Szoftverellenőrzési technikák

Ipari tapasztalatok
Missió kritikus és Biztonságorientált rendszerek fejlesztése során

Lantos Péter

Tartalom

- Bemutatkozás
- SW technológia
  - Biztonság szempontból
  - Hol a határ
  - Projekt szempontok
  - V&V technikák
  - Eszközök
- Pszichológia, management

Ipari tapasztalatok

Budapesti Értéktőzsde

- Kvázi realtime, elosztott rendszer
- Kereskedési idő alatt, memória adatbázis, magas rendelkezésre-állás (max. 30perc/év kiesés, majdnem 99,99%).
  - Kliens-szerver architektúra
  - Meleg tartalékolt szerver - alkalmazás is
  - Vastag kliensek
  - Saját tranzakciós log
  - Napközben folyton növekvő adathalmaz
  - Előre megszabott táblaméretek
  - Rendszer adatbázis karbantartás este – paraméterezhető felhasználók, papírok, szabályok, algoritmusok

Ipari tapasztalatok 2

Prolan Zrt.

- Ipari folyamatirányítás = SCADA + RTU
- Villamos energia elosztás, Vasúti irányító Központok
  - 7*24 órás működés
  - 99,9% Rendelkezésre-állás

Meghibásodási osztály	Rendszer-meghibásodási mód	A működésre gyakorolt hatás	MTBF [év]
döntő jelentőségű	teljes meghibásodás	a biztosítóberendezés kezelése lehetetlen	25
nagy jelentőségű	kritikus funkcionális hiba	vágányutak beállítása nem lehetséges, ezért a biztosítóberendezés kezelése gyakorlatilag lehetetlen	9
közepes jelentőségű	nem kritikus funkcionális hiba	különleges kezelések kiadása nem lehetséges	3
csekély jelentőségű	nem elhanyagolható funkcionális hiba	több funkció csoportos kiesése	3
elhanyagolható	elhanyagolható funkcionális hiba	egy funkció kiesése	1

- Vasút: biztonság kritikus: THR:10-9 -> SIL 4 (objektum és funkció tévesztés)

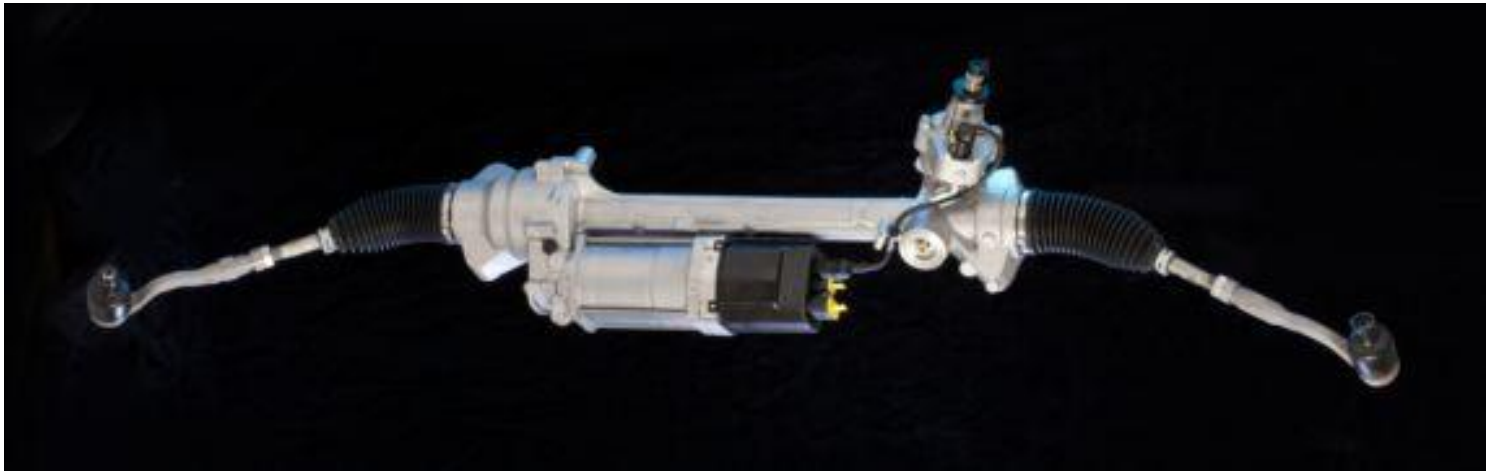
Ipari tapasztalatok 2

- Elosztott rendszer – több központra kapcsolódó kliensek
- Adatok
  - Pillanatnyi (helyes jelenlegi állapot), közbenső nem annyira fontos
  - Naplók, történet 1-naptól 1 hónapig. (Mozi funkció, menetdiagram)
  - Konfigurálhatóság -> magas paraméterehetőség (Alap sw. minden állomásra ugyanaz…)
  - Megoldás: operációs rendszer és X platform felett minden saját, gyors, adatbázis: memória, fájlok saját szervezés.

