

Szoftver architektúra tervek ellenőrzése

Majzik István

Budapesti Műszaki és Gazdaságtudományi Egyetem

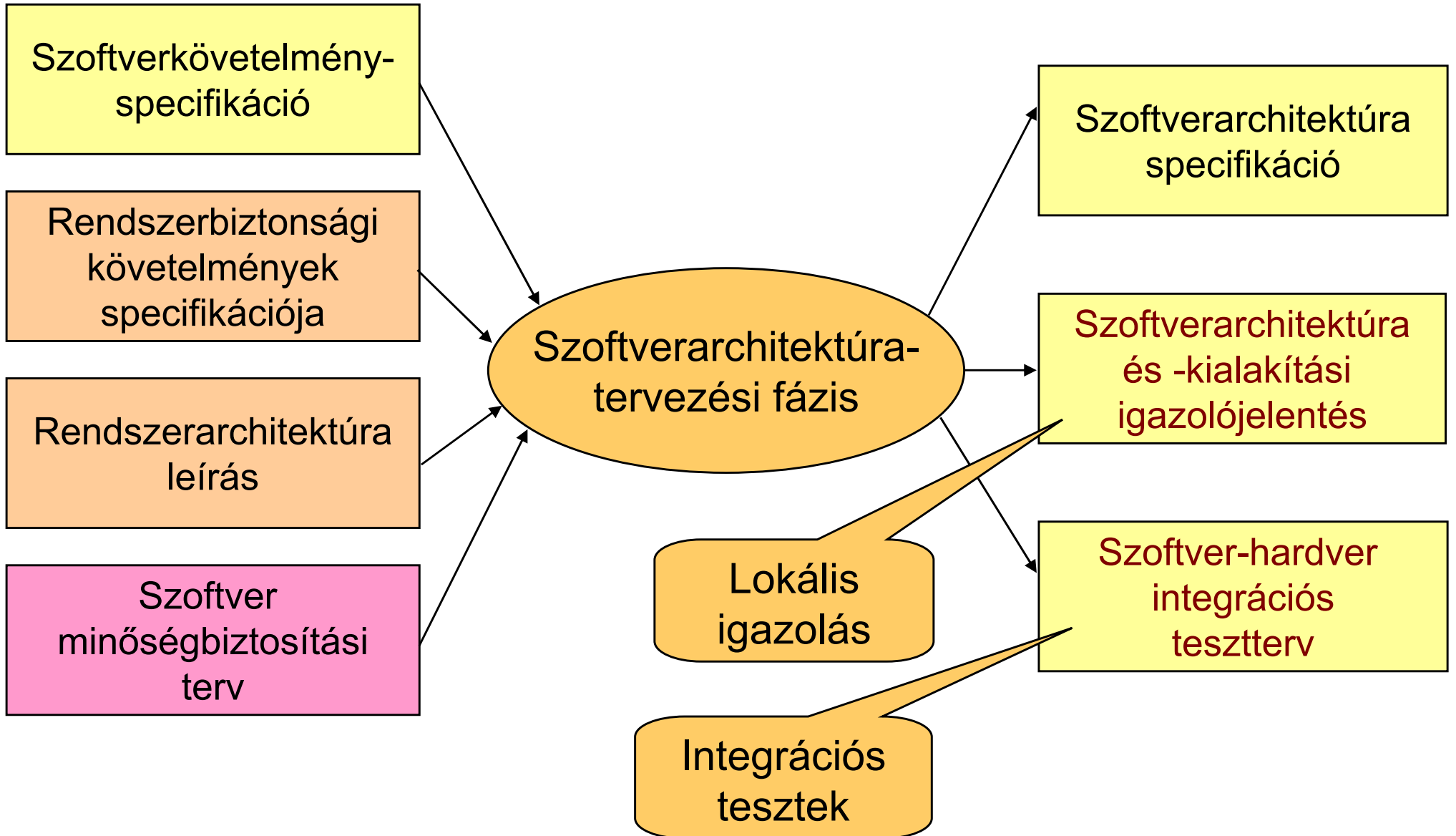
Méréstechnika és Információs Rendszerek Tanszék

<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 megfelelés, követhetőség
 - Hibahatások vizsgálata
 - Extra-funkcionális követelmények vizsgálata
- Teszt tervezés

Kimenetek és bemenetek



Tartalomjegyzék

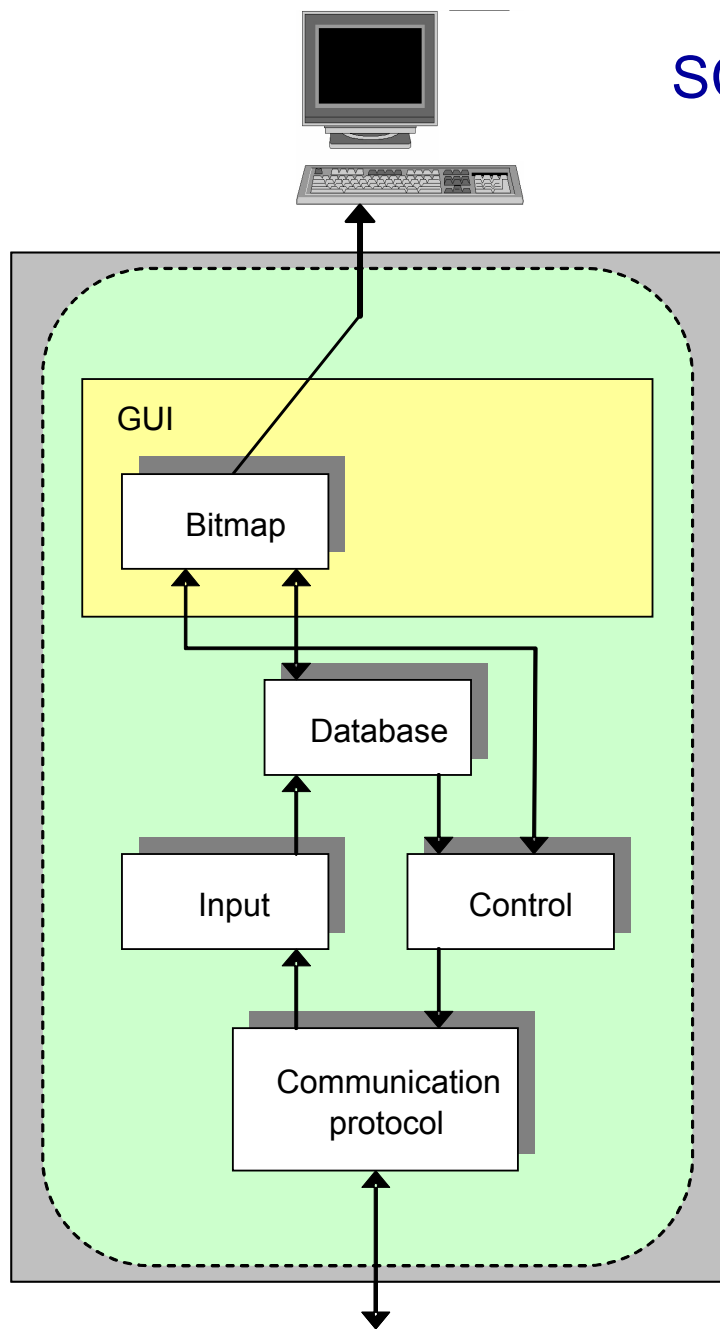
- A fázis ki- és bemenetei
- Az architektúra tervek elkészítése
 - A komponensek azonosítása
 - Mit határoz meg az architektúra?
- Ellenőrzési feladatok
 - Követelményeknek való megfelelés, követhetőség
 - Hibahatások vizsgálata
 - Extra-funkcionális követelmények vizsgálata
- Teszt tervezés

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
 - 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).
- Szoftverkövetelményeknek való megfelelés
 - Változások hatásvizsgálata
 - SIL megfelelés

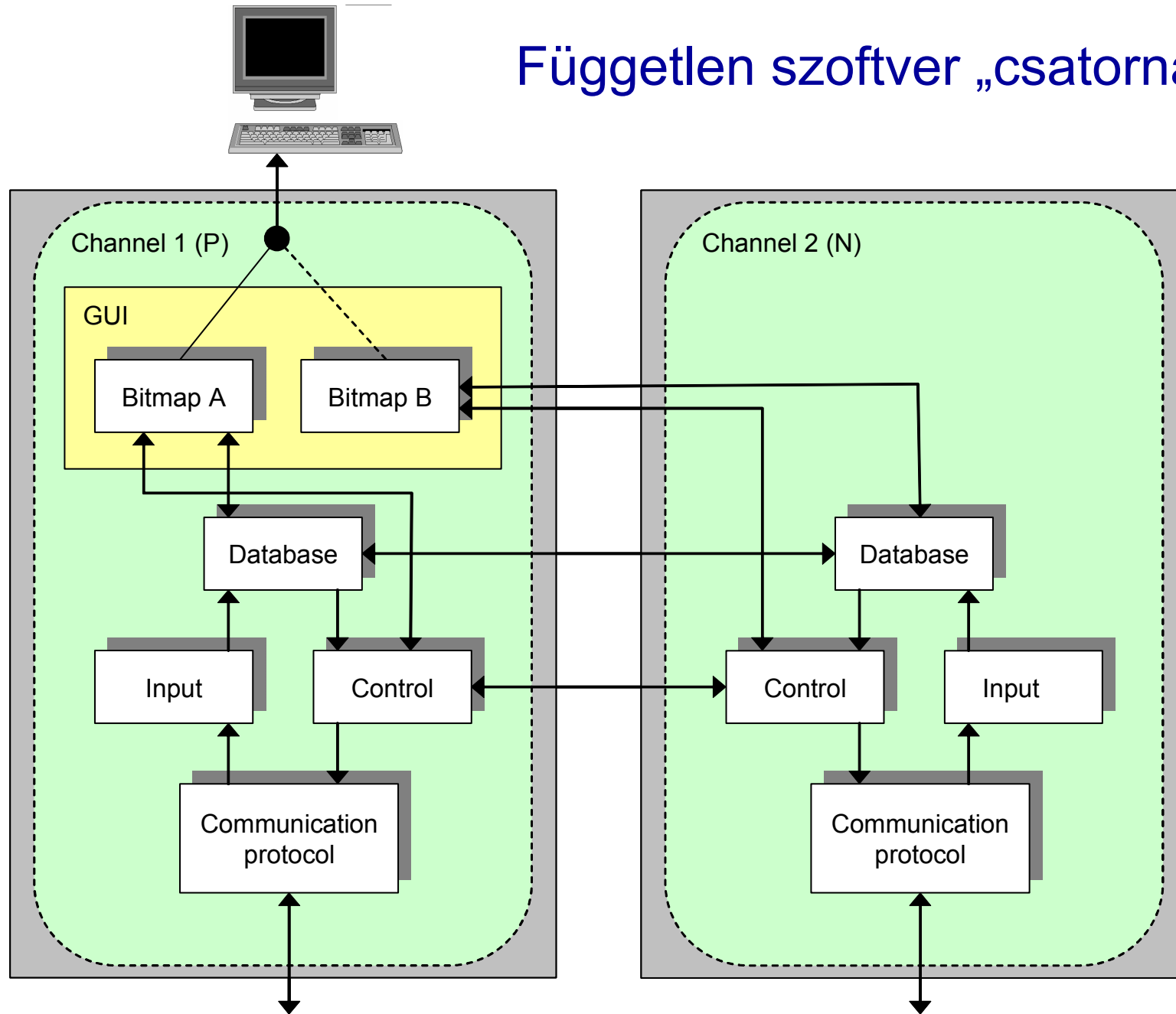
Architekturális megoldás (példa) SIL csökkentésre

