

Szoftverellenőrzési technikák (vimim148)

Specifikáció alapú teszttervezési módszerek

Majzik István, Micskei Zoltán

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

1

Utolsó módosítás: 2010.10.18.

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 tesztek terveznénk ehhez a programhoz?
- Eltérő megoldási javaslatok:
 - Beck (6 teszt), Binder (65 teszt), Jorgensen (185)
 - 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

0,1,1 -- specifikus hibaüzenet?

-3,-5,-3 -- specifikus hibaüzenet?

2,2,a -- bemenet beolvasása nem volt specifikálva

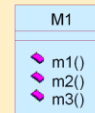
3,4 -- lehet-e háromnál több vagy háromnál kevesebb bemeneti paramétert megadni?

Glen Myers könyvében szereplő ellenőrző kérdések:

Teszttervezés módszerei

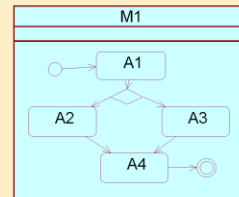
I. Specifikáció alapú

- 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: **belső működés**: programgráf bejárása



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

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 tartományát, szakterületét ismerni kell!
 - Tesztelő tudásán múlik a módszer hatékonysága

Ekvivalencia osztályok meghatározása

Meghatározás heurisztikus folyamat:

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

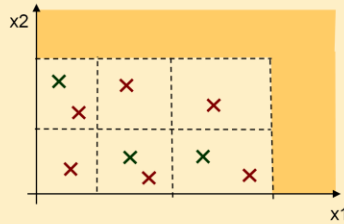
Heurisztikák a meghatározáshoz:

- Tartomány (pl. 1-1000)
 - $< \text{min}$, min-max , $>\text{max}$
- 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

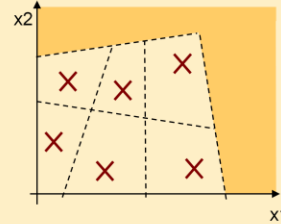
Tesztesetek származtatása

Tesztek meghatározása több bemenet esetén:

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



- Gyenge ill.
- **Erős** normál ekvivalencia osztályok



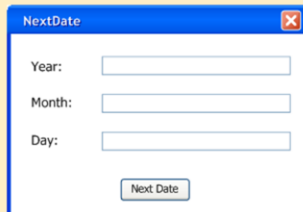
- Dimenzióként nem függetlenül alakítható partíciók: **Erős** osztályok

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

A screenshot of a small graphical user interface window titled "NextDate". The window has a blue title bar with a close button (X) in the top right corner. Inside the window, there are three text input fields labeled "Year:", "Month:", and "Day:". Below these fields is a button labeled "Next Date".

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

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 négygyel osztható év, kivéve a százal is oszthatókat. Szökőévek viszont a 400-zal osztható évek.
- V13: Angliában ekkor vezették be a Gergely-naptárt, 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

Példa: Tesztesetek származtatása

Egy lehetséges kombináció:

Teszt	Hónap	Nap	Év	Egyéb	Kimenet
T1	V1 ∪ V2 ∪ V3	V6	V8		Érvényes
T2	V1	V4	V9 ∩ V8		
T3	V2	V5	V10 ∩ V8		
T4	V3	V6	V11 ∩ V8		
T5		V7	V12 ∩ V8		Érvényes
T6				V13	Érvényes
T7					Hiba
T8	I2				Hiba
T9	I3				Hiba
T10	I4				Hiba
T11		I1			Hiba
...					

Helyes érték
véletlenszerű
választása

Szerepeljen
minden osztály

Egy paraméter
érvénytelen,
többi érvényes

T1: Ha tudunk automatikus, jó és gyors órakulumot 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 szököévnnek 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:

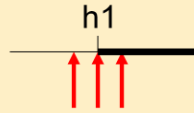
1. Ekvivalencia particionálás
- 2. Határérték-analízis**
3. Ok-hatás analízis / Döntési táblák
4. Kombinatorikus módszerek
5. Véges automata alapú
6. Használati eset tesztelés

2. Határérték-analízis (Boundary Value Analysis)

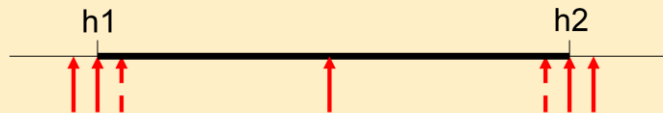
- Az adattartományok határait vizsgálja
 - egy-egy ekvivalencia osztály hatáira koncentrálni
 - bemeneti és kimeneti tartományokra is
 - alsó és felső határookra
- Tipikus megtalált hibák
 - Hibás relációs operátorok
 - Hibák a ciklusok be- és kilépési feltételeinél
 - Hibák az adatstruktúrák méreténél
 - ...

