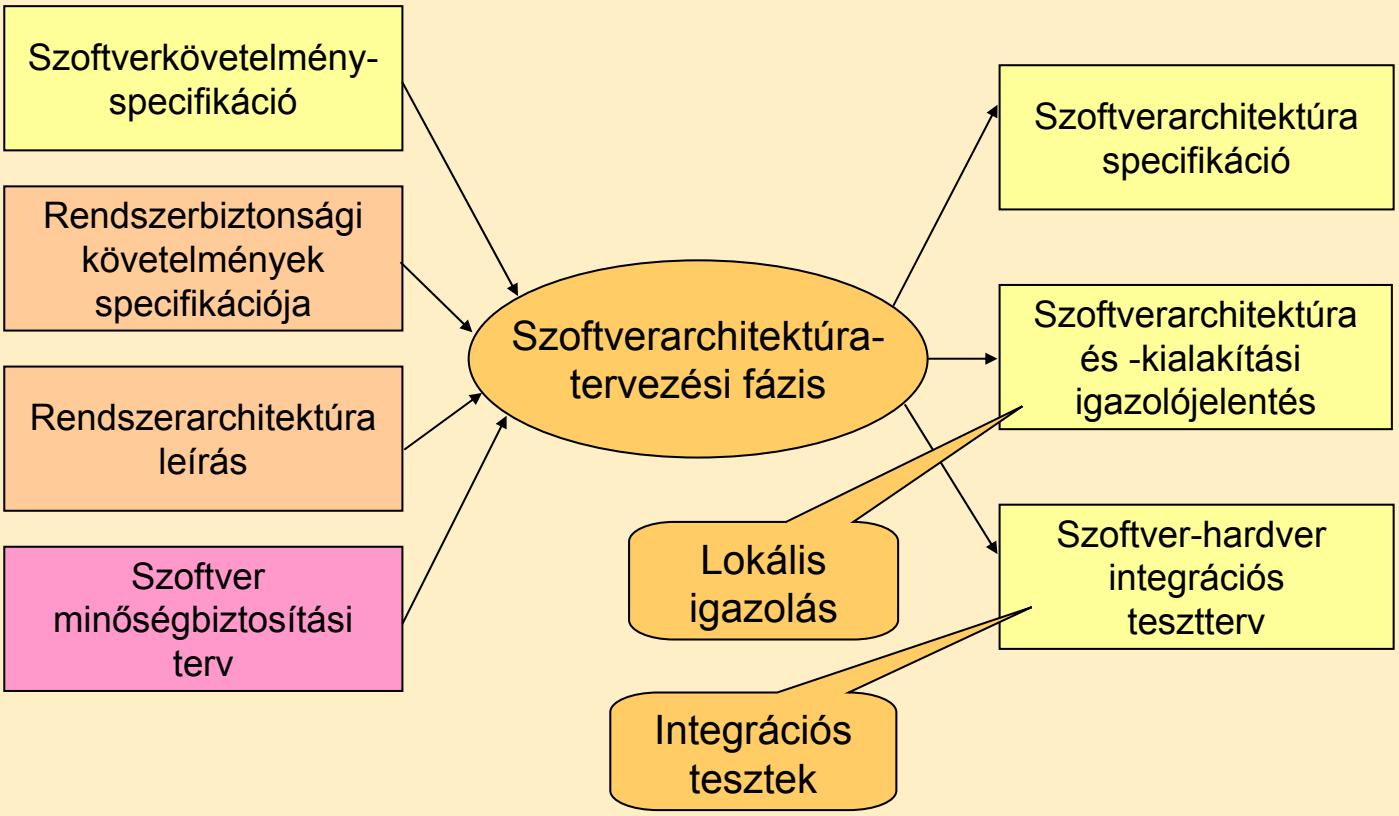
Szoftver architektúra
tervek ellenőrzése

Majzik István
Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék
http://www.mit.bme.hu/~majzik/

Tartalomjegyzék

- A fázis ki- és bemenetei
- Az architektúra tervek elkészítése
  - A komponensek azonosítása
  - Mit határoz meg az architektúra?
- Ellenőrzési feladatok
  - Követelményeknek való megfelelőség, követhetőség
  - Hibahatások vizsgálata
  - Extra-funkcionális követelmények vizsgálata
- Teszt tervezés

Kimenetek és bemenetek



Tartalomjegyzék

- A fázis ki- és bemenetei
- Az architektúra tervek elkészítése
  - A komponensek azonosítása
  - Mit határoz meg az architektúra?
- Ellenőrzési feladatok
  - Követelményeknek való megfelelőség, követhetőség
  - Hibahatások vizsgálata
  - Extra-funkcionális követelmények vizsgálata
- Teszt tervezés

Szoftverarchitektúra terv

- Architektúra: Komponensek + közöttük lévő kapcsolatok
- Hardver-szoftver együttműködés azonosítása
- Szoftverkomponensek (modulok) azonosítása
  - Korábban fejlesztett, érvényesített (validált) elemek: alkalmasság igazolásával használhatók
- (C)OTS komponensek használata
  - SIL 0: előfeltételek nélkül elfogadható
  - SIL 1,2: validációnak tartalmaznia kell
  - SIL 3,4: validációnak tartalmaznia kell +
    + lehetséges meghibásodások elemzése
    + védelmi stratégia és ennek tesztelése
    + hibanaplók és kiértékelésük

- Szoftverkövetelményeknek való megfelelés
- Változások hatásvizsgálata
- SIL megfelelés

- Szoftver modulok SIL szintje:
  - Alapértelmezés: Egyező a rendszer(modulok) SIL szintjével
  - Csökkentés: Van olyan mechanizmus, amely megakadályozza, hogy a szoftver meghibásodása a rendszer nem biztonságos állapotba kerülését okozhassa (a függetlenség bizonyítható).

Hibakezelés: Redundancia

- Hardver redundancia:
  - Azonos komponensek:
    Tranziens és állandósult működés közbeni hibák detektálása és (diagnosztika után) hibakezelés
    - Két komponens: Hibadetektálásra alkalmas, diagnosztika szükséges
    - Kettőnél több komponens (NMR): Hiba maszkolható (szavazás)
- Szoftver redundancia:
  - Eltérő tervezés („diverziter programozás”):
    A szisztematikus szoftver tervezési hibák detektálása és kezelése
    - Aktív redundancia: N-verziós programozás (NVP)
    - Passzív redundancia: Javító blokkok (RB)
    - Adaptív redundancia: Önkonfiguráló programozás (SCOP)
- Információ redundancia:
  - Hibafelismerő illetve hibajavító kódolás
- Idő redundancia:
  - Újrapróbálás: Tranziens hibák esetén lehet hatásos
  - Közvetett idő redundancia más technikák esetén is

Előírt módszerek (50128) - hibakezelés

- SIL 1-től R, SIL 3-tól tipikusan HR technikák
  - Defenzív programozás
  - Hibafelfedés és hibadiagnózis
  - Hibafelismerő kódok
  - Meghibásodásbizonyító programozás
  - Diverziter programozás
    - Eltérő tervezésű modulok
  - Megvalósított esetek tárolása
  - Szoftverhiba-hatáselemzés

  -> Szoftver, információ és idő redundancia

Sokféle kombináció megengedett

Failure assertion

Referencia

Bennmaradó hibák elleni védekezés

- Ellenjavallt technikák (NR)
  - Előre / visszalépő helyreállítás
  - Mesterséges intelligencia módszerek hibajavításra
  - Dinamikus szoftver rekonfiguráció

Mit határoz meg az architektúra?