SCADA rendszer, SIL4 követelmény



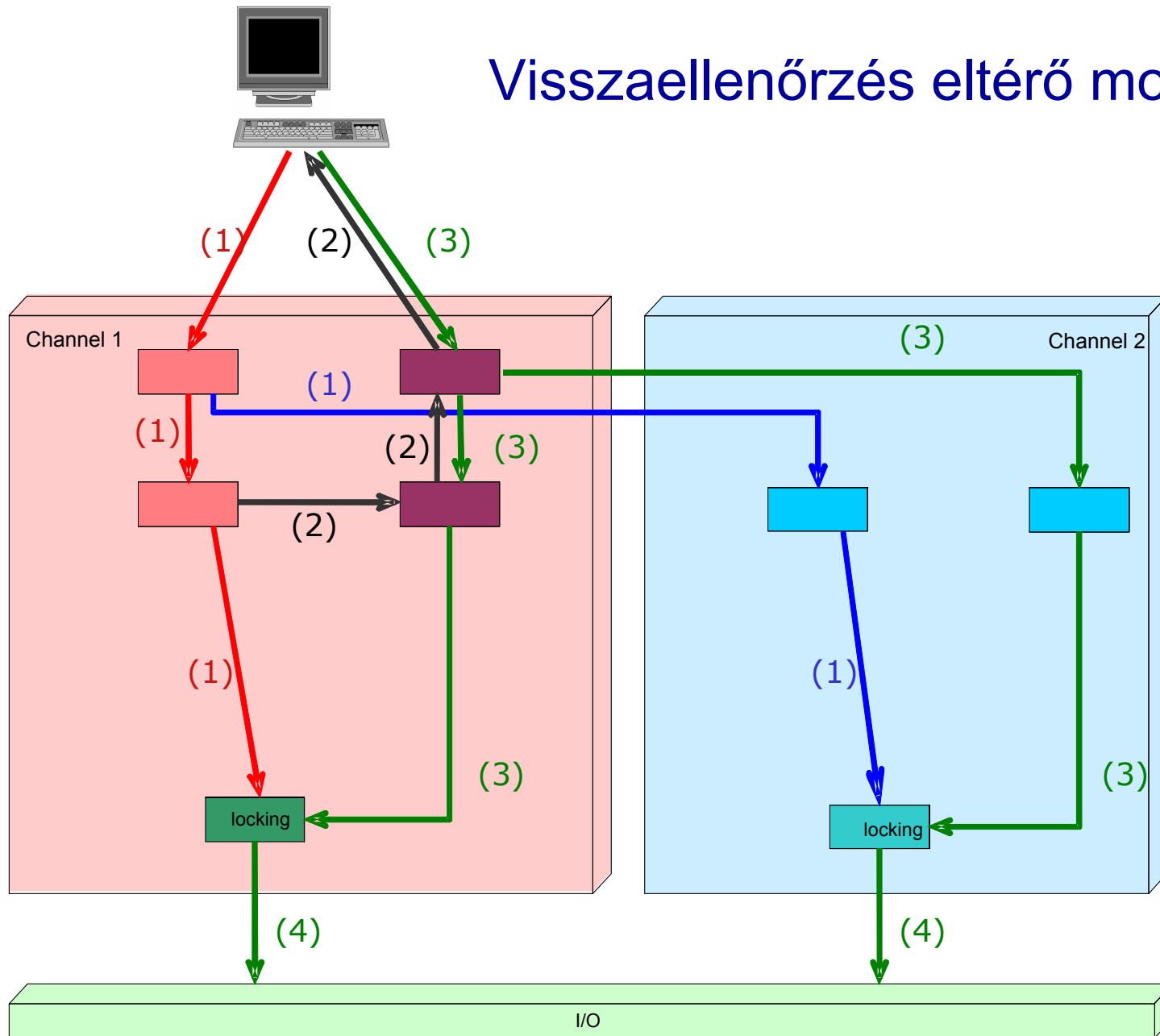
Architekturális megoldás (példa) SIL csökkentésre

Független szoftver „csatornák”



Architekturális megoldás (példa) SIL csökkentésre

Visszaellenőrzés eltérő modulokon



Hibakezelés: Redundancia

- **Hardver redundancia:**

- **Azonos komponensek:**

- Működés közbeni tranziens és állandósult 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:**

Előírt módszerek (EN50128) - 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 (releváns döntések) |
|-----------------------------|--|
| Szolgáltatásbiztonság | Hibadetektálás, hibabehatárolás, hibakezelés (redundancia) |
| Teljesítmény | Erőforrás hozzárendelés, erőforrás menedzsment |
| Biztonságosság | Veszély csökkentés, veszély kontroll (monitorozás) |
| Adatbiztonság | Integritás ellenőrzés, 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 megfelelés, követhetőség
 - Hibahatások vizsgálata
 - Extra-funkcionális követelmények vizsgálata
- Teszt tervezés

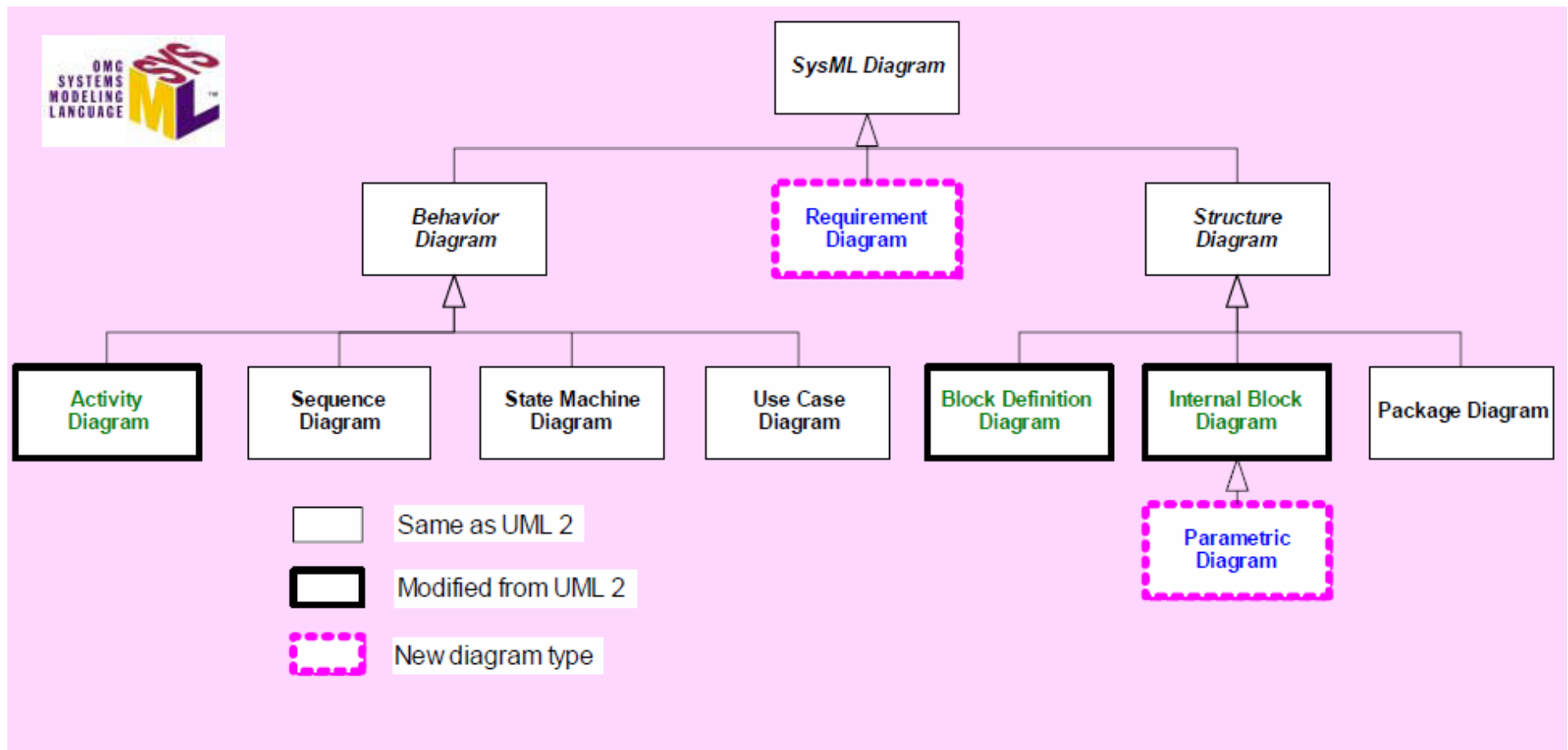
A szoftverarchitektúra igazolása (áttekintés)

Vizsgálati technikák áttekintése:

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

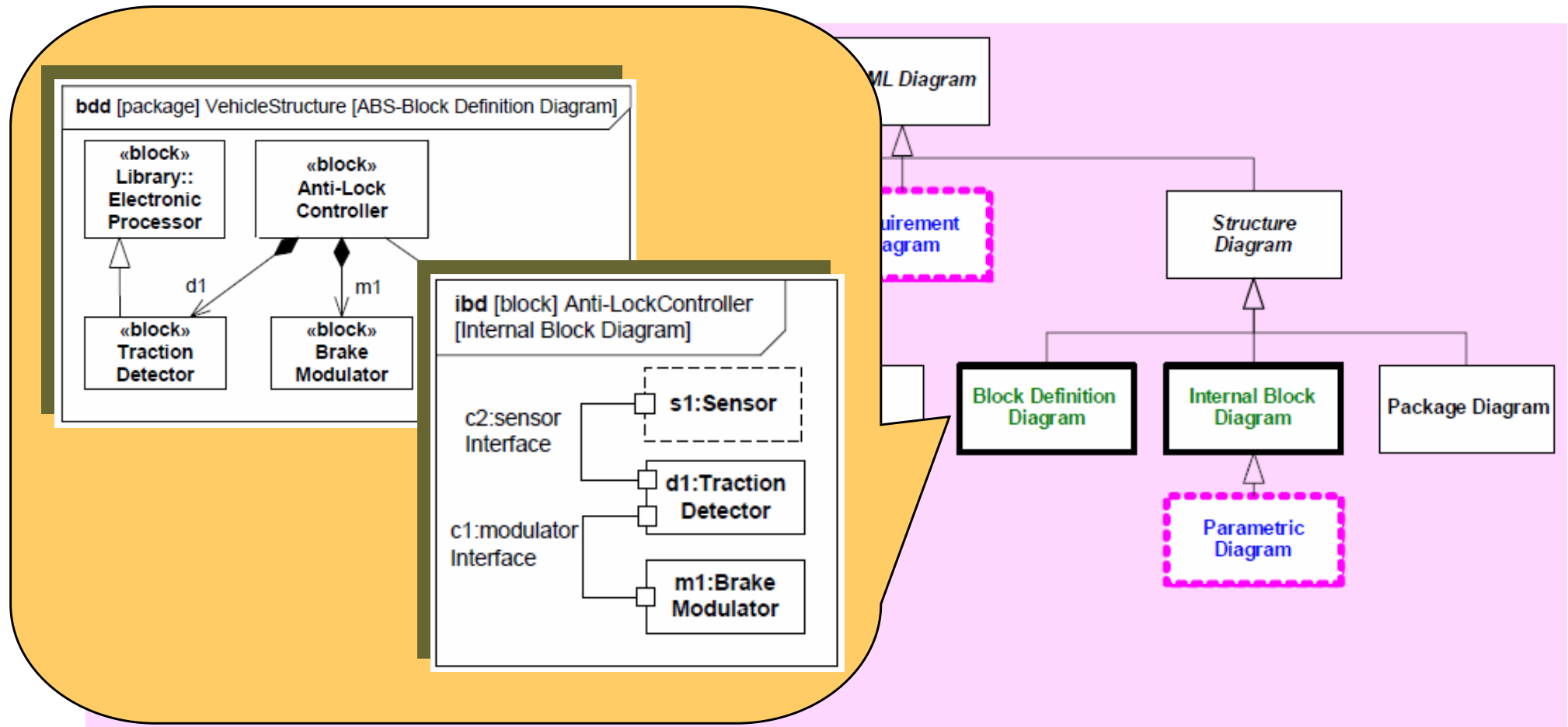
Az architektúra terv igazolása: Követhetőség

