

2.6 | Szoftverek korszerű tesztelése – 1.5 | szabványok, módszerek

*Tárgyszavak: szoftver; teszt; szabvány; minőségbiztosítás;
minőségirányítás; szabvány.*

A szoftverek előállítása a hajdani intuitív, művelői által gyakran „művészek” mondott tevékenységből mára ipari tevékenységgé vált. Ennek megfelelően egyre nagyobb a specializáció: egyre inkább csak egy-egy részterületet művelnek a munkatársak – hol van már az az idő, amikor egyetlen programozó találta ki, tervezte meg, programozta le, tesztelte és telepítette a programot, valamint ő végezte el az ügyfélnél esedékes összes feladatot, a paraméterek beállításától az oktatáson át a mindig fellépő hibák következményeinek kijavításáig.

A minőség kérdése minden iparágban előtérbe került az utóbbi években – nem kivétel ez alól a szoftvergyártás sem. A konkurencia és a gazdasági vezetés együttes nyomása arra ösztönzi az informatikusokat, hogy minél gyorsabban a piacra dobják a szoftvertermékeket. Ez korábban sok esetben oda vezetett, hogy tesztetetlen vagy alig tesztelt állapotú programokkal találkoztak a felhasználók. Ez azonban mára már a múlté: a szoftvergyártók megtapasztalták (és ez azon kevés dolgok egyike, amelyben az egymással élesen konkuráló cégek vezetői egyetértenek), hogy a sok reklamáció feldolgozása, a garanciális igények kielégítésének költségei és a cég piaci megítélésének romlása sokkal nagyobb költségeket okoz, mint amekkora előnyt a konkurenciát megelőző piacra jutás jelent. A szoftverek minőségének témaköre, a tesztelés és annak minősége egyre jelentősebb szerepet játszik a szoftvercégek életében. A professzionális, módszeres és sokféle segédeszköz bevetésével lefolytatott tesztelés egyes becslések szerint a fejlesztési költségek 25%-át is felemésztheti. Ez az összeállítás a vonatkozó szabványok fényében ismerteti a szoftverek minőségbiztosításának – különösen tesztelésének – módszereit és eljárásait.

Tesztelési módszerek és eljárások

Kezdetben a programozók tesztelték saját programjaikat. Ez a módszer a későbbiekben (ellentétben a később ismertető „black box” módszerrel) a „white box” módszer nevet kapta. Ennél ugyanis a program és az adatbázisok belső felépítésének, struktúrájának ismeretében határozzák meg a vizsgálandó funkciókat és a tesztelés körülményeit, a tesztadatokat. A programozó ismeri a legjobban a rendszer minden ágát, így ő tud a legjobban olyan teszteseteket (test case) kitalálni, amelyek a kód minden ágát-bogát, az egyes programágazások minden változatát kipróbálják. Itt tehát a vizsgálandó objektum ismeretében tervezik meg a tesztelés folyamatát és paramétereit. Ennek a módszernek nagy előnye a kis ráfordításigény, további pozitívuma a kevés összehangolás szükségessége: a programozó magával „beszéli meg” a teszt eredményeit, és ő küszöböli ki a talált hibákat (szakzsargonban bugokat, a javítási folyamat neve debug).

Az előnyök forrása okozza egyben a módszer korlátait, hibáit is: a hibák okozójának kell azokat megtalálni. Még a legnagyobb korrektséget feltételezve is megvannak ennek a korlátai: a programozó gondolkodásmódja meghatározott, ezért bizonyos irányokba nem fordul, egyes esetekre nem gondol. Másként fogalmazva: hajlamos arra, hogy a teszt tervezésénél megkerülje a hibákat („körbeteszte” azokat anélkül, hogy észrevenné). A tesztelés tehát nem objektív. Ennek bizonyos mértékű kiküszöbölését lehet elérni, ha több, egy projekten dolgozó programozó együtt teszteli a szoftver egyes részeit. Az előző módszer számos előnye megmarad, a ráfordítások csak kevéssé nőnek, a programnyelvet, a fejlesztőeszközöket a fejlesztők egyformán ismerik, nagyjából kiismerik magukat a projekt többi részében is. Növekszik viszont az összehangolás szükségessége, és ennek költsége is. A programozók együttműködése ugyanakkor szinergia-hatásokkal is jár, ötleteket adhatnak egymásnak, ugyanakkor még mindig csak fejlesztők végzik a tesztelést, a végfelhasználók szempontjai továbbra is háttérbe szorulnak. A módszer átmenetinek tekinthető a független tesztelő által végzett ellenőrzés irányába.

