Követelménykezelés
Specifikáció készítés
A specifikáció ellenőrzése

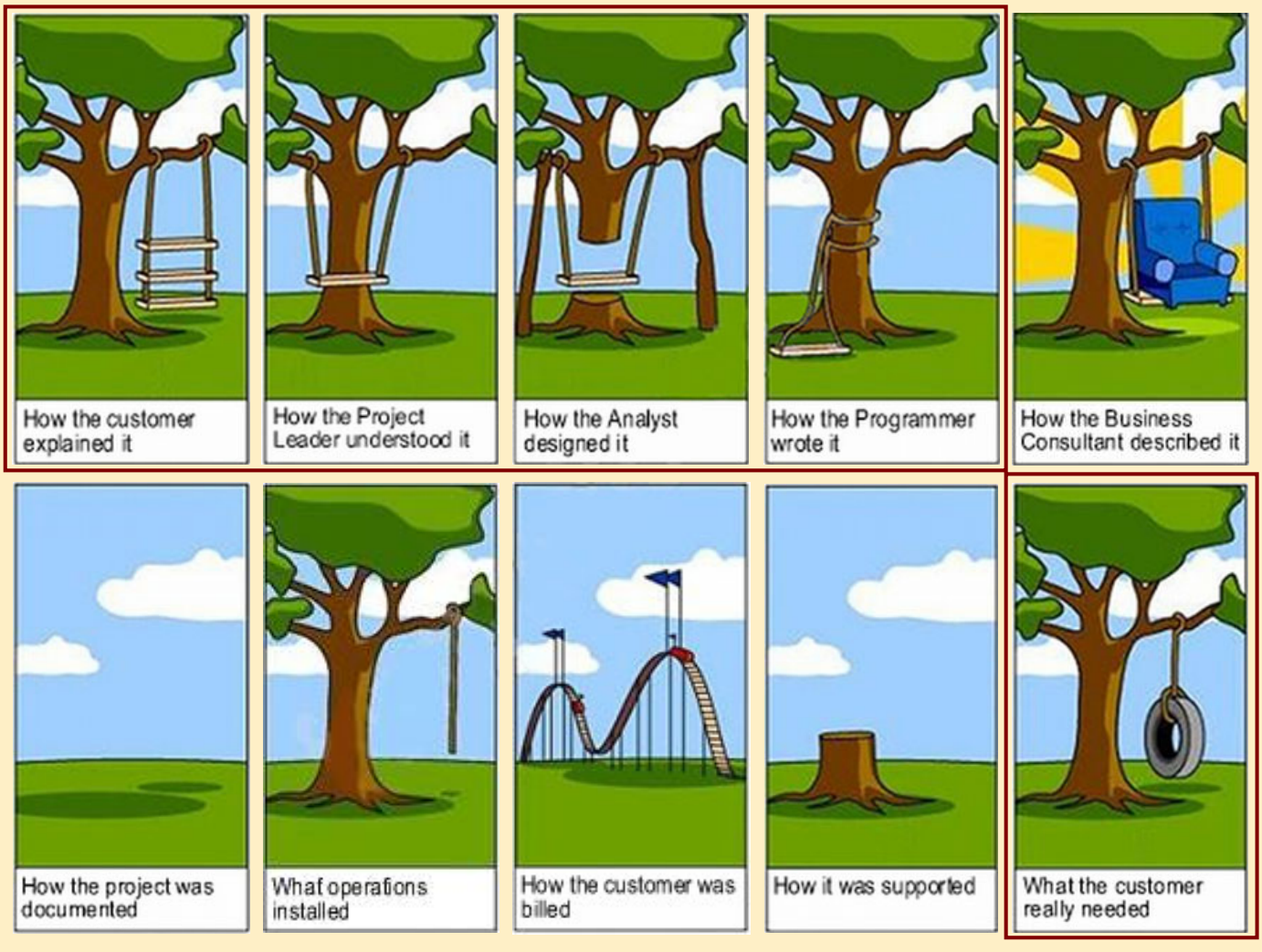
Majzik István
Egyes ábrák: Pap Zsigmond, Polgár Balázs

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

Tartalomjegyzék

Motiváció

- Tapasztalat: Sok hiba visszavezethető hiányos vagy ellentmondásos specifikációra
  - Példa: Meta Group felmérés, 2003:
    - Az IT projekt kudarcok 60%-70%-a a nem kielégítő követelményelemzésre vezethető vissza
  - Példa: 203 szoftverfejlesztési projekt utólagos felülvizsgálata
    - „An analysis of defect densities found during software inspections” (Journal on Systems Software)
    - Gyakoribbak a hibák a specifikáció elkészítésének fázisában, mint a későbbi, implementációs fázisokban
  - Példa: Voyager és Galileo űrszondák szoftver tesztelése során felfedezett hibák okainak elemzése
    - 78% (149/192) specifikációs hiányosság, ebből
    - 23% veszélyes állapotban ragadás (nincs kilépés)
    - 16% időzítési kényszerek megadásának hiánya
    - 12% nincs specifikált reakció külső eseményre
    - 10% bemeneti érték ellenőrzésének hiánya



Elvárások, követelmények és specifikáció

- Követelmény (requirement):
  - Bejövő igény, vízió, elvárás
    - Felhasználóktól (user)
    - Érdekeltektől (stakeholder: hatóság, vezetőség, operátor, ...)
  - Validáció alapja
- Specifikáció (specification, requirement specification, system requirements):
  - Tervezők, fejlesztők felé átalakított elvárások
  - A követelményelemzés (absztrakció, strukturálás, szűrés) eredménye
  - Sokféle típus
    - Rendszerspecifikáció, architektúra specifikáció, tervspecifikáció
  - Verifikáció alapja

Elvárások a specifikációval szemben

- A követelmények teljes lefedése
  - Fukcionális követelmények
  - Extra-funkcionális követelmények
- Megfogalmazás: Egyértelmű, igazolható, megvalósítható
- Javasolt megoldások:
  - Szigorú specifikációs nyelv (pl. formális)
  - Ellenőrzött (tervezési illetve specifikációs) minták használata
  - Utólagos ellenőrzés
- Példa: EN 50128 szabvány által adott lehetőségek
  - Formális módszerek (VDM, Z, B, TL, PN, ...)
  - Félformális módszerek (diagram alapú technikák, UML)
  - Strukturált metodika (JSD, SADT, SSADM)
  - Emellett természetes nyelvű megadás is szükséges!

Tartalomjegyzék

- Motiváció
  - Miért fontosak a tervezési folyamat ezen szakaszai?
  - Milyen elvárások vannak a specifikációval szemben?
  - Milyen módszerei vannak a specifikáció készítésnek?
- Az általános követelménykezelés feladatai
  - Követelmények nyilvántartása
  - Követhetőség a verifikációhoz
- Félformális specifikáció
  - Specifikus technikák: SysML
- A követelményspecifikáció verifikációja
  - Általános kritériumok
  - Specifikus kritériumok UML állapottérképekre (mintapélda)

A követelménykezelés feladatai (áttekintés)

- Követelmények hatékony, strukturált tárolása
  - Hierarchikus elrendezés, tulajdonságokkal
- Követelmények finomítása és kapcsolódása
  - Specifikáció -> Rendszerterv -> Modulterv -> Forráskód -> Teszt
- Követelmény életciklus támogatása
  - Felvétel, törlés, változás, finomítás, kapcsolatok megjelenése
- Analízis lehetőségek
  - Hatás analízis (impact analysis): változáskezelés
    - Mit befolyásol, ha a követelmény megváltozik?
  - Eredet analízis (derivation analysis): költség-haszon elemzés
    - Milyen követelményre vezethető vissza? Miért van itt, szükséges-e?
  - Fedettség analízis (coverage analysis): projekt követés
    - Mely követelmények nincsenek implementálva?
    - Mely követelmények nincsenek tesztelve?

Követhetőség (traceability) szükséges

A követelménykezelés „kézi” módszerei

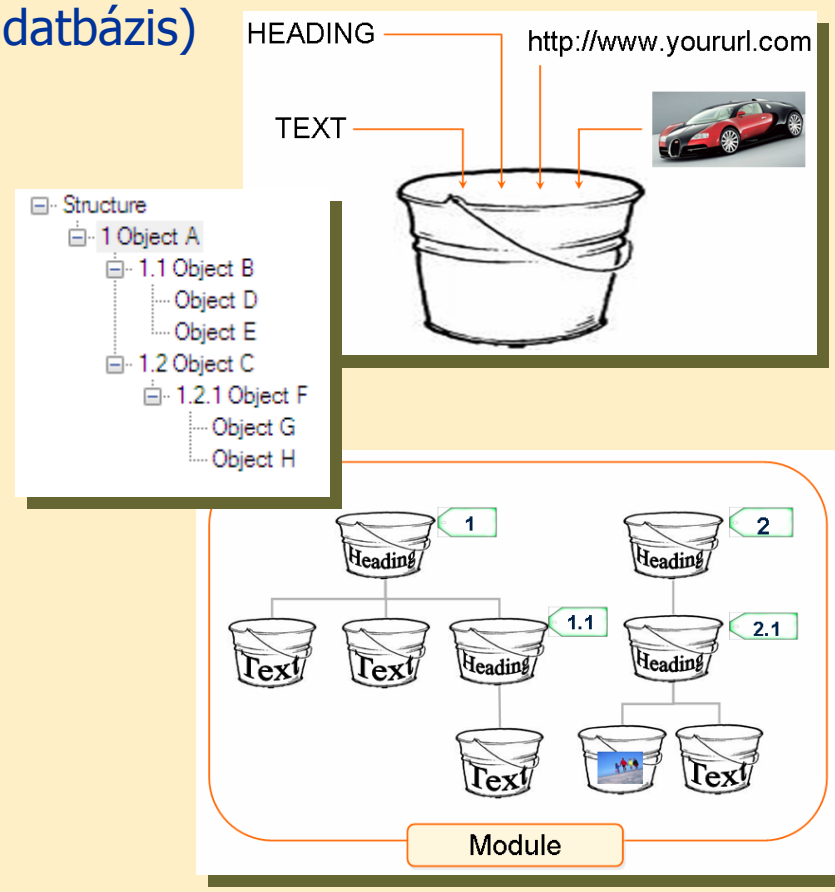
- Természetes nyelvű követelmény dokumentum
  - Strukturálás adott (fejezetek, alfejezetek)
  - Követelményazonosítók felvétele
- Követelményfinomítás: Táblázatos nyilvántartás
  - Követelményazonosítók szerepelnek
    - Különböző dokumentumokból (SRS, SA, MDS, MTS, ...)
  - Analízis makrókkal támogatható
    - Üres, többszörös, ... mezők kikeresése
- Követhetőségi mátrix: Táblázatos forma
  - Követelmények azonosítói
  - Kódrészlet azonosítók (funkció szint tipikus)
  - Teszt azonosítók
  - Sikeres/sikertelen teszteredmény bejelölése

Automatikus követelménykezelők feladatai

Követelmények nyilvántartása	Hierarchikus felépítés
Kapcsolatok nyilvántartása	Sokféle reláció: Kompozíció, származtatás, finomítás, bizonyítás, .. Követelmény – Modell – Kód – Teszt – Teszteredmény között
Követelmény változások kezelése	Időbeli struktúra, triggerek
Navigáció a kapcsolatokon	Előre: pl. hatás analízishez Vissza: pl. motiváció analízishez
Fedettségi listák készítése	Lefedetlen követelményekhez Indokolatlan megvalósításhoz
Jogosultságok kezelése	Hozzáférés szerepek
Értesítési rendszer	Változások
Biztonsági megoldások	Sértetlenség

Megvalósítás: Strukturált tárolás

- Objektum modell (adatbázis)
  - Általános „tároló”
  - Egyedi azonosító
  - Modulok
- Hierarchia
- Tulajdonságok
  - Fejléc / szöveg
  - Hozzáférési jogok
  - Történet
  - Attribútumok
    - Prioritás
    - Státusz
    - Költség
    - …
  - Linkek



Példa: DOORS

ID	Last Modified By	Car user requirements	Priority	Percentage cost	Comments
TRN-CSR-1	Bill Young	**1 Introduction**	Mandatory	0.172835	
TRN-CSR-2	Bill Young	This module contains the user requirements for a new car to be commercially available by 1 August 2006.	Mandatory		e form text a text field.
TRN-CSR-3	Bill Young	**2 User types**	Desirable	1.370889	
TRN-CSR-4	Bill Young	**2.1 Nationalities**	Mandatory	0.642687	
TRN-CSR-5	Bill Young	The car will be used in the following countries: UK, USA, Northern Europe, Eastern Europe, Japan, Russia, Australia.	Mandatory	0.769025	
TRN-CSR-6	Bill Young	**2.2 User sizes**	Mand		
TRN-CSR-7	Bill Young	People come in all shapes and s es. The car must be suitable for people maximum and minimum sizes fgfgf to 2 m weighing 25 kilograms to	Mand		

