Szoftverellenőrzési technikák

Specifikáció alapú teszttervezési módszerek

Majzik István, Micskei Zoltán

http://www.inf.mit.bme.hu/

1

Utolsó módosítás: 2014.10.26.

Klasszikus tesztelési feladat

- A tesztelendő program beolvas 3 egész szám paramétert, egy háromszög három oldalának hosszait. Válaszként kiírja, hogy a háromszög általános, egyenlő szárú vagy egyenlő oldalú.
  - » Glen Myers, The Art of Software Testing, 1979
- Milyen teszteket terveznénk ehhez a programhoz?
- Eltérő megoldási javaslatok:
  - Beck: 6 teszt
  - Binder: 65 teszt
  - Jorgensen: 185 teszt!
  - Specifikáció hiányosságok?

2

Leírások:
-http://testdesigners.com/testingstyles/triangleexample.html
-http://www.testingeducation.org/conference/wtst3_collard5.pdf

---
Lehetséges tesztesetek:
3,3,3 -- egyenlő oldalú
5,5,2 -- egyenlő szárú (elég-e ezt csak erre a felállásra megnézni, vagy kell egy 4,7,7 és 9,3,9 teszteset is?)
5,6,7 -- általános
1,2,5 -- nem háromszög (így viszont csak az egyik lehetőséget nézzük meg, az a + c > b, b + c < a eseteket nem. Összetett feltételnek csak az egyik részét vizsgáljuk!)
1,2,3 -- épp nem háromszög (más kombinációkban is tesztelhető)
0,1,2 -- nulla hosszú oldal (lehet más, vagy több is)

0,1,1 -- specifikus hibaüzenet?
-3,-5,-3 -- specifikus hibaüzenet?
2,2,a -- bemenet beolvasása nem volt specifikálva
3,4 -- lehet-e háromnál több vagy háromnál kevesebb bemeneti paramétert megadni?

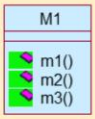
Glen Myers könyvében szereplő ellenőrző kérdések:

1. Do you have a test case that represents a valid scalene triangle? (Note that test cases such as 1, 2, 3 and 2, 5, 10 do not warrant a "yes" answer because there does not exist a triangle having these dimensions.)
2. Do you have a test case that represents a valid equilateral triangle?
3. Do you have a test case that represents a valid isosceles triangle? (Note that a test case representing 2, 2, 4 would not count because it is not a valid triangle.)
4. Do you have at least three test cases that represent valid isosceles triangles such that you have tried all three permutations of two equal sides (such as, 3, 3, 4; 3, 4, 3; and 4, 3, 3)?
5. Do you have a test case in which one side has a zero value?
6. Do you have a test case in which one side has a negative value?
7. Do you have a test case with three integers greater than zero such that the sum of two of the numbers is equal to the third? (That is, if the program said that 1, 2, 3 represents a scalene triangle, it would contain a bug.)
8. Do you have at least three test cases in category 7 such that you have tried all three permutations where the length of one side is equal to the sum of the lengths of the other two sides (for example, 1, 2, 3; 1, 3, 2; and 3, 1, 2)?
9. Do you have a test case with three integers greater than zero such that the sum of two of the numbers is less than the third (such as 1, 2, 4 or 12,15,30)?
10. Do you have at least three test cases in category 9 such that you have tried all three permutations (for example, 1, 2, 4; 1, 4, 2; and 4, 1, 2)?
11. Do you have a test case in which all sides are zero (0, 0, 0)?
12. Do you have at least one test case specifying noninteger values (such as 2.5, 3.5, 5.5)?
13. Do you have at least one test case specifying the wrong number of values (two rather than three integers, for example)?
14. For each test case did you specify the expected output from the program in addition to the input values?