Ipari tapasztalatok 3

ThyssenKrupp Presta

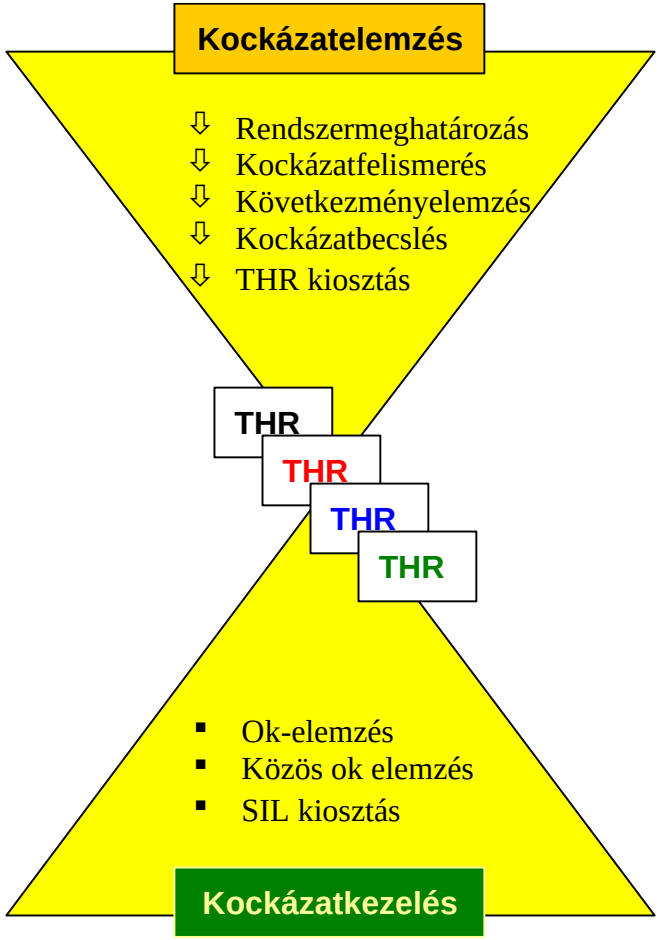
- Elektromechanikus rendszer – kormánymű beágyazott számítógép által vezérelt rásegítés

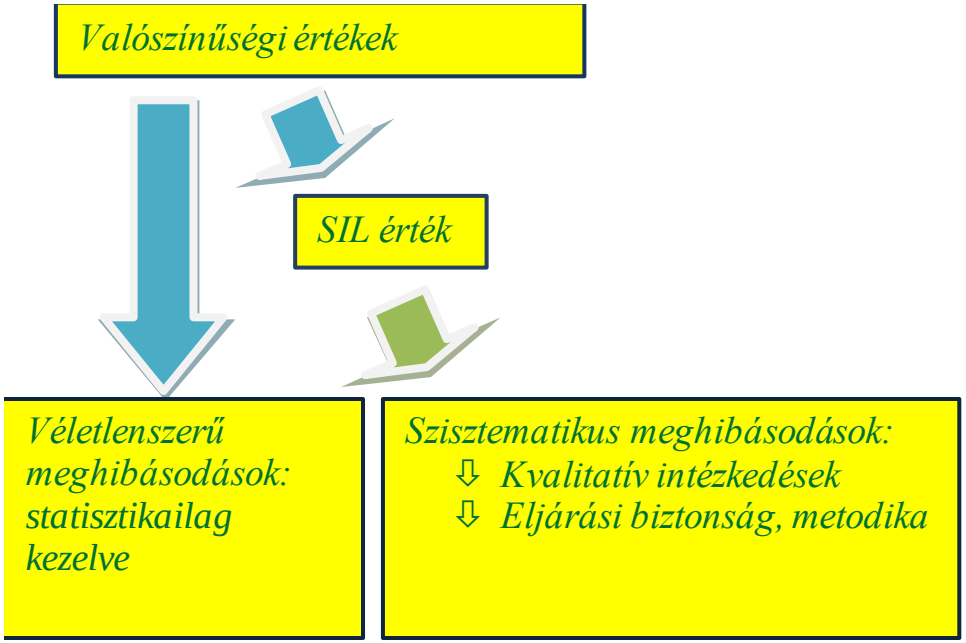


- Biztonság kritikus: ASIL D szabályzás
  - Fix tasking; erőforrás előre kiosztva; de paraméterezés itt is igény
- Nagysorozatú gyártás!!!

Biztonsági (Safety) szabványok

- EN 50126, 50129, 50128; ISO 26262
- Rendszerek tervezése -> kiindulás magas szinten.