Tulajdonságok

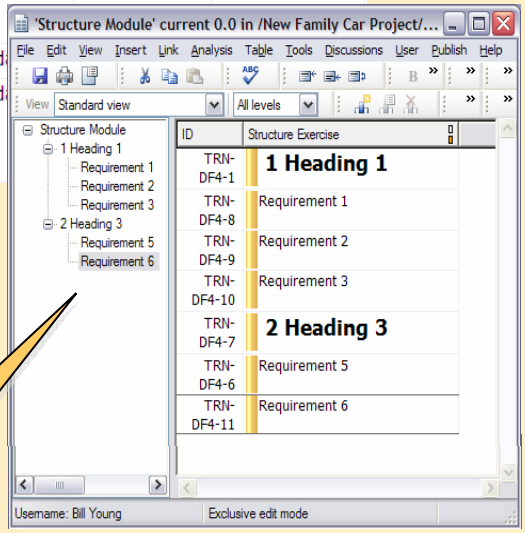
Objektum azonosító

Változás-jelző

Fejléc objektum

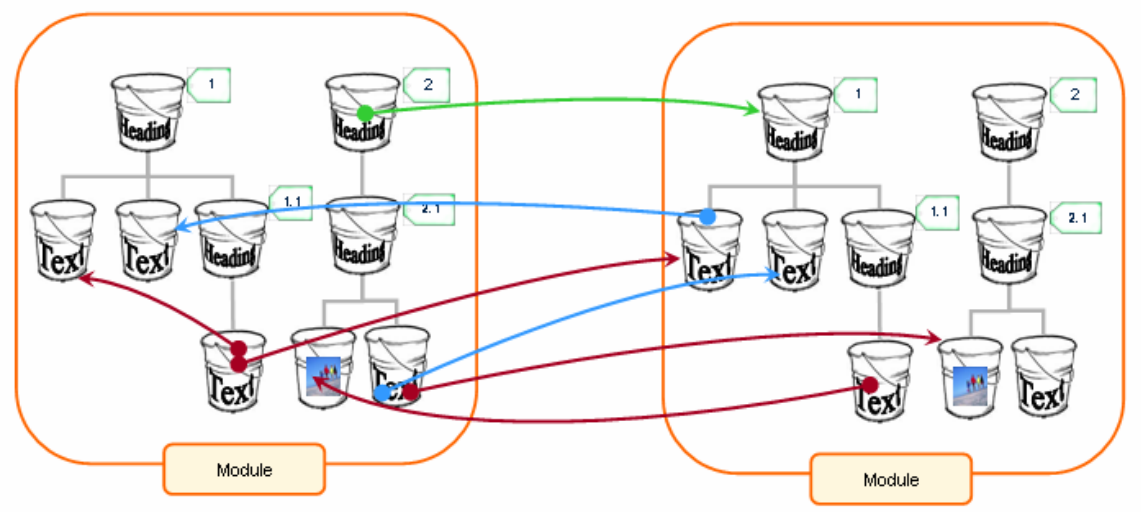
Szöveges objektum

Hierarchia

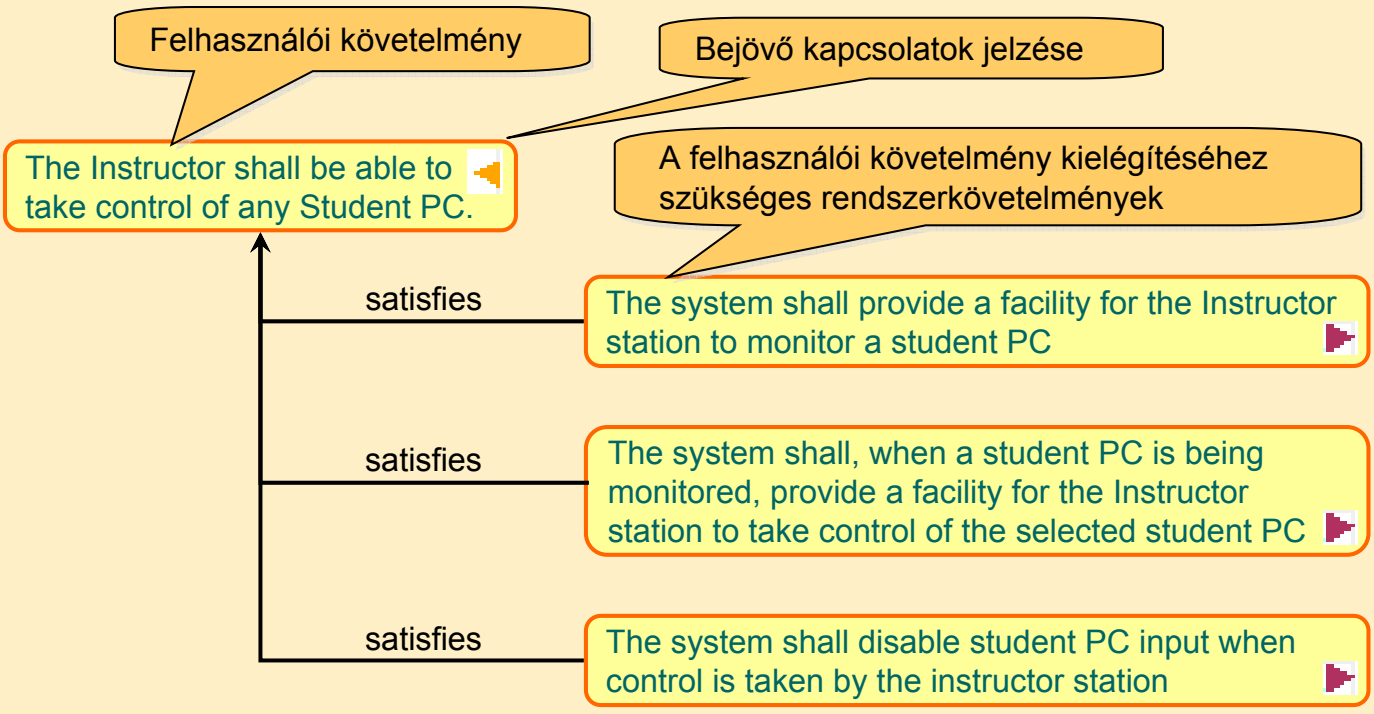


Megvalósítás: Kapcsolatok (linkek)

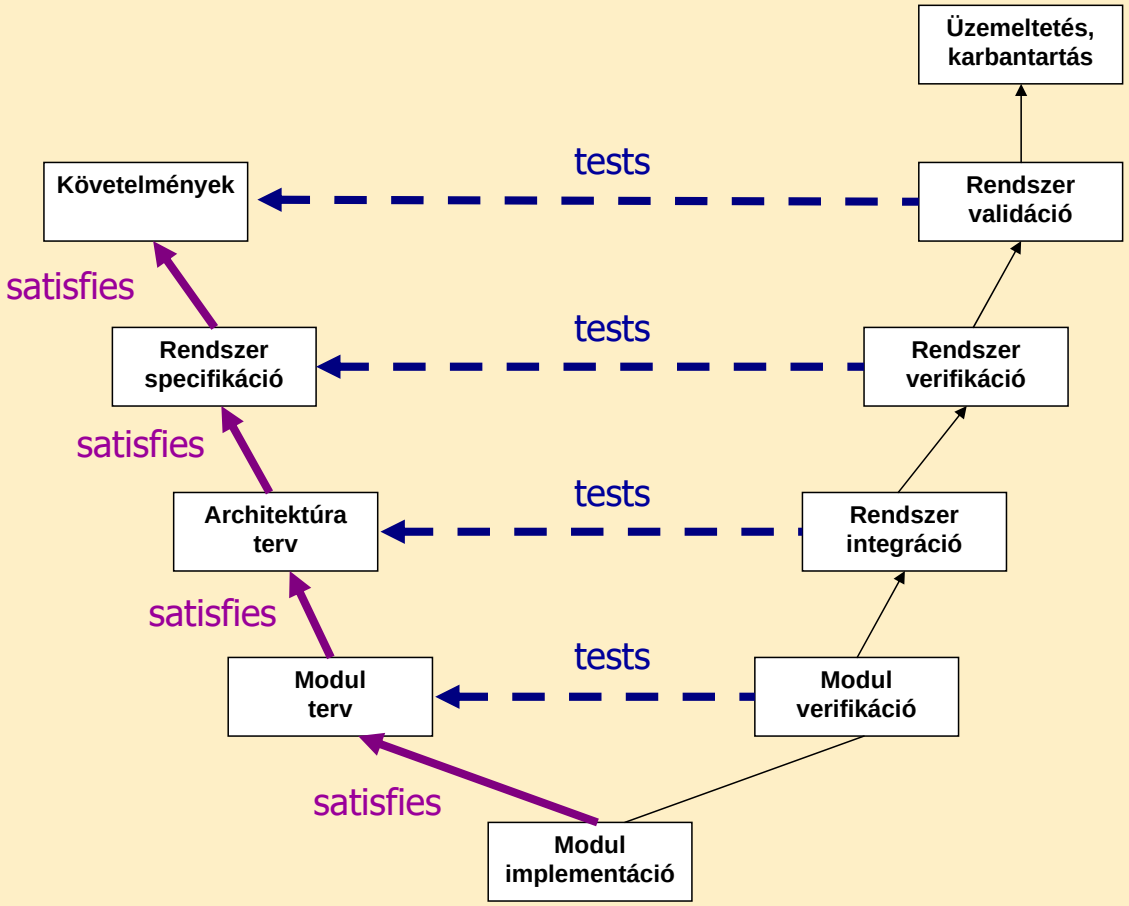
- Relációk: Rendezett párok
  - Objektumok között
  - Külső kapcsolatok
- Típusok
  - Finomítja
  - Kielégíti
  - Teszteli
  - ...



Példa: DOORS



Relációk a V-modellben



Megvalósítás: Követelmény életciklus

- Objektum szintű változások
  - Közvetlen szerkesztés (létrehozás, módosítás, törlés)
  - Megváltozott objektum kapcsolatok jelzése (suspect link)
  - Változtatási javaslatok menedzselése (elkészítés, csoportosítás, felülvizsgálat, érvényre juttatás)
- Változások mint események
  - Scriptek indítására használhatók
- Nagyléptékű változások
  - Baseline definiálás (állapot rögzítése)
    - Inkrementális változások vizsgálhatók
    - Összehasonlítás lehetséges
  - Követelmény-partíciók kiajánlása, távoli szerkesztés, szinkronizálás, visszavétel

