Rendszerfejlesztés

1.

Rendszerfejlesztés

Heti 3 óra, 3 kredit, gyakorlati jegy

Teljesítés feltétele:

- beadandó feladat
- csoportmunka
- (2)-3 fős csoportok

Beadandó

- Feladatmegosztás
- Rendszerterv
- Adatbázisterv (ha releváns)
- Fejlesztési dokumentáció
- Felhasználói dokumentáció
- Prezentáció

Rendszertervezés

Komplex folyamat, célja egy adott cél elérése érdekében működő **információs rendszer:**

- Tervezése
- Létrehozása
- Telepítése
- Működési feltételeinek biztosítása

Információs rendszer

Szervezetek, cégek működését segíti elő.

Tervezése jellemzően szervezeti aspektusból történik.

Alkalmazkodva a szervezet méretéhez és információs igényeihez, amihez meg kell értenünk:

- a szervezet felépítését és működését
- a szervezet információs folyamatait

Rendszerelemzési, -tervezési folyamat

Eredménye: átfogó számítógépes rendszer megszületése, amely hatékonyan támogatja a szervezet feladatainak megvalósítását:

- Szoftver
- Hardver
- Rendszerszoftver (futtatási környezet)
- Dokumentáció
- Felhasználók oktatása

A rendszertervezés története

- 1950-es évek: szobányi gépek, egy részleg, egy feladata (pl. beszerzés, raktározás), gépi kód, assembly, saját fejlesztők
- 1960-as évek: miniszámítógépek (szekrényméretű) 3G (procedurális, strukturált) nyelvek, szoftveripar kezdetei, rendszerfejlesztés mint tudományág
- 1980-as évek: mikroszámítógépek, 4G (specializált, OOP) nyelvek (pl. SQL), szoftverfejlesztő cégek

A rendszertervezés története (II)

- 1990-es évek: Rendszerintegráció, Grafikus fejlesztőkörnyezet (pl. Visual Basic), Kliens-szerver architektúra (Oracle, Microsoft, Ingres stb.), Vállalati (ERP) szoftverrendszerek (pl. SAP)
- 2000-es évektől: internet, www, felhő alapú rendszerek, SaaS, IaaS

