Modelltranszformációk

Agenda

- Recap: Model Transformation
- Introduction to VIATRA2
- Research Highlights
- Summary

MODELLTRANSZFORMÁCIÓK

Modellvezérelt rendszertervezés

Tervezés

Analízis

Szintézis

Modellvezérelt rendszertervezés

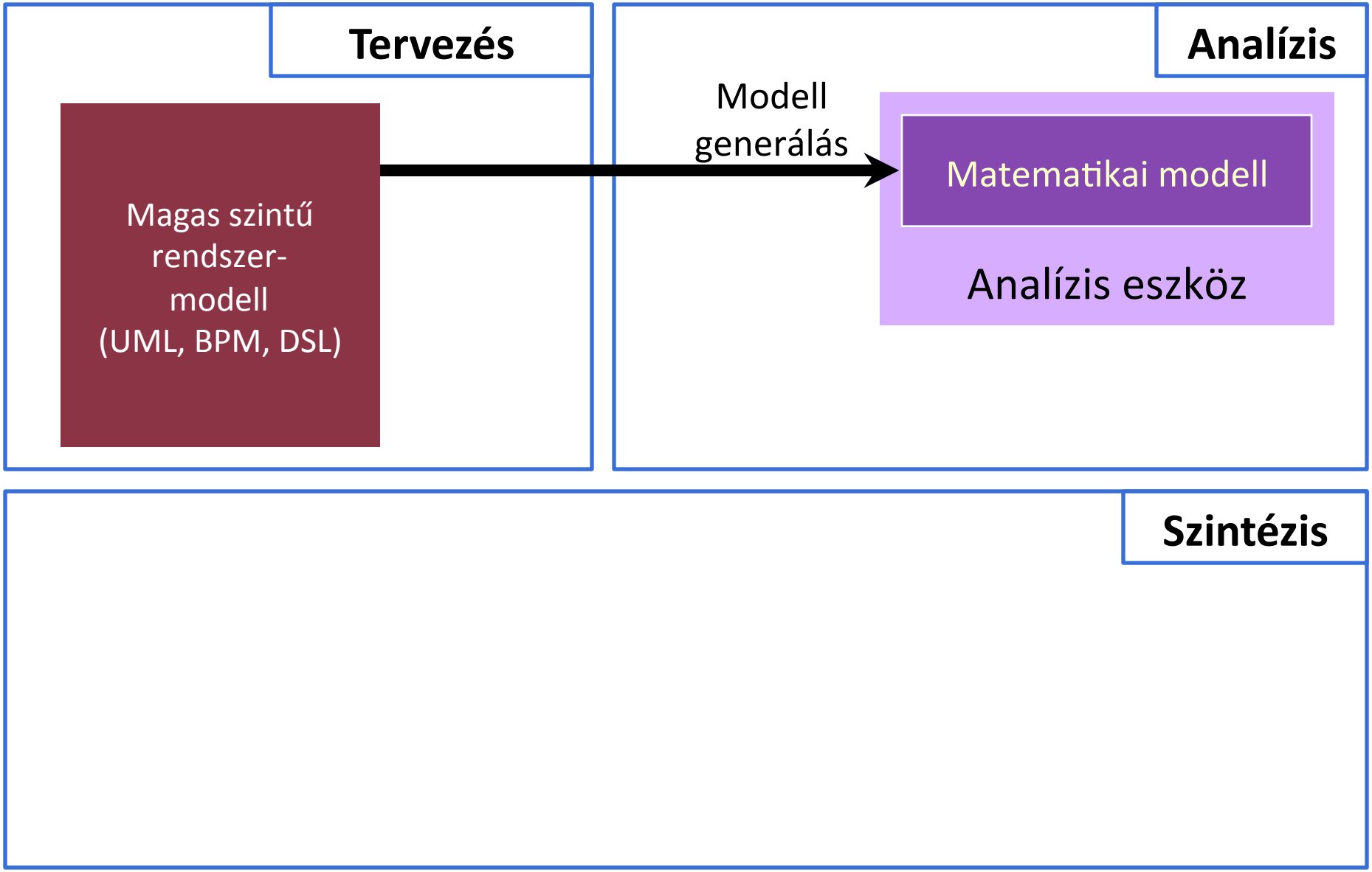
Tervezés

Magas szintű rendszer-modell (UML, BPM, DSL)