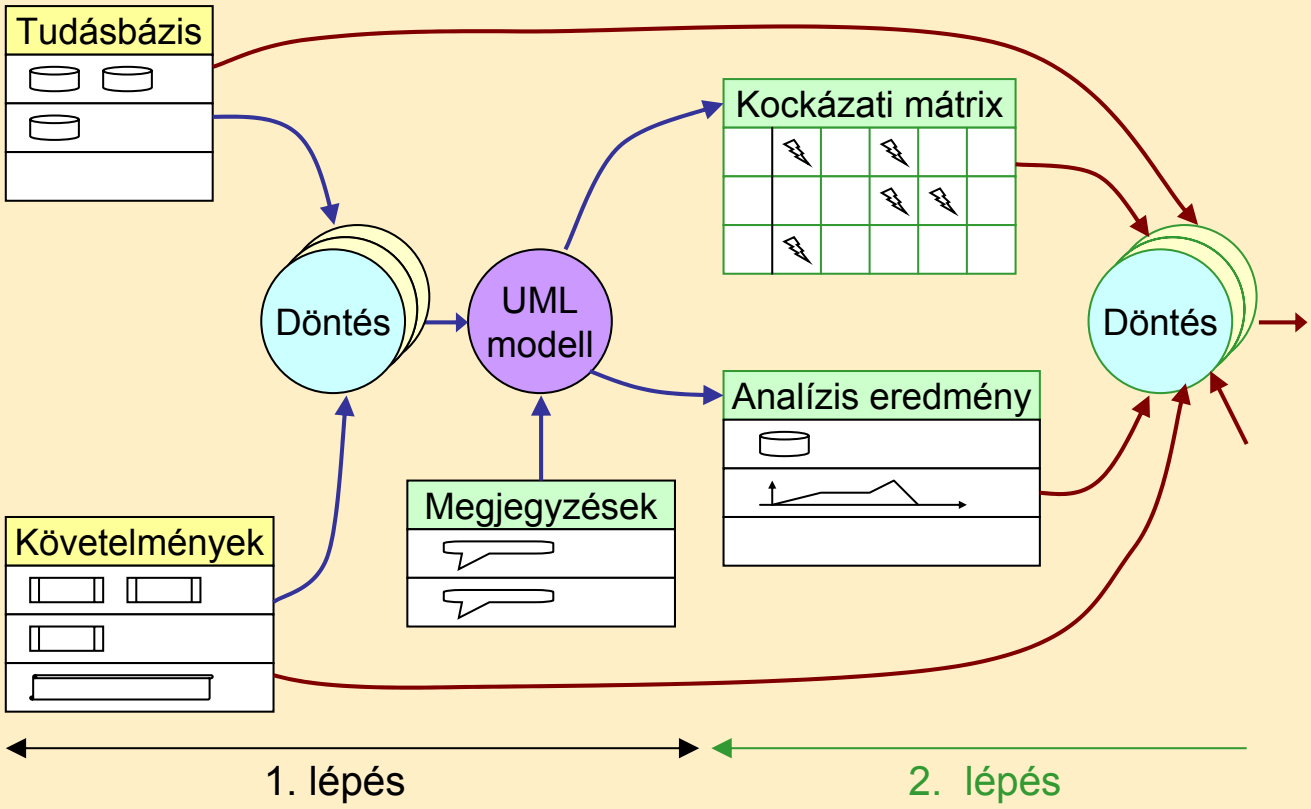
Megvalósítás: Analízisek

- Követhetőség alapja:
  Navigálás a kapcsolatokon keresztül
  - Kiterjedés (scope) és mélység kijelölhető
  - Irány kijelölhető (előre, hátra)
- Script nyelv használható
  - Bejárás, kigyűjtés
  - Tulajdonságok megváltoztatása
- Jelentések készítése
  - Hatás analízis: Előre navigálás alapján
  - Eredet analízis: Hátra navigálás alapján
  - Fedettség analízis
    - Szűrés: Navigálás kód objektumokig, teszt objektumokig
    - Objektumok kigyűjtése: Nincs kapcsolat adott célhalmazig
      - Pl. nincs megvalósítás, nincs sikeres teszt

Megvalósítás: Járulékos funkciók

- Nézetek az objektum listában
  - Szűrés hierarchiaszintekre, tulajdonságokra, …
- Űrlapok a tulajdonságok szerkesztésére
- Megosztott szerkesztés (csoportmunka)
  - Objektum (követelmény) szintű zárolás
- Dokumentáció generálás
  - Hierarchia (=> fejezetek) alapján rendezett szöveges és egyéb objektumok
  - Exportálás: Strukturált külső dokumentumban (pl. Word) történt változtatások vissza is olvashatók (struktúra megőrzése)
  - Importálás: Előkészítés szükséges
- Webes hivatkozások (pl. e-mailben küldhető)
  - Adatbázis, projekt, objektum hivatkozás

Egy más aspektus: Tervezői döntés adatbázis



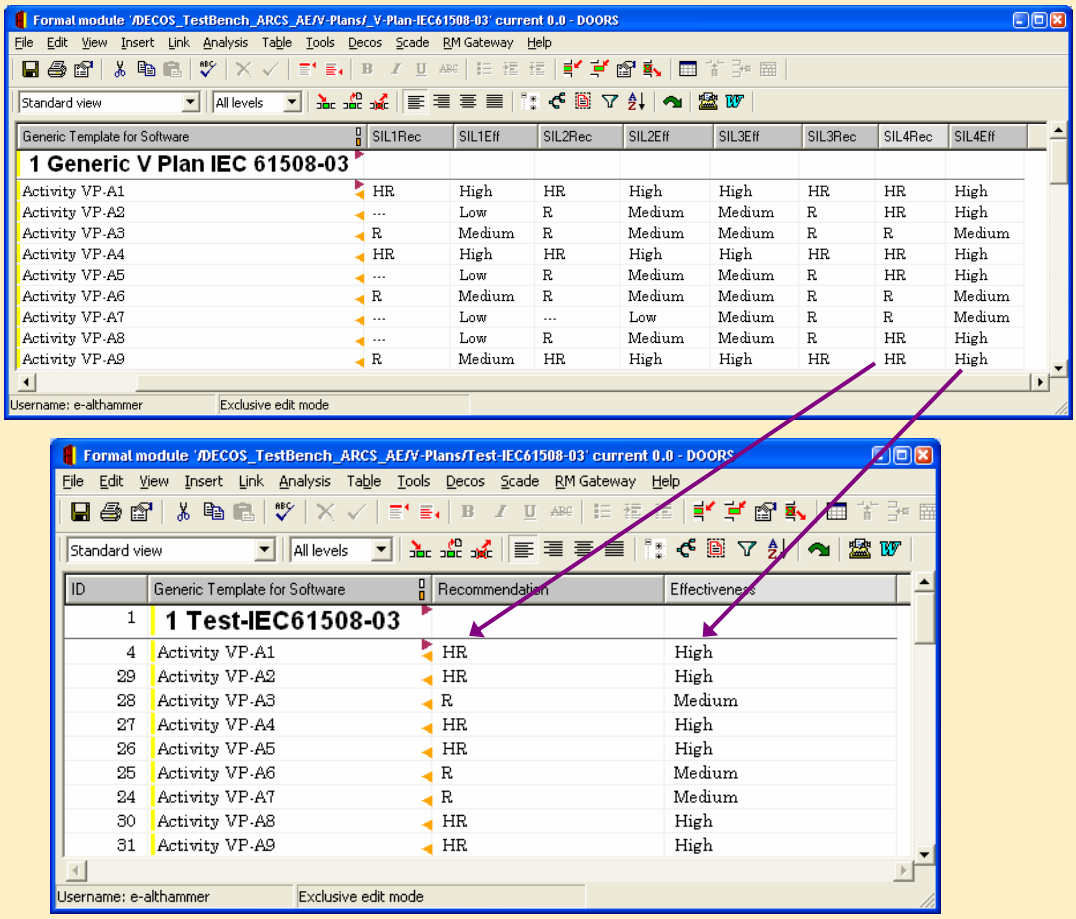
Követelmény alapú eszközláncok: DECOS Test Bench

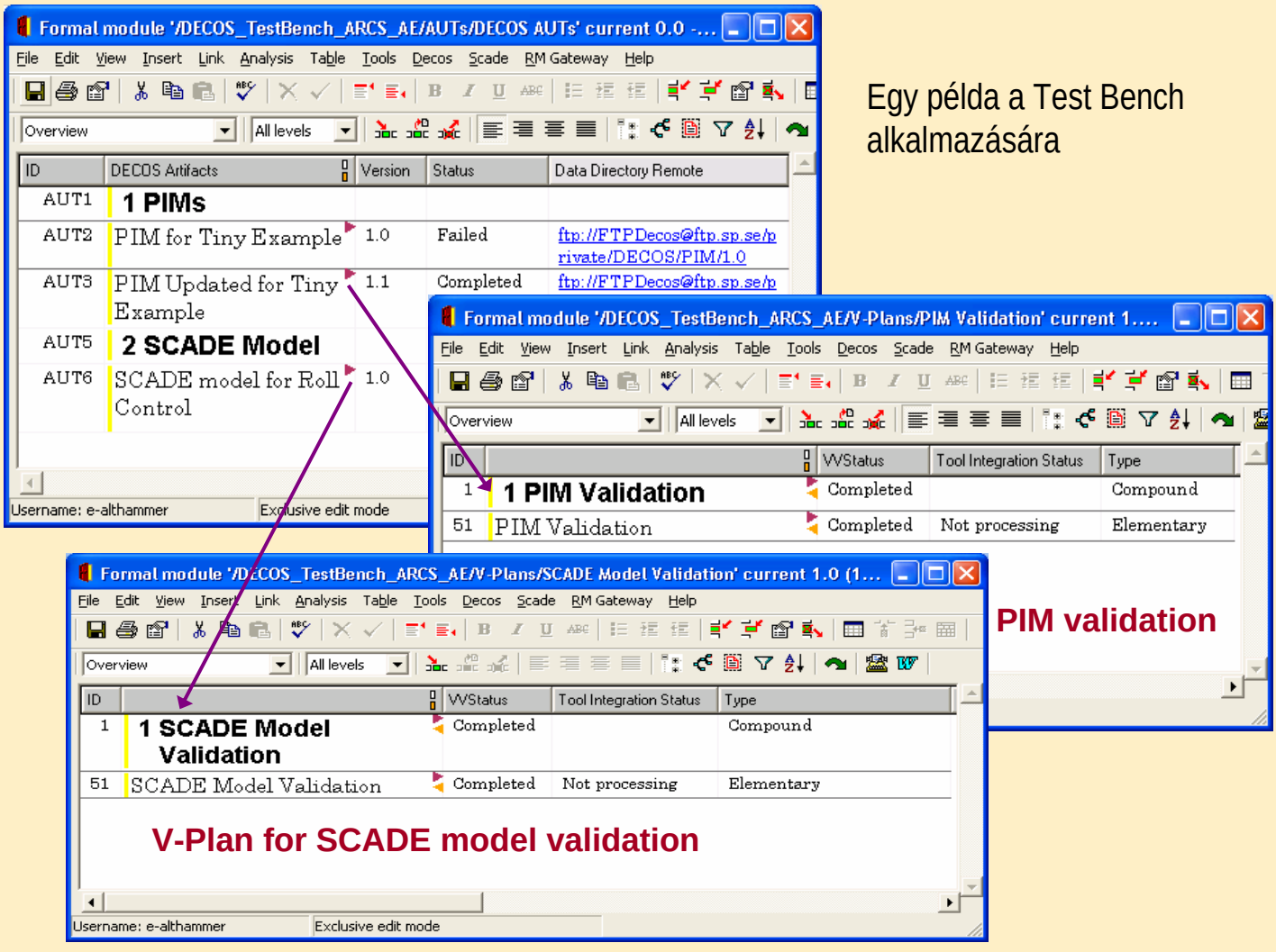
- Verifikációs tevékenység felvétele a követelmények mellé
  - Tervezett és rögzített verifikáció (pl. biztonsági ügy)
  - Ellenőrzések a követelmények és szabványok alapján
  - Többféle tevékenység kombinálható
- Verifikációs eszközláncok kialakítása (általában külső)
  - Analízis: analízis modell generálása, analízis végrehajtása, eredmények visszacsatolása
  - Tesztelés: modell alapú tesztgenerálás, teszt végrehajtás, teszt eredmény kiértékelés
  - Mérések: mérési konfigurálása, végrehajtása, eredmények értékelése
- Verifikációs eszközláncok indítása a követelménykezelőből
  - Triggerek alapján; script nyelven programozható
- Verifikáció státuszának rögzítése
  - Ellenőrzött modell, sikeresen tesztelt követelmény

Verifikációs terv

Módszerek kiválasztása:

- V-plan SIL4 rendszerhez a generikus IEC 61508-3 V-plan alapján
- Referenciákat tartalmazhat V&V eszközökre és módszerekre





Egy példa a Test Bench alkalmazására

Eszközláncok indítása a DOORS-ból

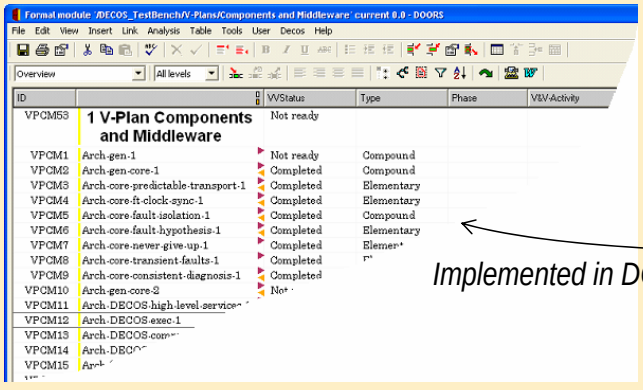
Eszközök végrehajtása:

- Belső eszköz: Pl. egyszerű ellenőrző lista
  - Megvalósítható a DOORS keretein belül
- Automatikus helyi eszközvégrehajtás: pl. RACER (ontológia alapú ellentmondásmentesség vizsgálat)
  - Automatikus indítás (scriptből) és eredmény becsatolás
- Távoli eszközvégrehajtás: pl. PROPANE (SWIFI)
  - Indítás távoli hozzáféréssel (pl. üzenet alapú)
- Kézi eszközvégrehajtás: pl. EMI Hardware Test Bench
  - Eszköz indítási és eredmény beviteli dialógus

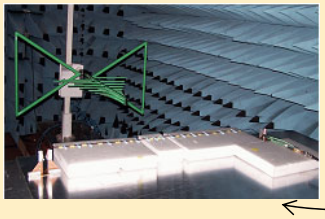
Eredmények tárolása:

- Távolról is elérhető adattár (repository)

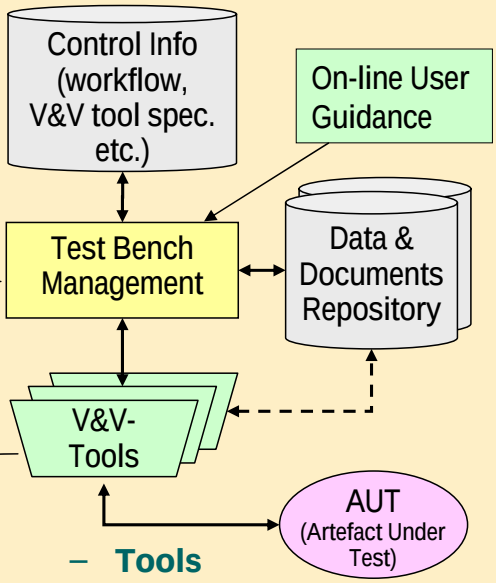
Verifikáció „menedzselése” a követelménykezelővel



Implemented in DOORS™

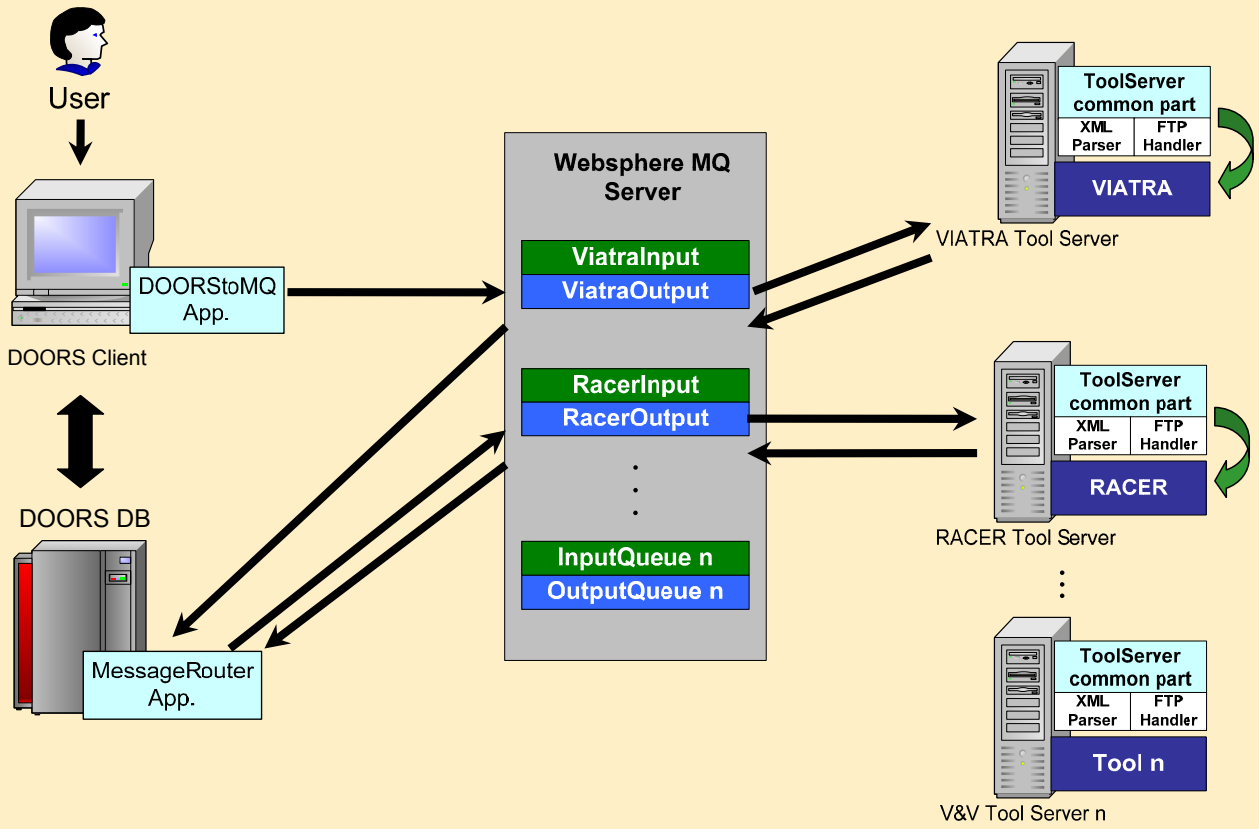


- ITEM (Hazard and Risk Analysis)
- RACER (Formal Verification)
- SCADE MTC (Simulation)
- LDRA (Testing)
- PROPANE (Fault Injection)
- EMI Test Bench



- Tools
- Middleware
- Hardware
- Models
- Applications

Példa végrehajtás: PIM ellenőrés RACER-rel

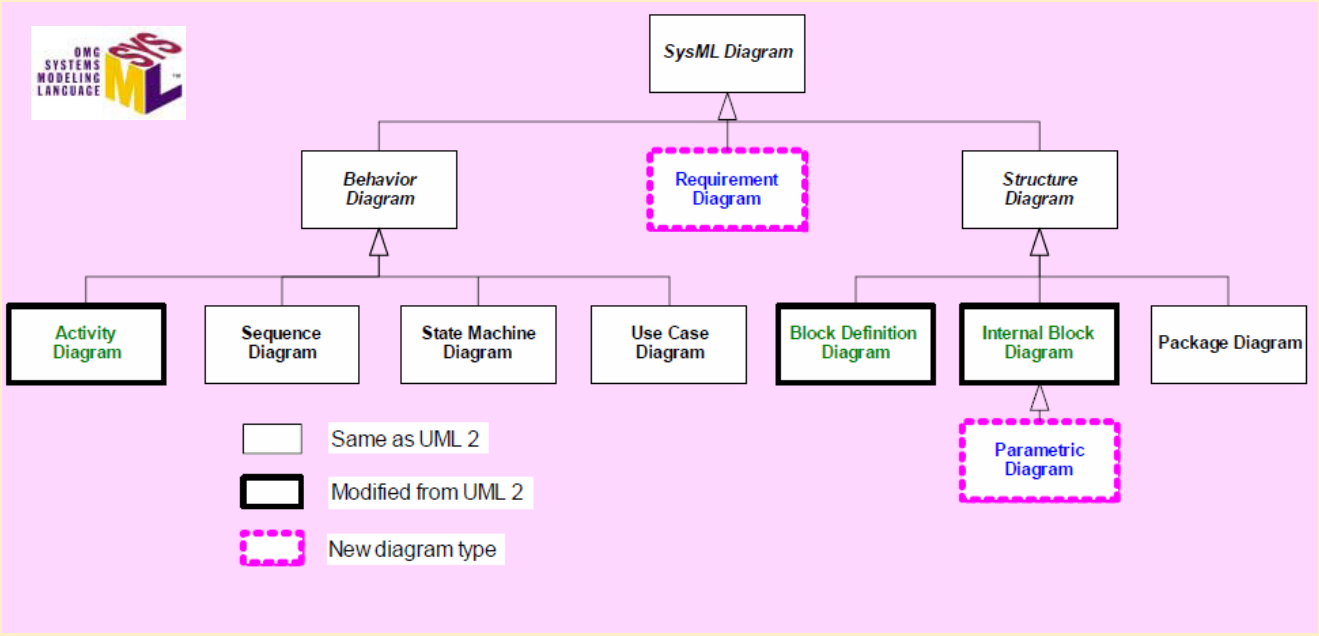


Tartalomjegyzék

- Motiváció
  - Miért fontosak a tervezési folyamat ezen szakaszai?
  - Milyen elvárások vannak a specifikációval szemben?
  - Milyen módszerei vannak a specifikáció készítésnek?
- Az általános követelménykezelés feladatai
  - Követelmények nyilvántartása
  - Követhetőség a verifikációhoz
- Félformális specifikáció
  - Specifikus technikák: SysML
- A követelményspecifikáció verifikációja
  - Általános kritériumok
  - Specifikus kritériumok UML állapottérképekre (mintapélda)

Félformális követelményspecifikáció: SysML

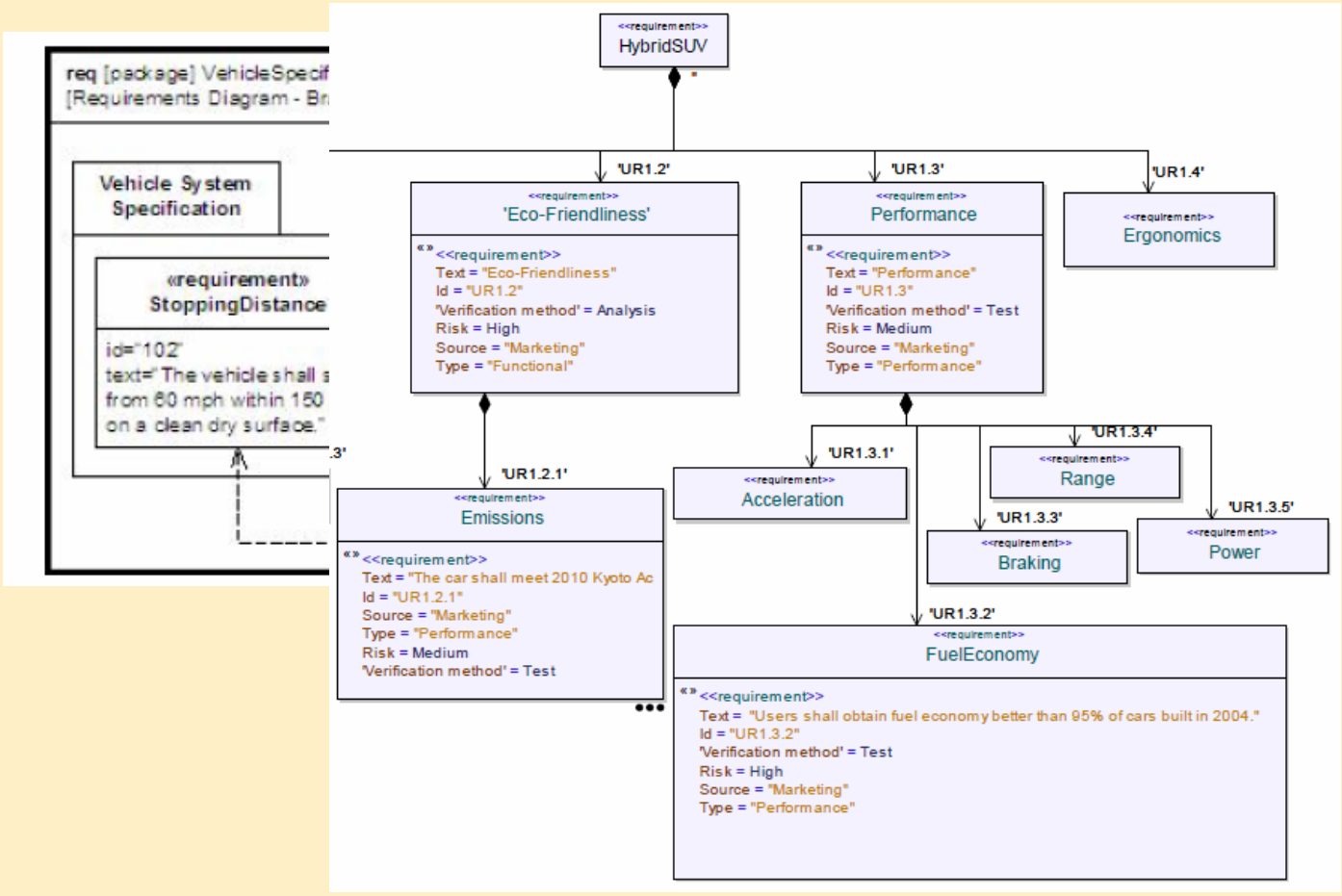
- Systems Modeling Language
  - UML részhalmaz egy kiterjesztése rendszertervezéshez
  - Fő újdonságok: Requirements és Parametric diagram



