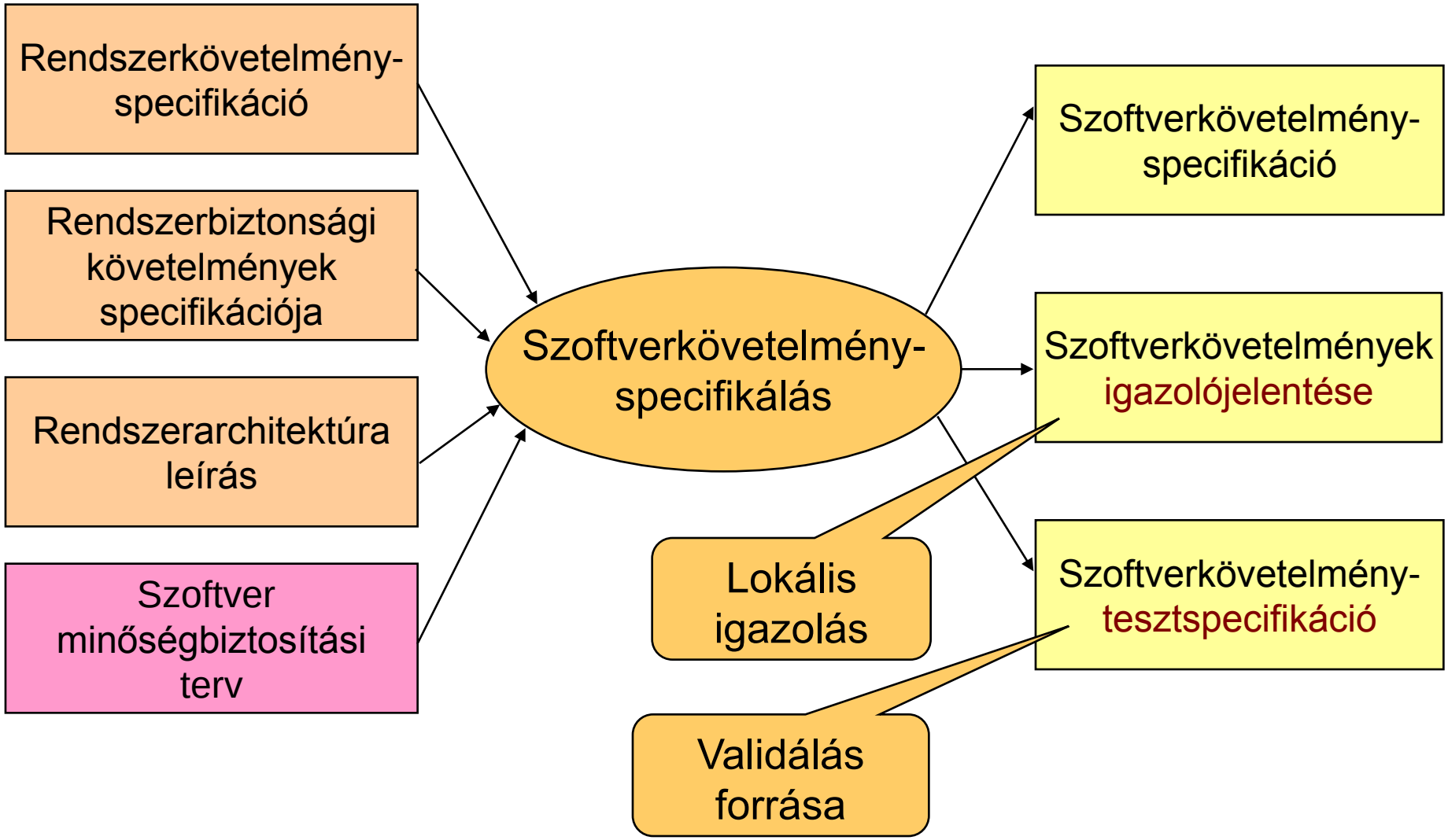
Követelmény-specifikáció készítés és ellenőrzés

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

Tartalomjegyzék

Kimenetek és bemenetek



Szoftver minőségbiztosítási terv

- Célkitűzés:
  - Védelem a szisztematikus hibák ellen
  - Az összes technikai és irányítási tevékenység meghatározása, figyelése, ellenőrzése
- Meghatározandó pontok:
  - Tevékenységek, be/kimeneti kritériumok
  - Számszerű minőségi elvárások (pl. ISO/IEC 9126)
  - Saját korszerűsítés / felülvizsgálat lépései
- Külső beszállítók ellenőrzésére szolgáló eljárások
  - Külső szállító minőségbiztosítási tervének megfelelősége
  - Szoftverek ellenőrzése, fejlesztési eszközök megfelelősége
- Probléma bejelentés és javítás rögzítése
  - Dokumentáció és visszajelzések
  - Elemzések (okok azonosítása)
  - Felderítés és megoldás (javítás) módszerei, megelőző tevékenységek
  - Javítás ellenőrzése, ismételt igazolás és érvényesítés

Szoftverigazolási terv (verifikációs terv)

- Biztonságkritikus rendszerekben külön terv
- Verifikációs tevékenységek tervezése
  - A szabvány szerinti választási lehetőségek kiaknázása
  - Kritériumok és eszközök meghatározása
- Általános igazolási célok:
  - „Lokális” helyesség: Teljesség, ellentmondás-mentesség
  - Megfelelőség: Az előző fázis(ok) követelményeinek
- Részletezés:
  - Eszközök (pl. teszt berendezés)
  - Eredmények kiértékelése (elfogadás feltételei), megbízhatósági követelmények kiértékelése
  - Résztvevők szerepe és felelőssége
  - Teszt lefedettség foka

Szoftverkövetelmény-tesztspecifikáció

- Az összes követelmény igazolására
  illetve a kész szoftveren végzett tesztek leírására
  - Teszteléssel vizsgálható követelmény
  - Elemzéssel vizsgálható követelmény (pl. architektúra)
- Tesztesetek rögzítése:
  - Bemenetek és sorrendjük
  - Elvárt kimenetek és sorrendjük
  - Teljesítési feltételek (siker / kudarc ismérve)
- Nagy rendszereknél célszerű:
  - Felhasználói interfészek forgatókönyv alapú ellenőrzése
  - Magas kockázatú elemek esetén gyors prototípuskészítés

Szoftverkövetelmény-specifikáció

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

Tartalomjegyzék

- A fázis ki- és bemenetei
- A szoftverkövetelmény-specifikáció elkészítése
  - Formális nyelvek (áttekintés)
  - Félformális és strukturált technikák
  - Példa: SysML
- Ellenőrzési feladatok
  - Általános szempontok és módszerek
  - Teljesség és ellentmondás-mentesség
- Követelménykezelés
  - Feladatok
  - Automatikus eszközök

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 „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!

Formális nyelvek áttekintése

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Processz leíró nyelvek (CSP, CCS, …)
- Logikai nyelvek (HOL, CTL*, …)
- Konstruktív nyelvek (NUPRL, …)
- Hibrid illetve széles spektrumú nyelvek (CPN, E-LOTOS, …)

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek
- Processz leíró nyelvek
- Logikai nyelvek
- Konstruktív nyelvek
- Hibrid illetve széles spektrumú nyelvek (CPN, E-LOTOS, …)

Matematikai modell:
- Állapotok mint halmazelméleti struktúrák
- Műveletek, események (elő- és utófeltételekkel)

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)

Egy beléptető rendszer specifikációja (Event-B):

személyek:	$prs \neq 0$	(halmaz)
épületek:	$bld \neq 0$	(halmaz)
jogosultság:	$aut \in prs \leftrightarrow bld$	(bináris reláció)
tartózkodás:	$sit \in prs \rightarrow bls$	(teljes függvény)
invariáns:	$sit \subseteq aut$	

absztrakt esemény („lehetséges történés”):

```
pass = ANY p,b WHERE (p,b)∈aut ∧ sit(p)≠b
       THEN  sit(p):=b END
```

Bizonyítható, hogy az esemény nem sérti az invariánst.
Bizonyítható az absztrakt események konzisztens finomítása.

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Processz leíró nyelvek
- Logikai nyelvek
- Konstruktív nyelvek
- Hibrid illetve széles spektrumú nyelvek (CPN, E-LOTOS, …)

Absztrakt algebra és kategória-elmélet
- Absztrakt adattípusok: hordozó halmaz és műveletek
- Elsőrendű logika tipikus

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Proce
- Log
- Kor
- Hibr

Absztrakt adattípusok: hordozó halmaz és műveletek

```
Type Boolean is
        sorts Bool
        opns
                false, true : -> Bool
                not : Bool -> Bool
                and : Bool, Bool -> Bool
        eqns
                forall x, y: Bool
                ofsort Bool
                        not(true) = false;
                        not(false) = true;
                        x and true = x;
```

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Processz leíró nyelvek (CSP, CCS, …)
- Logikai nyelvek
- Konstruktív nyelvek
- Hibrid illetve széles sp (CPN, E-LOTOS, …)

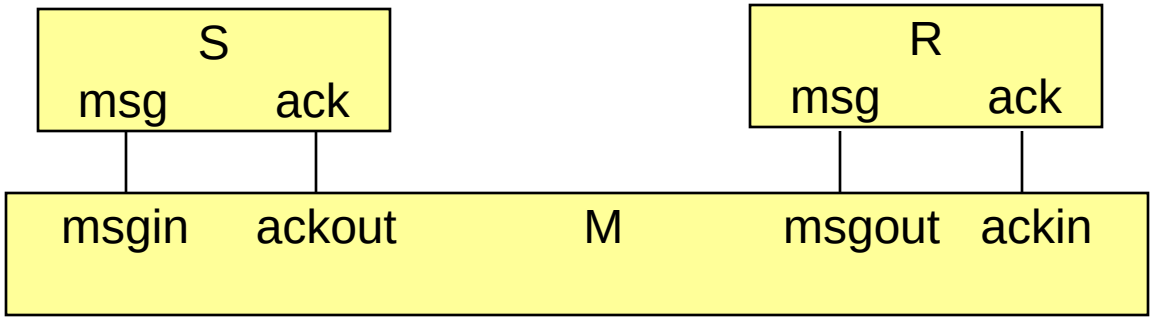
• Processzek (szekvenciális utasítás végrehajtás)
• Operátorok a processzek között (szinkronizáció, kommunikáció)

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Processz leíró nyelvek (CSP, CCS, …)
- Logi
- Ko
- Hib

Processz algebra (CCS):

$$S = \overline{msg}.ack.S$$
$$R = msg.\overline{ack}.R$$
$$M = msgin.\overline{msgout}.M + ackin.\overline{ackout}.M$$
$$P = S[msgin/msg, ackout/ack] \mid M \mid P[msgout/msg, ackin/ack]$$



Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Processz leíró nyelvek (CSP, CCS, …)
- Logikai nyelvek (HOL, CTL*, …)
- Konstruktív nyelvek
- Hibrid illetve széles sp

• Formális matematikai logika
(elsőrendű vagy magasabb rendű)
• Temporális logikák
(temporális operátorok)

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Processz leíró nyelvek (CSP, CCS, …)
- Logikai nyelvek (HOL, CTL*, …)
- Konstruktív nyelvek (NUPRL, …)
- Hibrid illetve széles s

Konstruktív logikai rendszerek (hatékonyan számítható függvények): A tulajdonság bizonyítása egyben a konstrukciót (implementációt) is megadja.