Főleg a közepes méretű és nagyobb szoftverházak és rendszerintegrátorok körében terjed az utóbbi időben a független tesztelés, tesztlaborok létesítése. Ezek fejlett módszertan alapján dolgoznak, a vezetők és munkatársak mély minőség-ellenőrzési és tesztelési szaktudás birtokában vannak, jelentős hardver- és tesztelő célszoftver-ráfordítások nyomán profi eszközeik vannak. A tesztek itt a vizsgálandó programok

felépítésének ismerete nélkül, a dokumentációk, specifikációk alapján, tehát a rendszerrel szemben támasztott elvárások szerint valósulnak meg. Az itt alkalmazott módszertan a „*black box*” tesztelés, ahol tehát a próbadarab fekete doboz, ismeretlen a tesztelését végzők számára.

A jelenséget másfelől is meg lehet közelíteni, illetve a programtesztelés kétféle alapállás szerint végezhető. Az egyik az ún. *pozitív teszt*, itt azt ellenőrzik, hogy normál adatokkal, „jóindulatú” kezelőt feltételezve teljesíti-e a szoftver az elvárásokat, a specifikáció szerinti eredményeket hozza-e „elszállás” nélkül. Ennek tipikus esete az imént vázolt módszerek közül az első, a programozók öntesztje. A programnak ugyanakkor működnie kell szélsőséges adatokkal is, el nem várható, sőt meg nem engedett kezelői beavatkozások mellett is. Ez a *negatív teszt* esete, ahol a tesztelő azon igyekszik, hogy zavarba hozza, hibás működésbe kergetse a programot. Közismert, hogy a fejlesztőnek eszébe sem jut olyasmi, amit a felhasználó első ismerkedése során képes a szoftverrel művelni. A negatív tesztek ezért igazán csak a független tesztrésztelők tudják jól megvalósítani.

Szabványok

A szoftverek tesztelésénél két szabvány előírásai adnak megfelelő útmutatót. A következők ezek főbb jellemzőit taglalják.

DIN EN ISO/IEC 17025:2005-05

A DIN EN ISO/IEC 17025:2005-05 szabvány címe „Általános követelmények a teszt- és kalibráló laboratóriumok kompetenciáival szemben”. A szabvány alapvetően nem a szoftverek tesztelését hivatott szabályozni, de alapelvei elég általánosak ahhoz, hogy erre a területre is megfelelő útmutatót adjanak. Az elvek megvalósításának módjára nincs előírás, azt a tesztlabor létrehozóira és vezetőire bízva. Íme néhány fontosabb a szabvány elvei közül:

- A vizsgálati laboratóriumnak függetlennek és részrehajlástól mentesnek kell lennie, a mérési eredményeket nem befolyásolhatják esetleges érdekek vagy konfliktusok.
- A megbízók védelme érdekében gondoskodni kell az adat- és információbiztonságról.
- Minőségirányítási rendszert kell folyamatosan működtetni, az eljárásokat és módszereket írásban, minőségirányítási kézikönyv formájában rögzíteni kell. Ennek kockázatkezelési tervet is magá-

ban kell foglalnia, valamint az ebből levezetett megelőző intézkedéseket és a folyamatok javítására irányuló törekvéseket is.

