1. A tesztelés alapjai 180 perc

Kulcsszavak

emberi eredetű hiba, hiba, hibakeresés, kiváltó ok, lefedettség, meghibásodás, minőség, minőségbiztosítás, műszaki teszttervezés, tesztadat, tesztbázis, tesztcél, tesztelemzés, tesztelés, teszteljárás, teszteredmény, teszteset, tesztfeltétel, tesztfelügyelet, tesztirányítás, tesztlezárás, tesztmegvalósítás, teszt tárgya, teszttervezés, tesztvégrehajtás, tesztver, validáció, verifikáció

Tanulási célok az első fejezethez:

1.1 Mi a tesztelés?

AK-1.1.1 (K1) Azonosítson tipikus tesztcélokat

AK-1.1.2 (K2) Különböztesse meg a tesztelést és a hibakeresést

1.2 Miért szükséges a tesztelés?

AK-1.2.1 (K2) Szemléltesse a tesztelés szükségességét

AK-1.2.2 (K1) Idézze fel a tesztelés és a minőségbiztosítás közötti kapcsolatot

AK-1.2.3 (K2) Tegyen különbséget a kiváltó ok, az emberi eredetű hiba, hiba és meghibásodás között

1.3 Tesztelési alapelvek

AK-1.3.1 (K2) Ismertesse az általános tesztelési alapelveket (A hét tesztelési alapelv)

1.4 Teszttevékenységek, tesztver és teszt szerepkörök

AK-1.4.1 (K2) Foglalja össze a különböző teszttevékenységeket és feladatokat

AK-1.4.2 (K2) Ismertesse a kontextus tesztfolyamatra gyakorolt hatását

AK-1.4.3 (K2) Tegyen különbséget a teszttevékenységeket támogató tesztverek között

AK-1.4.4 (K2) Ismertesse a nyomonkövethetőség fenntartásának fontosságát

AK-1.4.5 (K2) Hasonlítsa össze a tesztelés különböző szerepköreit

1.5 Alapvető készségek és bevált gyakorlatok a tesztelésben

AK-1.5.1 (K2) Adjon példákat a teszteléshez szükséges készségekre

AK-1.5.2 (K1) Idézze fel a teljes csapat megközelítés előnyeit

AK-1.5.3 (K2) Tegyen különbséget a tesztelés függetlenségének előnyei és hátrányai között

1.1 Mi a tesztelés?

A szoftverrendszerek az élet szerves részét képezik. A legtöbb embernek volt már tapasztalata olyan szoftverrel, ami nem az elvártaknak megfelelően működött. A helytelenül működő szoftver sok problémához vezethet, beleértve a pénz- és időveszteséget, az üzleti hírnév elvesztését, illetve extrém esetekben akár sérülést vagy halált is eredményezhet. A szoftvertesztelés a szoftver minőségét értékeli, illetve segít a szoftver működés közbeni meghibásodási kockázatának csökkentésében.

A szoftvertesztelés olyan tevékenységek összessége, amelyek célja a hibák feltárása és a szoftvertermékek minőségének értékelése. Az ilyen termékeket a tesztelés folyamán a teszt tárgyának nevezzük. Gyakori tévhit a teszteléssel kapcsolatban, hogy az csak tesztek végrehajtásából áll (azaz a szoftver futtatásából és a teszteredmények ellenőrzéséből). A szoftvertesztelés azonban más tevékenységeket is magában foglal, valamint hozzá kell igazítani az adott szoftverfejlesztési életciklushoz (lásd a 2. fejezetet).

Egy másik gyakori félreértés a teszteléssel kapcsolatban, hogy teljes mértékben a teszt tárgyának verifikációjára fókuszál. A tesztelés valóban kiterjed a verifikációra, azaz a meghatározott követelményeknek való megfelelés ellenőrzésére, azonban tartalmazza a validációt is, amely azt ellenőrzi, hogy a rendszer megfelel-e a felhasználói, illetve más érdekelt felek igényeinek a működési környezetben.