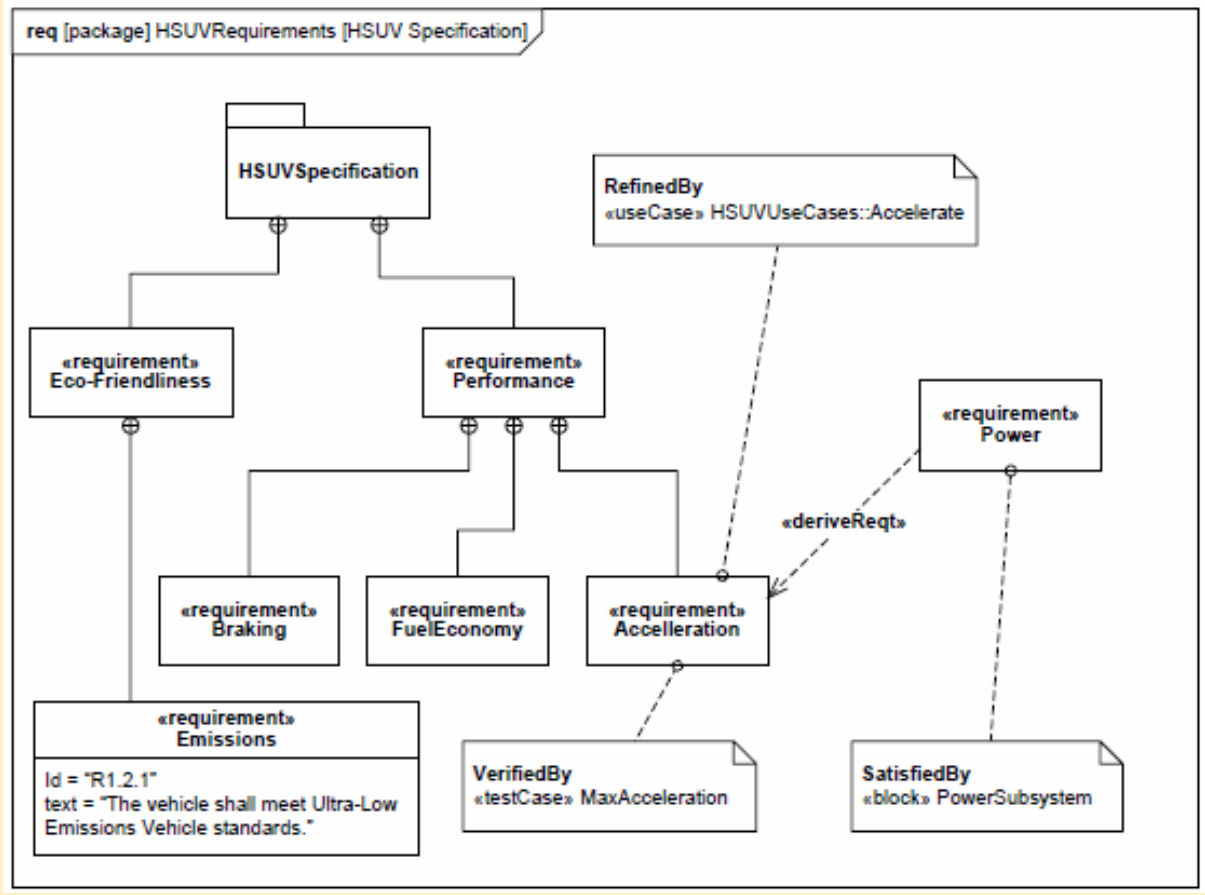
Requirements diagram

- Követelmények (szöveges is) tárolása azonosítóval
  - <<requirement>> stereotype
  - Id és text mezők
  - Felhasználói attribútumok: pl type, source, risk, ...
  - Táblázatos forma is támogatott
- Követelmények hierarchikus csomagokba rendezhetők
  - Funkcionális, teljesítmény, ... kategóriák
- Követelmények közötti finomítás (~ subclass), kompozíció
- Relációk használhatók (callout: megjegyzésekben):
  - Copy: követelmények között (master – slave)
  - Trace: követelmények között (client – supplier)
  - DeriveReqt: követelmények között (forrás – származtatott)
  - Refine: követelmények és terv elemek között (pl. szövegeshez)
  - Satisfy: követelmények és terv vagy implementáció elemek között
  - Verify: követelmények és teszt elemek között

Requirement diagram példa: Struktúra

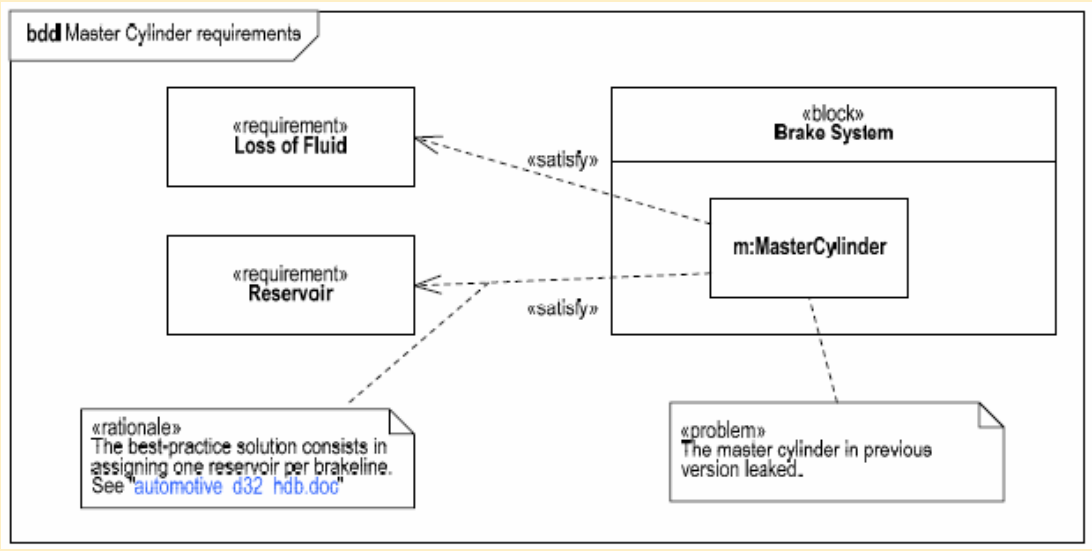


Requirement diagram példa: Relációk



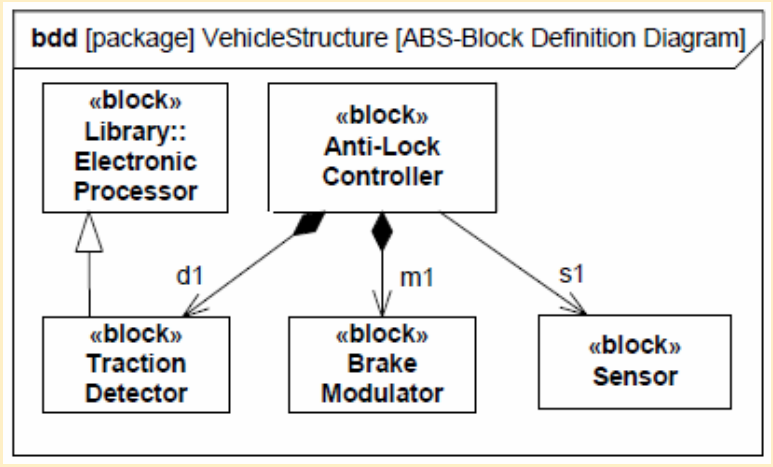
Requirement diagram példa: Tervezői döntések

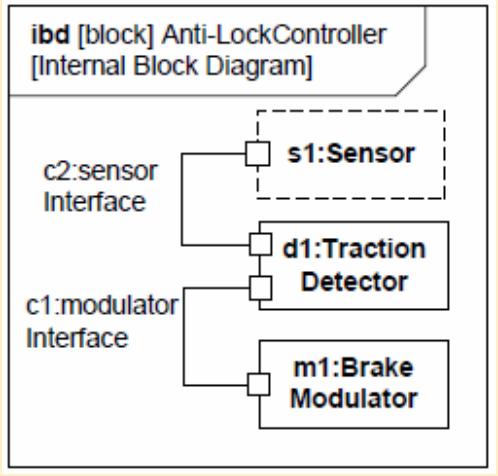
- Tetszőleges modell elemhez köthető megjegyzések (előredefiniált stereotype):
  - <<problem>>: Probléma, döntést igénylő felvetés
  - <<rationale>>: Megoldás, magyarázat



Block diagram

- Block: A struktúra eleme (fekete / üveg doboz)
  - Komponens (nem csak szoftver)
  - A SysML-ben az UML 2.0 osztályokon alapul
- Internal block diagram:
  - Konkrét szerepek; típust a Block adja meg