Elérendő tulajdonság	**Tervezési tér**
Szolgáltatásbiztonság	Hibadetektálás, hibabehatárolás, hibakezelés (redundancia)
Teljesítmény	Erőforrás hozzárendelés, erőforrás menedzsment
Biztonságosság	Veszély csökkentés, veszély kontroll (monitorozás)
Adatbiztonság	Támadás felderítés, helyreállítás
Tesztelhetőség	Vezérelhetőség, megfigyelhetőség
Karbantarthatóság	Elkülönítés (pl. MVC minta)

Tartalomjegyzék

- A fázis ki- és bemenetei
- Az architektúra tervek elkészítése
  - A komponensek azonosítása
  - Mit határoz meg az architektúra?
- Ellenőrzési feladatok
  - Követelményeknek való megfelelőség, követhetőség
  - Hibahatások vizsgálata
  - Extra-funkcionális követelmények vizsgálata
- Teszt tervezés

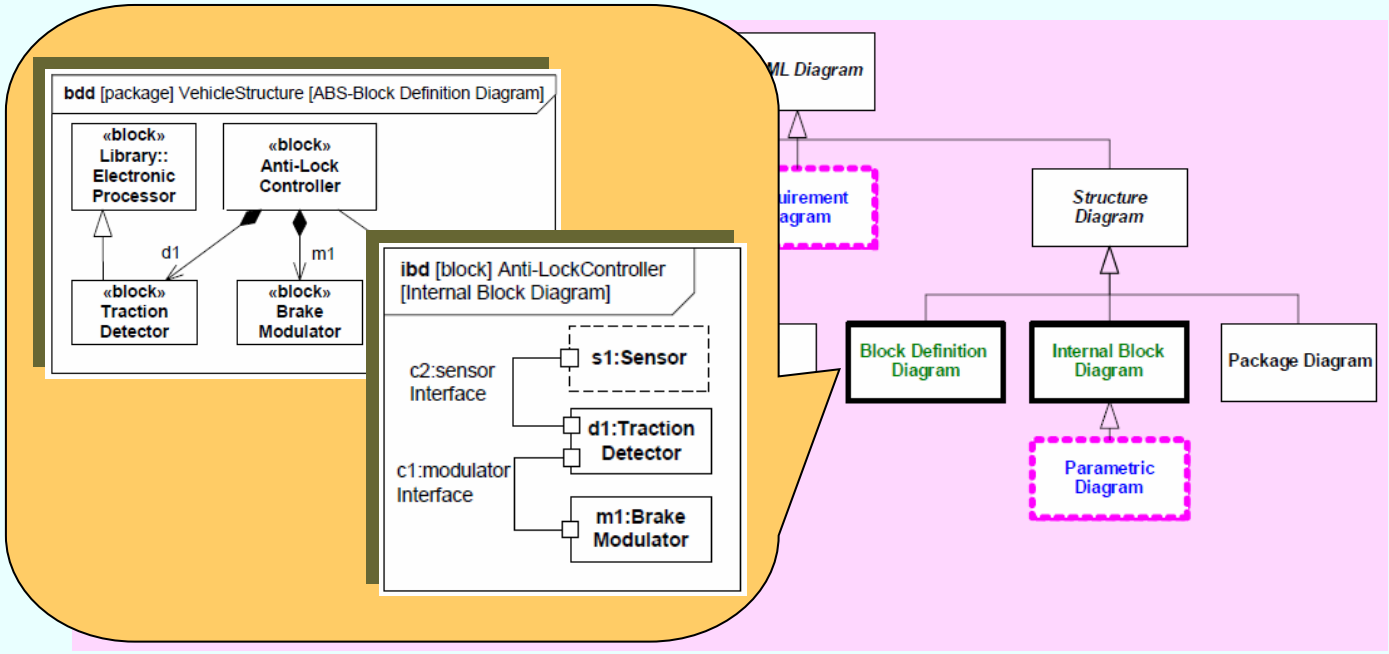
A szoftverarchitektúra igazolása (áttekintés)

Vizsgálati technikák áttekintése:

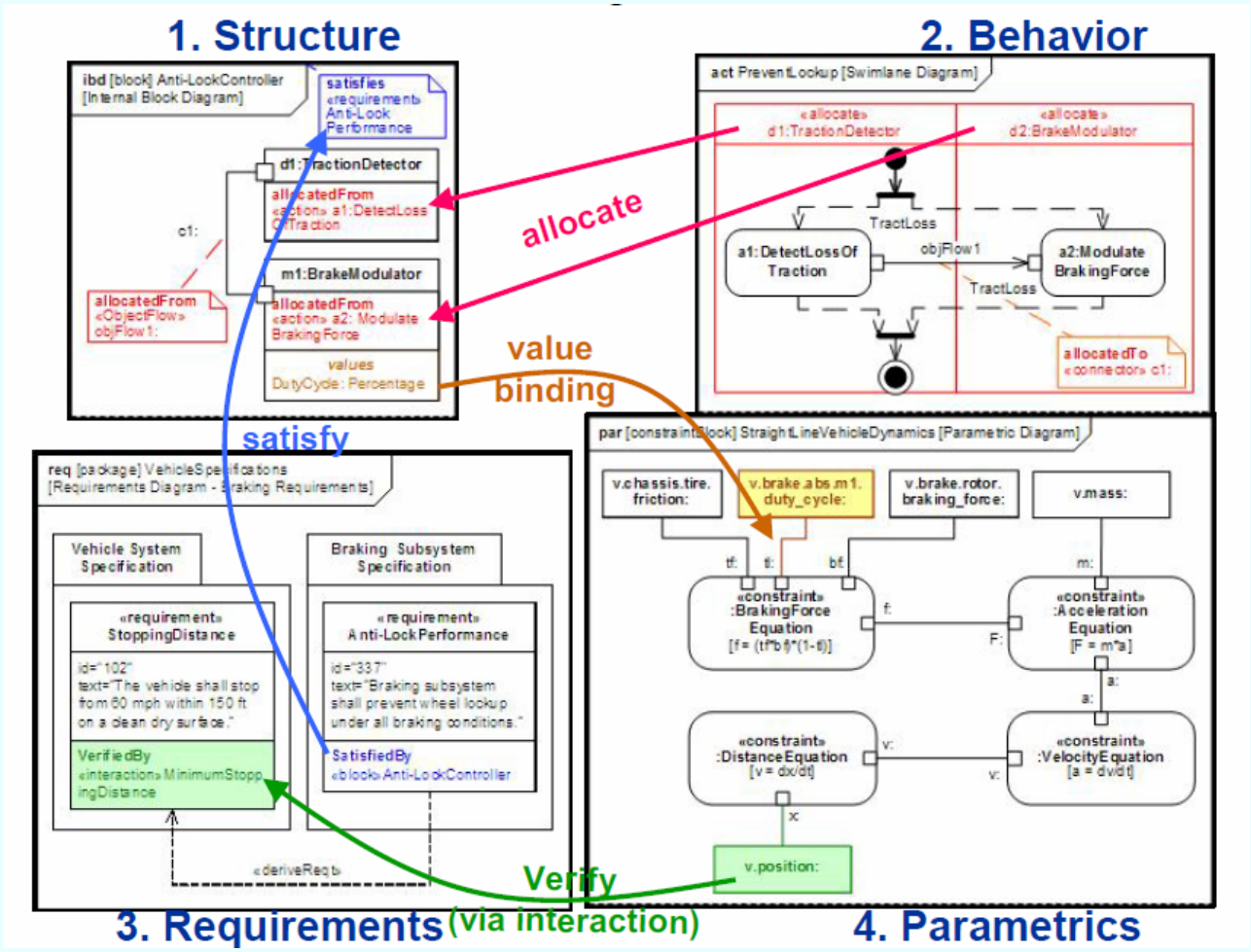
- Követelményeknek való megfelelés ellenőrzése
  - Követhetőség
- Szisztematikus elemzés:
  - Hibák hatásainak felmérése kombinatorikus módszerekkel: Kapcsolat a rendszerszintű (veszélyes) állapotok és a lokális (komponens szintű) hibák és események között
  - Az architektúra alapján végigellenőrizve a hatásokat
- Modell alapú vizsgálatok:
  - Extra-funkcionális jellemzők meghatározása (tipikusan sztochasztikus) módszerekkel:
    Lokális (komponens/kapcsolati szintű) paraméterek alapján rendszerszintű jellemzők számítása
  - Az architektúra alapján konstruálva az analízis modellt

Az architektúra terv igazolása: Követhetőség

- Hogyan segítik ezt a félformális nyelvek?
- Példa: SysML (Systems Modelling Language)
  - Architektúra tervezés első lépése: Block diagram



Relációk explicit megjelenítése és ellenőrzése



A szisztematikus hibaelemzés módszerei

1. Hibafa
2. Eseményfa
3. Ok-következmény analízis
4. Hibamód és -hatás analízis (FMEA)

1. Hibafa analízis

Rendszerszintű veszély okainak vizsgálata

- tipikusan felülről lefelé haladó analízis
- felderíti a kezelendő hibaokokat és -kombinációkat

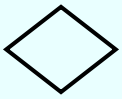
Hibafa konstrukció:

- (rendszerszintű) veszély, veszélyes állapot:
  azonosítás: környezet, követelmények, szabványok
- közbenső események, pszeudo-események:
  veszélyhez vezetnek,
  alacsonyabb szintű események Boole-logikai kombinációi (AND, OR)
- elsődleges események: további felbontás nincs

