Szoftverfejlesztés

2. Előadás

Dr. Mileff Péter

Miskolci Egyetem
Általános Informatikai Tanszék

Folyamattevékenységek

- Alapvetően négy különböző folyamattevékenység:
  - **Specifikáció (követelménytervezés)**
  - **Tervezés és implementáció**
  - **Validáció**
  - **Evolúció**
- Ezeket a különféle fejlesztési folyamatmodellek különféleképpen szervezik.
  - Vízesésmodell – szekvencia
  - Evolúciós fejlesztés – összemosás

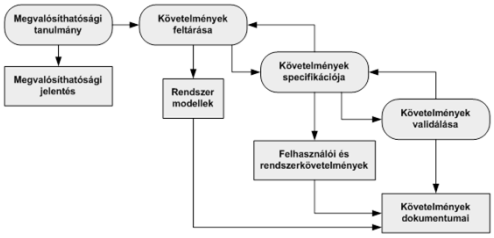
SZOFTVERSPECIFIKÁCIÓ…

Szoftverspecifikáció (Követelménytervezés)

- Az a folyamat, ahol megértjük és definiáljuk, hogy a rendszernek milyen szolgáltatásokat kell biztosítani.
  - Azonosítjuk a rendszer üzemeltetésének és fejlesztésének megszorításait.
- Kritikus szakasz:
  - Az itt vétett hibák nagy problémákhoz vezetnek majd a rendszertervezés későbbi szakaszában és az implementációban.

Szoftverspecifikáció

- A folyamat eredménye a követelmény dokumentum előállítása. Leírása két szinten:
  - Végfelhasználók ügyfelek számára leírt követelmények
  - fejlesztők sokkal részletesebb rendszer-specifikációja



5

1. fázis: Megvalósíthatósági tanulmány

- A felhasználók, megrendelő kívánságainak ellenőrzése:
  - kielégíthetők-e az adott szoftver- és hardvertechnológia mellett.
- A vizsgálatoknak el kell döntenie:
  - a rendszer költséghatékony-e,
  - kivitelezhető-e.
    - A megvalósíthatóság elemzésének relatíve olcsónak és gyorsnak kell lennie.
- Eredménye: **megvalósítási jelentés**.

6

2. fázis: *Követelmények feltárása és elemzése*

- A folyamat a következőkön alapszik:
  - rendszerkövetelmények meglévő rendszereken történő megfigyelése,
  - a potenciális felhasználókkal és beszerzőkkel folytatott megbeszélések.
- Akár egy vagy több különböző rendszermodell, illetve prototípus elkészítését is magában foglalhatja.

7

3. fázis: *Követelmény specifikáció*

- Az elemzési tevékenységek során összegyűjtött információk egységes dokumentummá alakítása:
- **A felhasználói követelmények:**
  - a rendszerkövetelmények absztrakt leírása a végfelhasználóknak, megrendelőnek.
- **A konkrét rendszerkövetelmények:**
  - részletezik az elkészítendő rendszer által nyújtandó funkciókat.

8

4. fázis: *Követelmény-validáció*

- **Ellenőrzés:** hogy mennyire valószerűek, konzisztensek és teljesek a követelmények.
- A folyamat során:
  - fel kell tárni a követelmények dokumentumában található hibákat, hiányosságokat
  - és ki kell javítani.
- A követelménytervezés tevékenységeit nem kötelező szigorú sorrendben végrehajtani.

9

SZOFTVERTERVEZÉS ÉS IMPLEMENTÁCIÓ…

10

Szoftvertervezés és implementáció

- Nem más, mint a rendszer-specifikáció futtatható rendszerré történő alakítása.
- Magában foglalja:
  - a szoftver tervezését,
  - a programozását,
  - esetleg a specifikáció finomítását.