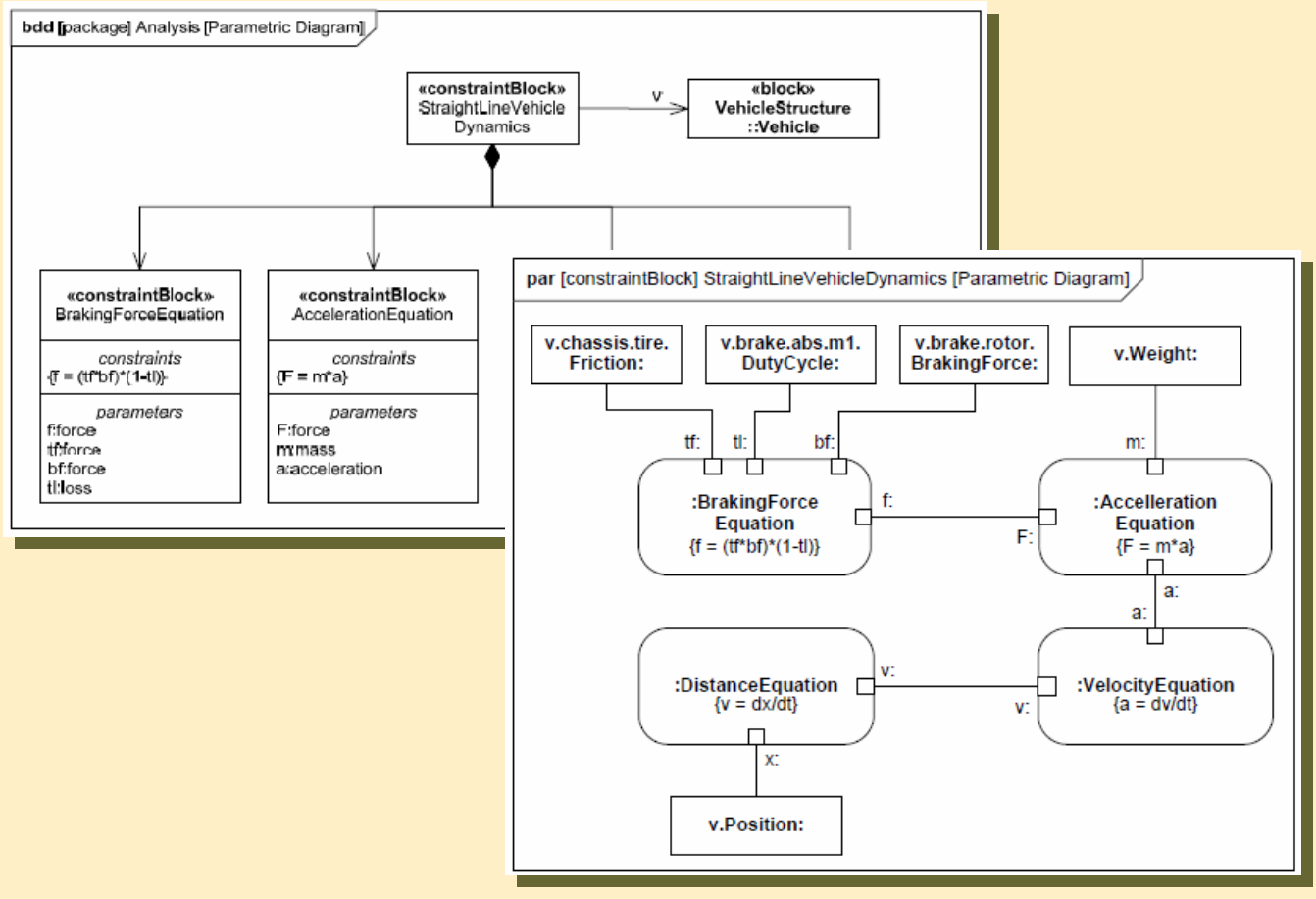




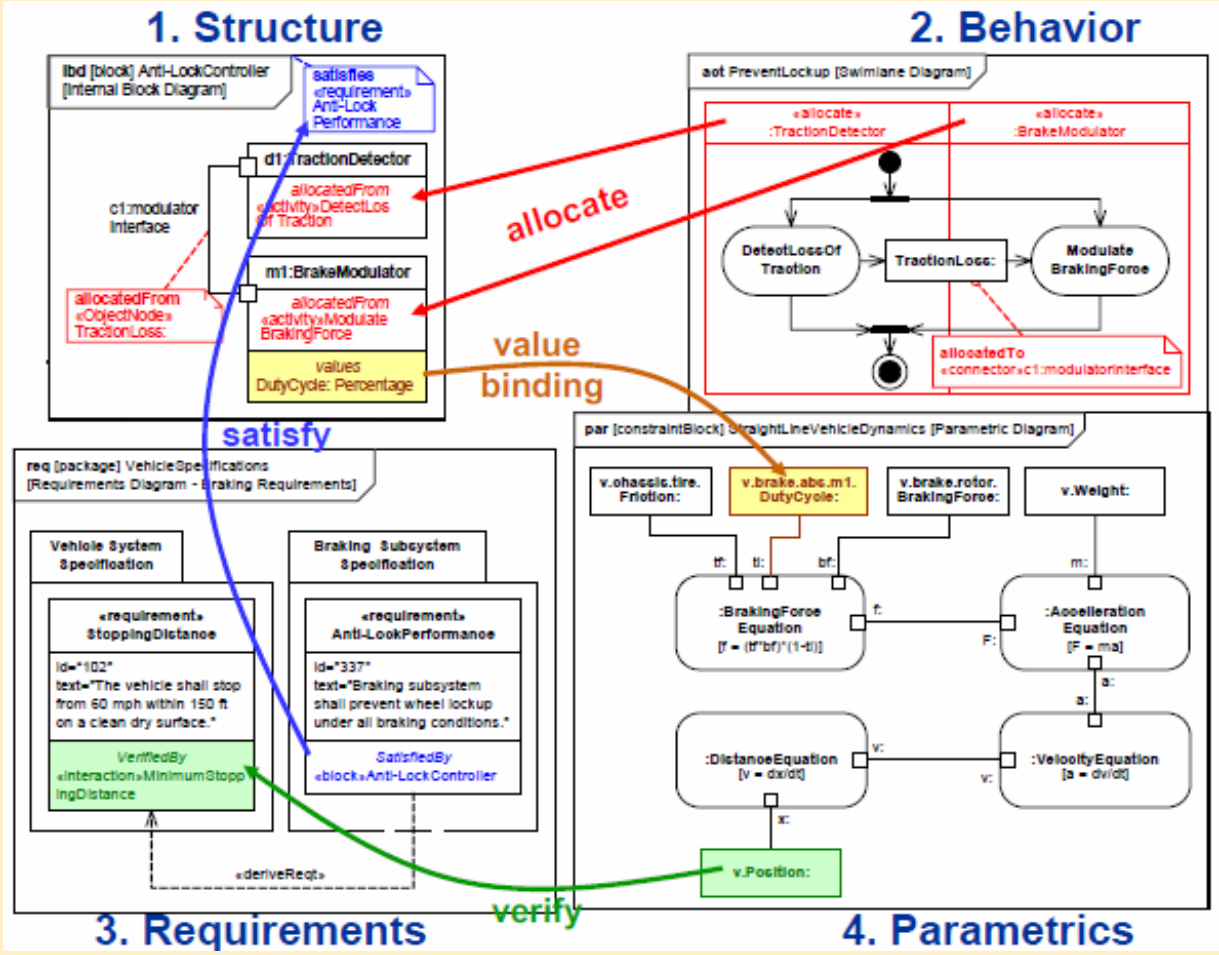
Parametric diagram

- Cél: Ellenőrizhető számszerű követelmények (kényszerek) megfogalmazása tulajdonságokra
  - Nem-funkcionális követelmények aspektusa
  - Analízis (pl. teljesítmény, megbízhatóság) támogatása
- ConstraintBlock: Összefüggések megadása
  - Formális (pl. MathML, OCL), vagy informális alakban
  - Analízis eszközhöz igazítható (nem SysML specifikus)
- Parametric diagram: Alkalmazás
  - Az összefüggések (Constraint block) alkalmazása egy adott környezetben
  - Kötések értékek között

Parametric diagram példa



Relációk diagramok között: Követhetőség



Tartalomjegyzék

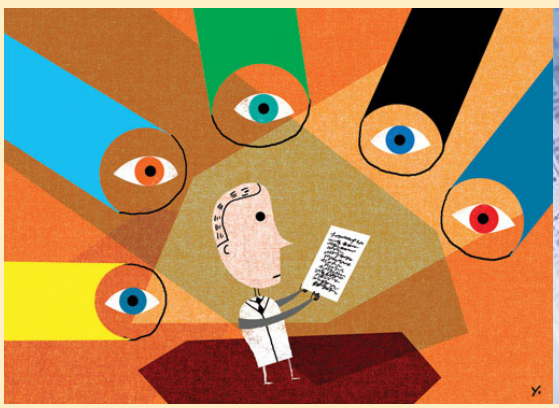
- Motiváció
  - Miért fontosak a tervezési folyamat ezen szakaszai?
  - Milyen elvárások vannak a specifikációval szemben?
  - Milyen módszerei vannak a specifikáció készítésnek?
- A követelménykezelés általános feladatai
  - Követelmények nyilvántartása
  - Követhetőség a verifikációhoz
- Félformális specifikáció
  - Specifikus technikák: SysML
- A követelményspecifikáció verifikációja
  - Általános kritériumok
  - Specifikus kritériumok UML állapottérképekre (mintapélda)

A jó specifikáció jellegzetességei: IEEE Std 830-1998

- Helyes
  - A szoftverre vonatkozó követelményeknek (elvárásoknak) megfelelő
  - Konzisztens a külső forrásokkal (pl. szabványok)
- Egyértelmű
  - Nem félreérthető, egy jelentése van
  - Hasznosak a formális, félformális specifikációs nyelvek
- Teljes
  - Minden (érvényes, érvénytelen) bemenetre van specifikált viselkedés
  - TBD csak indoklással és a feloldás módjával
- Konzisztens
  - Nincs belső ellentmondás, egységes a terminológia
- Fontosság és stabilitás szempontjából rendezett
  - Követelmények szükségessége, változatlansága felmérve
- Ellenőrizhető
  - Megállapítható egyértelműen, ha nem teljesül egy követelmény
- Módosítható
  - Nem redundáns, jól strukturált, jól elválasztott követelmények
- Követhető
  - Eredet becsatolható, további hatások hivatkozhatók

Ellenőrzési módszerek

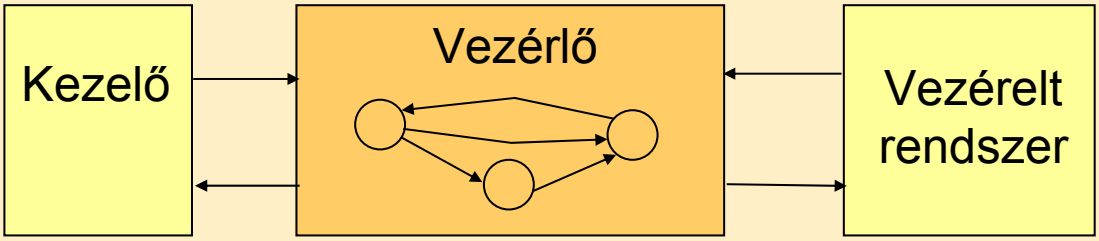
- Ellenőrző listák
  - Tipikus hibák esetén hatékony (újra ne kövessük el)
  - Teljességet nem várhatunk
- Statikus analízis
  - Hiányosságok, ellentmondások kiszűrése a specifikáció (ill. terv, kód) végrehajtása nélkül
  - Analógia: Átolvasás „sorról sorra”
  - Szerepek: Szerző, átvizsgáló, tesztelő



Példa: Állapotgép modellek vizsgálati szempontjai

Teljesség és ellentmondásmentesség (N. Leveson):

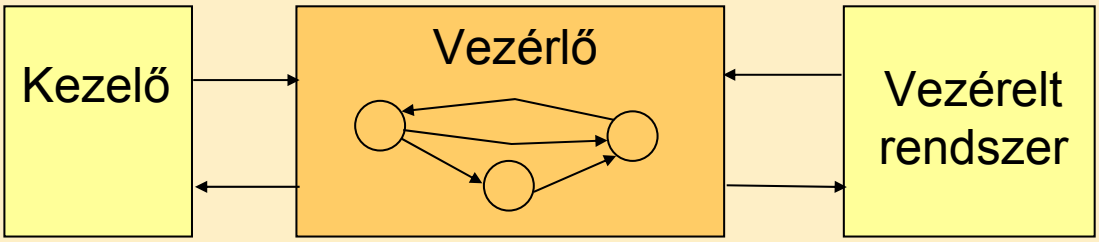
- Állapotdefiníció
- Bemenetek (események)
- Kimenetek
- Kimenetek és trigger kapcsolata
- Állapotátmenetek
- Ember-gép interfész



Példa: Állapotgép modellek vizsgálati szempontjai

- Állapotdefiníció
- Bemen
- Kimen
- Kimen
- Állapo
- Ember-gép interfész

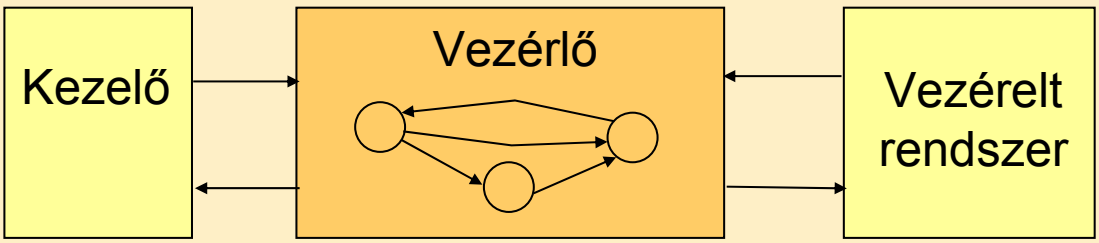
- Biztonságos a kezdőállapot
- Belső modell aktualizálva van a környezettel
  (kimaradó bemeneti események esetén time-out van, és nincs a kimeneten akció)



Példa: Állapotgép modellek vizsgálati szempontjai

- Állapotdefiníció
- Bemenetek (események)
- Kimenetek
- Kimenet
- Állapotá
- Ember-g

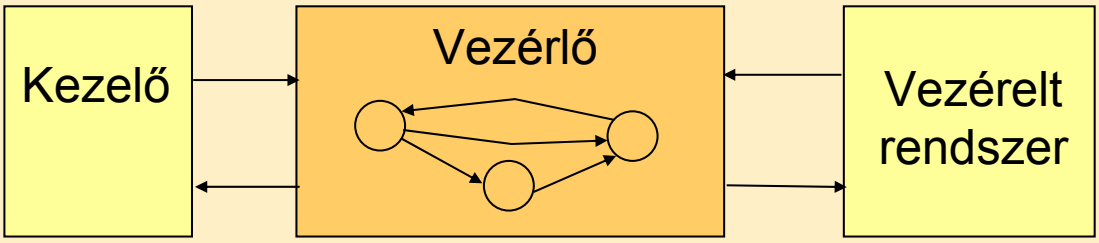
- Minden bemenetre, minden állapotban van specifikált viselkedés (reakció)
- Egyértelműek (determinisztikusak) a reakciók
- Van bemeneti ellenőrzés (értékbeli, időbeli)
- Hibás bemenet kezelése specifikálva
- Megszakítások gyakorisága korlátozva



Példa: Állapotgép modellek vizsgálati szempontjai

- Állapotdefiníció
- Bemenetek (események)
- Kimenetek
- Kimenet
- Állap
- Embe

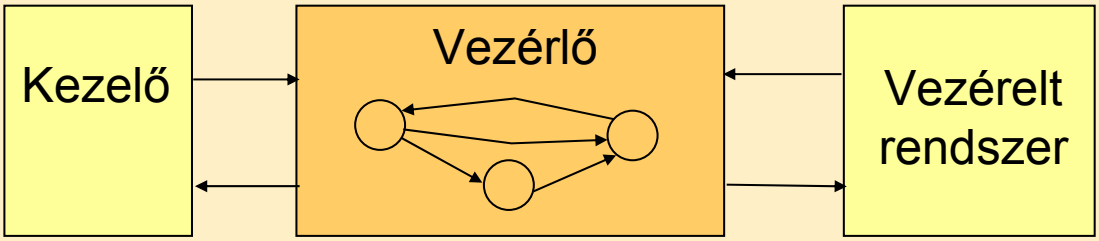
- Hihetőségvizsgálat kritériumai specifikáltak
- Nincsenek fel nem használt kimenetek
- Környezeti feldolgozóképesség be van tartva



Példa: Állapotgép modellek vizsgálati szempontjai

- Állapotde
- Bemenet
- Kimenetek
- Kimenetek és trigger kapcsolata
- Állapotátmenetek
- Ember-gép interfész

