



Modellalapú szoftverfejlesztés

XI. előadás

Modellalapú
fejlesztések

Dr. Semeráth Oszkár

Modellalapú fejlesztések

I. Fejlesztési fogalmak

II. Kritikus rendszerek fejlesztése

III. Funkciómodellezés

IV. Generatív programozás

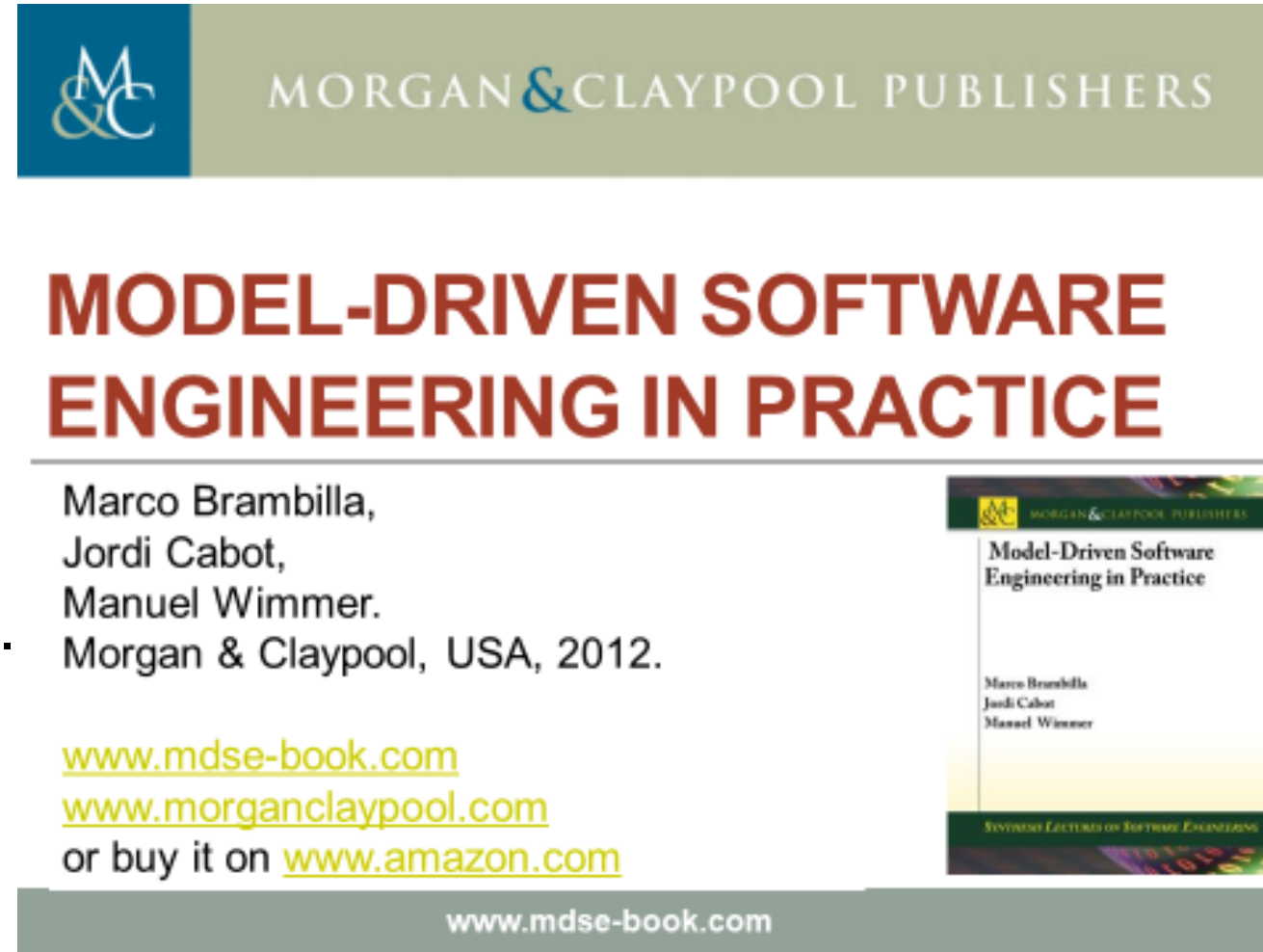
V. Parciális modellezés



Az MDSE hagyományos motivációi

Alapelvek és célkitűzések

- Melyek a modellalapú fejlesztés motivációi?
- Milyen kérdések merülnek fel?
- Ezt a rész fejezetet egy tankönyvből vettük át.
- A szerzők beleegyeztek abba, hogy az oktatásban használjuk a könyvüket.



www.mdse-book.com



Az MDSE hagyományos motivációi

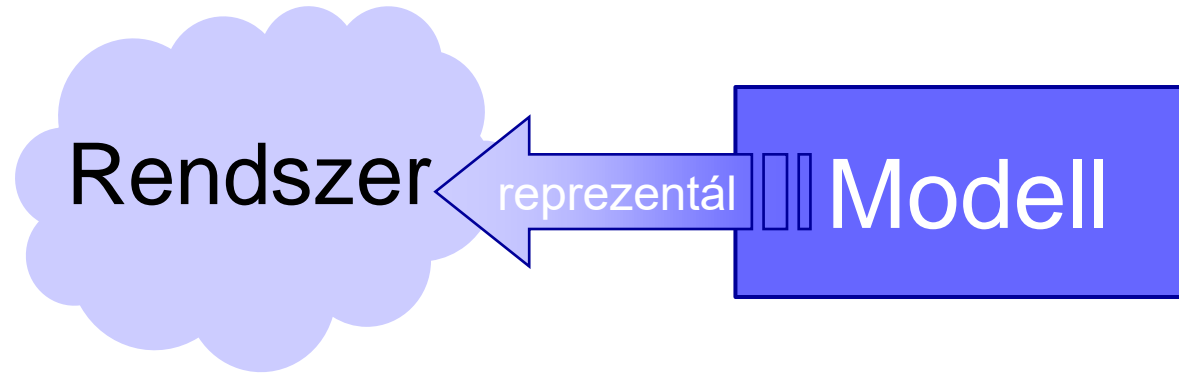
Alapelvek és célkitűzések

- **Absztrakció** a konkrét megvalósítási technológiáktól
 - Olyan modellező nyelveket igényel, amelyek nem tartalmazzák a megvalósítási technológiák specifikus fogalmait (pl. Java EJB).
 - A szoftverek jobb **hordozhatósága** új/változó technológiákhoz – model once, build everywhere
 - A különböző technológiák közötti **átjárhatóság** automatizálható (ún. Technológiai Hidak / Technology Bridges)
- **Automatizált kódgenerálás** absztrakt modellekből
 - pl. Java-API-k, XML-sémák stb. generálása UML-ből
 - Kifejező és pontos modelleket igényel
 - Fokozott **termelékenység** és **hatékonyság** (a modellek naprakészek maradnak)
- Az alkalmazás és az infrastruktúra **különálló fejlesztése**
 - Az alkalmazáskód és az infrastruktúrakód (pl. Application Framework) szétválasztása növeli az **újrafelhasználhatóságot**
 - **Rugalmas** fejlesztési ciklusok, valamint **különböző fejlesztési szerepek lehetségesek**



Modellek

Mi az a modell?



Leképezési Jellemző

A modell egy eredeti (=rendszer) alapján készül

Redukciós Jellemző

A modell az eredeti tulajdonságainak csak egy (releváns) részét tükrözi

Pragmatikus Jellemző

A modellnek használhatónak kell lennie az eredeti helyett valamilyen cél szempontjából

Célok:

- leíró célok
- előíró célok



MDSE Egyenlet

Modellek + Transzformációk = Szoftver



Modeling Languages

- **Szakterület-specifikus nyelvek (DSL):** olyan nyelvek, amelyeket kifejezetten egy adott szakterületre vagy kontextusra terveztek
- A DSL-eket nagymértékben használják az informatikában. Példák: HTML, Logo, VHDL, Mathematica, SQL
- **Általános célú modellező nyelvek** (GPML-ek, GML-ek vagy GPL-ek): olyan nyelvek, amelyek bármely ágazatban vagy területen alkalmazhatók (szoftver)modellezési célokra
- A tipikus példák: UML, Petri-hálók vagy állapotgépek



A modell típusai

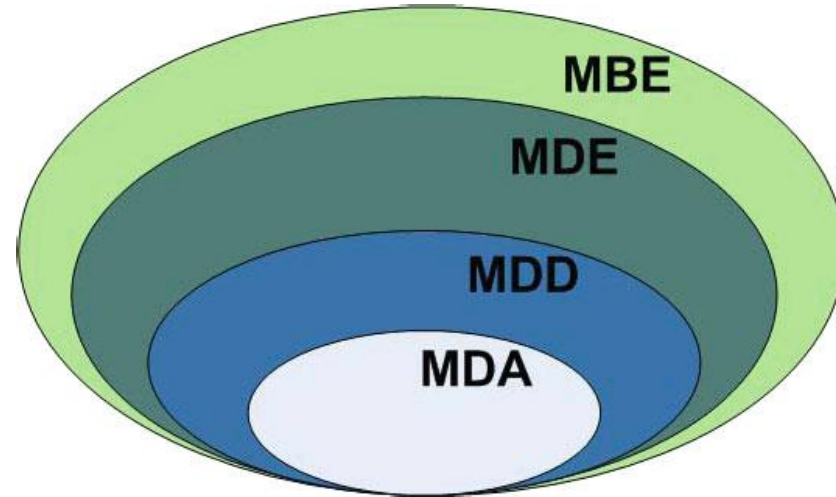
- **Statikus modellek:** A rendszer statikus aspektusaira összpontosítanak a kezelt adatok, valamint a rendszer szerkezeti formája és felépítése szempontjából.
- **Dinamikus modellek:** A rendszer dinamikus viselkedését hangsúlyozzák a végrehajtás bemutatásával.
- **Runtime modellek:** A rendszer működés közbeni állapotát mutatják be.
- Gondoljunk csak az UML-re!

Felhasználás / Cél:

- Nyomonkövethetőségi Modellek
- Végrehajtáskövetési Modellek
- Elemzési Modellek
- Szimulációs Modellek



A rövidítések MD* dzsungel



- **A modellvezérelt fejlesztés (Model-Driven Development, MDD)** egy olyan fejlesztési paradigma, amely modelleket használ a fejlesztési folyamat elsődleges tárgyaként.
- **A modellvezérelt architektúra (Model-Driven Architecture, MDA)** az MDD sajátos elképzelése, amelyet az Object Management Group (OMG) javasolt.
- **A modellvezérelt tervezés (Model-Driven Engineering, MDE)** az MDD egy szuperhalmaza, mivel túlmutat a pusztán fejlesztésen.
- **A modellalapú tervezés** (vagy "modellalapú fejlesztés") **(Model-Based Engineering, MBE)** az MDE lazább változata, ahol a modellek nem "irányítják" a folyamatot



Modellezési Szintek

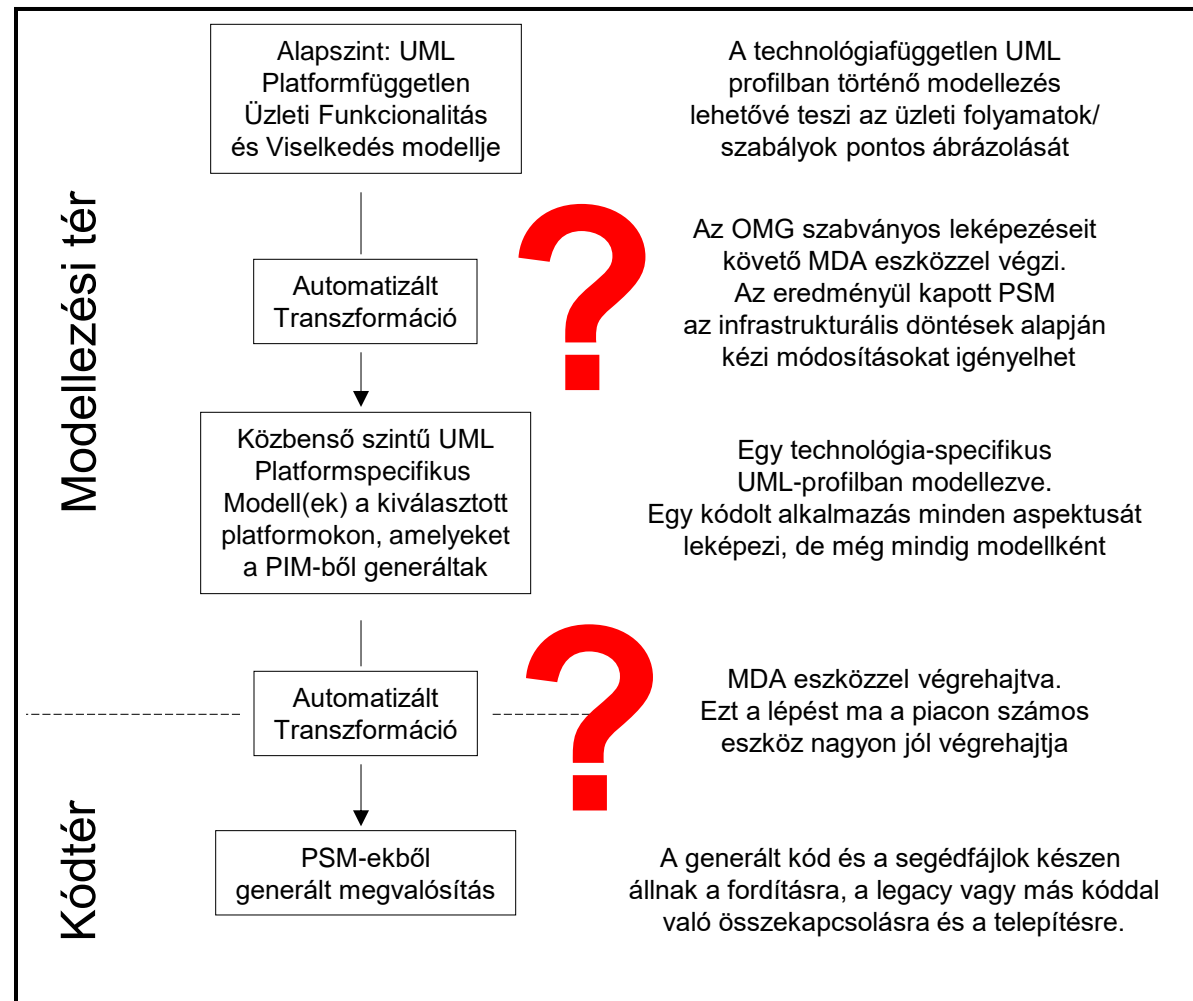
CIM, PIM, PSM

- **Számításfüggetlen (CIM):** a követelmények és igények leírása nagyon absztrakt szinten, a megvalósítás aspektusaira való hivatkozás nélkül (pl. felhasználói követelmények vagy üzleti célok leírása);
- **Platformfüggetlen (PIM):** a rendszerek viselkedésének meghatározása a tárolt adatok és a végrehajtott algoritmusok szempontjából, minden technikai vagy technológiai részlet nélkül;
- **Platformspecifikus (PSM):** minden technológiai szempont részletes meghatározása.