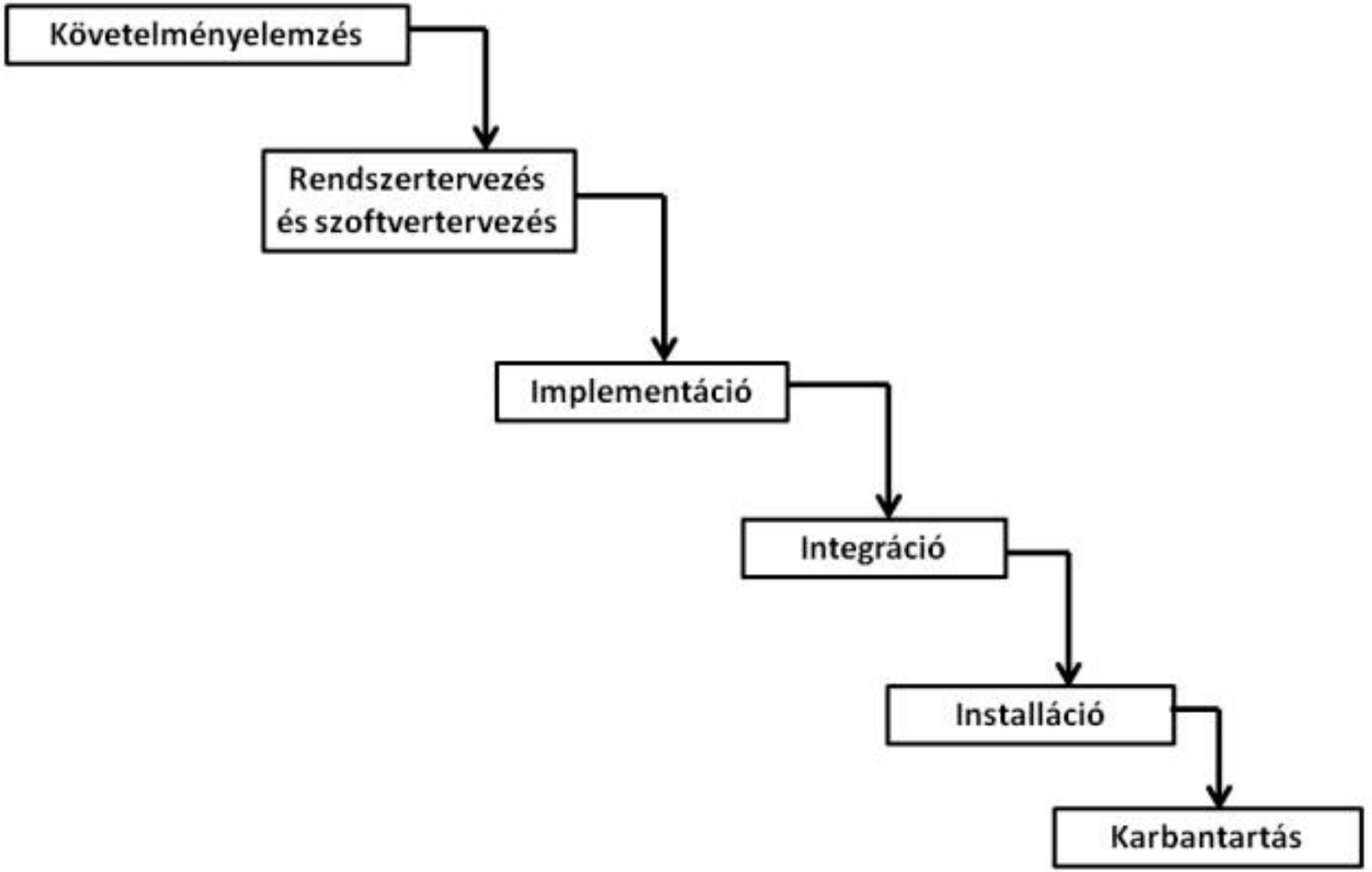
Információs rendszerek életciklusa

- Rendszertervezés
- Implementálás
- Bevezetés
- Üzemeltetés, karbantartás
- Új rendszer bevezetésének előkészítése

Információs rendszerek tervezése

- A rendszer egymáshoz kapcsolódó elemek halmaza, amelyek együttműködve egy vagy több célt valósítanak meg. Működésére jellemző, hogy valamilyen bemenetből (input) egy folyamat végén kimenet (output) jön létre.
- Az informatikai rendszer adatokat és információkat tárol, azokat különböző rendszerspecifikus módon feldolgozza, és elérhetővé teszi.

Vízesés Modell



Evolúciós modell

A fejlesztők létrehoznak egy kezdeti információs rendszert, azt a felhasználókkal véleményeztetik, majd sok-sok verzión keresztül addig finomítják, amíg a minden igényt kielégítő rendszert el nem érik.

- Feltáró fejlesztés
- Eldobható prototípus készítés

<u>Hátrányai:</u>

- nehezen átlátható a fejlődési folyamat
- A sok változat miatt naprakész dokumentáció nincs
- Ezért az üzemeltetés és a karbantartás problémás