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Processz leíró nyelvek (CSP, CCS, …)
- Logikai nyelvek (HOL, CTL*, …)
- Konstruktív nyelvek (NUPRL, …)
- Hibri

Matematika: Nem konstruktív bizonyítások

- Példa: Valami létezése bizonyítható anélkül, hogy kideríthetnénk, hogy mi az a valami:
  - Létezik $a,b \notin Q$ úgy, hogy $a^b \in Q$

- Szoftver specifikációhoz nem alkalmas, ezért korlátozás szükséges a konstruktív bizonyításokra, amik a specifikáció betarthatóságát (szintézist) garantálják

Formális nyelvek

- Modell-orientált nyelvek (VDM, Z, B, …)
- Algebrai nyelvek (ADT, OBJ, …)
- Processz leíró nyelvek (CSP, CCS, …)
- Logikai nyelvek (HOL, CTL*, …)
- Konstruktív nyelvek (NUPRL, …)
- Hibrid illetve széles spektrumú nyelvek (CPN, E-LOTOS, …)

• Kombinálják az egyes formalizmusok előnyeit, pl.
  • LOTOS: processz algebra és ADT
  • CPN: Petri-háló és adatkezelés (ML)

Félformális nyelvek

- Struktúra leírás:
  - Funkcionális blokkdiagramok
- Adatáramlás leírása:
  - Adatáramlási diagramok
- Vezérlés leírása
  - Állapotátmeneti diagram, állapottérkép
  - (Üzenet) szekvencia diagram
- Logikai feltételek leírása
  - Igazságtáblázatok
  - Kényszer nyelvek (pl. OCL a struktúra alapján)

Strukturált metodika

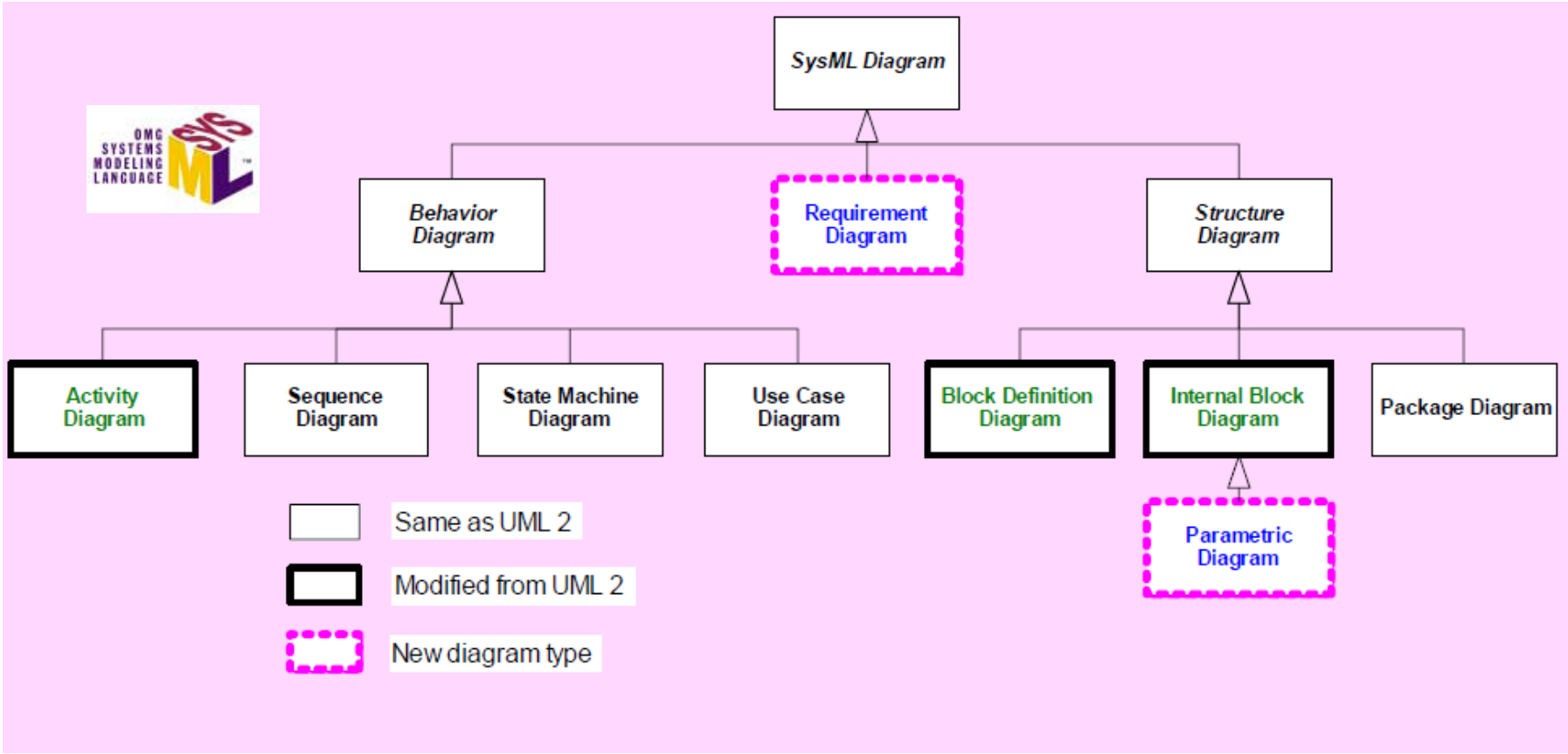
- Jellemzők:
  - Teljes rendszer + környezet leírása
  - Kezelhető részekre osztás (strukturáltság)
  - Intuitív, pragmatikus próbál lenni
  - Ellenőrző listák csatolhatók
  - Jelölésrendszer: Rendszerelemek (viselkedés általában informális)

Strukturált metodika: klasszikus példák

- Jackson System Development (JSD)
  - Entitás struktúra: Entitások + akciók (sorrend) + processzek
  - Hálózati: Kommunikáló szekvenciális processzek
- Valósidejű Yourdon (Ward-Mellor)
  - Alapvető: Környezet (események) + viselkedés (válasz)
  - Konstrukciós: Folyamatok (processzorokon)
- SSADM
  - Logikai adatmodell (entitás relációs diagram)
  - Adatfolyam diagram (processzek, adattárak)
  - Entitás (life history) diagram
  - Entitás hatás diagram
- Structured Analysis and Design Technique (SADT)
  - Aktivitás-faktor diagram: Tevékenységek + faktor relációk; bemenet, vezérlés, erőforrás, kimenet
- ROOM: Real-Time Object-Oriented Modeling

Félformális követelményspecifikáció: SysML

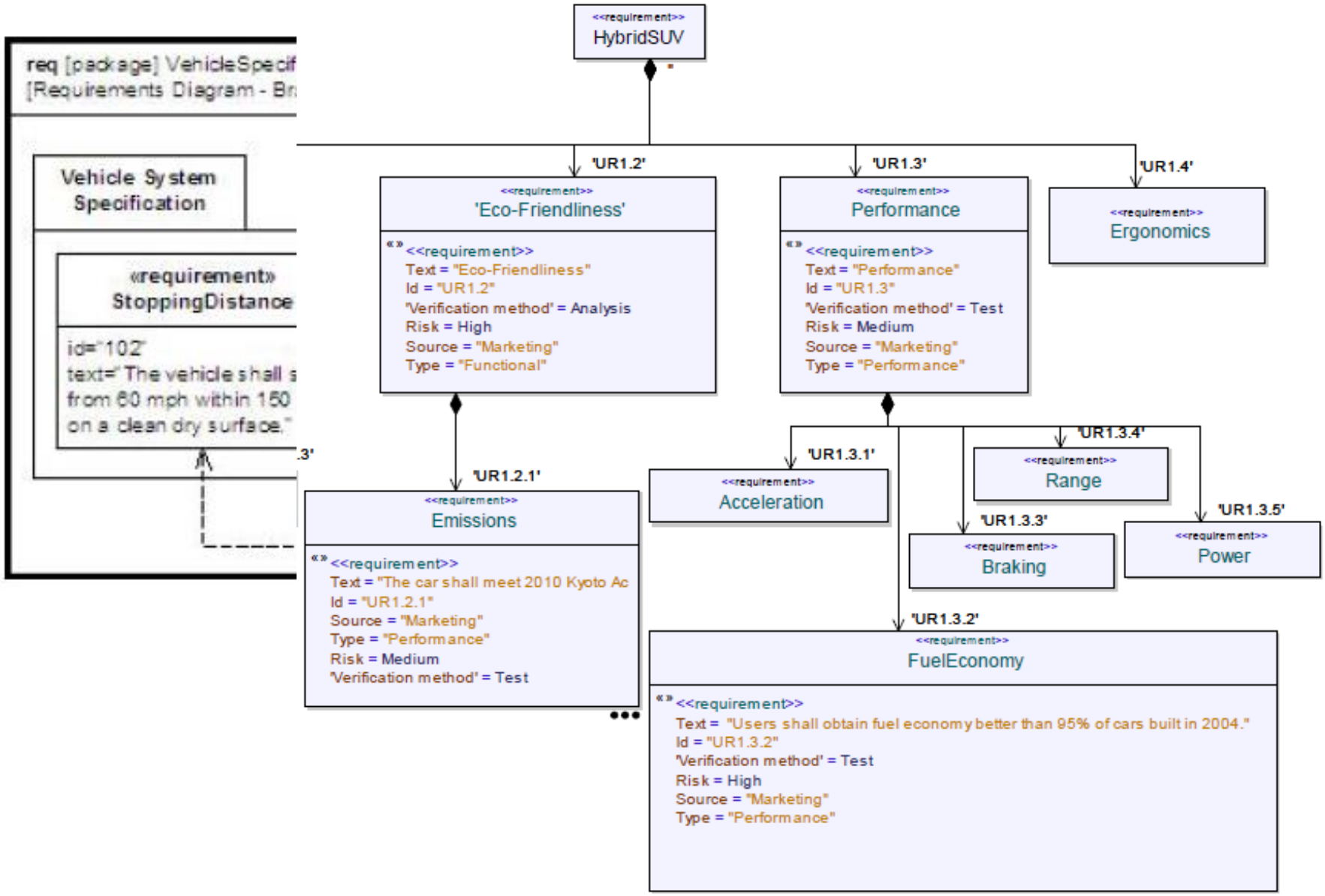
- Systems Modeling Language
  - UML részhalmazának egy kiterjesztése rendszertervezéshez
  - Fő újdonságok: Requirement és Parametric diagram