- A minőségirányítási rendszer előírásainak megfelelően rendszeresen belső auditokat kell lefolytatni a rendszer, az eljárások és módszerek értékelése céljából. Az eredményeket a laboratórium vezetőinek ki kell értékelniük, és szükség esetén helyesbítő intézkedéseket kell hozniuk.
- Az összes dokumentumot és feljegyzést meghatározott dokumentációs rend és eljárás szerint kell kezelni, amely rögzíti az iratok nyilvántartásának, jelölésének, verziószámozásának, engedélyezésének és átadásának módszereit. A hibás iratok megsemmisítésének rendjét is ki kell alakítani.
- Lehetővé kell tenni, hogy a vizsgálatok elvégzése előtt semmilyen (időbeli, tartalmi, pénzügyi) akadály ne legyen. Ezt lehetőleg írásbeli megbízások és szerződések támasszák alá.
- Az alvállalkozók bevonását szigorúan szabályozni kell. Csak olyan alvállalkozónak szabad tesztelési megbízásokat adni, amelyek bizonyítják, hogy maguk is megfelelnek e szabvány előírásainak.
- A vizsgálati eredményeknek érthetőeknek és reprodukálhatóknak kell lenniük. A dokumentumok minden ehhez szükséges információt tartalmazzanak.
- Gondoskodni kell arról, hogy a szoftvert vizsgáló laboratórium minden munkatársának meglegyen a munkája kifogástalan elvégzéséhez szükséges végzettsége.
- A helyiségeknek és a környezeti viszonyoknak nem szabad befolyásolniuk a mérési eredményeket.
- Bizonyítani kell, hogy a tesztelési eljárások és módszerek alkalmasak megbízható eredmények szolgáltatására. Ebből a célból minden vizsgálati eljárás és módszer alkalmasságát ellenőrizni kell.
- Az alkalmazott műszaki berendezések állapotát folyamatosan ellenőrizni és dokumentálni kell, és gondoskodni kell arról, hogy az eredményeket ne lehessen meghamisítani.
- A mérési jelentéseknek egyértelműeknek kell lenniük, és tartalmazniuk kell a mért eredményeket. A világos és egyértelmű megfogalmazásnak és információknak ki kell zárniuk a félreérthetőség lehetőségét.

A szabályok alkalmazhatóak a szoftvertesztelő részlegekre és laboratóriumokra is. A megvalósítás módja, a szabályok közötti prioritások meghatározása a tesztlabor létrehozóinak és vezetőinek feladata.

DIN EN ISO/IEC 12119:1995-08

A DIN EN ISO/IEC 12119:1995-08 szabvány címe „Információtechnika – szoftvertermékek – minőségi követelmények és tesztelési előírások”. A szabvány – amint azt a címe is sugallja – egyrészt rendszerezetten összefoglalja a szoftverek által teljesítendő követelményeket, másrészt szabályozza ezek ellenőrzését.

A szabvány rögzíti a dokumentációval (termékleírás és felhasználói dokumentáció) szemben támasztott tartalmi és formai követelményeket. A termékleírásnak ki kell terjednie a termék funkcióira, megbízhatóságára és használhatóságára. A formai előírások magukban foglalják az érthetőséget, teljeskörűséget, valódiságot, ellentmondásmentességet és az áttekinthetőséget.