Teszttervezés módszerei

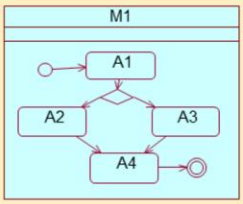
I. Specifikáció alapú

- A rendszer mint „fekete doboz” adott
- Csak a külső viselkedés (funkció) ismert, a belső felépítés (pl. forráskód) nem
- Tesztelés alapja: specifikált funkciók léte; extra funkciók hiánya



II. Struktúra alapú

- A rendszer mint „üvegdoboz” adott
- A belső struktúra is ismert
- Tesztelés alapja a belső működés: programgráf bejárása



3

I. Specifikáció alapú tesztelési módszerek

Cél:

- A funkcionális specifikációra építve,
- reprezentatív adatok keresése

az egyes funkciók teszteléséhez.

Módszerek:

1. **Ekvivalencia particionálás**
2. Határérték-analízis
3. Ok-hatás analízis / Döntési táblák
4. Kombinatorikus módszerek
5. Véges automata alapú
6. Használati eset tesztelés

4

1. Ekvivalencia particionálás

- Equivalence Class Partitioning (ECP)
- Bemenet és kimenet ekvivalencia osztályai:
  - Olyan adatok, amelyek várhatóan ugyanazt a hibát fedik le (ugyanazt a programrészt járják be)
  - Cél: Egy-egy ekvivalencia osztályból egy-egy teszt adat (az adott bemenethez illetve kimenet alapján); a többi adat esetén a helyesség induktívan következik
- Bemenet értelmezését ismerni kell!
  - Tesztelő tudásán múlik a módszer hatékonysága

5

Ekvivalencia osztályok meghatározása

Meghatározás heurisztikus folyamat:

1. Érvényes és érvénytelen bemeneti adatok
2. Partíciók tovább finomítása

Heurisztikák a meghatározáshoz:

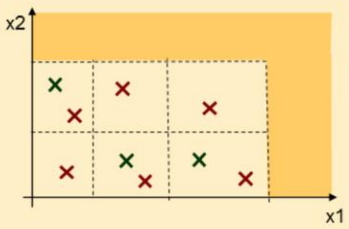
- Tartomány (pl. 1-1000)
  - < min, min-max, >max
- Halmaz (pl. RED, GREEN, BLUE)
  - érvényes elem, érvénytelen
- Specifikus (pl. első karakternek @-nak kell lennie)
  - feltétel teljesül, feltétel nem teljesül
- Egyéni (pl. február hónap)
  - egyéni eset külön partícióba

6

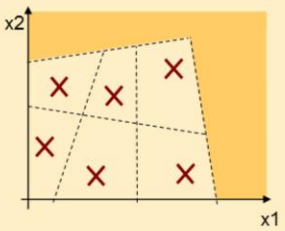
Tesztesetek származtatása

Tesztek meghatározása több bemenet esetén:

- Érvényes ekvivalencia osztályok:
  egy teszt minél több osztályt fedjen le
- Érvénytelen ekvivalencia osztályok:
  először minden érvénytelen osztályhoz külön teszt legyen
  (egymás hatását ne oltsák ki), majd több osztály kombinációja is



- Gyenge ill.
- Erős normál ekvivalencia osztályok



- Dimenziónként nem függetlenül alakítható partíciók: Erős osztályok

7

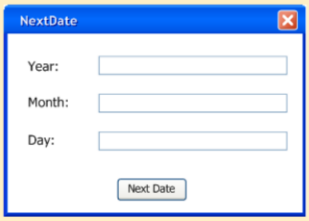
• Gyenge normál ekvivalencia osztályok: minden ekvivalencia osztályból szerepeljen legalább egy elem a tesztesetekben

  • szükséges tesztesetek száma = ekvivalencia osztályok száma azon dimenzió mentén, ami a legtöbb ekvivalencia osztállyal rendelkezik

• Erős normál ekvivalencia osztályok: direkt szorzat minden elemének szerepelnie kell a tesztesetekben

  • szükséges tesztesetek száma = ekvivalencia osztályok számának szorzata

Példa: NextDate program



- NextDate program
- Következő naptári napot határozza meg a Gregorián naptár alapján
- Mik a bemenet ekvivalencia osztályai?
- Mik a kimenet ekvivalencia osztályai?