Az MDA Megközelítés

MDA fejlesztési ciklus



Modellalapú fejlesztések

I. Fejlesztési fogalmak

II. Kritikus rendszerek fejlesztése

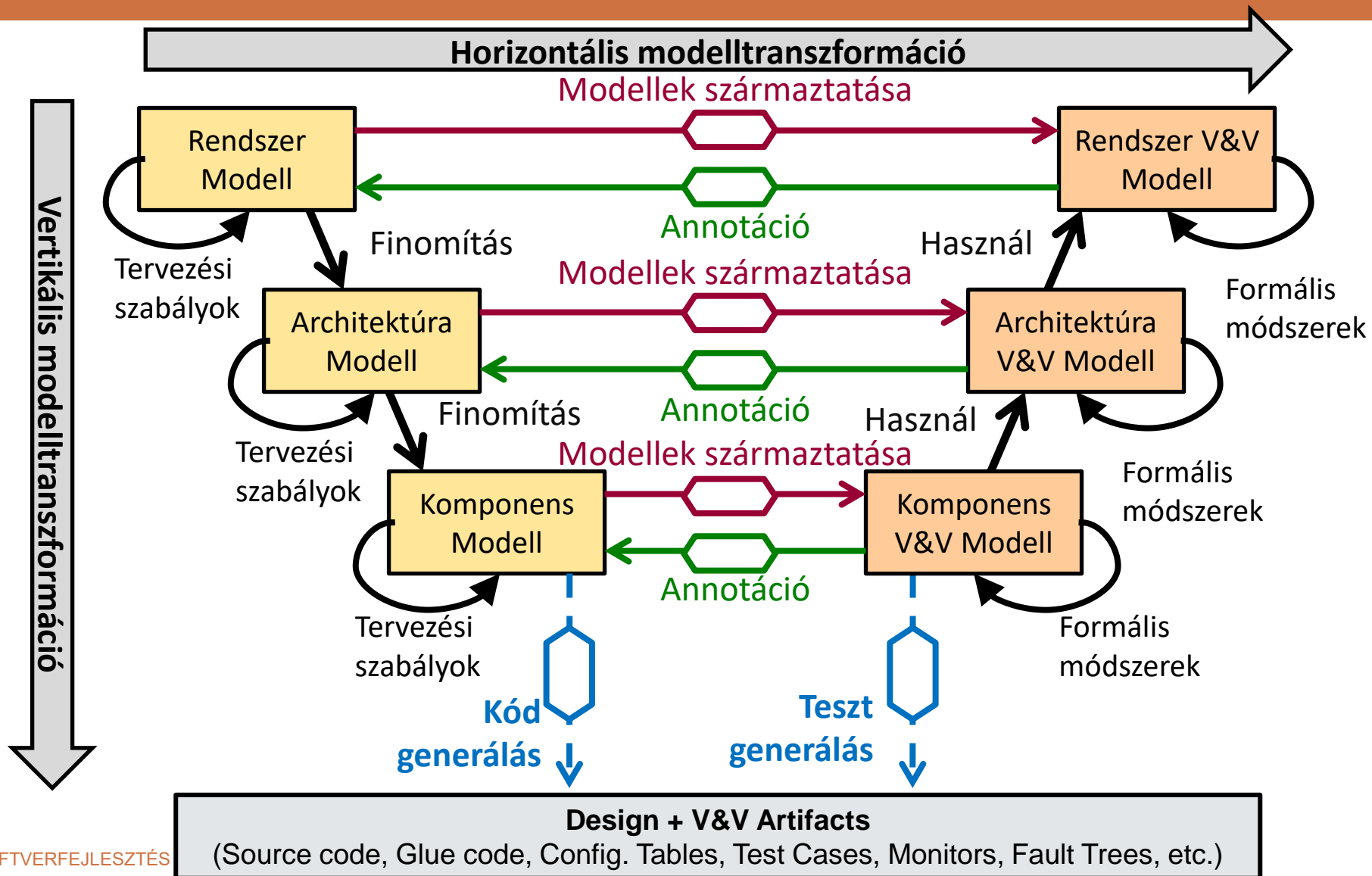
III. Funkciómodellezés

IV. Generatív programozás

V. Parciális modellezés



Modellek és Transzformációk kritikus rendszerek fejlesztésében



Development Process for Critical Systems

Egyedi Fejlesztési Folyamat (Hagyományos V-modell)



Kritikus rendszerek tervezése

- **tanúsítványozási folyamatot** igényel
- **alátámasztott bizonyítékok** kidolgozásához,
- hogy a **rendszer hibamentes**

Szoftvereszközök minősítése

- **tanúsítvány** megszerzése
- egy **szoftvereszközre**
- amelyet a **kritikus rendszerek tervezéséhez használnak**

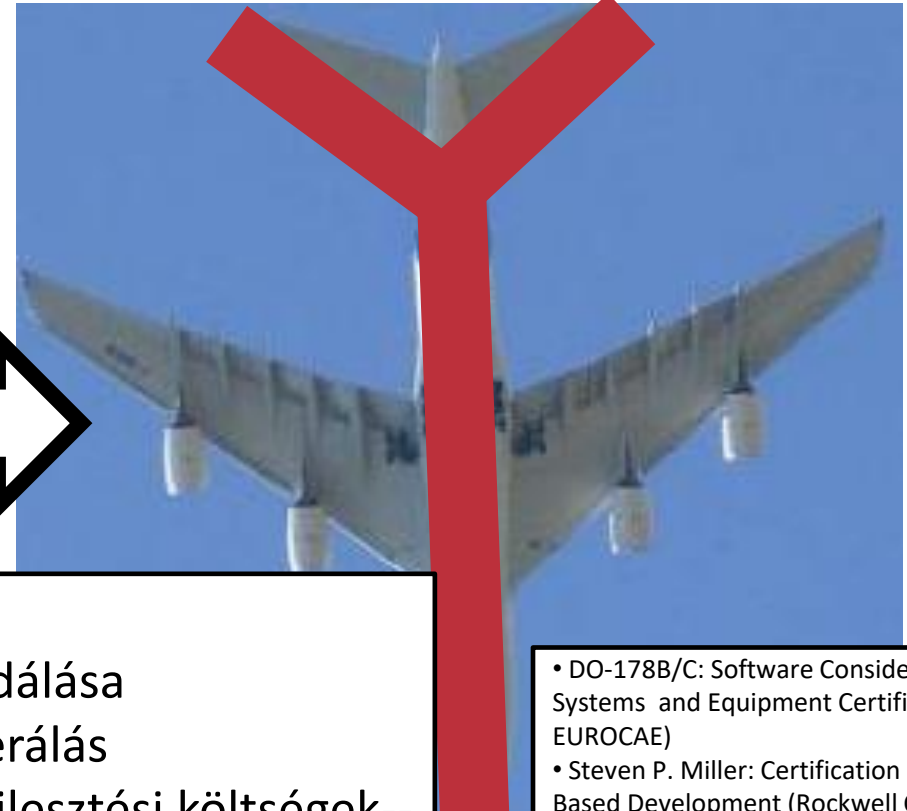
Innovatív Eszköz → Jobb Rendszer
Tanúsított Eszköz → Tanúsított Kimenet

Development Process for Critical Systems

Egyedi Fejlesztési Folyamat (Hagyományos V-modell)



Modellvezérelt Tervezés (Y-modell)



Az MDE fő gondolatai

- rendszermodellek korai validálása
- automatikus forráskód-generálás
- ➔ minőség++ eszközök ++ fejlesztési költségek--

- DO-178B/C: Software Considerations in Airborne Systems and Equipment Certification (RTCA, EUROCAE)
- Steven P. Miller: Certification Issues in Model Based Development (Rockwell Collins)

Modellalapú fejlesztések

I. Fejlesztési fogalmak

II. Kritikus rendszerek fejlesztése

III. Funkciómodellezés

IV. Generatív programozás

V. Parciális modellezés



Funkciómodellezés – bevezetés

- Termékcsalád elemei közti különbségek
 - > Mobiltelefonok
 - Kijelző típusa
 - I/O interfészek
 - > Autógyártás
 - Ajtók száma
 - Motor típusa
- Szakterületi nyelv a különbségek összefogására: funkciómodellezés

Funkciómodellezés

- Funkciómodell (feature model)

- > *implementációtól független, tömör leírása a különböző szakterületi változatoknak*
- > a konkrét termékpéldányok közti különbségek
- > termékcsalád konfigurációs lehetőségei

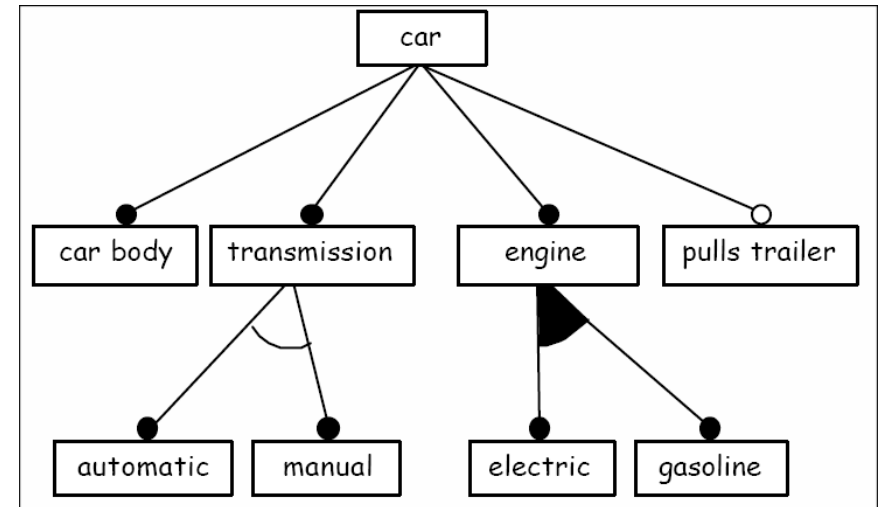
- Kulcs: újrahasznosítás

- Segít elkerülni:

- > Fontos funkció / variáció kimaradjon
- > Feleslegesen vegyünk fel funkciót / variációt

Funkciómodellezés

- Modellelemek
 - > Csomópontok
 - > Irányított élek
 - > Jelölés az éleknél
- Gyökérelem: fogalom (concept)
- Funkciók (feature node)
- Konfiguráció: funkciók részhalmaza
- A konfigurációnak be kell tartania bizonyos szabályokat!

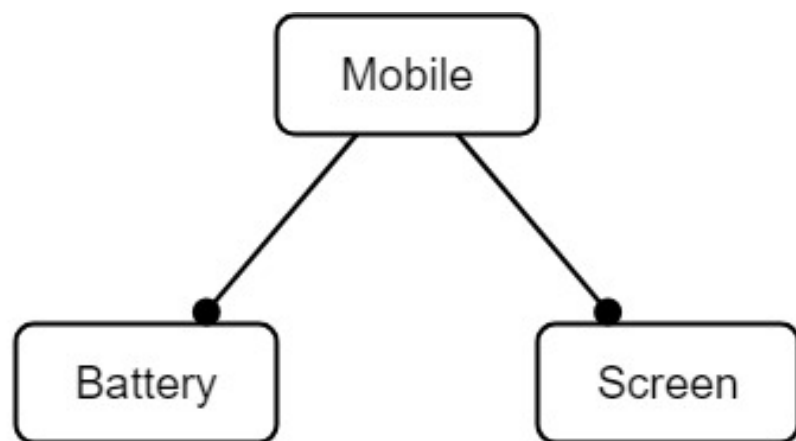


Funkciómodellezés

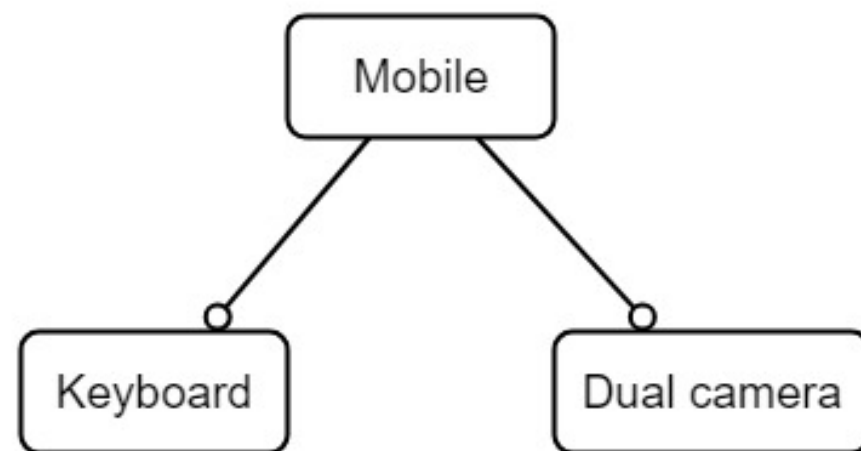
- Központban a funkció (feature)
 - > *Tudás egy része, a fogalom építőeleme*
 - > Segít megtalálni az azonosságokat, különbségeket termékek, termékcsoporthoz között
 - > Hasznos ha több változat van ugyanabból a termékből
- <https://modeling-languages.com/analysis-of-feature-models/>

Funkciómodellezés

■ Kötelező funkció

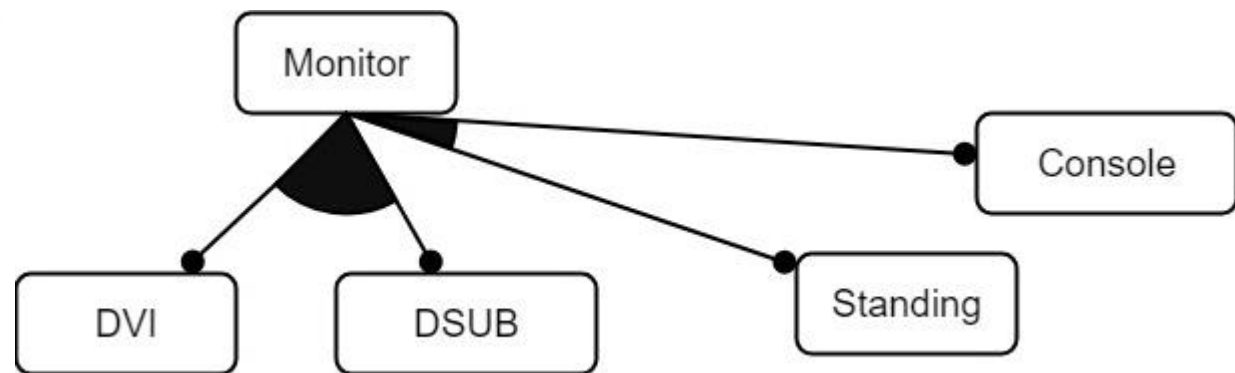
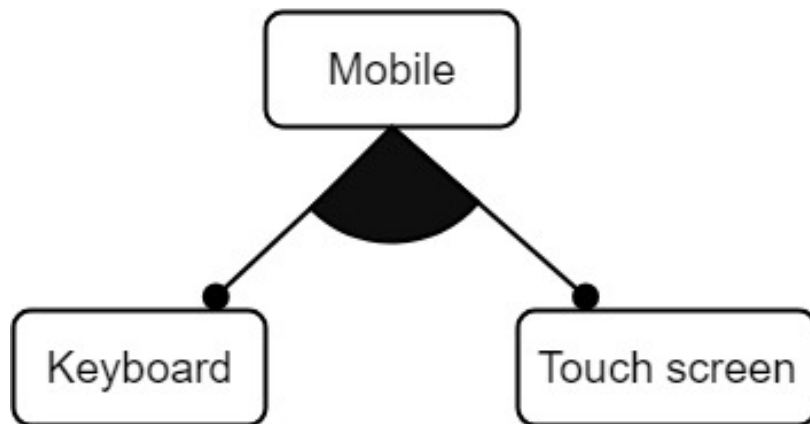