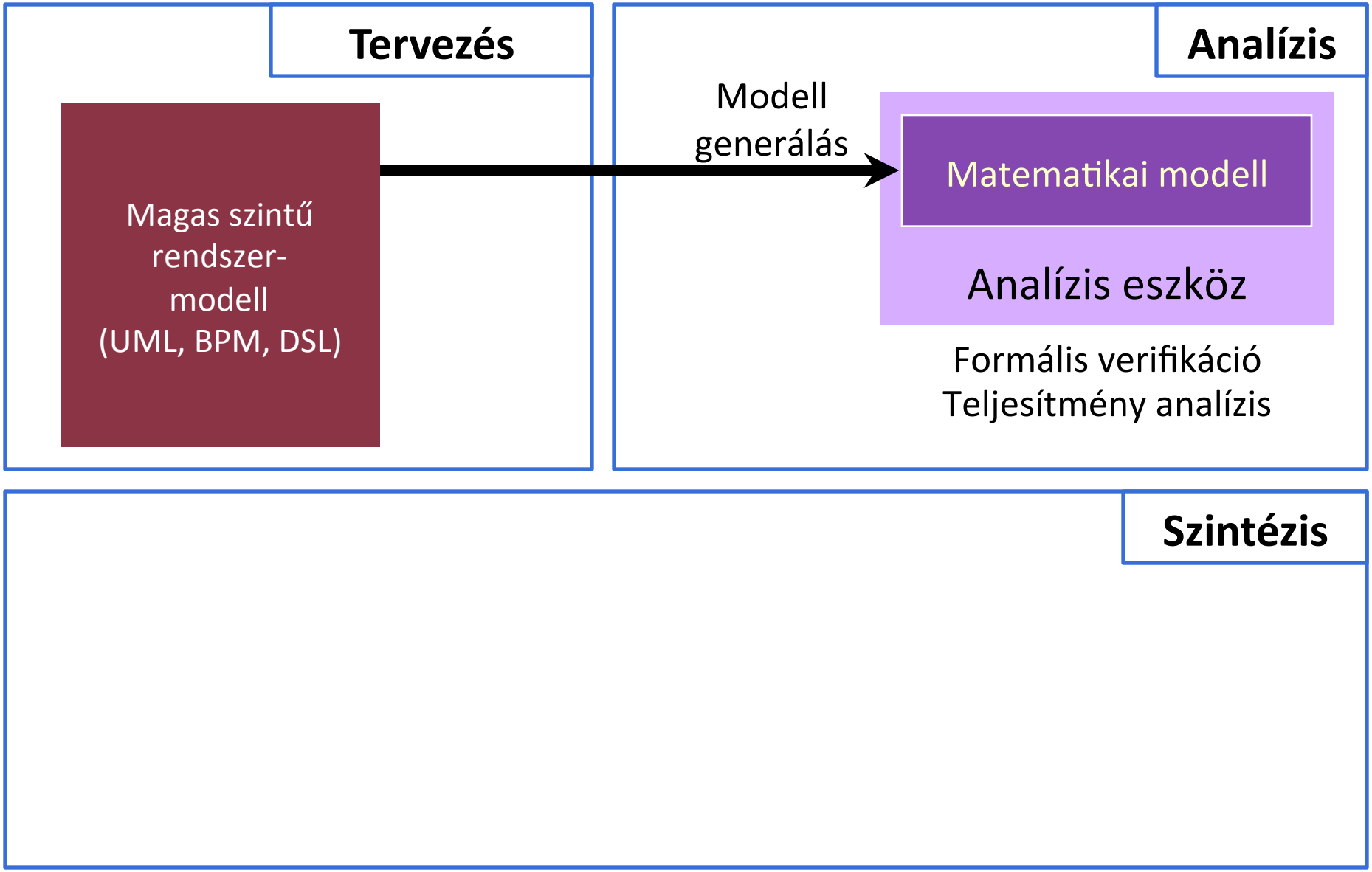
Analízis

Szintézis

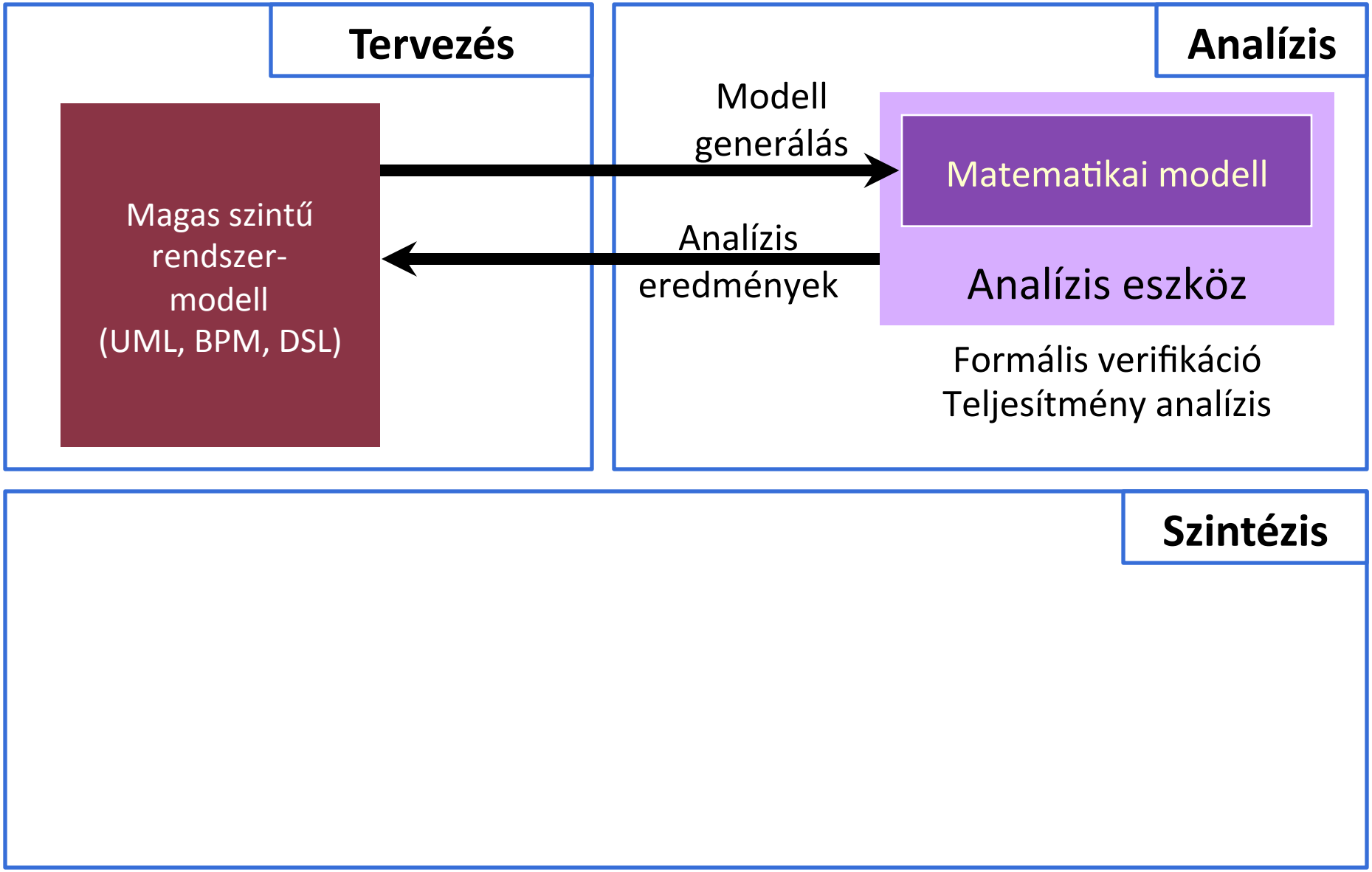
Modellvezérelt rendszertervezés



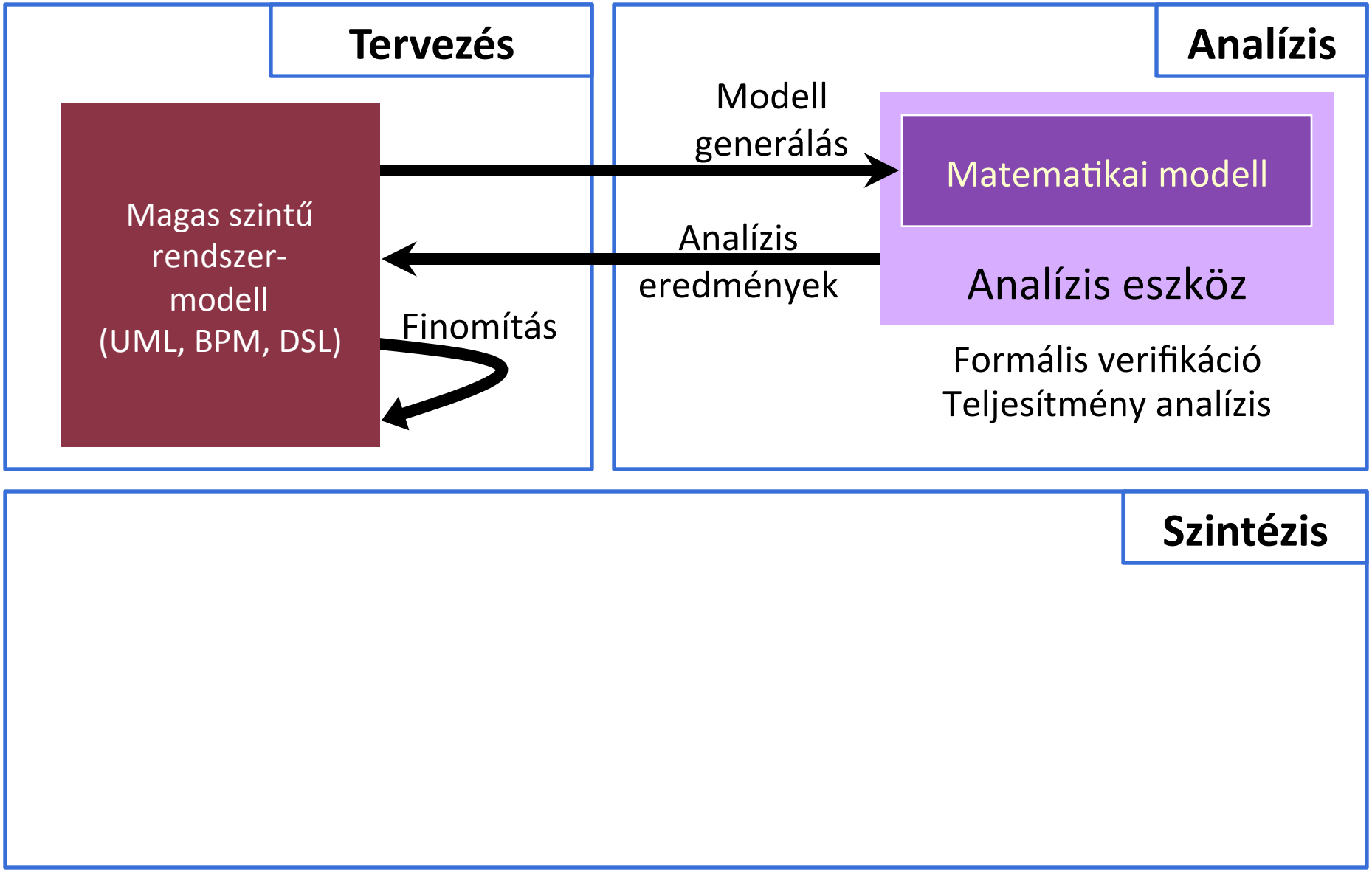
Modellvezérelt rendszertervezés



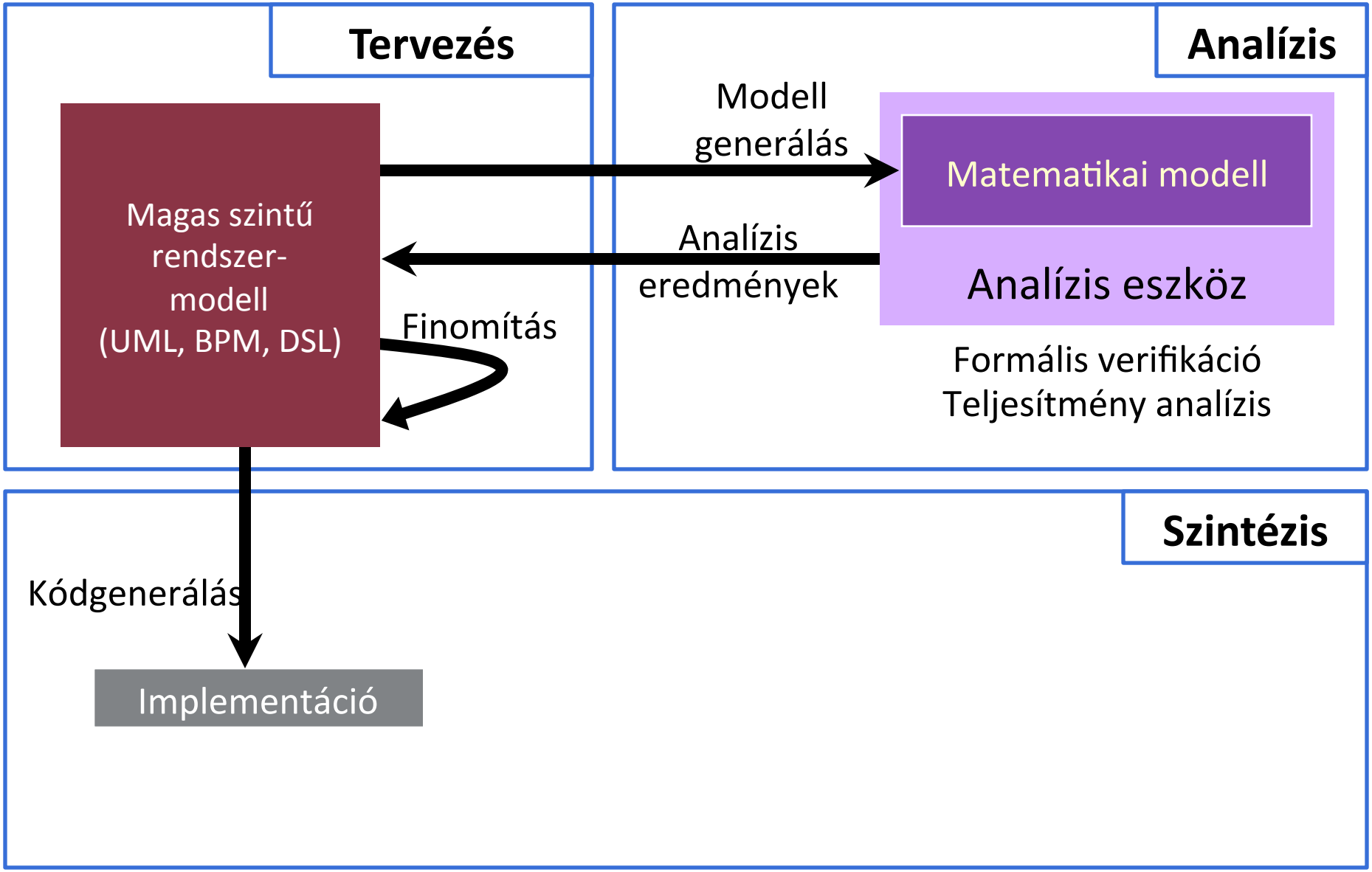
Modellvezérelt rendszertervezés



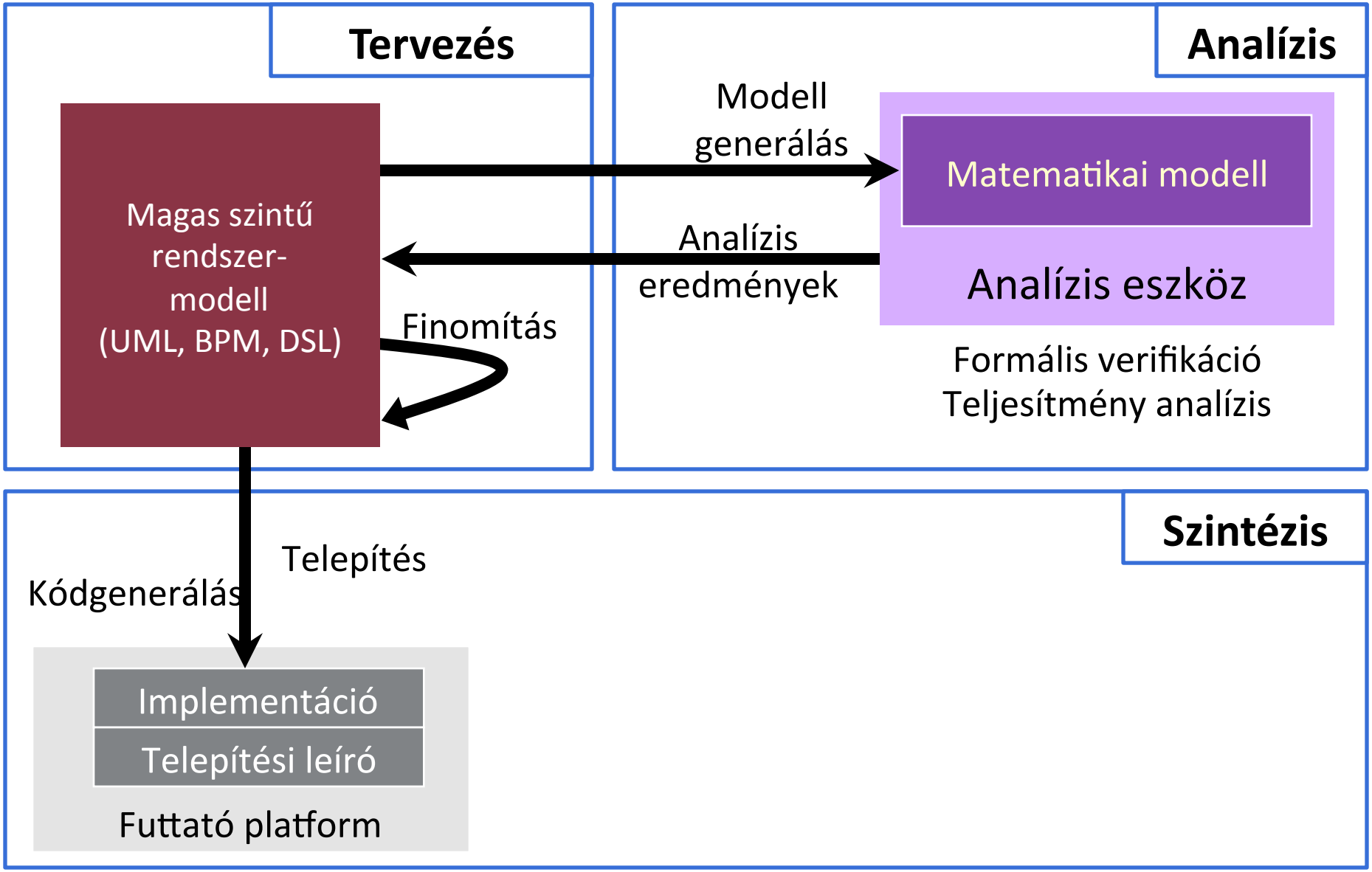
Modellvezérelt rendszertervezés



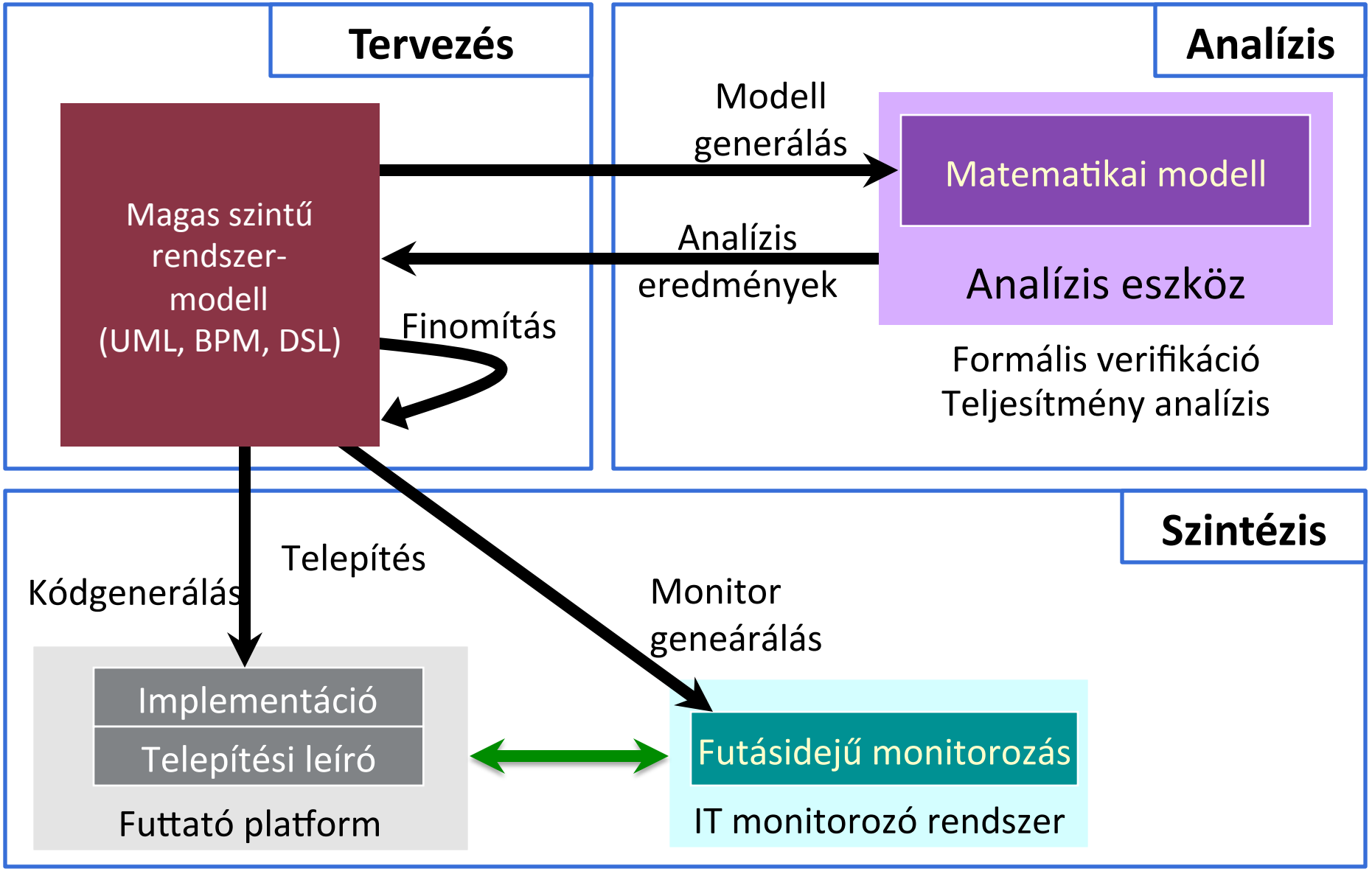
Modellvezérelt rendszertervezés



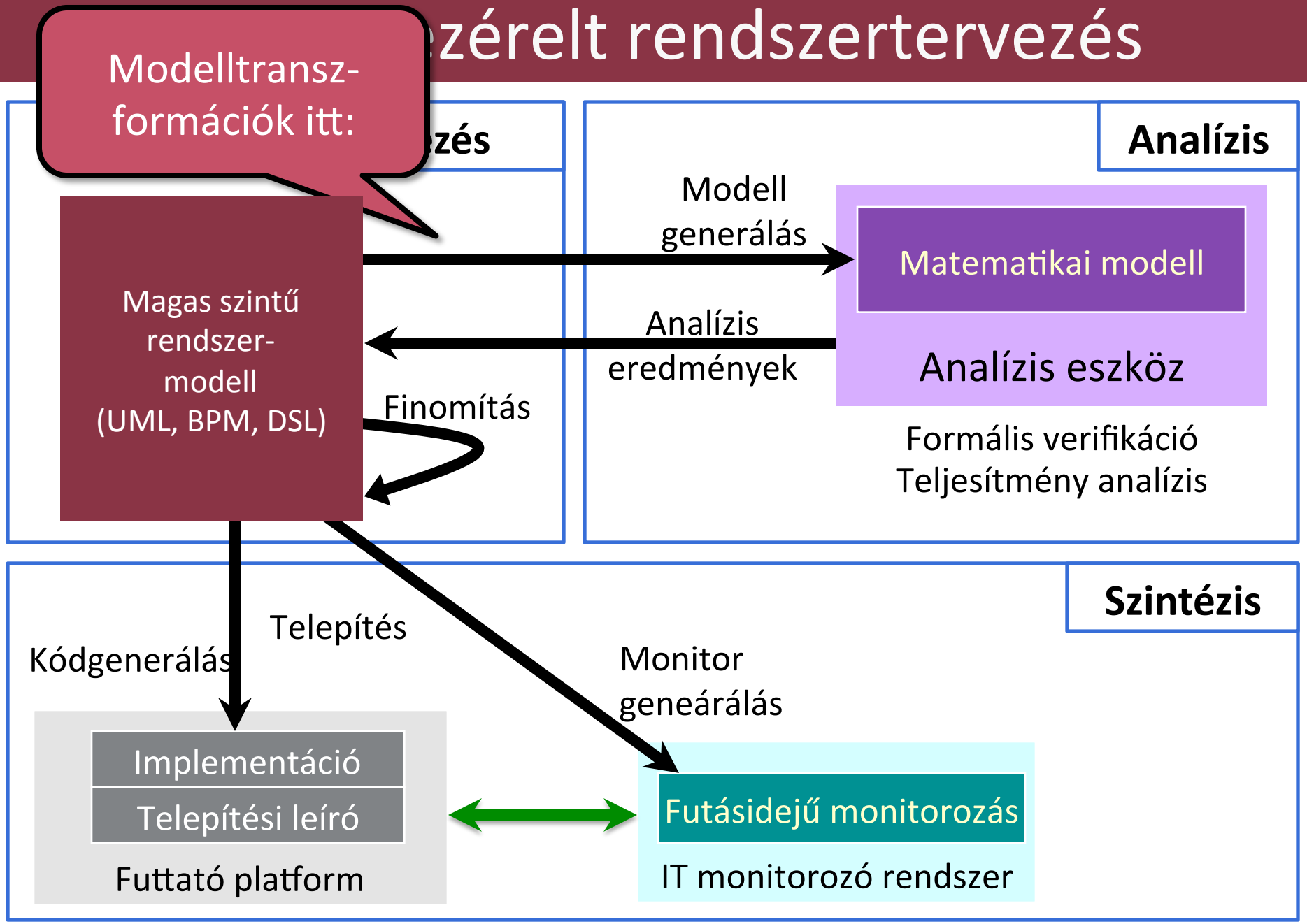
Modellvezérelt rendszertervezés



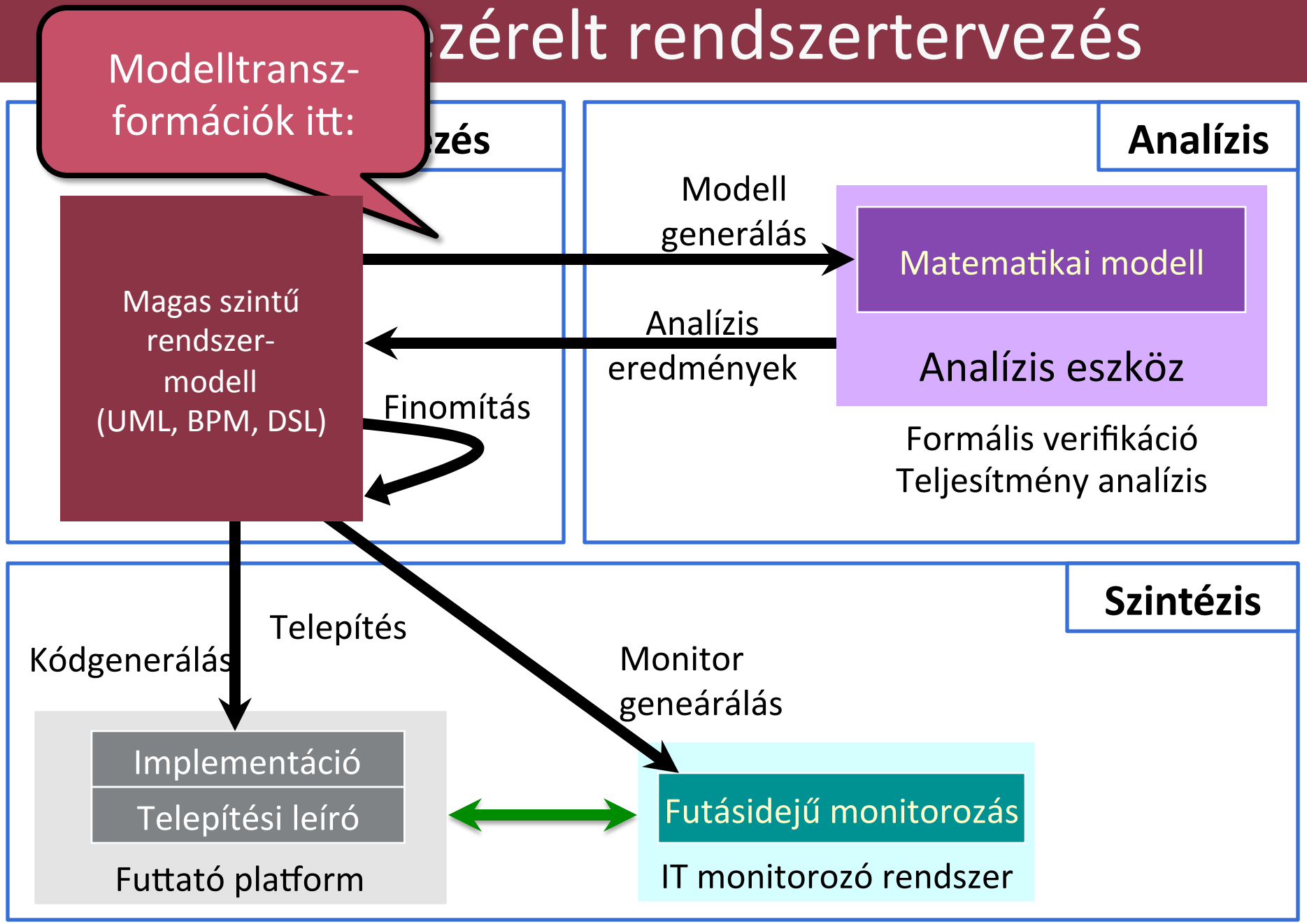
Modellvezérelt rendszertervezés



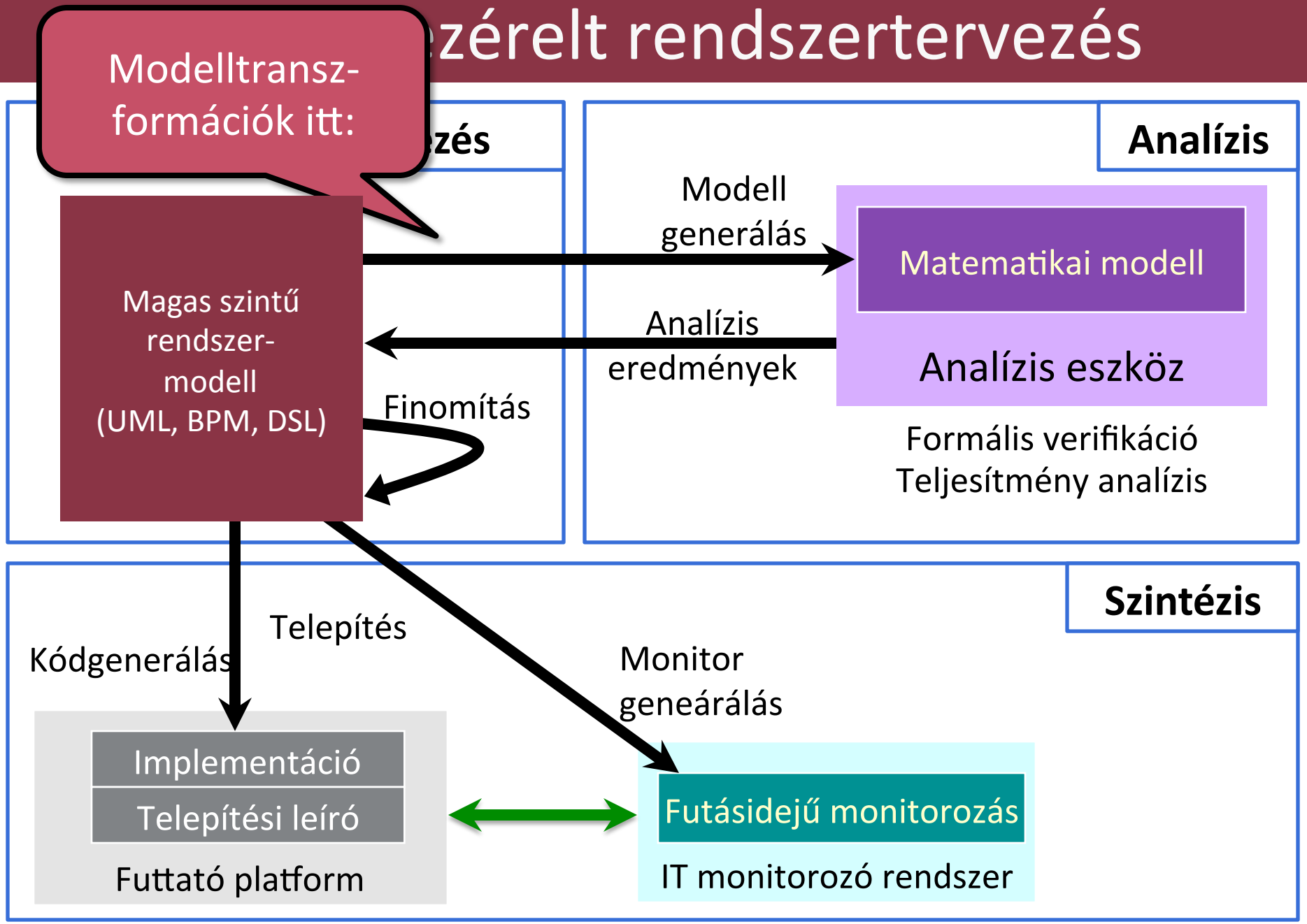
ezérelt rendszertervezés



ezérelt rendszertervezés



ezérelt rendszertervezés



Modelltranszformációk

- Modelltranszformációk (MT) modellek automatikus
  - lekérdezése,
  - módosítása,
  - leképzése,
  - származtatása
- Eszköztámogatás javítandó…
  - Fejlesztési idő
  - Transzformáció hibakeresés
  - Végrehajtás teljesítménye
  - Ellenőrizhetőség

Modelltranszformációs alkalmazások

- Beágyazott rendszerek
  - Eszközsorozat kritikus beágyazott rendszerekhez (DECOS)