Hibafa grafikus elemkészlet

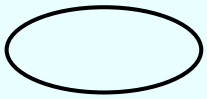
legfelső szintű vagy közbenső esemény

elsődleges (alapszintű) esemény

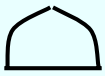


tovább nem vizsgált esemény

normál esemény (nem hiba vagy veszély)



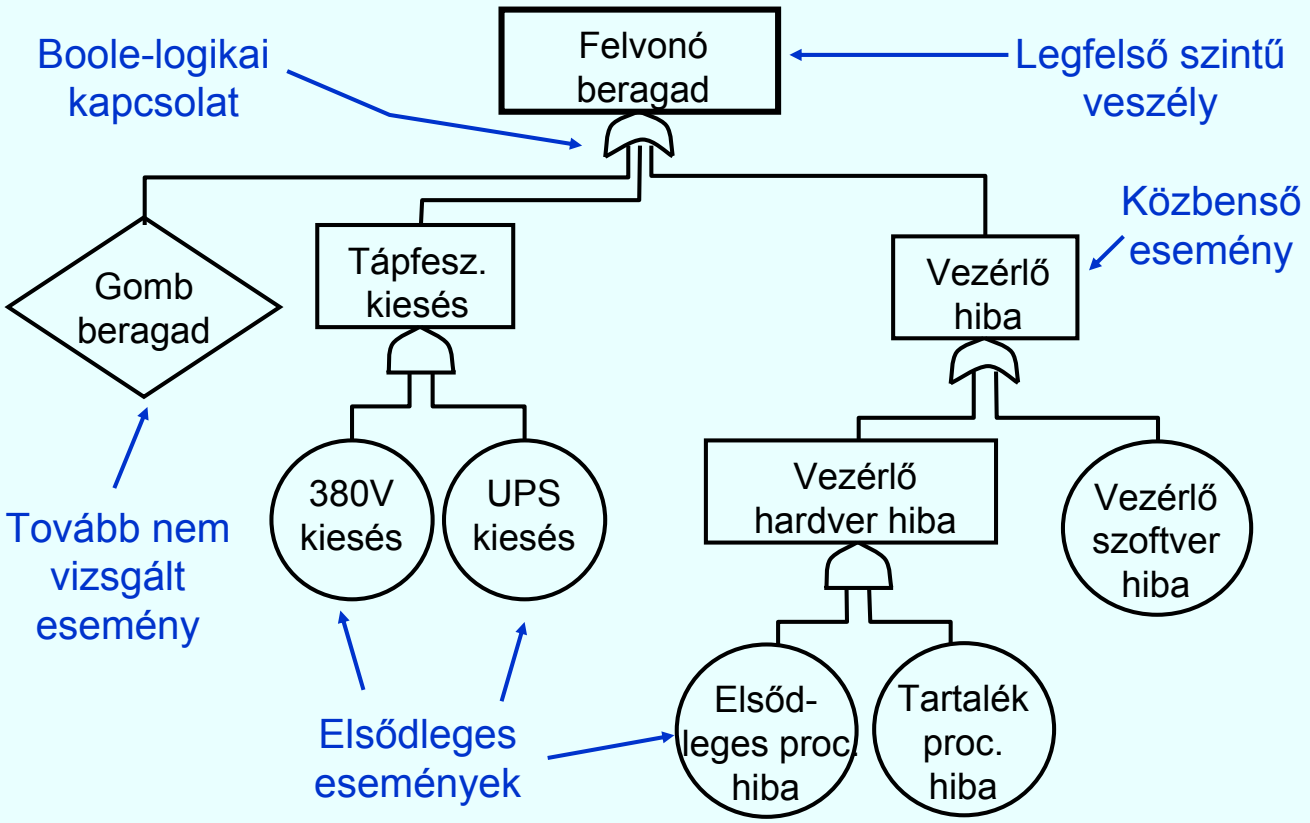
feltétel egy összetett esemény bekövetkezéséhez



ÉS kapu

VAGY kapu

Hibafa példa: Felvonó



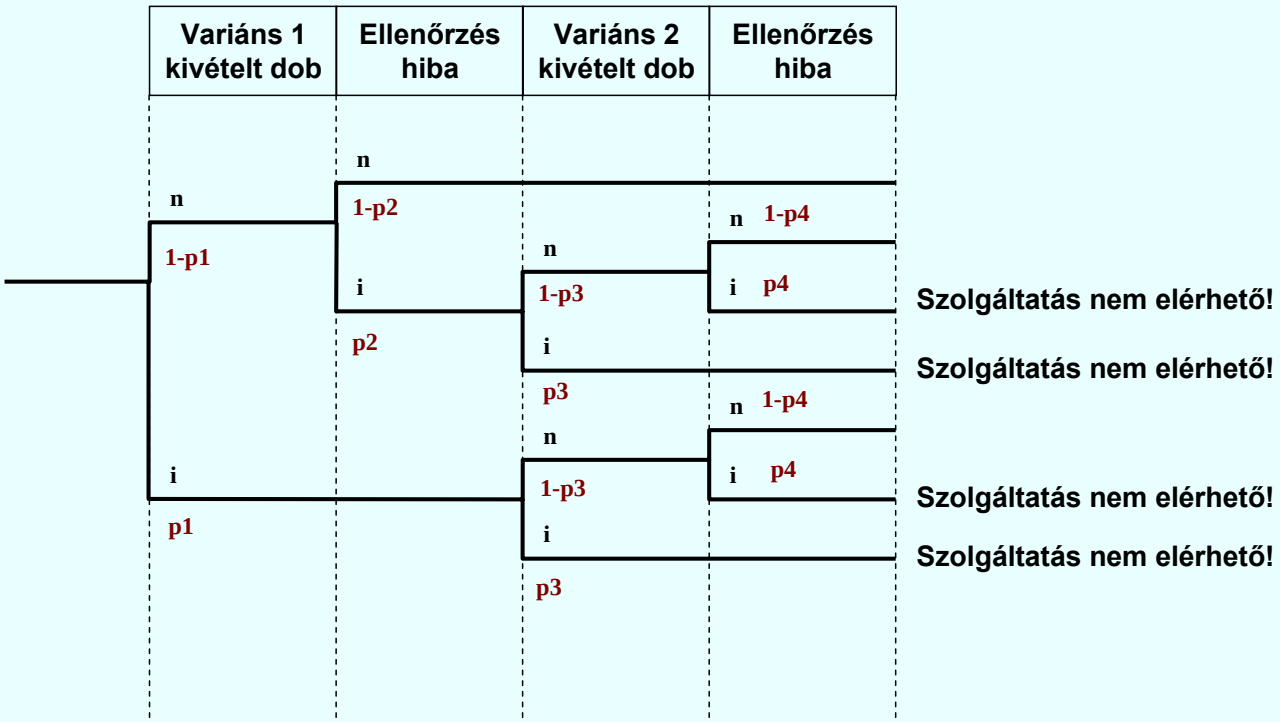
Hibafa analízis

- Minőségi (kvalitatív) analízis:
  - Hibafa redukció: Közbenső események feloldása
    → diszjunktív normál forma (OR a legtetején)
  - Azonosítható
    - egyszeres hibapont (SPOF)
    - kritikus esemény (több vágatban is szerepel)
- Mennyiségi (kvantitatív) analízis:
  - Alapszintű eseményekhez rendelt valószínűségek
    - komponens-adat, tapasztalat, becslés
  - Rendszerszintű veszély valószínűség számítása
    - AND kapu: szorzat (független eseményekre)
    - OR kapu: összegzés (felső becslés)
  - Problémák:
    - korreláló hibák, időbeli (hiba)szekvenciák kezelése

2. Eseményfa analízis

- Előrelépő analízis:
  Elsődleges események következményeit vizsgálja
  - kiváltó esemény: pl. egy komponens hibája
  - következmények: más komponensek állapotától függ
  - sorrendezés: oksági kapcsolat, időbeli viszony
  - elágazások: események bekövetkezése
- Hibázási „forgatókönyvek” vizsgálata
  - utak valószínűsége (elágazások valószínűsége alapján)
  - védelmi rendszerek hatékonysága
- Előnyök: Eseményszekvenciák vizsgálhatók
  - Korlátok: Komplexitás, többszörös események,
    minden kiváltó eseményhez külön diagram