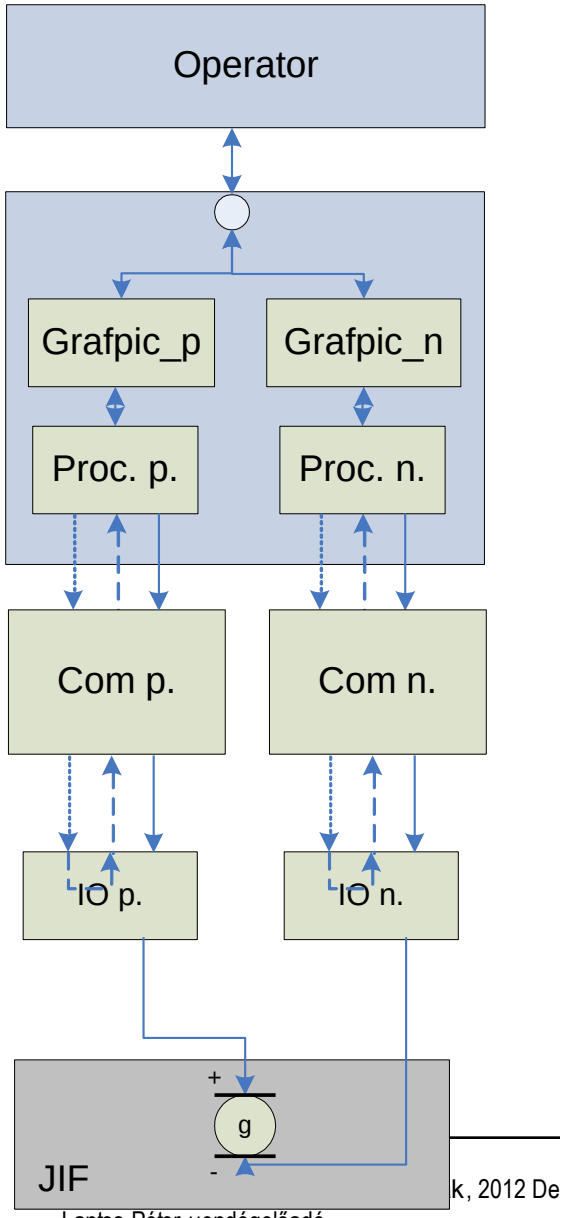


Biztonsági követelmény

Function class	Fail	Faulty operation (object selection error, function selection error, false status shown by the indicator)	Unintended issue of a command
Normal control mode and confirmed special control mode	SIL 0	SIL 2, THR 10-7	SIL 2, THR 10-7
Special critical control mode	SIL 0	SIL 4, THR 10-9	SIL 4, THR 10-9
Indicators	SIL 2, THR 10-7	SIL 2, THR 10-7	NA
Other	SIL 0	SIL 0	SIL 0

Biztonsági architektúra

Biztonsági architektúra



- 2oo2 = 2-channel:
  - JIF „g” relay + control check of RTU = fail-safe comparator
  - JIF checked by MMI
  - Alternated picture from both channel
  - Differences between channel detected by MMI PC and operator
  - Diversity of addressing
- Duplicated for RAM

Biztonsági (Safety) szabványok

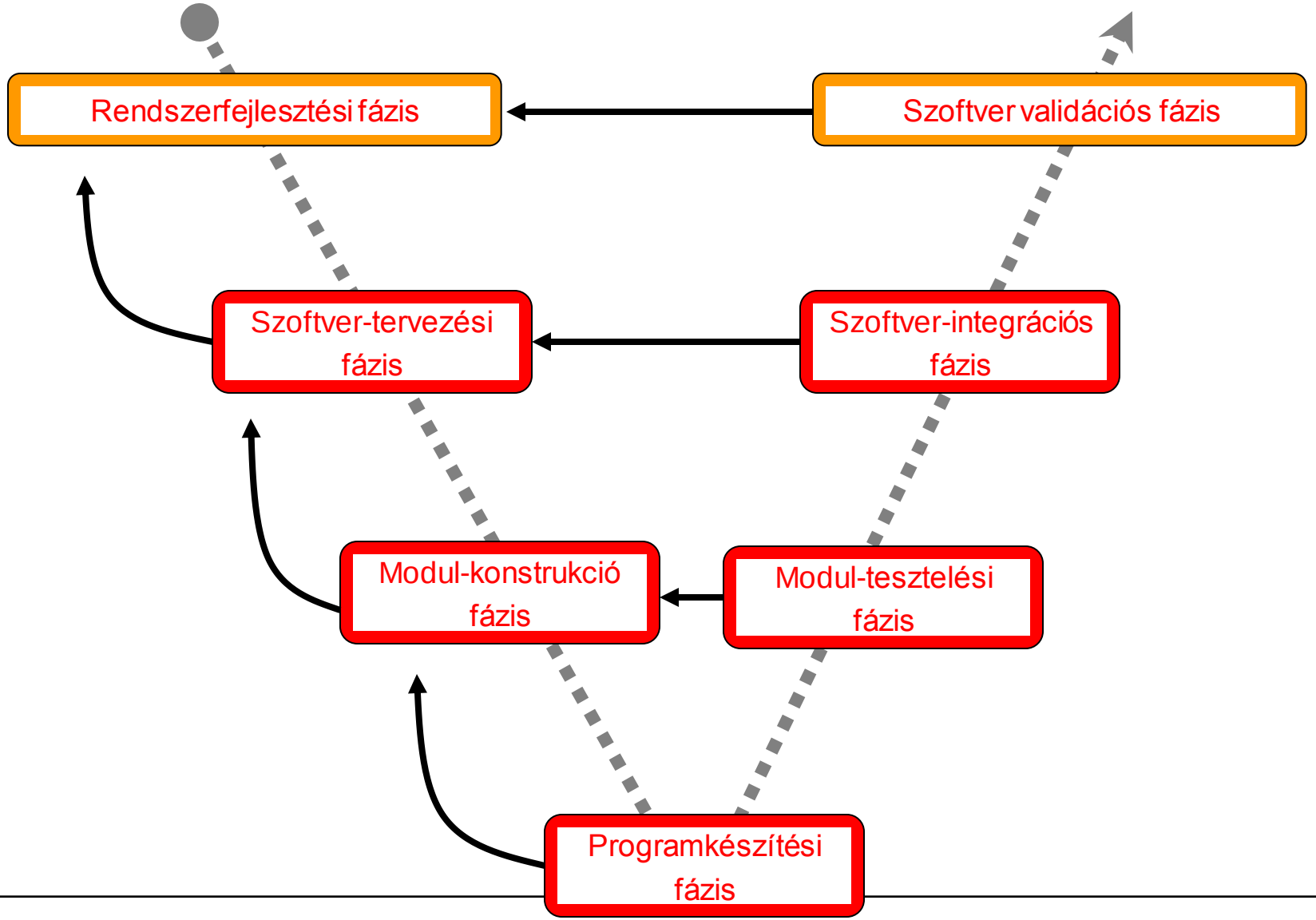
Szoftver biztonság (rendszer logika is…)