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



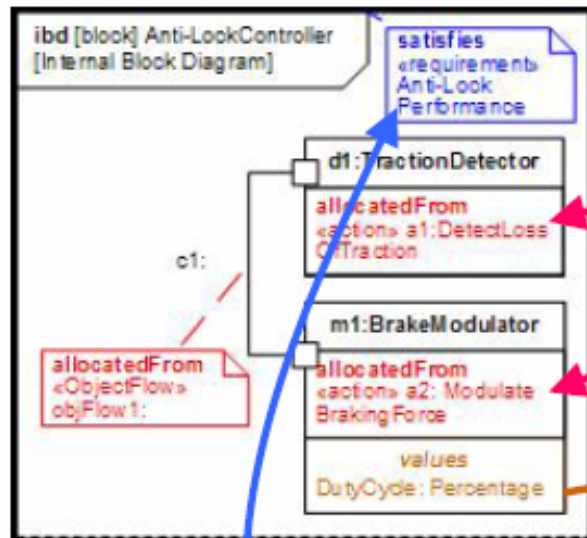
Az architektúra terv igazolása: Követhetőség

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

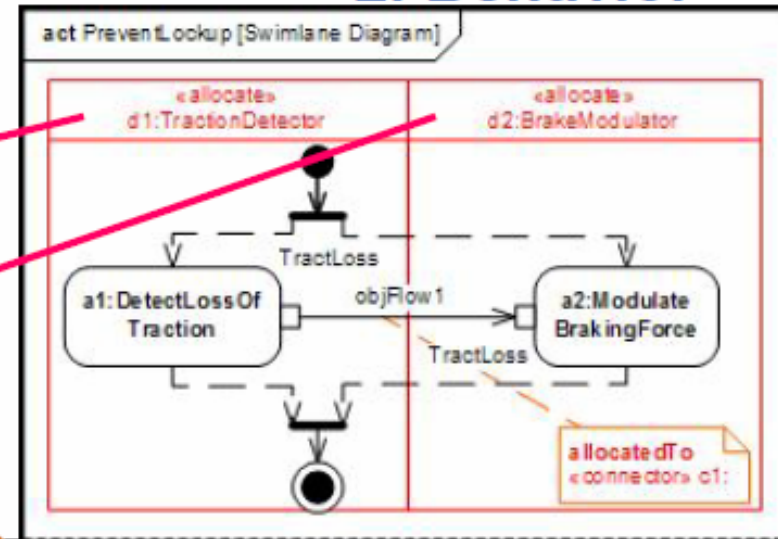


Relációk explicit megjelenítése és ellenőrzése

1. Structure



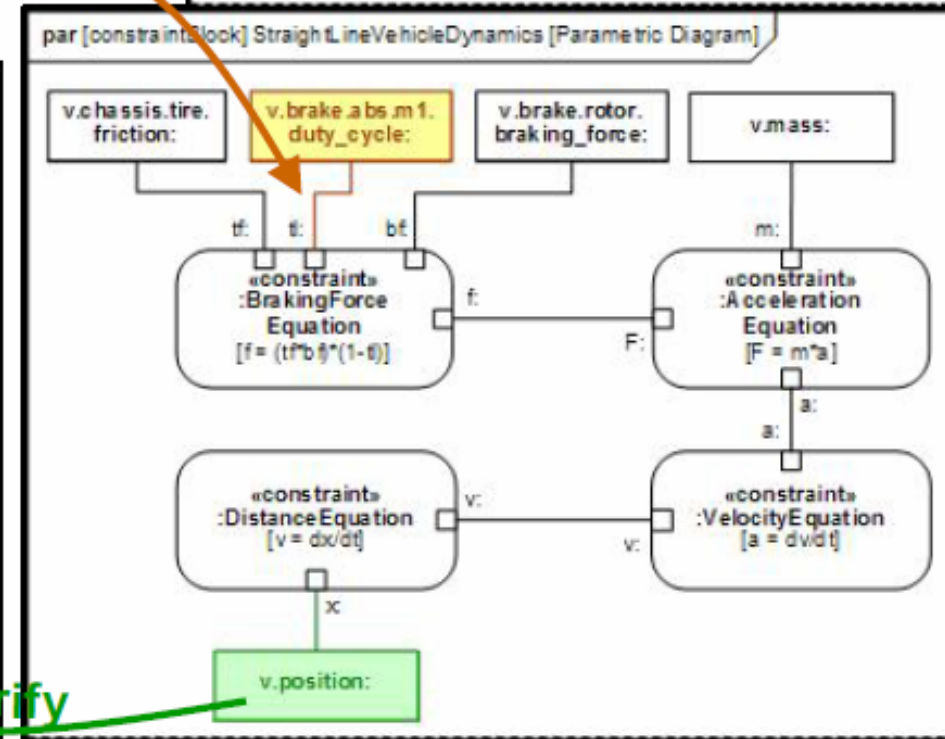
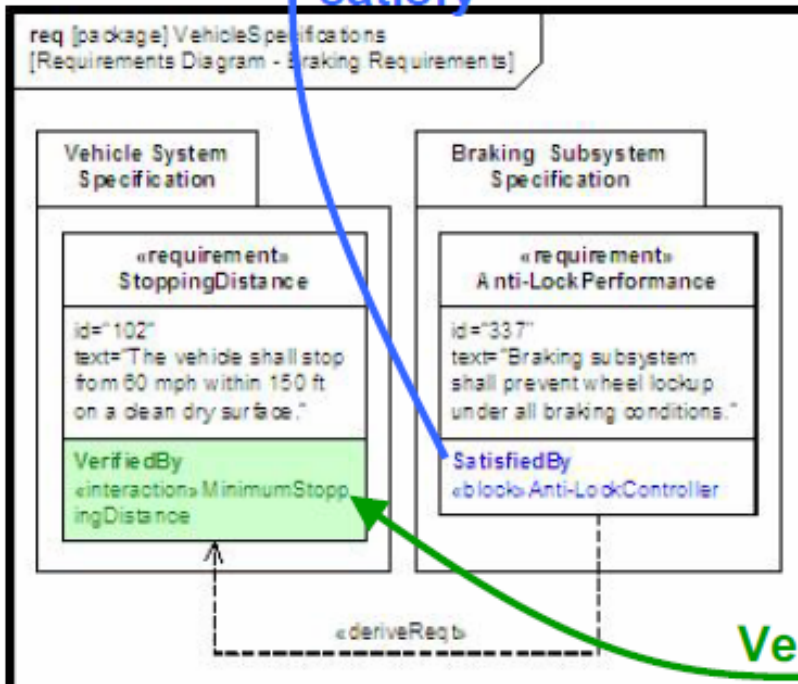
2. Behavior



allocate

value binding

satisfy



Verify

3. Requirements (via interaction)

4. Parametrics

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

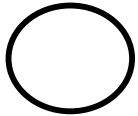
Hibafa konstrukció:

- (Rendszerszintű) veszély, veszélyes állapot:
Azonosítás: környezet, követelmények, szabványok
- Közbenső események, pseudo-események:
Veszélyhez vezetnek,
Boole-logikai kombinációi (AND, OR) alacsonyabb szintű eseményeknek
- Elsődleges események: további felbontás nincs

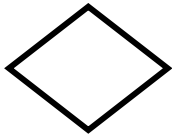
Hibafa grafikus elemkészlet



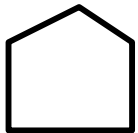
legfelső szintű vagy közbenső esemény



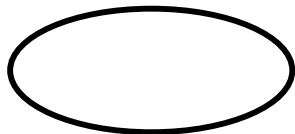
elsődleges (alapszintű) esemény



tovább nem vizsgált esemény



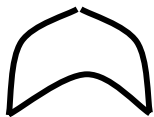
normál esemény (nem hiba vagy veszély)