■ Opcionális funkció



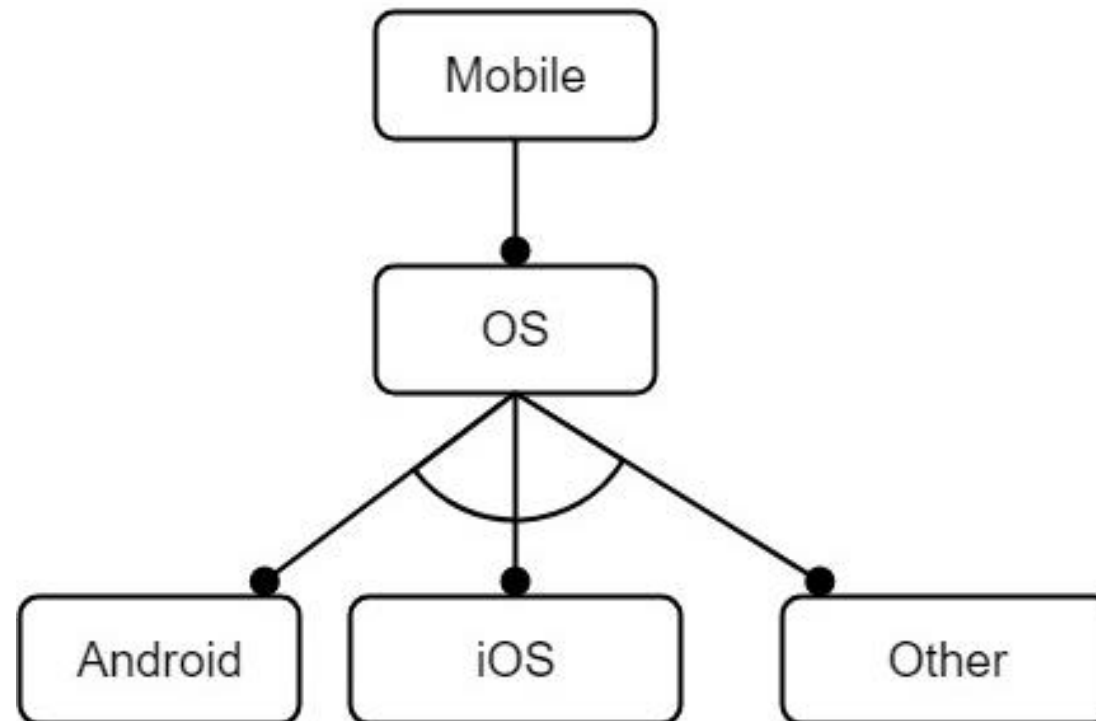
Funkciómodellezés

- „Vagy” kapcsolat (legalább 1)



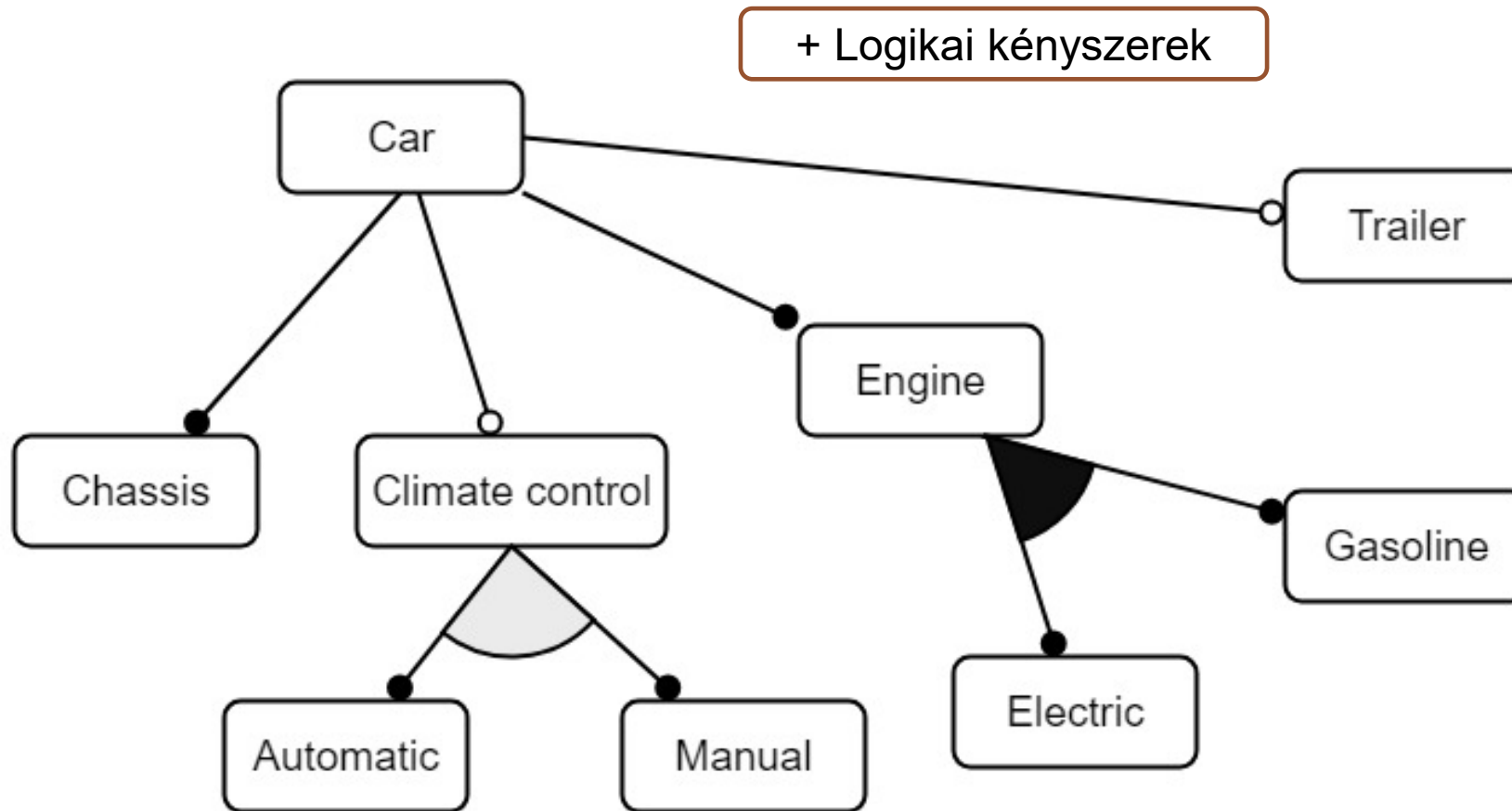
Funkciómodellezés

- Alternatív funkciók (valamelyik elem)



Funkciómodellezés

■ Példa: autó

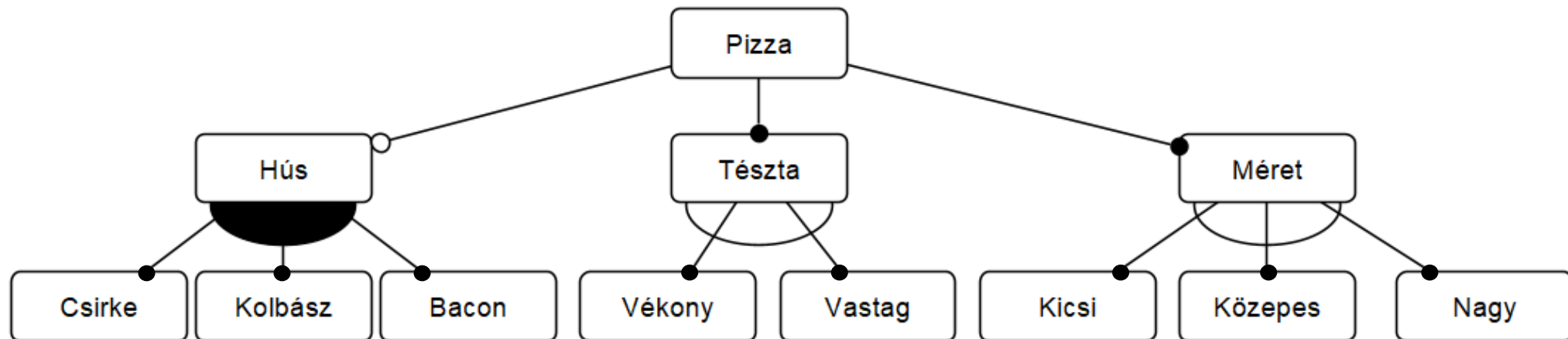


Funkciómodellezés – Példa

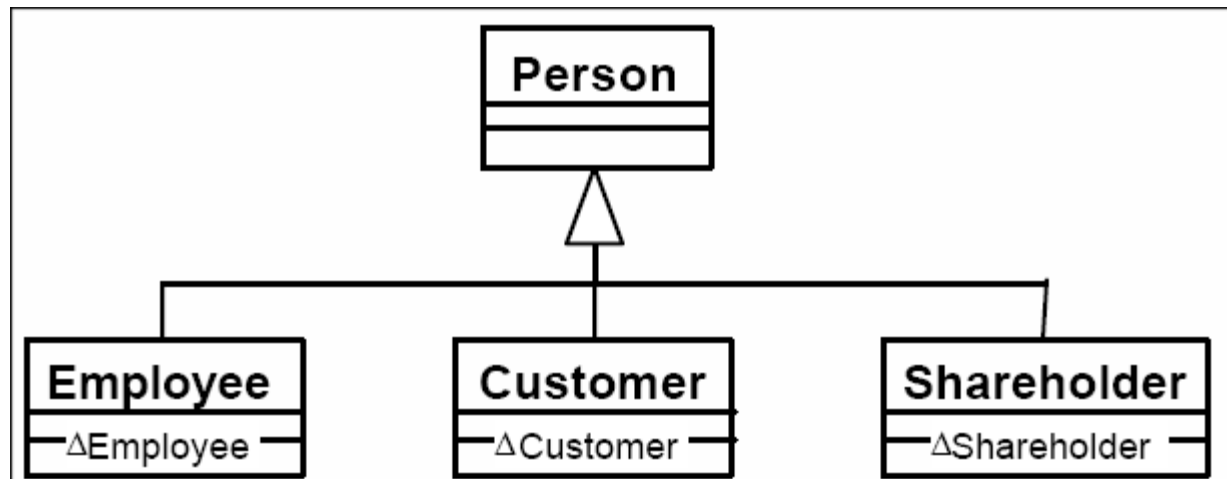
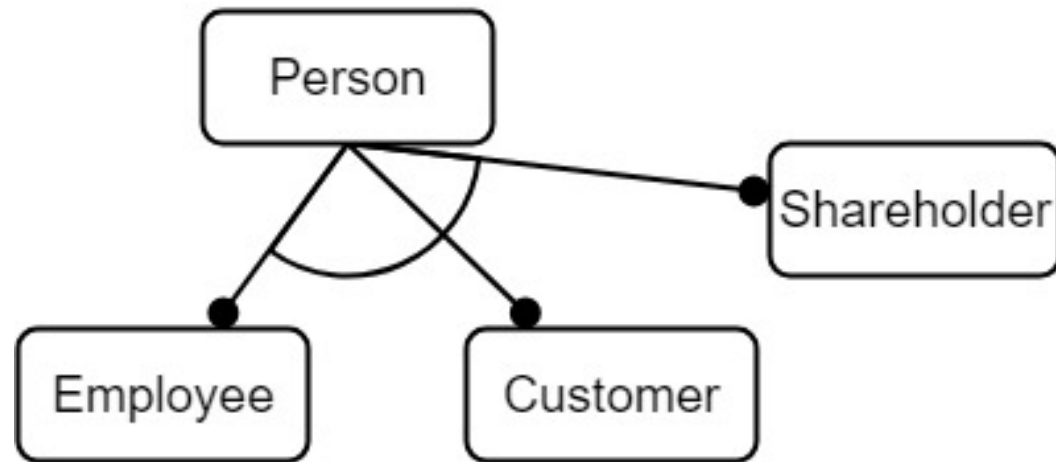
- Készítsen funkciómodellt a következő feladathoz: pizza elkészítése. A modellekben többek között legyen lehetőség megadni húsokat (csirke, kolbász, bacon), további feltéteket (paradicsom, hagyma, paprika), tészta típusokat (hagyományos, light), méretet (kicsi, közepes, nagy). A modell tartalmazzon opcionális-, kötelező és kizáró (OR) funkciót. Szövegesen indokolja röviden a modell felépítését!