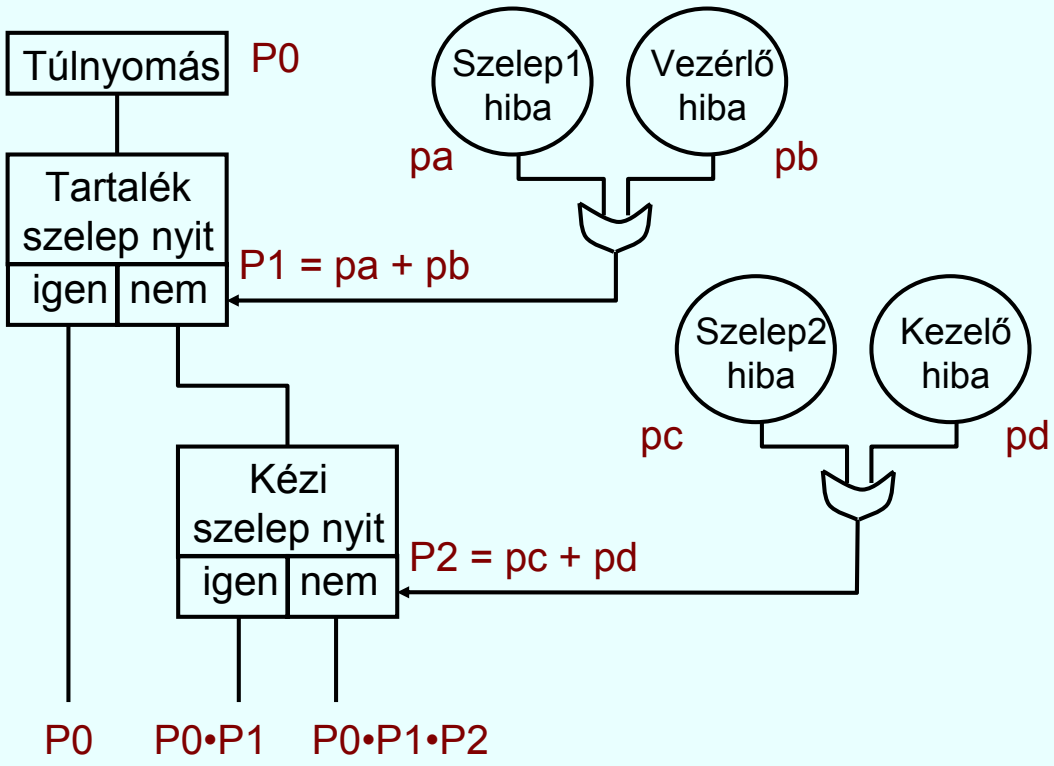
Eseményfa példa: Helyreállító blokkok



3. Ok-következmény analízis

- Eseményfa és hibafa összekapcsolása
  - eseményfa: forgatókönyvek (szekvencia)
  - csatolt hibafa: esemény bekövetkezés „indoklása”, rendelkezésre állás számítása
- Előnyök:
  - szekvenciák (előrelépő analízis) és ok-okozati kapcsolatok (hátralépő analízis) együtt
- Korlátok:
  - minden kritikus eseményhez külön diagram szükséges

Példa ok-következmény analízisre



4. Hibamód és -hatás analízis (FMEA)

- Hibák és hatásaik felsorolása
- Előny:
  - szisztematikus áttekintés
  - redundancia felismerése

Komponens	**Hibamód**	**Valószínűség**	**Hatás**
L határérték-túllépés vizsgálat	> L átmegy ≤ L nem megy át	65% 35%	- túlnyomás - technológiai hiba
...	...	...	...

A modell alapú elemzés módszerei

Cél: Architektúra változatok kiértékelése

- Analízis modellek készítése és paraméterezése az architektúra modellje alapján
  - Matematikai modell, jellemzői számíthatók („megoldható”)
- Mit ír le az analízis modell?
  Komponens paraméterek -> Rendszerszintű jellemzők
  - Teljesítmény
  - Megbízhatóság
  - Biztonság
  - Adatbiztonság
- Moduláris modellalkotás a tipikus
  - Architektúra: Komponensek és kapcsolatok
  - Analízis modell: Ehhez analízis modellkönyvtár

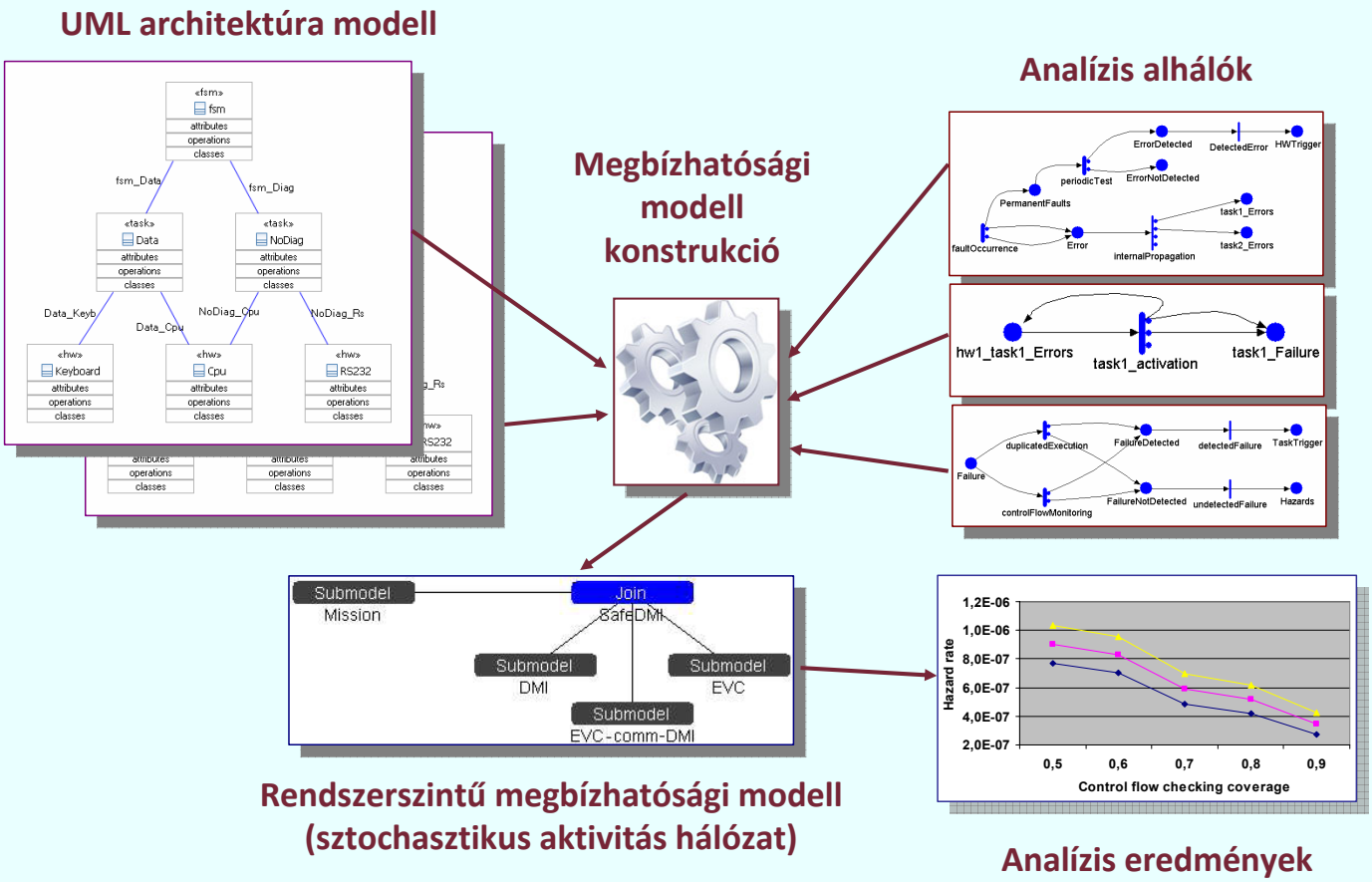
Modell alapú architektúra vizsgálatok

	Megbízhatósági modell	**Teljesítmény modell**	**Biztonsági modell**
Komponens paraméterek	Meghibásodási tényező, lappangási idő, javítási tényező, hibafedés, …	Funkció lokális végrehajtási idő, taszk prioritás, processzor ütemezés	Veszély gyakoriság
Kapcsolat paraméterek	Hibaterjedési valószínűség, hibaterjedés feltételei, javítási stratégia	Hívás továbbítási gyakoriság, hívás szinkronitás	Veszély forgatókönyv, veszély kombinációk
Modell	Markov-lánc, Petri-háló	Sorbanállási háló	Markov-lánc, Petri-háló
Rendszer jellemzők (számított)	Megbízhatóság, rendelkezésre állás, készenlét, MTTF, MTTR, MTBF	Kiszolgálási idő, taszk áteresztő-képesség, processzor kihasználtság	Rendszerszintű veszély gyakoriság