    - PIM → PSM leképezés
    - Kódgenerálás, konfigurációs állományok szintézise
    - Formális ellenőrzések
    - Tesztgenerálás (MOGENTES)
  - Jármű hálózatok (HIDENETS), repülőipar (DIANA)
- Szolgáltatás-orientált rendszerek (SENSORIA)
  - Szolgáltatás ellenőrzés, teljesítmény elemzés
  - Kódgenerálás, konfigurációs állományok szintézise
- Biztonságkritikus tervezés változó környezetben (SECURECHANGE)

MODELLTRANSZFORMÁCIÓS RENDSZEREK

Modelltranszformációs eszközök

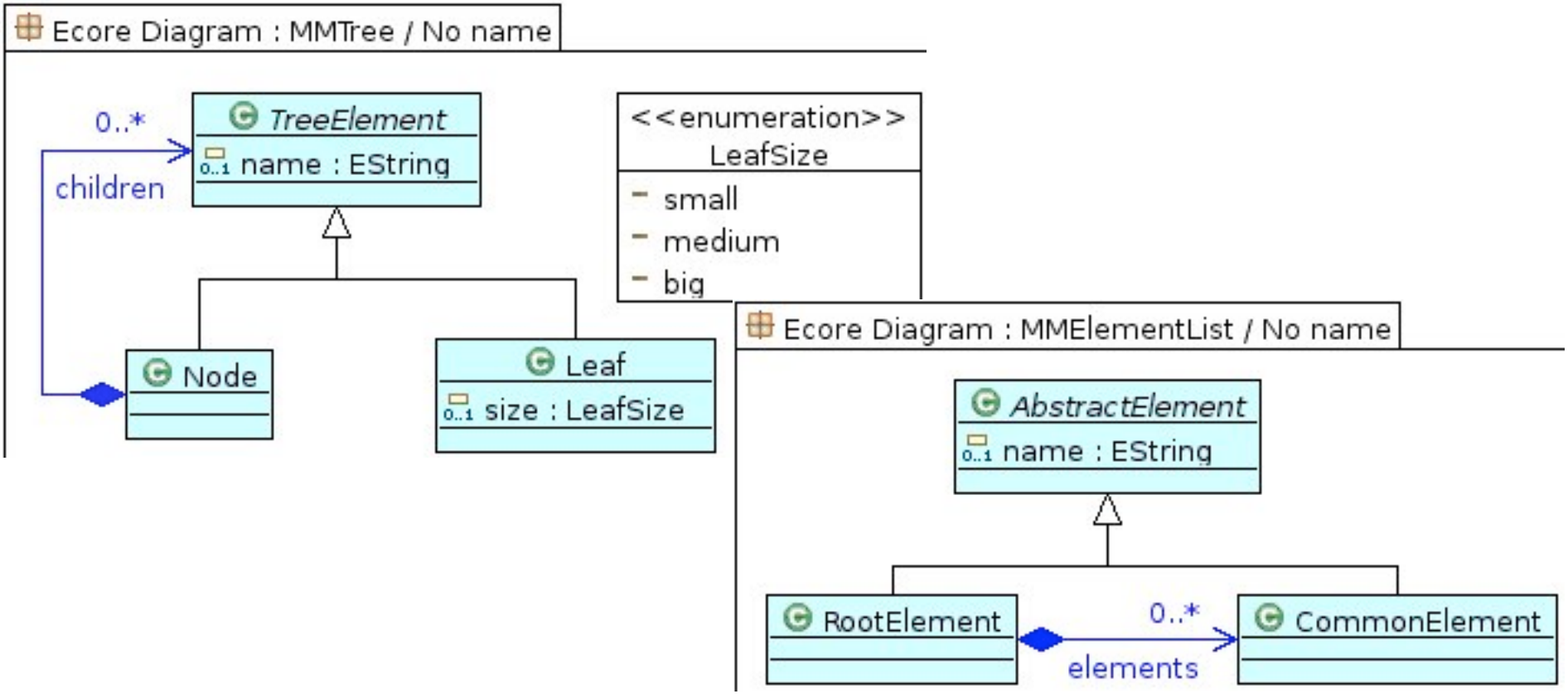
- Sokféle eszköz
  - Transformation Tool Contest 2010
  - GReTL, Epsilon Flock, UML-RSDS, Mola, Fujaba, GrGen, COPE, PETE, ATL+Java, Groove, Astra, Mitra, Henshin, VIATRA2, ETL

ATL (ATL Transformation Language)

- Deklaratív és imperatív transzformációs nyelv
- QVT (Query/View/Transformation) alapú

ATL Példa: Tree to List

- http://www.eclipse.org/m2m/atl/basicExamples_Patterns/article.php?file=Tree2List/index.html



Egyszerű részszabály

```
rule TreeNode2RootElement {
 from
   rt : MMTree!Node
 to
   lstRt : MMElementList!RootElement (
      name <- rt.name
    )
 }
```