Funkciómodellezés – Példa – Megoldás

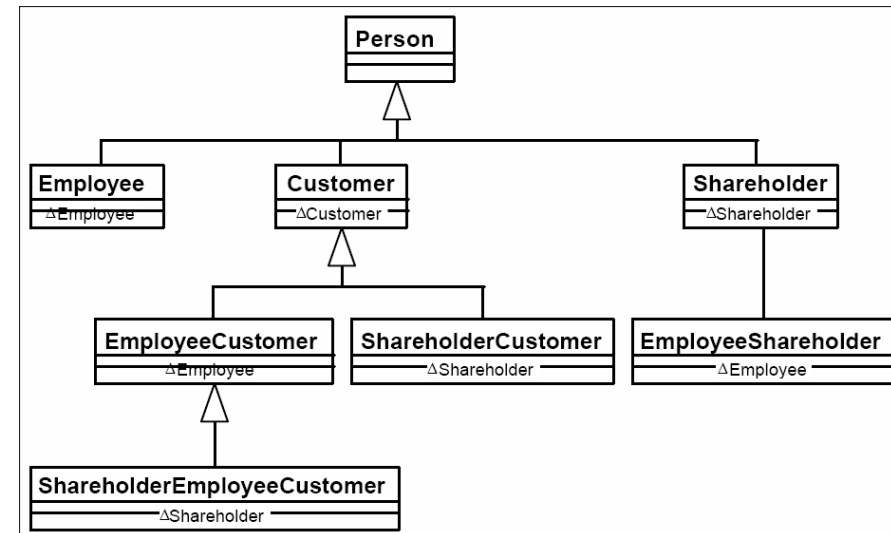
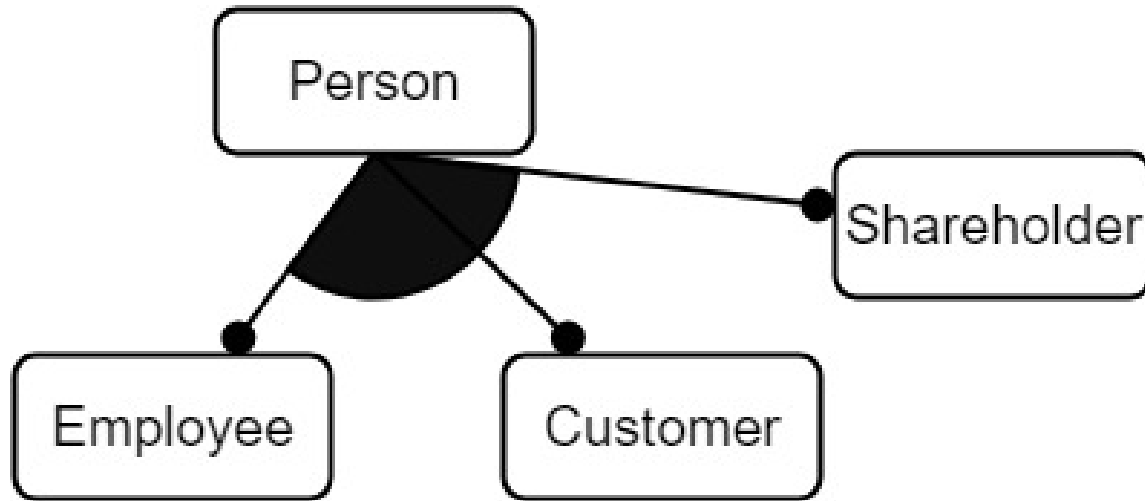
- Készítsen funkciómodellt a következő feladathoz: pizza elkészítése. A modellekben többek között legyen lehetőség megadni húsokat (csirke, kolbász, bacon), további feltéteket (paradicsom, hagyma, paprika), tészta típusokat (hagyományos, light), méretet (kicsi, közepes, nagy). A modell tartalmazzon opcionális-, kötelező és kizáró (OR) funkciót. Szövegesen indokolja röviden a modell felépítését!



Funkciómodellezés – kódgenerálás



Funkciómodellezés – kódgenerálás



Funkciómodellezés a gyakorlatban

- Cél alkalmazás
 - > A funkciómodellnek megfelelő termékeket tartalmazó webes katalógus
 - > Funkciómodell szerint megvalósított keresés a katalógusban
- Generálás a modell alapján
 - > Web alkalmazás
 - > Adatbázis tábladefiníciók

Funkciómodellezés a gyakorlatban

The screenshot displays a software interface for a bicycle product line. On the left, the 'FEATURE MODEL' is shown with a hierarchical tree structure. The root is 'Trek Bikes', which branches into 'Category' and 'Rider Category'. 'Category' further branches into 'bike-path', 'cruiser', 'kids', 'mountain-full-suspension', 'mountain-hardtail', 'road', 'track', 'triathlon', and 'urban'. 'Rider Category' branches into 'Price - US\$' and 'Frameset'. 'Price - US\$' branches into '1,000 or under', '1,001-2,000', '2,001-3,000', '3,001-5,000', and '5,001-10,000'. 'Frameset' branches into 'Wheels', 'Drivetrain', 'Components', and 'Saddle'. 'Saddle' branches into 'Seat Post' and 'Alloy'. 'Seat Post' branches into 'Bontrager-Alloy-Infinite-Angle' and 'Bontrager-Approved'. On the right, the 'PRODUCT CATALOG (9 products)' is shown as a list of seven bicycles with their names, prices, and images. Each product has a 'View this model' link.

Product Name	Price (US\$)	Image
1. KDR 7.2 FX-34 cm	\$439.99	
2. Wasabi 24"-24"	\$329.99	
3. MT 240-24"	\$549.99	
4. Drift 20"-20"	\$269.99	
5. MT 60-20"	\$309.99	
6. Mystic 20-20"	\$219.99	
7. Float-16"	\$219.99	

- Funkciók lemodellezése
- Konfiguráció kiválasztása
- Kódgenerálás
- Feladat megoldása
- #Konfiguráció > #Termék

Marcilio Mendonca, Andrzej Wąsowski, and Krzysztof Czarnecki. 2009. **SAT-based analysis of feature models is easy.** In *Proceedings of the 13th International Software Product Line Conference (SPLC '09)*. Carnegie Mellon University, USA, 231–240.

Modellalapú fejlesztések

I. Fejlesztési fogalmak

II. Kritikus rendszerek fejlesztése

III. Funkciómodellezés

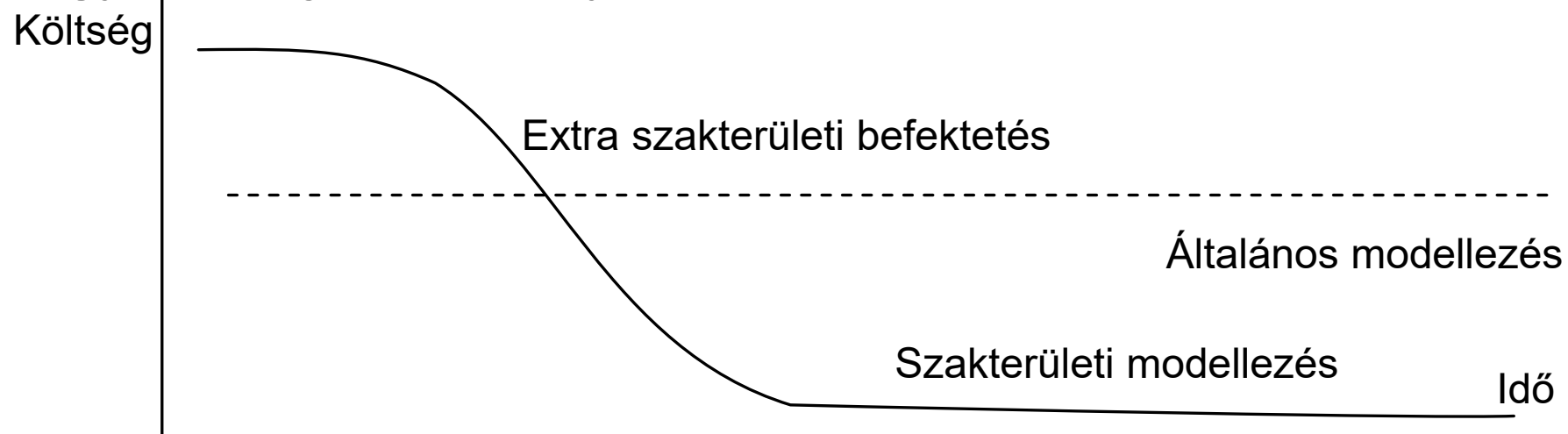
IV. Generatív programozás

V. Parciális modellezés



Generatív programozás

- Programozási módszertan, alapja az automatikus forráskód-generálás
- Párhuzamba vonható a komponens-alapú szoftverfejlesztéssel és a termékcsalád tervezéssel
- Újrahasznosítható termék
- Nem egyszeri fejlesztés, folyamatos evolúció



Generatív programozás

- Generatív paradigma
 - > Működés: modellezőnyelv+generátorok
 - > Többszöri használatnál éri meg
- Kódgenerálás
 - > Nincsenek univerzális DSL fordítók (mint C fordítók)
 - > Gyakran a DSL-t és a generátort ugyanott fejlesztik
 - Gyors fejlesztés, finomhangolási lehetőség
 - Hibalehetőségek

Alkalmazás generálása

- Tipikus modellfeldolgozás: alkalmazás generálása
 - > Amire szükségünk van
 - > Szakterületi nyelv modellje
 - > Generátor
 - > Keretrendszer (pl. osztálykönyvtár)
- Kihívások:
 - > Túl részletes/általános nyelv → kicsi absztrakciós szint ugrás → kis előny a generátorból
 - > szakterület nem illeszkedik → komplex generátor
 - validálás + hozzáadott infó miatt
 - Jel: fejlesztők úgy építik a modellt, hogy a generátor elfogadja

Modellalapú fejlesztések

I. Fejlesztési fogalmak

II. Kritikus rendszerek fejlesztése

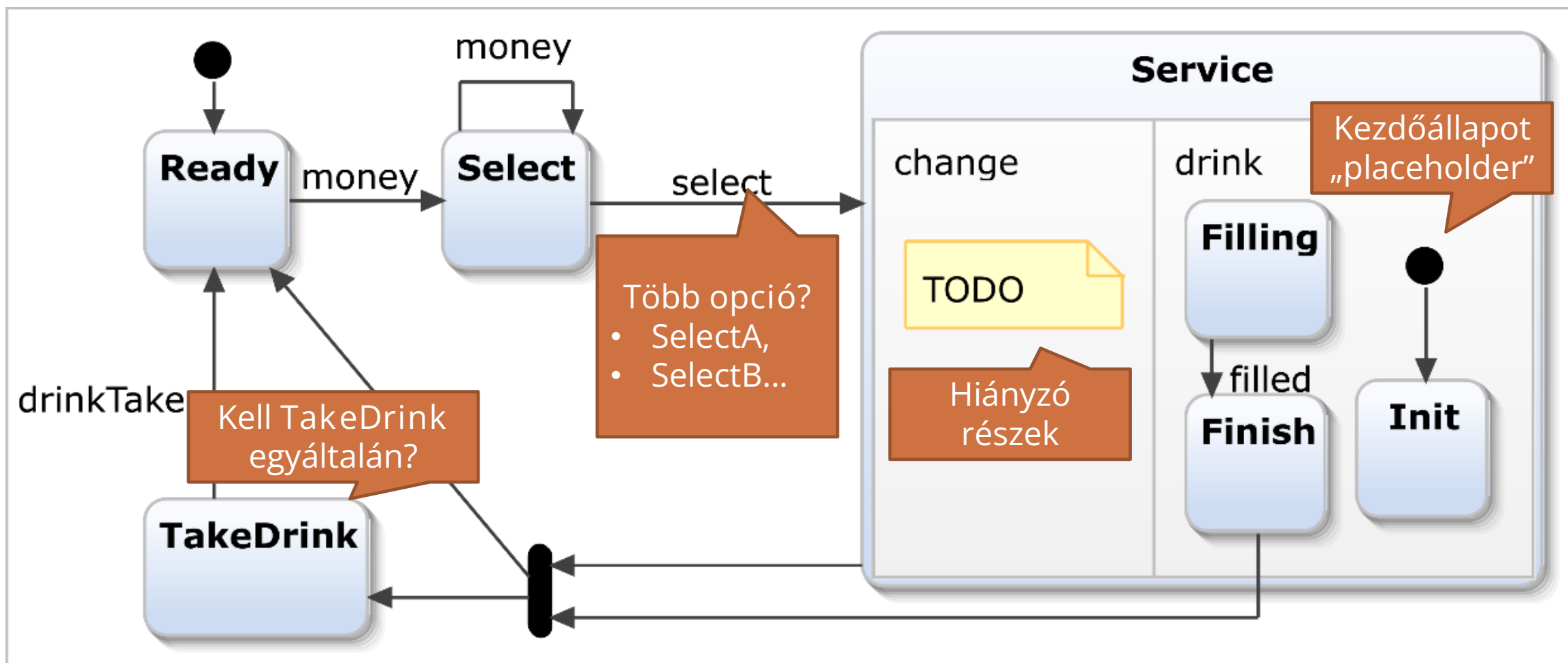
III. Funkciómodellezés

IV. Generatív programozás

V. Parciális modellezés



Példa: befejezetlen modellek



Motiváció

- A fejlesztés korai fázisa → a modellek nagy részében bizonytalanok vagyunk
- Viszont a fejlesztőkörnyezet rákényszerít, hogy kész modellekkel dolgozzunk

Hiányzó elem \Leftrightarrow Nem meghozott döntés, ismeretlen érték

Modellek fejlesztése \Leftrightarrow Modellek átírása

Kihívások:

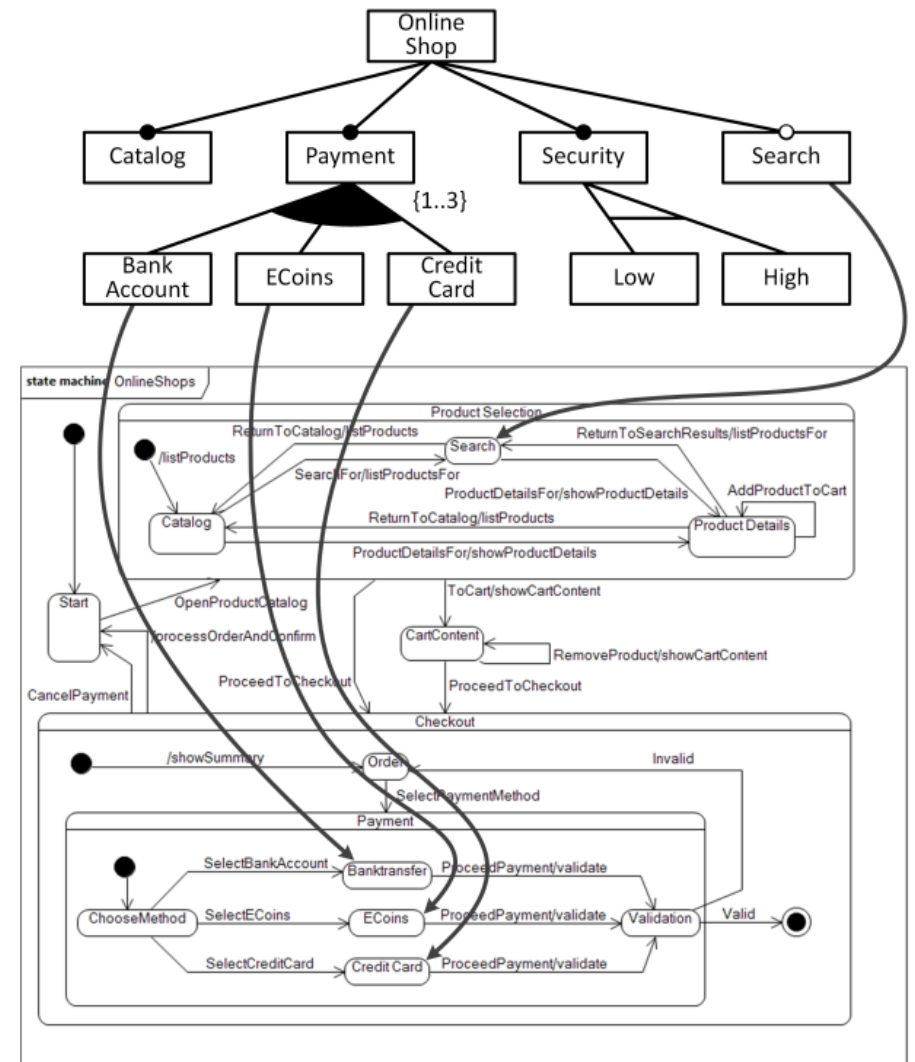
- A fejlesztőnek olyan döntéseket kell meghoznia, amire nem készült fel
- Nem tudjuk felsorolni a lehetőségeket
- Nincs megkülönböztetve a hibás és a félkész modell