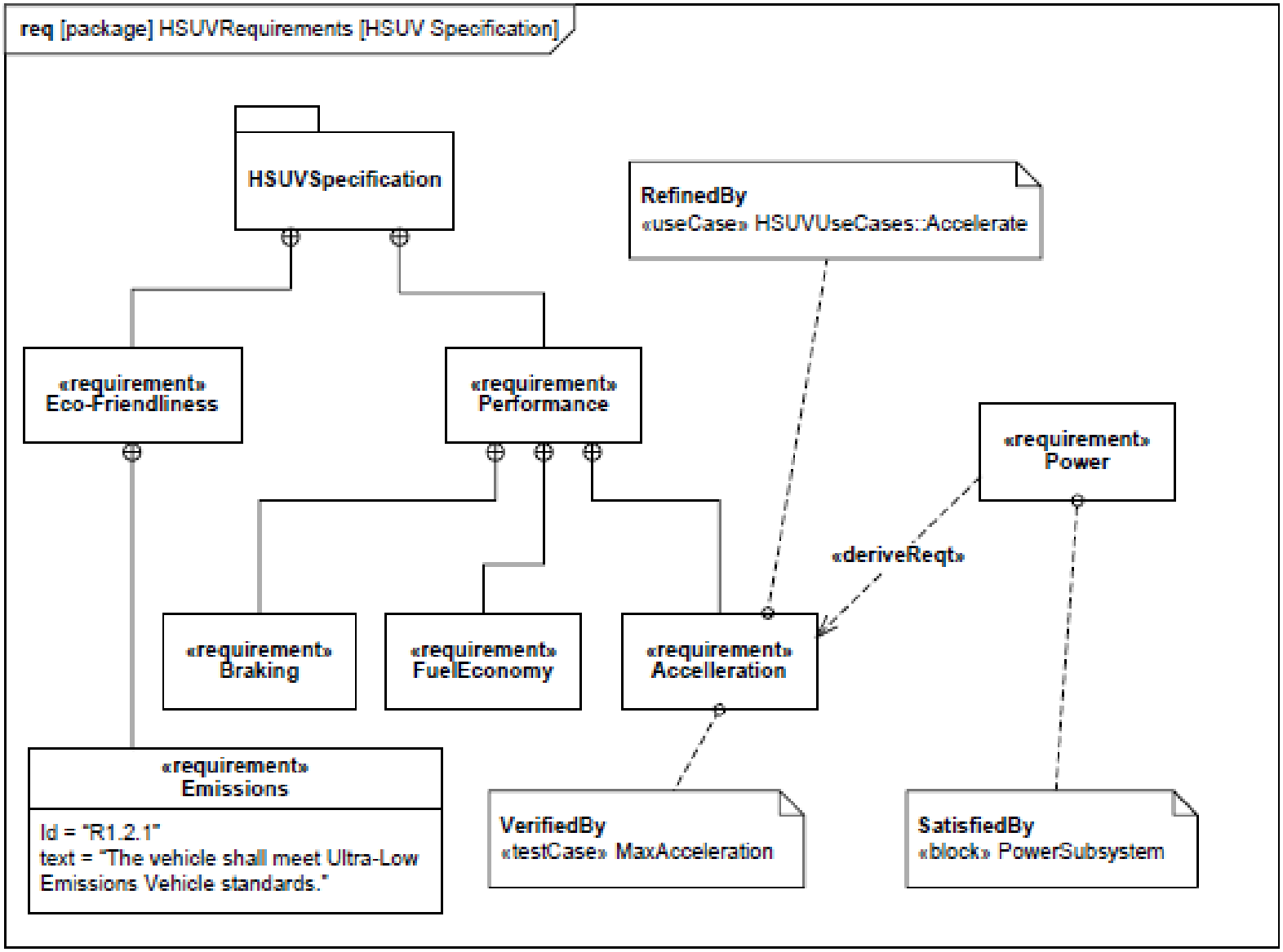
Requirement diagram

- 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

Requirements diagram példa: Struktúra

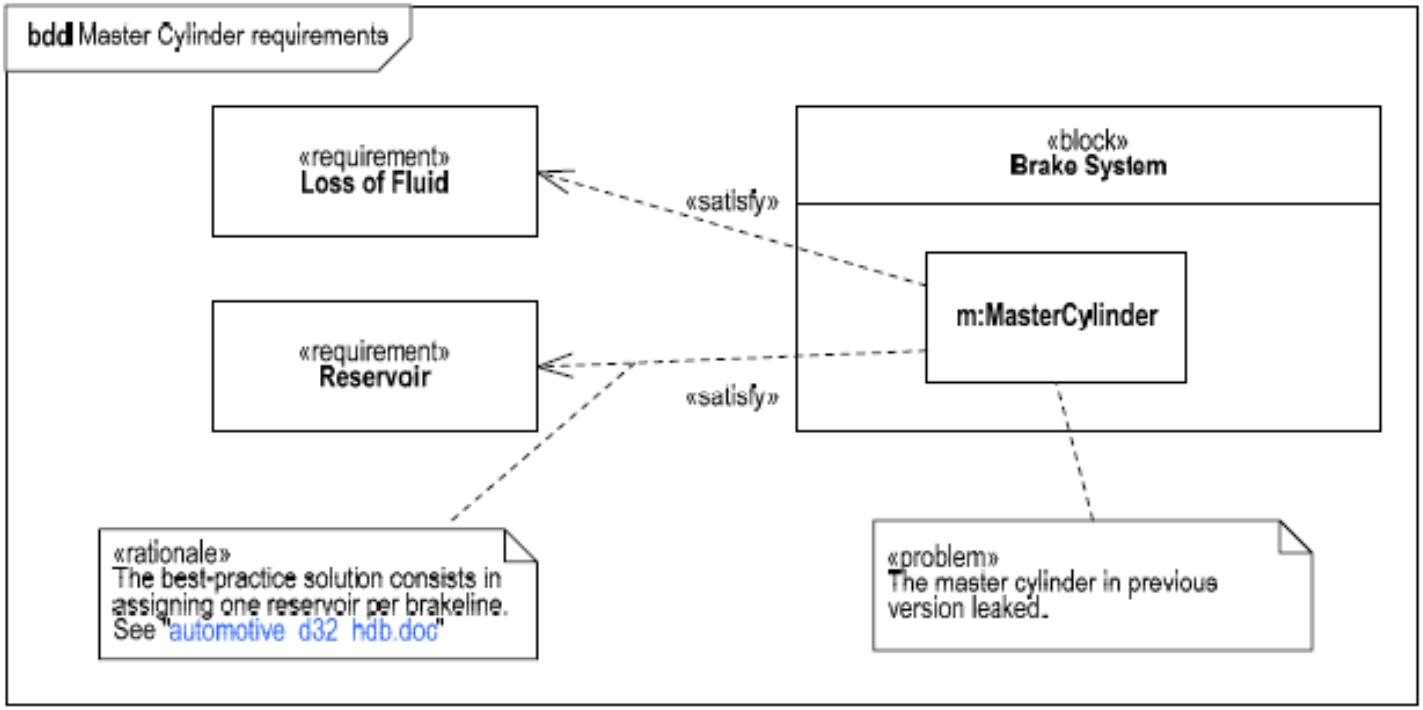


Requirements diagram példa: Relációk



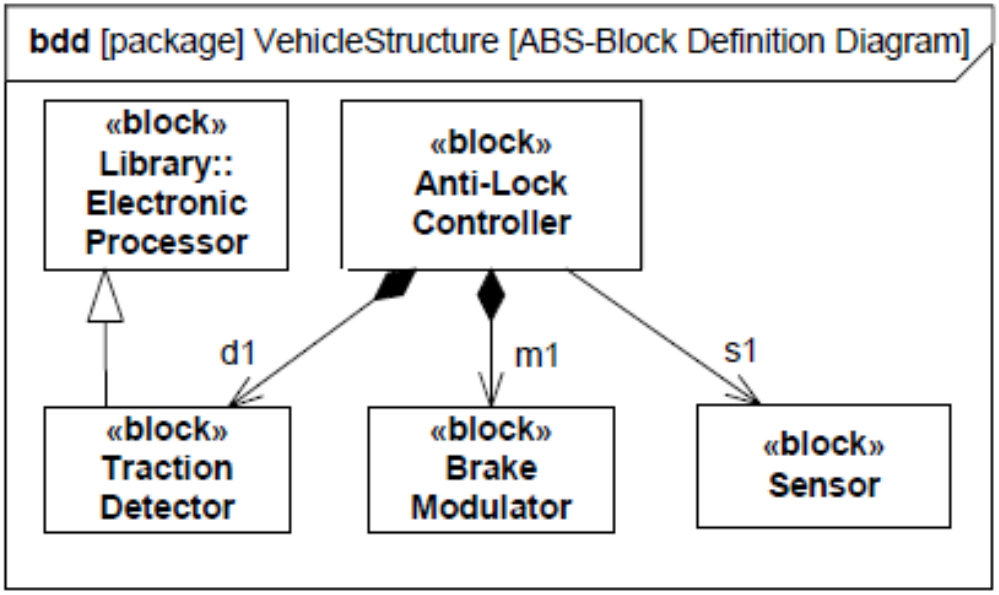
Requirements diagram: Döntések

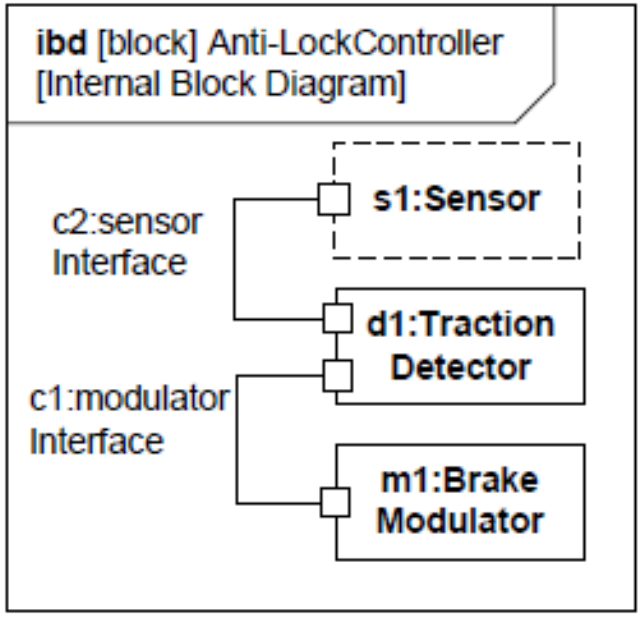
- Tetszőleges modell elemhez köthető speciális megjegyzések (előredefiniált stereotype):
  - <<problem>>: Probléma, döntést igénylő felvetés
  - <<rationale>>: Megoldás, magyarázat