8

Példa: Ekvivalencia osztályok meghatározása

Bemenet	Érvényes	Érvénytelen
Hónap	V1: 30 napos hónap V2: 31 napos hónap V3: február	I1: >= 13 I2: <= 0 I3: nem szám I4: üres
Nap	V4: 1-30 V5: 1-31 V6: 1-28 V7: 1-29	I5: >= 32 I6: <= 0 I7: nem szám I8: üres
Év	V8: 1582-9999 V9: nem szökőév V10: szökőév V11: század nem szökőév V12: század szökőév	I9: <=1581 I10: >= 9999 I11: nem szám I12: üres
Speciális	V13: 1752.09.03-1752.09.13.	I13: 1582.10.5-1582.10.14.

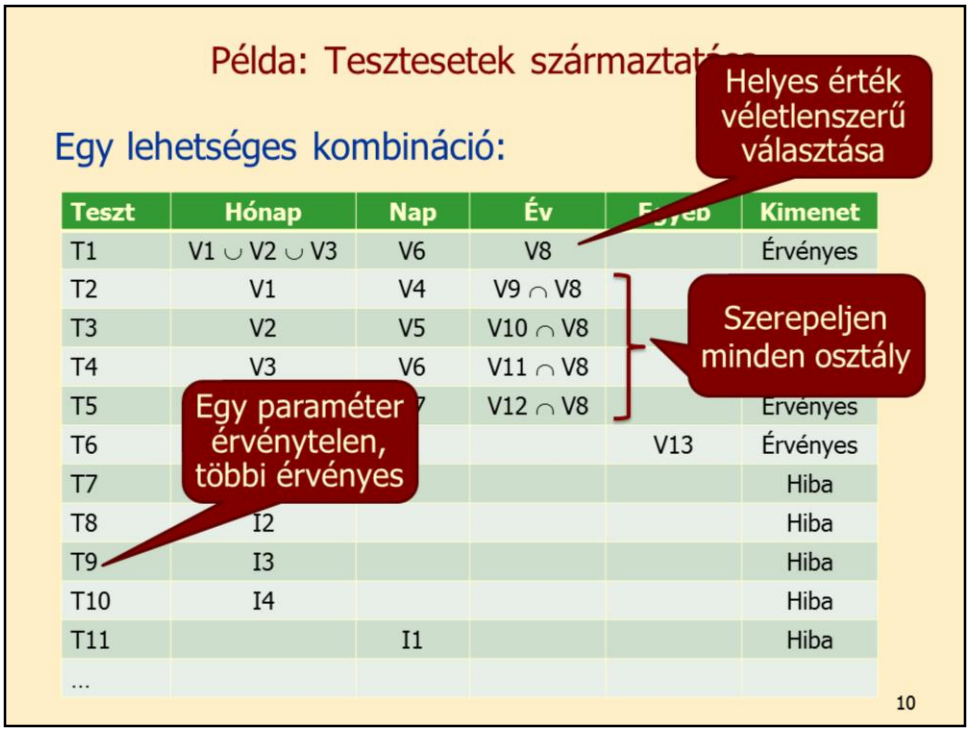
Forrás: „How we test software at Microsoft", Microsoft Press, ISBN 0735624259, 2008.

9

- szökőév: minden néggyel osztható év, kivéve a százzal is oszthatókat. Szökőévek viszont a 400-zal osztható évek.

- V13: Angliában ekkor vezették be a Gergely-naptárat, így ezeket a napokat akkor kihagyták. Ezek a napok speciálisan Angliában nem érvényesek, viszont az általános Gergely-naptár szerint igen

- I13: a Gergely-naptár bevezetésekor ezeket a napokat kihagyták, 1582. október 4-e után 15-e következett



T1: Ha tudunk automatikus, jó és gyors orákulumot készíteni, akkor hasznos lehet az is, hogy az ekvivalencia osztályból nem a teszt tervezéskor választjuk ki az teszt adatot, hanem minden teszt futtatáskor más és más az adott ekvivalencia osztályba tartozó elemet használunk, így növeljük a lefedett lehetséges bemeneteket számát. (Viszont ilyenkor a teszt futtatás során mindenféleképp rögzíteni kell, hogy éppen milyen értéket használtunk, különben hiba esetén nem lesz reprodukálható a probléma!)

T1-T4: itt a pontos kombináció lehet más is, annyi kényszer van csak, hogy V7 esetén szökőévnek megfelelő osztályt kell választani, és minden ekvivalencia osztálynak legalább egyszer szerepelnie kell

I. Specifikáció alapú tesztelési módszerek

Cél:

- A funkcionális specifikációra építve,
- reprezentatív adatok keresése

az egyes funkciók teszteléséhez.

Módszerek:

1. Ekvivalencia particionálás
2. **Határérték-analízis**
3. Ok-hatás analízis / Döntési táblák
4. Kombinatorikus módszerek
5. Véges automata alapú
6. Használati eset tesztelés

11

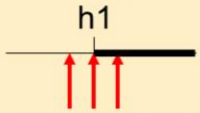
2. Határérték-analízis (Boundary Value Analysis)

- Az adattartományok határait vizsgálja
  - Egy-egy ekvivalencia osztály határaira koncentrál
  - Bemeneti és kimeneti tartományokra is
  - Alsó és felső határokra
- Tipikus megtalált hibák
  - Hibás relációs operátorok
  - Hibák a ciklusok be- és kilépési feltételeinél
  - Hibák az adatstruktúrák méreténél
  - ...

12

Határérték analízis

- Tipikus adatok:
  - Egy határérték 3 tesztet jelent



  - Egy tartomány 5-7 tesztet jelent

13

Példa: határértékek a NextDate esetén

- Mik a határértékek a NextDate esetén?
- Hónap
  - 1, 12
  - Tesztelendő: 0, 1, (2), 3-10, (11), 12, 13
- Nap
  - 1, 31
  - Tesztelendő: 0, 1, (2), 3-29, (30), 31, 32
  - Finomítás: 28, 29, 30 is határérték lehet
- Év
  - 1582, 9999
  - Tesztelendő: 1581, 1582, (1583), 1584-9997, (9998), 9999, 10000

14

I. Specifikáció alapú tesztelési módszerek

Cél:

- A funkcionális specifikációra építve,
- reprezentatív adatok keresése

az egyes funkciók teszteléséhez.

Módszerek:

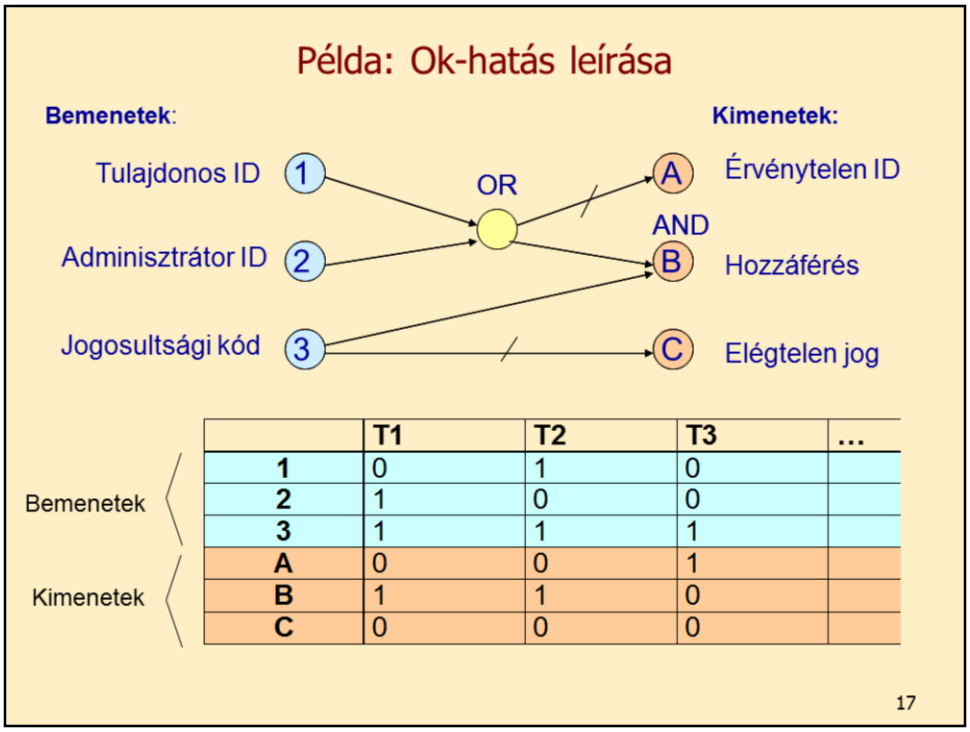
1. Ekvivalencia particionálás
2. Határérték-analízis
3. **Ok-hatás analízis / Döntési táblák**
4. Kombinatorikus módszerek
5. Véges automata alapú
6. Használati eset tesztelés

15

3. Ok-hatás analízis

- A bemenetek és kimenetek kapcsolatának vizsgálata (ha ez egyszerűen leírható)
  - Ok: egy-egy bemeneti ekvivalencia osztály
  - Hatás: egy-egy kimeneti ekvivalencia osztály
  - Ezekből logikai változókat képzünk
- Boole-gráf: Okok és hatások összekapcsolása
  - ÉS, VAGY kapcsolatok
  - Meg nem engedett kombinációk
- Tesztelési cél: A gráf szisztematikus végigjárása
  - Logikai hálózat igazságtáblázatának lefedése
  - Egy oszlop egy tesztnek felel meg

16



	T1	T2	T3	…
1	0	1	0	
2	1	0	0	
3	1	1	1	
A	0	0	1	
B	1	1	0	
C	0	0	0	

(Az áthúzott nyíl a negálást jelenti.)

Tehát olyan szabályaink vannak például, hogy

- T1: ha az adminisztrátor ID-val hajtjuk végre a műveletet és megfelelő a jogosultság, akkor hozzáférünk.

I. Specifikáció alapú tesztelési módszerek

Cél:

- A funkcionális specifikációra építve,
- reprezentatív adatok keresése

az egyes funkciók teszteléséhez.

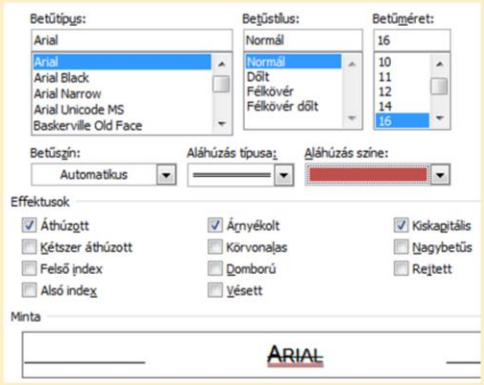
Módszerek:

1. Ekvivalencia particionálás
2. Határérték-analízis
3. Ok-hatás analízis / Döntési táblák
4. **Kombinatorikus módszerek**
5. Véges automata alapú
6. Használati eset tesztelés

18

4. Kombinatorikus módszerek

- Paraméterek kombinációja
  - Paraméterek kombinációja okozza a legtöbb hibát
  - 3-nál több paraméter esetén már rengeteg eset
  - Ritka kombinációk veszélyesek lehetnek



19

Betűtípus beállító ablak:

-sok paraméter, néhánynak nagy értékkészlete is van

-Legalább 50*4*72*256*15*256*2*2*2*2*2*2*2*2*2*2*2= 1,5*10^13 eset

-Paraméterek összefüggenek

-Pl. nem lehet egyszerre felső és alsó index is; bizonyos betűtípus esetén adott stílus nem létezik

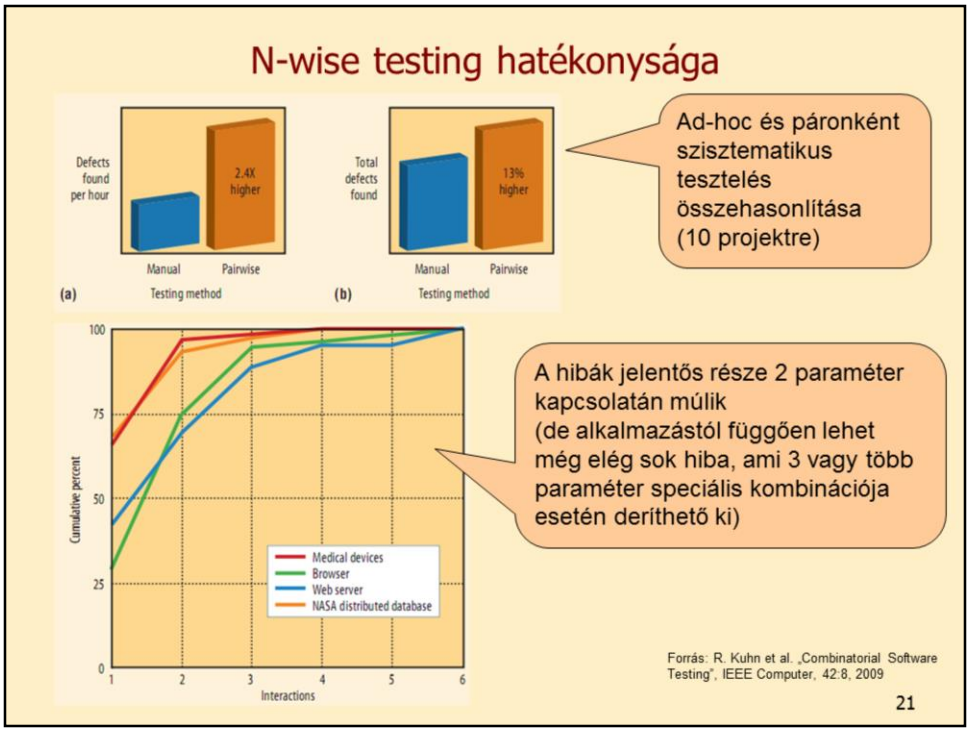
-Paraméterek együtt határozzák meg az eredményt

-Ha van aláhúzás, akkor azt is kell árnyékolni

Kombinatorikus tesztelési technikák

- Ad hoc, „best guess”
  - Intuíció, követelmények, tipikus hibák alapján
- Minden választás (each choice)
  - Minden lehetőség szerepeljen egyszer
  - Alap tesztkészletnek hasznos csak
- N-szeres tesztelés (n-wise testing)
  - Tetszőlegesen választott n darab paraméter minden lehetséges kombinációjának lefedése a tesztelési cél
  - Elnevezés még: n-wise coverage
  - Speciális eset (n=2): Páronkénti tesztelés (pair-wise testing)

20



Felső kép: 10 projekt kapcsán egy tesztelő csapat ad hoc módszerek alapján készített teszteseteket, egy másik pedig pairwise testing technikát alkalmazott

Alsó kép: A hibák jelentős része 2 paraméter kapcsolatán múlik, de azért alkalmazástól függően lehet még elég sok hiba, ami 3 vagy többi paraméter speciális kombinációja esetén jön csak elő

Példa: Pair-wise tesztelés

- Adottak a következő konfigurációs lehetőségek:
  - OS: Windows, Linux
  - CPU: Intel, AMD
  - Protocol: IPv4, IPv6
- Kombinációk száma?
- Páronkénti tesztelést megvalósító tesztkészlet?
- Lehetséges megoldás:
  - 1: Windows, Intel, IPv4
  - 2: Windows, AMD, IPv6
  - 3: Linux, Intel, IPv6
  - 4: Linux, AMD, IPv4

22

N-szeres tesztelés a gyakorlatban

- Feladat: coverage array előállítása

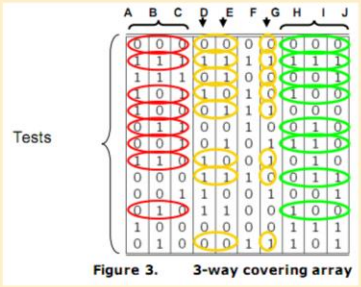


Figure 3. 3-way covering array

Forrás: D. R. Kuhn, R. N. Kacker, Y. Lei
Practical Combinatorial Testing
NIST Special Publication 800-142

- Támogató eszközök
  - http://www.pairwise.org
  - PICT - Pairwise Independent Combinatorial Testing (MS)
  - ACTS - Advanced Combinatorial Testing Suite (NIST)

23

I. Specifikáció alapú tesztelési módszerek

Cél:

- A funkcionális specifikációra építve,
- reprezentatív adatok keresése

az egyes funkciók teszteléséhez.

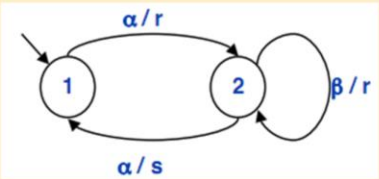
Módszerek:

1. Ekvivalencia particionálás
2. Határérték-analízis
3. Ok-hatás analízis / Döntési táblák
4. Kombinatorikus módszerek
5. **Véges automata alapú**
6. Használati eset tesztelés

24

5. Véges automata alapú

- Specifikáció egy véges automatával adott
- Tipikus tesztelési célok:
  - Minden állapot, minden átmenet, nem megengedett átmenetek tesztelése, stb.



- Problémák:
  - Milyen állapotban van a rendszer?
  - Végállapot / kezdőállapot
- Módszerek
  - Automatikus tesztgenerálás (ld. később)
  - W, Wp módszerek

25

Bővebb információ: **Model-Based Testing of Reactive Systems**

Advanced Lectures Series: Lecture Notes in Computer Science , Vol. 3472, ISBN: 3-540-26278-4

I. Specifikáció alapú tesztelési módszerek

Cél:

- A funkcionális specifikációra építve,
- reprezentatív adatok keresése

az egyes funkciók teszteléséhez.

Módszerek:

1. Ekvivalencia particionálás
2. Határérték-analízis
3. Ok-hatás analízis / Döntési táblák
4. Kombinatorikus módszerek
5. Véges automata alapú
6. **Használati eset tesztelés**

26

6. Használati eset tesztelés

- Tesztek származtathatók a használati esetekből
- Tesztesetek:
  - 1 teszt: fő ág („happy path”, „mainstream”)
    - Ellenőrzés: utófeltételek vizsgálata
  - Alternatív lefutások: mindegyikhez külön teszteset
  - Előfeltételek (nem)teljesülése
- Tipikusan integrációs és elfogadási tesztek

27

Módszerek együttes alkalmazása

Alap módszerek tipikus sorrendje:

1. Ekvivalencia particionálás
2. Határérték-analízis
3. Ok-hatás analízis, vagy kombinatorikus, vagy véges automata alapú

Kiegészítés: Véletlen tesztek

- Véletlen teszt adatok generálása
- Kis számítási teljesítményt igényel, gyors
- Hibafedése nem garantálható
- Teszt eredmény kiértékelése:
  - Válasz számítása, szimulálása
  - Csak „elfogadhatósági vizsgálat” (durva hibák kiszűrése)

28