Kihívás: Hiányzó elemek szemantikája

Funkciók leképezése modellelemekre

- Mit tegyünk, ha nem kód, hanem modell kell?
- Funkciók leképezése modellelemekre
- 1 kombináció = 1 modell
- Σ konfiguráció = 150% modell
- A 150%-os modell nem feltétlenül szabályos modell

Stephan Weißleder, Hartmut Lackner: Top-Down and Bottom-Up Approach for Model-Based Testing of Product Lines

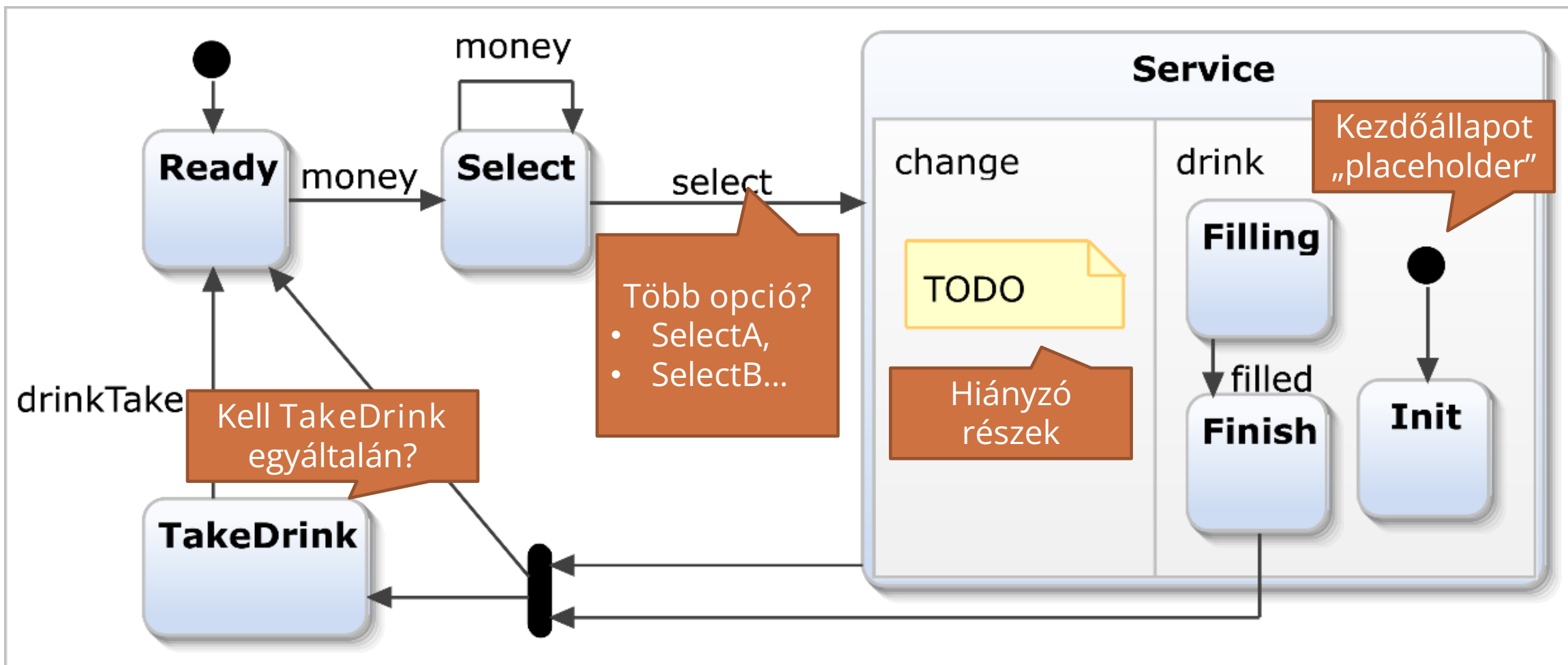


MAVO szemantika

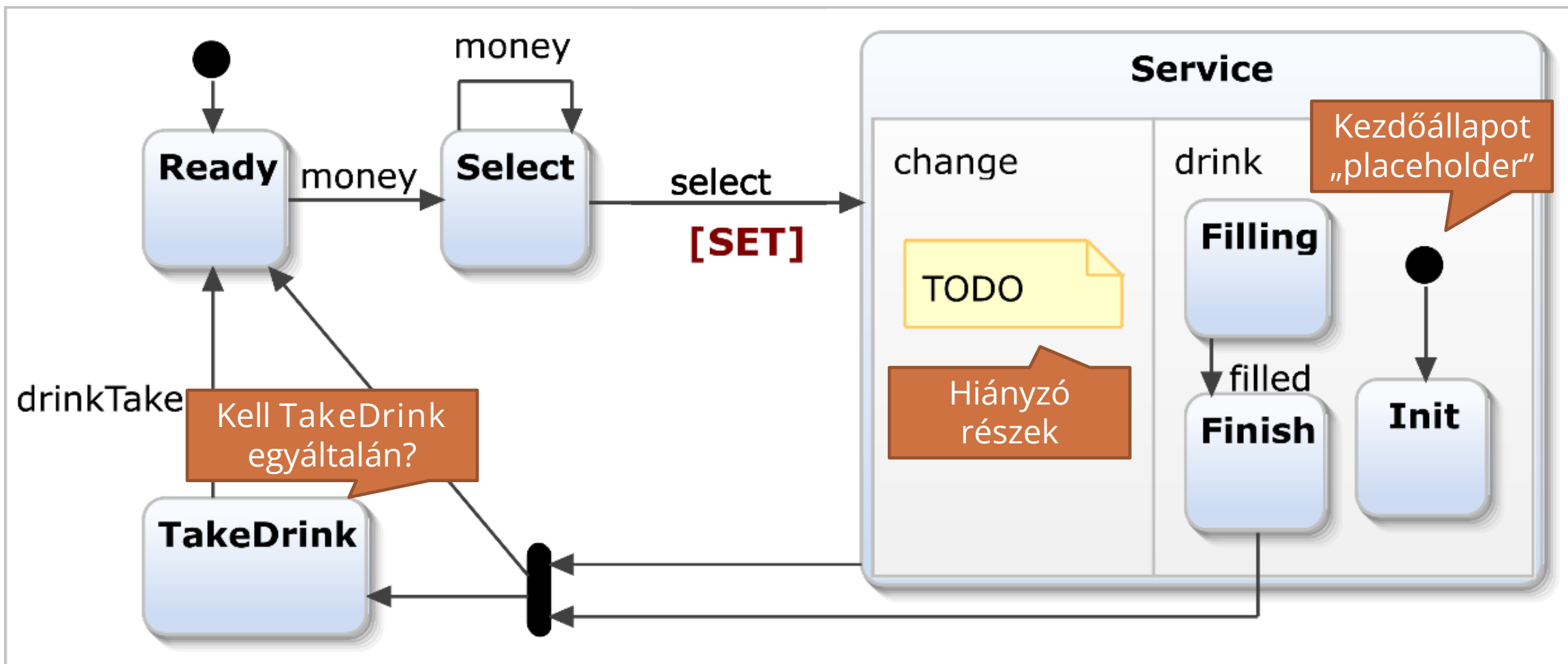
- **MAVO:** praktikus módja a modell bizonytalansággal való annotálásának
 - > **May:** elemek elhagyhatók
 - > **Abstract (Set):** elemek halmazait reprezentálja
 - > **Var:** összevonható elemek
 - > **Open:** új elemek adhatók hozzá
- **Automatizálás:** alternatívák generálása, összes alternatíva ellenőrzése

Michalis Famelis, Rick Salay, and Marsha Chechik. Partial models: towards modeling and reasoning with uncertainty. In: Proceedings of the 34th International Conference on Software Engineering, pp. 573–583. IEEE Press, 2012.

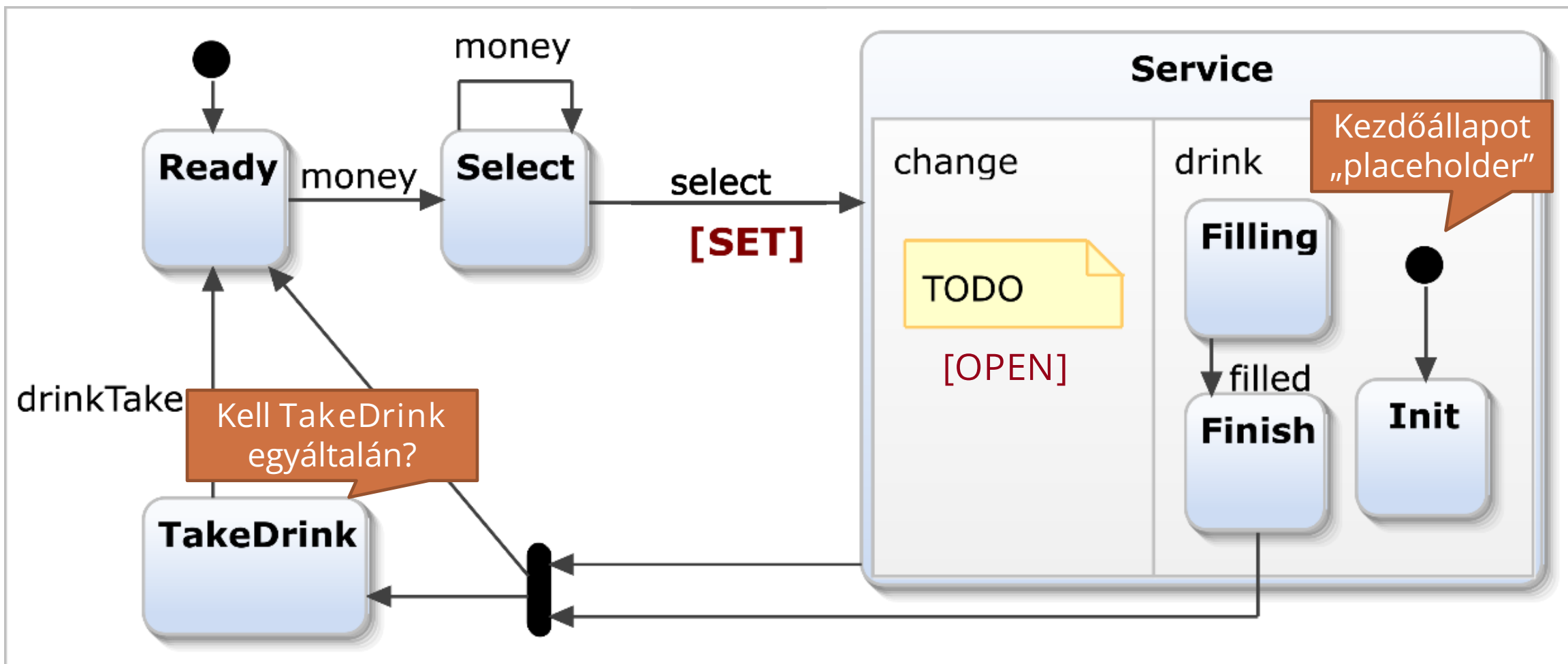
Példa: MAVO-val készült befejezetlen modellek



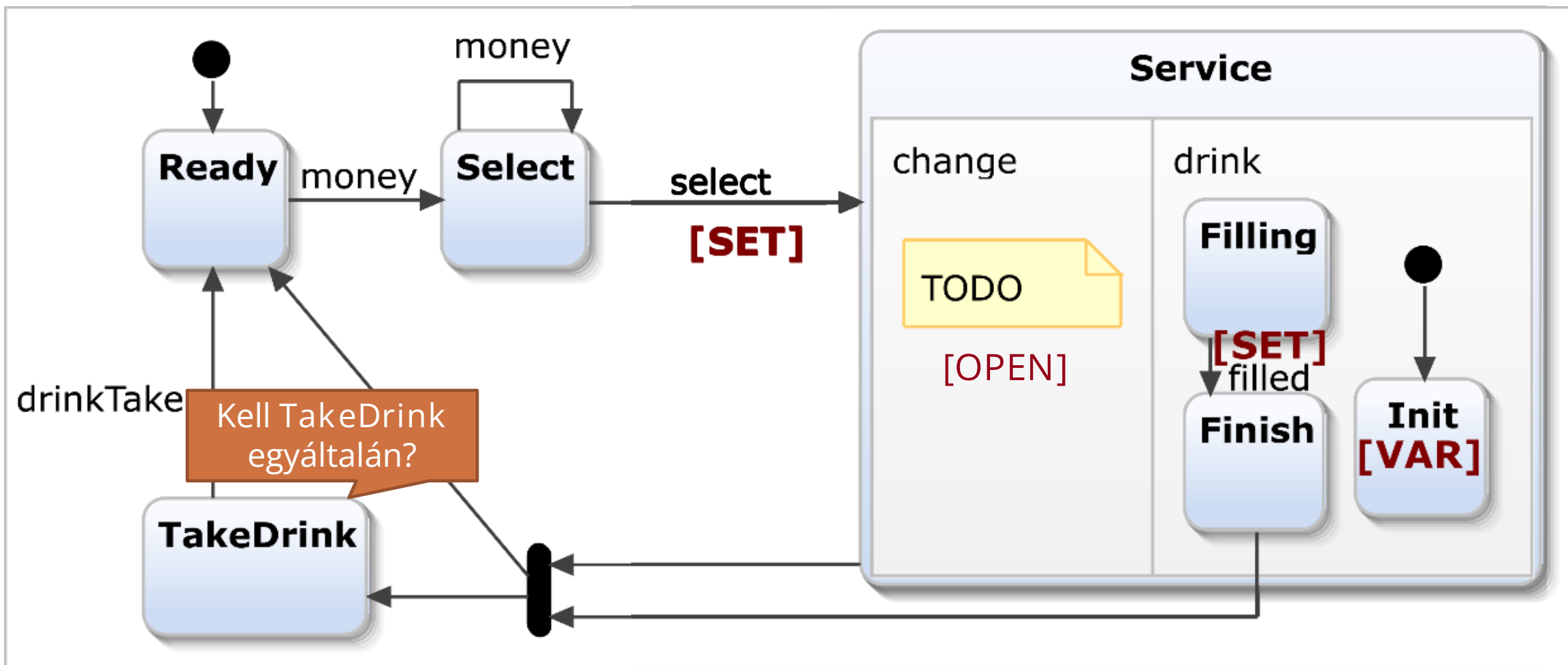
Példa: MAVO-val készült befejezetlen modellek