A tesztelés lehet dinamikus vagy statikus. A dinamikus tesztelés magában foglalja a szoftver végrehajtását, míg a statikus tesztelés nem. A statikus tesztelés a felülvizsgálatot (lásd a 3. fejezetet) és a statikus elemzést jelenti. A dinamikus tesztelés különböző típusú teszttechnikákat és tesztmegközelítéseket használ a tesztesetek tervezésére (lásd a 4. fejezetet).

A tesztelés nem csak technikai tevékenység. Alaposan meg kell tervezni, menedzselni, becsülni, nyomon követni és ellenőrizni (lásd az 5. fejezetet).

A tesztelők eszközöket használnak (lásd a 6. fejezetet), de fontos megjegyezni, hogy a tesztelés nagyrészt intellektuális tevékenység, amely speciális ismereteket, elemzőkészségeket, valamint kritikus és rendszer-szemléletű gondolkodást kíván meg a tesztelőktől (Myers 2011, Roman 2018).

Az ISO/IEC/IEEE 29119-1 szabvány további információkat nyújt a szoftvertesztelési koncepciókról.

1.1.1 Tesztcélok

A tesztelés jellemző céljai az alábbiak:

- Követelmények, felhasználói történetek, műszaki tervek és a kód kiértékelése
- Meghibásodások okozása és hibák megtalálása
- A szükséges lefedettség biztosítása a teszt tárgyának
- A nem megfelelő szoftverminőség kockázati szintjének csökkentése
- Annak igazolása, hogy bizonyos követelmények teljesültek-e
- Annak igazolása, hogy a teszt tárgya megfelel a szerződésben foglalt, jogi vagy szabályozási követelményeknek
- Információ biztosítása az érdekelt feleknek, hogy ezáltal megalapozott döntéseket hozhassanak
- Bizalom kiépítése a teszt tárgyának minőségével kapcsolatban
- Annak validálása, hogy a teszt tárgya elkészült és az érdekelt felek elvárásainak megfelelően működik.

A tesztelés céljai a környezettől függően változhatnak, mely magába foglalja a tesztelés alatt álló munkaterméket, a tesztszintet, a kockázatokat, a követett szoftverfejlesztési életciklusmodellt és az üzleti környezethez kapcsolódó tényezőket, mint például a vállalati struktúra, a versenyszempontok vagy a piacra jutás ideje.

1.1.2 Tesztelés és hibakeresés

A tesztelés és a hibakeresés különálló tevékenységek. A tesztelés olyan meghibásodásokat indukálhat, amelyeket a szoftver hibái okoznak (dinamikus tesztelés), vagy közvetlenül találhat hibákat a teszt tárgyában (statikus tesztelés).

Amikor a dinamikus tesztelés (lásd a 4. fejezetet) meghibásodást vált ki, a hibakeresés ennek a meghibásodásnak az okait (hibák) tárja fel, elemzi és megszünteti ezen okokat. A tipikus hibakeresési folyamat ebben az esetben a következőket tartalmazza:

- A meghibásodás reprodukálása
- Diagnózis (a kiváltó ok megtalálása)
- Az ok kijavítása

Ezt követően az ellenőrző tesztelés során ellenőrzik, hogy a javítások orvosolták-e a problémát. Lehetőség szerint az ellenőrző tesztelést ugyanaz a személy végzi el, aki az első tesztet is elvégezte. Később regressziós tesztelés is elvégezhető annak ellenőrzésére, hogy a javítások nem okoztak-e hibákat a teszt tárgyának más részein (az ellenőrző és a regressziós teszteléssel kapcsolatos további információkért lásd a 2.2.3. fejezetet).