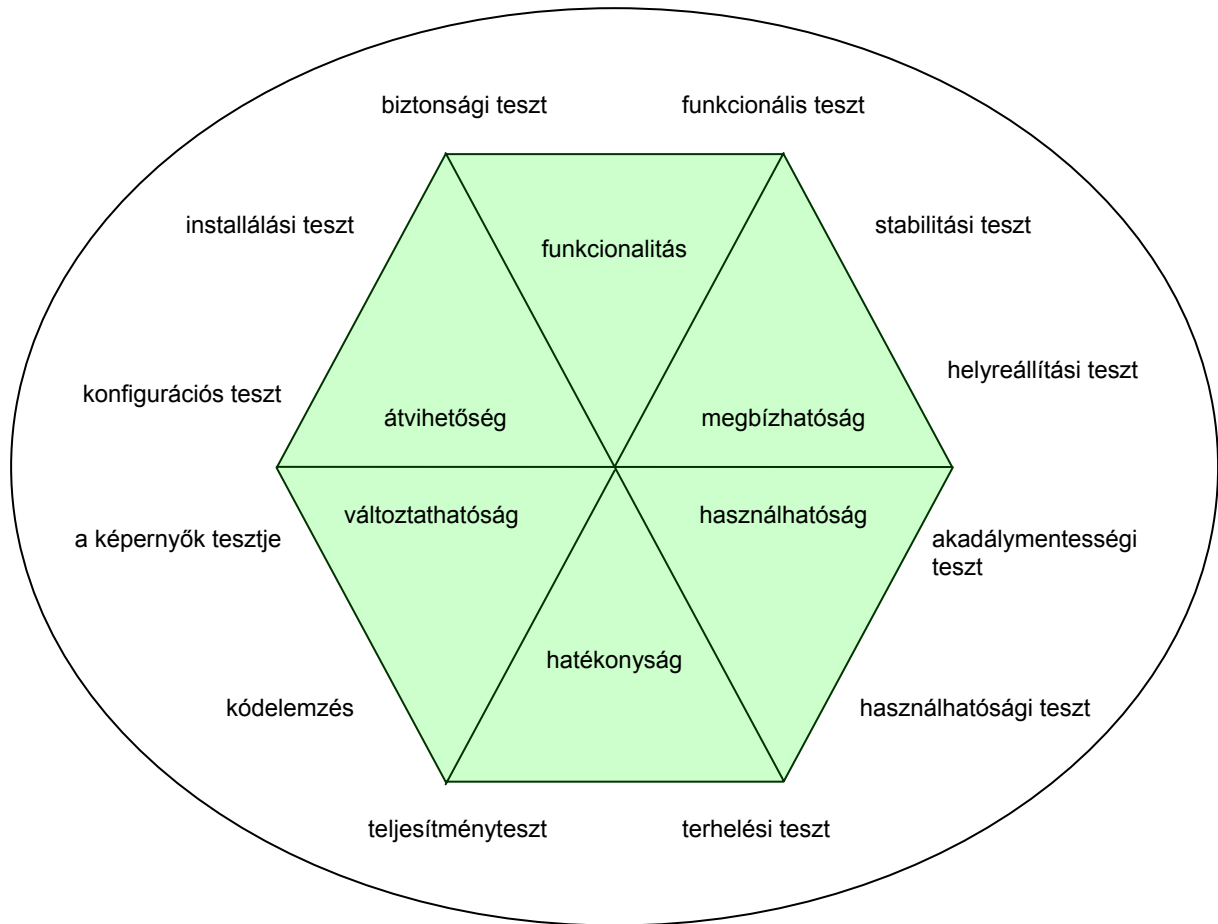
Mind a programoknak, mind az általuk kezelt adatoknak hibátlanul teljesíteniük kell a leírásokban megnevezett és leírt funkciókat. A megvalósított funkcióknak önmagukban és egymáshoz viszonyítva is ellentmondásmenteseknek kell lenniük. A szoftvereknek az előírtnál nagyobb terhelés és meg nem engedett kezelői beavatkozások esetén is meg kell őrizniük működőképességüket. A programoknak a felhasználó számára érthetőeknek, áttekinthetőeknek és kezelhetőeknek kell lenniük.

Amennyiben a termékleírásban a hatékonyságra, módosíthatóságra és átvihetőségre („portolhatóság”, például egyik operációs rendszerről a másikra) irányuló adat szerepel, a programoknak és adatoknak meg kell ezeknek az előírásoknak felelniük. A megfelelést ellenőrző teszteseteket szisztematikusan és meghatározott módszerek alapján kell kidolgozni, és ellenőrzési tervben kell azokat írásban rögzíteni. A szabvány azt is előírja, hogy mit kell a vizsgálati jegyzőkönyvnek tartalmaznia. Ezek között a teszteredményekén kívül szerepelnie kell a szabvány előírásaival vagy ajánlásaival ellentétes esetleges megoldásoknak, a meg nem felelést alátámasztó bizonyítékoknak és a tesztelés során alkalmazott eszközöknek is.