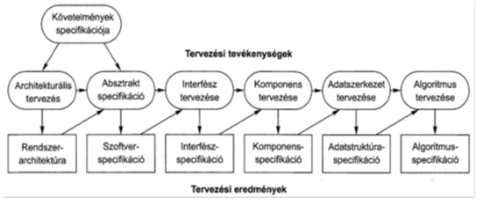
11

A tervezés

- A szoftver **tervezése** leírása a
  - szoftver struktúrájának és az adatoknak,
  - a rendszerkomponensek közötti interfészeknek,
  - és néha a használt algoritmusoknak.
- Iteratív folyamat:
  - A terv folyamatosan fejlődik, finomodik.
- A tervezés számos különféle absztrakciós szintű rendszermodell kifejlesztését is tartalmazhatja.
- A tervet részekre osztásával napvilágra kerülnek a korábbi hibák
  - Visszacsatolások biztosítják a rendszer fejlesztését.

12

A tervezési folyamat



1. ***Architekturális tervezés***
2. ***Absztrakt specifikáció***
3. ***Interfész tervezése***
4. ***Komponens tervezése***
5. ***Adatszerkezet tervezése***
6. ***Algoritmus tervezése***

13

A tervezés lépései

- ***1. Architekturális tervezés:*** A rendszert felépítő alrendszerek és a köztük található kapcsolatok azonosítása és dokumentálása.
- ***2. Absztrakt specifikáció:*** Minden alrendszer esetében meg kell adni szolgáltatásaik absztrakt specifikációját és azokat a megszorításokat, amelyek mellett azok működnek.

14

A tervezés lépései

- 3. ***Interfész tervezése:*** interfész tervezése az alrendszerek felé. Az interfész specifikációnak egyértelműnek kell lennie.
- ***4. Komponens tervezése:*** *A* szolgáltatások elhelyezése a különböző komponensekben, és meg kell tervezni a komponensek interfészeit.

15

A tervezés lépései

- ***5. Adatszerkezet tervezése:*** Meg kell határozni és részletesen meg kell tervezni a rendszer implementációjában használt adatszerkezeteket. (pl.: adatbázis, xml)
- ***6. Algoritmus tervezése:*** Meg kell tervezni és pontosan meg kell határozni a szolgáltatások biztosításához szükséges algoritmusokat.

16

Az **implementáció**

- A programozás személyekre szabott tevékenység, nincs általános szabály, amit követni lehet.
  - Különböző cégeknél lehetnek megkötések.
- A fejlesztés gyakori menete:
  - azon komponensekkel kezdik, melyeket a legjobban megértettek,
  - csak azután fejlesztik a kevésbé érthető komponenseket.
- A fejlesztés során a programozók tesztelnek is, amely hibákat tár fel.
  - Ezeket javítani kell.

17

SZOFTVERVALIDÁCIÓ…

18

Szoftvervalidáció

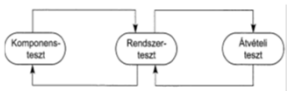
- Általánosan: verifikáció és a validáció (V & V).
- **Cél:**
  - megmutassa, a rendszer konform a saját specifikációjával,
  - a rendszer megfelel a rendszert megvásárló ügyfél elvárásainak.
- Egy ellenőrzési folyamat a szoftverfolyamat minden egyes szakaszában
  - A követelmények meghatározásától kezdve egészen a program kifejlesztéséig.
  - Pl.: szemlék, felülvizsgálatok.

19

Szoftvervalidáció

- A rendszerek többsége nem tesztelhetők magukban mint monolitikus egységek.
  - Ezért többlépéses tesztelés.
- A programban felderített hibákat ki kell javítani. Következménye:
  - a tesztelési folyamat egyéb szakaszait is meg kell ismételni.

Egy háromlépéses tesztelési folyamat:



20

A tesztelési folyamat szakaszai:

- ***1. Komponens (vagy egység) tesztelése:***
  - Az egyedi komponenseket tesztelni kell, és biztosítani a tökéletes működésüket.
  - Minden egyes komponens tesztelése az egyéb rendszerkomponensektől függetlenül történik.
- ***2. Rendszer tesztelése.***
  - Az alrendszerek és interfészeik közötti előre nem várt kölcsönhatásokból adódó hibák megtalálásával foglalkozik.
  - A rendszer eleget tesz-e a rendszer funkcionális és nemfunkcionális követelményeinek és az eredendő rendszertulajdonságoknak.