Első példa: Megbízhatósági modellezés

	Megbízhatósági modell	**Teljesítmény modell**	**Biztonsági modell**
Komponens paraméterek	Meghibásodási tényező, lappangási idő, javítási tényező, hibafedés, …	Funkció lokális végrehajtási idő, taszk prioritás, processzor ütemezés	Veszély gyakoriság
Kapcsolat paraméterek	Hibaterjedési valószínűség, hibaterjedés feltételei, javítási stratégia	Hívás továbbítási gyakoriság, hívás szinkronitás	Veszély forgatókönyv, veszély kombinációk
Modell	Markov-lánc, Petri-háló	Sorbanállási háló	Markov-lánc, Petri-háló
Rendszer jellemzők (számított)	Megbízhatóság, rendelkezésre állás, készenlét, MTTF, MTTR, MTBF	Kiszolgálási idő, áteresztő-képesség, processzor kihasználtság	Rendszerszintű veszély gyakoriság

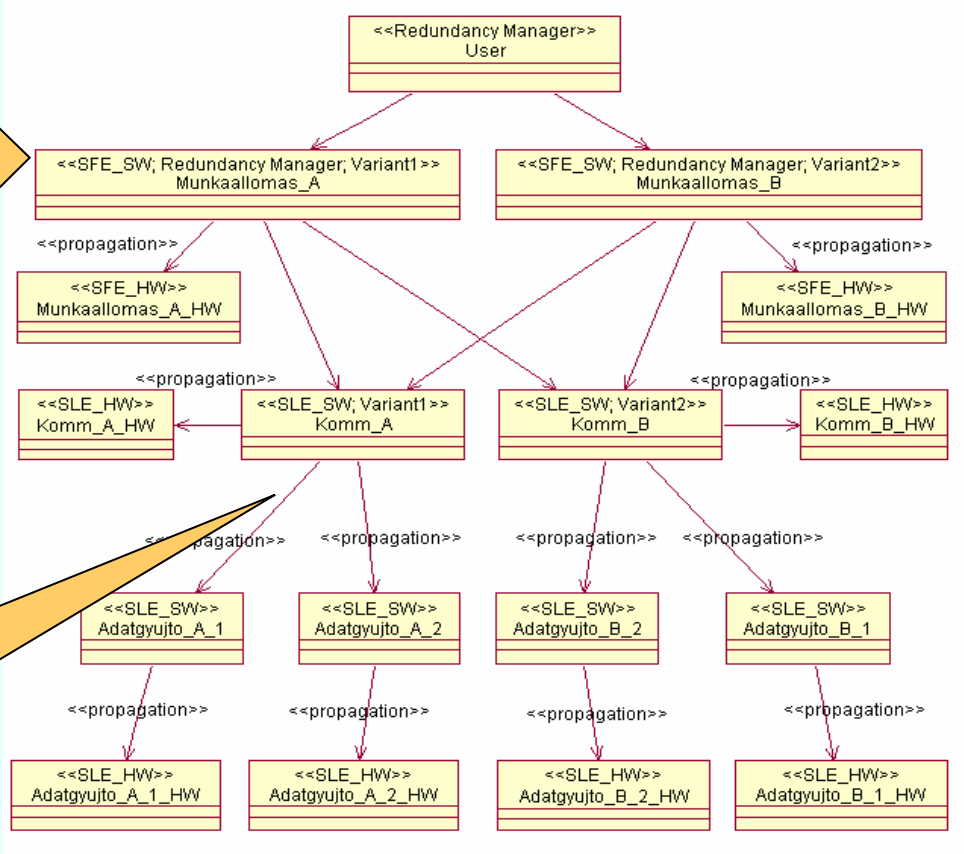
UML alapú megbízhatósági modellezés



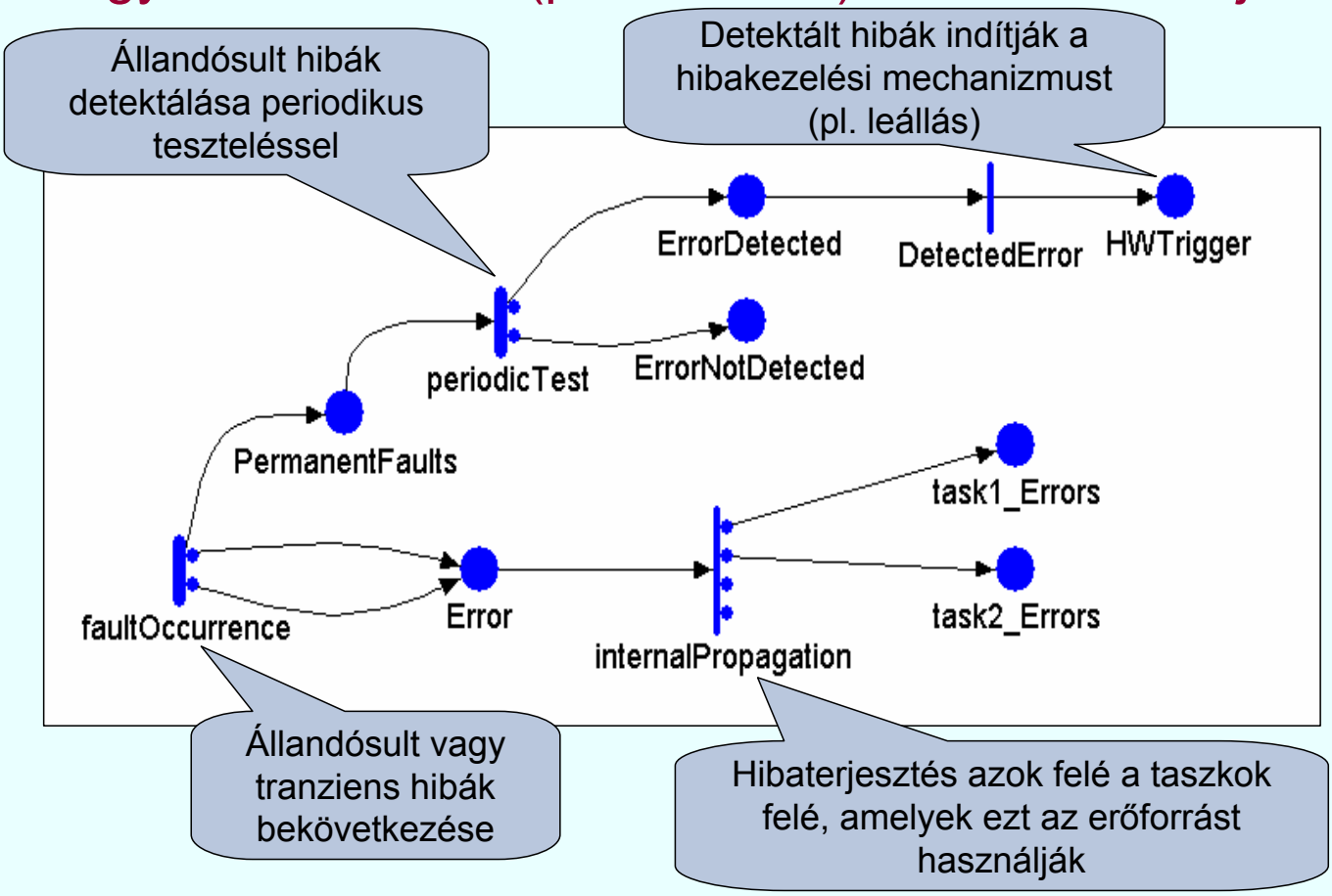
Megbízhatósági modellezés – Tervezői modell

Komponens:
- Típus (HW, SW)
- Szerep (variáns, red. menedzser)
- Meghibásodási jellemzők:
  * meghibásodási gyakoriság,
  * lappangási idő,
  * javítási idő

