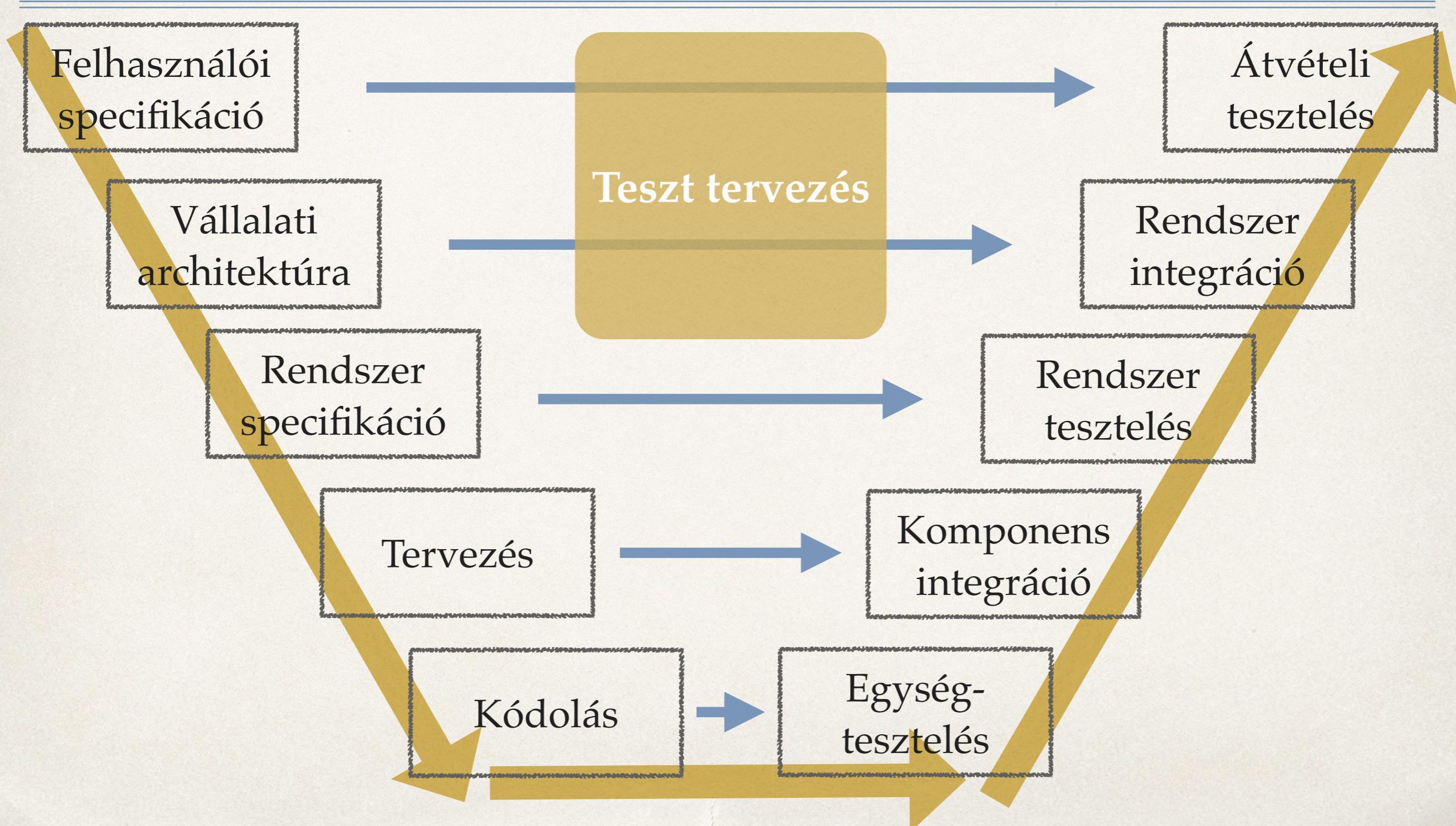
- ❖ Hibák állapotának ellenőrzése
- ❖ Összefoglaló tesztjelentés készítése
- ❖ Tesztkörnyezet, adatok, infrastruktúra archiválása

Munkatermékek:

- ❖ Összefoglaló tesztjelentés
- ❖ Változás-kérés
- ❖ Mentések
- ❖ Véglegesített tesztkörnyezet

- ❖ Tesztver átadása a karbantartó csapatnak
- ❖ Tesztfolyamat elemzése és fejlesztése

Tesztelési szintek (előzetes)



Tesztelestípusai (előzetes)

I. Funkcionális / nem-funkcionális teszt

2. Egyszeri / megismételt

3. Statikus / dinamikus

4. Fehérdoboz / feketedoboz teszt

- Terhelés-teszt
- Stressz-teszt
- Használhatósági teszt
- Biztonsági teszt
- Alfa / Béta teszt
- "Smoke" teszt
- Robosztussági, random teszt
- "Csimpánz" teszt
- Kombinatorikai módszerek
- Konfigurációs adatok tesztelése
- Teszt adatok tesztelése, "tesztek tesztelése", stb.