feltétel egy összetett esemény
bekövetkezéséhez

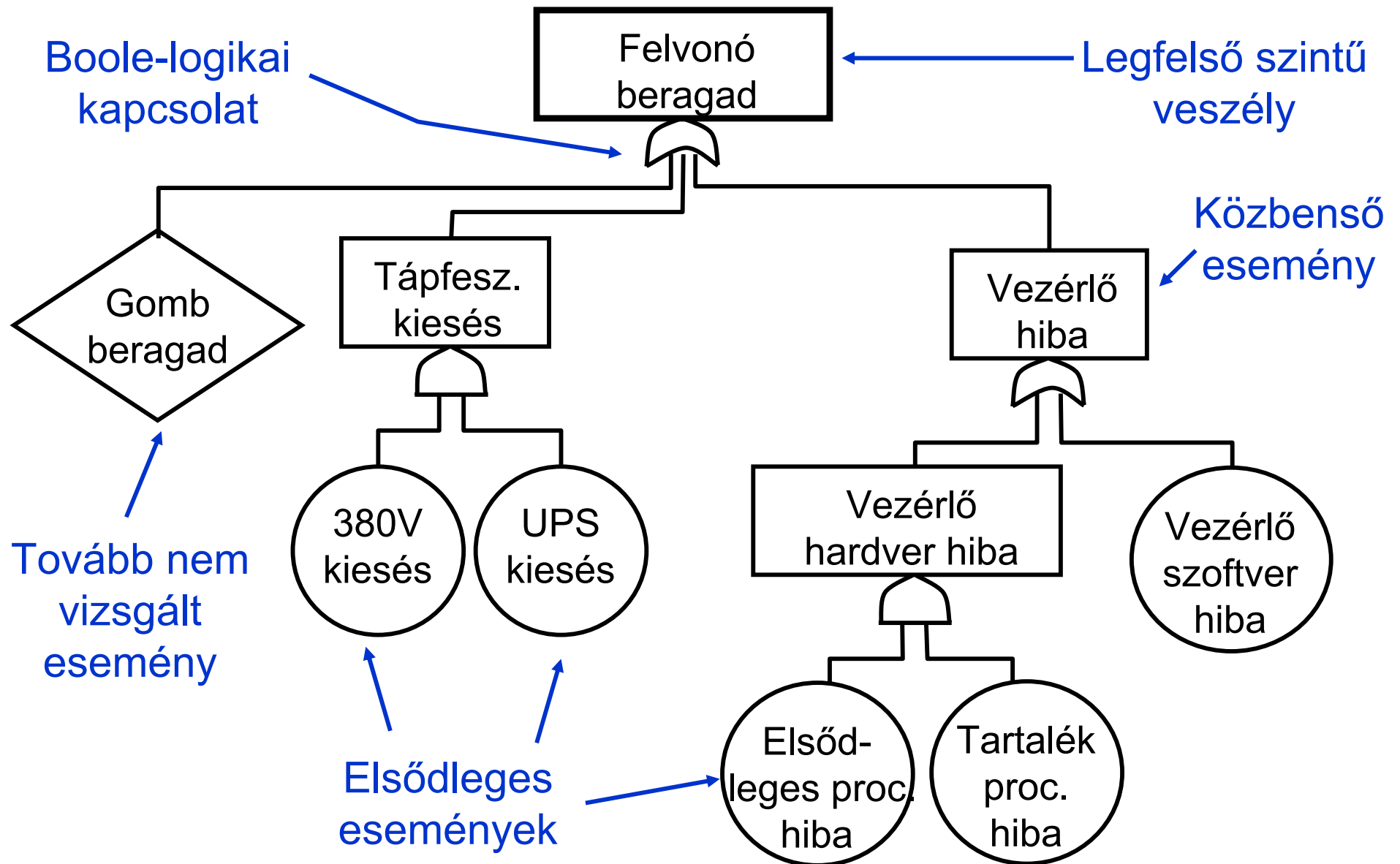


ÉS kapu



VAGY kapu

Hibafa példa: Felvonó



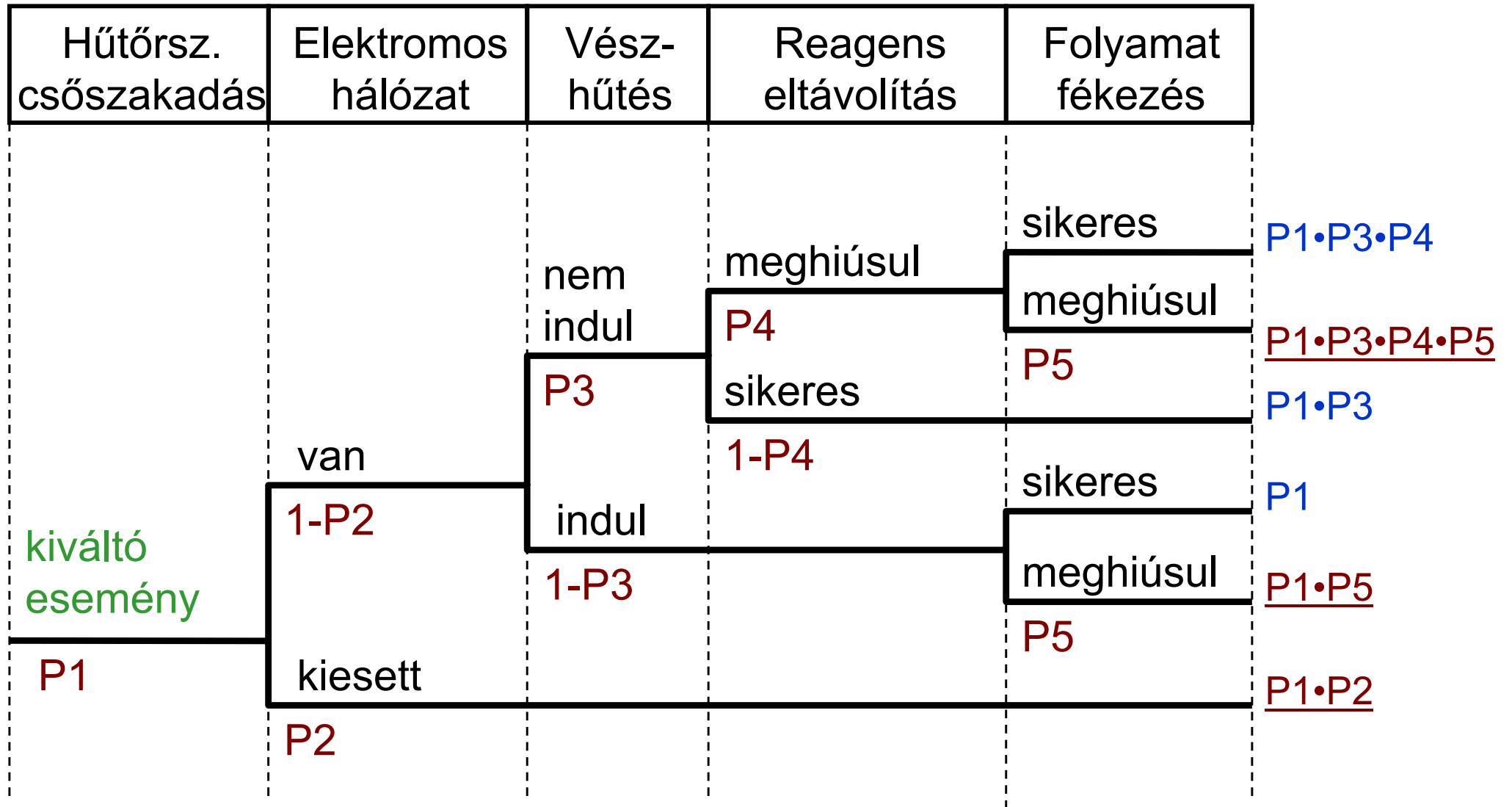
Hibafa analízis

- **Minőségi (kvalitatív) analízis:**
 - Hibafa **redukció**: Közbenső események feloldása
→ diszjunktív normál forma (OR a legtetején)
 - Azonosítható
 - Egyszeres hibapont (SPOF)
 - Kritikus esemény (több ágban 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