- Jó (hibátlan ?!) szoftver – szisztematikus hibák elleni védekezés
- SW képes működni?
  - Számítógép modellje…
  - HW – SW együtt működés
    - Robosztusság -> hibabeültetés vizsgálat, SW FMEA
    - Common Cause Failure (CCF) -> hibaterjedés vizsgálat, CCF elemzés SW FMEA részeként

SW Biztonság

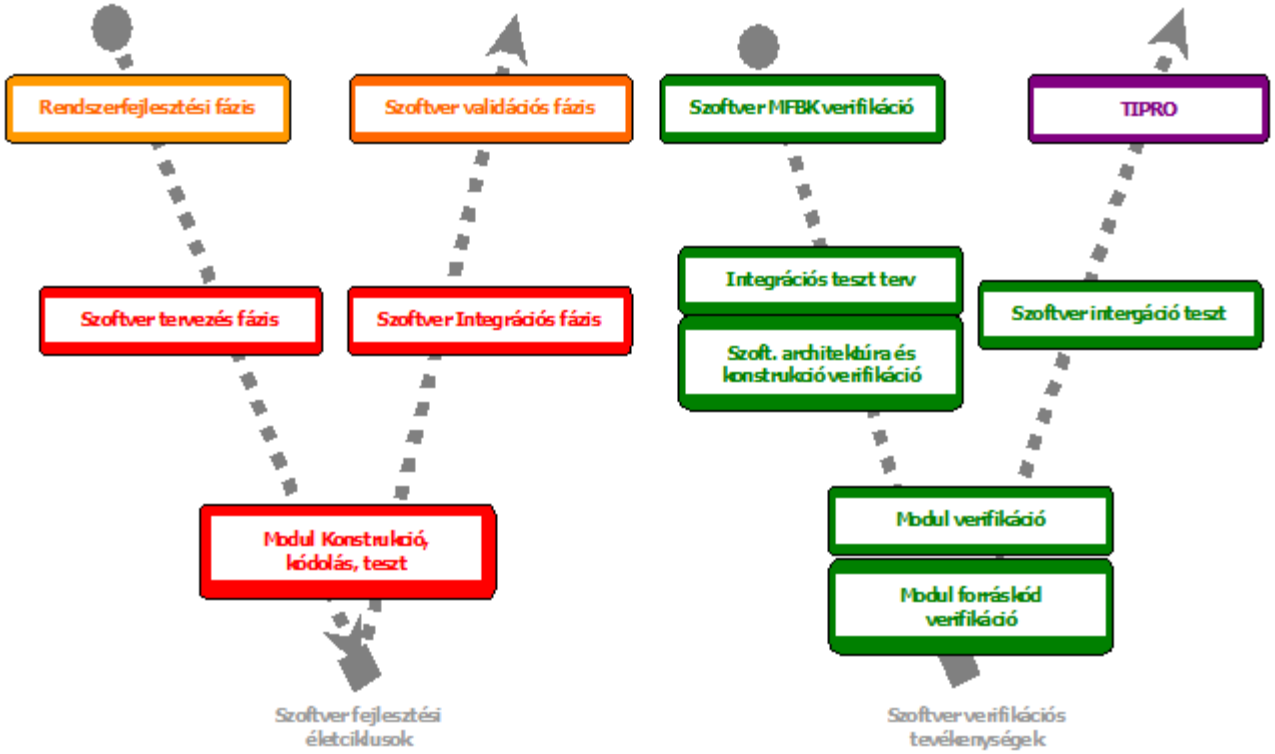
- A hibátlan(?!) Szoftver
  - Végtermék minősége mérhető?
  - Eljárás alapú biztonság
  - „Software Engineering” alkalmazása
    - Bevált, széles körben elterjedt módszerek kiválasztása
    - Termékre gyakorolt hatásukra vannak tapasztalatok
  - Fejlesztési eljárás „V”
    - Generikus termék és alkalmazás elválasztása
    - Többszöri alkalmazás - ciklusok

Egy „V”


Rendszerfejlesztési fázis
Szoftver validációs fázis
Szoftver-tervezési fázis
Szoftver-integrációs fázis
Modul-konstrukció fázis
Modul-tesztelési fázis
Programkészítési fázis

Verifikációs és validációs terv

- Alap V&V stratégia megalkotása



Verifikációs és validációs terv

MÓDSZER/INTÉZKEDÉS	Ref.	SW SIL0	SW SIL1	SW SIL2	SW SIL3	SW SIL4
1. Formális bizonyítás	B.31	-	R	R	HR	HR
2. Valószínűségi tesztelés	B.47	-	R	R	HR	HR
3. Statikus elemzés	D.8	-	HR	HR	HR	HR
4. Dinamikus elemzés és tesztelés	D.2	-	HR	HR	HR	HR
5. Mértékek	B.42	-	R	R	HR	HR
6. Követhetőségi mátrix	B.69	-	R	R	HR	HR
7. Szoftverhiba-hatáselemzés	B.26	-	R	R	HR	HR

Követelmények

1. A 3-as és 4-es szoftver biztonságintegritási szintek esetén az alábbi módszerkombinációk alkalmazhatóak:
   a) 1 és 4
   b) 3 és 4
   c) 4,6 és 7
2. Az 1-es és 2-es szoftver biztonságintegritási szintek esetén az alkalmazott módszer az 1-es vagy a 4-es.