Block diagram

- Block definition 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

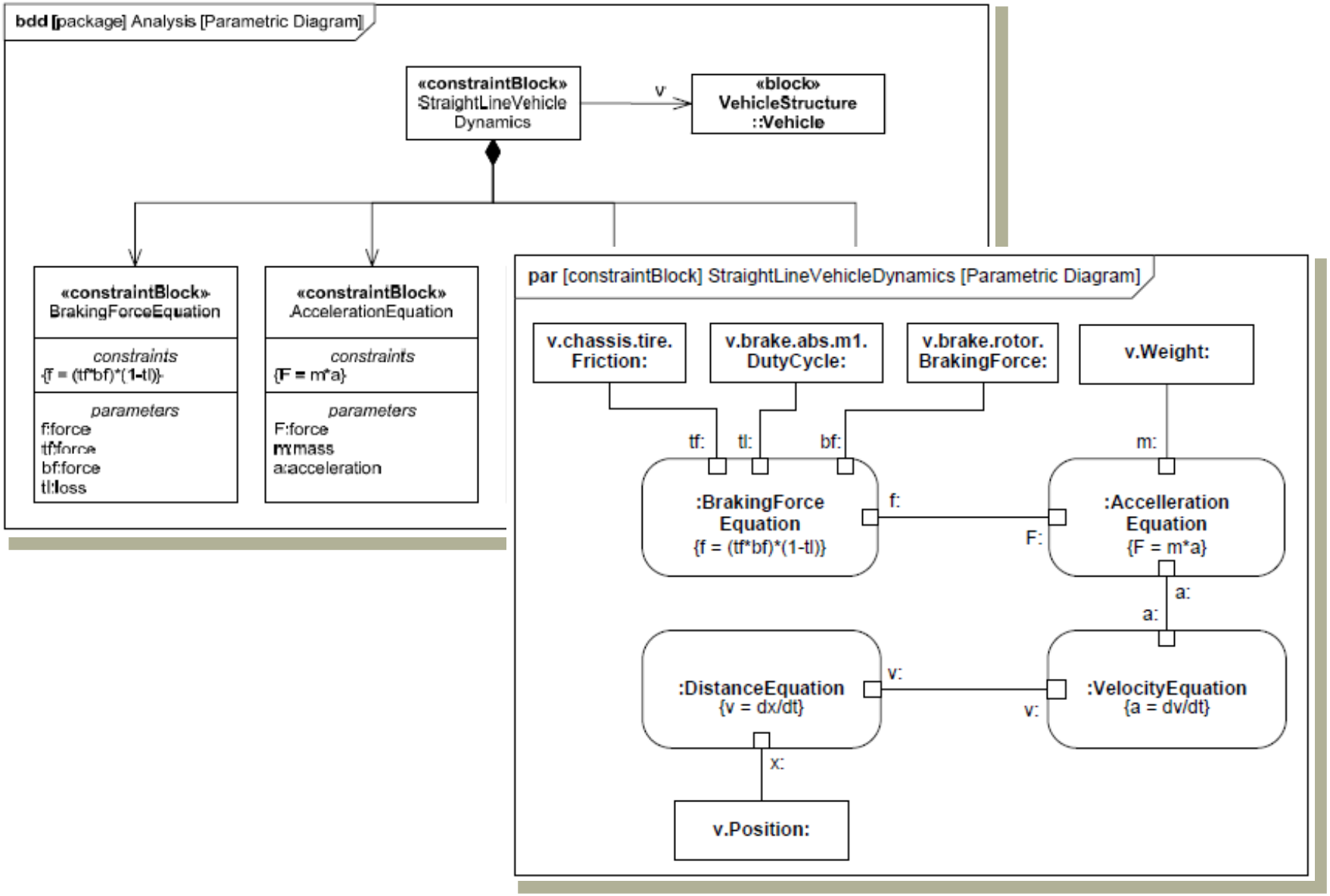




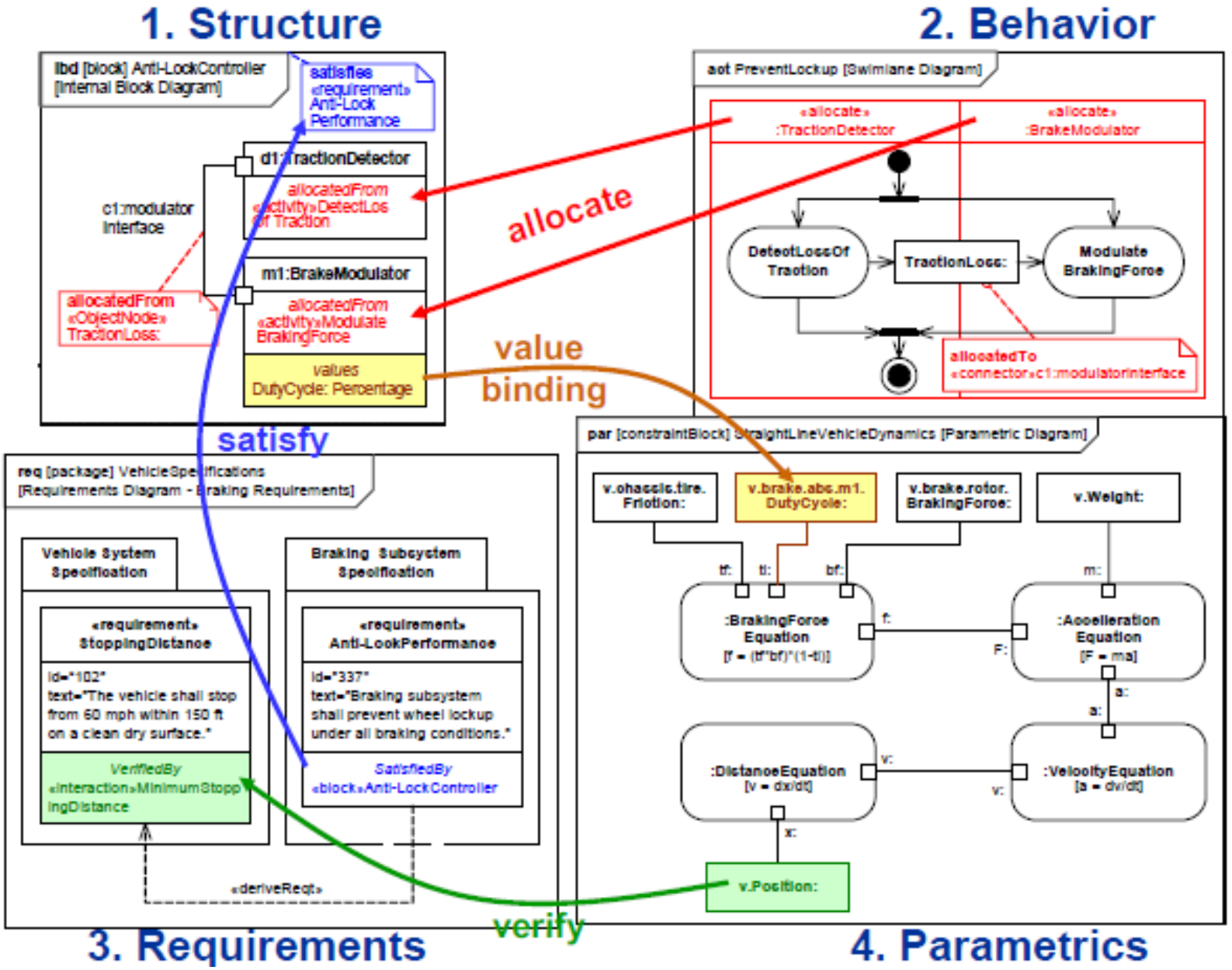
Parametric diagram

- 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

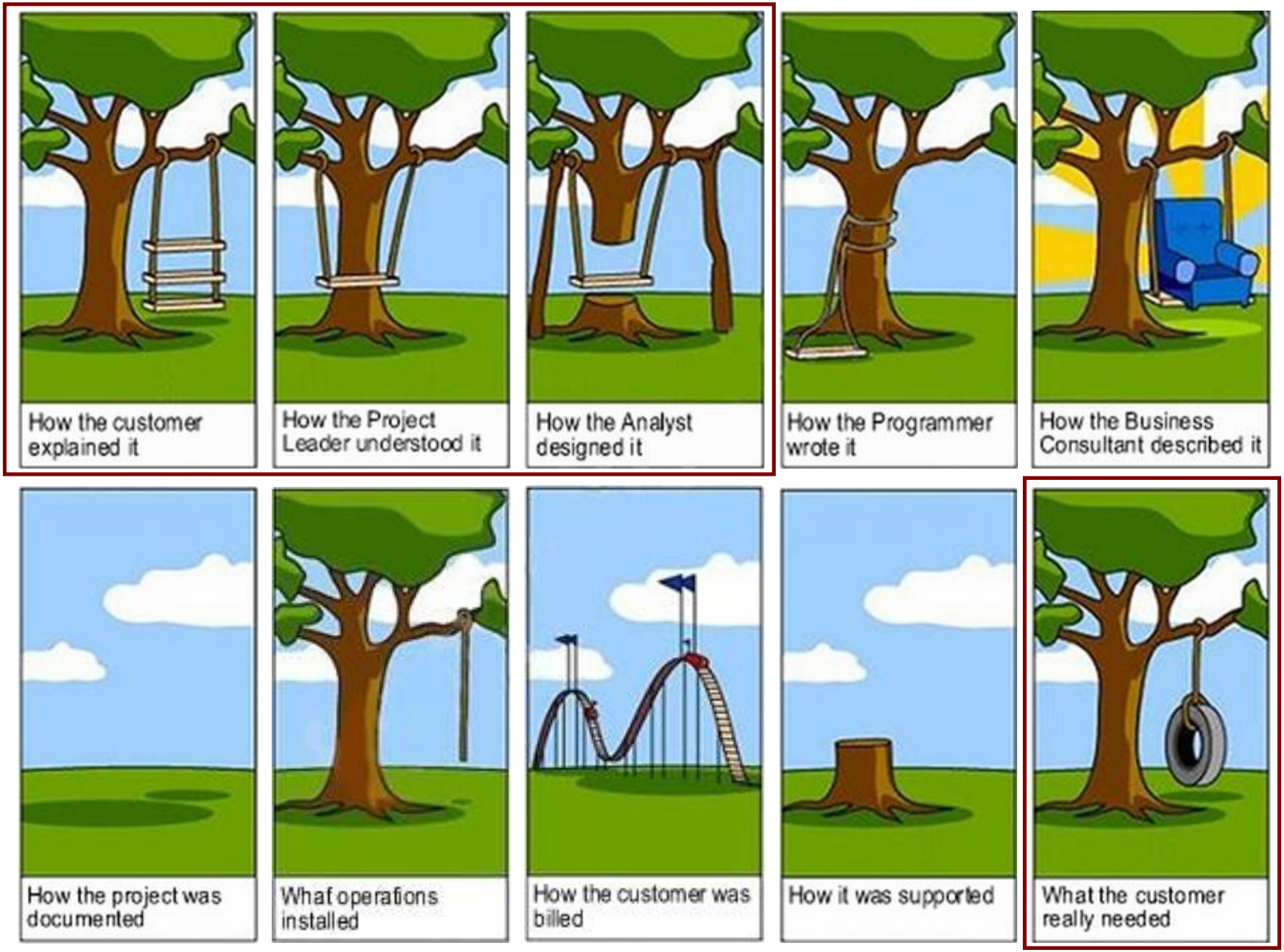


Tartalomjegyzék

- A fázis ki- és bemenetei
- A szoftverkövetelmény-specifikáció elkészítése
  - Formális nyelvek (áttekintés)
  - Félformális és strukturált technikák
  - Példa: SysML
- Ellenőrzési feladatok
  - Általános szempontok és módszerek
  - Teljesség és ellentmondás-mentesség
- Követelménykezelés
  - Feladatok
  - Automatikus eszközök

Verifikációs feladatok

- Szoftverkövetelmény-specifikáció ellenőrzése
  - Statikus analízis
    - Hiányosságok, ellentmondások kiszűrése végrehajtás nélkül
    - Analógia: Átolvasás „sorról sorra”
  - Ellenőrző listák
    - Tipikus hibák esetén hatékony (újra ne kövessük el)
    - Teljességet nem várhatunk
- Szoftverkövetelmény-tesztspecifikáció ellenőrzése
  - Végrehajtás (ki, mikor) és technika (hogyan) meghatározott-e?
  - Siker, kudarc ismérve adott-e?
- Megvalósítás:
  - Felülvizsgálat vagy egyenrangú átvizsgálás


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Általános ellenőrzési szempontok

- Teljesség
  - funkciók, hivatkozások, eszközök
- Konzisztencia
  - külső és belső
  - követhetőség
- Megvalósíthatóság
  - erőforrások
  - használhatóság
  - karbantarthatóság
  - kockázatok: költségbeli, technikai, környezeti
- Tesztelhetőség
  - specifikus
  - egyértelmű
  - számszerűsíthető

A jó specifikáció az IEEE 830-1998 alapján

- Helyes
  - A szoftverre vonatkozó követelményeknek (elvárásoknak) megfelelő
  - Konzisztens a külső forrásokkal (pl. szabványokkal)
- 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

A jó specifikáció az IEEE 29148-2011 alapján

- Szükséges
  - Kihagyása hiányosságot jelent, más követelmények nem pótolják, nem fedik le.
- Megvalósítástól független