Határérték analízis

- Tipikus adatok:
 - egy határérték 3 tesztet jelent



- egy tartomány 5-7 tesztet jelent



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

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

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

Döntési táblázat: A gráf szisztematikus végigjárása

- logikai hálózat igazságtáblázata
- egy oszlop egy tesztnek felel meg

Ok-hatás leírása

Bemenetek:

Tulajdonos ID **1**

Adminisztrátor ID **2**

Jogosultsági kód **3**

OR

Kimenetek:

A Érvénytelen ID

AND

B Hozzáférés

C Elégtelen jog

		T1	T2	T3
Bemenetek	1	0	1	0
	2	1	0	0
	3	1	1	1
Kimenetek	A	0	0	1
	B	1	1	0
	C	0	0	0

I. Specifikáció alapú tesztelési módszerek

Cél:

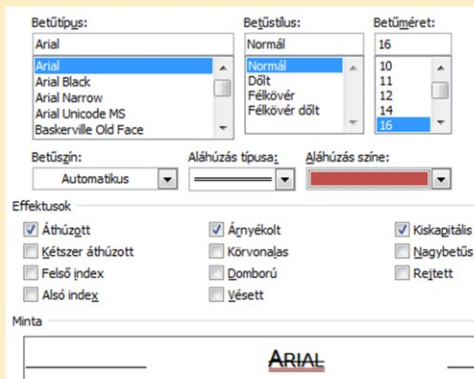
- A funkcionális specifikációra építve,
- reprezentatív adatok keresése az egyes funkciók teszteléséhez.

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. Kombinatorikus módszerek

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



19

Betűtípus beállító ablak:

-sok paraméter, néhánynak nagy értékkészlete is van

-Legalább $50 \cdot 4 \cdot 72 \cdot 256 \cdot 15 \cdot 256 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 1,5 \cdot 10^{13}$ eset

-Paraméterek összefüggenek

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

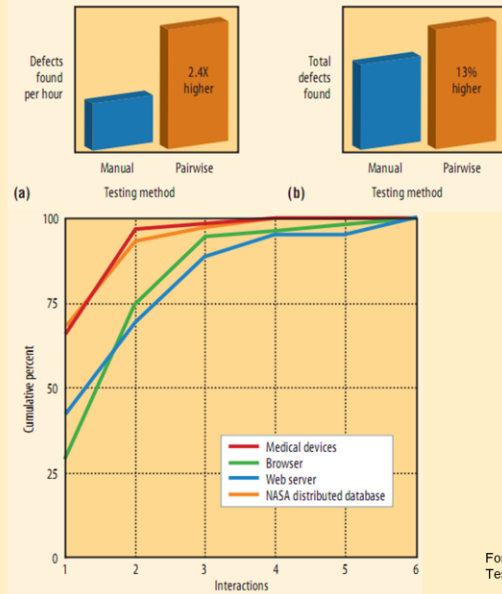
-Paraméterek együtt határozzák meg az eredményt

-Ha van aláhúzás, akkor azt is kell árnyákolni

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)
 - tetszőleges n darab paraméter minden lehetséges kombinációját lefedtük
 - elnevezés még: n-wise coverage
 - speciális eset ($n = 2$): pair-wise testing

N-wise testing hatékonysága



21

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?
- Pair-wise fedést megvalósító tesztkészlet?
- Lehetséges megoldás:
 - 1: Windows, Intel, IPv4
 - 2: Windows, AMD, IPv6
 - 3: Linux, Intel, IPv6
 - 4: Linux, AMD, IPv4

N-szeres tesztelés a gyakorlatban

- Feladat: coverage array előállítás



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)

I. Specifikáció alapú tesztelési módszerek

Cél:

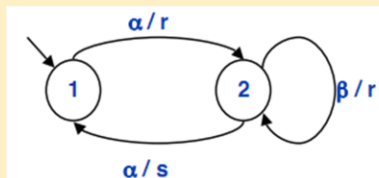
- A funkcionális specifikációra építve,
- reprezentatív adatok keresése az egyes funkciók teszteléséhez.

Módszerek:

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

5. Véges automata alapú

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



- Problémák:
 - State identification: milyen állapotban van a rendszer
 - Homing sequence: végállapot egyértelmű
- Módszerek
 - W, Wp módszer
 - Distinguishing sequence

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**

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

Módszerek együttes alkalmazása

Tipikus sorrend:

1. Ekvivalencia particionálás
2. Határérték-analízis
3. Ok-hatás analízis

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)