Szoftverek életciklusa

1. A felhasználókban új igény merül fel.
2. Az igények, követelmények elemzése, meghatározása:
   követelmény specifikáció; funkcionális specifikáció, rendszerspecifikáció.
3. Logikai és fizikai tervezés.
4. Implementáció
5. Tesztelés
6. Átadás
7. Üzemeltetés, karbantartás.

Szoftverfejlesztési modellek

A különböző szoftverfejlesztési módszertanokban a fenti lépések különböző sorrendben követhetik egymást:

- **lineáris**: az ún. **vízesés modell** az első, széles körben elterjedt szoftverfejlesztési módszertan, nagy megrendelők nagy projektjeihez alakították ki. Az életciklus lépései egymás után, átfedés nélkül következnek, nem engedi, hogy visszatérjünk egy korábbi fázisba. Ebben a tiszta formában ritkán alkalmazzák. Egyik módosítása a **V-modell**, melyben a V egyik szára megegyezik a vízesés modellel, másik szára az egyes szintekhez tartozó teszteléseket tartalmazza.

Prototípus modellek: a végső átadás előtt több prototípus is elkészül, hogy a megrendelő láthassa, mit várhat a rendszertől és mielőbb kiderüljenek a félreértések. Az életciklus 2-5. lépései többször ismétlődnek a fejlesztés folyamán.

- **spirális**
- **iteratív (inkrementális)**

A tesztelési folyamat legutolsó fázisa a rendszer használata előtt az ún. átvételi tesztelés. A rendszert ilyenkor a megrendelő adataival kell tesztelni, amely olyan hiányosságokat vethet fel, ami más esetben nem derül ki.

Egy verzión belüli életciklus

Pre-Alfa, Alfa

Az átvételi tesztelést alfa-tesztelésnek is szokták nevezni. Az alfa-tesztelési folyamatot addig kell folytatni, amíg a rendszerfejlesztő és a megrendelő egyet nem ért abban, hogy a leszállított rendszer a rendszerkövetelményeknek megfelelő.

Valamely termék alfa változata még teljes körű hibakeresésre vagy a szolgáltatások teljes megvalósítására vár, azonban eleget tesz a legfontosabb követelményeknek. Gyakran hiányoznak belőle a végleges változatba ígért funkciók, azonban jól demonstrálja a szoftver megvalósíthatóságát és alapszerkezetét. Mivel ez a kiadási életciklus első legfontosabb szakasza, ezért nevezték el a görög ábécé első, alfa betűjéről.

A szoftver alfa kiadása általában a szoftvertesztelők számára átadott első build.

Béta

Amikor egy rendszer, mint szoftvertermék piacra kerül, gyakran egy másik tesztelés is végbemegy, amelyet **béta-tesztelés**nek nevezünk. A béta-tesztelés magában foglalja a rendszer számos potenciális felhasználójához történő leszállítását, akikkel megegyezés történt a rendszer használatára, és ők jelentik a rendszerrel kapcsolatos problémáikat a rendszerfejlesztőknek. Ezáltal **a rendszer valódi használatba kerül**, és számos olyan hiba válik felfedezhetővé, amelyeket a rendszer építői esetleg nem láthattak előre. Ezután a visszacsatolás után a rendszer módosítható és kiadható további béta-tesztelőknek, vagy akár általános használatra is.

A **béta verzió vagy a béta kiadás** egy számítógépprogramnak általában azt az első kiadását jelenti, labilis, és még nem érett meg a kiadásra. Némely fejlesztő erre a szakaszra előnézetként, technikai előnézetként, vagy korai kiadásként hivatkozik. Mivel a kiadási életciklusban az alfa szakaszt követően a második fontos szakasz, ezért a görög ábécé második betűjéről, a bétáról nevezték el. A fejlesztők vagy zárt béta, vagy nyitott béta változatban adják ki; a zárt béta verziókat egy kiválasztott csoport számára adják ki tesztelési célból, míg a nyitott bétákat nagyobb közösségnek szánják, többnyire a nagyközönségnek. A tesztelők bejelentik az általuk talált hibákat, és néha kisebb funkciókat is javasolnak, amiket szeretnének látni a végleges változatban.