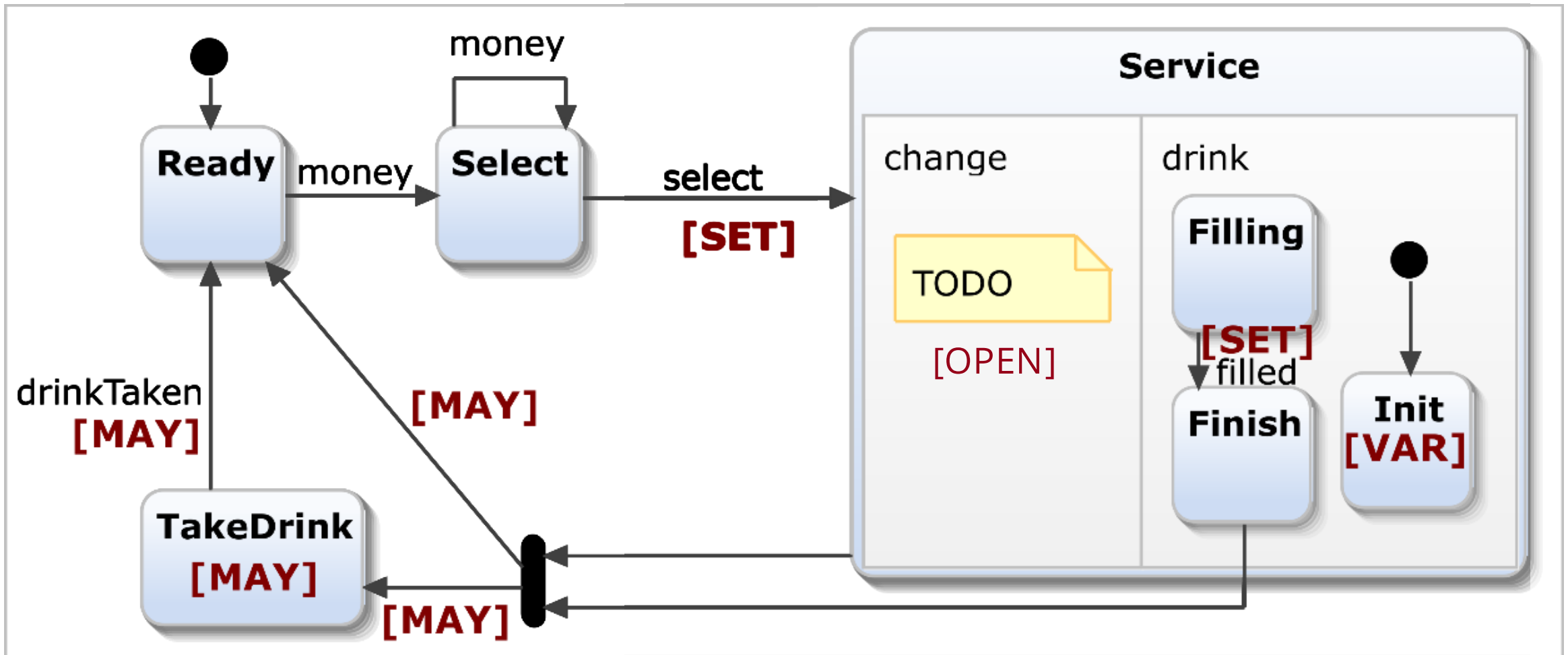
Példa: MAVO-val készült befejezetlen modellek



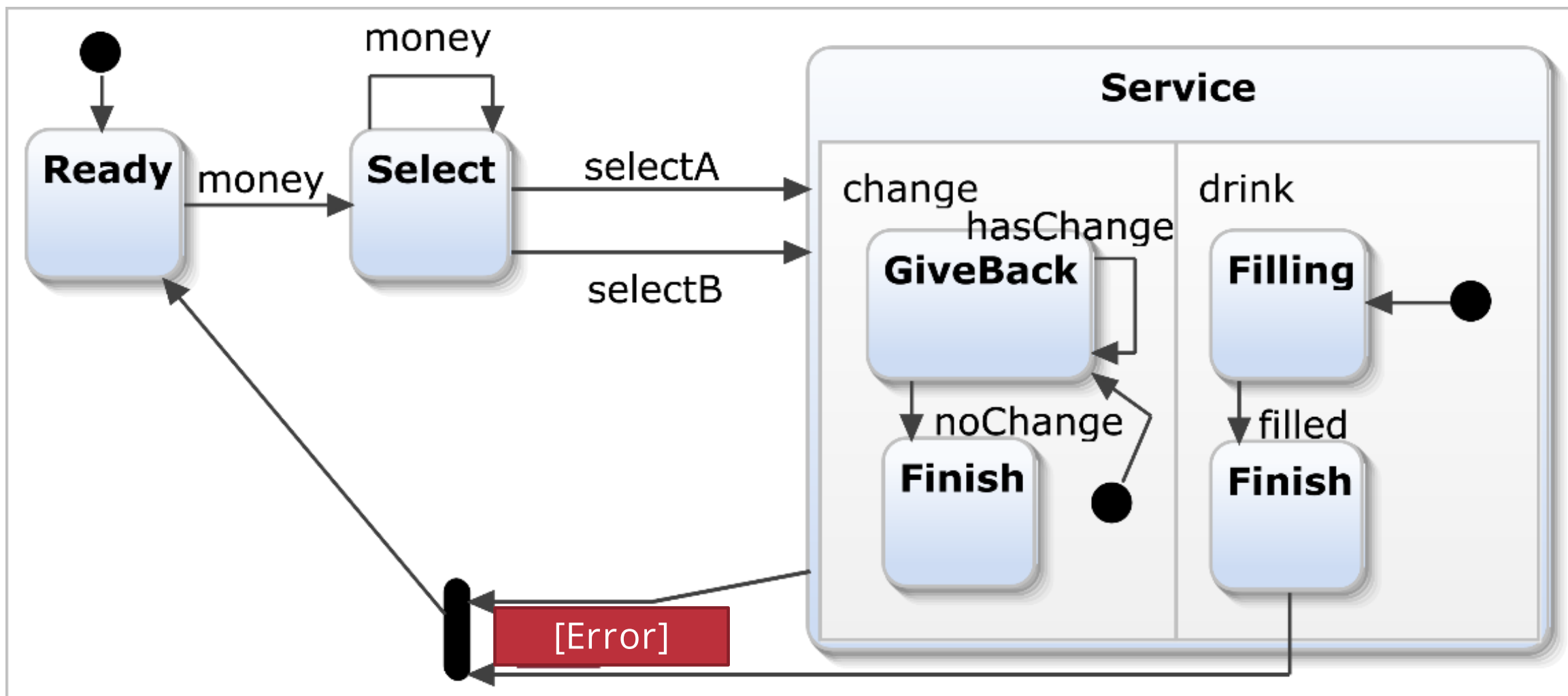
Példa: MAVO-val készült befejezetlen modellek



Példa: MAVO-val készült befejezetlen modellek



Példa: Példa konkretizálás



MAVO Modellezés Összefoglalása

- A részleges modellezés a modellek bizonytalanságát ragadja meg
- 1 részleges modell = teljes modellek halmaza
- MAVO: keretrendszer a bizonytalanság annotációjához + eszközkészlet
- A hiányzó és a befejezetlen elemek szemantikája

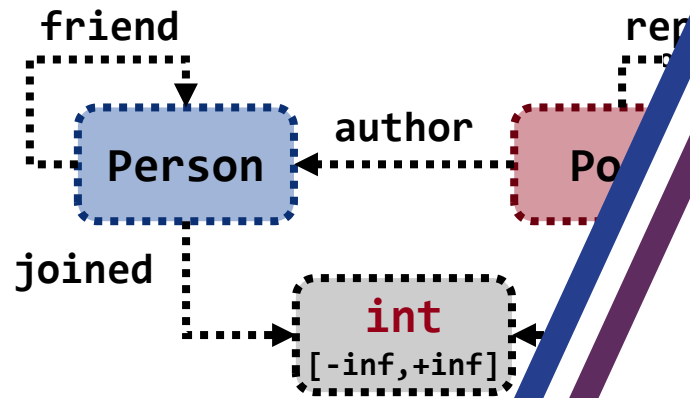
Partial models: Towards modeling and reasoning with uncertainty

M Famelis, R Salay, M Chechik

2012 34th International Conference on Software Engineering (ICSE), 573-583

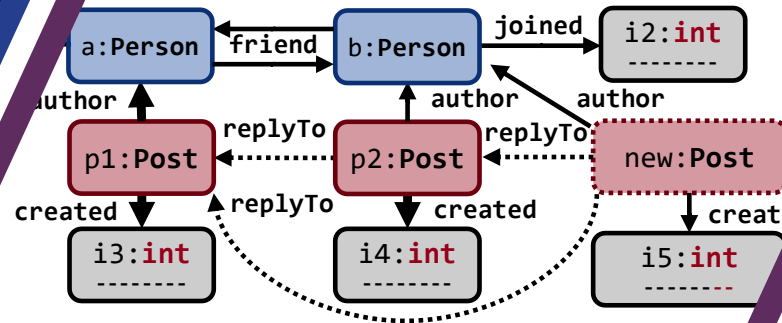
Parciális Modellek

Absztrakt modellek (Metamodell + Kényszerek)

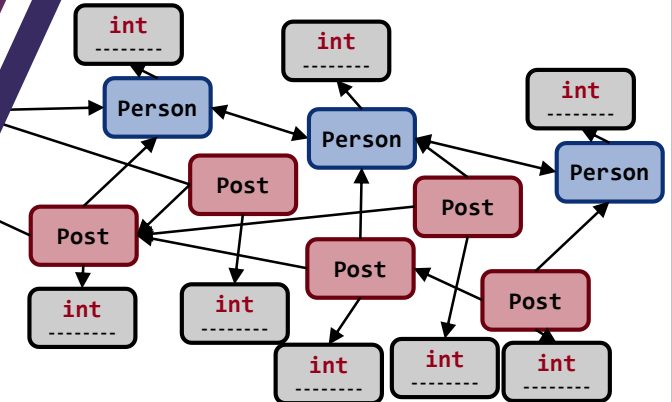


Köztes állapot

Parciális modellek:
explicit módon reprezentálja a bizonytalanságot a modellben

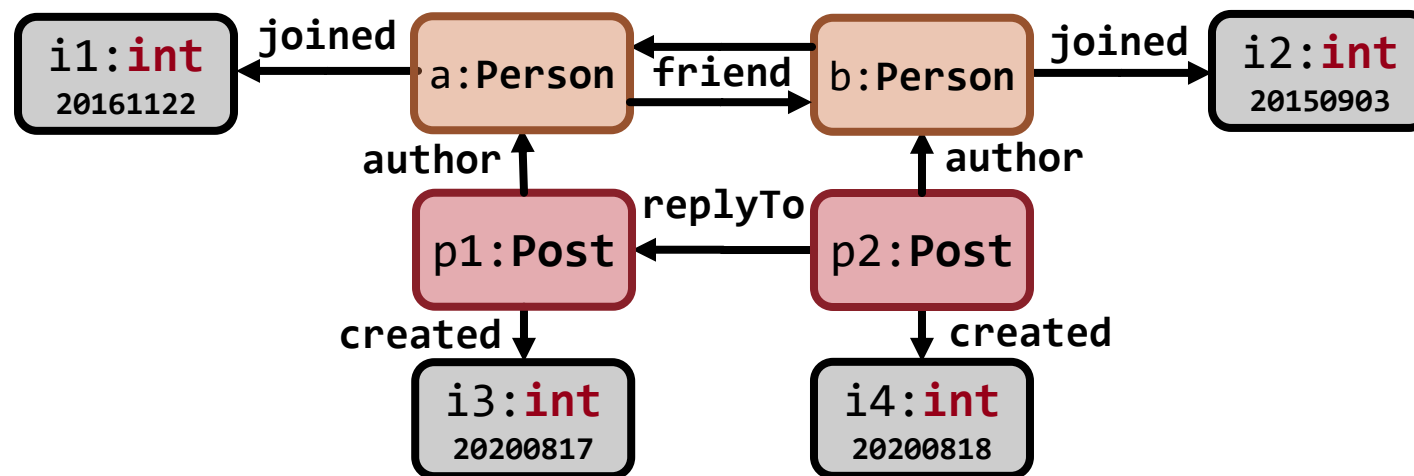


Konkrét modellek (Címkézett gráfok)

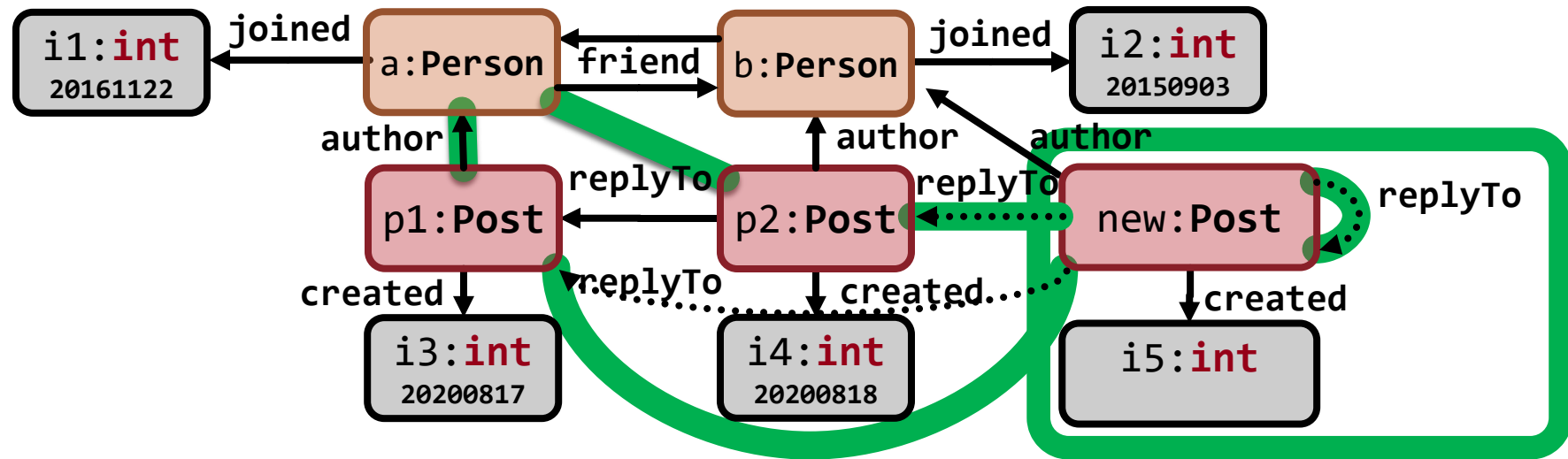


Modellgenerálás: feltárási folyamat, amely fokozatosan csökkenti a bizonytalanságot.