Verifikációs és validációs terv

MÓDSZER/INTÉZKEDÉS	Ref.	SW SIL0	SW SIL1	SW SIL2	SW SIL3	SW SIL4
1. Tesztesetek megvalósítása a határérték-elemzésből	B.4	-	HR	HR	HR	HR
2. Tesztesetek megvalósítása hibabecslésből	B.21	R	R	R	R	R
3. Tesztesetek végrehajtása hibakeresésből	B.22	-	R	R	R	R
4. Teljesítmény-modellezés	B.45	-	R	R	HR	HR
5. Ekvivalencia-osztályok és bemeneti adatfelosztás szerinti tesztelés	B.19	-	R	R	HR	HR
6. Struktúrált alapú tesztelés	B.58	-	R	R	HR	HR

Követelmények

A tesztesetekre vonatkozó analízis alrendszer szinten valósul meg; és a specifikáción és/vagy a specifikáción és a kódon alapszik.

A szoftver-biztonságintegritási szintnek megfelelő módszerkészletet kell kiválasztani.

ISO 26262 – SW Integration Testing

Methods for SW integration testing	ASIL			
	A	B	C	D
Requirement based testing	++	++	++	++
External Interface test	++	++	++	++
Fault injection test	+	+	++	++
Resource usage test	+	+	+	++
Back-to-back test between model and code	+	+	++	++
Methods for deriving test cases for SW integration testing				
Analysis of requirements	++	++	++	++
Generation and analysis of equivalence classes	+	++	++	++
Analysis of boundary values	+	++	++	++
Error guessing	+	+	+	+
Structural coverage metrics on the software unit level				
Function coverage *(execute at least once every SW function)*	+	+	++	++
Call coverage *(execute at least one every SW function call)*	+	+	++	++

Egy lehetséges választás

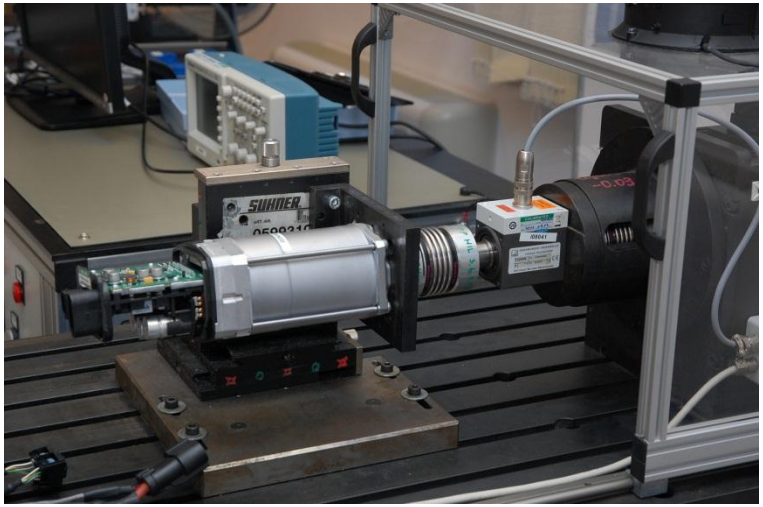
Xgram verifikációs módszer	MSZ EN 50128 által javasolt módszer
Modulteszt unit teszt része	B.58 Struktúrált alapú tesztelés, (B.21 Tesztesetek megvalósítása hibabecslésből), B.19 Ekvivalencia-osztályok és bemeneti adatfelosztás szerinti tesztelés
Modulteszt funkcionális teszt	B.4 Tesztesetek megvalósítása a határérték-elemzésből, (B.21 Tesztesetek megvalósítása hibabecslésből), B.19 Ekvivalencia-osztályok és bemeneti adatfelosztás szerinti tesztelés
Szoftverintegráció teszt	B.4 Tesztesetek megvalósítása a határérték-elemzésből, B.19 Ekvivalencia-osztályok és bemeneti adatfelosztás szerinti tesztelés, B.21 Tesztesetek megvalósítása hibabecslésből, B.45 Teljesítmény-modellezés, B.48 Folyamatszimmuláció, B.3 Lavina- / stressztesztelés
Adatteszt	B.4 Tesztesetek megvalósítása a határérték-elemzésből, B.21 Tesztesetek megvalósítása hibabecslésből, B.19 Ekvivalencia-osztályok és bemeneti adatfelosztás szerinti tesztelés
Gyári elfogadási próba	B.4 Tesztesetek megvalósítása a határérték-elemzésből, B.19 Ekvivalencia-osztályok és bemeneti adatfelosztás szerinti tesztelés, B.21 Tesztesetek megvalósítása hibabecslésből, B.45 Teljesítmény-modellezés, B.48 Folyamatszimmuláció, B.3 Lavina- / stressztesztelés

Hol a határ?

- Adott szintű szoftver követelménnyel szemben
- Mit is jelent a tesztelés – statikus/dinamikus
  - Dinamikus - milyen elemei vannak?
    - SW valamilyen futtatható formában
    - Működésre bírni
    - Elvárt eredményt vizsgálni
    - Automatizált/ kézi
    - Környezet:
      - min fut: cél HW, vagy PC eszköz függően (lefedettség mérés, teszt menedzsment...)
      - Környezet szimuláció hitelesen!!!
      - Kompromisszum kell – változtatás nélkül nincs teszt!
      - Rendszer szinten túl sok minden elfedve illetve veszélyes: több teszt eggyütese kell – lásd Csernobil

Hol a határ?

- De rendszert szállítunk
- Sok-sok rendszer követelmény egy az egyben SW követelmény
- Paraméterehetőség -> minden eset letesztelhető?



Hol a határ?

- Mikor mondjuk hogy letesztelve?
- Felelőségi kérdés
- Lefedettség -> szabványok
- Teszt módszerek kiválasztása -> szabványok
- Mérnöki gyakorlat...
  - Eddigi hibák
  - Módosítás utáni tesz -> Delta teszt
  - Regresszió
    - ki mit ért alatta, hiba terjedés – életbeli példák...
    - Speciális teszt lehetőség –> elvárt eredmény az előző (lehet random input, korábbi felvett forgalom, életbeli forgalom...)
- Mérték statikus analízisre, review-ra -> szabály lista, ellenőrző lista → ló túlsó oldala

Projektek szempontok

- Egy projekt élete
  - ötlettől az első telepítésig / gyártásig
  - További verziók
  - Nem zöldmezős – előzmény projektek
  - Kapcsolódó projektek – pl. Szolgáltatási réteg használata
- SW újrahasznosítás
  - Modularizáció
    - „V” nem egy hanem több száz!
    - Különböző életciklusok – évtizedes múlt!
  - Elágazások (branches) – merge
- Teszt minden változtatás után → tesztautomatizálás de drága, kézi esetén teszt vakság

V&V technikák

- Milyen V&V technikákat tanultak?
- Funkcionális tesztelés spec. alapján
  - Testeset tervezés (Equivalence partitioning, boundary value analysis, decision tables, state-based testing)
- Funkcionális tesztelés struktúra alapján
  - Control flow, statement, desicion, loop, condition, Decision/condition, MC/DC → külön szakma lett!
  - Stub kell, lefedettség mérés, teszt management → erős eszköztámogatás kell.
- Hiba katalógus, tapasztalat alapú teszteset bővítés
  - Milyen hibákra is készülünk fel?
  - Szabványok termék független listát adnak (lást fent), de lehet architektúra függő kiegészítés – ma ez még nagy kérdés.

V&V technikák

- Statikus és dinamikus elemzések (Eszközök, instrumentálók – ehhez is vannak hibatípus listák, pl. Memória kezelés, változók életciklusa...)
- További tesztek:
  - Alkalmazhatóság, pontosság, security..
  - Rendelkezésre állás, hibaállóság, recoverability...
  - Teljesítmény/teljesítés tesztek (erőforrás, idő) → nagyon függ az program feladatától, emlékezetétől. Egy fajta dinamikus elemzés, itt is fontos az eszköz. Grafikonok készítése a kimenet.
- Review-k
  - Ellenőrző lista alapján
  - Team munka

Egy példa - Milyen módszer a legerősebb

- #include <stdio.h>

```
int main(){
   int i;
   float f;
   f=4;
   for(i=0;i<100000000;i++){
     f=f/2;
     f=f*f;
   }

   return 0;
}
```

Ha a program körülbelül 1 másodperc alatt fut le, mennyi idő alatt fut le, ha a f=4 sort f=5 -re cseréljük?
a) Intel vagy AMD processzoron?
b) körülbelül szintén 1 másodperc alatt
c) lényegesen lassabban
d) lényegesen gyorsabban

Másik példa

```
printf("%f\n", 5/2 );
```

- Mit ír ki a program?