Amikor a statikus tesztelés hibát azonosít, a hibakeresés közvetlenül annak eltávolítását célozza. Nincs szükség reprodukálásra vagy diagnózisra, mivel a statikus tesztelés közvetlenül találja meg a hibákat, és nem okozhat meghibásodást (lásd a 3. fejezetet).

1.2 Miért szükséges a tesztelés?

A tesztelés, mint a minőségellenőrzés egyik formája, segít a kitűzött célok elérésében a meghatározott hatókör-, idő-, minőség- és költségvetési korlátok között. A tesztelés hozzájárulása a sikerhez nem korlátozódhat a tesztcsoport tevékenységeire. Bármely érdekelt fél felhasználhatja tesztelési készségeit, hogy közelebb vigye a projektet a sikerhez. A komponensek, rendszerek és a kapcsolódó dokumentáció tesztelése segít a szoftverhibák azonosításában.

1.2.1 A tesztelés hozzájárulása a sikerhez

A tesztelés költséghatékony módszert kínál a hibák észlelésére. Ezek a hibák ezután eltávolíthatók (hibakereséssel – ami nem tesztelési tevékenység), így a tesztelés közvetetten hozzájárul a teszt tárgyainak jobb minőségéhez.

A tesztelés lehetőséget biztosít, hogy közvetlenül értékeljük a teszt tárgyának minőségét a szoftverfejlesztési életciklus különböző szakaszaiban. Ezek az intézkedések egy nagyobb projektmenedzsment tevékenység részét alkotják, hozzájárulva a szoftverfejlesztési életciklus következő szakaszába lépésről hozott döntésekhez, mint például a kiadási döntés.

A tesztelés segítségével a felhasználók közvetett módon vehetnek részt a fejlesztési projektben. A tesztelők biztosítják, hogy a felhasználók igényeit a fejlesztési életciklus során figyelembe vegyék. A másik lehetőség, hogy a felhasználók egy reprezentatív körét bevonják a fejlesztési projektbe, viszont ez a magas költségek és a megfelelő felhasználók elérhetőségének hiánya miatt általában nem megoldható.

A tesztelést a szerződéses vagy jogi követelményeknek, illetve a szabályozói szabványoknak való megfelelés érdekében is megkövetelhetik.

1.2.2 Tesztelés és minőségbiztosítás (QA)

Bár az emberek a „tesztelés” és a „minőségbiztosítás” (vagy QA) kifejezéseket szinonimaként használják, a két fogalom nem ugyanaz. A tesztelés a minőségellenőrzés (vagy QC) egyik formája.

A minőségellenőrzés egy termék-orientált, javító jellegű megközelítés, amely a megfelelő minőségi szint elérését támogató tevékenységekre összpontosít. A tesztelés a minőségellenőrzés egyik fontos része, további részei a formális módszerek (modellellenőrzés és a helyesség igazolása), a szimuláció és a prototípus készítés.

A minőségbiztosítás folyamatorientált, megelőző jellegű megközelítés, amely a folyamatok megvalósítására és fejlesztésére összpontosít. Az alapelve, hogy ha egy jó folyamatot helyesen követnek, akkor jó terméket hoz létre. A minőségbiztosítás mind a fejlesztési, mind a tesztelési folyamatra értelmezhető, és a projektben mindenki felelőssége.

A teszteredményeket a minőségbiztosítás és a minőségellenőrzés is használja. A minőségellenőrzésben a hibák javítására szolgálnak, míg a minőségbiztosításban visszajelzést adnak a fejlesztési és tesztelési folyamatok teljesítményéről.

1.2.3 Emberi eredetű hibák, hibák, meghibásodások és kiváltó okok

Emberi hibákat (angolul error, mistake) emberi lények követhetnek el. Ezek hibákat (angolul defect, fault, bug) eredményeznek, amelyek meghibásodásokhoz (angolul failure) vezethetnek. Az emberi eredetű hibáknak különféle okai lehetnek, például az idő szorossága, a munkatermékek, folyamatok, infrastruktúra vagy interakciók összetettsége, vagy egyszerűen a fáradtság, esetleg a nem megfelelő képzettség.