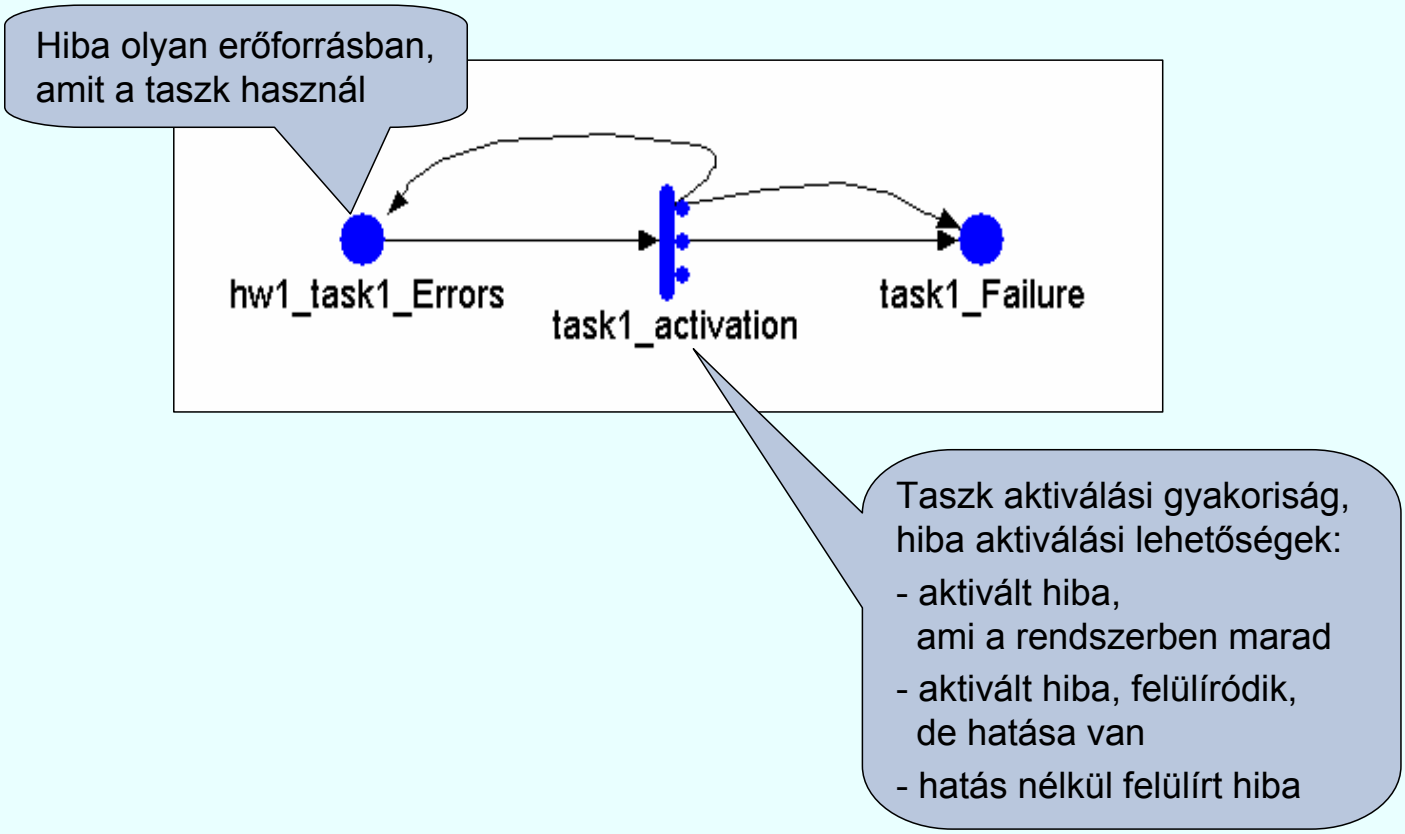
Kapcsolat:
- Hibaterjesztési jellemzők:
  * hibaterjesztési valószínűség



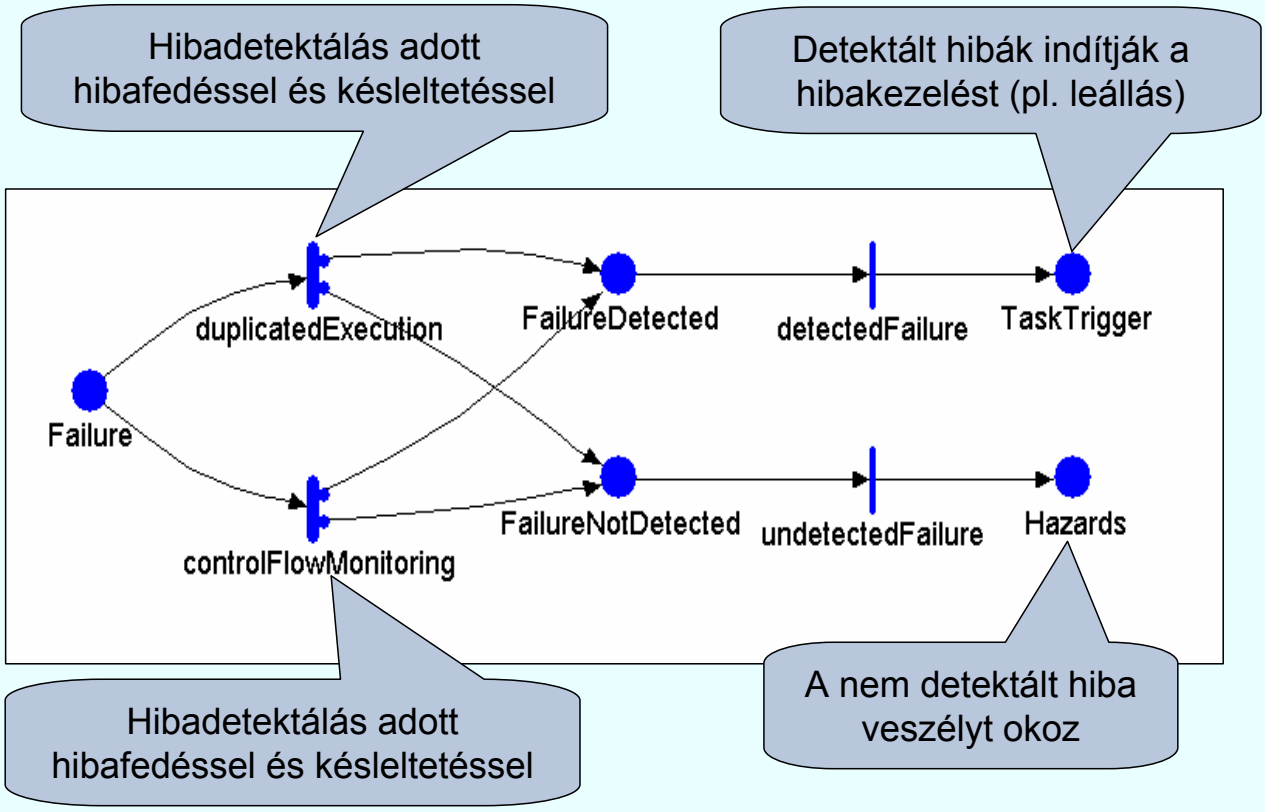
Egy HW erőforrás (pl. memória) analízis modellje