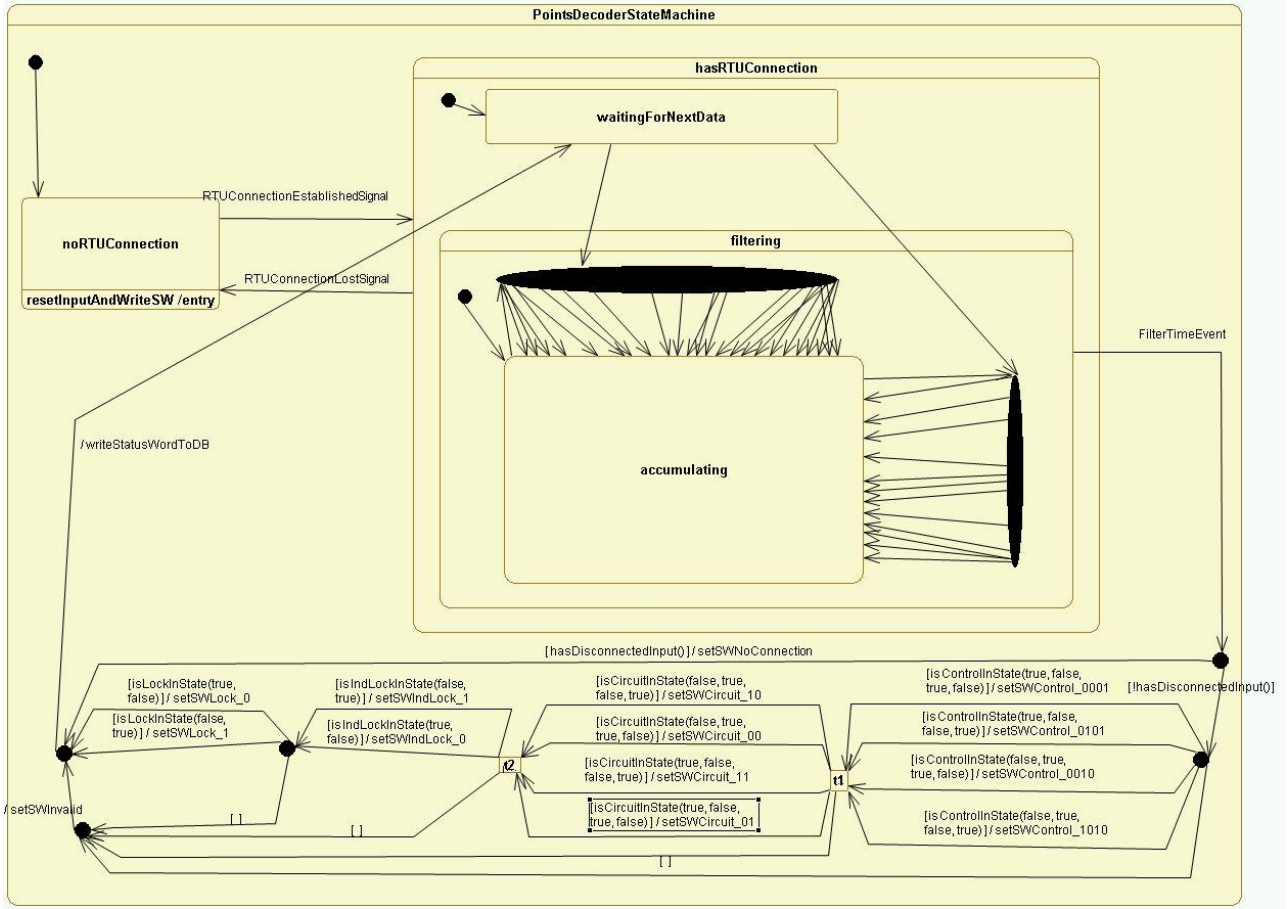
  a) 0-t
  b) 2.5 -öt
  c) 2 -t
  d) nem fordul le
  e) minden futáskor azonos hibás értéket
  f) minden futáskor más hibás értéket

- Egymásra épülő technikák:
  - Problémás pontok jelentése
  - Hibasűrüsődés

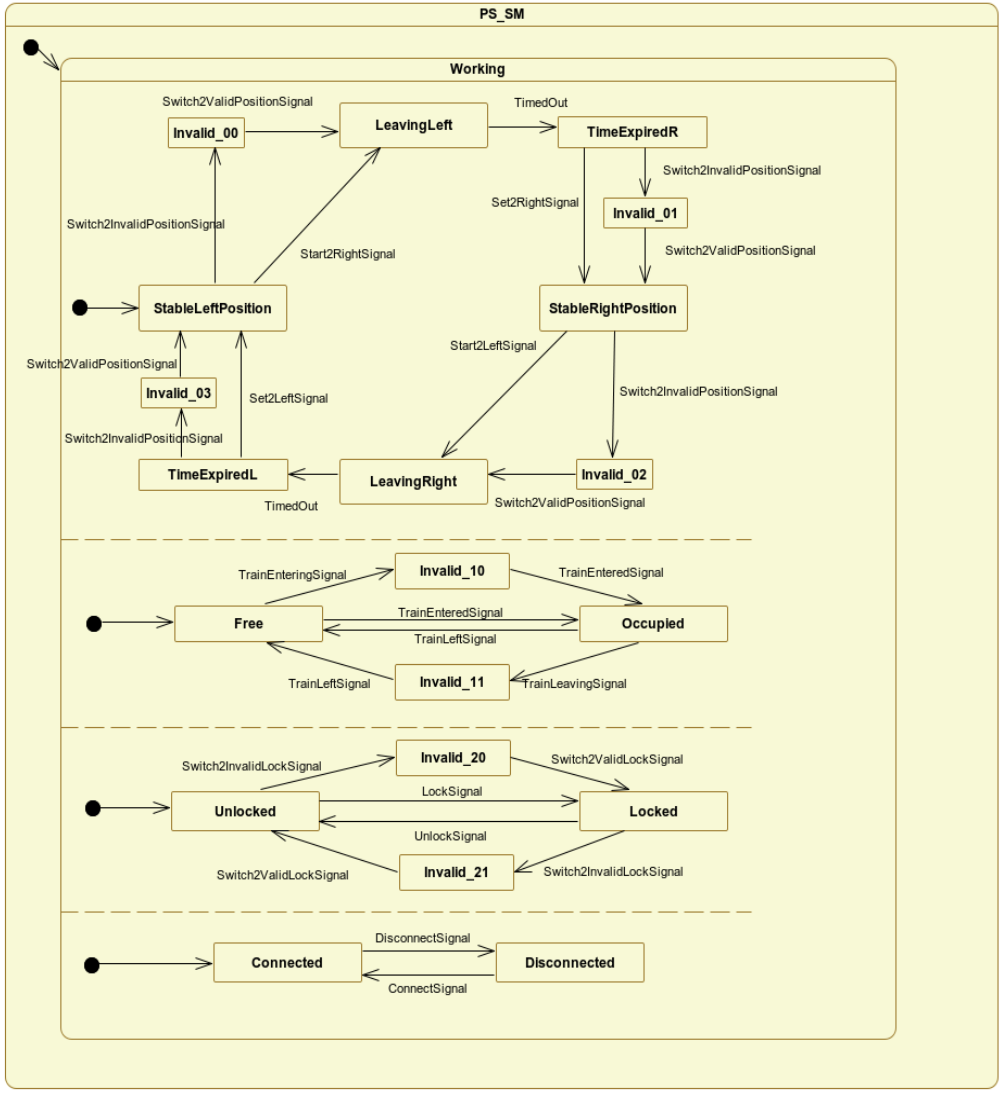
Eszközök

- Teszt automatizálás
  - Futtatás
  - Menedzsment
  - Archiválás
  - Input generátorok, elvárt eredmény ellenőrzök
    - Grafika (GUI teszt) külön kihívás
  - Mérések – pl. Lefedettség
  - Elemzők (statikus, dinamikus, teljesítmény)
  - Traceability (kézi, Doors)
  - Teszteset generálás → specifikáció alapon egy kutatási projectben: MOGENTES

MOGENTES


PointsDecoderStateMachine
hasRTUConnection
waitingForNextData
noRTUConnection
resetInputAndWriteSW /entry
RTUConnectionEstablishedSignal
RTUConnectionLostSignal
filtering
accumulating
FilterTimeEvent
/writeStatusWordToDB
[hasDisconnectedInput()] / setSWNoConnection
[isLockInState(true, false)] / setSWLock_0
[isLockInState(false, true)] / setSWLock_1
[isIndLockInState(false, true)] / setSWIndLock_1
[isIndLockInState(true, false)] / setSWIndLock_0
[isCircuitInState(false, true, false, true)] / setSWCircuit_10
[isCircuitInState(false, true, true, false)] / setSWCircuit_00
[isCircuitInState(true, false, false, true)] / setSWCircuit_11
[isCircuitInState(true, false, true, false)] / setSWCircuit_01
[isControlInState(true, false, true, false)] / setSWControl_0001
[isControlInState(true, false, false, true)] / setSWControl_0101
[isControlInState(false, true, true, false)] / setSWControl_0010
[isControlInState(false, true, false, true)] / setSWControl_1010
[!hasDisconnectedInput()]
t2
t1
/setSWInvalid
[ ]
[ ]
[ ]

MOGENTES


PS_SM
Working
Switch2ValidPositionSignal
Invalid_00
LeavingLeft
TimedOut
TimeExpiredR
Switch2InvalidPositionSignal
Invalid_01
Set2RightSignal
Switch2InvalidPositionSignal
Start2RightSignal
Switch2ValidPositionSignal
StableLeftPosition
StableRightPosition
Start2LeftSignal
Switch2ValidPositionSignal
Invalid_03
Set2LeftSignal
Switch2InvalidPositionSignal
Switch2InvalidPositionSignal
TimeExpiredL
LeavingRight
Invalid_02
TimedOut
Switch2ValidPositionSignal
TrainEnteringSignal
Invalid_10
TrainEnteredSignal
Free
TrainEnteredSignal
Occupied
TrainLeftSignal
TrainLeftSignal
Invalid_11
TrainLeavingSignal
Switch2InvalidLockSignal
Invalid_20
Switch2ValidLockSignal
LockSignal
Unlocked
Locked
UnlockSignal
Switch2ValidLockSignal
Invalid_21
Switch2InvalidLockSignal
DisconnectSignal
Connected
Disconnected
ConnectSignal

MOGENTES

Simulator	UPPAAL
Nr. of test suites	2
Nr. of test cases	1287
Steps in a case	[4-17]
Aborted suites	0
Avg. time/TC	~1.6 sec

Objs	UPPAAL	OOAS	Manual
Nr. of test suites	5	2	1
Nr. of test cases	182	429	31
Steps in a case	[1-11]	[2-54]	[1-8]
Aborted suites	2	?	-
Avg. time/TC	~0.6/~2/233 sec	~1.5/86 sec	-

Pszihológia, management

- Egyedül vagy csapatban?
- Emberek használják, ha másként nem közvetve
  - Ergonómia, felhasználó elképzelése az eszközről, rendszerről
    - Pl sütő,
    - Pl automatikák – bizom benne? GPS és komp
- Munkánk nagy része (fele!)
  - Kifelé
    - Megrendelő használat közben tudja meg mit is szeretne
    - Fontos a gyors prototipus
    - Kommunikáció fontossága → legyünk bátrak és beszéljük meg a megrendelővel, ne legyünk misztikussak

Pszichológia, management

- Belül:
  - Együtt működés
    - közös modell világ
    - Szerepek, ki tesztel? Ki review-zik?
    - Fejlesztőkkel megtalálni a hangot
    - Erőforrás hiány, időszorítás
    - Motiválhatóság – V&V esetén ez nehéz. Fejlesztők és tesztelők versenye, konfliktusa
    - Teszt elismertsége, kihívások ismerete
  - Tesztelés és Quality mint menedzsment
    - Stratégiai és részletes tervezés
    - Minőségi szempontok érvényesítése
    - Metrikák
    - Részvétel az irányításban:
      - » Hibák kezelése, hibák életciklusának követése
      - » Konfiguráció kezelés....

Pszichológia, management

- Függetlenség
  - Saját hibám felfedése (saját gondolataim csapdájában)
  - Független nem alkotó ellenőrzés → kompetencia kérdése:
    - Értem a modellt? Kinél is van a hiba?

- Modern technikák
  - Testdriven
  - Team munka: párban, rotációban

Köszönöm a figyelmüket!