A szabvány által előírt szabályokat műveleti utasítások, előnyomtatott kérdőívek vagy más tesztelési segédanyagok formájában kell megvalósítani és írásban rögzíteni. Ezeknek minden munkatárs által hozzáférhetőnek kell lenniük, ugyanakkor rájuk nézve kötelezőnek is. A rendszer kidolgozása jelentős erőfeszítést és anyagi ráfordítást igényel, de már középtávon is kifizetődik, mivel a módszeres munka, a jól definiált viszonyok és felelősségek növelik a hatékonyságot.

Mit kell tesztelni?

A szabványok alapján meg lehet határozni a szoftverek minőségét meghatározó tulajdonságok főbb kategóriáit, vonatkozásait. Ezeket hat csoportba lehet rendezni (1. ábra):



1. ábra A szoftverek minőségét meghatározó tulajdonságok főbb kategóriái és az ezekkel kapcsolatos standard tesztek

- *Funkcionalitás*: képes-e a program az expliciten leírt és impliciten elvárt/elvárható feladatokat teljesíteni, illetve rendelkezik-e ilyen tulajdonságokkal? Ide tartoznak az interoperabilitás (képeség az együttműködésre más szoftverekkel és rendszerekkel), a biztonság (a feljogosítatlan hozzáférés kizárása mind a programhoz, mind az adatbázishoz), és a megfelelés az érvényes jogszabályoknak és szabványoknak.

- *Megbízhatóság*: hogyan reagál a rendszer a hibás kezelői beavatkozásokra, illetve milyen gyorsan és teljeskörűen képes működőképességét visszanyerni különböző kritikus helyzetek (hálózati zavarok, szerverek leállása, rendszerösszeomlás stb.) után?
- *Használhatóság*: könnyen érthető és könnyen kezelhető, „felhasználóbarát”-e a szoftver? Minden szóba jöhető felhasználói csoport számára egyformán fennállnak-e ezek a jellemzők?
- *Hatékony*ság: elfogadhatóak a válaszidők nagy adatbázisméretűk és sok felhasználó általi terhelés esetén is?
- *Változtathatóság*: lehet-e, és egyszerűen lehet-e a módosításokat és bővítéseket elvégezni, milyen széles a programozás nélkül, csak paraméterek átállításával elérhető módosítások köre? Az esetleges hibák megtalálása és kijavítása mennyire fáradságos, illetve mekkora ráfordítások szükségesek ehhez?
- *Átvihetőség (portolhatóság)*: működik-e a szoftver változtatás nélkül különböző környezetekben (más operációs rendszeren, internetes böngészővel vagy más adatbáziskezelővel)?

Természetesen a minőség vonatkozásait a szoftver kidolgozása során folyamatosan szem előtt kell tartani, és minden részletet ezek figyelembevételével kell kialakítani. Az 1. ábra az említett kategóriákba sorolható legfontosabb, már standard vizsgálatokként önállósult tesztek mutatja be. A tesztelés tervezése során (amelyet ideális esetben már a vizsgálandó szoftver tervezésével egy időben, vagy attól csak kissé lemaradva végeznek) meg kell határozni, hogy mely jellemzők a legfontosabbak az adott rendszernél, és az ellenőrzést ezekre intenzíven, a többi szempontra pedig csak kevésbé vagy éppen csak mintavétel jelleggel kell elvégezni. Mivel az internetes alkalmazások szerepe egyre nő, és ezeknél a sok felhasználó egyidejű bejelentkezése komoly problémákat okozhat, illusztrációként a következők a hatékonyság vonatkozásának két tesztelési módszerét, a teljesítménytesztet és a terhelési tesztet mutatják be.

Hatékonyság: a teljesítmény és a terhelhetőség tesztelése

Az internetes vállalati alkalmazások ma már jóval túlmennek az egyszerű információszerzésen, a vállalati értékteremtési folyamat integráns részeivé váltak. Mind az úton levő munkatársak, mind a vevők számára sokféle alkalmazás és a mögöttük levő sokféle adat, adatbázis érhető el az interneten keresztül. A sok felhasználó jelentős terhelést

okozhat, ami a válaszdíók elfogadhatatlan meghosszabbodását, rosszabb esetben a rendszer lebénulását, kiesését okozhatja. Ezek a jelenségek komoly anyagi veszteségekkel, a cég imidzsének romlásával járhatnak. Jelentős hiba lenne tehát a szoftverek tesztelésénél megelégedni a funkcionalitás vizsgálatával, meg kell ismerni a terhelhetőség határait, és azokat össze kell vetni a várható átlagos, illetve csúcsidejű igénybevéttel.