Hibák a dokumentációban találhatók, például a követelményspecifikációban vagy a tesztszkriptben, a forráskódban vagy egy kiegészítő termékben, például egy build fájlban. A szoftverfejlesztési életciklus korai szakaszaiban készült termék hibái, ha nem vesszük észre őket, gyakran hibás termékekhez vezetnek az életciklus későbbi szakaszában. Ha egy kódhiba végrehajtásra kerül, előfordulhat, hogy a rendszer nem tudja megtenni azt, amit tennie kell, vagy megtesz valamit, amit nem kellene, és ezek meghibásodást okoznak. Egyes hibák végrehajtása mindig meghibásodást okoz, míg mások csak adott körülmények között, megint mások pedig soha nem vezetnek meghibásodáshoz.

A meghibásodásoknak nem a hibák és az emberi eredetű hibák a kizárólagos okai. Környezeti feltételek is okozhatják, például amikor a sugárzás vagy az elektromágneses mező hibákat okoz az alapszoftverben.

A kiváltó ok egy probléma (pl. egy hibához vezető helyzet) előfordulásának az alapvető oka. A kiváltó okok azonosítása a kiváltó okok elemzésével történik, amelyet általában meghibásodás vagy hiba azonosításakor hajtanak végre. Általánosan elfogadott, hogy a további hasonló meghibásodások vagy hibák megelőzhetők vagy gyakoriságuk csökkenthető a kiváltó ok kezelésével, például annak eltávolításával.

1.3 Tesztelési alapelvek

Számos tesztelési alapelvet javasoltak az elmúlt években, amelyek általános, mindennemű tesztelésre vonatkozó irányelveket adnak. A tanterv hét ilyen alapelvet ismertet.

1. **A tesztelés a hibák jelenlétét mutatja, nem a hiányukat**

A tesztelés kimutathatja a hibák jelenlétét a teszt tárgyában, de azt nem képes igazolni, hogy nincsenek hibák (Buxton 1970). A teszteléssel csökken annak az esélye, hogy a teszt tárgyában felfedezetlen hibák maradnak, de még ha nem is találnak hibát, az nem bizonyítja a rendszer helyességét.

2. **Nem lehetséges kimerítő teszt**

Mindenre kiterjedő tesztelés a triviális eseteket leszámítva nem lehetséges (Manna 1978). Ahelyett, hogy megkísérelnénk a kimerítő tesztelést, a tesztelési erőforrások összpontosításához alkalmazzunk

teszttechnikákat (lásd a 4. fejezetet), teszteset-priorizálást (lásd az 5.1.5 fejezetet) és kockázatalapú tesztelést (lásd 5.2 szakasz).

3. **A korai tesztelés időt és pénzt spórol**

A folyamat korai szakaszában eltávolított hibák nem okoznak későbbi hibákat a származtatott munkatermékekben. A minőség költsége csökken, mivel a szoftverfejlesztési életciklus későbbi szakaszaiban kevesebb meghibásodás következik be (Boehm 1981). A hibák korai felismerése érdekében mind a statikus tesztelést (lásd a 3. fejezetet), mind a dinamikus tesztelést (lásd a 4. fejezetet) a lehető legkorábban el kell kezdeni.

4. **Hibafürtök megjelenése**

A kiadást megelőző tesztelés során megtalált hibák többsége rendszerint néhány rendszer komponensben koncentrálódik, vagy ezen modulok felelősek a működési meghibásodások többségéért (Enders 1975). Ez a jelenség a Pareto-elvet szemlélteti. A megjósolt hibafürtök és a tesztelés vagy működés során ténylegesen megfigyelt hibafürtök fontos bemenetként szolgálnak a kockázatalapú tesztelés számára (lásd az 5.2 fejezetet).