21

A tesztelési folyamat szakaszai:

- **3.** ***Átvételi tesztelés:***
  - A tesztelési folyamat legutolsó fázisa a rendszer használata előtt.
  - A rendszert ilyenkor a megrendelő adataival kell tesztelni
    - Olyan hiányosságokat vethet fel ami, más esetben nem derül ki.
- **Alfa-tesztelésnek** is szokták nevezni.
  - Ezt addig kell folytatni, amíg a rendszerfejlesztő és a kliens egyet nem ért abban, hogy a leszállított rendszer a rendszerkövetelményeknek megfelelő.

22

SZOFTVEREVOLÚCIÓ…

23

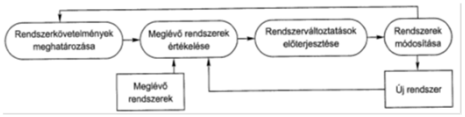
Szoftverevolúció

- A szoftver karbantartása a rendszeren történő változtatások folyamata, a rendszer működésbe állítása után.
- Korábbi felfogás szerint éles a határ a fejlesztés és a karbantartás között.
- A karbantartás költségei a fejlesztés költségeinek többszörösét is elérhetik.
  - Ez a folyamat sokkal kisebb kihívás, mint a kifejlesztés.
- Napjainkban ez az elkülönítés egyre lényegtelenebb.

24

Szoftverevolúció

- Kevés szoftverrendszer tekinthető teljesen újnak
  - így a fejlesztés és karbantartás egy egységnek tekinthető.
- Valószerűbb a szoftvertervezést evolúciós folyamatként kezelni, ahol
  - a szoftver élettartama alatt a követelményekkel és az ügyfél elvárásaival együtt folyamatosan változik.



25

RUP, MINT FOLYAMAT MODELL...

26

Rational Unified Process

- Rational Software Corporation fejlesztette ki
  - Az IBM felvásárolta 2003-ban.
- Jó példája a modern folyamatmodelleknek.
  - UML-ből és a hozzá kapcsolódó Unified Software Development Process-ből származik.
- Jó példája a hibrid modelleknek is:
  - mindegyik korábban tárgyalt általános folyamatmodellből tartalmaz elemeket.
  - támogatja az iterációt, és jól illusztrálja a specifikáció és a tervezés tevékenységeit.

27

RUP

- A RUP felismeri, hogy a konvencionális folyamatmodellek a folyamatoknak csak egy egyszerű nézetét adják.
- Nem egy kész, követendő eljárást ad minden projektre,
  - inkább egy könnyen változtatható keretet azok kézbentartásához.
  - A RUP szabvány általában nagyméretű projektekhez ajánlott, ahol akár több fejlesztő csapat dolgozik közösen egy adott problémán.

28

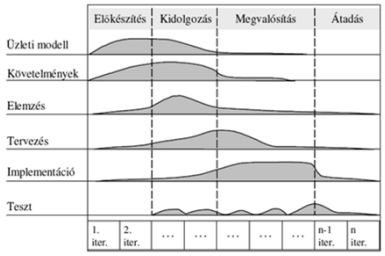
RUP

- A rendszerfejlesztés folyamatát alapvetően **három** dimenzióval írja le:
- **1. dinamikus perspektíva**: a modell fázisait mutatja be.
- **2. statikus perspektíva**: a végrehajtandó folyamattevékenységeket mutatja be.
- **3. gyakorlati perspektíva**: jól hasznosítható gyakorlatokat javasol a folyamat alatt.

29

RUP

- Az időbeliség alapján az RUP a rendszerfejlesztést négy nagyobb egységre, ***négy diszkrét fázisra*** bontja.
  - A fázisok sokkal közelebb állnak az üzleti vonatkozásokhoz, mint a technikaiakhoz.