Epsilon projekt

- Szöveges nyelvek EMF modellek kezeléséhez
  - Modellek leírása
  - Transzformáció (ETL: Epsilon Transformation Language)
  - Migráció (Epsilon Flock)
  - …
- Kezelő környezet
  - Végrehajtó motor

ETL

- Deklaratív szabálydefiníció
- Imperatív szabálytörzs

ETL példa: Tree model to Graph Model

http://www.eclipse.org/gmt/epsilon/examples/index.php?example=org.eclipse.epsilon.examples.tree2graph

```
@namespace(uri="Tree", prefix="Tree")
package Tree;

class Tree {
    val Tree[*]#parent children;
    ref Tree#children parent;
    attr String label;
}
```

```
@namespace(uri="Graph", prefix="Graph")
package Graph;

class Graph {
    val Node[*] nodes;
}

class Node {
    attr String name;
    val Edge[*]#source outgoing;
    ref Edge[*]#target incoming;
}

class Edge {
    ref Node#outgoing source;
    ref Node#incoming target;
}
```

ETL példa

```
rule Tree2Node
  transform t : Tree!Tree
  to n : Graph!Node {
    n.name := t.label;
    if (t.parent.isDefined()) {
      var e : new Graph!Edge;
      e.source ::= t.parent;
      e.target := n;
    }
  }
```

EMFT Henshin

- Gráftranszformáció
- EMF modellek felett
- Grafikus szintaxis

Példa: Henshin

extlibrary.henshin_diagram

ReplaceBook:Library

<<preserve>>
w:Writer
name=n->n.toUpperCase()

<<create>>
:BookOnTape
title=t

<<create>>
author

<<delete>>
author

<<delete>>
:Book
<<forbid:0>> category=Mystery
title=t

<<forbid:1>>
:Borrower

<<forbid:1>>
borrowers

Palette

Rule
Edge
Attribute

extlibrary

Addressable
AudioVisualItem
Book
BookOnTape
Borrower
CirculatingItem
Employee
Item
Lendable
Library
Periodical
Person
VideoCassette
Writer

extlibrary.henshin

Resource Set

platform:/resource/org.eclipse.emf.henshir
Transformation System
Rule ReplaceBook
Graph LHS
Node root:Library
Node w:Writer
Node :Book
Edge root -> w
Edge
Edge
And
Graph RHS
Node root:Library
Node w:Writer
Node :BookOnTape
Edge root -> w
Edge
Edge
Mapping root -> root
Mapping w -> w

Selection | Parent | List | Tree | Table | »1

BEVEZETŐ A VIATRA2 RENDSZERHEZ

VIATRA2 – A mi transzformációs eszközünk

- **VIATRA2**
  - **Eclipse** Modelling Subproject
  - http://wiki.eclipse.org/VIATRA2
  - http://www.eclipse.org/gmt/VIATRA2/
- BME Hibatűrő Rendszerek Kutatócsoport fejlesztés
  - Több EU projektben felhasználva
    - DECOS
    - SENSORIA
    - DIANA
    - MOGENTES
    - SECURECHANGE

Mi a VIATRA2?

Mi a VIATRA2?

Mi a VIATRA2?

- MT platform
  - Transzformáció végrehajtás
  - Modellreprezentáció
- Transzformációs nyelv
  - Transzformációleírásra szabva
- Fejlesztőkörnyezet
  - Eclipse technológiákra épül
- Bővíthető keretrendszer
  - Különböző modellreprezentációk kezelése (UML, EMF, etc.)
  - Transzformációs funkciók kiterjesztése

Mi a VIATRA2?

- MT platform
  - Transzformáció végrehaj
  - Modellreprezentáció
- Transzformációs nyelv
  - Transzformációleírásra szabva
- Fejlesztőkörnyezet
  - Eclipse technológiákra épül
- Bővíthető keretrendszer
  - Különböző modellreprezentációk kezelése (UML, EMF, etc.)
  - Transzformációs funkciók kiterjesztése

Deklaratív modellfeldolgozás optimalizált végrehajtással

Mi a VIATRA2?

- MT platform
  - Transzformáció végrehajtás
  - Modellreprezentáció
- Transzformációs nyelv
  - Transzformációleírásra szabva
- Fejlesztőkörnyezet
  - Eclipse technológiákra épül
- Bővíthető keretrendszer
  - Különböző modellreprezentációk kezelése (UML, EMF, etc.)
  - Transzformációs funkciók kiterjesztése

Mi a VIATRA2?

- MT platform
  - Transzformáció végrehajtás
  - Modellreprezentáció
- Transzformációs nyelv
  - Transzformációleírásra szabva
- Fejlesztőkörnyezet
  - Eclipse technológiákra épül
- Bővíthető keretrendszer
  - Különböző modellreprezentációk kezelése (UML, EMF, etc.)
  - Transzformációs funkciók kiterjesztése

Mi a VIATRA2?

- MT platform
  - Transzformáció végrehajtás
  - Modellreprezentáció
- Transzformációs nyelv
  - Transzformációleírásra szabva
- **Fejlesztőkörnyezet**
  - **Eclipse technológiákra épül**
- Bővíthető keretrendszer
  - Különböző modellreprezentációk kezelése (UML, EMF, etc.)
  - Transzformációs funkciók kiterjesztése

Első pillantás az eszközre

VIATRA2 - ViatraLabSession/activity2petrinet/Activity2PetriNet.vtcl - Eclipse Platform

File Edit Navigate Search Project Window Help

Navigator

- SensusDeployment
- testing.repository 1177 [https://svn.inf.mit.bme.h
- UmlToAS_Viatra
- > ViatraLabSession 49 [https://svn.inf.mit.bme.hu
  - > activity2petrinet 22
    - > meta 21
      - > ad2petri_tracemeta.vtml 21
      - > ad2petri_tracemeta2.vtml 21
      - petri_meta.vtml 18
    - pnml 22
      - ComplexActivity.pnml 22
      - ForkJoinActivity.pnml 22
      - Myactivity.pnml 22
    - uml 21
      - act_prb.uml 18
      - act_prb.umlact 18
      - activity_complex.uml 21
      - activity_complex.umlact 21
      - activity_forkjoin.uml 21
      - activity_forkjoin.umlact 21
    - activity2petrinet.vpml 21
    - > Activity2PetriNet.vtcl 22
    - PetriNet2PNML.vtcl 20
  - faultInjection 17

VIATRA2 Model spaces

- modelSpace1 (uml_galileo.vpml)
  - Native functions
  - Program models
    - ad2petri.Activity2PetriNet (Activity2PetriNet.v
    - petrinet.petrinet2PNML (PetriNet2PNML.vtcl)
  - Triggers (serial execution mode)

*uml_galileo.vpml | *Activity2PetriNet.vtcl (modelSpace1) | PetriNet2PNML.vtcl (modelSpace1)

```
240 // TRANSFORM Fork/Join/Final Nodes
241
242 gtrule transformForkJoinFinalNode(out ActivityNode, in PetriNet) = {
243     precondition pattern unmappedForkJoinFinalNode(ActivityNode) = {
244         ad2petri.helpermetamodel.forkJoinFinalNode(ActivityNode);
245         neg find activityNodeTransitionMapping(ActivityNode, NoPetriTransition);
246     }
247     postcondition pattern mapped(ActivityNode, PetriTransition, PetriNet) = {
248         ad2petri.helpermetamodel.forkJoinFinalNode(ActivityNode);
249         find activityNodeTransitionMapping(ActivityNode, PetriTransition);
250         find transitionOfNet(PetriTransition, PetriNet);
251     }
252     action {
253         call copyName(ActivityNode, PetriTransition);
254         forall EdgePlace with apply connectNodeToOutgoing(ActivityNode, PetriTransition, EdgePlace) do skip;
255         forall EdgePlace with apply connectNodeToIncoming(ActivityNode, PetriTransition, EdgePlace) do skip;
256     }
257 }
258
259 // TRANSFORM Decision/Merge Nodes
```

VIATRA2 Textual Output | Properties | Problems | History | Progress | Search | VIATRA2 Testing Results | Error Log

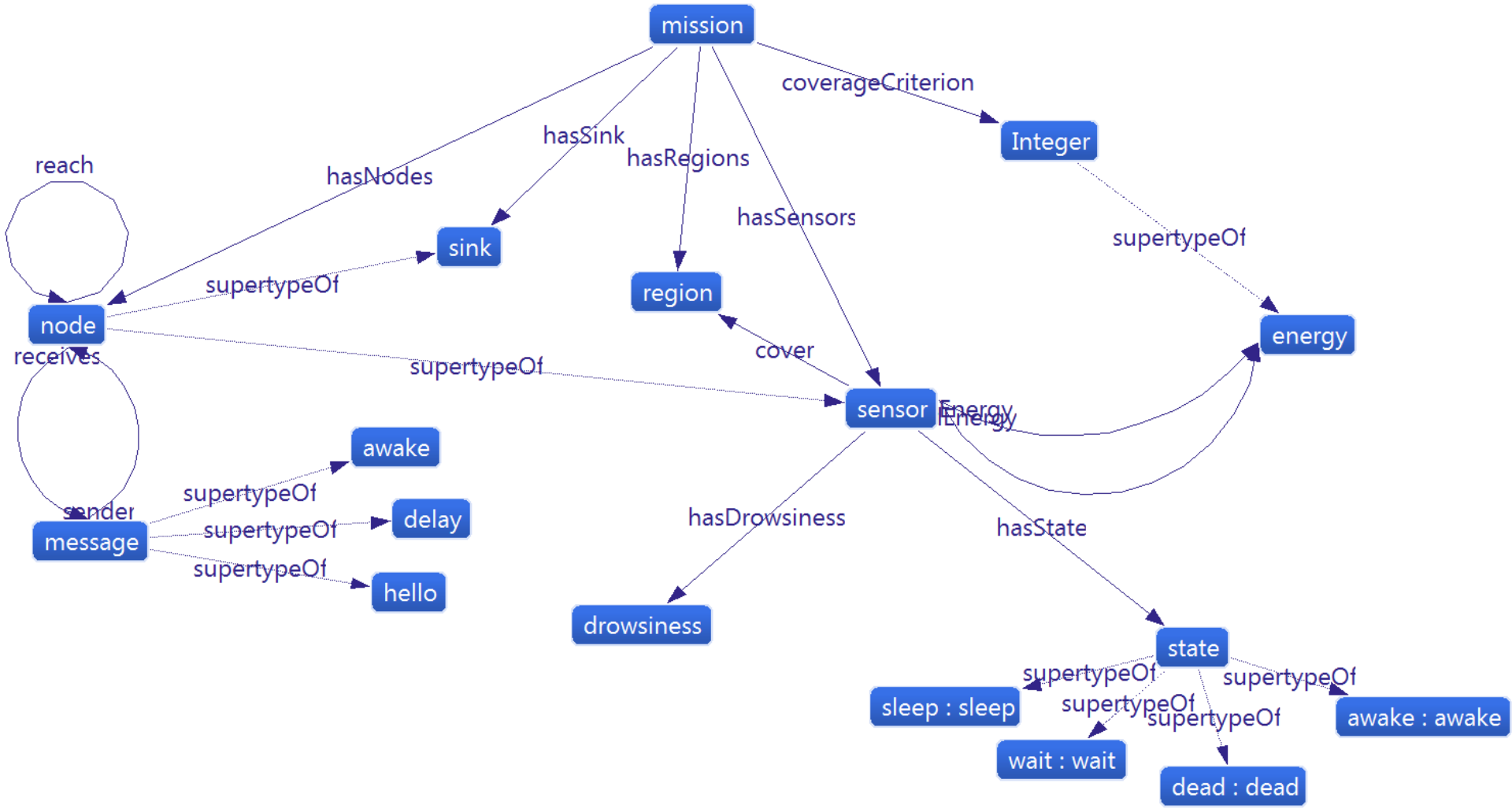
```
<pnml>
 <net id="n0" type="http://www.informatik.hu-berlin.de/top/pntd/ptNetb">
  <place id="OpaqueActionuN623_60">
   <name>
    <text>OpaqueActionuN623_60</text>
   </name>
   <initialMarking>
    <text>0</text>
   </initialMarking>
  </place>
  <place id="uN694_60">
   <name>
```

default | core://Myactivity.pnml

modelSpace1

Writable | Insert | 255 : 41 | 36M of 250M

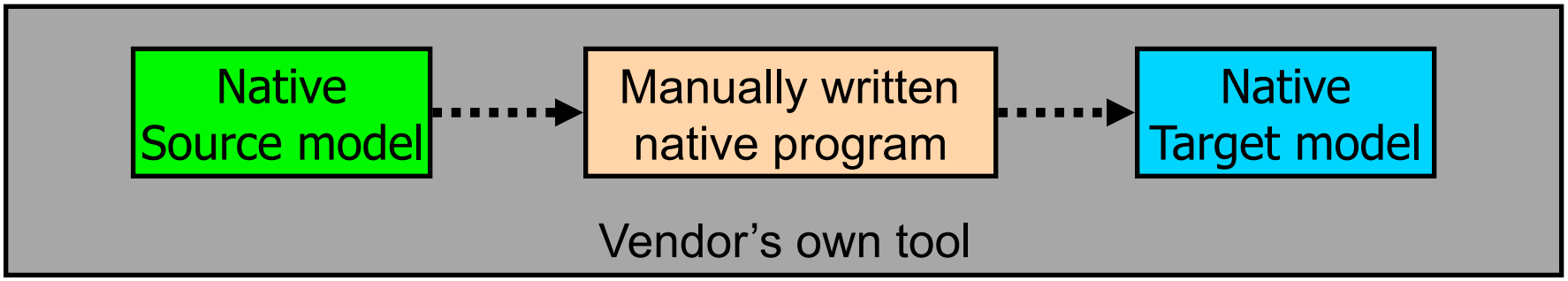
Modellek, metamodellek gráfként



Mi a VIATRA2?