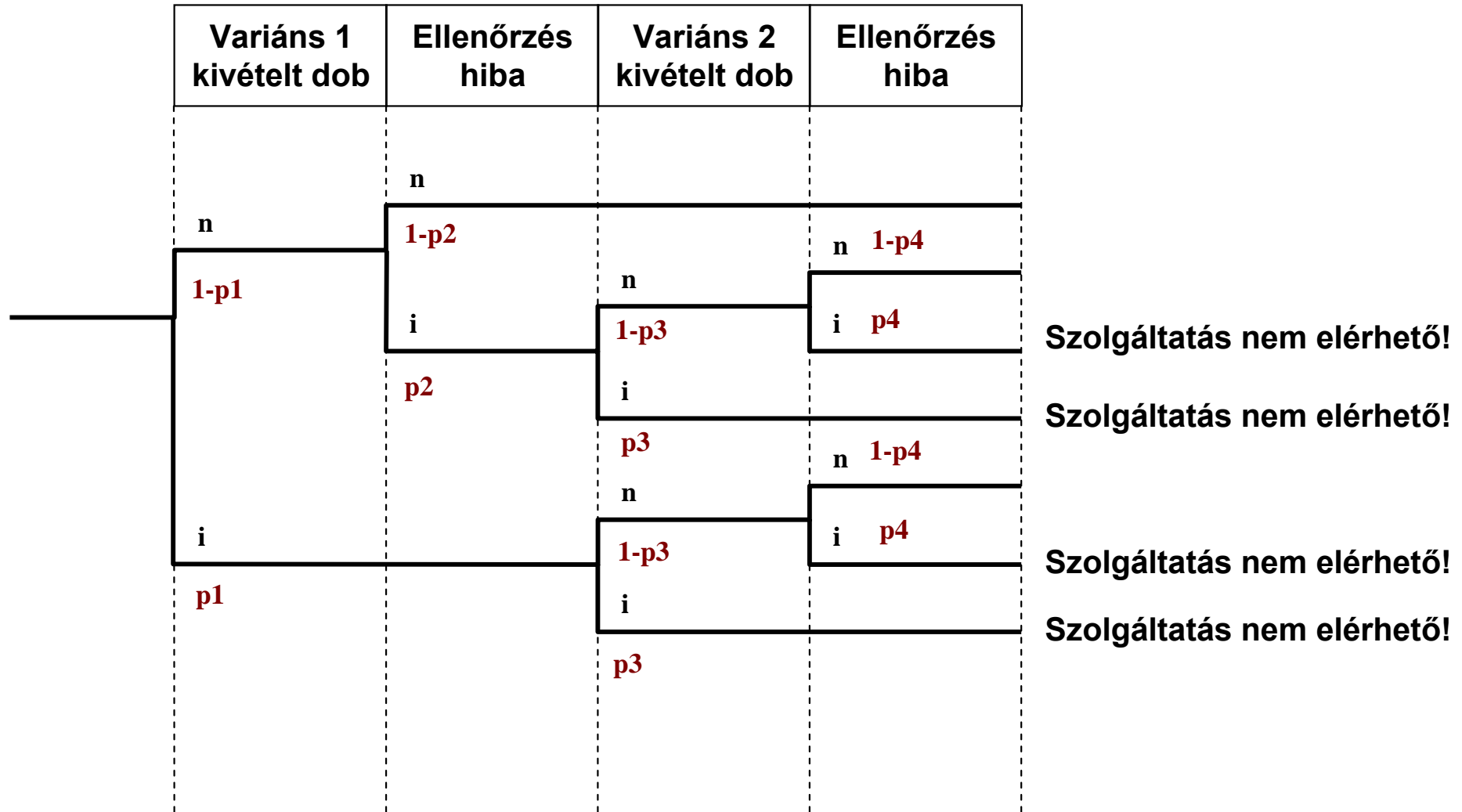
2. Eseményfa analízis

- **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: Reaktorhűtés



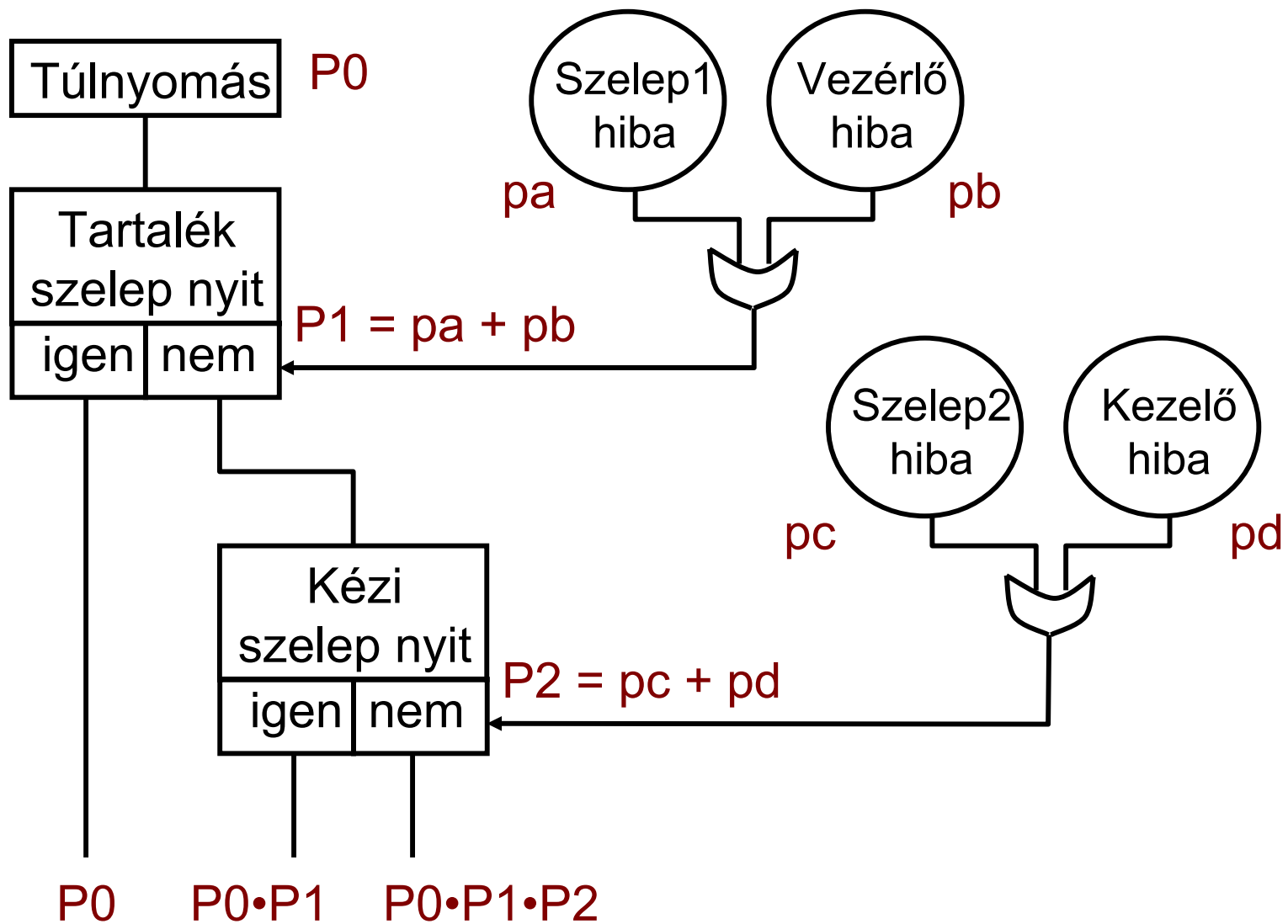
Eseményfa példa: Helyreállító blokkok



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 áttekintése

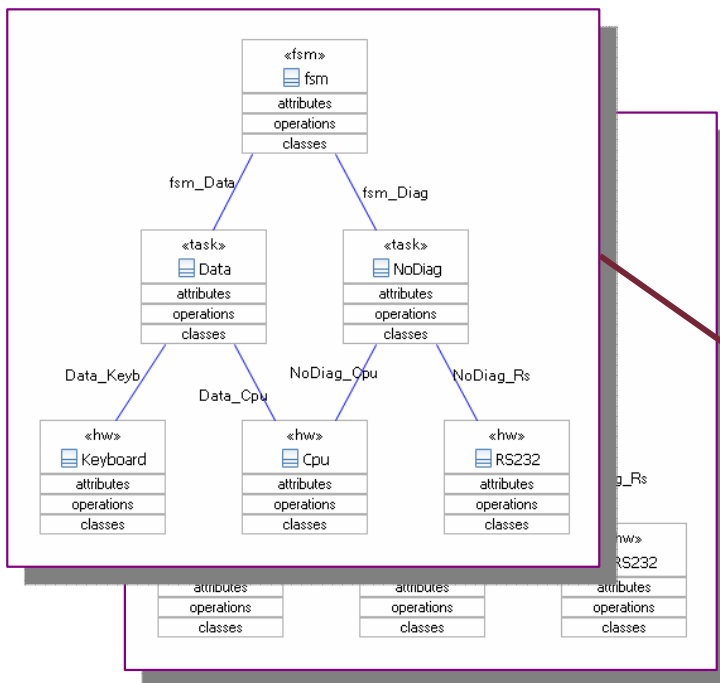
| | Megbízhatósági modell | Teljesítmény modell | Biztonsági modell |
|---------------------------------------|---|--|---|
| Komponens paraméterek | Meghibásodási tényező, lappangási idő, javítási tényező, hibafedés, ... | Funkció lokális végrehajtási idő, taszk prioritás, processzor ütemezés | Veszély 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 áteresztőképesség, 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, készlet, MTTF, MTTR, MTBF | Kiszolgálási idő, taszk áteresztő-képesség, processzor kihasználtság | Rendszerszintű veszély gyakoriság |

UML alapú megbízhatósági modellezés

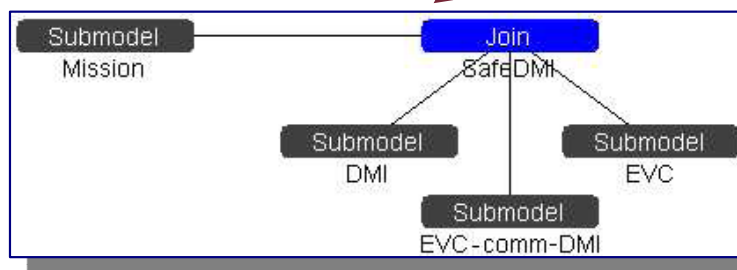
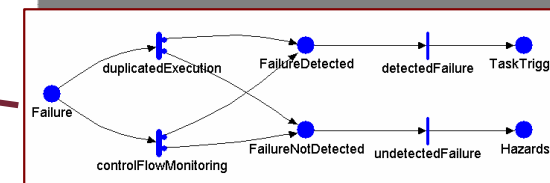
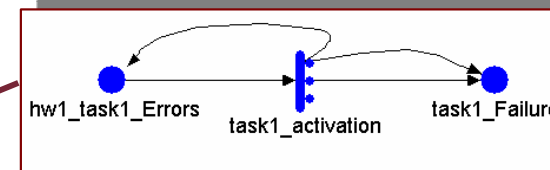
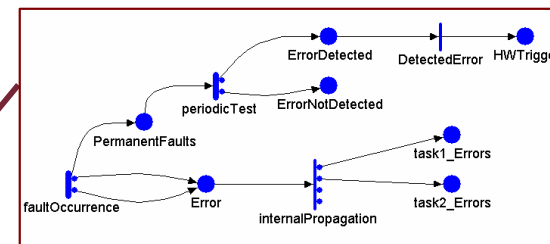
UML architektúra modell



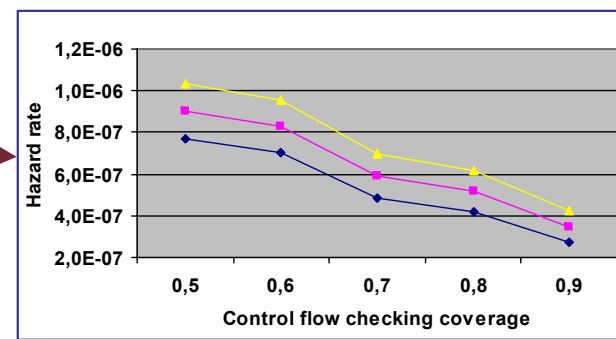
Megbízhatósági modell konstrukció



Analízis alhálók



Rendszerszintű megbízhatósági modell (sztochasztikus aktivitás hálózat)



Analízis eredmények

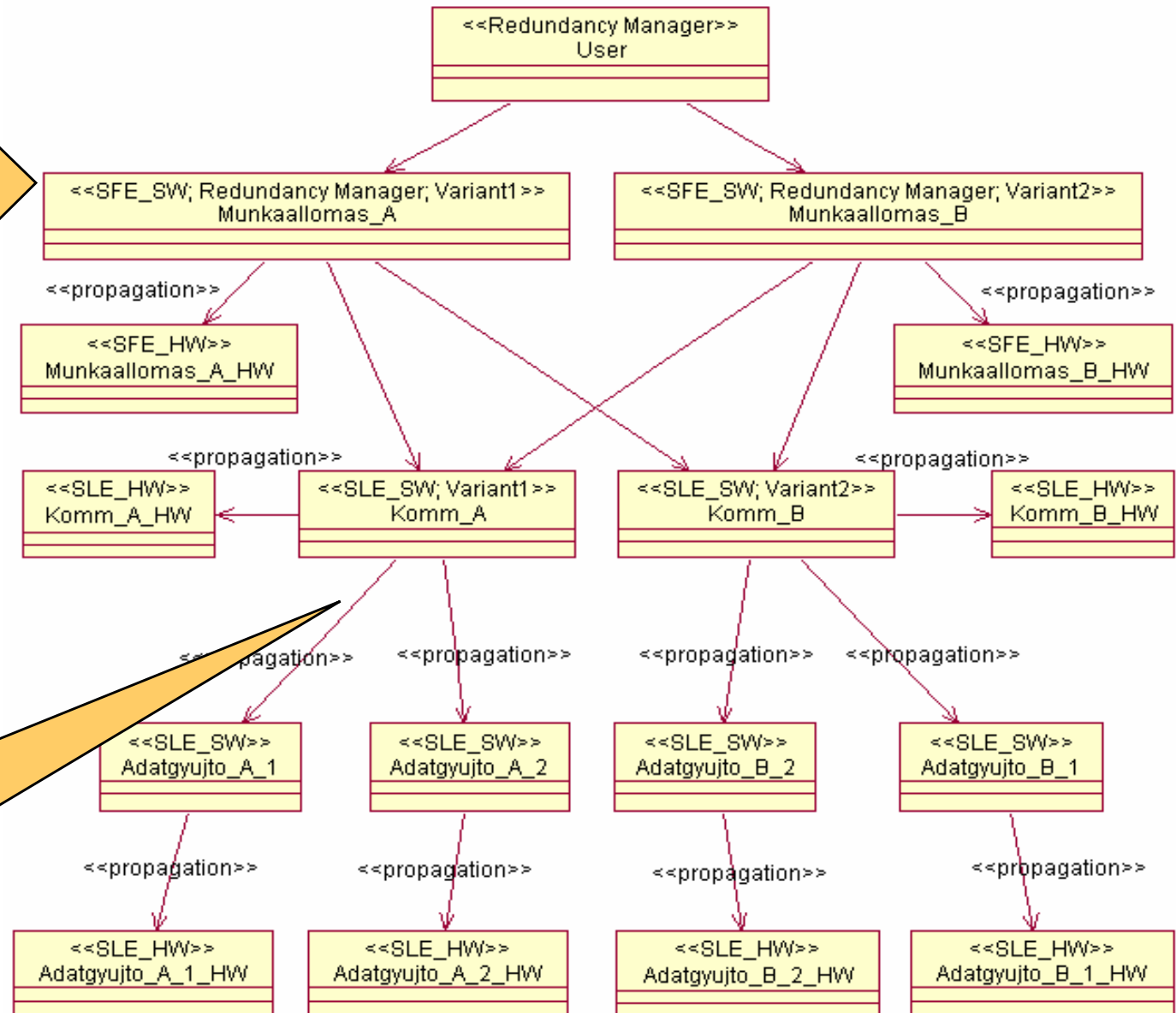
Megbízhatósági modellezés – Tervezői modell

Komponens:

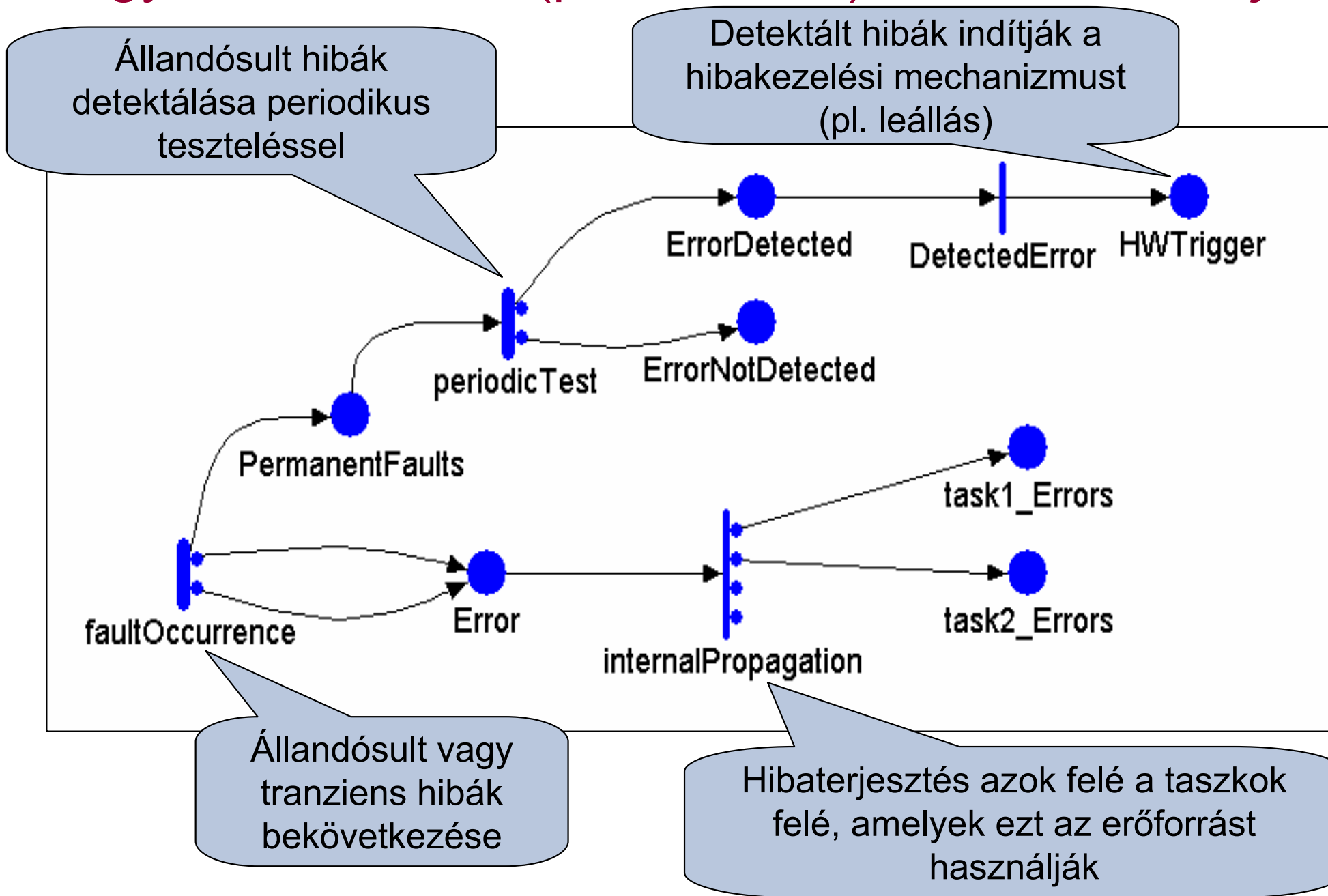
- Típus (HW, SW)
- Szerep (variáns, red. menedzser)
- Meghibásodási jellemzők:
 - * meghibásodási gyakoriság,
 - * lappangási idő,
 - * javítási idő