Adott hardverkonfiguráció esetén a felhasználók számának növekedésével a válaszdíók mérsékelt ütemben, lineáris módon nőnek. A tapasztalat azt mutatja, hogy a legtöbb rendszernél egy bizonyos ponton ez a viselkedés megváltozik, a válaszdíók hirtelen exponenciális növekedésbe kezdenek. A válaszdíók növekedése azt is jelzi, hogy a rendszer által időegység alatt elvégzett műveletek, tranzakciók száma stagnál, majd az „átbillenési pont” után csökken. A teljesítménytesztek a lineáris szakasz vizsgálatára szolgálnak, vagyis azt ellenőrzik, hogy a normálisan elvárható terhelések mellett stabil-e az alkalmazás, illetve a válaszdíók az elfogadható mértéken belül maradnak-e. A terhelési tesztek ezzel szemben az átbillenési pont meglétét, illetve helyét, a nagy terhelés mellett esetleg fellépő újabb hibákat vizsgálják.

A funkcionális tesztek során általában kiderül, hogy melyek azok a műveletek, amelyek a legjellemzőbbek az alkalmazás éles üzemelése során, illetve amelyek a leginkább igénybe veszik az erőforrásokat. A terhelési tesztek során ezekből a műveletekből kell a teszteseteket összeállítani, mert ezáltal lehet a tényleges terhelést modellezni, illetve a túlterhelést szimulálni. Gyakori az is, hogy bizonyos hibák egyáltalán nem lépnek fel a funkcionális tesztek során, mivel azokat a terhelés növekedése okozza, így csak a terhelési tesztek során derülnek ki. Az is tipikus, hogy a terhelés növelésével az ilyen hibák eleinte csak szórványosan jelentkeznek, erőteljesebb terhelés esetén állandósulnak. Az ilyen hibák nem jelentik automatikusan az egész rendszer instabilitását, mivel gyakran csak egy-egy jól körülhatárolható részegység okozza azokat. Az adott modul kijavítása után, a többi részegység változatlanul hagyásával a rendszer hibamentesen képes működni terhelés alatt is.

Külön figyelmet érdemel a tesztadatok témaköre is. Közismert jelenség főleg az ügyviteli, vállalatirányítási rendszerek esetén, hogy az adatbázisok növekedése a válaszdíók növekedésével a stabilitás romlásával jár. Ezért a tesztelést sem szerencsés „üres” adatbázissal végezni, hanem vagy az éles adatbázis adott, adatokkal jól feltöltött verzióját, vagy külön a teszt céljára létrehozott nagy, véletlenszerű adatsorozatot kell használni. Sok tesztnél az ismert kezdeti állapotból kiindulva adott műve-

letsorozatot elvégezve kell egy másik ismert állapotba eljutni, aminek meglétét egy-két képernyővel vagy kinyomtatott listával lehet ellenőrizni. Ilyen esetekben minden újabb teszt esetén vissza kell tölteni a kiinduló adatbázist.

A tesztprojektek tipikus menete

Ahány ház, annyi módszertan. Mind hasonló alapelveken nyugszik, az egyes lépések is hasonlóak, de a részletekben sok a különbség, ezen a téren a szabványosítás még várat magára. Minden önálló tesztlaborokra szabott rendszer nagy súlyt helyez a dokumentáltságra, az írásbeliségre. Ennek illusztrálásaként a következők a tesztprojektek lefolyását, egyes fázisait két módszertanon keresztül mutatják be.

A T-Systems Multimedia módszertana

Mint minden projektnél, itt is a *tervezéssel* kezdődik a projekt. Ennek során a koncepciót, a projekttervet és az erőforrástervet kell kidolgozni. A tervezés döntő fontosságú, sok hamvába holt projekt már a tervezés során (vagy annak hiánya miatt) halálra van ítélve, csak azt sajnos akkor még senki sem tudja. Adott teszt tervezése során többek között a következőket kell megtervezni, kitalálni:

- a teszt céljai és stratégiája,
- a környezet (hardver és szoftver, illesztések stb.),
- a tesztelés terjedelme, lépcsőfokai és módjai,
- az automatizálhatóság, és annak eszközei,
- a tesztadatok kialakításának alapelvei,
- a szempontok és tesztelendő részek prioritásai,
- a tesztek után elvárt állapot, adathelyzet és azok megjelenítése,
- a tesztek adminisztrálásával és dokumentálásával szembeni igények,
- feladatok, felelősségek és szerepek,
- a hibaosztályok és a hibakezelés módja,
- a hibákat elemző csoportok összeállítása, eskaláció (kiket kell értesíteni a különböző súlyosságú hibák esetén?),
- a jelentési rendszer (tartalom, címzettek, gyakoriság),
- teamek szervezése, a menedzsment kialakítása,
- minőségbiztosítás,
- kockázatkezelés.

A terveket a lehető legtöbb érintettel konzultálva kell kidolgozni. Mivel a tesztelés az idő–költség–minőség hármasságának nyomása alatt zajlik, a kockázatkezelésnek kiemelt jelentőséget kell tulajdonítani. A gyakorlat azt mutatja, hogy a tesztelendő programváltozat tesztelésre leadásának késése, vagy annak tesztelés közbeni megváltoztatása mindennapos jelenségek, így azok hatását megfelelő ellenintézkedésekkel csökkenteni lehet, vagy akár teljesen ki is lehet küszöbölni.

Az *előkészítés* fázisában össze kell állítani a tesztkörnyezetet, meg kell szerezni a tesztelés lefolytatásához és menedzseléséhez szükséges és betervezett szoftver-segédesszközöket (tool), azokat megfelelően konfigurálni kell, ki kell alakítani a tesztadatokat, ki kell dolgozni a manuális és automatizált teszteseteket, meg kell írni az automatizáláshoz szükséges programokat, ún. szkripteket, és mindezt a projektvezetésnek folyamatosan menedzselnie kell.

A *kivitelezés* fázisában történik meg a tényleges tesztelés. Ez magában foglalja a megtervezett tesztek elvégzését, a hibaüzenetek feldolgozását és elemzését, statisztikák és különféle mérőszámok előállítását, a dokumentálást és a részjegyzőkönyv elkészítését minden egyes teszt esetén.

A *dokumentálás és kiértékelés* fázisában össze kell foglalni és dokumentálni kell az egész folyamat során előállított eredményeket és dokumentumokat. Ennek során a következő részfeladatokat kell elvégezni:

- a hibák mennyiségének, eloszlásának és trendjeinek kiértékelése,
- grafikus ábrázolás,
- a tesztet lezáró jegyzőkönyv elkészítése,
- a megfelelés eldöntése: a szoftver jónak minősítése, vagy visszaküldése a fejlesztőkhöz,
- a projekt lezárása (utókalkuláció, átadás-átvételi jegyzőkönyv stb.)

A LogicaCMG „TestFrame” módszertana

A módszertan négy fő fázist alkalmaz, ezek sokban hasonlítanak az imént ismertetett rendszer megközelítésmódjához.

Az *előkészítés* fázisában végzik el a megközelítésmód megtervezését, a kockázatok elemzését, a teszt stratégiájának megalkotását és a konkrét tervezést. A módszertan kidolgozói fontosnak tartják a tervezés minél előbbi megkezdését, lehetőleg a vizsgálandó szoftver tervezésével egy időben. A specifikáció ismeretében és sok tapasztalat birtokában már meg lehet határozni a várható problematikus területeket, és elemezni lehet a kockázatokat. Ennek alapján meg lehet határozni az egyes

tesztelendő részprogramok és tesztek prioritásait. Minden programrészt minden lehetséges adatkombinációval megtesztelni lehetetlen, ezért létfontosságú azon részek körülhatárolása, amelyeket a nagy hibakockázat miatt alapos „nyaggatásnak” kell alávetni, míg a többit takarékosági okokból kevés adattal, vagy mintavételes jelleggel elegendő vizsgálni.