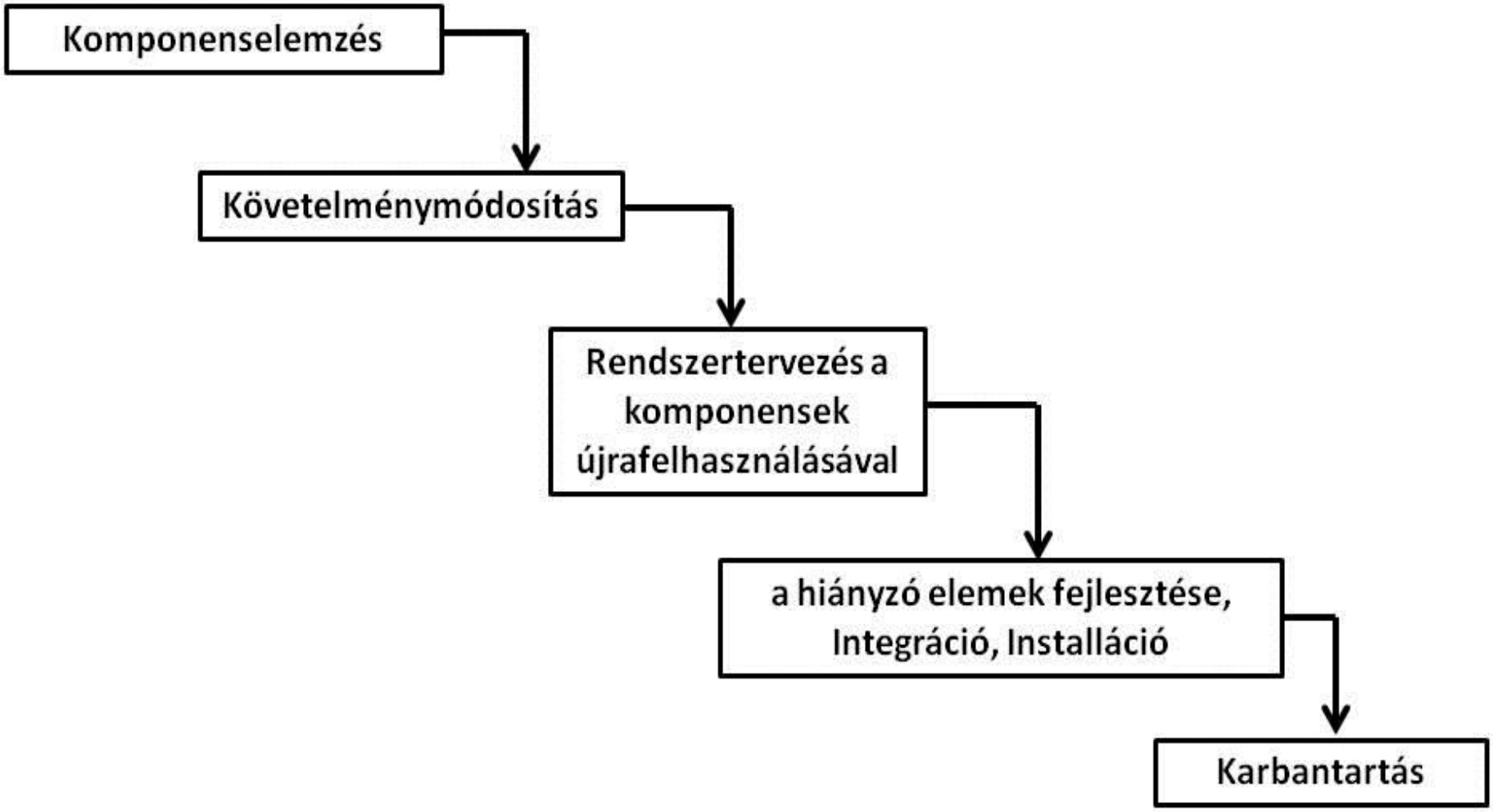
Evolúciós modell - Feltáró fejlesztés

A felhasználóval történő egyeztetés után az adott alapkövetelményeknek megfelelő működőképes rendszert adnak át. A rendszer továbbfejlesztése során az alaprendszer működését látva a felhasználók újabb és újabb követelményeket határoznak meg a rendszer fejlesztői számára, közösen alakítva ki a rendszer végleges felépítését.

Evolúciós modell - Eldobható prototípus készítése

- A fejlesztés alapja a felhasználó elvárásainak minél pontosabb megismerése, amelyekre alapozva pontosan definiálhatók azok a tulajdonságok, amelyeket a rendszernek tudnia kell.
- Azokat a követelményeket, amelyek nem érthetők teljesen pontosan, egy-egy prototípusban valósítják meg, és a felhasználó véleménye alapján finomodik a követelmények meghatározása és a rendszer funkcionalitása.