A hibaterjesztés analízis modellje

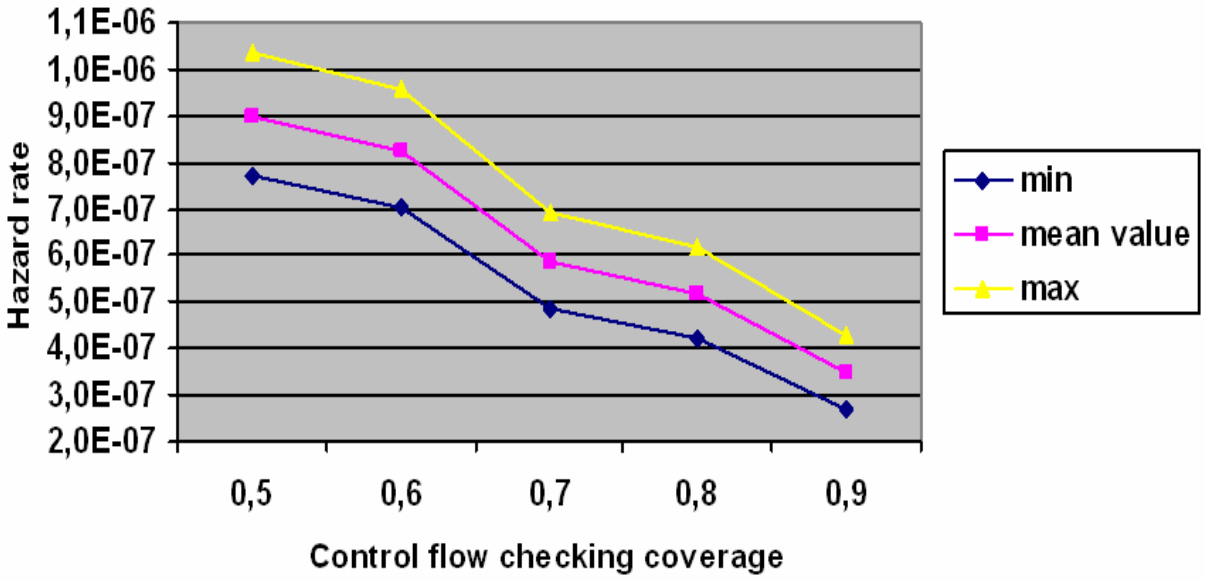


Egy szoftver taszk analízis modellje

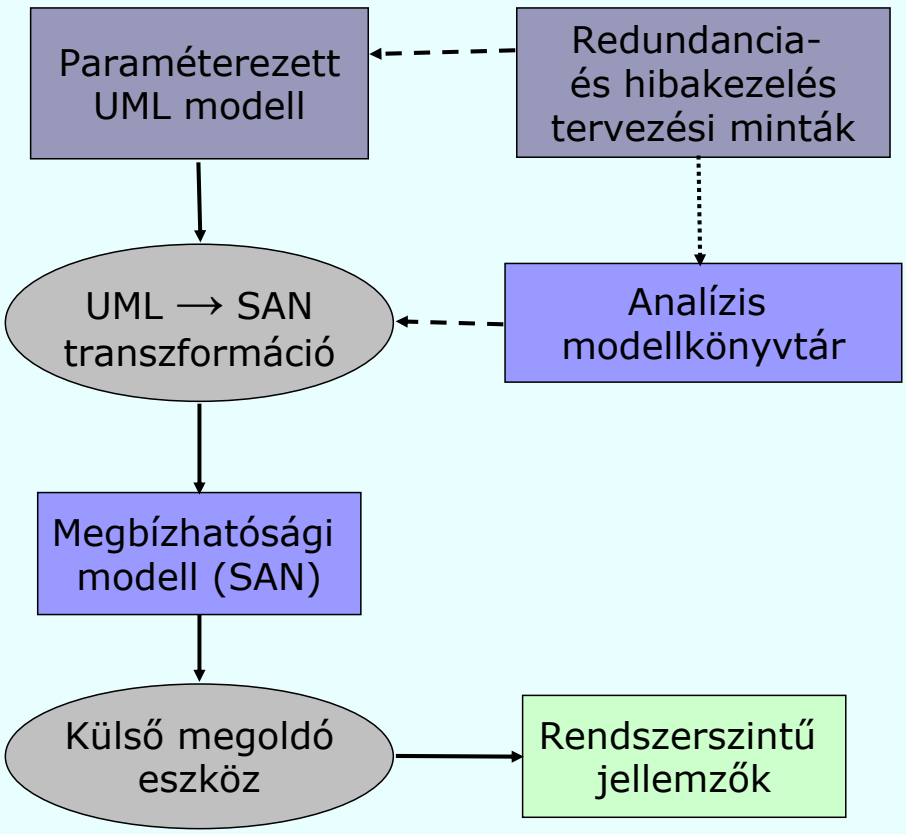


Analízis eredmény (példa)

- Ha a hibadetektálás hibafedése 50% alá csökken, akkor a SIL2 követelmény (10^{-7}<THR<10^{-6}) nem teljesül



Automatikus analízis eszköz



Második példa: Teljesítmény modellezés

	Megbízhatósági modell	**Teljesítmény modell**	**Biztonsági modell**
Komponens paraméterek	Meghibásodási tényező, lappangási idő, javítási tényező	**Funkció lokális végrehajtási idő, taszk prioritás, processzor ütemezés**	Veszély gyakoriság
Kapcsolat paraméterek	Hibaterjedési valószínűség, hibaterjedés feltételei, javítási stratégia	**Hívás továbbítási gyakoriság, hívás szinkronitás**	Veszély forgatókönyv, veszély kombinációk
Modell	Markov-lánc, Petri-háló	**Sorbanállási háló**	Markov-lánc, Petri-háló
Rendszer jellemzők (számított)	Megbízhatóság, rendelkezésre állás, készenlét, MTTF, MTTR, MTBF	**Kiszolgálási idő, áteresztő-képesség, processzor kihasználtság**	Rendszerszintű veszély gyakoriság

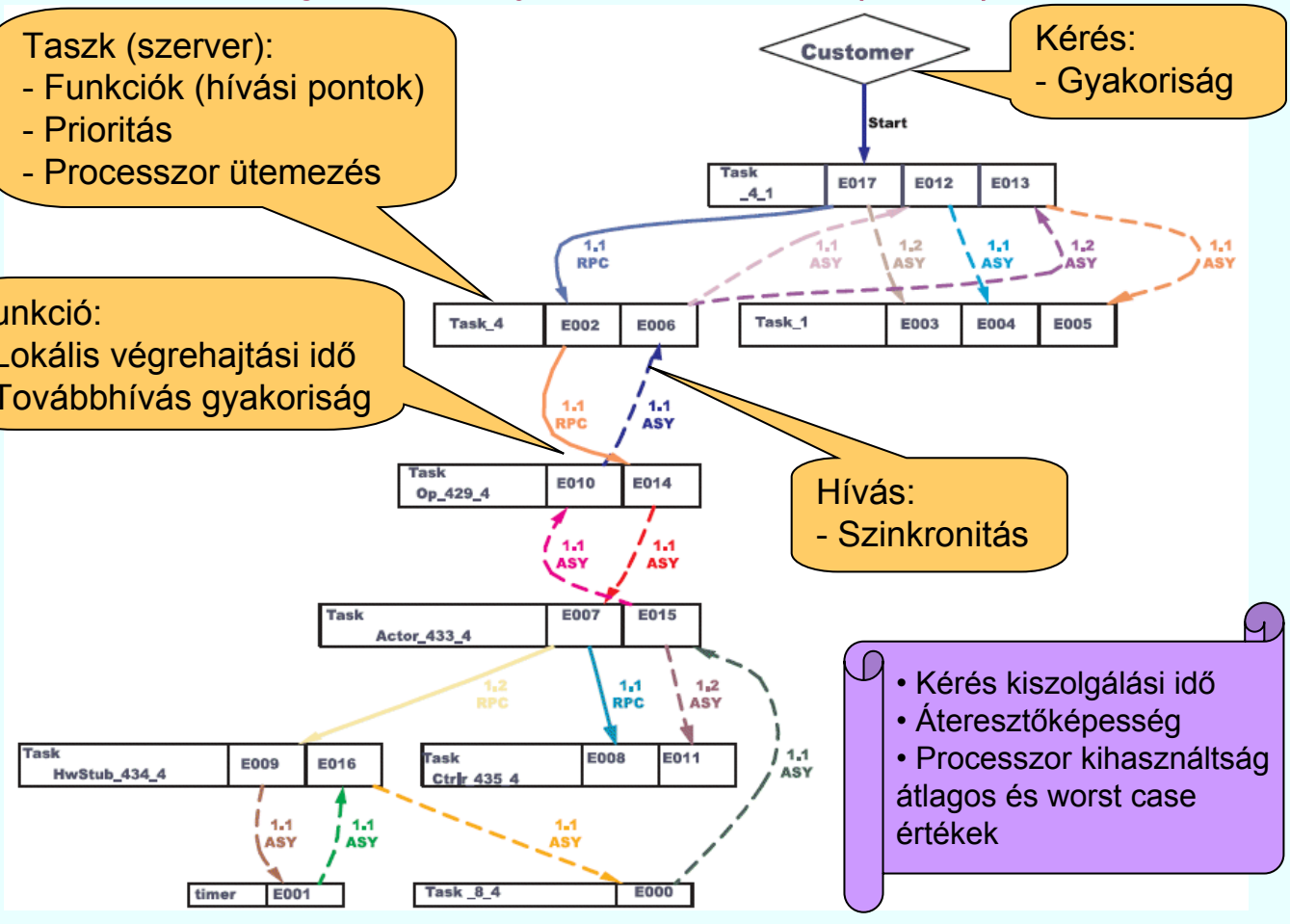
Teljesítmény modellezés (LQN)

Taszk (szerver):
- Funkciók (hívási pontok)
- Prioritás
- Processzor ütemezés

Kérés:
- Gyakoriság

Funkció:
- Lokális végrehajtási idő
- Továbbhívás gyakoriság

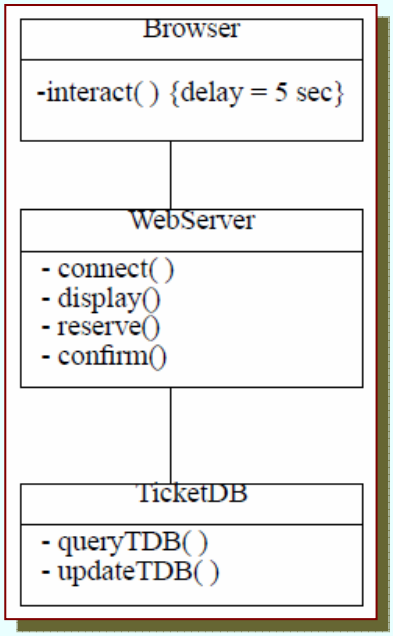
Hívás:
- Szinkronitás

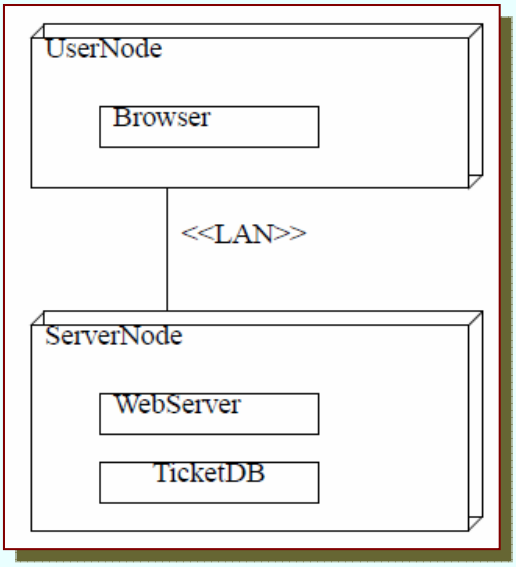


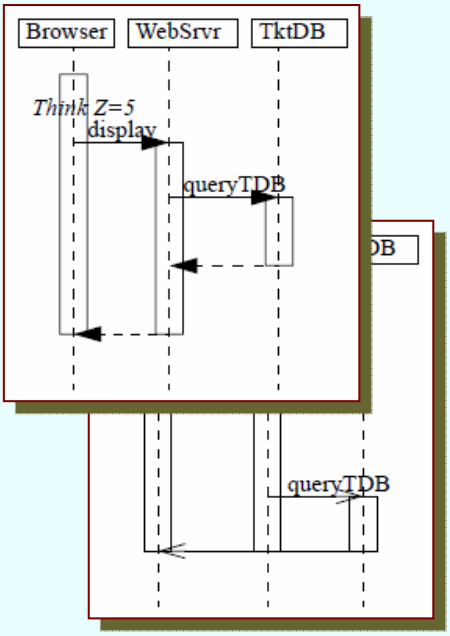
- Kérés kiszolgálási idő
- Áteresztőképesség
- Processzor kihasználtság

átlagos és worst case értékek

Az architektúra leképezése teljesítménymodellre





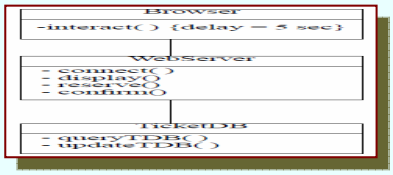


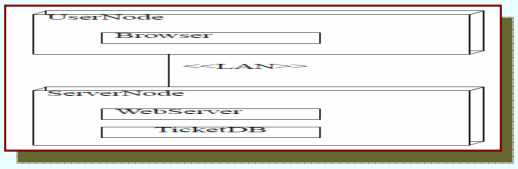
Osztályok Telepítés Interakciók

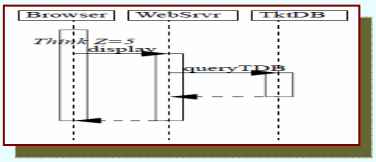
Lokális teljesítmény paraméterek megadása:

- Attribútumok (konvenció alapján értelmezhető)
- UML tagged value

Az architektúra leképezése teljesítménymodellre

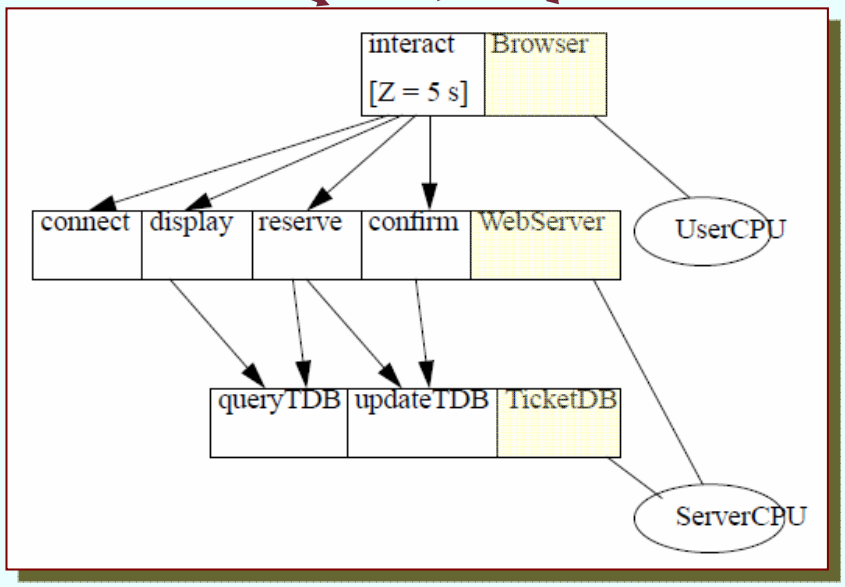






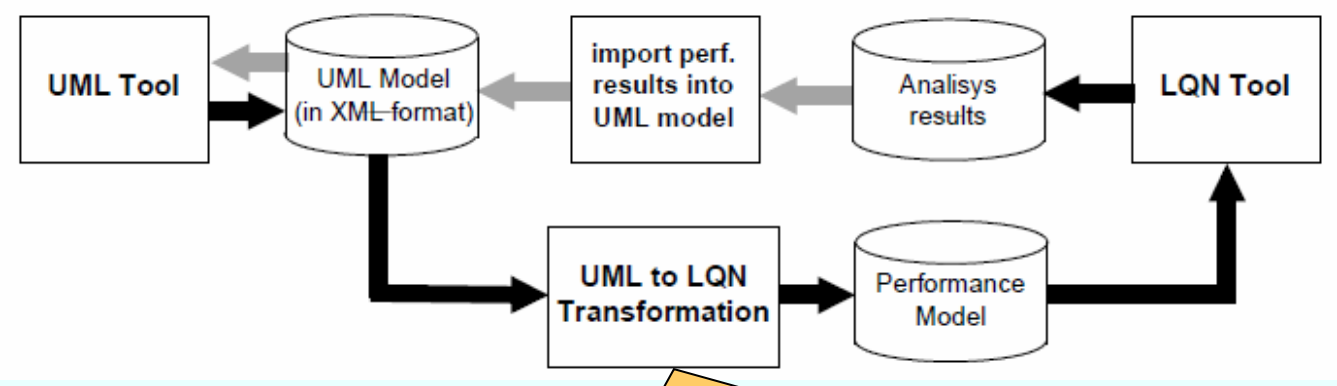
Osztályok Telepítés Interakciók

Modell-transzformáció



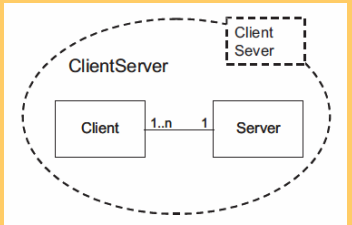
LQN teljesítmény-modell

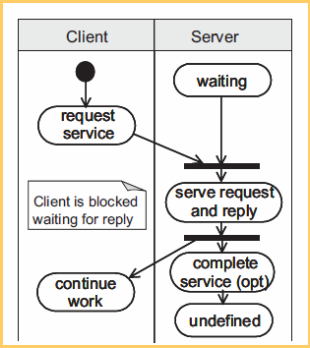
Az architektúra leképezése teljesítménymodellre



- Architektúra minták használata

Szinkron üzenetküldés





Tartalomjegyzék

- A fázis ki- és bemenetei
- Az architektúra tervek elkészítése
  - A komponensek azonosítása
  - Mit határoz meg az architektúra?
- Ellenőrzési feladatok
  - Követelményeknek való megfelelőség, követhetőség
  - Hibahatások vizsgálata
  - Extra-funkcionális követelmények vizsgálata
- Teszt tervezés

Szoftver-hardver integrációs teszterv

- Dokumentálandó:
  - Teszt esetek, típusok
  - Teszt környezet (eszközök, konfiguráció leírás)
  - Teszt kritériumok (teljes elvégzés megítélhető)
- Tevékenységek
  - Telepítés és rendszerintegráció megkülönböztetve
  - Telephelyi és alkalmazói tesztek elkülönítése
  - Teszt esetek és eredmények gépi rögzítése
- Teszt módszerek: Ld. az integrációs fázisban!