  - Nem jelent szükségtelen megkötéseket (csak azt írja le, hogy mi az elvárás).
- Egyértelmű
  - Csak egyféleképpen értelmezhető, könnyen érthető.
- Konzisztens
  - Nincs konfliktusban más követelményekkel.
- Teljes
  - Nincs szükség további kiegészítésekre az érintettek igényeinek megértéséhez.
- Egyedi
  - Csak egy követelményt ír le (nincs benne konjunkció).
- Megvalósítható
  - Illeszkedik a meglévő technikai, költségbeli, ütemezésbeli, szabályzási keretek és kényszerek közé.
- Követhető
  - Visszafelé a felhasználói igényekhez, előrefelé a rendszer további elemeihez (terv, implementáció, teszt stb.) köthető.
- Ellenőrizhető
  - Megállapítható, hogy a rendszer teljesíti-e a követelményt (erről bizonyosság szerezhető, pl. teszteléssel, elemzéssel, mérésekkel).

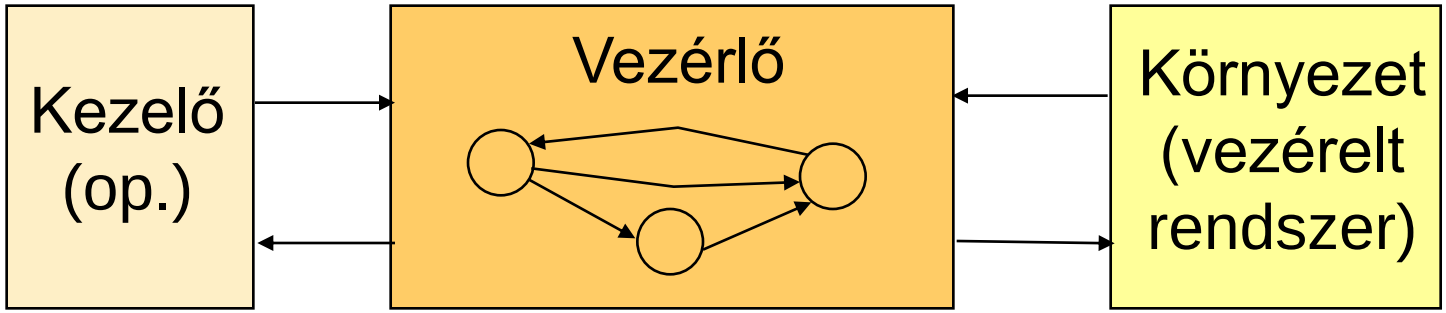
Teljesség és ellentmondás-mentesség

- Motiváció: Sok hiba visszavezethető hiányos vagy ellentmondásos specifikációra
  - Példa: Voyager és Galileo űrszondák szoftver tesztelése során felfedezett hibák statisztikája

    78% (149/192) specifikációs hiányosságból adódott, ebből
    - 23% veszélyes állapotban ragadás (nincs kilépés)
    - 16% időzítési kényszerek megadásának hiánya
    - 12% nincs specifikált reakció külső eseményre
    - 10% bemeneti érték ellenőrzésének hiánya
- Megoldás lehet:
  - Szigorú specifikációs nyelv
  - Ellenőrzött tervezési minták használata
  - Specifikáció ellenőrzése (verifikáció)

Példa: Reaktív rendszerek specifikációjának ellenőrzése (N. Leveson kritériumai)

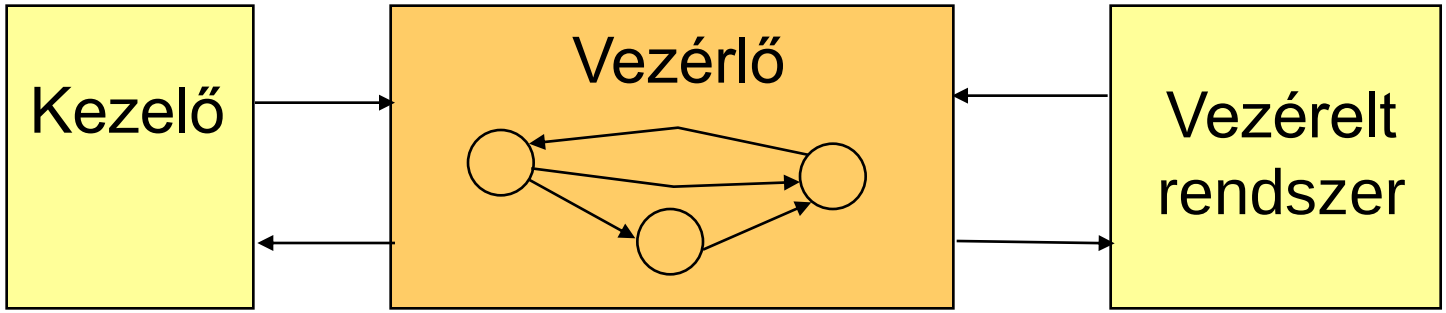
- Állapotdefiníció
- Bemenetek (események) specifikációja
- Kimenetek specifikációja
- Kimenetek és trigger kapcsolata
- Állapotátmenetek specifikációja
- Ember-gép interfész specifikációja



Vizsgálati szempontok

- Állapotdefiníció
- Beme
- Kime
- Kime
- Állap
- Ember-gép interfész

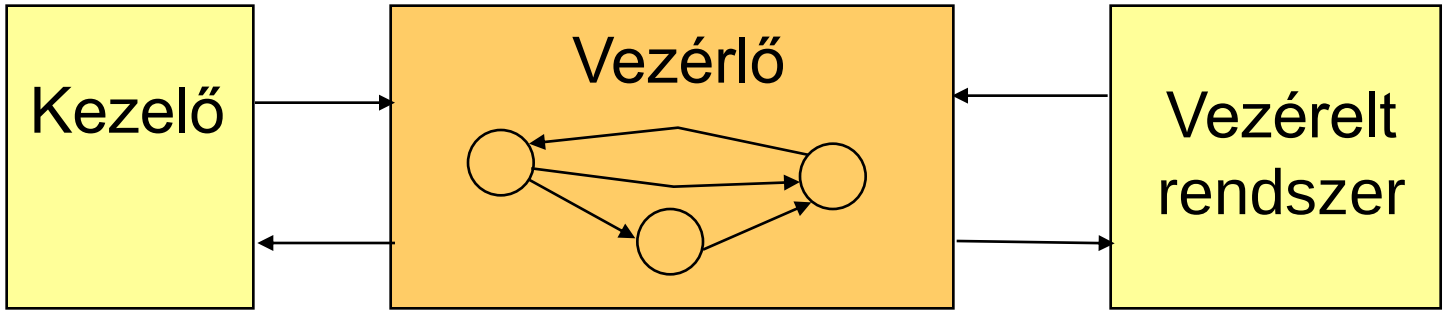
- Biztonságos a kezdőállapot
- Belső modell aktualizálása biztosított
  (kimaradó bemeneti események esetén
  time-out és nincs vezérlés)