Parciális Modellezés: 4 értékű logika

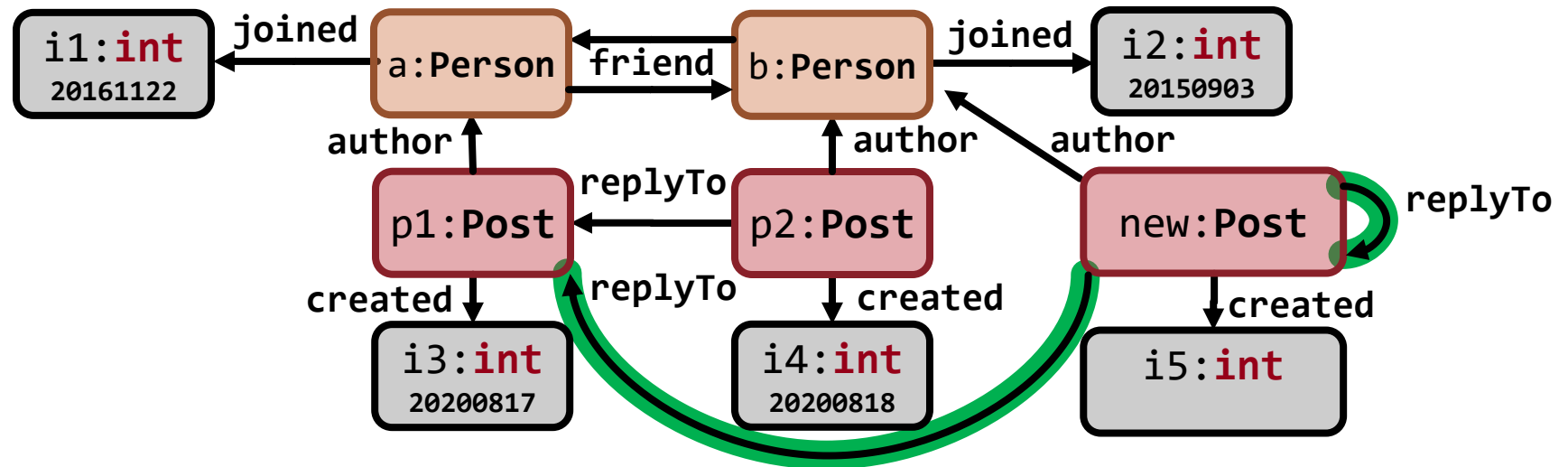


Parciális Modellezés: 4 értékű logika



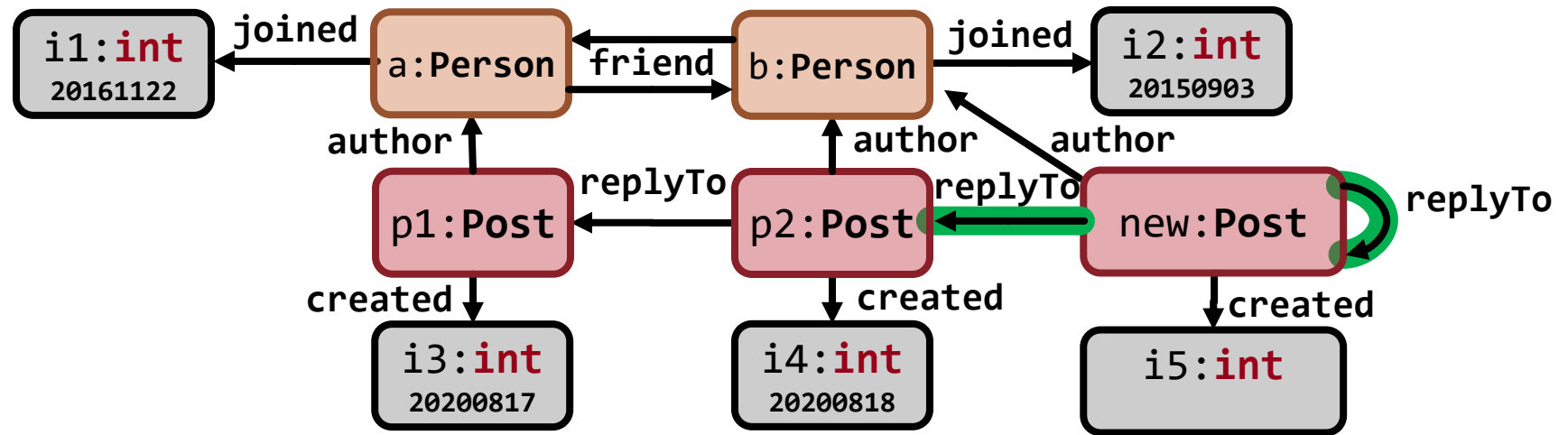
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: `true` | `false` | `unknown` | `error`

Parciális Modellezés: 4 értékű logika



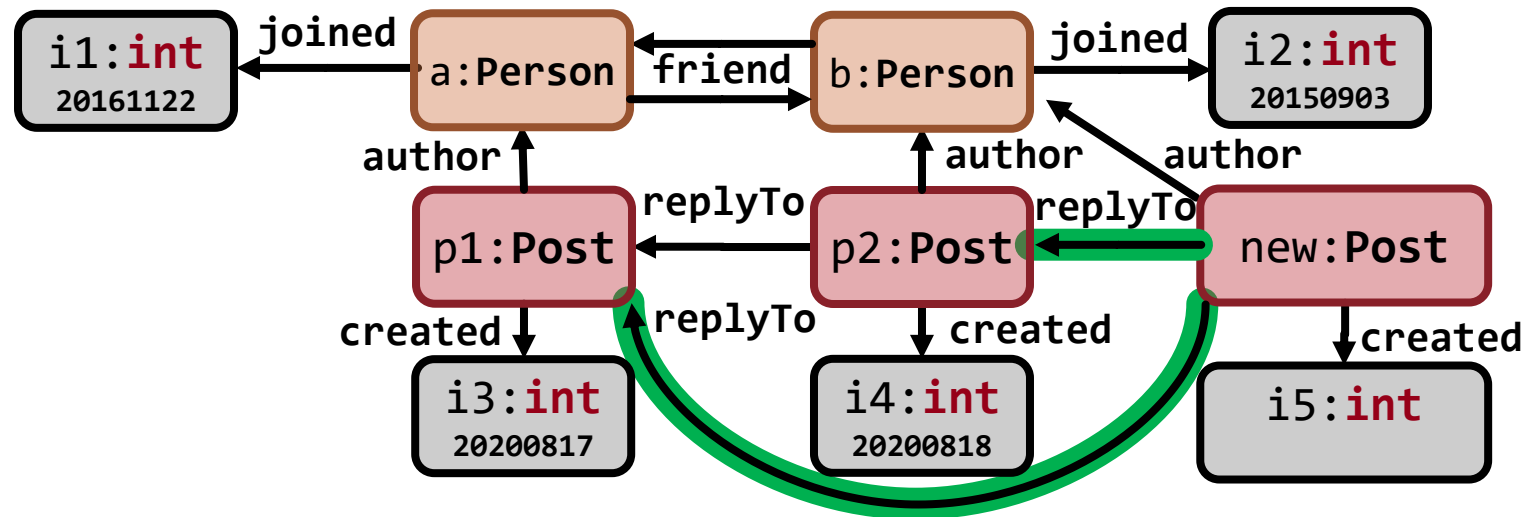
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | **unknown** | error

Parciális Modellezés: 4 értékű logika



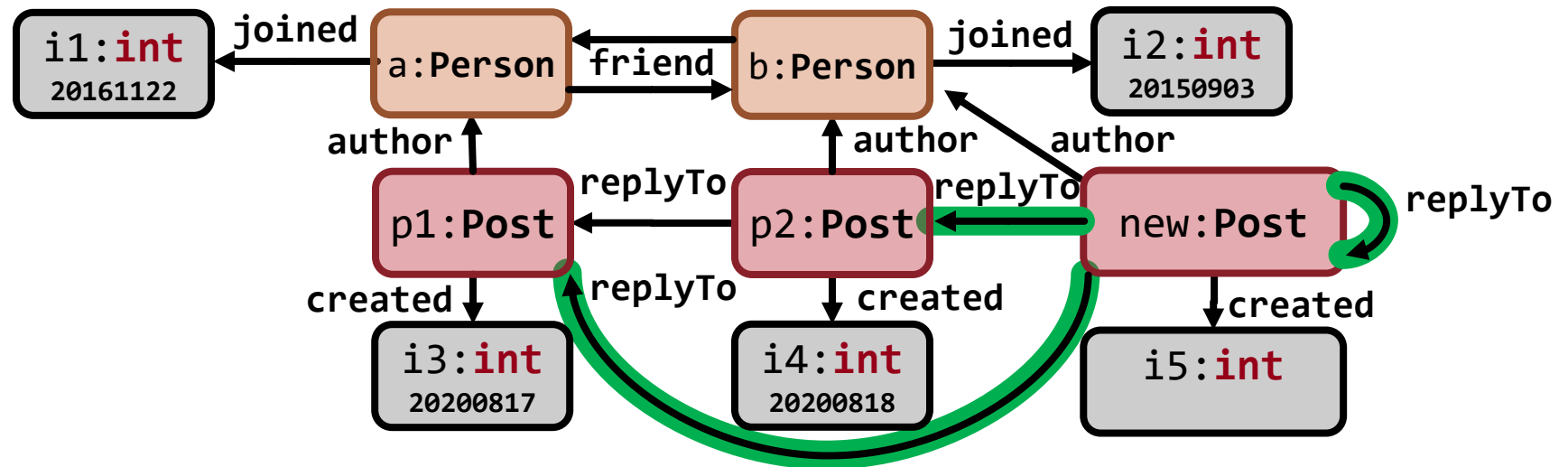
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | **unknown** | error

Parciális Modellezés: 4 értékű logika



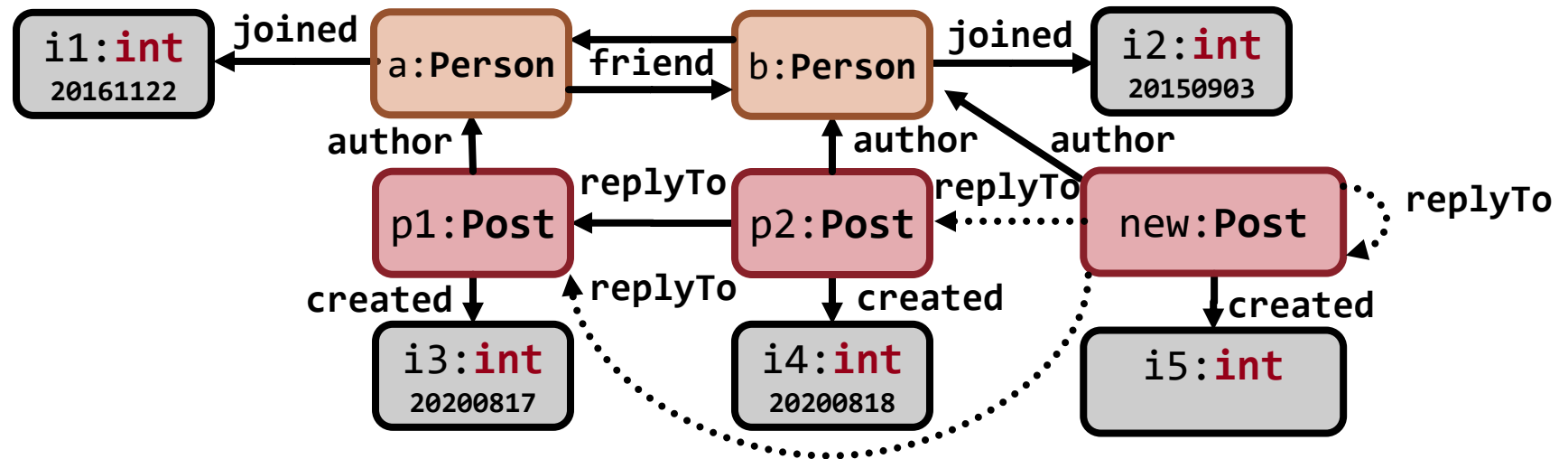
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | **unknown** | error

Parciális Modellezés: 4 értékű logika



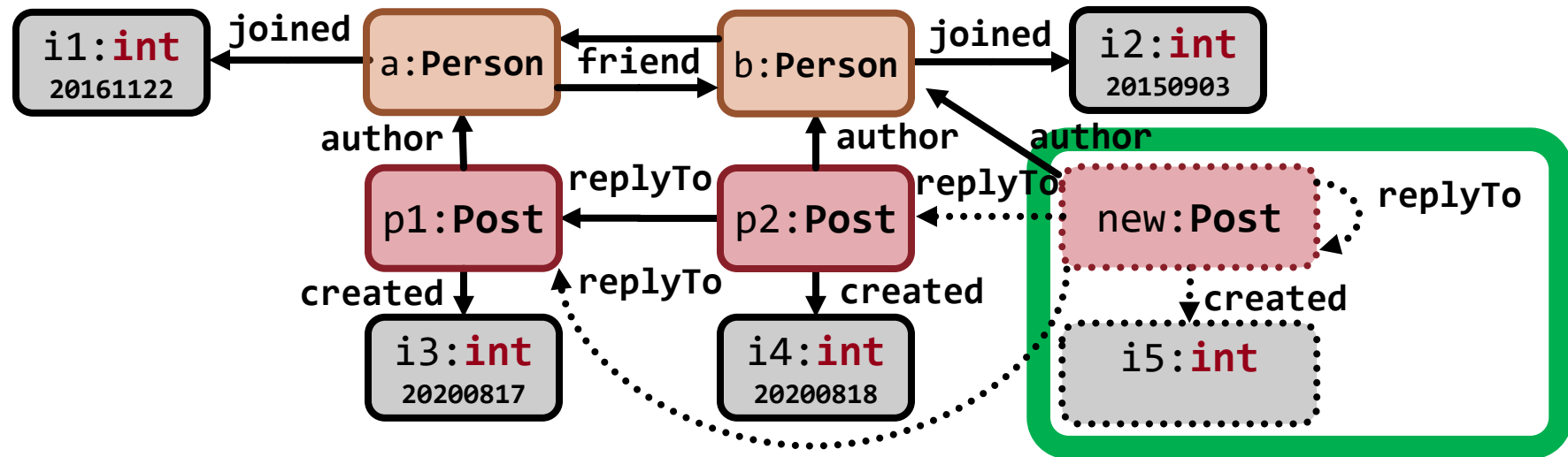
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | **unknown** | error