Komponens alapú és újrafelhasználható rendszerfejlesztés



Spirális fejlesztés

1. Az adott ciklus céljainak meghatározása.
2. Kockázati tényezők figyelembe vétele.
3. Fejlesztés és tesztelés: a kockázatanalízis alapján megfelelő fejlesztési modellt kell választani.
4. El kell dönteni, hogy folytatódjon-e a fejlesztési folyamat egy következő ciklussal. Ha a folytatás mellett döntünk, akkor kezdődik a következő kör a célok meghatározásával.

Gyors Alkalmazásfejlesztés (RAD)

- Ciklikus fejlesztés
- Működő prototípusok létrehozása
- Szoftverfejlesztést támogató számítógépes programok, például integrált fejlesztői környezet használata
- Kompromisszumokkal járhat a használhatóság, tulajdonságok és a futási sebesség terén
- Ezt ellensúlyozza a jelentősen lecsökkent fejlesztési idő, a felhasználói igények magas fokú kielégítése, és a grafikus felhasználói felület.

Rational Unified Process (RUP)

Folyamatközpontú, nem szigorúan végrehajtandó, egymás utáni eljárások sorozata, sokkal inkább egy flexibilis keretet a fejlesztési folyamat irányításához.

A rendszerfejlesztés folyamatát alapvetően három dimenzióval írja le:

1. dinamikus perspektíva - a modell fázisai

2. statikus perspektíva -a végrehajtandó folyamattevékenységek

3. gyakorlati perspektíva - jól hasznosítható gyakorlatok a folyamat alatt

RUP rendszerfejlesztési fázisok

- Kezdeti fázis (inception): a rendszerrel szembeni elvárásokat és a rendszer funkcionalitását olyan mélységig kidolgozzák, hogy az alapján a fejlesztési folyamat részletesen tervezhető lesz.
- Kidolgozási fázis (elaboration): részletekbe menően meghatározzák, hogy a felhasználók hogyan fogják használni a rendszert, az hogyan fog felépülni, és elkészül az stabil alaparchitektúra terve (architecture baseline), melynek segítségével a teljes fejlesztés folyamata ütemezhető és a költségei is tisztázhatók.

RUP rendszerfejlesztési fázisok (II)

- Megvalósítási fázis (construction): Kifejlesztik a teljes rendszert. A folyamat a rendszertervre, a programozásra és a tesztelésre fókuszál. A részek párhuzamosan fejleszthetők, majd integrálhatók. A fázis végére elkészül a működőképes rendszer, és a hozzá kapcsolódó dokumentáció.
- Átalakítási fázis (transition): a rendszer működésének tesztelése, részletesen dokumentálva. A tesztelés végén dönteni kell a rendszer átalakításáról vagy késszé nyilvánításáról.

eXtrém Programozás (XP)

- a megrendelő részt vesz a fejlesztési folyamatban
- szinte valamennyi fázis egy folyamatba olvad
- a fejlesztés és tesztelés időben közel kerül
- a fejlesztők párban dolgoznak
- a fejlesztők saját programkódjaikat ellenőrzik

eXtrém Programozás (II)

Előnyei:

- magasabb produktivitás
- jobb kommunikáció (f-f, f-m)
- magasabb színvonalú végeredmény

Hátrányai:

- nem alkalmazható minden esetben
- kiválóan képzett programozók kellenek
- az intenzív kommunikáció sok idő és energia
- a dokumentáció nem kielégítő

Agilis szoftverfejlesztés

- Egy sikertelen tanácskozás eredménye
- 2001 február, Utah
- Új módszertant nem tudtak kidolgozni
- De létrehozták az Agile Manifesto kiáltványt

Az Agilis Kiáltvány

A szoftverfejlesztés jobb módjait fedezzük fel azáltal, hogy csináljuk, és segítünk másoknak is csinálni. Ennek során az alábbi hangsúly-eltolódásokat találtuk:

- Egyének és interakcióik ←→ eljárások és eszközök
- Működő szoftver ←→ teljes körű dokumentáció
- Együttműködés ←→ szerződésről való alkudozás
- Változásokra való reagálás ←→ terv követése

A jobb oldalon szereplő értékek is fontosak, de a bal oldalon lévőket fontosabbnak tartjuk.

Az Agilis Kiáltvány

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas