

2021. 01. 08.

KARTH SÁTA' PELLÉK

3.

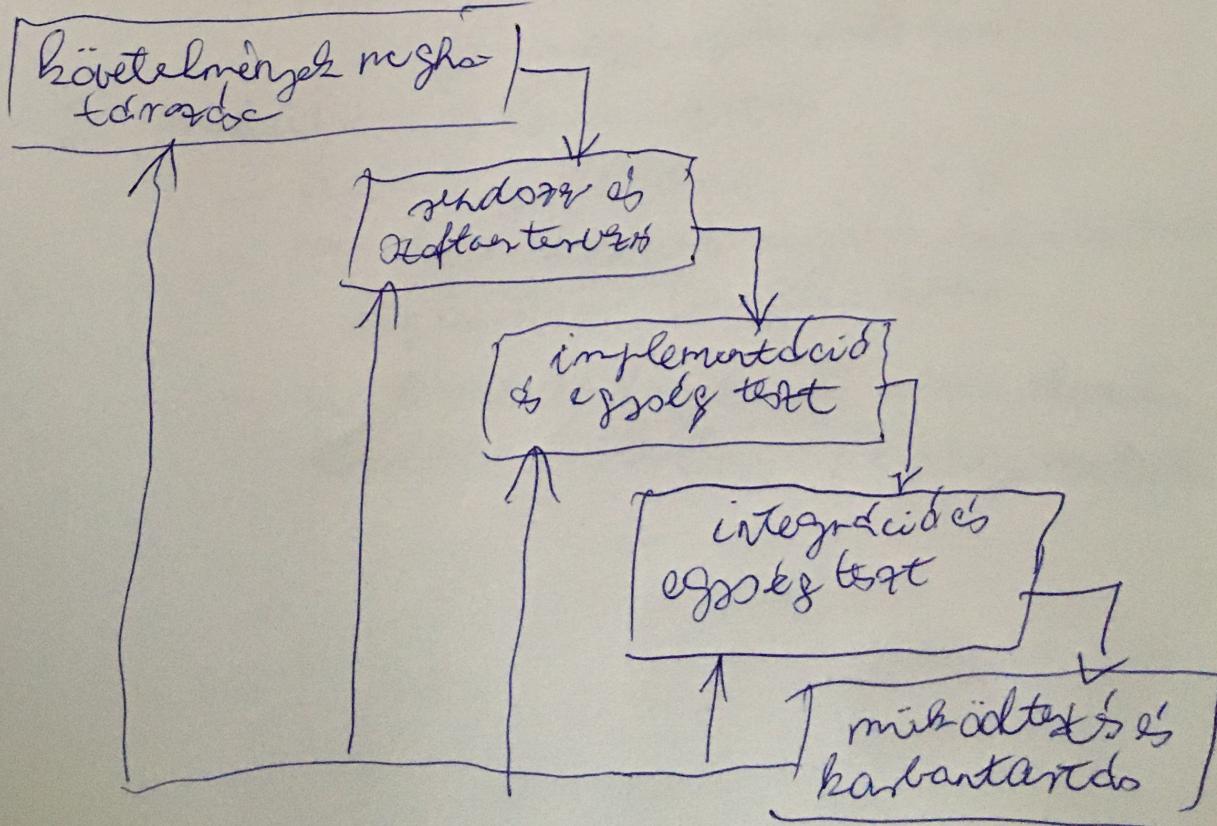
Software életciklus (SDLC) SOFTWARE DEVELOPMENT LIFE CYCLE):

- Visszamodell
- V-modell
- Prototípus modell
- ~~iteratív~~ incrementális fejlesztés
- Gyors alkalmazás fejlesztés
- a Gilek fejlesztés
- extrém programozás

Visszamodell leírásától - :

- gyors
- fázisok során követően egyszerű dokumentum
- egyszerűbb és indukthatóbb az előző végénél
- szélesebb megállományban

~~követelmények meghatározása~~



Folytatás a következő osztályon

2021. 9. 08.

Kauhava Satán PELL 84

3. Vízszabályozás modell fázisai:

1. követelmények elemzése & működési rendszer felhasználása:

- alja
- meghatározai
- rendszer szolgáltatásai mi lesz

2. rendszer és softveren

- rendszer architektúrajának megvalósítása

3. implementáció és tesztelés

- softveren körözés

4. integráció és rendszeresítés

- rendszerekkel való tömörülés, a bővítések programjaihoz integrálása

5. működési és karbantartási

- hibák javítása

- rendszer egységek implementációjainak fejlődése & tavolítsan.

- nyílt követelményeket a rendszer tárolja fejlődési célokra rögzítve

2021. 01. 08.

KARATHI ZOLTÁN PERCPT

B1. 3.

V-modell → "V"-RÉSZES RENDSZER ALB

Rendszerek negatívai.

követelmény
specifikáció

funkcionális
specifikáció

rendszertan

implementációs komponenset

Törzsvégrehajtás

felhasználói átvételek törzst

rendszertörzst

integrációtörzst

fejlesztésben
vissza felelősen
lehet használni

fejlesztés majd törzst

törzst oszt hibát ad akkor vissza a fejlesztőhöz

- egy minden leb fejleszt. és törzsi lépések összetartoznak
a törzsi lépés a fejlesztési lépés oszt hibát letrajtott termékét
törzeli vagy a letrajtott dokumentum. használja

- ha a követelmények nem változnak a fejlesztés alatt, akkor
já modosítan, ha változnak akkor inkább most kell
alkalmazni: iteratív v. agilis modosítan

- V-modellhez felmerül az igényeket és elkerülik a követelmény spec.
követelmény specifi. negatívozott átvételei kritériumot fogal-
maznak funkcionális és nem funkcionális igények) → ez az alapja
az UAT-ek felhasználói átvételei törzsekre. A köv. speci. tényező
gyer minden igényt le kell fedni az ügyfélnek. Rögt köv.
specifikációt oszt nem os igények megfelelő osztásiuk lesz.

- a funkcionális specifikáció leírja, hogyan kell működni a
szoftver (rendszertörzst alapja)

- a rendszertor leírja az egys funkciókat - melyek komplexitással
szemben, melyekkel valószínűleg. A rendszertor leírja
hogyan a komponensek hogyan működnek maguk között. Ez a zinteg-
rácio törzst alapja

* FAVITATA'S KÖRÉTKEZÉS OLDALON!

2021. 01. 08 KOLÁTH ZOLTÁN DEUPA

V-modell

* TÖTTÉTÉS 1.

- Rendszertípusok megfelelő következők az implementáció.
Minden rendszertípusról egy v. több unit - többet kell készíteni
- Az integrálás során minden rendszertípusnak következők.
Hiba esetén viszonylag könnyű a V betű mögötti számra a rendszertípus. Megismerésük / hozzá a hiba rendszertípusnál vagy rendszertípusban van - e. Ha nincs ilyen hibakód, akkor ezeket
- Az integrálás során minden rendszertípusnak következők
specifikációval adjunk. Hiba esetén a V betű mögötti számra megijedt a funkcionális specifikáció.
- Általában több a köb. spec. alapján írjuk nekem lehet hiba, MERT NEM EXISTENCIÁL PROJEKTUNK

2021. 01. 08.

KALÁTHI ZOLTÁN PÉTER

5. KÜLÖNBÖZŐ STRATÉGIÁK

- Bing - bang stratégiák
- Incrementális
- Top - down
- Bottom - up

• Bing - bang stratézia

- minden egység rendelkezésre áll
- legből a teljes rendszert építjük fel
- Incrementális integráció
- a minden elemet felerősítünk
- minden részben többlek nézeteitől

- Top - down integrációs stratézia :

1. legfelső szinten leny testesí
2. (fűz) részrészről kiélezés elemek
3. test sikeres, az ideiglenes elemeket a valódiakkal helyettesíti

- Bottom - up integráció

1. legelőző részben lévő modulok
2. modulokat testeli, melyekkel hossz

2. (test driver) felbőrítését részrészről testesíkonyer

3. test sikeres, a test driverhez a valódi implementált elemeket csatol

2021. 01. 08. KALÁTH ZOLTÁN PELLÉK

1.

Testtelői funkciók:

- testtelői elkötelezettsége: minden kiemelkedő, hogyan kell törtélni, mikor okozna a test. QA-hoz tartozik
- törzsettel törlesztés: minden törzsettel kell meghajtani a test tervezését, mikor elvárt viszonyt kell elérni
- felkerülés a vegrehajtásra: test környezetére vonatkozik (használhatók az előző környezetek)
- testtel vegrehajtás: test napról vezetés, minden leírást hajtunk végre miatt az elvárt. Napról alapján regisztrálhatók a testek
- kihívási feltételök vizsgálat: megvizsgáljuk, hogy kihívás teljesítettük-e a kihívás feltételét. A test-szabályok leíró elvárásai nem minősít használhatóságot a test napról, ha az elvárásnak nem működik
- eredmények értékelése: ennek miatt alapján tanácsot adhatunk ezen lehetséges
- jelentésekkel:
 - információt gyűjtse a törzsről
 - kódban hibák találása
 - hibák: rövid időn belül

21.01.08

KAUFTH TANÁN PELL SA

2.

A

Felürt egy nyílt forrásból modultestet rendszert
(kreativitás)

Testek: test futtadásra vagy az összes metodus
végrehajtása bonyol, ha van kimenet lehet:

- sikeres végrehajtás (pass)
- sikertelen végrehajtás (failure)
- hiba (error)

Sikeres végrehajtás esetén lefutott a testünk és az
ellenőrzési feltételek ellenültek

Sikertelen végrehajtás (failure) a testünk lefutott
de valamelyik feltétel (ellenőrzési) nem ellenül-

Hiba esetén a test futtadás esetén koncreta
probléma keletkezik: pl.: Exception keletkezett

Parametrikus testek: @ Parametrikus annotációval
elhídított publikus statikus metodus, egyszerűsítés
tömbök索引címét adja vissza
egy-egy tömb索引címét az eggy testek
végrehajtásra minden használt adatot el
tartalmaz. Méretük azt mutatja meg hogy
ezek hányszámban vannak az eggy testekben
tömbök minden elemet működő pont ~~az~~ címéi
árványi a ~~test~~ test konstruktora para.
a parametrikus