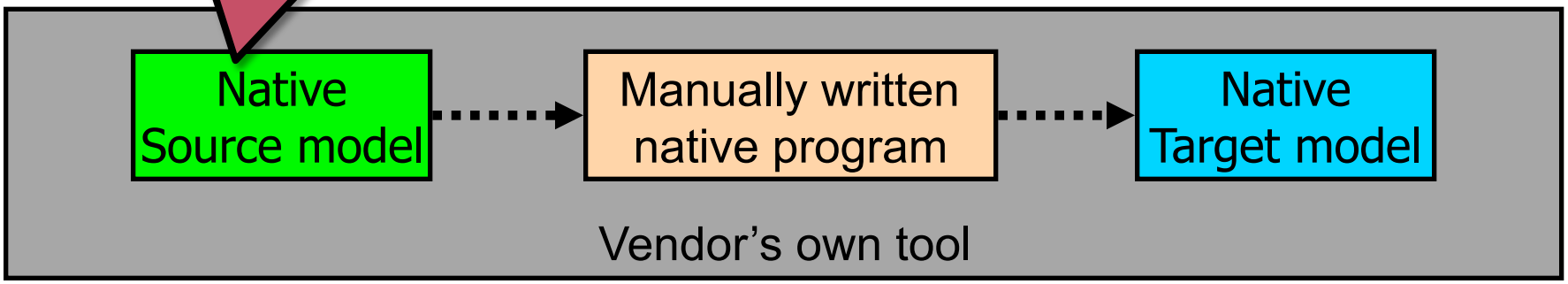
- **MT platform**
  - **Transzformáció végrehajtás**
  - **Modellreprezentáció**
- Transzformációs nyelv
  - Transzformációleírásra szabva
- Fejlesztőkörnyezet
  - Eclipse technológiákra épül
- Bővíthető keretrendszer
  - Különböző modellreprezentációk kezelése (UML, EMF, etc.)
  - Transzformációs funkciók kiterjesztése

Natív (Ad Hoc) modell transzformációk



Natív (Ad Hoc) modell transzformációk

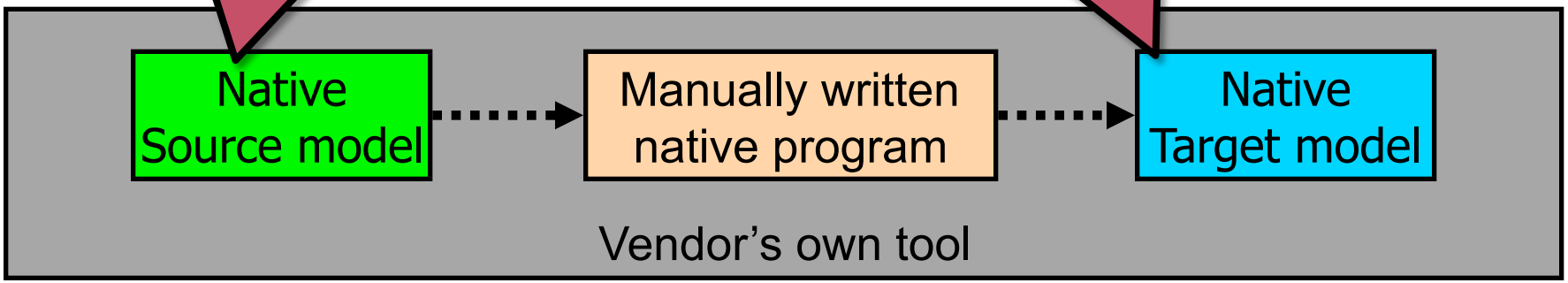
Natív forrásmodell
- EMF
- XML documentum
- Adatbázisok
- Domain-specifikus modell
- UML



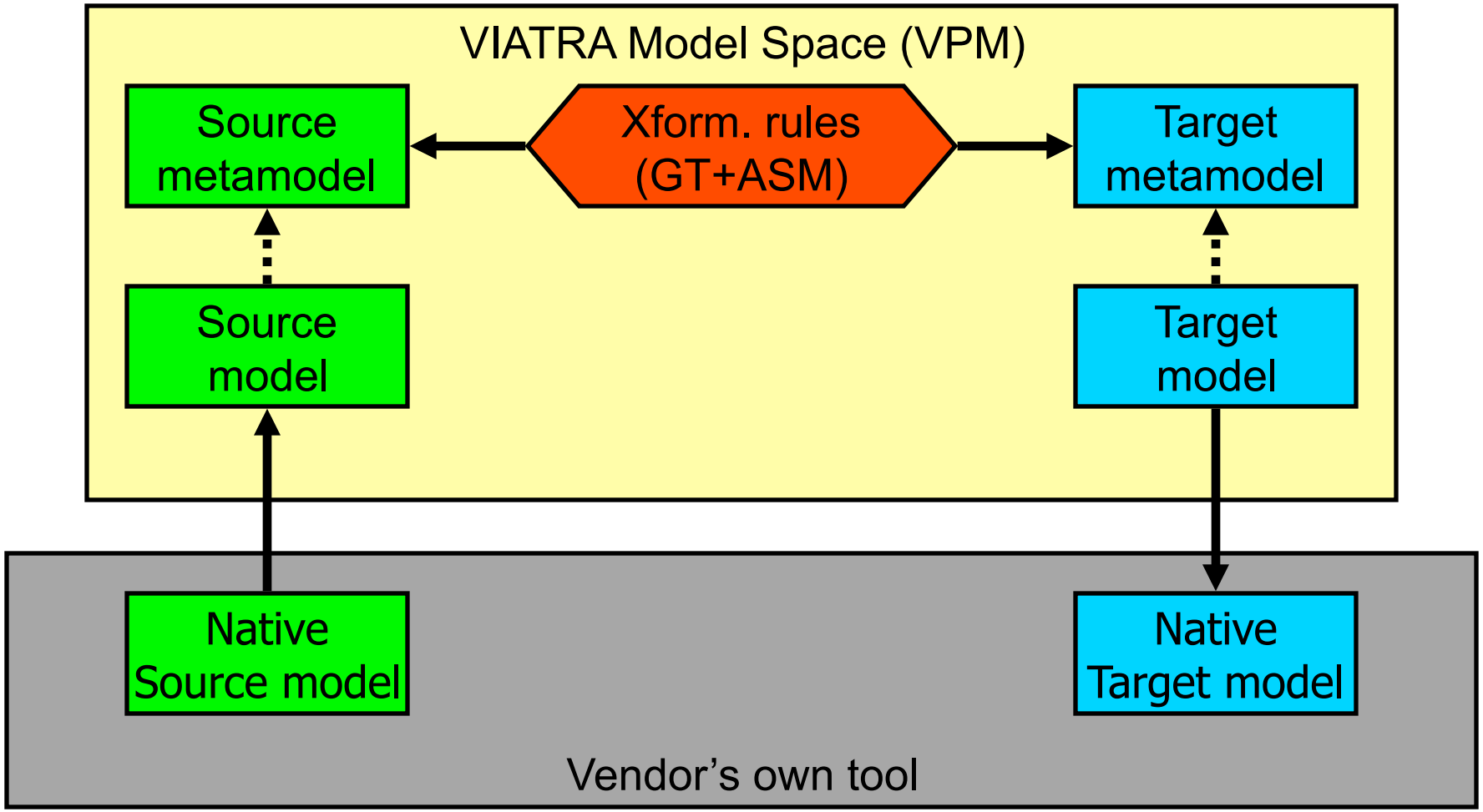
Natív (Ad Hoc) modell transzformációk

Natív forrásmodell
- EMF
- XML documentum
- Adatbázisok
- Domain-specifikus modell
- UML

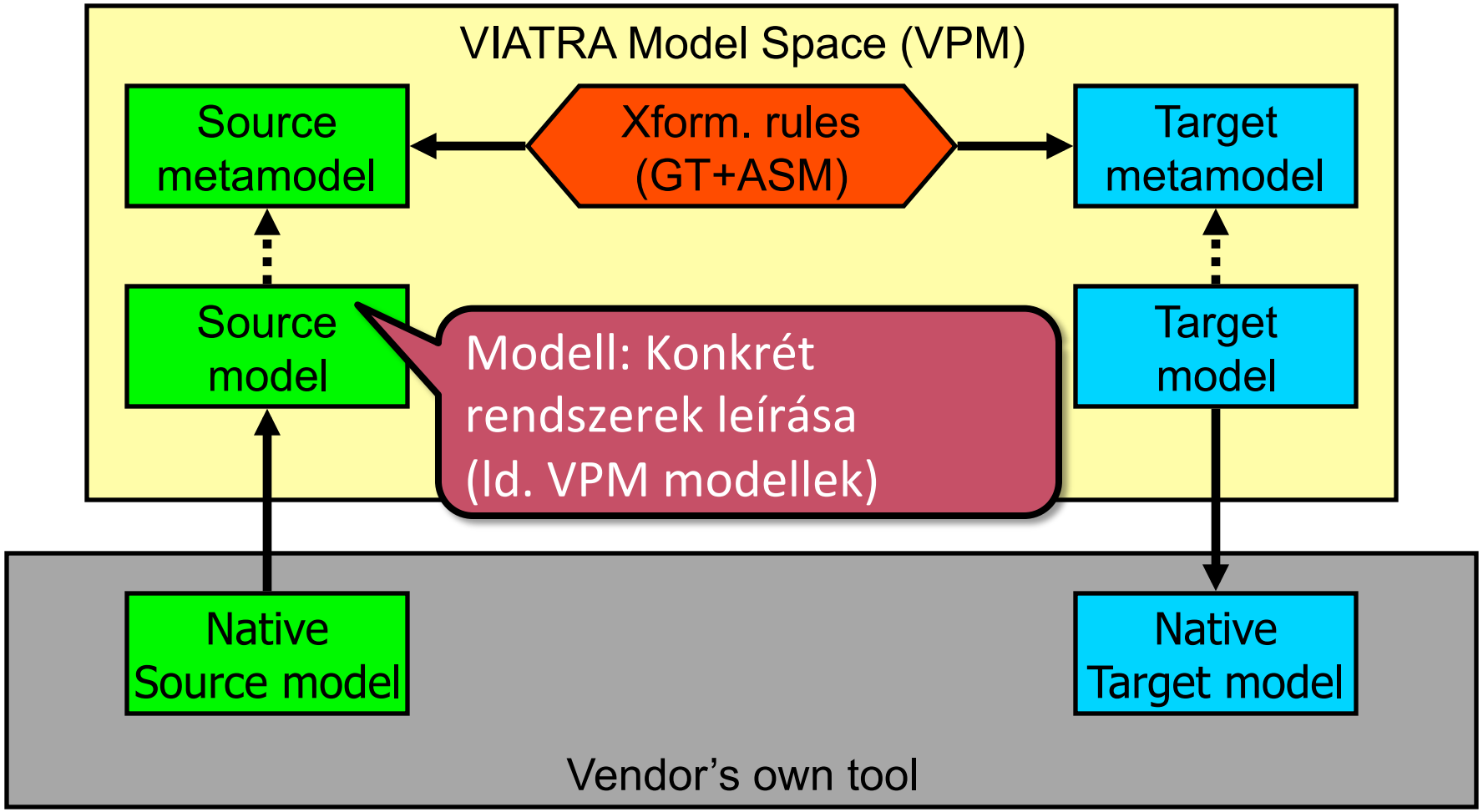
Native célmodell:
- EMF
- Forráskód
- XML leírók
- Adatbázisok
- Analízis eszközök