5. **A tesztek elkopnak**

Ha ugyanazokat a teszteket ismételjük számos alkalommal, akkor ezen tesztek nem lesznek hatékonyak új hibák megtalálásában (Beizer 1990). Ahhoz, hogy elkerüljük ezt a hatást, a létező teszteket és tesztadatokat módosítani kell, illetve új teszteket kell írni. Ugyanakkor, néhány esetben - például az automatikus regressziós tesztelés esetében - előnyös oldala is van ugyanazon tesztek ismétlésének (lásd a 2.2.3 fejezetet).

6. **A tesztelés függ a körülményektől**

A tesztelésnek nincs egyetlen univerzálisan alkalmazható megközelítése. A tesztelés különböző környezetekben eltérő módon történik (Kaner 2011).

7. **A hibamentesség téveszméje**

Tévedés (vagy tévhit) azt várni, hogy a szoftververifikáció biztosítja a rendszer sikerét. Az összes meghatározott követelmény alapos tesztelése és a feltárt hibák kijavítása még mindig produkálhat olyan rendszert, amely nem felel meg a felhasználók igényeinek és elvárásainak, amely nem segíti az ügyfél üzleti céljainak elérését, és a versenytársak rendszereihez képest gyengébb. A verifikáció mellett a validálást is el kell végezni (Boehm 1981).

1.4 Teszttevékenységek, tesztver és tesztszerepkörök

A tesztelés függ a környezettől, de magas szinten léteznek gyakori teszttevékenységek, amik nélkül kevésbé valószínű, hogy a tesztelés eléri a tesztcélokat. Ezen teszttevékenységek alkotják a tesztfolyamatot. A tesztfolyamat különböző tényezők alapján az adott helyzetre szabható. Általában az adott helyzetre vonatkozó teszttervezés részeként dől el, hogy a tesztfolyamat melyik teszttevékenységeket foglalja magában, hogyan implementálják ezeket, és mikor valósulnak meg.

A következő szakaszok a tesztfolyamat általános vonatkozásait írják le a teszttevékenységek és -feladatok, a környezet hatása, a tesztverek, a tesztbázis és a tesztverek közötti nyomonkövethetőség, valamint a tesztelési szerepkörök tekintetében.

Az ISO/IEC/IEEE 29119-2 szabvány további információkat nyújt a tesztfolyamatokról.

1.4.1 Teszttevékenységek és -feladatok

A tesztfolyamat általában a lentebb bemutatott fő tevékenységcsoportokból áll. Bár ezen tevékenységek közül több logikailag szekvenciálisnak tűnhet, gyakran iteratív módon vagy párhuzamosan kerülnek megvalósításra. Ezeket a teszttevékenységeket gyakran szükséges lehet a rendszerhez vagy a projekthez igazítani.

A **teszttervezés** részét képezi a tesztcélok meghatározása, majd egy olyan megközelítés kiválasztása, amely a legjobban eléri ezeket a célokat a környezet által támasztott korlátokon belül. A teszttervezést bővebben az 5.1 fejezetben tárgyaljuk.

Tesztfelügyelet és -irányítás. A tesztfelügyelet magában foglalja minden teszttevékenység folyamatos ellenőrzését, valamint az aktuális és a tervezett előrehaladás összehasonlítását. A tesztirányítás a teszt célok eléréséhez szükséges tevékenységeket foglalja magában. A tesztfelügyelet és -irányítás az 5.3 fejezetben kerül bővebben kifejtésre.

A **tesztelemzés** során a tesztbázist elemezzük annak érdekében, hogy azonosítsuk a tesztelhető funkcionalitásokat, valamint meghatározzuk és priorizáljuk a kapcsolódó tesztfeltételeket a kapcsolódó kockázatokkal és azok kockázati szintjével együtt (lásd az 5.2 fejezetet). Értékeljük a tesztbázist és a teszt tárgyát, hogy azonosítsuk a bennük előforduló hibákat, és megállapítsuk a tesztelhetőségüket. A tesztelemzést gyakran támogatják a teszttechnikák alkalmazásával (lásd a 4. fejezetet). A tesztelemzés választ ad a „mit teszteljünk?” kérdésre, mint mérhető lefedettségi kritérium.