Kapcsolat:

- Hibaterjesztési jellemzők:
 - * hibaterjesztési valószínűség

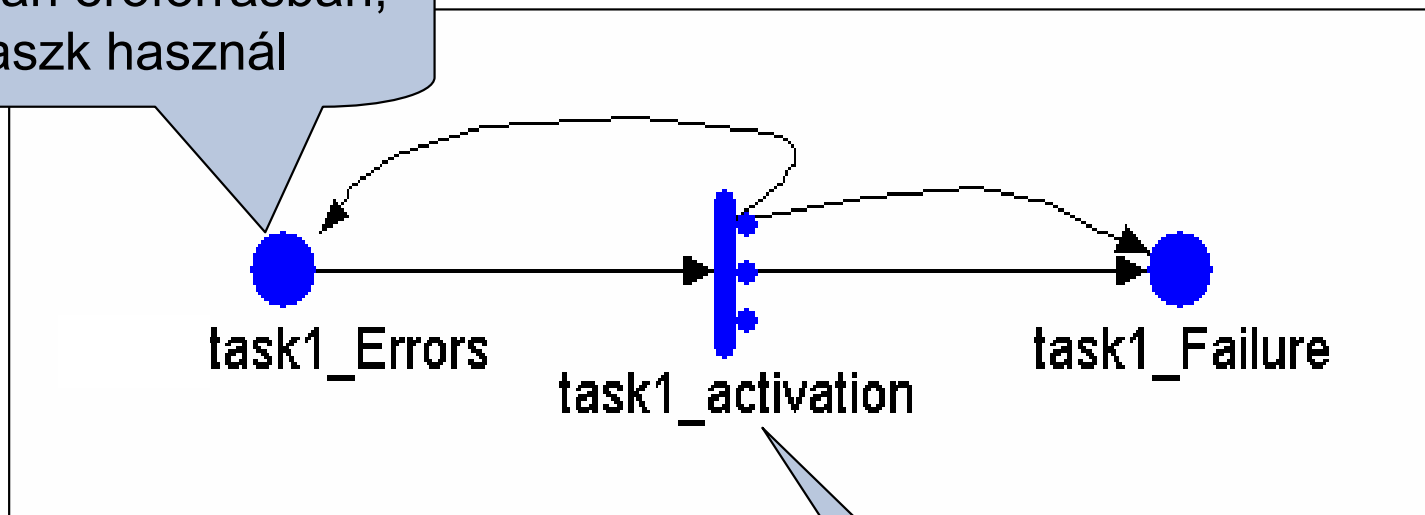


Egy HW erőforrás (pl. memória) analízis modellje



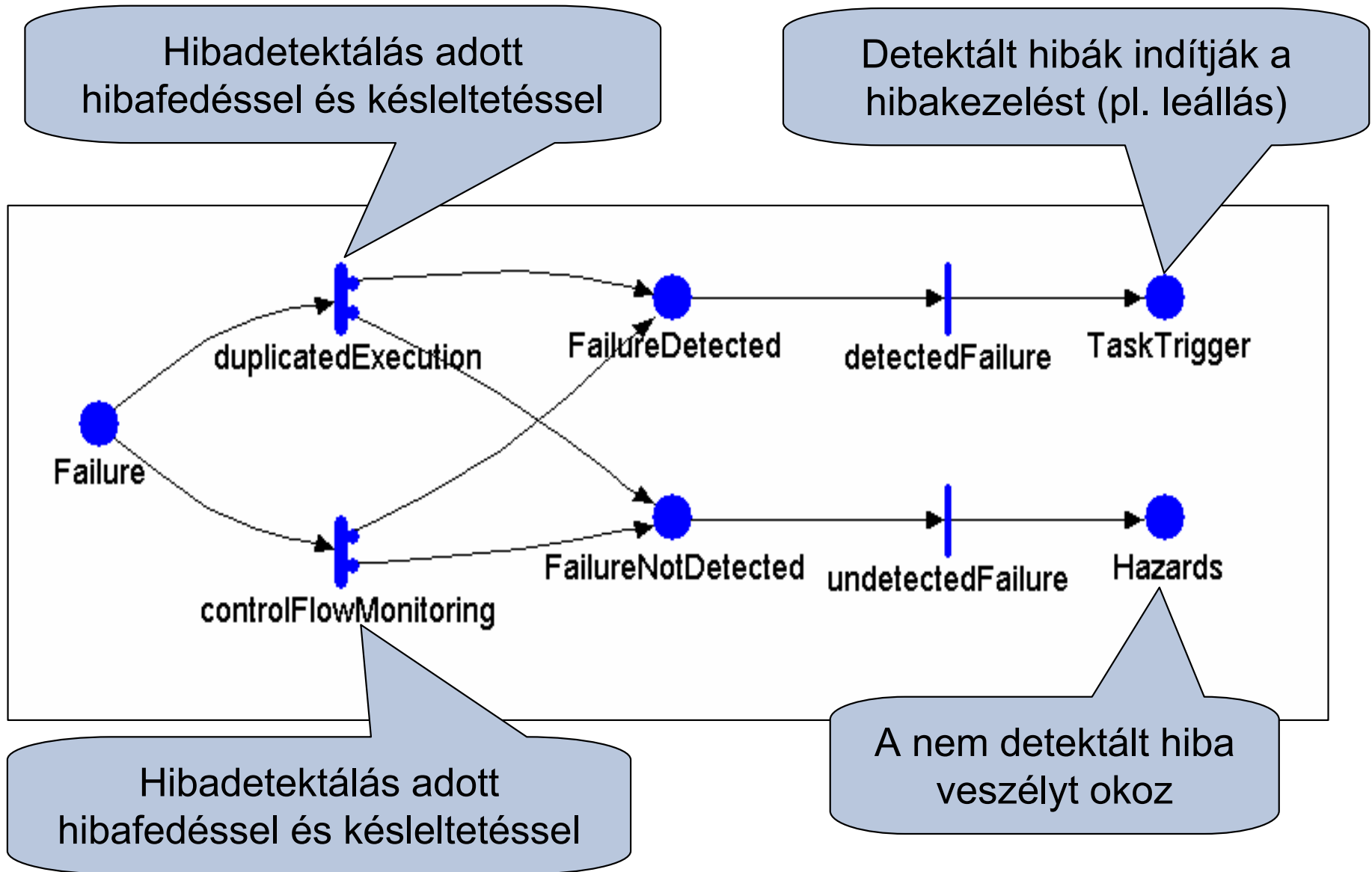
A hibaterjesztés analízis modellje

Hiba olyan erőforrásban,
amit a taszk használ



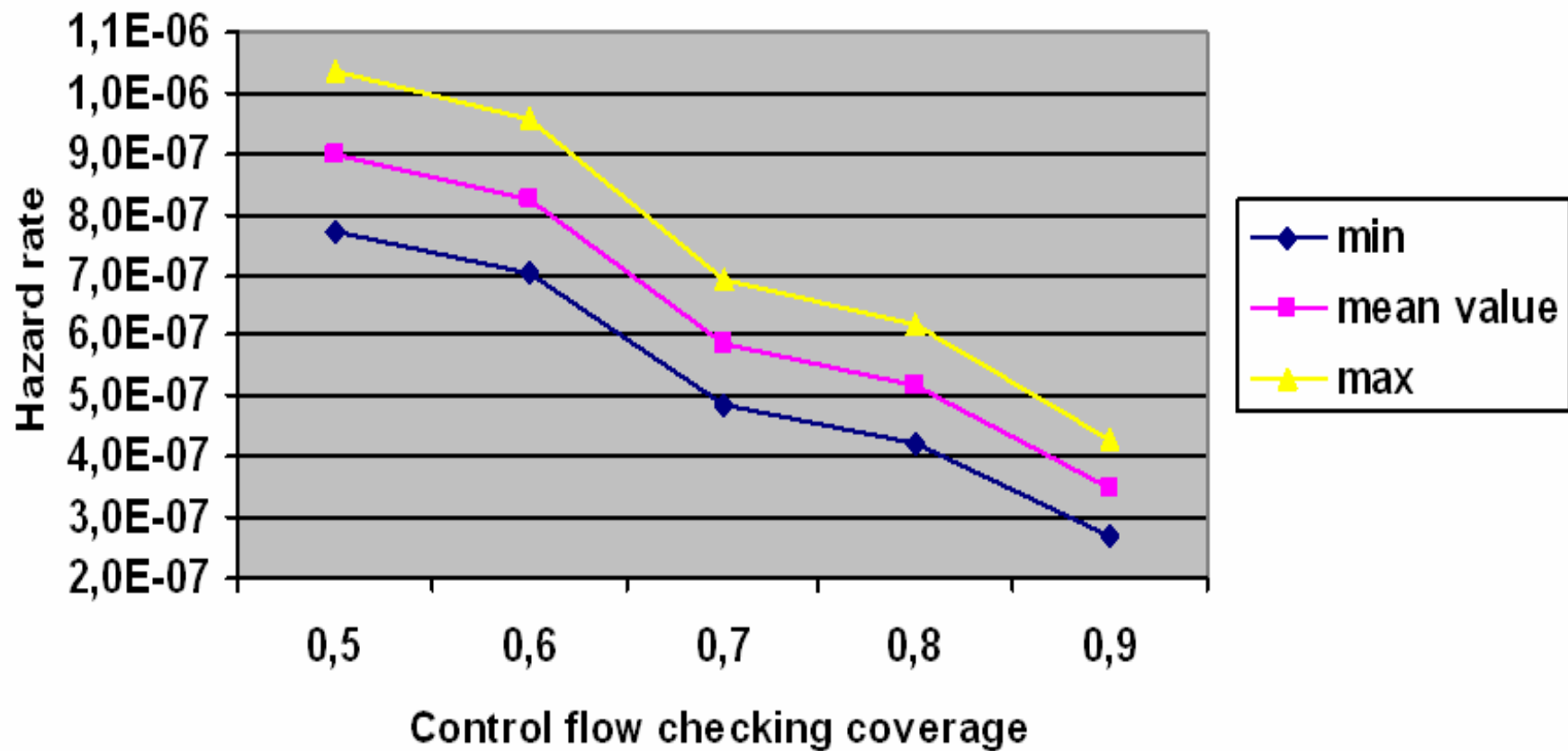
- Taszk aktiválási gyakoriság,
hiba aktiválási lehetőségek:
- aktivált hiba,
ami a rendszerben marad
 - aktivált hiba, felülíródik,
de hatása van
 - hatás nélkül felülírt hiba