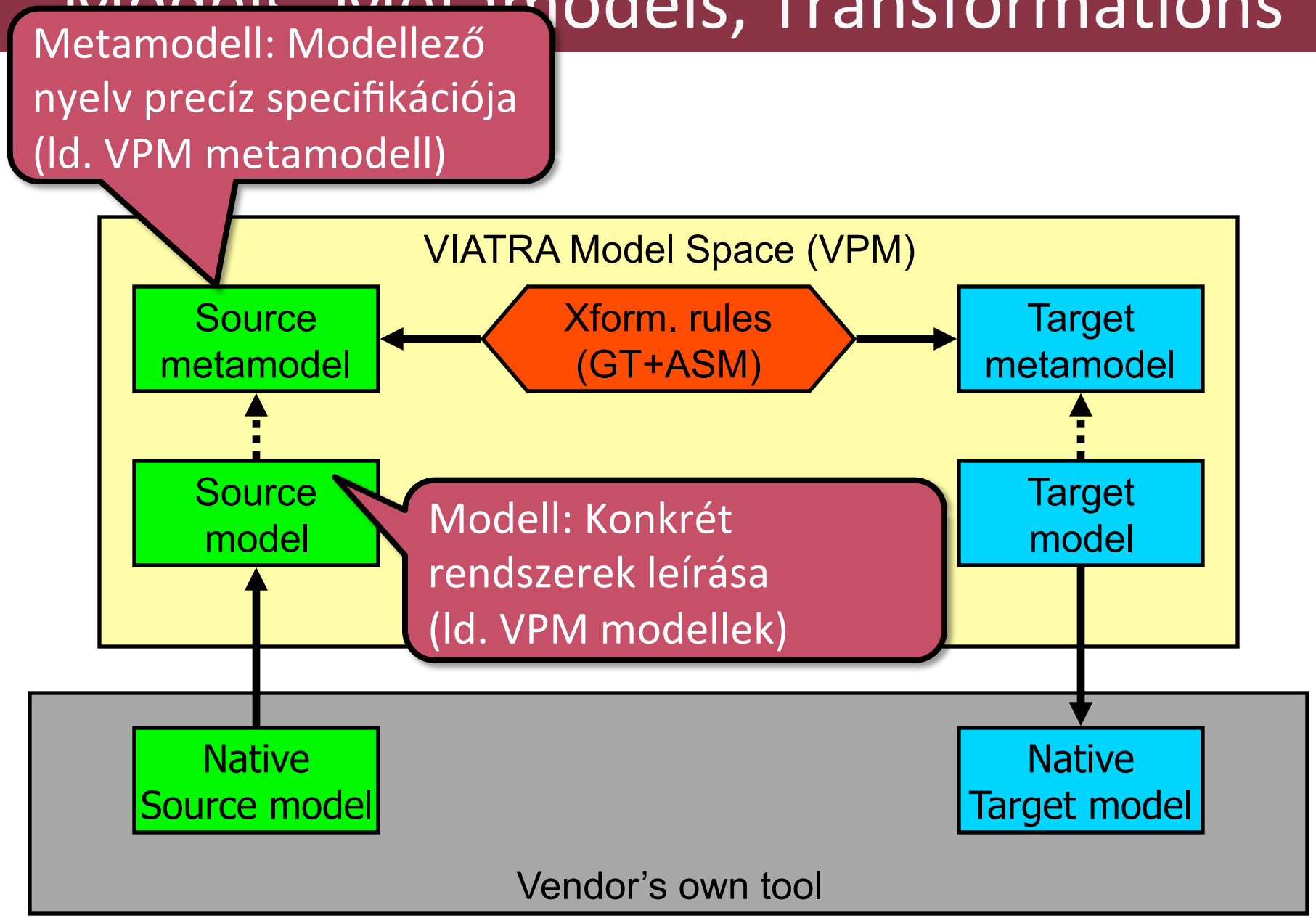
Models, Metamodels, Transformations



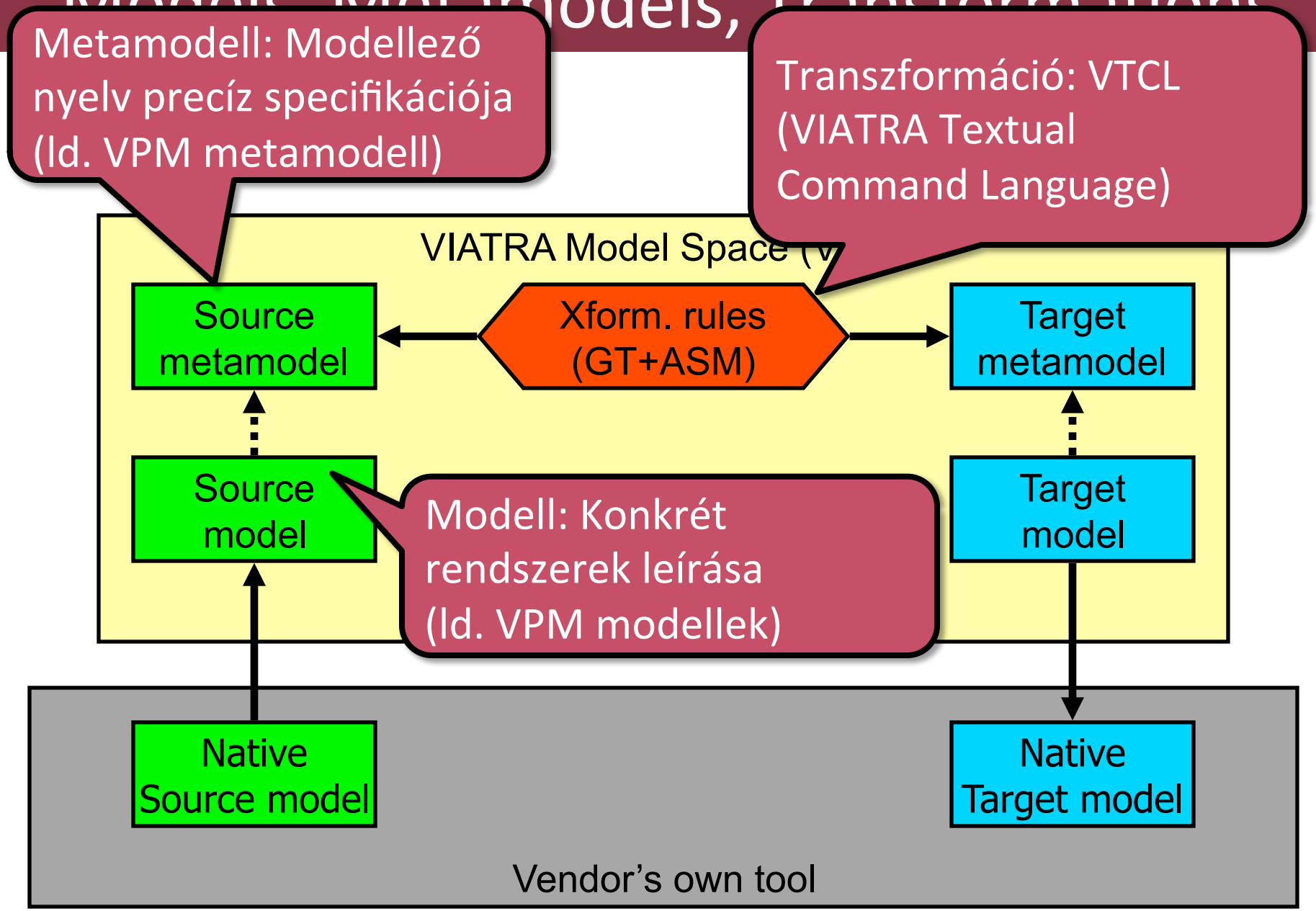
Models, Metamodels, Transformations



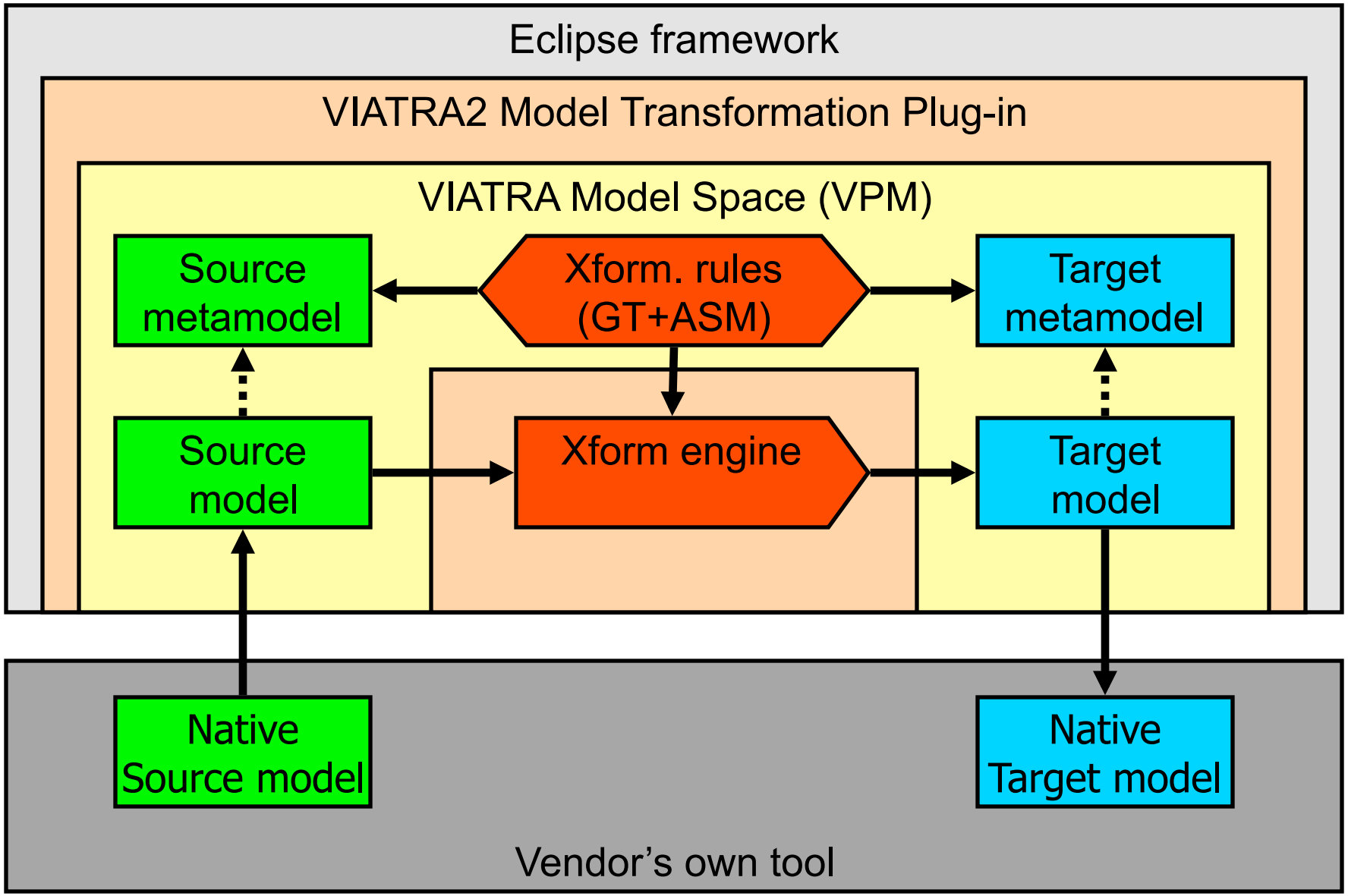
Models, Metamodels, Transformations



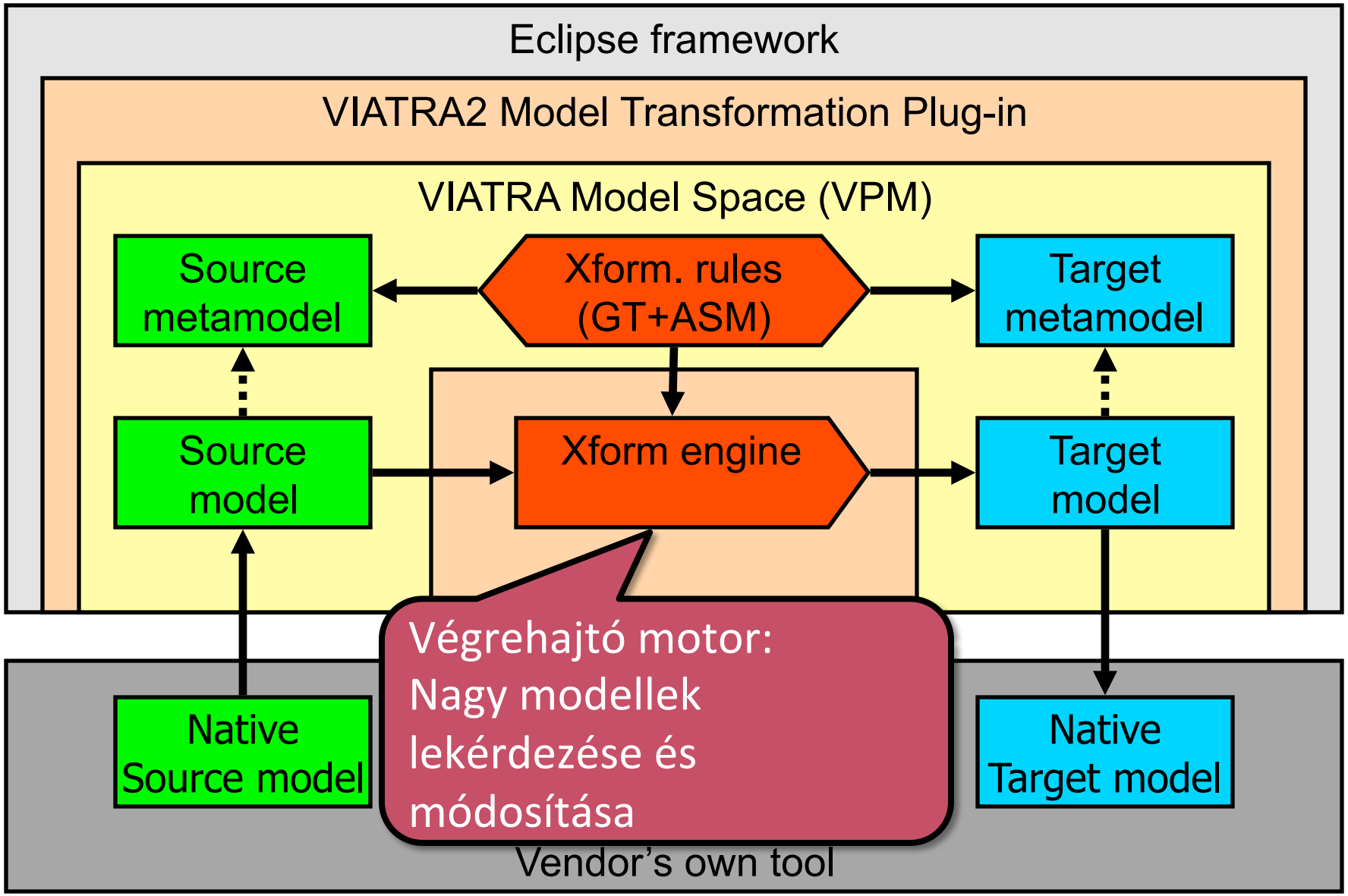
Models, Metamodels, Transformations



Transformation Engine



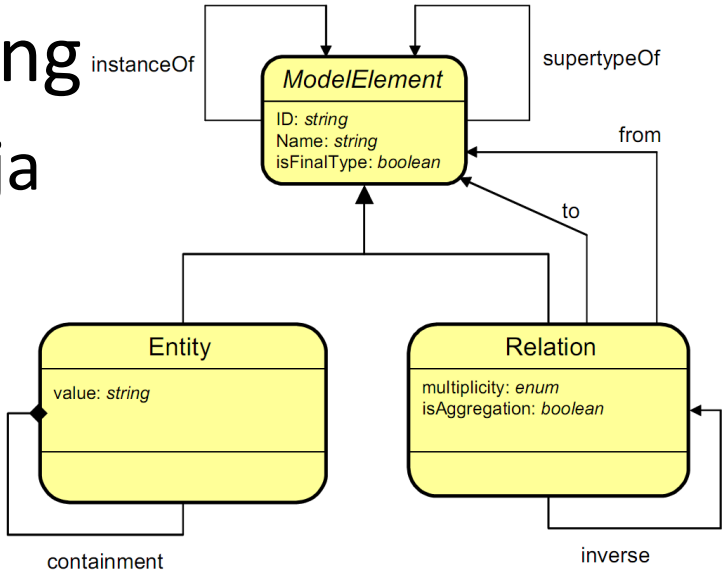
Transformation Engine



Végrehajtó motor:
Nagy modellek lekérdezése és módosítása

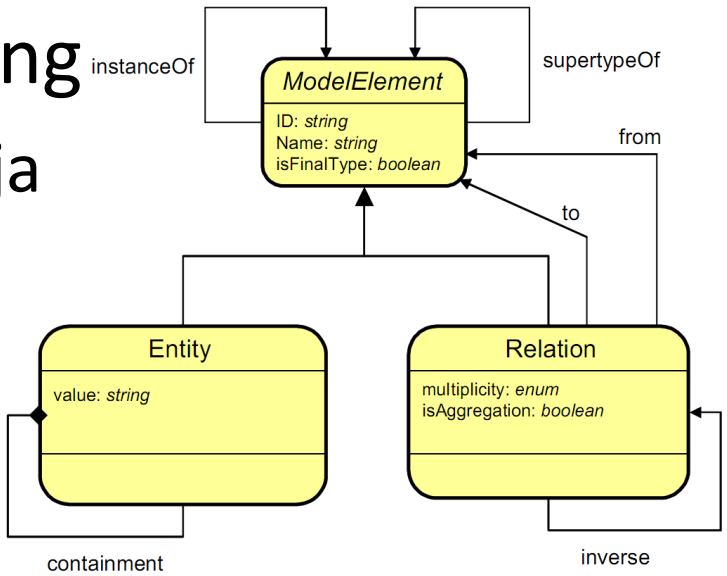
VPM – Belső modellreprezentáció

- Visual and Precise Metamodeling
  - Entitások tartalmazás hierarchiája
  - Összekötő relációk
- Nagyon szabad és általános
  - Attribútum értékek entitások
  - Relációk közötti relációk
  - Többszörös típusok és többszörös öröklődés
  - Többszintű modellezési hierarchia
  - Metamodellek módosítása
  - Dinamikus újratipizálás