30

RUP fázisok

- **1. Előkészítés (inception) :**
- Ebben a fázisában a rendszer eredeti ötletét részletes elképzeléssé dolgozzuk át:
  - Amely alapján a fejlesztés tervezhető lesz, a költségei pedig megbecsülhetők.
- Alapvető dolgokat fogalmazunk meg:
  - A felhasználók milyen módon fogják használni a rendszert (használati esetek),
  - és annak milyen alapvető belső szerkezetet, architektúrát alakítunk ki.

31

RUP fázisok

- **2. Kidolgozás (elaboration):**
- Ebben a fázisban a használati módokat, a „használati eseteket" részleteiben kell kidolgozni
  - Össze kell állítani egy stabil alaparchitektúrát *(architecture baseline)*.
- A Unified Process készítőinek gondolata alapján a teljes rendszer egy testnek tekinthető:
  - Csontváz, bőr és izmok.
  - Alaparchitektúra: a bőrrel borított csontváz. Mindössze a minimális összekötő izomzatot tartalmazza, annyit, amennyi a legalapvetőbb mozdulatokhoz elegendő.
  - Az alaparchitektúra segítségével a teljes fejlesztés folyamata ütemezhető és a költségei is tisztázhatók.

32

RUP fázisok

- **3. Megvalósítás (construction):**
- Ezen fázisban a teljes rendszert kifejlesztjük, beépítjük az összes „izomzatot”.
  - Legfőképpen a rendszertervvel, a programozással és a teszteléssel foglalkozik.
  - A rendszer különböző részei párhuzamosan fejleszthetők, és ez alatt a fázis alatt integrálhatók.
  - A fázis teljesítése után már rendelkezünk egy működő szoftverrendszerrel és a hozzá csatlakozó dokumentációval,
  - Ez már leszállítható a felhasználónak.

33

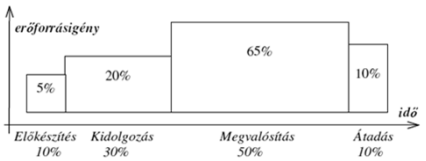
RUP fázisok

- **4. Átadás (transition):**
- A rendszer bétaváltozatának kipróbálását jelenti.
- Ekkor néhány gyakorlott felhasználó teszteli a rendszert
  - jelentést készít annak helyességéről vagy a hibáiról és hiányosságairól.
  - A megjelenő hibákat ki kell küszöbölni, a még hiányzó részeket ki kell fejleszteni.

34

Mérföldkő (milestone)

- Minden fázis vége a fejlesztés egy-egy jól meghatározott **mérföldköve.**
  - Olyan pont, ahol egy célt kell elérni és kritikus döntéseket meghozni.
- Minden fázis végén megvizsgáljuk az eredményeket és döntünk a folytatásról.



Fázisok idő- és erőforrásigénye

35

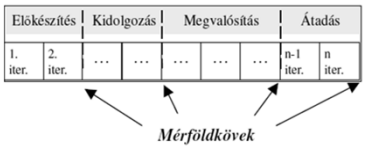
Mérföldkő

- **Iterációk:** a fejlesztés nagyobb egységeit jelentő fázisok további kisebb egységekre bonthatók.
  - egy teljes, illetve részben önálló fejlesztési ciklust jelent.
  - végén egy működő és végrehajtható alkalmazásnak kell előállnia.
- Az iterációk végén így a teljes rendszer egyre bővülő részét kapjuk.
  - Ezek a rendszer egymás utáni kibocsátásai (RELEASE).

36

Mérföldkő

- Az iterációk végén belső változatok alakulnak ki.
  - Verziószámmal jelölt.
- A belső változatok: a fejlesztők kipróbálhatják a rendszert.
  - Segítik a fejlesztést: módosítások eszközölése
  - Tapasztalatok szerzése.