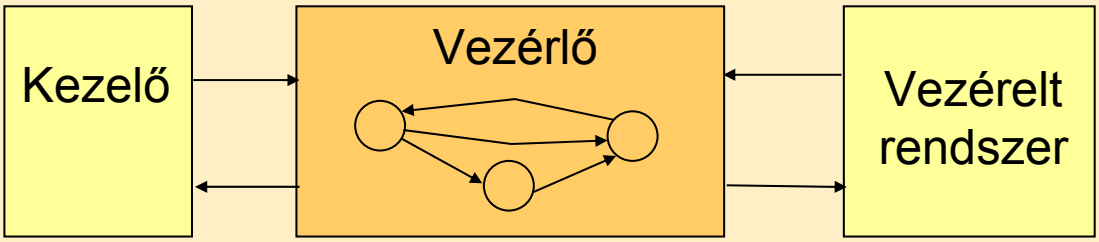
- Kimenetek hatása a bemeneteken keresztül ellenőrizve
- A szabályzási kör stabil



Példa: Állapotgép modellek vizsgálati szempontjai

- Álla
- Be
- Kin
- Kime
- Állapotátmenetek
- Ember-gép interfész

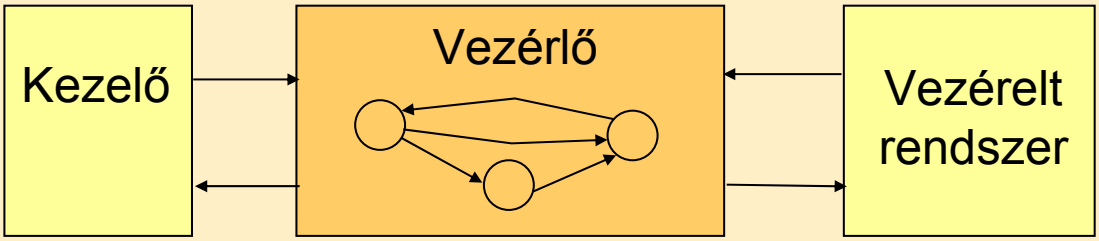
- Minden állapot elérhető statikusan
- Állapotátmenetek visszafordíthatók (visszaút van)
- Több átmenet van veszélyes állapotból biztonságosba
- Megerősített átmenet van biztonságos állapotból veszélyes állapotba



Példa: Állapotgép modellek vizsgálati szempontjai

- Álla
- Be
- Kir
- Kim
- Állapotátr tek
- Ember-gép interfész

Kezelő felé kimenő események specifikációja:
- Sorrendezés előírt (prioritással)
- Frissítés előírt
- Gyakoriság korlátozott (kezelő terhelhetősége)



Tartalomjegyzék

Példa: IAR VisualState eszköz

A statikus ellenőrzés UML állapottérképeken:

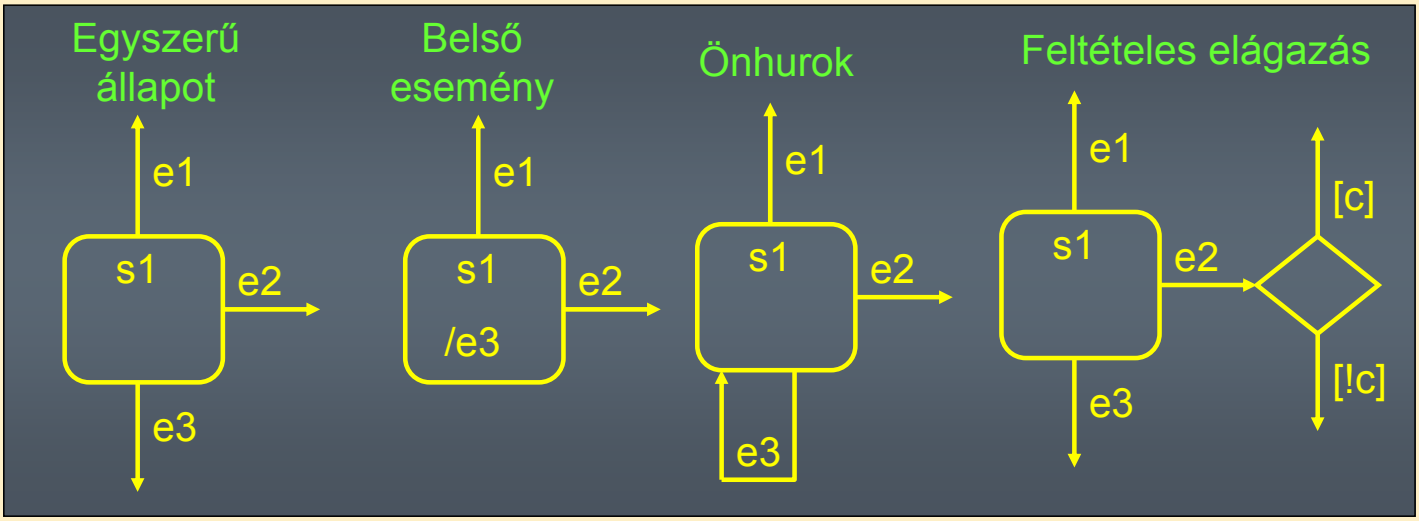
- Reset funkcionalitás
- Események, akciók felhasználása
- Állapotátmenetek statikus engedélyezettsége
- Állapotátmenet konfliktusok (azonos trigger)
- Állapotok statikus elérhetősége
- Nyelő állapot keresése (nincs kimenet)

Példa: Nehézségek UML állapottérképek vizsgálata esetén

- Hierarchikus állapot-definíció:
  (1) Állapot helyett állapot-konfiguráció ellenőrzése kell
  (2) A prioritásokat is figyelembe kell venni
  - Teljesség: Minden eseményre, minden állapot-konfigurációban van specifikált viselkedés (tranzíció vagy helybenmaradás)
  - Ellentmondásmentesség: Egy eseményre egy állapot-konfigurációban csak egy tranzíció lehet engedélyezett
- Konkurens régiók: Konkurens tranzíciók
  - Akciók determinisztikus végrehajtása szükséges
- Őrfeltételek használata: Kiértékelés
  - Teljesség: Bármely állapot-konfigurációban egy esemény által triggerelt tranzíciók őrfeltételeinek VAGY kapcsolata igaz értéket ad (tautológia)
  - Ellentmondásmentesség: Bármely állapot-konfigurációban egy esemény által triggerelt tranzíciók őrfeltételei közül csak egy lehet igaz

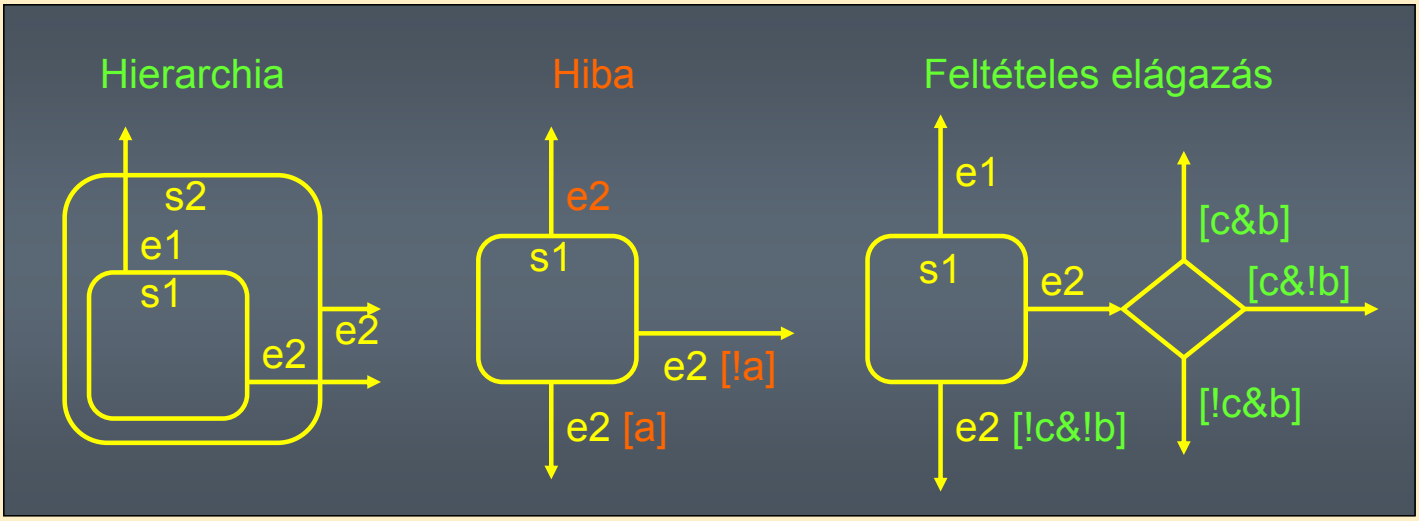
Teljesség

- Minden állapotkonfigurációból, minden eseményre vonatkozóan, minden őrfeltétel-kiértékelés esetén kell lennie definiált tranzíciónak



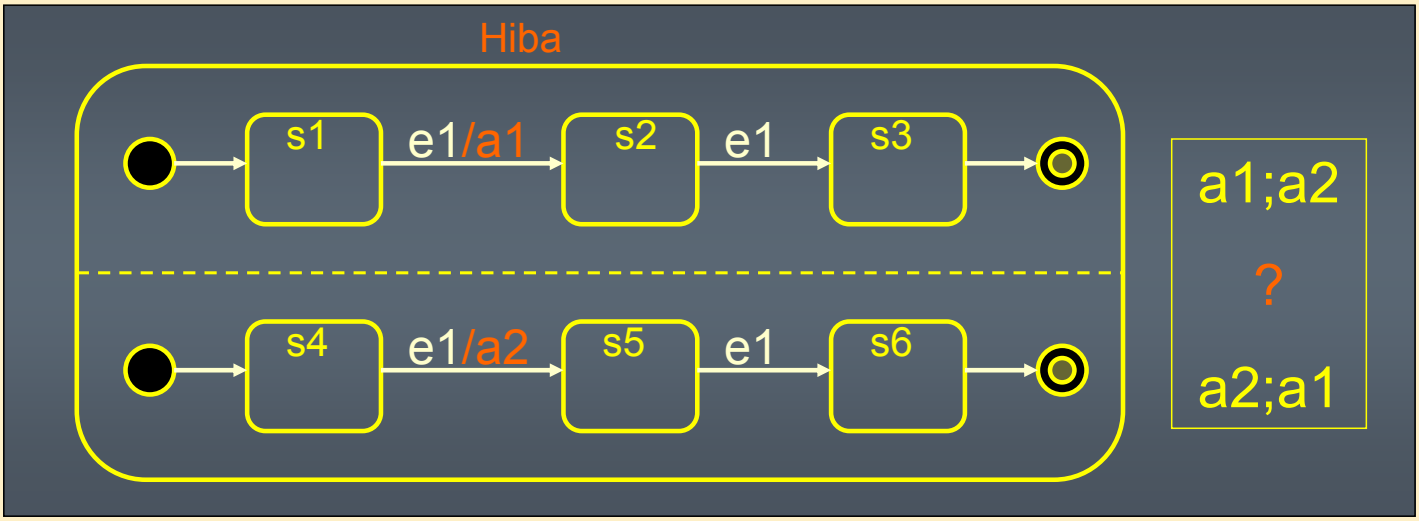
Egyértelműség I.

- Minden állapotkonfiguráció és minden esemény esetében az összes őrfeltétel-kiértékelés mellett egy hierarchia szinten belül egy időben csak egy tranzíció lehet aktív



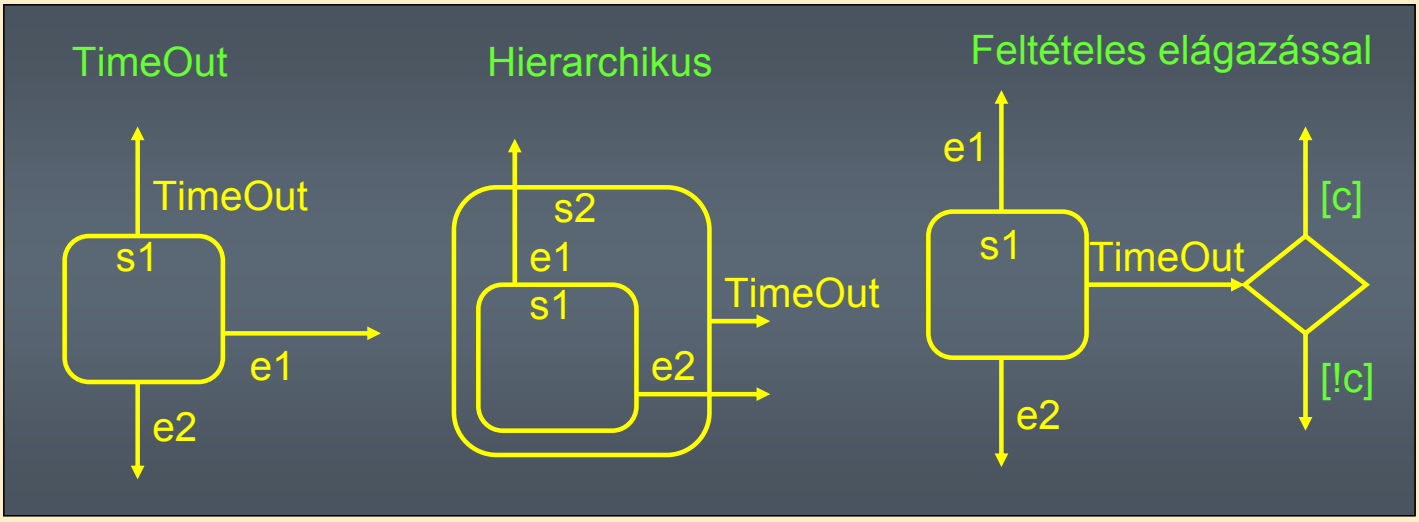
Egyértelműség II.