Parciális Modellezés: 4 értékű logika



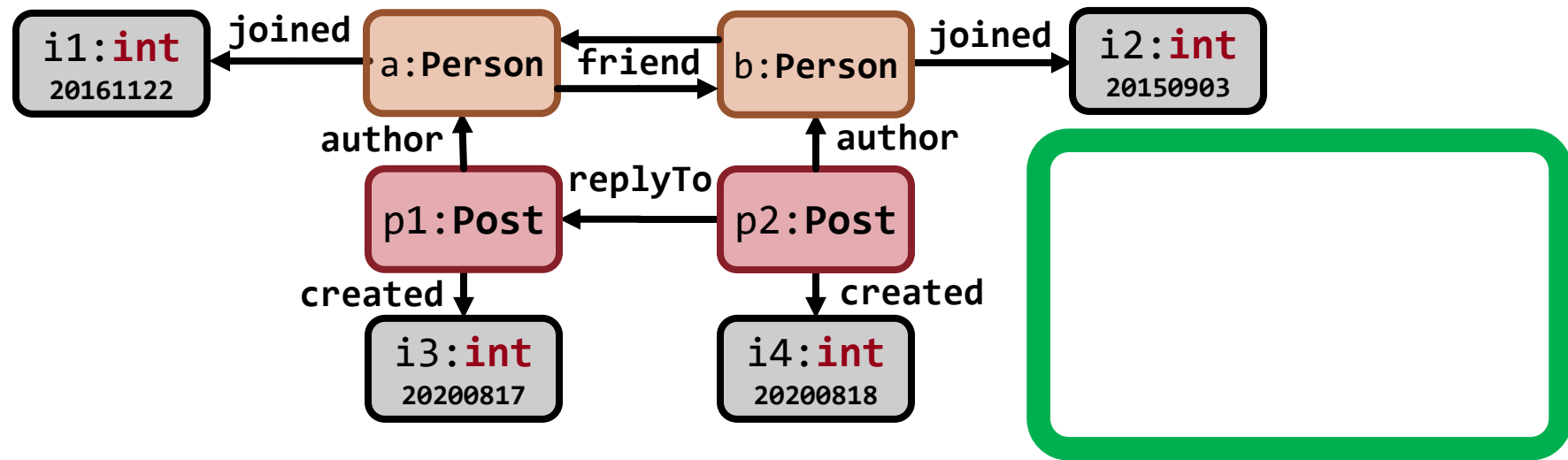
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | unknown | **error**

Parciális modellezés: létezés



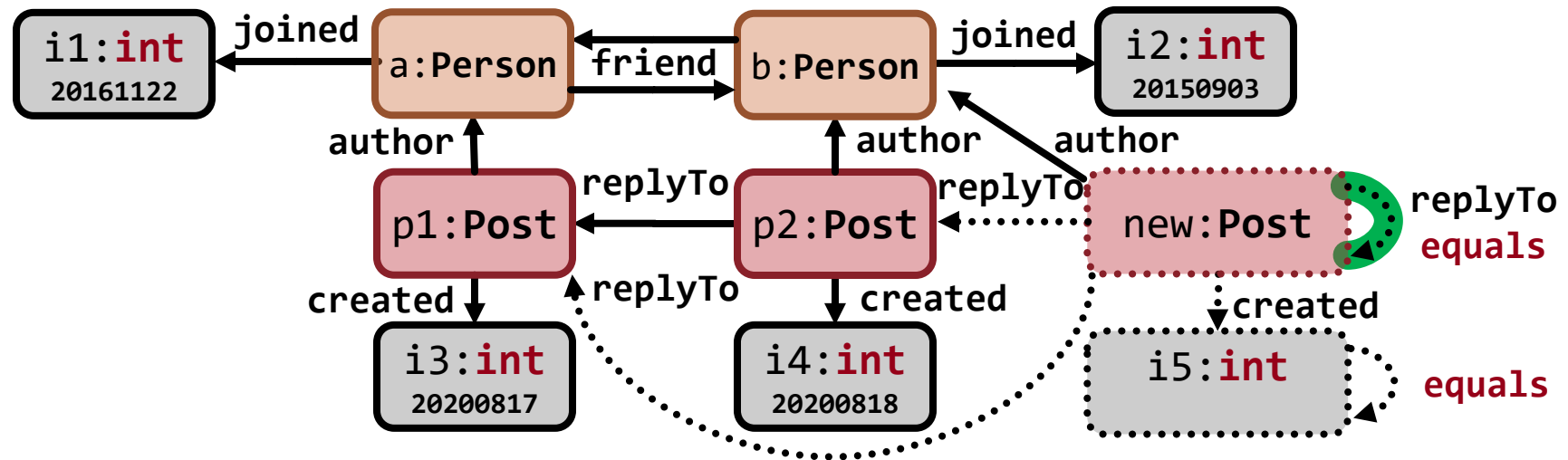
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | unknown | error
 - > 4 értékű **létezés**: **hozzáadás** vagy eltávolítás

Parciális modellezés: létezés



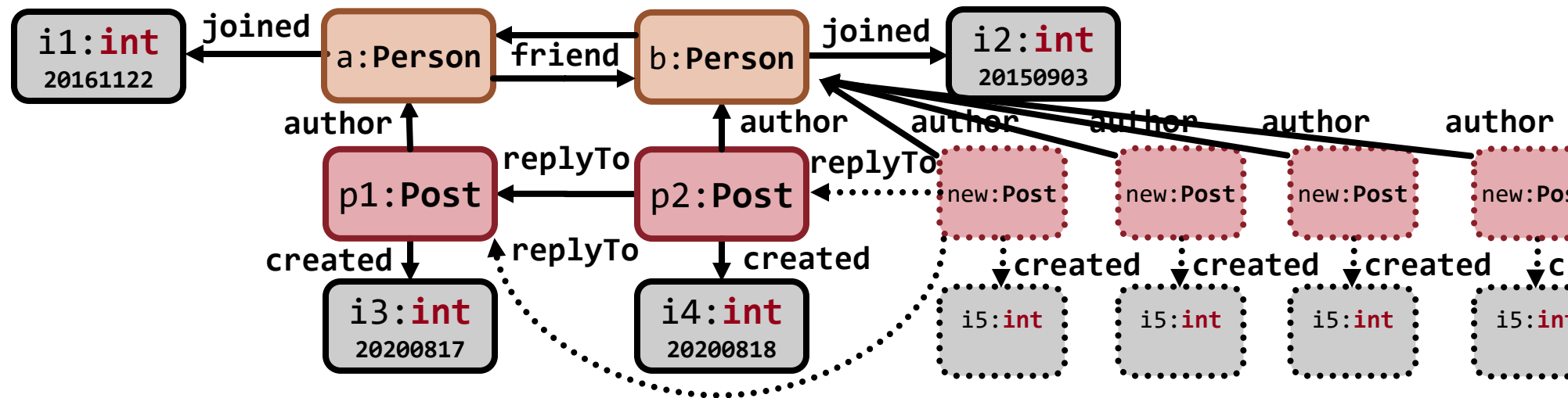
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: `true` | `false` | `unknown` | `error`
 - > 4 értékű **létezés**: hozzáadás vagy eltávolítás

Parciális modellezés: egyenlőség



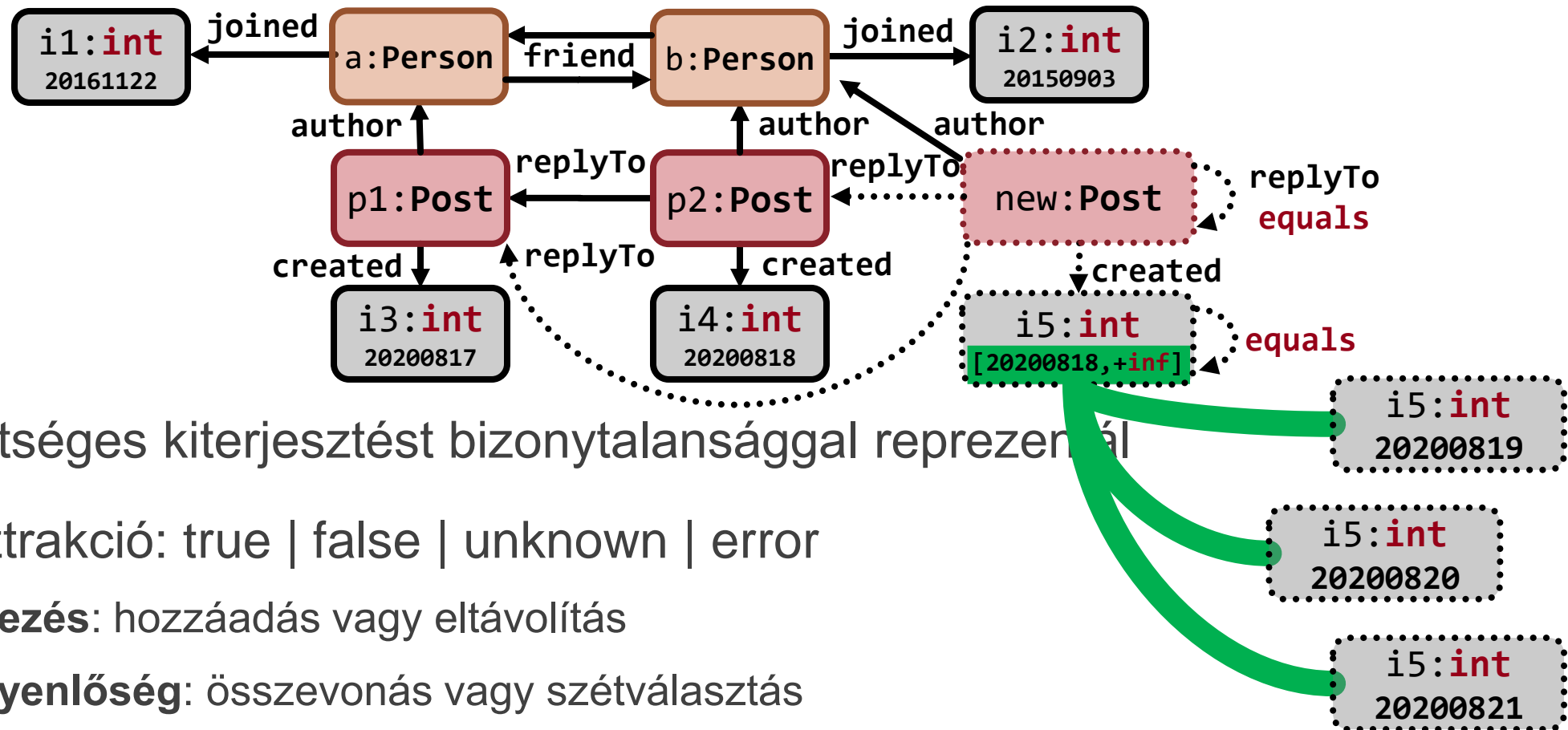
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: `true` | `false` | `unknown` | `error`
 - > 4 értékű **létezés**: hozzáadás vagy eltávolítás
 - > 4 értékű **egyenlőség**: összevonás vagy szétválasztás

Parciális modellezés: egyenlőség



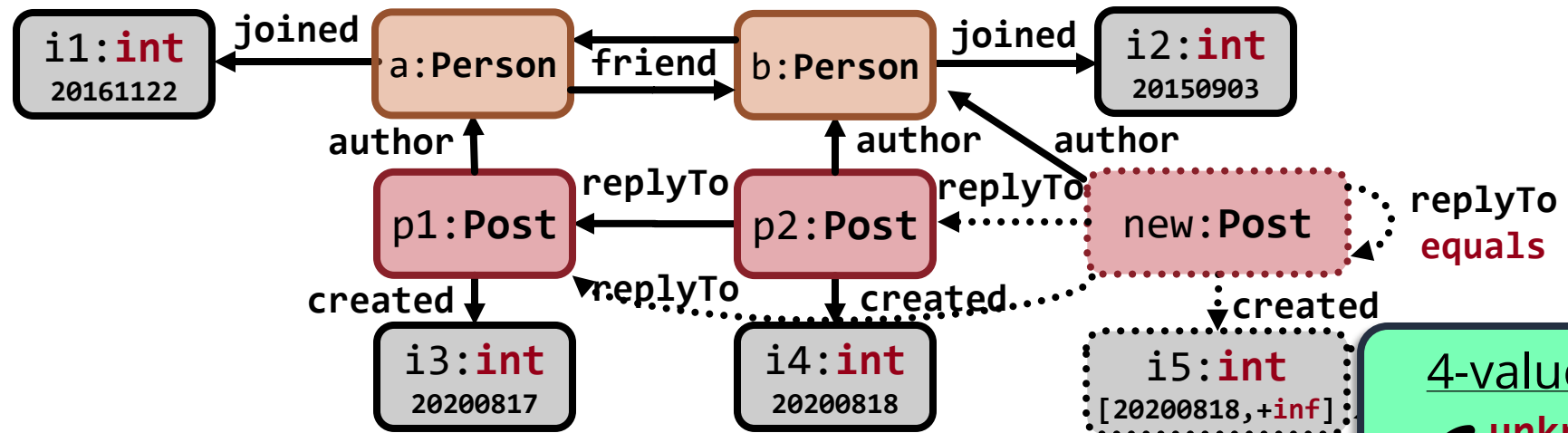
- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | unknown | error
 - > 4 értékű **létezés**: hozzáadás vagy eltávolítás
 - > 4 értékű **egyenlőség**: összevonás vagy szétválasztás

Parciális modellezés: egyenlőség

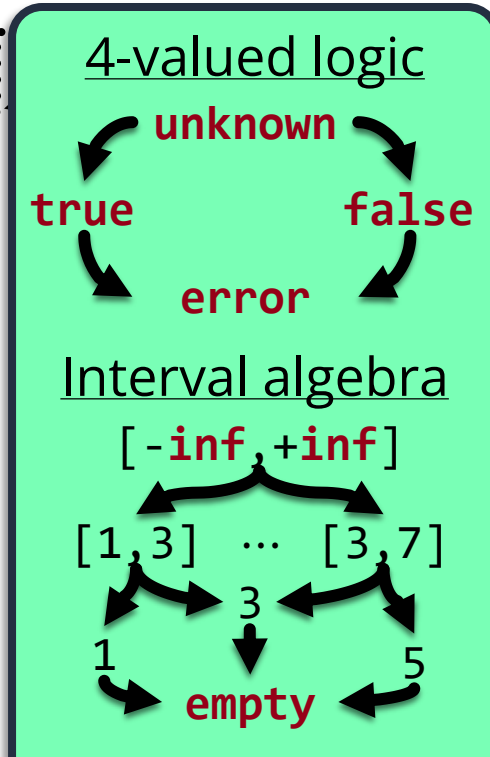


- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | unknown | error
 - > 4 értékű **létezés**: hozzáadás vagy eltávolítás
 - > 4 értékű **egyenlőség**: összevonás vagy szétválasztás
- Numerikus absztrakció: konkrét értékek → intervallumok

Parciális modellezés: egyenlőség



- Minden lehetséges kiterjesztést bizonytalansággal reprezentál
- Logikai absztrakció: true | false | unknown | error
 - > 4 értékű **létezés**: hozzáadás vagy eltávolítás
 - > 4 értékű **egyenlőség**: összevonás vagy szétválasztás
- Numerikus absztrakció: konkrét értékek → intervallumok
- **Finomítás**: csökkenti a bizonytalanságot → konkrét modellek



Finomítás

- A P parciális modellből Q -ra történő **finomítást** egy $ref: O_P \rightarrow 2^{O_Q}$ függvény határozza meg, amely tiszteletben tartja az információrendezést:
 - > Minden szimbólumra $s \in \Sigma$: $I_P(s)(\bar{p}) \sqsubseteq I_Q(s)(\widehat{ref(\bar{p})})$
 - > Q minden objektumát P valamelyik objektumából finomítjuk, és a létező $p \in O_P$ objektumoknak nem üres finomítással kell rendelkezniük.
- Egy **konkretizálás** egy konkrét modellre való finomítás.
- **Reguláris modellek:** a vizsgált parciális modellek részhalmaza (pl. kizárja az objektumok egyesítését, ha az nem praktikus)



Köszönöm a figyelmet!