Az *elemzés* fázisában kell meghatározni a tesztelés terjedelmét, módjait (cluster), a tesztfeltételeket és a teszteseteket. Az elemzés céljára a tesztmódokat (modul, integrációs, rendszer, felhasználói elfogadhatósági, gyártási elfogadhatósági) és a belső érintetteket (végfelhasználók, marketing, támogató részleg, fejlesztők, üzemeltetők és belső biztonságiak) mátrixba lehet foglalni. A mátrix kitöltése után ebből lehet levezetni az egyes résztesztek feltételeit és eseteit.

A *navigációs* fázisban a műszaki teszt, a navigációs struktúra, a tesztmotor és a navigációs szkriptek részfeladatait kell megoldani. A teszt maga sokszor ismételt, az ember számára unalmas feladatok végrehajtását jelenti. Speciális szoftvereszközökkel ezek nagy része automatizálható: az eszköz betanítása után képes a kezelő minden akcióját (egérmozgatás és -kattintás, betűk és számok bebillentyűzése) szimulálni. A betanítás eredményei a navigációs szkriptek, amelyek egyébként egyfajta programnak is tekinthetők. A tesztelendő programot és a navigációs szkriptet együtt elindítva a program úgy működik, mintha ember kezelné, pedig emberi beavatkozás nem történik. Az automatikus végrehajtás azonban nem minden esetben alkalmazható, ennek eldöntése az előkészítés és elemzés fázisának egyik feladata. Az ember nem küszöbölhető ki teljesen, például a képernyők rossz elrendezése vagy hibás szövegei, a kezelhetőség és „felhasználóbarátság” megítélése továbbra is a tesztet végző szakember feladata.

A *végrehajtás* fázisa a hibák megtalálását, a tesztjegyzőkönyv és a kiértékelő összefoglaló elkészítését, valamint az eredmények és értékelő dokumentumok illetékesekhez való eljuttatását foglalja magában. A teszteseteket többször egymás után kell elvégezni, mert az újrafutások egyes esetekben az elsőtől eltérő eredményekhez vezethetnek.

A tesztprojekt menedzselése végighúzódik az összes fázison. Kiemelkedő szerepet játszik ebben a megtalált hibák menedzselése és a lezáró jegyzőkönyv elkészítése. A statisztikákban fontos jellemző a hibamentes és hibás futások száma és aránya, és jelentőséget kell tulajdonítani a szépséghibának minősíthető kisebb, illetve formai eltéréseknek is.

Összefoglalás

A szisztematikus tesztelés a minőségbiztosítás fontos eszköze a szoftverek előállítása terén. A tesztelés folyamatát a vizsgálandó program fejlesztésével egy időben indítva, strukturált és dokumentált módon kell tervezni és végrehajtani. A különféle segédeszközök lehetővé teszik a programtesztelés nagymértékű automatizálását, ami a költségek és az emberi hibák lehetőségének csökkentését teszi lehetővé. Az automatizálás eszközei, a szkriptek az ismételhetőséget és az újrafelhasználhatóságot is eredményezik.

Az önálló tesztrészlegek kialakítása csak közepes és nagy cégeknél gazdaságos. A kisebbek programtesztelési feladataikat erre szakosodott szakszervezetekhez, tesztlaboratóriumokhoz kiszervezve oldhatják meg. A vonatkozó szabványok és elvek betartása, a strukturált, tervszerű és jól dokumentált megközelítésmód, illetve az ingyen hozzáférhető segédeszközök a legkisebb szoftverfejlesztő vállalkozások saját belső tesztjeit tekintve is a piacra kerülő szoftverek hibáinak jelentős csökkentéséhez vezethetnek.

Összeállította: Kis Miklós

Schneider, M.; Bienstock, S.: Software auf Qualität abklopfen. = Information Management Consulting, 21. k. 1. sz. 2006. febr. p. 57–64.

Schlatter, A.: Methodisches Testen als Grundlage für Qualitätssicherung. = Information Management Consulting, 21. k. 1. sz. 2006. febr. p. 53–56.