Kiadásra jelölt

A szoftvert akkor hívják **kiadásra késznek**, amikor az alkotócsoport egyetért abban, hogy **a rendszer eleget tesz a funkcionális követelményeknek, és nem tesznek már új funkciót a kiadásba**, ugyanakkor **felmerülhetnek még fontos hibák** a szoftverben.

A kiadásra jelölt (angolul: release candidate, rövidítve: RC) kifejezés olyan verzióra utal, aminek nagy esélye van arra, hogy az a végleges termék legyen, amelyik kiadásra kész, hacsak nem jelentkeznek súlyos hibák. Ebben a szakaszban a termék valamennyi megtervezett szolgáltatást tartalmazza, és nincsenek benne különösebb hibák. A termék ebben a fázisban általában kódkész.

Átadás, üzemeltetés, karbantartás

A szoftver termék élete a telepítéssel csak a legritkább esetben ér véget. A karbantartás során derül ki mennyire sikerült jó, a megrendelőt igényeit kielégítő, illetve a későbbi változtatási és továbbfejlesztési igényeket is jól tűrő rendszert elkészíteni. Az általános célú, nagypéldányszámú szoftverek esetén a módosítási igények folyamatosan jelentkeznek. A gyakorlatban a legtöbb szoftver életének nagyobbik részét teszi ki – és a költségek nagyobb részét is igényli – az egyre újabb változatok, verziók előállítása, a módosítások végrehajtása. Ezt a tevékenységet szokás karbantartás (maintenance) néven említeni.

Az esetek többségében, általában csak kisebb változás történik, ami nem érinti az adat szerkezetet. Ekkor viszonylag egyszerű a dolog: feltelepítjük az új programot, ami ugyanúgy csatlakozik az adatbázishoz, és a frissítés után minden működik tovább. Ilyenkor akár az is megengedhető, hogy frissített és nem frissített kliensek egyszerre használják a rendszert. Amikor azonban a rendszer adatszerkezetében jelentős változás van, vagy jelentős működésbeli eltérés van, akkor közel sem ilyen egyszerű a frissítés. Ilyenkor adatkonverzióra is szükség van vagy valamilyen módon meg kell oldani, hogy a régi adatok használhatóak legyenek. Ha például az adatbázisba új mezőt kell felvennünk, akkor meg kell akadályozni, hogy valaki a szoftver régi verziójával csatlakozzon az adatbázishoz, mert ez beláthatatlan következményekkel járhat. A fejlesztés folyamán kell megteremteni annak feltételét, hogy a program meg tudja kérdezni mi a legújabb verzió, és ez alapján dönteni tudjon, hogy elindulhat-e vagy sem.

Minden változtatás után tesztelni kell a teljes rendszert. Ez egyrészt az új funkciók tesztelését, másrészt regressziós tesztek elvégzését jelenti. Az új verzió tesztelése után kerülhet csak sor a frissítésre, vagyis az éles rendszerben való bevezetésre.
Ha az új verzió működőképes, akkor kell meghatározni az élő rendszer frissítésének idejét. Ehhez esetenként az ügyféllel is kell egyeztetni, hogy pontosan hány gépen kell elvégezni a frissítést. Meg kell becsülni, hogy mennyi ideig nem lesz használható a szoftver és ezt az ügyfél igényeinek megfelelően időzíteni.
Mint minden esetben itt is fontos, hogy készítsünk biztonsági mentést, hogy probléma esetén vissza tudjuk állítani az eredeti állapotot.

Ha az új verzió új funkciókat tartalmaz, akkor azokat általában valamilyen módon le kell oktatni a felhasználóknak. A frissítéssel egyidőben a felhasználói kézikönyv új verzióját is el kell készítenünk és eljuttatni a felhasználóhoz.