

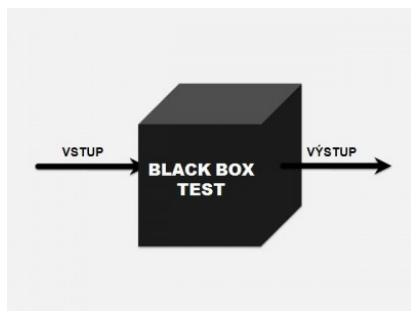
IV113 Validace a verifikace

Techniky Testování I

doc RNDr. Jiří Barnat, Ph.D.

Black-box testování

- Na testovaný produkt je nahlíženo jako na černou skříňku.
- Vnitřní chování produktu je pro testera nepozorovatelné.
- Produkt je analyzován skrze jeho vstup-výstupní chování.
- Testování nezohledňuje zdrojový kód produktu.



White-box testování (Glass-box)

- Vnitřní chování produktu je možné pozorovat a využít.
- Tvorba a provádění testů vzhledem k vnitřnímu chování produktu, např. pro dosažení určitého stupně pokrytí.
- Vložení vnitřních chyb, kódu do produktu za účelem provedení vybraných testů.
- **Často rozšiřuje testování Black-box technikou.**

Gray-box testování

- Mezi Black-box a White-box testováním.
- Některými autory neodlišováno od White-box testování.

Obecná strategie

1. Proveďte základní jednoduché testy.
 - Pokud program selže na těchto jednoduchých testech, nemá smysl dále pokračovat. Jedná se o fatální chybu v systému.
2. Testujte jednotlivé funkce bez zaujetí kritického postoje.
 - Je dobré prvně pochopit důvod vzniku a smysl existence funkce. Teprve poté je vhodné ji podrobit kritice. (Proč je zde taková funkce? Kdy a jak ji bude uživatel potřebovat? Čím konkrétně naplní tato funkce uživatelskou potřebu?)
3. Testujte prvně raději do šířky, než do hloubky.
 - Snažte se testy postihnout základní funkcionalitu všech komponent systému.
 - Než začne hlubší testování, produkt by měl být bez zjevných zásadních chyb.

4. Pokud bylo dosud vše v pořádku, začněte s důkladnějším testováním.
 - Otestujte chování produktu s krajními hodnotami vstupů.
 - Aplikujte jednu či více zvolených technik testování.
5. Zaměřte se a testujte jednotlivé funkce systému do hloubky.
 - Dlouhodobé testy,
 - Testy jednotlivých komponent pod zátěží.
 - ...
6. Průběžně testujte “volným stylem”.
 - Pokud čas dovolí, zkoušejte co vás napadne.
 - Motto: každý týden jeden nový druh testu.
 - Buďte inovativní.

Techniky testování

- Doménové testování
- Kombinatorické testování
- Regresní testování
- Scénářové testování
- Funkční testování
- Fuzz testování
- Risk-based testování
- ...

Je výčet úplný?

- Publikováno víc jak 100 technik \Rightarrow výčet není úplný.

Je dána následující specifikace

- Program sčítá dvě celá čísla, která se mu zadají na vstup
- Každé číslo by mělo mít jednu nebo dvě cifry
- Program vytiskne součet
- Program očekává “Enter” za každým číslem
- Program se spouští příkazem `adder`

Jaké jsou nedostatky specifikace?

Je dána následující specifikace

- Program sčítá dvě celá čísla, která se mu zadají na vstup
- Každé číslo by mělo mít jednu nebo dvě cifry
- Program vytiskne součet
- Program očekává “Enter” za každým číslem
- Program se spouští příkazem adder

Jaké jsou nedostatky specifikace?

- Pracuje se zápornými čísly? (Ano)
- Jak se program ukončí? (Ctrl+C)
- Vytiskne kam? (Na display/obrazovku)

- Jaký bude základní test?

- Jaký bude základní test?
(něco ala $3+7$)
- Kolik je možných specifikovaných vstupů?

- Jaký bude základní test?
(něco ala $3+7$)
- Kolik je možných specifikovaných vstupů?
(celkem $199 \times 199 = 39,601$)
- Budete je testovat všechny?

- Jaký bude základní test?
(něco ala $3+7$)
- Kolik je možných specifikovaných vstupů?
(celkem $199 \times 199 = 39,601$)
- Budete je testovat všechny?
(je to možné, ale ne)
- Budete testovat $3+8$, $4+7$, $2+7$, $3+6$, $3+3$?

- Jaký bude základní test?
(něco ala $3+7$)
- Kolik je možných specifikovaných vstupů?
(celkem $199 \times 199 = 39,601$)
- Budete je testovat všechny?
(je to možné, ale ne)
- Budete testovat $3+8$, $4+7$, $2+7$, $3+6$, $3+3$?
(zřejmě ne)
- Budete testovat 100 a více, a -100 a méně?

- Jaký bude základní test?
(něco ala $3+7$)
- Kolik je možných specifikovaných vstupů?
(celkem $199 \times 199 = 39,601$)
- Budete je testovat všechny?
(je to možné, ale ne)
- Budete testovat $3+8$, $4+7$, $2+7$, $3+6$, $3+3$?
(zřejmě ne)
- Budete testovat 100 a více, a -100 a méně?
(ano)

- Jaký bude základní test?
(něco ala $3+7$)
- Kolik je možných specifikovaných vstupů?
(celkem $199 \times 199 = 39,601$)
- Budete je testovat všechny?
(je to možné, ale ne)
- Budete testovat $3+8$, $4+7$, $2+7$, $3+6$, $3+3$?
(zřejmě ne)
- Budete testovat 100 a více, a -100 a méně?
(ano)

= **Doménové testování**

Doménové testování

Princip techniky

- Všech možných hodnot vstupů je příliš mnoho.
- Testování podobných hodnot vstupů je nezajímavé, testy jsou **ekvivalentní** např. vzhledem k průchodu grafem toku řízení.
- Je vhodné omezit provádění podobných testů, naopak je vhodné realizovat testy s hraničními a jinak zajímavými hodnotami vstupů.

”Matematický pohled”

- Relace ekvivalence na množině testů umožňuje rozdělit prostor všech možných testů na třídy ekvivalence.
- Produkt lze otestovat s použitím reprezentantů tříd rozkladu.

Tabulkou hraničních případů (THP)

Proměnná	Třída ekvivalence platných vstupů	Třída ekvivalence neplatných vstupů	Hraniční a zvláštní případy	Poznámky
První číslo	$[-99,99]$	< -99 > 99	$-100, -99$ $99, 100$	
Druhé číslo	$[-99,99]$	< -99 > 99	$-100, -99$ $99, 100$	
Součet	$[-198,198]$	< -198 > 198	$(-99, -99)$ $(99, 99)$ $(-99, 99)$ $(99, -99)$	Nelze vytvořit test s neplatnou hodnotou

Tabulka hraničních případů (THP)

- Nepřímá definice tříd ekvivalence
- Nedílná součást testovací dokumentace
- Klíčový nástroj pro tvorbu v testovacího plánu

THP jako součást testovacího plánu

- Stanovuje, co může být předmětem testu
- Zahrnuje očekávané výsledky testů
- Slouží jako kompaktní seznam testů
- Umožňuje identifikaci testů
- Lze použít i jako míru testování

Zkušenosti z praxe

- Získat kompletní THP je náročné.
- V procesu testování je THP budována/doplňována během procesu testování.
- Kompletní THP se v praxi nedosahuje.
- THP obsahuje všechny proměnné, ale pouze u kritických proměnných je vyplněna.

Důvody

- Specifikace se tvoří za běhu. Samotný proces testování odkrývá nové vstupní proměnné.
- Proměnných je příliš mnoho, analýza mnohých proměnných by proběhla pouze za účelem vyplnění THP.

Příklad

- Vstupní hodnota je desetinné číslo x z intervalu $(y, z]$.

Možné řešení

- Δ – nejmenší možná změna ovlivňující výsledek testu.
- Hraniční hodnoty: $y, y + \Delta, z, z + \Delta$

Δ ovlivněna následujícími faktory:

- Jaká je přesnost vstupního pole.
- Jaká je přesnost jednotlivých procedur, které s hodnotou nějakým způsobem pracují.
- S jakou přesností se hodnota projeví ve výsledku.

Uspořadatelné množiny

- Testování reprezentantů tříd ekvivalence a identifikace hraničních hodnot nejsou svázány pouze s číselnými doménami.
- Obecně lze doménovým přístupem analyzovat vstupy jejichž potenciální hodnoty lze uspořádat.

Pozorování

- Pokud entity nelze uspořádat, lze využít obecných měřitelných a uspořadatelných vlastností.
 - Velikost a počet entit.
 - Časování událostí.
 - Pozorovaná granularita (např. rozlišení)
 - ...

Specifikace (Gerald Weinberg, 1969; Glen Myers 1979)

- Program přečte tři čísla a interpretuje je jako délky stran trojúhelníku
- Na výstup vypíše, zda se jedná o trojúhelník rovnostranný, rovnoramenný, či obyčejný (ani rovnostranný ani rovnoramenný).

Jaké jsou třídy rozkladu a hraniční hodnoty?

Specifikace (Gerald Weinberg, 1969; Glen Myers 1979)

- Program přečte tři čísla a interpretuje je jako délky stran trojúhelníku
- Na výstup vypíše, zda se jedná o trojúhelník rovnostranný, rovnoramenný, či obyčejný (ani rovnostranný ani rovnoramenný).

Jaké jsou třídy rozkladu a hraniční hodnoty?

- Obyčejný, rovnoramenný, rovnostranný trojúhelník.
- Délky, které netvoří trojúhelník.
- Délky, které tvoří trojúhelník s nulovou výškou.
- Případy s jednou a více nulovými délkami.
- Záporná délka strany.
- Zadány 2 nebo 4 hodnoty.
- Zadány nečíselné hodnoty.

Hodnoty vstupů více proměnných, jejichž kombinace je neplatná, avšak každá zvlášť je kombinovatelná s jinými hodnotami tak, aby tvořila platnou kombinaci.

Příklad – Trojúhelník

- [3,3,6], [3,6,3], [6,3,3]

Příklad – NHL computer game

- Podle bodů ze základní skupiny (vítěství 2, remíza 1, prohra 0) se určí 8 týmů, které postupují do play-off.
- 80 zápasů v základní části.
- Součet bodů uchováván v signed byte [-128,127].
- Jaká je pozorovaná veličina a jaká její hraniční hodnota?

Příklad – Adder

- Prodleva před zadáním druhého čísla.
- Zadání příliš velkého čísla.
- Vedoucí nuly či mezery.
 - Nechť $009 = 09 = 9$. Platí, že $000000000000000009 = 9$?
- Nenumerický vstup.
- Špatné chování v obsluze špatného vstupu.
(Je specifikováno?)

Příklad adder – rozšíření THP

Proměnná	Třída ekvivalence platných vstupů	Třída ekvivalence neplatných vstupů	Hraniční a zvláštní případy	Poznámky
První číslo	[-99,99]	< -99 > 99 desetinné nečísla výrazy	-100,-99 99, 100 2.5 '/' '.' 0, null	
Druhé číslo	- -	- -	- -	

Pozorování

- Specifikace je nedokonalá a dává možnost být interpretována různě (ač správně).

Příklad specifikace

- Uvažme algoritmus, který plánuje pro hypotetickou firmu rozvoz zakázek na následující den.
- Politika firmy je preferovat zákazníky, kteří mají platné speciální povolení od managementu být přednostně obslouženi, nebo nebyli obslouženi už 30 dnů.

Nepřesnosti vzhledem k hraničním hodnotám

- Nebyli obslouženi víc jak 30 dnů, nebo 30 dnů včetně?
- 30 dnů měřeno v době plánování, či v době dodávky?
- Dny měřeny od půlnoci, či od hodiny dodávky?
- Kdy se posuzuje platnost povolení?

Princip

- Vycházíme od možného problému a zpětným postupem hledáme vstupní data, která vedou k vyvolání problému.
- Na základě nalezených hodnot redefinujeme třídy ekvivalence.

Postup

- Identifikujeme problém.
- Kategorizujeme testy na *způsobí/nezpůsobí* chybu.
- Pokud neexistuje test, který problém vyvolá, je třeba podniknout modifikaci definice domén, aby odpovídající test nemohl být vynechán.

Pozorování

- Existují případy, kdy jednotlivé možné hodnoty vstupní proměnné nelze uspořádat.
- Nelze identifikovat hraniční případy.
- Jak testovat vstupní proměnnou pokud může nabývat velkého množství hodnot?

Doménové testování neuspořadatelných množinách

- Rozlišení objektů na základě jejich atributů.
- Identifikace ekvivalence a odpovídajících tříd.
- Volba vhodných reprezentantů jednotlivých tříd.

Testování výstupu na tiskárnu

- Existuje víc jak tisíc různých tiskáren.
- Požadujeme kompatibilitu i se staršími typy.
- Třídy / Podtřídy
 - Třídy: LJ II, LJ III, Postscript Level I, Postscript Level II, Epson
 - Podtřídy LJ II: LJ II, LJ II+, LJ IIP, LJ IID

Pozorování

- Jednotlivé nedostatky se projevují zřetelněji na různých typech tiskáren. (Analogie s uspořádanými množinami.)

Doporučení

- Je vhodné volit reprezentanty tak, aby pokryly co možná nejvíce nedostatků.

Příslušnost do skupiny

- Modré, červené, zelené objekty.
- Zaměstnanci, zaměstnavatelé.

Ekvivalentní HW

- Zvukové a grafické karty, síťové prvky.

Ekvivalentní prostředí, ve kterém produkt pracuje

- Windows 98/ME/2000/XP/Vista
- Lokalizace – do kolika jazyků bylo/bude lokalizováno?

Historie

- Domény (obory hodnot) a sub-domény byly v testování používány mnohem dříve, než bylo v rámci softwarového testování teoreticky doménové testování popsáno (1988, Testing Computer Software).

Teoretický pohled

- Doména je množina prvků, na kterých je pomocí nějaké relace ekvivalence definován rozklad.
- Testování domény probíhá pomocí vybraných reprezentantů jednotlivých tříd.

Intuitivní podobnost

- Dvě hodnoty (vstupních proměnných) jsou ekvivalentní, pokud jsou natolik podobné jedna druhé, že testování obou hodnot se zdá být nadbytečné.
 - Snadno pochopitelná a lehce aplikovatelná metoda.
 - Nevhodná na odlišení jemných detailů nebo na složité kombinace vstupů.

Specifikováno jako ekvivalentní

- Dvě hodnoty jsou ekvivalentní, pokud specifikace říká, že obě hodnoty program zpracovává stejným způsobem.
 - Testeři nuceni doplňovat specifikaci v nspecifikovaných částech.
 - Chybový program nemusí sledovat specifikaci.
 - Zaměřeno na to co bylo specifikováno, ne na to co nebylo specifikováno, či bylo specifikováno nedostatečně.

Ekvivalentní cesty

- Dvě hodnoty jsou ekvivalentní, pokud donutí program vykonávat se stejným způsobem, například projít stejnou větví daného větvení.
 - Posoudit ekvivalenci může programátor, nikoliv tester.
 - Návrh testů vychází ze zdrojového kódu.
 - Test všech hodnot v doméně nevynucuje kompletní pokrytí.
 - Dvě hodnoty mohou donutit projít program po stejné cestě, ale vyvolat odlišné efekty. (Timeout, odlišné výstupy na tiskárně,)

Shodné riziko

- Dvě hodnoty jsou ekvivalentní, pokud lze očekávat, že vzhledem k dané možné chybě, se testy nad oběma hodnotami budou chovat stejně, tj. oba testy chybu vyvolají, nebo ji nevyvolá ani jeden z testů.
 - Hodnoty mohou být ekvivalentní k jedné chybě, ale neekvivalentní k jiné chybě.
 - Subjektivní metrika, závisí na orákulu.

Pozorování

- Hraniční případy jsou dobrou volbou.

Příklad

INPUT < 10 \implies Error message

10 <= INPUT < 25 \implies Print "Hello"

INPUT >= 25 \implies Error message

- Možné chyby
 - Program nezpracuje správně číslo a INPUT=0
 - Špatně specifikovaná nerovnost (<= 25 místo < 25)
 - Překlep (52 místo 25)
- Hraniční hodnota 25 odhalí všechny 3 chyby
- Jiná hodnota (například 53) nemusí

Nevýhody hraničních případů

- Volba hraniční hodnoty jako reprezentanta je heuristika.
- Hraniční případ nemusí být ten nejvhodnější kandidát.
- Nejlepší reprezentant v dané třídě je ten, který má největší pravděpodobnost vyvolat chybu.
- Neexistují v neuspořadatelných doménách.

Praxe

- Volí se hraniční a zajímavé nehraniční případy.

Pozorování

- Závislost vstupních proměnných může způsobit, že daná hodnota jedné proměnné je hraničním případem pouze při konkrétní hodnotě jiné vstupní proměnné.

Doporučení

- V takovém případě je vhodné zvolit všechny hodnoty, které mohou být v nějaké konfiguraci hraničním případem.

Příklad závislých proměnných — měsíc a den v měsíci

- Měsíc: 2 – 28 až 29 dnů
- Měsíc: 4,6,9,11 – 30 dnů
- Měsíc: 1,3,5,7,8,10,12 – 31 dnů

Výhody

- Základní nástroj pro boj s neúplností testování zachovávající rozumné a rovnoměrné pokrytí stavového prostoru možných testů.
- Analýza hraničních hodnot domén vstupních proměnných dává přehled o rozsahu a množství nezbytných testů.
- Hraniční případy vstupních proměnných jsou případy, při kterých se často projeví chyby produktu.

Nevýhody

- Hraniční hodnoty mohou být nejasné nebo skryté, navíc dochází ke zjemňování rozkladu v průběhu procesu.
- Neredukuje množství kombinací způsobené binárními vstupy a počtem vstupních proměnných.
- Falešný pocit bezpečí.

Kombinatorické testování

Pozorování

- Při testování mnoha vstupních proměnných, počet možných testů roste exponenciálně.

Cíl kombinatorického testování

- Pokrýt exponenciální prostor možných vstupů rozumným způsobem s výrazně menším počtem testů.

Princip techniky

- Identifikace podmnožiny možných testů, které splňují jisté kombinatorické vlastnosti.

Mechanické (procedurální)

- Testované kombinace vstupů jsou určeny/vypočteny mechanickým postupem.

Založené na možném riziku

- Testované kombinace vstupů jsou určeny na základě možného rizika.

Založené na scénářovém testování

- Testované kombinace vstupů vyplývají ze zajímavých scénářů použití produktu.

Pozorování

- Každá jedna vstupní proměnná může sama o sobě mít mnoho možných vstupních hodnot.
- V případě testování kombinací vstupních proměnných, nelze testovat celou doménu každé proměnné.
- Domény hodnot jednotlivých proměnných se typicky předzpracují s využitím technik doménového testování.

Příklad volby reprezentantů v rámci jedné domény

- PM – příliš malá hodnota, NM – nejmenší platná hodnota
- NV – největší platná hodnota, PV – příliš velká hodnota

Pro N proměnných, je prostor možných testů

- $\{PM, NM, NV, PV\} \times \dots \times \{PM, NM, NV, PV\}$

Slabé testování, verze 1

- Cílem je vybrat takovou množinu testů, aby každá proměnná byla alespoň jednou testována na každou hodnotu ze své domény.

Možné testy

	V1	V2	V3
Test1	NM	NM	NM
Test2	NV	NV	NV
Test3	PM	PM	PM
Test4	PV	PV	PV

Pozorování

- Jaký je smysl testů 3 a 4?
- Je možné, že testy 3 a 4 vůbec nelze provést.

Slabé testování, verze 2

- Cílem je vybrat takovou množinu testů, aby každá proměnná byla alespoň jednou testována na každou hodnotu ze své domény, tak aby žádný test neobsahoval neplatné vstupy u víc jak jedné proměnné.

	V1	V2	V3
Test1	NM	NM	NM
Test2	NV	NV	NV
Test3	PM	NM	NV
Test4	NV	PM	NV
Test5	NM	NV	PM
Test6	PV	NV	NM
Test7	NM	PV	NM
Test8	NV	NM	PV

Slabé testování, verze 3

- Cílem je vybrat takovou množinu testů, aby každá proměnná byla alespoň jednou testována na každou platnou hodnotu ze své domény.
- Obecně označována jako technika “All singles”

Možné testy

	V1	V2	V3
Test1	NM	NM	NM
Test2	NV	NV	NV

Silné testování, verze 1

- Cílem je otestovat všechny možné kombinace vstupů.

Možné testy

- $4 \times 4 \times 4 = 64$ možností

Silné testování, verze 2

- Cílem je otestovat všechny možné kombinace platných vstupů.

Možné testy

- $2 \times 2 \times 2 = 8$ možností

Silné testování, verze 3

- Cílem je otestovat všechny N-tice, tj. zvolit takovou sadu testů, aby každá podmnožina vstupních proměnných o N prvcích byla silně testována na všechny (platné) vstupy.
- Obecné označení: “all N-tuples”, “**all-pairs**”, “all triples”.

Možné testy pro N=2 musí pokrýt následující kombinace

V1	V2
NM	NM
NV	NM
NM	NV
NV	NV

V1	V3
NM	NM
NV	NM
NM	NV
NV	NV

V2	V3
NM	NM
NV	NM
NM	NV
NV	NV

	V1	V2	V3
Test1	NM	NM	NM
Test2	NV	NM	NV
Test3	NM	NV	NV
Test4	NV	NV	NM

Tvorba tabulky pokrývající všechny dvojice

- Pozorované proměnné V1,V2,V3,V4,V5 a V6
- V1:A,B,C V2:D,E V3:F,G V4:H,I V5:J,K V6:L,M

V1	V2	V3	V4	V5	V6
A	D				
A	E				
B	D				
B	E				
C	D				
C	E				

- 6 testů

Tvorba tabulky pokrývající všechny dvojice

- Pozorované proměnné V1,V2,V3,V4,V5 a V6
- V1:A,B,C V2:D,E V3:F,G V4:H,I V5:J,K V6:L,M

V1	V2	V3	V4	V5	V6
A	D	F			
A	E	G			
B	D	G			
B	E	F			
C	D	G			
C	E	F			

- 6 testů

Tvorba tabulky pokrývající všechny dvojice

- Pozorované proměnné V1,V2,V3,V4,V5 a V6
- V1:A,B,C V2:D,E V3:F,G V4:H,I V5:J,K V6:L,M

V1	V2	V3	V4	V5	V6
A	D	F	H		
A	E	G	I		
B	D	G	I		
B	E	F	H		
C	D	G	H		
C	E	F	I		

- 6 testů

Tvorba tabulky pokrývající všechny dvojice

- Pozorované proměnné V1,V2,V3,V4,V5 a V6
- V1:A,B,C V2:D,E V3:F,G V4:H,I V5:J,K V6:L,M

V1	V2	V3	V4	V5	V6
A	D	F	H	J	
A	E	G	I	K	
B	D	G	I	K	
B	E	F	H	J	
C	D	G	H	J	
C	E	F	I	K	

- 6 testů

Tvorba tabulky pokrývající všechny dvojice

- Pozorované proměnné V1,V2,V3,V4,V5 a V6
- V1:A,B,C V2:D,E V3:F,G V4:H,I V5:J,K V6:L,M

V1	V2	V3	V4	V5	V6
A	D	F	H	J	
A	E	G	I	K	
B	D	G	I	J	
B	E	F	H	K	
C	D	G	H	K	
C	E	F	I	J	

- 6 testů

Tvorba tabulky pokrývající všechny dvojice

- Pozorované proměnné V1,V2,V3,V4,V5 a V6
- V1:A,B,C V2:D,E V3:F,G V4:H,I V5:J,K V6:L,M

V1	V2	V3	V4	V5	V6
A	D	F	H	J	L
A	E	G	I	K	M
B	D	G	I	J	L
B	E	F	H	K	M
C	D	G	H	K	M
C	E	F	I	J	L

V1	V2	V3	V4	V5	V6
A	D	F	H	J	L
A	E	G	I	K	M
B	D	G	I	J	M
B	E	F	H	K	L
C	D	G	H	K	L
C	E	F	I	J	M

Tvorba tabulky pokrývající všechny dvojice

- Pozorované proměnné V1,V2,V3,V4,V5 a V6
- V1:A,B,C V2:D,E V3:F,G V4:H,I V5:J,K V6:L,M

V1	V2	V3	V4	V5	V6
A	D	F	H	J	L
A	E	G	I	K	M
			H		M
B	D	G	I	J	M
B	E	F	H	K	L
			I		L
C	D	G	H	K	L
C	E	F	I	J	M

- 8 testů místo původních $3 \times 2 \times 2 \times 2 \times 2 = 96$ testů

Tabulka testů pokrývající jednotlivé případy (all singles)

- Pozorované proměnné V1,V2,V3,V4,V5 a V6
- V1:A,B,C V2:D,E V3:F,G V4:H,I V5:J,K V6:L,M

V1	V2	V3	V4	V5	V6
A	D	F	H	J	L
B	E	G	I	K	M
C					

Pozorování

- Pokud se doména jedné proměnné rozpadá na vícero tříd rozkladu, všechny třídy vystupují jako možné hodnoty v kombinatorickém testování.

Závislé proměnné

- Některé kombinace hodnot jsou neplatné, přestože hodnoty vyskytující se v neplatné kombinaci jsou platné v jiných kombinacích.
- Viz příklad s počty dnů v jednotlivých měsících.

Praxe

- Vypočítat all-pairs může být v praxi natolik komplikované, že se od all-pairs kombinatorického testování sklouzne k all-singles technice a náhodným kombinacím.
- Praktické řešení je používat all-singles plus kombinace, které dříve způsobily chyby, nebo jsou něčím zajímavé.

All-pairs kombinatorické testování

- Významně redukuje prostor možných testů.
- Nástavba doménového testování.
- Vhodná pro verifikaci kombinací **nezávislých** proměnných.

Použití all-pairs na závislých proměnných

- Neplatnou hodnotu v dané kombinaci je možné zaměnit jinou platnou hodnotou

Náhodné kombinace

- Po zpracování jednotlivých domén vstupních proměnných bývá testování náhodných kombinací možných vstupů stejně efektivní jako testování metodou all-pairs.
- Získání náhodných kombinací je výrazně snazší, než výpočet all-pairs tabulek.

Scénářové testování

Princip techniky

- Testovat produkt pomocí fiktivních příběhů použití.

Základní cíl

- Otestovat produkt způsobem, který připomíná/napodobuje skutečné použití produktu.

Motivační

- Udává jak, ale také proč, je produkt takto používán.
- Odpovědná osoba by měla trvat na nápravě chyby, pokud se uvedeným scénářem nějaká chyba projeví.

Reálný

- Scénář by měl být věrohodný, tj. nevyprávět příběh, který by se mohl stát, ale příběh, který se pravděpodobně stane.

Netriviální

- Scénář by měl zahrnovat komplexní použití programu v komplexním prostředí na netriviálních datech.

Čitelný

- **Výsledky scénářového testu by měly být čitelné a zřejmé, tj. neměly by být skryty v komplexním výstupu programu.**

- Vhodný způsob zaučení nových testerů a seznámení se s produktem jako takovým.
- Způsob testování dlouhodobého používání produktu a používání produktu expertním uživatelem.
- Jedna z metod zvýšení motivace pro odstranění chyby.
- Způsob testování kvalit produktu vůči zdokumentovaným požadavkům na produkt.

Pozorování

- Tvorba scénářových testů je podobná procesu analýzy požadavků a tvorby specifikace.
- Scénářové testování vynáší na povrch “pod koberec zametená” nevyřešená rozhodnutí ohledně požadavků na produkt a specifikace.

Nedostatky požadavků a specifikace

- Snaha testera je poukázat v jakém kontextu se problém projeví a jaké to může mít následky.
- Je důležité udržet si roli testera.
- Tester nehledá řešení problému.
- Tester nedělá kompromisy v návrhu produktu (nepromíjí), naopak snaží se prokázat dopady možných kompromisů.

Scénář pro test programu pro výpočet daně z příjmů

Na začátku loňského roku byl Pavel rozvedený a z předchozího manželství měl 2 dospělé děti, z nichž jedno v březnu zemřelo ve věku 19 let při autonehodě, druhé dosáhlo věku 26 let v srpnu. Pavel celý rok pracoval na částečný (2/3) úvazek jako dělník v zavedené firmě s hrubým ročním příjmem 360.000Kč. Ze začátku roku se seznámil s pohlednou Helenou, od února žili ve společné domácnosti, v březnu se vzali a v říjnu se jim narodila dcera. Aby mohl Pavel novou rodinu uživit, začal v červnu podnikat. Na rozjezd podnikání si vzal půjčku ze Stavebního spoření, kterou však splatil v prosinci, když konečně skončilo vleklé dědické řízení, ve kterém Pavel zdědil 421.456 Kč. Na úrocích z půjčky Pavel zaplatil 37.210 Kč. Na konci roku byla finanční bilance Pavla jako podnikatele záporná, konkrétně -13.000 Kč. Vzhledem k rizikovému těhotenství Helena pracovala pouze první dva měsíce v roce a její příjmy nepřesáhly 25.000 Kč. Pavel s Helenou uplatňují společné zdanění manželů.

- Modelujte možné životní cykly objektů v systému (vznik, interakce, modifikace, sloučení, rozpad, zánik a další).
- Vyjmenujte druhy uživatelů a pro každý druh uživatele analyzujte jeho motivaci a cíl jeho práce.
- Zvažte jak lze systém napadnout, poškodit, či zneužít.
- Jaké události v systému mohou nastat, a jak jsou systémem zpracovány.
- Analyzujte jednotlivé transakce, které se uživatelé snaží provést a 1) úspěšně dokončit, 2) předčasně přerušit.
- Použijte scénáře, které vedly k chybě v předchozí verzi systému.
- ...

Nevhodné pro testování v rané fázi vývoje

- Scénář by měl být komplexní, dosud neimplementované funkce brání provedení testu.
- Jednotlivé funkce má smysl testovat nejprve samostatně.

Nevhodné pro metriku pokrytí kódu programu

- Pokrytí kódu scénářem je náročné a vede scénář do netypických situací.

Znovupoužití scénářů může být neefektivní

- Příprava dobrého zcela nového scénáře je náročná.
- Tvorba nových scénářů modifikací existujících nemusí odhalit zcela jiné typy chyb.
- Na odhalené chyby lze připravit jiné typy testů a ty například použít v regresním testování.