Műszaki teszttervezés során a tesztfeltételekből teszteseteket és egyéb tesztvereket alakítunk ki (például tesztvázlatokat). Ez a tevékenység gyakran magában foglalja a lefedettségi elemek azonosítását, amelyek útmutatóul szolgálnak a tesztesetek bemeneteinek meghatározásához. Teszttechnikákat (lásd a 4. fejezetet) használhatunk a tevékenység támogatására. A teszttervezés szintén magában foglalja a tesztadatokra vonatkozó követelmények meghatározását, a tesztkörnyezet megtervezését és egyéb szükséges infrastruktúrák és eszközök azonosítását is. A teszttervezés választ ad a „hogyan teszteljünk?” kérdésre.

A **tesztmegvalósítás** során létrehozzuk vagy beszerezzük a tesztvégrehajtáshoz szükséges tesztvert (például a tesztadatokat). A tesztesetek teszteljárásokba szervezhetők, és gyakran tesztkészletté állnak össze. Kézi és automatizált tesztszkriptek jönnek létre. A teszteljárások prioritást kapnak, és bekerülnek a tesztvégrehajtási ütemtervbe a hatékony tesztvégrehajtás érdekében (lásd az 5.1.5. fejezetet). Elkészül a tesztkörnyezet, melyet ellenőrzünk, hogy megfelelően legyen beállítva.

A **tesztvégrehajtás** során a teszteket a tesztvégrehajtási ütemtervben meghatározott sorrendben hajtjuk végre (tesztfuttatás). A teszt végrehajtása lehet manuális vagy automatizált. A tesztvégrehajtás számos formát ölthet, beleértve a folyamatos tesztelést vagy a páros tesztelési szakaszokat. A tényleges teszteredményeket összehasonlítjuk a várt eredményekkel. A teszteredmények naplózásra kerülnek. Az anomáliákat elemezzük, ezáltal azonosítjuk valószínű okukat. Ez az elemzés lehetővé teszi számunkra, hogy a megfigyelt meghibásodások alapján jelentsük az anomáliákat (lásd az 5.5. fejezetet).

A **tesztlezárás**hoz kapcsolódó tevékenységeket projektmérföldkövek elérésekor hajtjuk végre (például kiadás, egy tesztszint befejezése), a lezáratlan hibákra változtatáskérést vagy termék-teendőlista elemet vehetünk fel. Minden olyan tesztvert, amely hasznos lehet a jövőben, azonosítunk és archiválunk, vagy átadunk a megfelelő csapatoknak. A tesztkörnyezetet egy előre meghatározott állapotba állítjuk. A teszttevékenységeket elemezzük, hogy meghatározzuk a jövőbeli iterációkhoz, kiadásokhoz és projektekhez szükséges változtatásokat (lásd a 2.1.6. fejezetet). Összefoglaló tesztjelentés készül, amelyet az érdekelt felek is megismernek.

1.4.2 Tesztfolyamat és környezete

A tesztelést nem elszigetelten végzik. A tesztelési tevékenységek a szervezeten belüli fejlesztési folyamatok szerves részét képezik. A tesztelést az érdekeltek finanszírozzák, és végső célja, hogy segítse az érdekeltek üzleti igényeinek kielégítését. Ennek következtében a tesztelés kivitelezése számos környezeti tényezőtől függ, beleértve:

- Érdekelt felek (igények, elvárások, követelmények, együttműködési hajlandóság, stb.)
- Csapattagok (készségek, ismeretek, tapasztalati szint, elérhetőség, képzési igények, stb.)