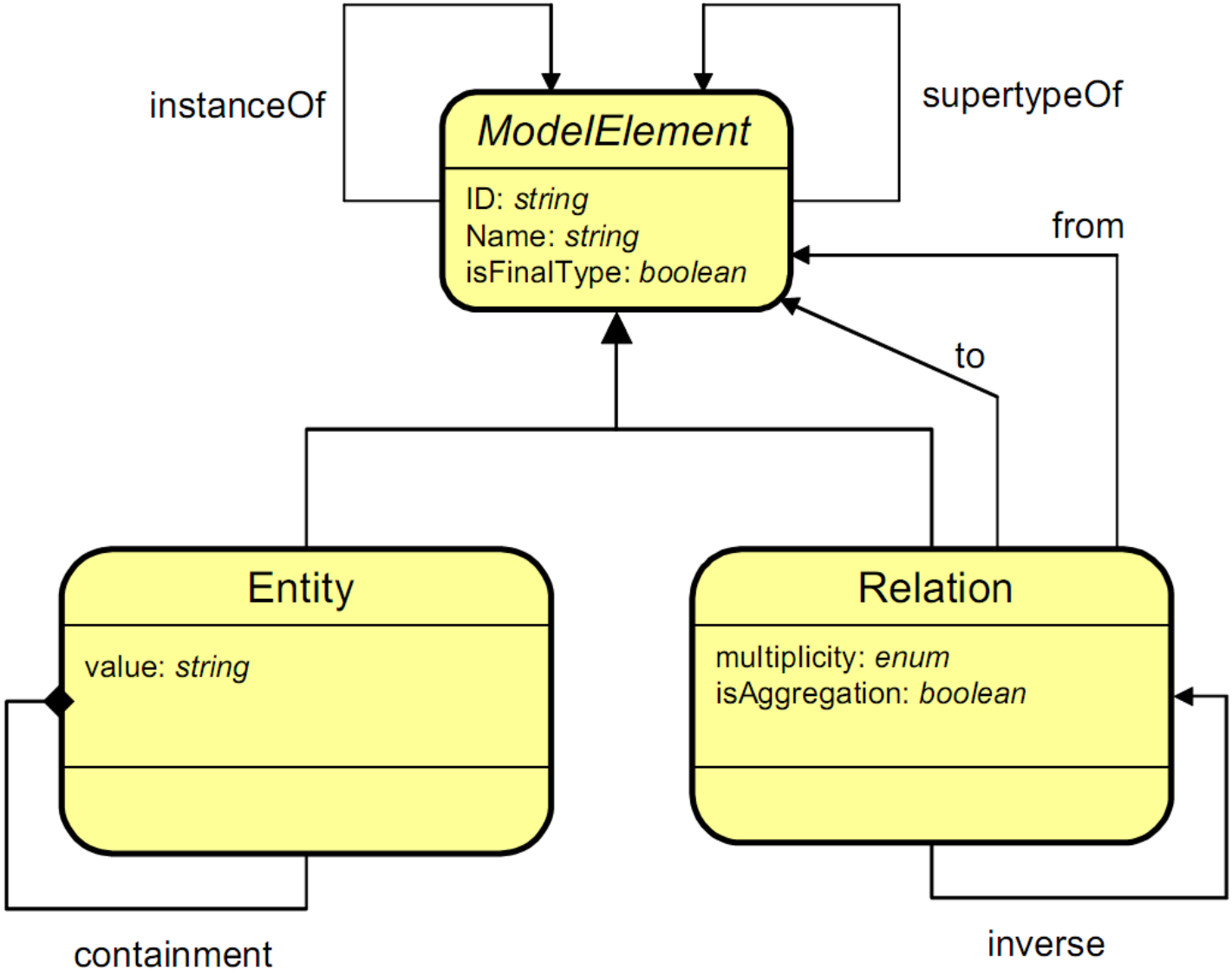
VPM – Belső modellreprezentáció

- Visual and Precise Metamodeling
  - Entitások tartalmazás hierarchiája
  - Összekötő relációk
- Nagyon szabad és általános
  - Attribútum é[illegible]ékek entitások
  - Relációk közötti [illegible]iók
  - Többszörös t[illegible]
  - Többszintű m[illegible]
  - Metamodellek módosítása
  - Dinamikus újratipizálás

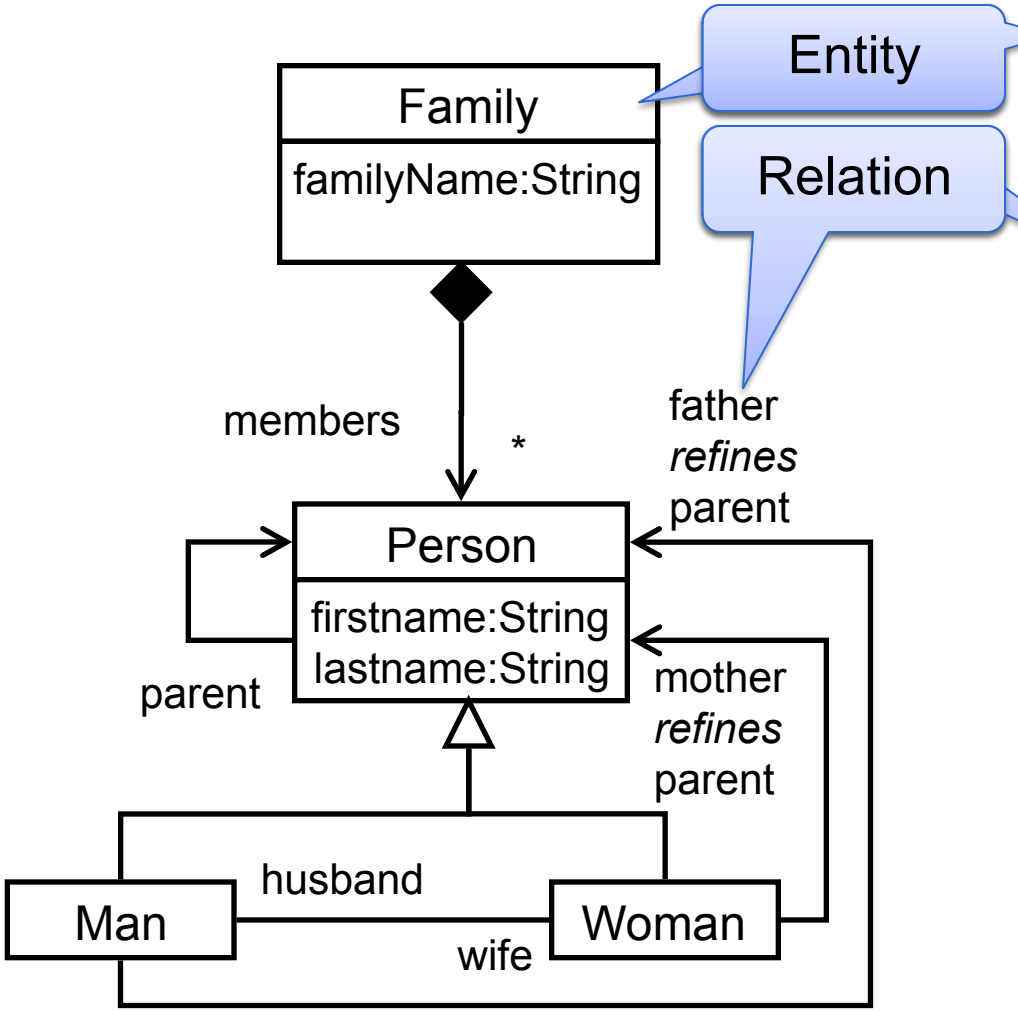


Modellező környezetben problémás, integrációkor hasznos

VPM – Belső modellreprezentáció

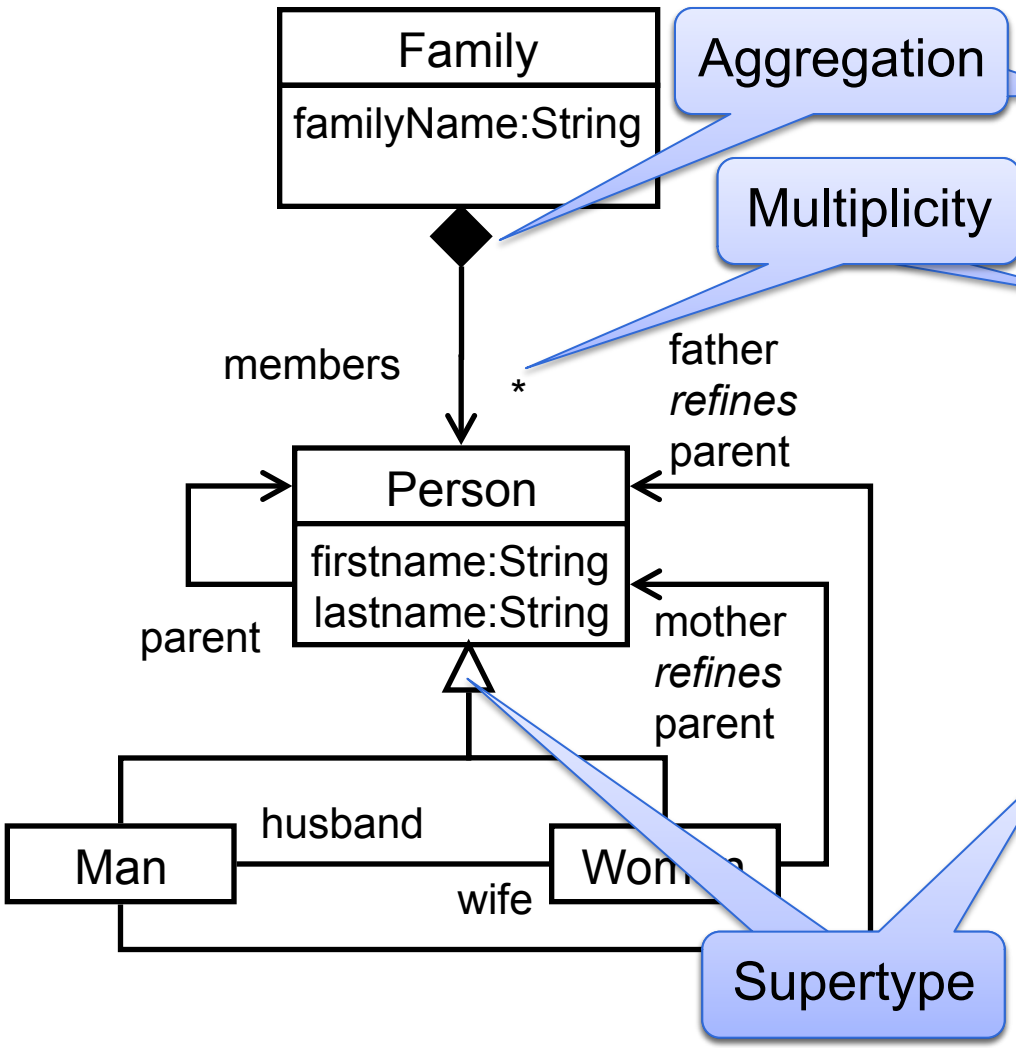


Metamodellezés (VTML)




```
entity(family);
relation(members, family, person);
isAggregation(family.members,
    true);
entity(person) {
    relation(parent,person,person);
    relation(father,person,man);
    multiplicity(father,
    many_to_one);
    supertypeOf(parent,father);
    relation(firstname, person,
    datatypes.String);
}
entity(woman) {
    relation(husband, woman, man);
    multiplicity(husband,
    one_to_one);
}
supertypeOf(person,woman);
entity(man) {
    relation(wife, man, woman);
    inverse(wife, woman.husband);
}
supertypeOf(person,man);
```

Metamodeling in VIATRA2 (VTML)




```
entity(family);
relation(members, family, person);
isAggregation(family.members,
    true);
entity(person) {
    relation(parent,person,person);
    relation(father,person,man);
    multiplicity(father,
    many_to_one);
    supertypeOf(parent,father);
    relation(firstname, person,
    datatypes.String);
}
entity(woman) {
    relation(husband, woman, man);
    multiplicity(husband,
    one_to_one);
}
supertypeOf(person,woman);
entity(man) {
    relation(wife, man, woman);
    inverse(wife, woman.husband);
}
supertypeOf(person,man);
```

Modellek (VTML)

Példánymodellek létrehozás modellező környezetben, vagy…

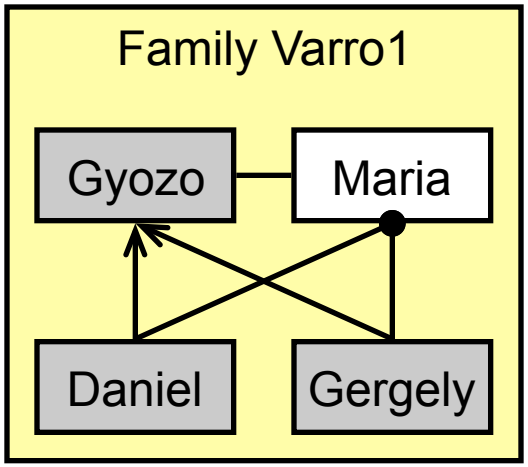
```
namespace people.models;
import people.metamodel;
import datatypes;
 family('Varro1') {
   man('Gyozo') {
        man.wife(wf1, Gyozo, Maria);
   }
   woman('Maria') {
        woman.husband(hb1, Maria, Gyozo);
   }
   man('Daniel');
   entity('Gergely');
   instanceOf(Gergely, man);
   person.father(f1,'Gergely','Gyozo');
   person.mother(m1,'Gergely','Maria');
   family.members(mb1, 'Varro1', 'Gyozo');
   family.members(mb2, 'Varro1', 'Maria');
   family.members(mb3, 'Varro1',
   'Gergely');
   family.members(mb4, 'Varro1',
   'Daniel');
   family.familyname(fn, 'Varro1', str);
   String(str) -> "Varro";
 }
```

Névtér, Imports

Példány megadás

Explicit típusviszony

Érték OO adattípus



man

woman

mother

father

wife/husband

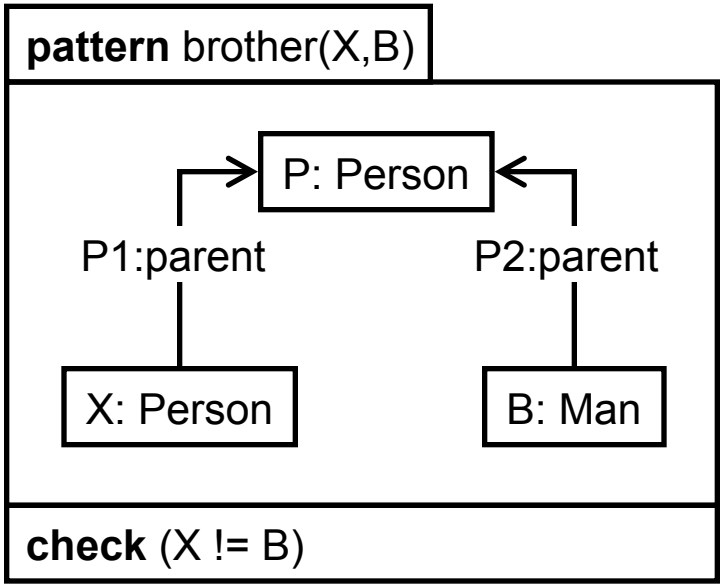
Mi a VIATRA2?