37

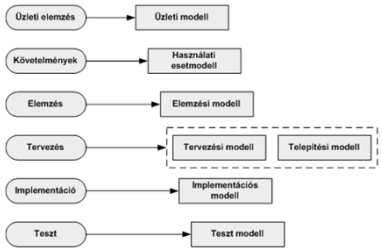
A fejlesztés másik dimenziója

- Meghatározza az eljárás elemeket:
  - a fejlesztés során milyen dokumentumokat, diagramokat, forráskódokat készítünk el.
  - Összesítően: **produktumok** (artifact).
- A produktumokat megfelelő szakértelemmel ellátott személy (dolgozó) állítja össze.
- A tevékenységek, azok időbeli sorrendje és az azt végrehajtó dolgozók együttesen egy **munkafolyamattal** (workflow) írhatók le.
  - a munkafolyamatok leírása UML-modellekkel történik.

38

Modellek

A fejlesztés folyamatában megjelenő tevékenységcsoportok és modellek:



39

Tevékenységcsoportok

- **1. üzleti modellezés***(business model)***:** a fejlesztéshez egy megfelelő kiindulópont keresése.
- Megkeressük a készítendő rendszer üzleti vagy más néven szakterületi környezetét:
  - alapvetően az üzleti fogalmakat és folyamatokat jelentik
  - illetve az azokra hatást gyakorló üzleti munkatársakat (vetélytársakat).

40

Tevékenységcsoportok

- **2. követelmények meghatározása** *(requirements capture):*
  - Összegyűjtjük és felsoroljuk a rendszer működésével szemben támasztott kezdeti elképzeléseket.
  - Leírjuk, hogy a rendszernek milyen környezetben kell működnie,
  - Felsoroljuk a funkcionális és nemfunkcionális követelményeket:
    - Funkcionális: működéssel kapcsolatos.
    - Nemfunkcionális: válaszidők, bővíthetőség, alkalmazott technológiák, stb.
  - A követelmények meghatározása során alapvetően a felhasználók szempontjából írjuk le a rendszert, így annak egy külső képét rögzítjük.

41

Tevékenységcsoportok

- **3. elemzés** *(analysis):*
  - A követelményeket a fejlesztők szempontjának megfelelően rendezzük át, így
    - azok együttessen a rendszer egy belső képét határozzák meg.
    - a további fejlesztés kiindulópontja lesz.
- Rendszerezzük és részletezzük az összegyűjtött használati eseteket, valamint azok alapján meghatározzuk a rendszer alapstruktúráját.
- Cél: a szerkezeti váz kialakítása.

42

Tevékenységcsoportok

- **4. tervezés** *(design):*
  - A szerkezeti vázat teljes alakká formálja, és tartalommal tölti fel.
    - az összes – funkcionális és nem-funkcionális követelménynek is eleget tesz.
  - Az implementációval kapcsolatos összes kérdést meg kell válaszolnia.
    - Amely alapján a rendszer leprogramozható minden kérdés nélkül.
  - Felmerülő kérdések:
    - összes felhasznált technológia, a rendszer részekre bontása, alrendszerek és kapcsolódásuk, protokollok.

43

Tevékenységcsoportok

- **5. *implementáció*** *(implementation):*
  - A rendszert az UML terminológiája szerinti komponensekként állítjuk elő.
    - *Forráskódok, bináris, futtatható állományok, szövegek, képek, stb.*
  - Az állományok előállítása egyben azok függetlenül végrehajtható, önálló tesztjei is.
  - Az iteráció esetén szükséges rendszerintegráció tervezése, az osztottság *(distribution)* tervezése.

44

Tevékenységcsoportok

- **6. teszt** *(test): utolsó munkafolyamat.*
  - Ütemtervek összeállítása:
    - az iterációkon belüli integrációs tesztek ütemtervei
    - az iterációk végén végrehajtandó rendszertesztek ütemtervei.
  - Megtervezzük és implementáljuk a teszteket
    - Tesztesetek előállítása. Programokat készítünk, ha lehetséges a tesztek automatizálása.
  - A tesztek végrehajtásával párhuzamosan azok eredményeit szisztematikusan feldolgozzuk.
  - Hibák vagy hiányosságok esetén újabb tervezési vagy implementációs tevékenységeket hajtunk végre.

45

Köszönöm a figyelmet!

46