Vizsgálati szempontok

- Állapotdefiníció
- Bemenetek (események)
- Kimen
- Kime
- Állap
- Embe

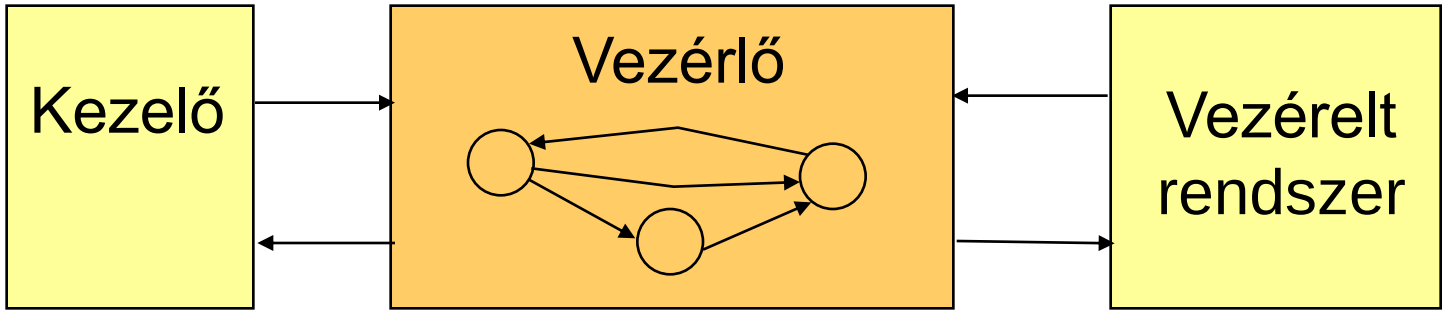
- Minden bemenetre van megadott reakció
- Egyértelmű (determinisztikus) reakciók
- Van bemeneti ellenőrzés (érték, idő)
- Hibás bemenet kezelése specifikált
- Megszakítások gyakorisága specifikált



Vizsgálati szempontok

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

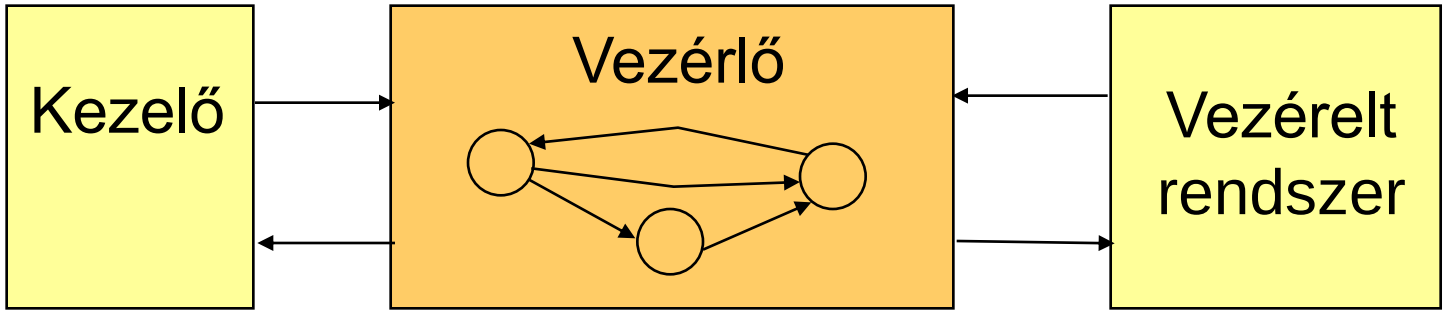
- Hihetőség-vizsgálat kritériumai adottak
- Fel nem használt kimenetek nincsenek
- Környezeti feldolgozó kapacitás betartva



Vizsgálati szempontok

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

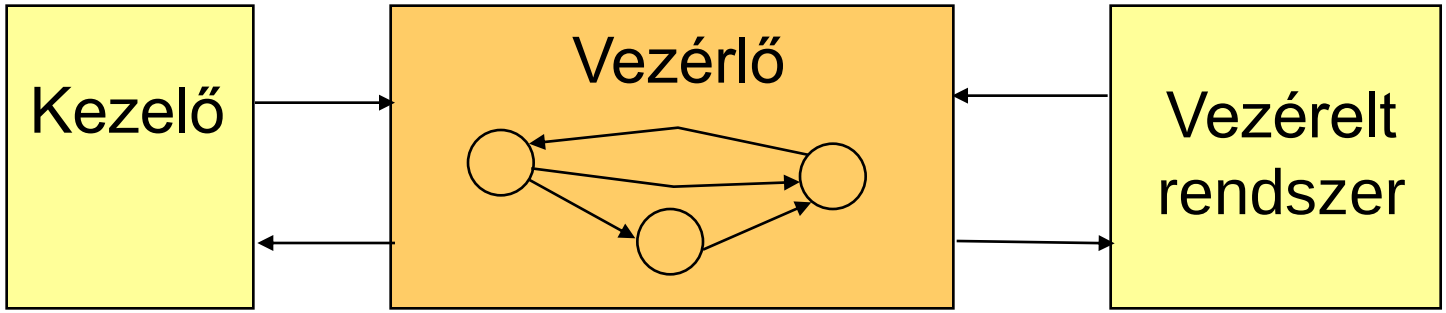
- Kimenetek hatása ellenőrizve van a bemeneteken keresztül
- Szabályzási kör stabilitása biztosított



Vizsgálati szempontok

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

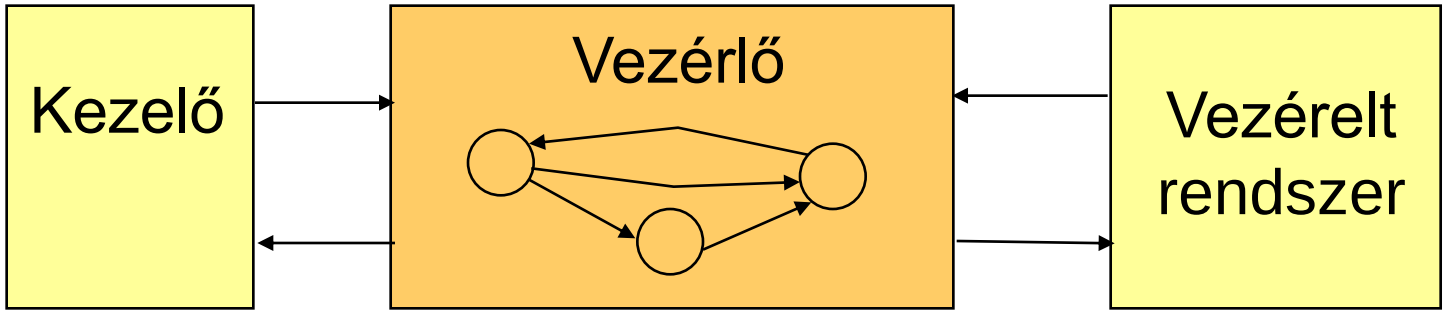
- Minden állapot statikusan elérhető
- Állapotátmenetek visszafordíthatók
- Több átmenet van veszélyes állapotból biztonságosba
- Megerősített átmenet van biztonságosból veszélyes állapotba



Vizsgálati szempontok

- Állapotátmenetek
- Ember-gép interfész

Kezelő felé kimenő események specifikációja:
- Sorrendezés (prioritás) definiált
- Frissítés definiált
- Időbeliség (kezelő terhelhetősége) figyelembe véve



Példa: UML állapottérkép modell vizsgálata

Specifikációs hiányosságok ellenőrzése:

Teljesség:

1. Minden eseményre, minden állapotban van specifikált viselkedés (itt: állapot-konfigurációk)
2. Egy eseményre minden állapotban tautológiát képeznek az őrfeltételek (itt: őrfeltétel kiértékelés)

. . .

Ellentmondás-mentesség:

1. Egy eseményre minden állapotban csak egy átmenet lehet engedélyezett (itt: állapot-konfigurációk)
2. Minden állapotban van egy és csakis egy „time-out” triggerelt átmenet

. . .

Tartalomjegyzék

- A fázis ki- és bemenetei
- A szoftverkövetelmény-specifikáció elkészítése
  - Formális nyelvek (áttekintés)
  - Félformális és strukturált technikák
  - Példa: SysML
- Ellenőrzési feladatok
  - Általános szempontok és módszerek
  - Teljesség és ellentmondás-mentesség
- Követelménykezelés
  - Feladatok
  - Automatikus eszközök

A követelménykezelés feladatai (áttekintés)

- Követelmények hatékony, strukturált tárolása
  - Hierarchikus elrendezés, tulajdonságokkal
- Követelmény életciklus támogatása
  - Felvétel, törlés, változás, finomítás, kapcsolatok megjelenése
- Követelmények finomítása és követhetősége
  - Specifikáció -> Architektúra terv -> Modulterv -> Forráskód -> Teszt -> Teszt eredmény
- 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 / tesztelve?