- Konkurens állapotgépeken belül egyazon eseményre csak az egyik gépben lehet akció definiálva



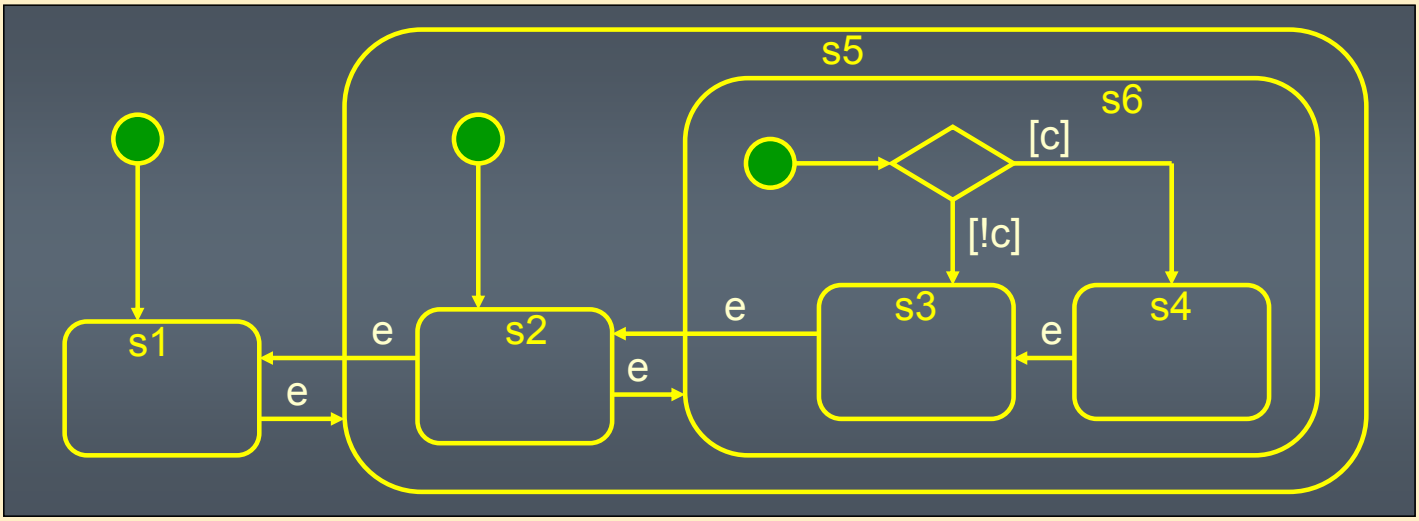
Időtúllépés

- Minden állapotkonfigurációra vonatkozóan definiálva kell lennie olyan tranzíciónak, mely a TimeOut nevű eseményre van triggerelve (lehet örökölt tranzíció is)



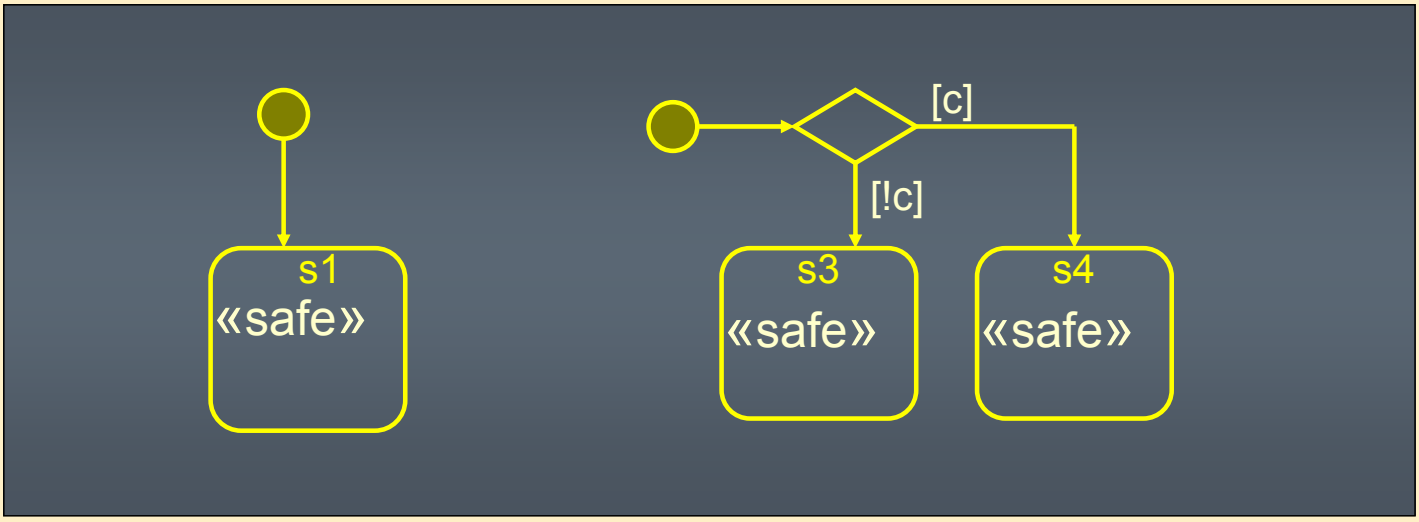
Indulási állapot I.

- Minden rész-automatában szerepelnie kell Start állapotnak, beleértve a legfelső szintű régiót is



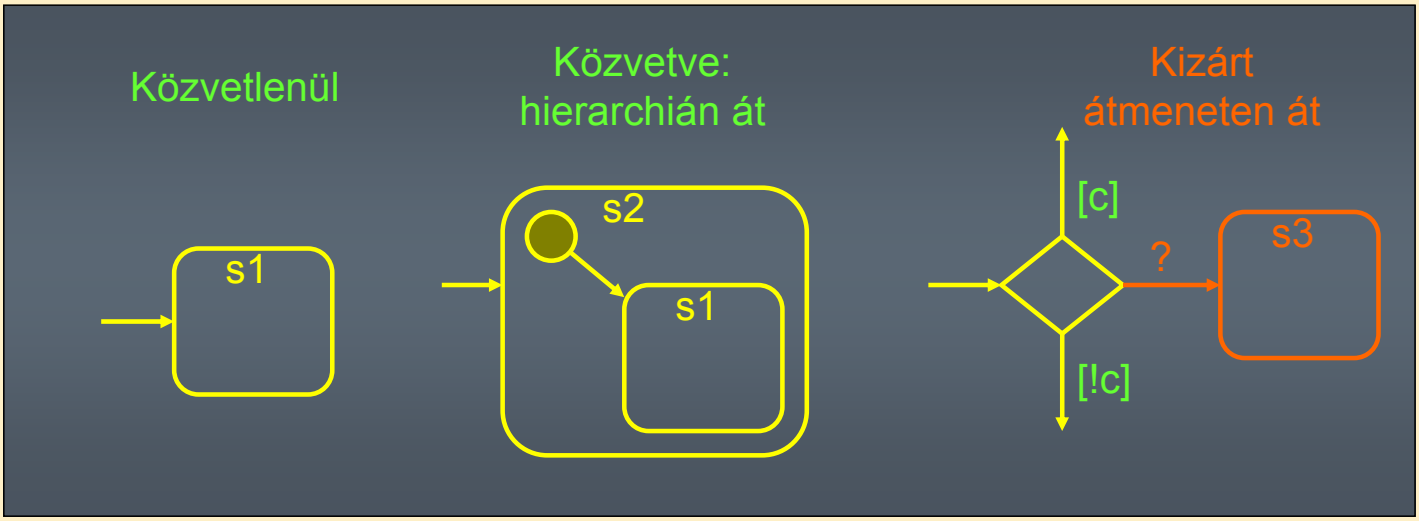
Indulási állapot II.

- A legfelső szintű régió induló állapotának biztonságosnak kell lennie:
  a Start-ból közvetlenül elérhető állapotoknak «safe» sztereotípiával jelöltnek kell lennie



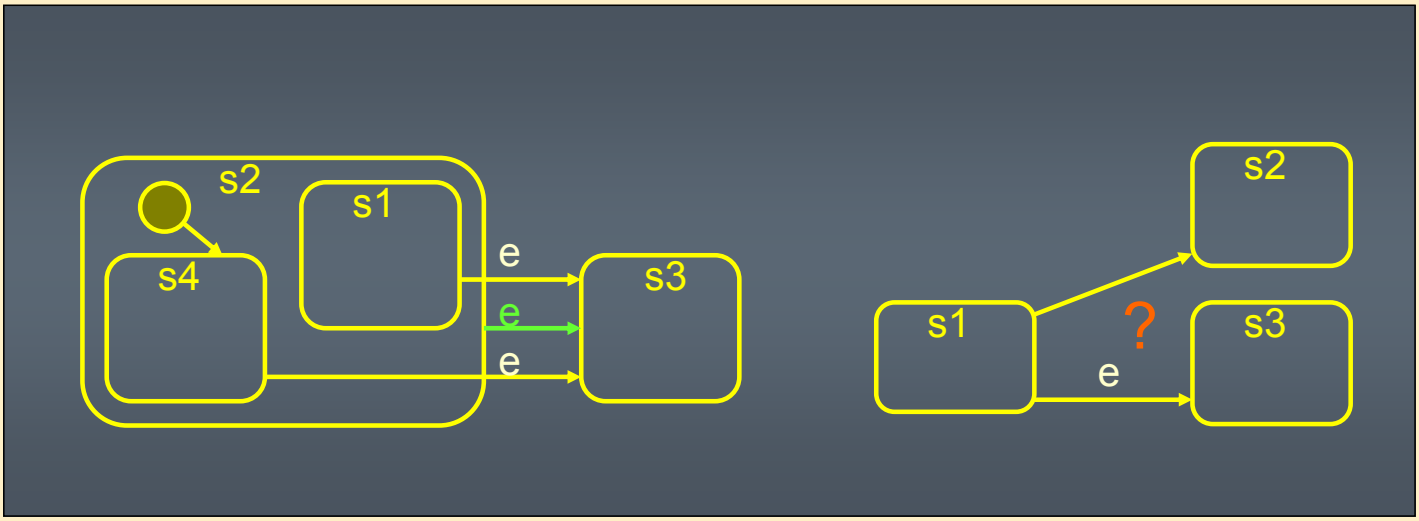
Elérhetőség

- A rendszer minden egyszerű állapotának elérhetőnek kell lennie vagy közvetlenül, vagy közvetve



Takarás, completion

- A hierarchia miatt tranzíciók takartak lehetnek
- Completion és eseménnyel triggerelt tranzíció nem keverhető



Az ellenőrzés menete

Miről volt szó?

- Motiváció
  - Miért fontosak a tervezési folyamat ezen szakaszai?
  - Milyen elvárások vannak a specifikációval szemben?
  - Milyen módszerei vannak a specifikáció készítésnek?
- A követelménykezelés általános feladatai
  - Követelmények nyilvántartása
  - Követhetőség a verifikációhoz
- Félformális specifikáció
  - Specifikus technikák: SysML
- A specifikáció verifikációja
  - Általános kritériumok
  - Specifikus kritériumok UML állapottérképekre (mintapélda)

Hogyan dokumentálható az ellenőrzés eredménye?

- Szoftverkövetelmények igazolójelentése
  - Megvalósítás dokumentálása
    - Felülvizsgálat
    - Egyenrangú átvizsgálás
  - Vizsgálati szempontok szerinti eredmények dokumentálása
    - Ellenőrző lista
    - Követhetőségi vizsgálatok
    - Statikus analízis
    - Formális verifikáció
    - …
  - Összefoglaló vélemény
    - Minőségi értékelés
    - Szükséges javítások előírása