Egy szoftver taszk analízis modellje

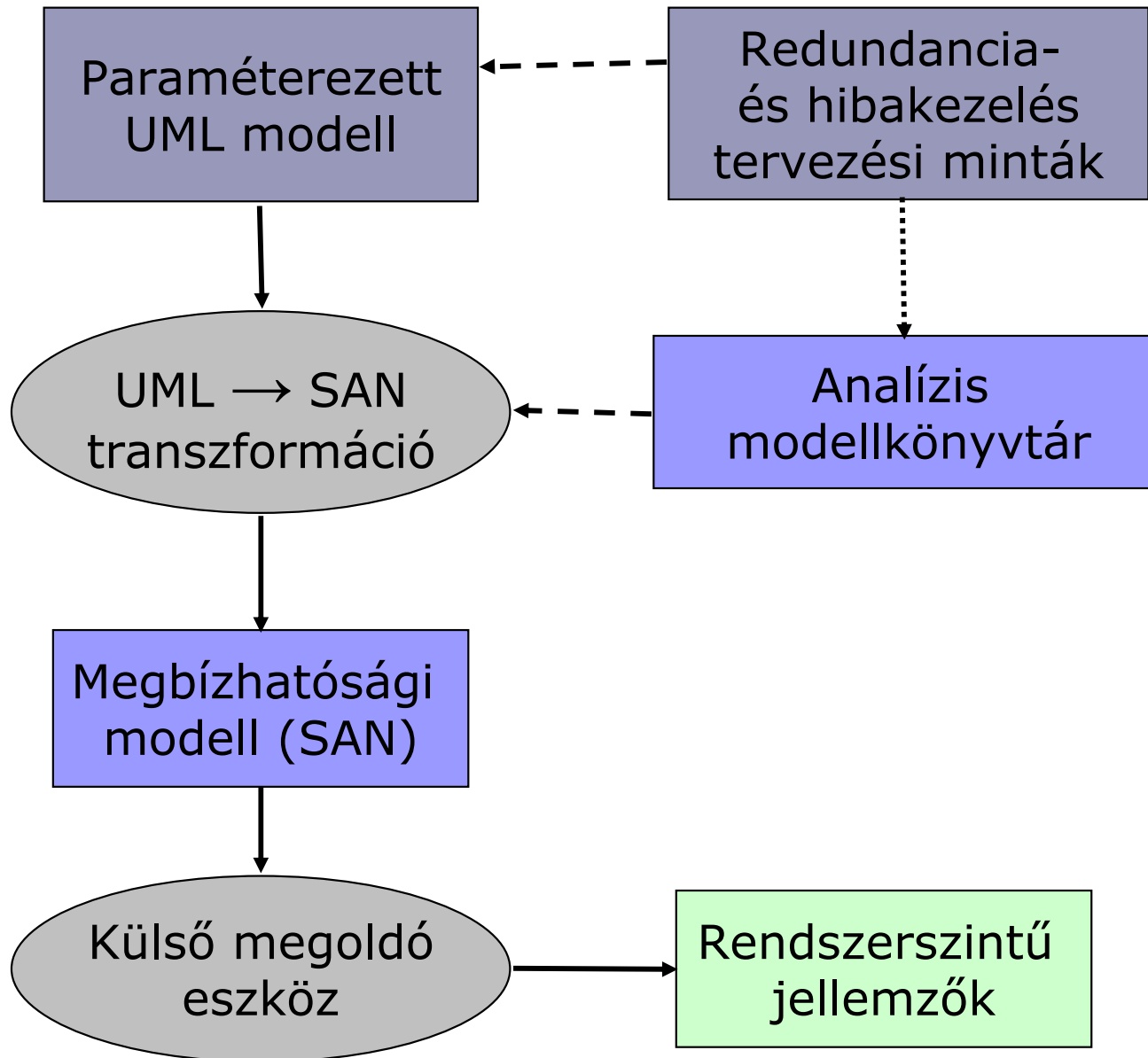


Analízis eredmény (példa)

- Ha a hibadetektálás hibafedése 50% alá csökken, akkor a SIL2 követelmény ($10^{-7} < \text{THR} < 10^{-6}$) 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, készlet, MTTF, MTTR, MTBF | Kiszolgálási idő, áteresztő-képesség, processzor kihasználtság | Rendszerszintű veszély gyakoriság |

Teljesítmény modellezés (LQN)

Taszok (szerver):

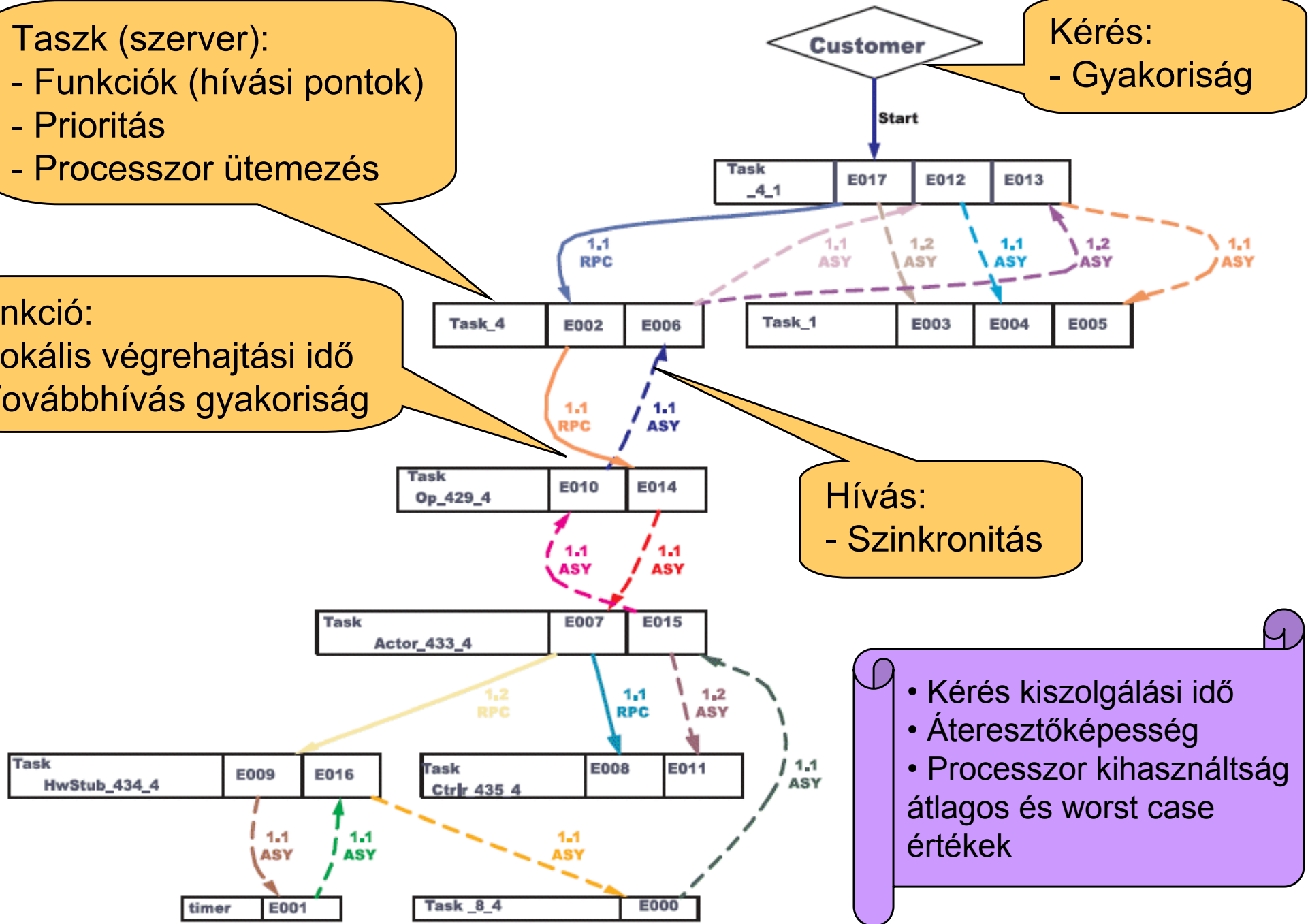
- Funkciók (hívási pontok)
- Prioritás
- Processzor ütemezés

Funkció:

- Lokális végrehajtási idő
- Továbbhívás gyakoriság

Kérés:

- Gyakoriság

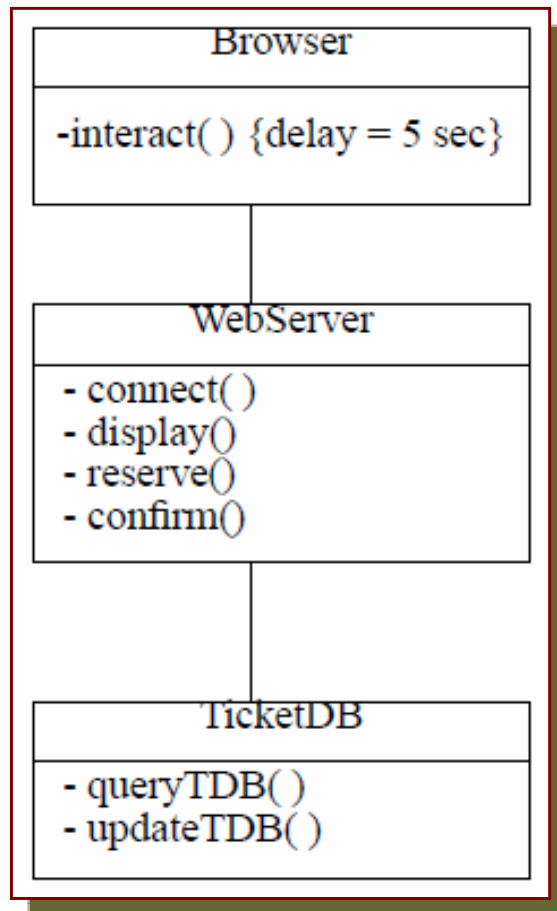


Hívás:

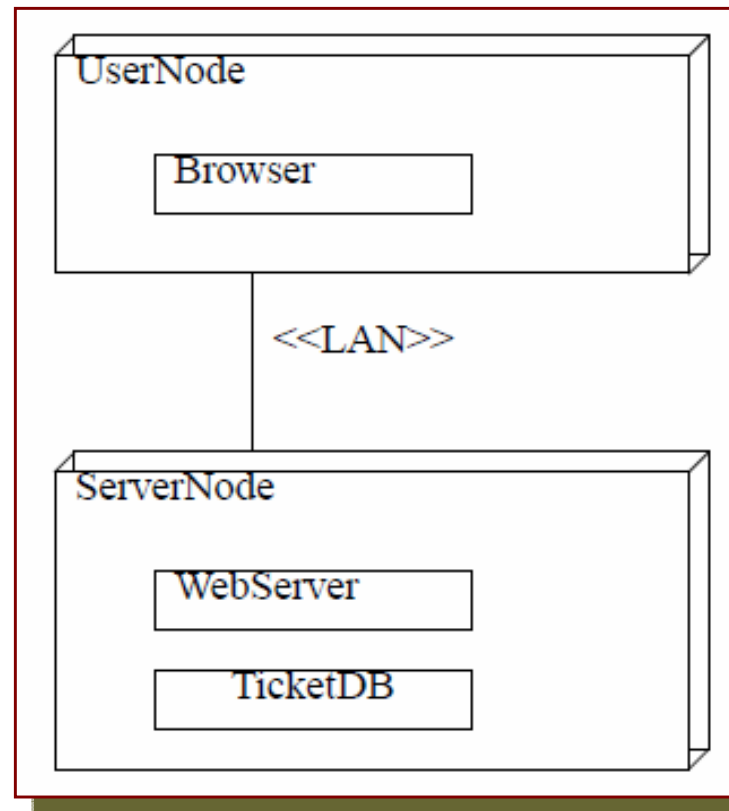
- Szinkronitás

- Kérés kiszolgálási idő
- Áteresztőképesség
- Processzor kihasználtság átlagos és worst case értékek

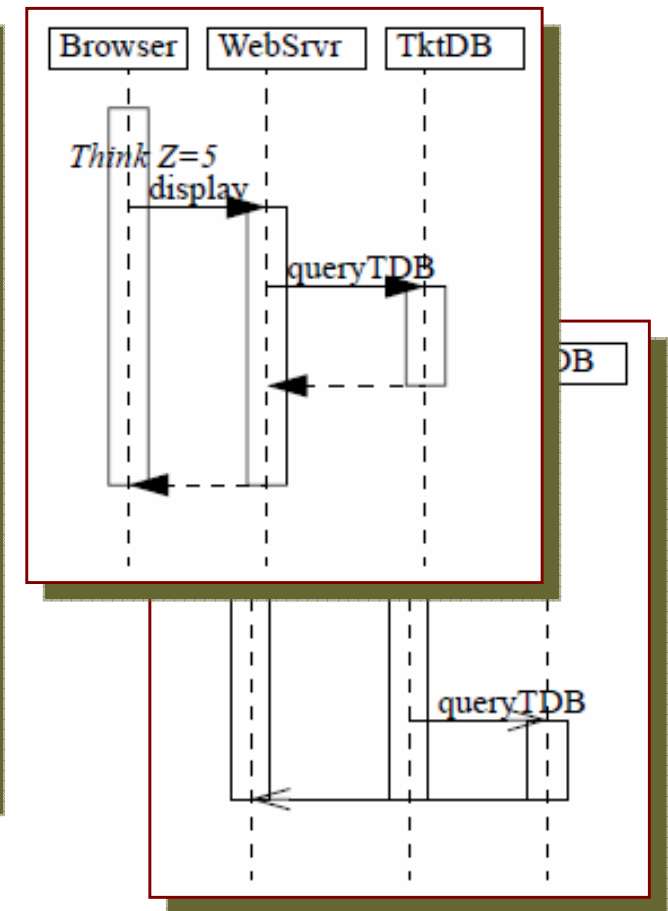
Az architektúra leképezése teljesítménymodellelre



Osztályok



Telepítés

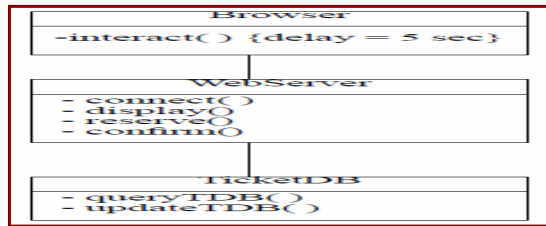


Interakciók

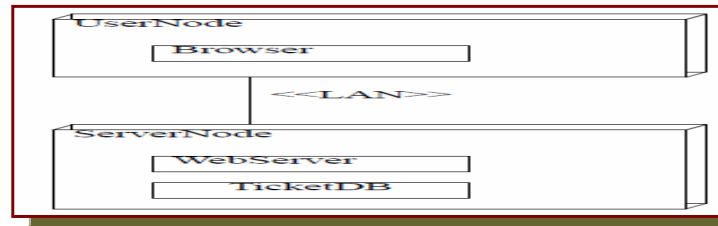
Lokális teljesítmény paraméterek megadása:

- Attribútumok (konvenció alapján értelmezhető)
- UML tagged value

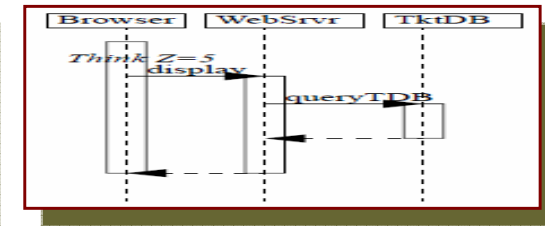
Az architektúra leképezése teljesítménymodelle



Osztályok

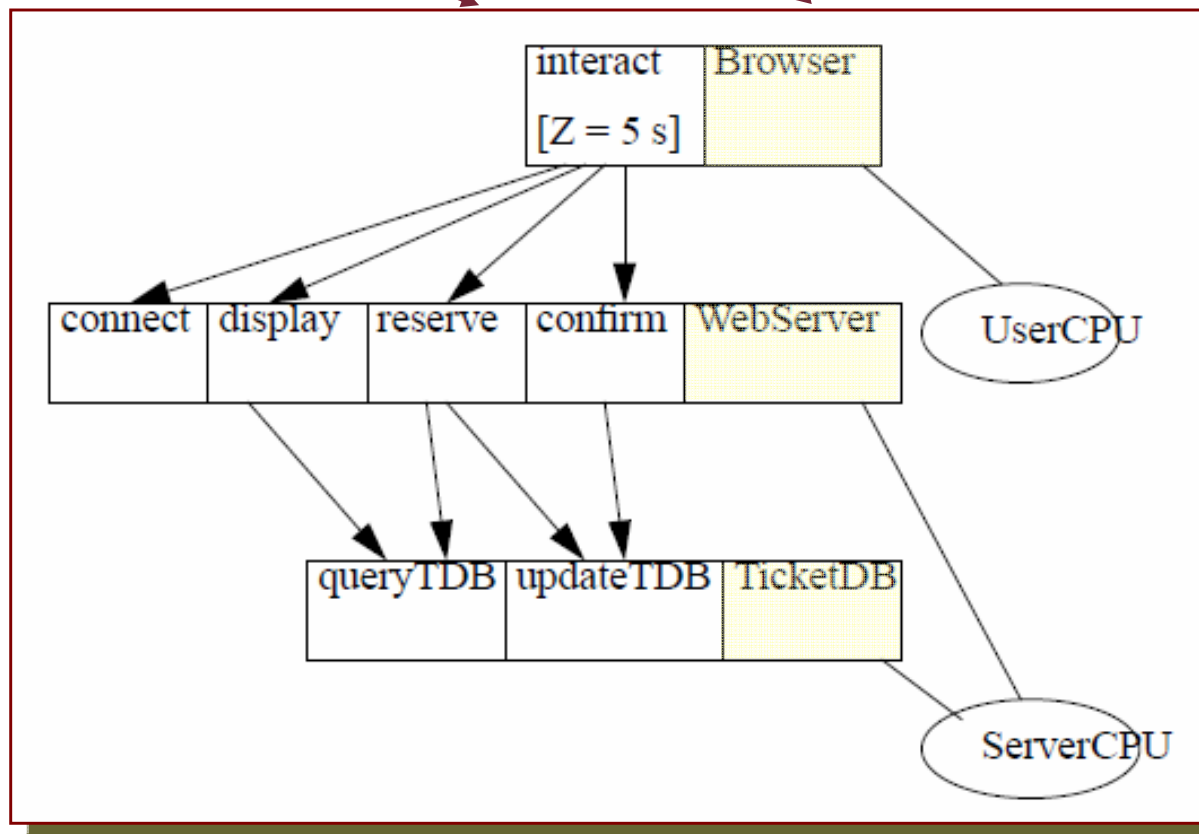


Telepítés



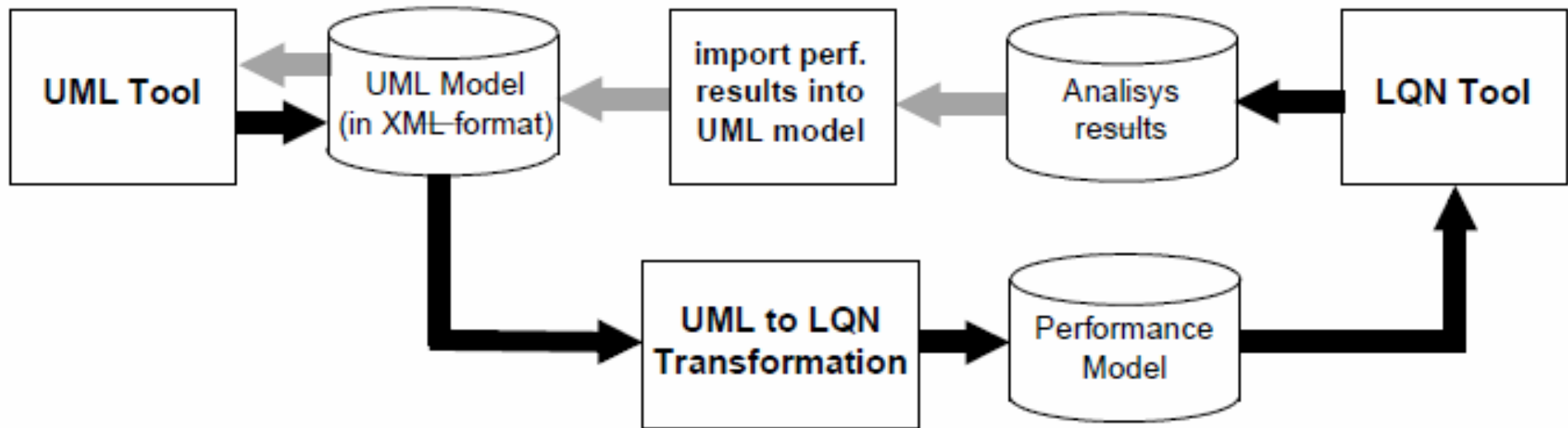
Interakciók

Modell-
transzformáció



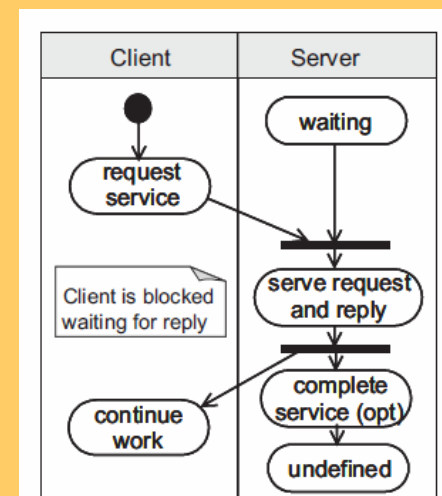
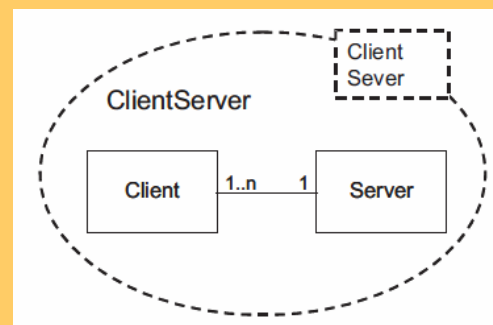
LQN
teljesítmény-
modell

Az architektúra leképezése teljesítménymodelle



- 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 megfelelés, 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 fázistól!
 - Integrációs tesztek
 - Rendszer tesztek

Összefoglalás

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