A követelménykezelés „kézi” módszerei

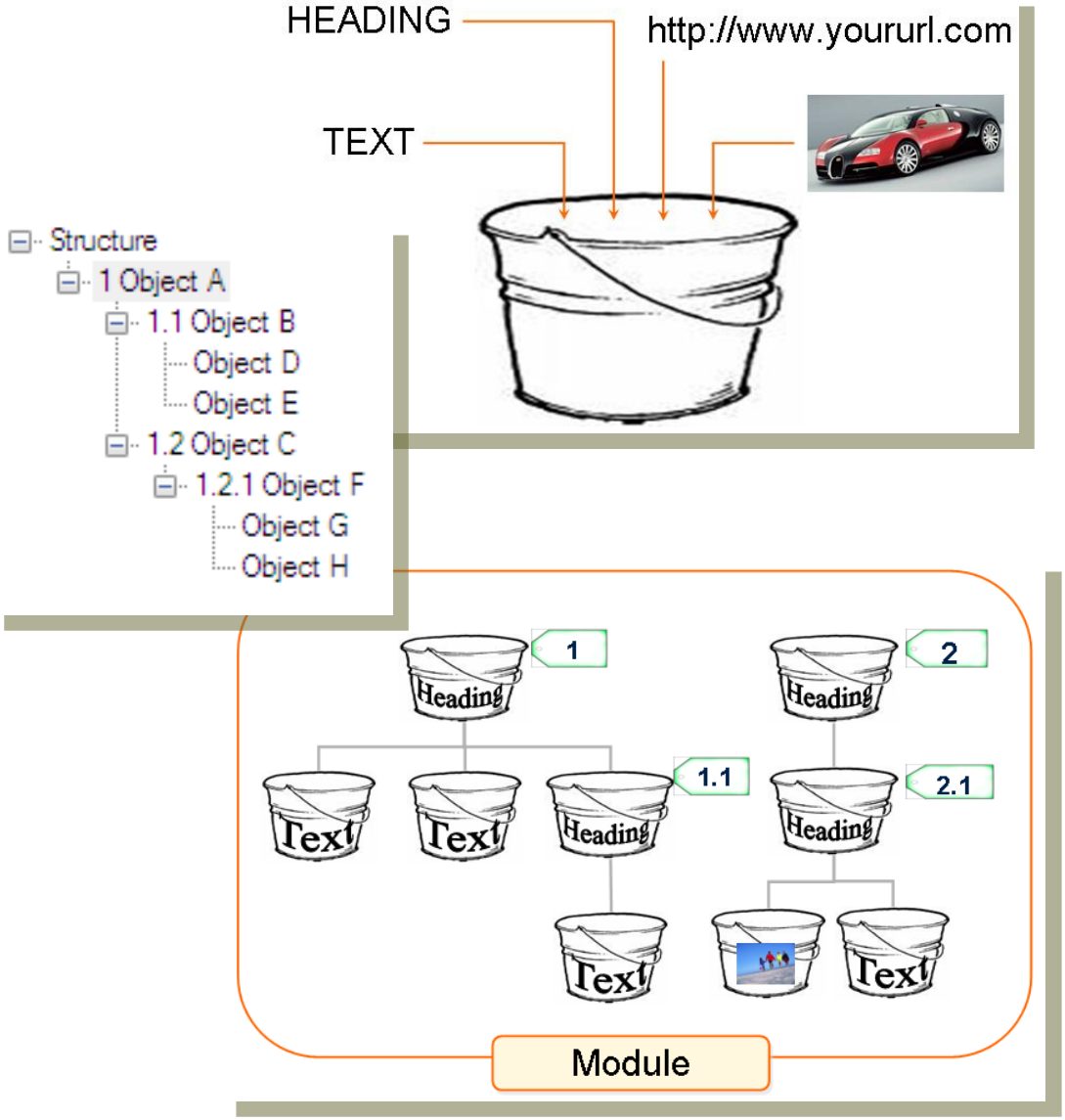
- Természetes nyelvű követelmény dokumentum
  - Strukturálás adott (fejezetek, alfejezetek)
  - Követelményazonosítók felvétele
- Követelményfinomítás: Táblázatos nyilvántartás
  - Követelményazonosítók szerepelnek
    - Különböző dokumentumokból (SRS, SA, MDS, MTS, ...)
  - Analízis makrókkal támogatható
    - Üres, többszörös, árva ... mezők kikeresése
- Követhetőségi mátrix: Táblázatos összerendelés
  - 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. eredet analízishez
Fedettségi listák készítése:	Lefedetlen követelményekhez Indokolatlan megvalósításhoz
Jogosultságok kezelése:	Hozzáférés szerepek
Értesítési rendszer:	Változások
Biztonsági megoldások:	Sértetlenség

Megvalósítás: Strukturált tárolás

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



Példa: IBM Rational 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 te 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**	Manda		
TRN-CSR-7	Bill Young	People come in all shapes and s es. The car must be suitable for people maxi um and minimum sizes fgfgf to 2 m ighing 25 kilograms t	Manda		

Tulajdonságok

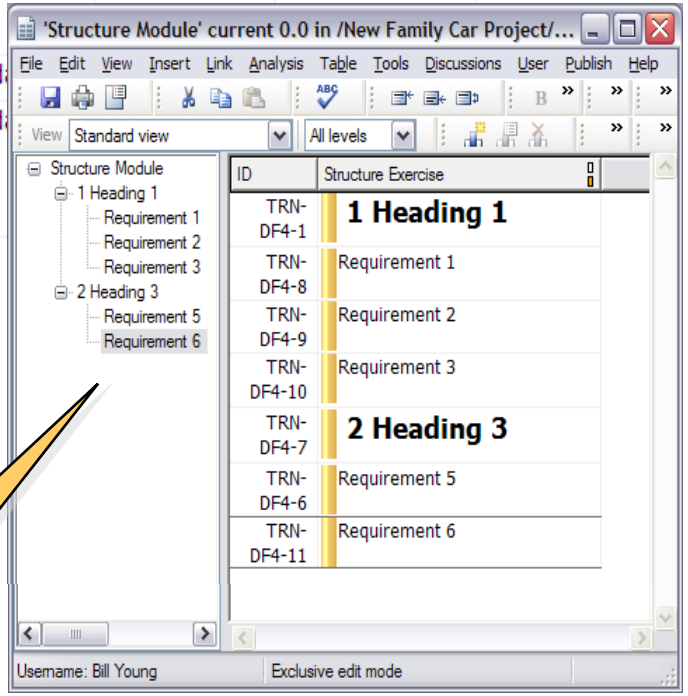
Objektum azonosító

Változás-jelző

Fejléc objektum

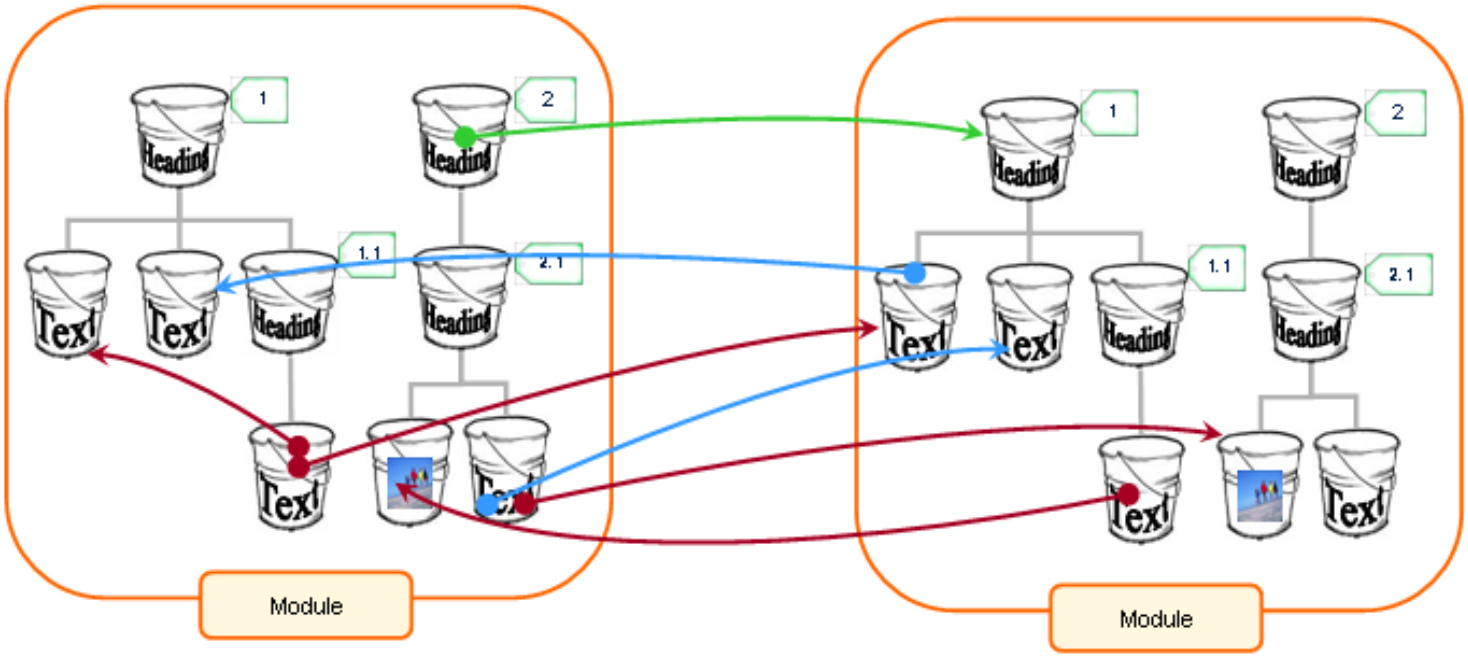
Szöveges objektum

Hierarchia

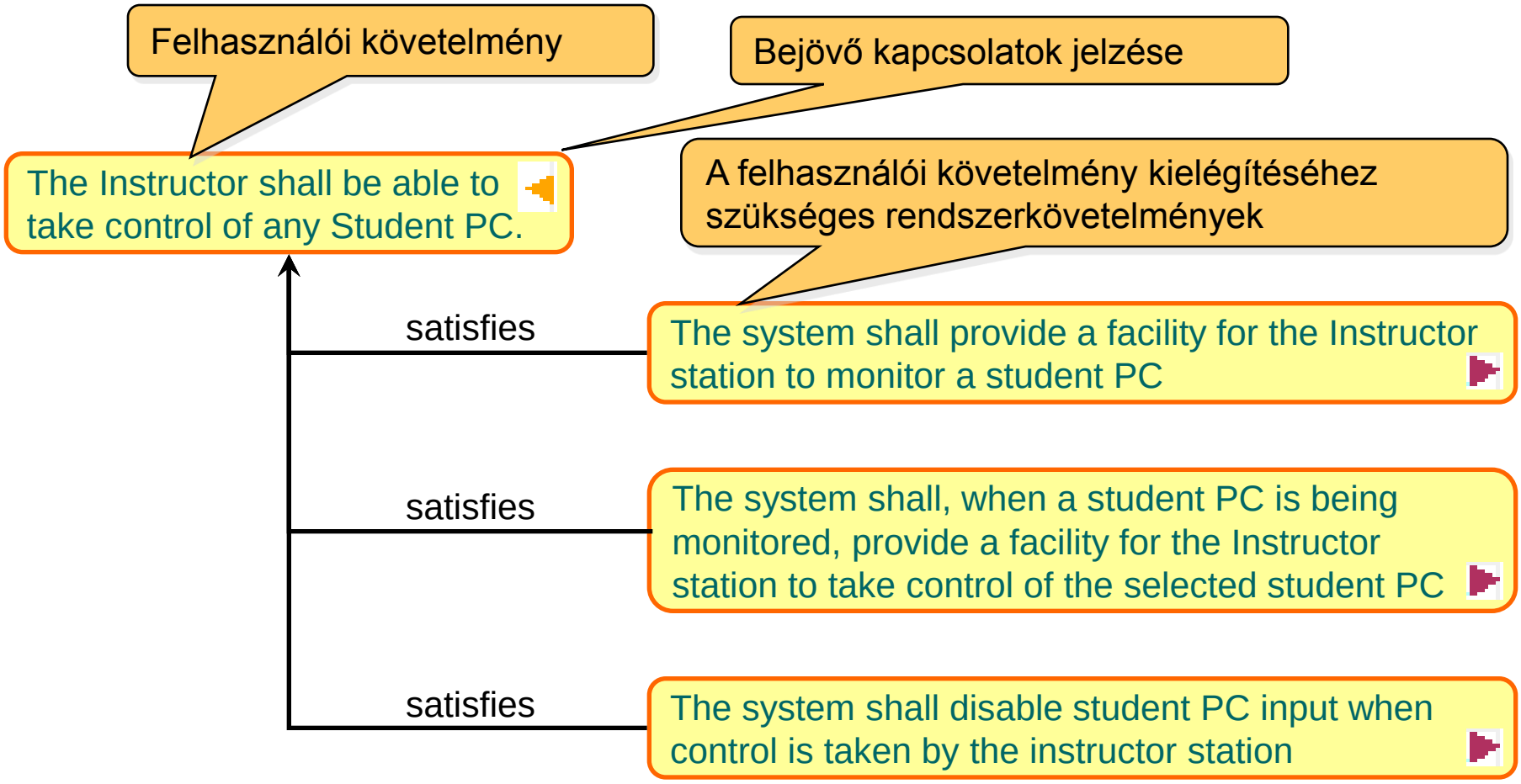


Megvalósítás: Kapcsolatok (linkek)

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



Példa: DOORS


Felhasználói követelmény
Bejövő kapcsolatok jelzése
The Instructor shall be able to take control of any Student PC.
A felhasználói követelmény kielégítéséhez szükséges rendszerkövetelmények
satisfies
The system shall provide a facility for the Instructor station to monitor a student PC
satisfies
The system shall, when a student PC is being monitored, provide a facility for the Instructor station to take control of the selected student PC
satisfies
The system shall disable student PC input when control is taken by the instructor station

Megvalósítás: Követelmény életciklus

- Objektum szintű változások
  - Közvetlen szerkesztés (létrehozás, módosítás, törlés)
  - Megváltozott kapcsolatok jelzése (suspect link)
  - Változtatási javaslatok menedzselése (elkészítés, csoportosítás, felülvizsgálat, érvényre juttatás)
- Változások mint események
  - Scriptek indítására használhatók (pl. teszt érvénytelenítés)
- 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

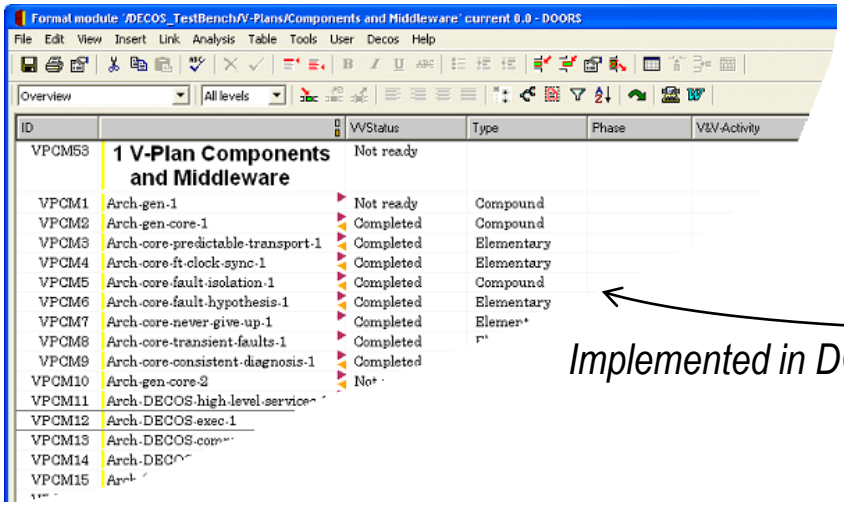
Megvalósítás: Járulékos funkciók

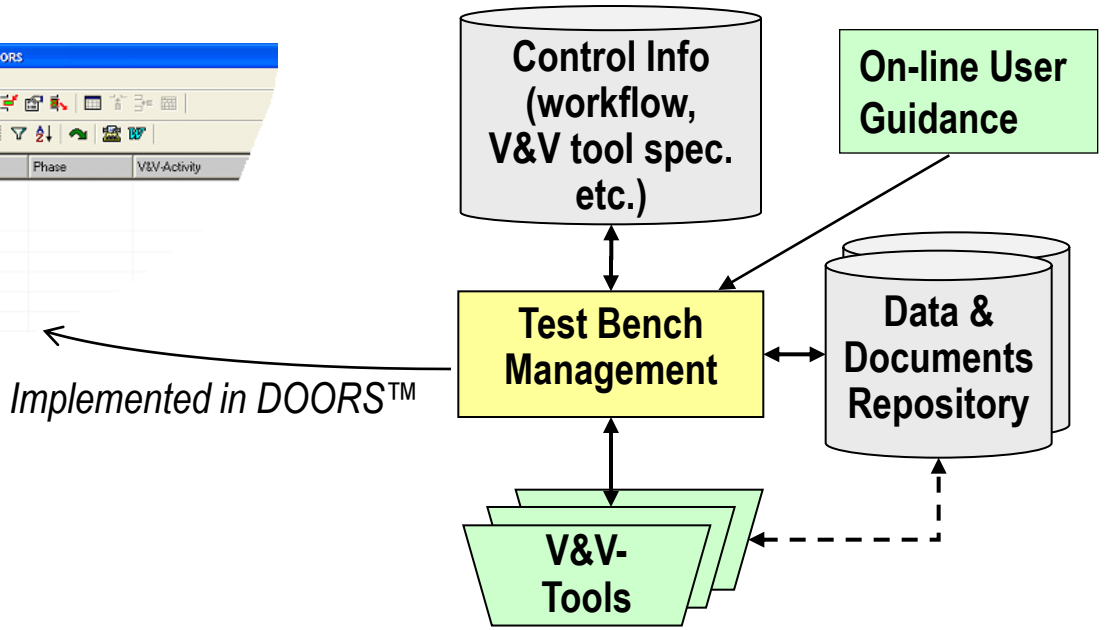
- Nézetek az objektum (követelmény) listában
  - Szűrés hierarchiaszintekre, tulajdonságokra, …
- Űrlapok a tulajdonságok szerkesztésére
- Megosztott szerkesztés (csoportmunka)
  - Objektum (követelmény) szintű zárolás
- Dokumentáció generálás
  - Hierarchia (=> fejezetek) alapján rendezett szöveges 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

Követelmény alapú verifikációs eszközláncok

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

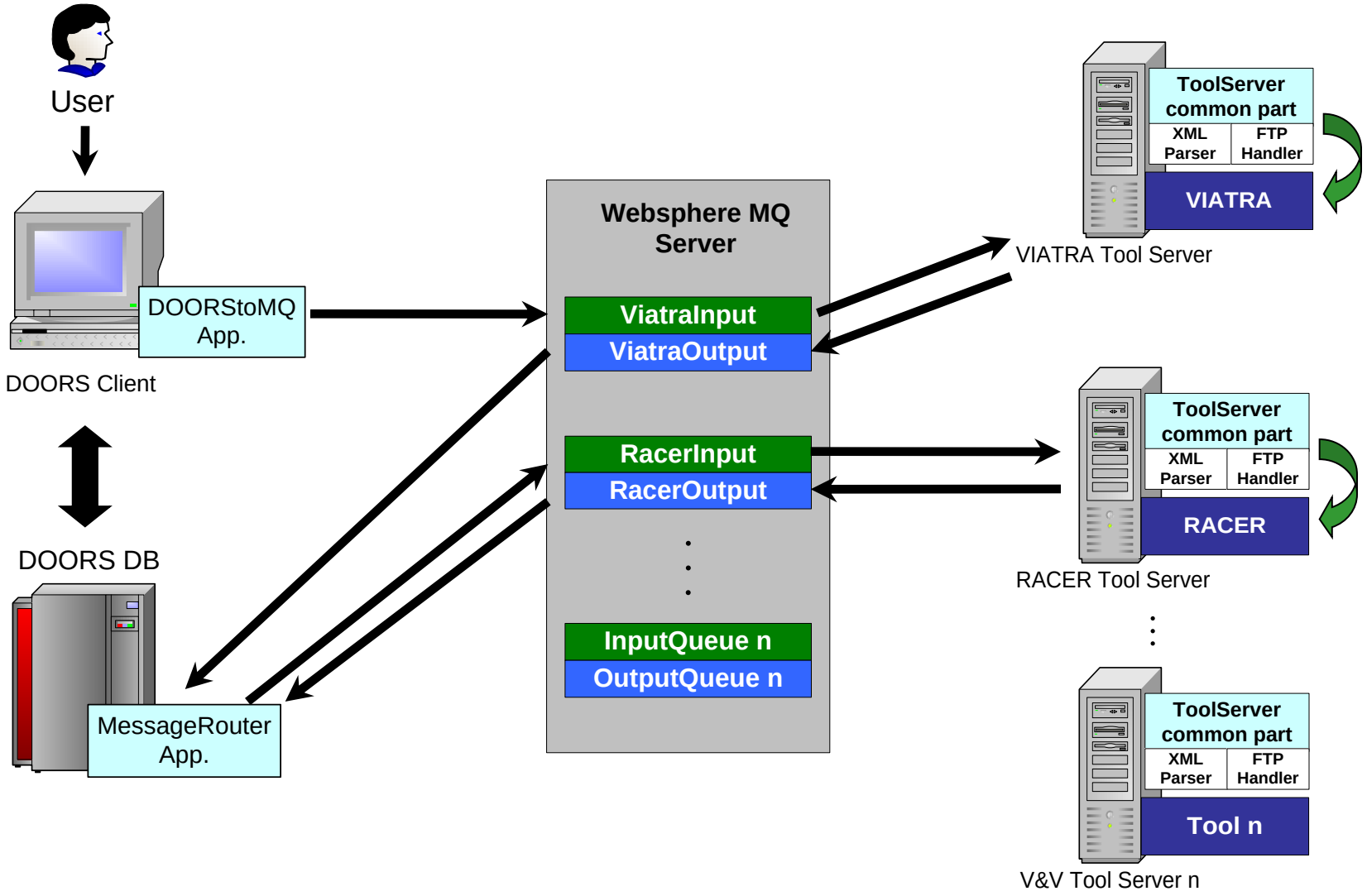
Verifikáció „menedzselése” a követelménykezelővel





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

Példa: Eszközlánc végrehajtása



Összefoglalás

- A fázis ki- és bemenetei
- A szoftverkövetelmény-specifikáció elkészítése
  - Formális nyelvek (áttekintés)
  - Félformális és strukturált technikák
  - Példa: SysML
- Ellenőrzési feladatok
  - Általános szempontok és módszerek
  - Teljesség és ellentmondás-mentesség
- Követelménykezelés
  - Feladatok
  - Automatikus eszközök