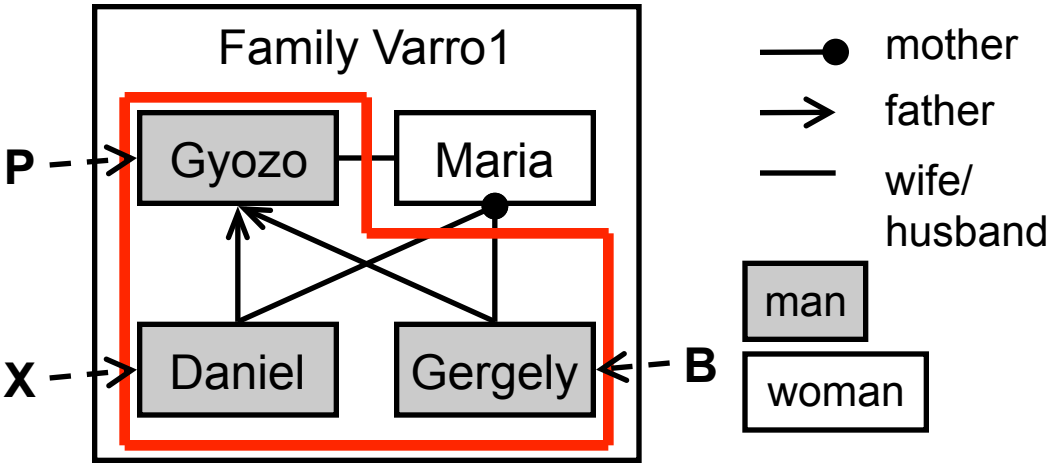
- MT platform
  - Transzformáció végrehajtás
  - Modellreprezentáció
- **Transzformációs nyelv**
  - **Transzformációleírásra szabva**
- Fejlesztőkörnyezet
  - Eclipse technológiákra épül
- Bővíthető keretrendszer
  - Különböző modellreprezentációk kezelése (UML, EMF, etc.)
  - Transzformációs funkciók kiterjesztése

GTASM – Transzformációk megadása

- **Vezérlés** ≈ absztrakt állapotgépek
- **Gráf minták** modell részeinek azonosítására
- **Gráf transzformációs szabályok** modellmanipulációhoz

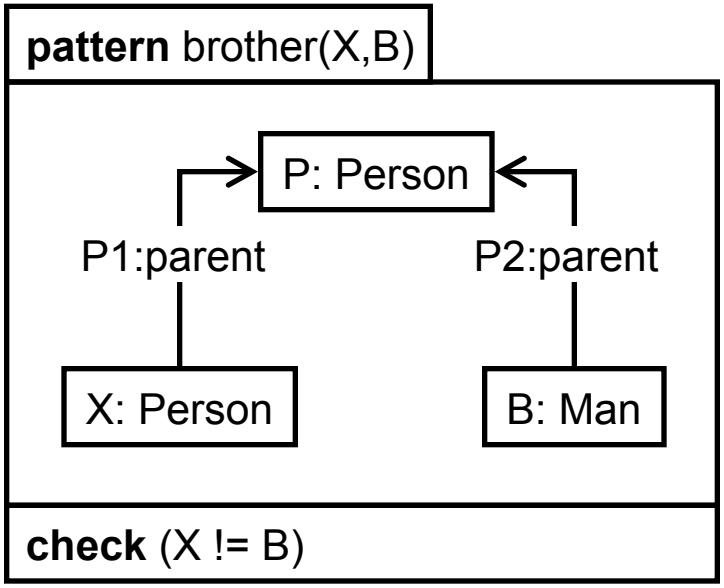
Gráf minták (VTCL)

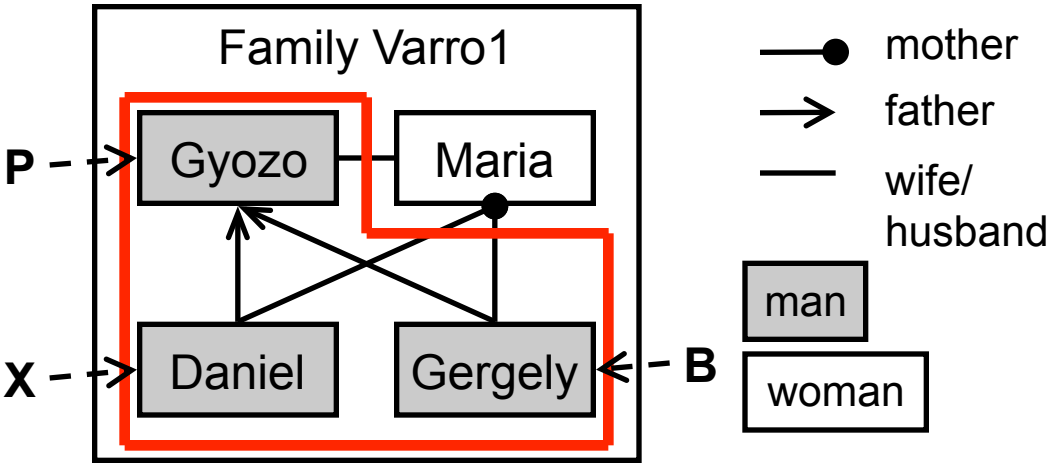


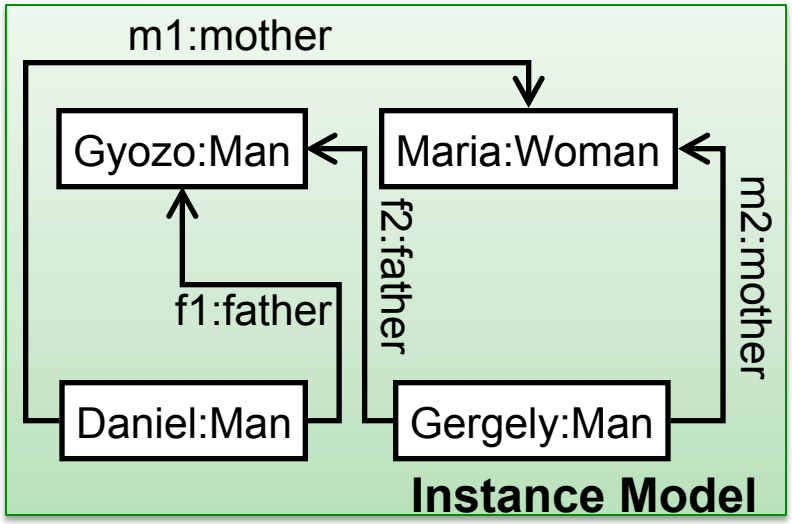


- Gráf minta:
  - Strukturális feltétel, amelyet a modell egy része kielégít
- Gráf minta illesztés:
  - Egy modell (pl. a modelltér része) kielégít egy mintát,
  - ha a minta illeszthető a modell egy részgráfjára

Gráf minták (VTCL)

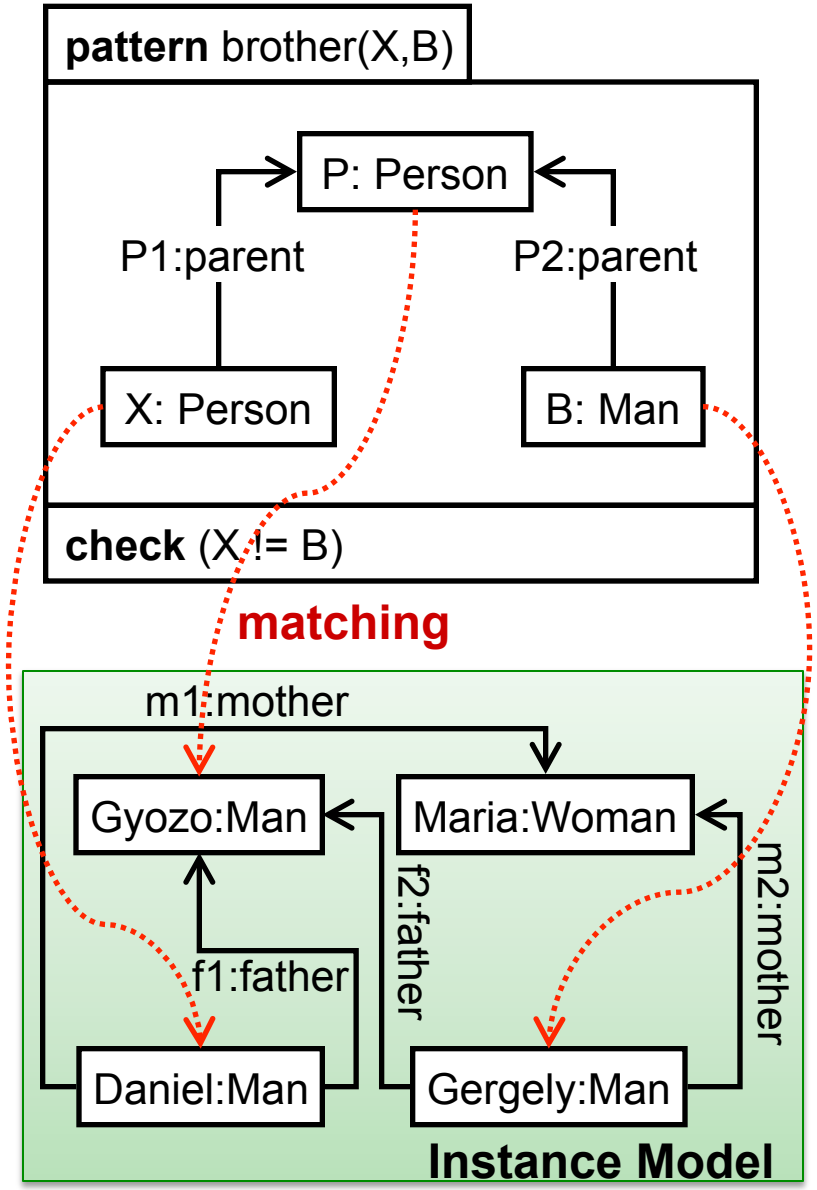


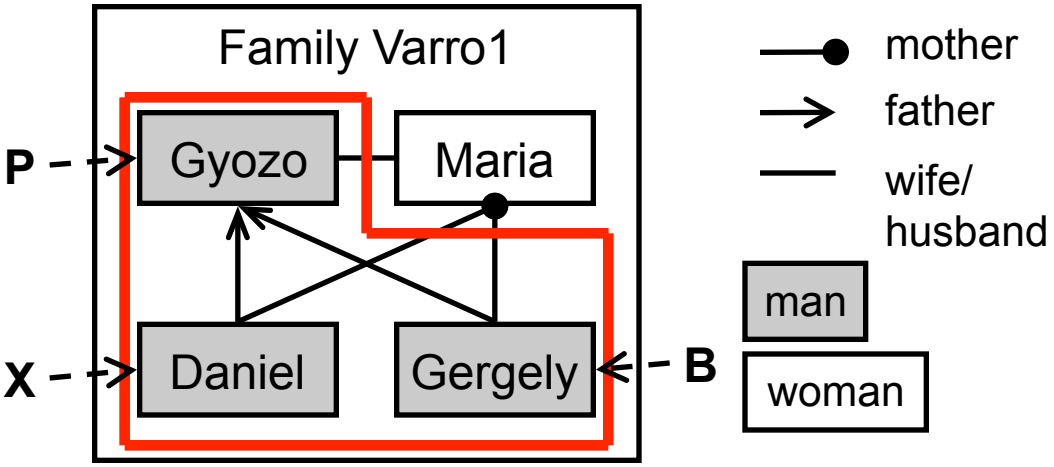




- Gráf minta:
  - Strukturális feltétel, amelyet a modell egy része kielégít
- Gráf minta illesztés:
  - Egy modell (pl. a modelltér része) kielégít egy mintát,
  - ha a minta illeszthető a modell egy részgráfjára

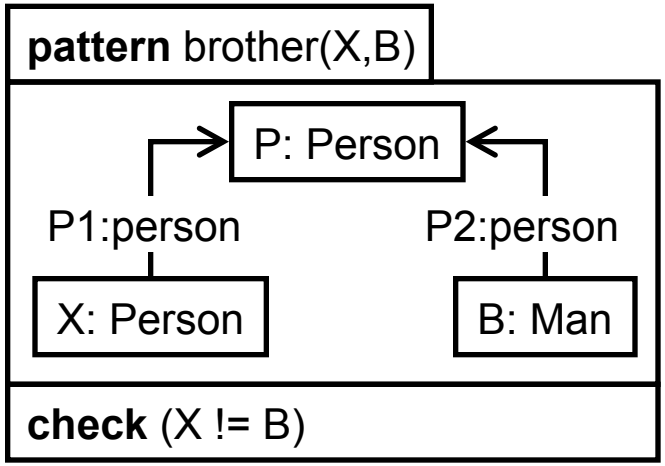
Gráf minták (VTCL)





- Gráf minta:
  - Strukturális feltétel, amelyet a modell egy része kielégít
- Gráf minta illesztés:
  - Egy modell (pl. a modelltér része) kielégít egy mintát,
  - ha a minta illeszthető a modell egy részgráfjára

Gráf minták (VTCL)

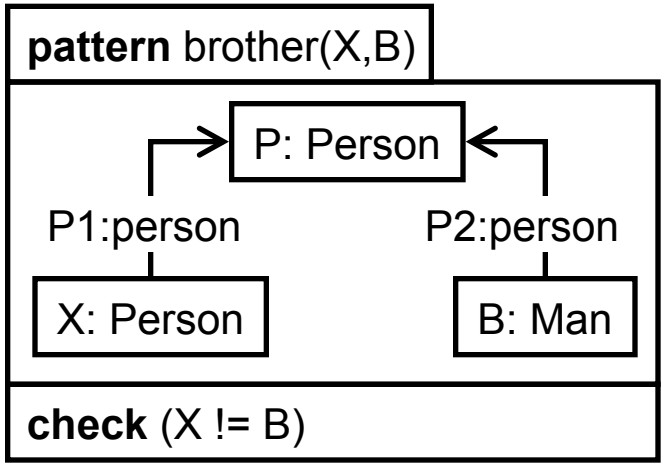



```
// B is a brother of X
 pattern brother(X, B) =
 {
    person(X);
    person.parent(P1,X,P);
    person(P);
    person.parent(P2,B,P);
    man(B);
    check (X != B)
 }
```

Injectivity

```
// B is a brother of X
 shareable pattern
    brother(X, B) =
 {
    person(X);
    person.parent(P1,X,P);
    person(P);
    person.parent(P2,B,P);
    man(B);
}
```

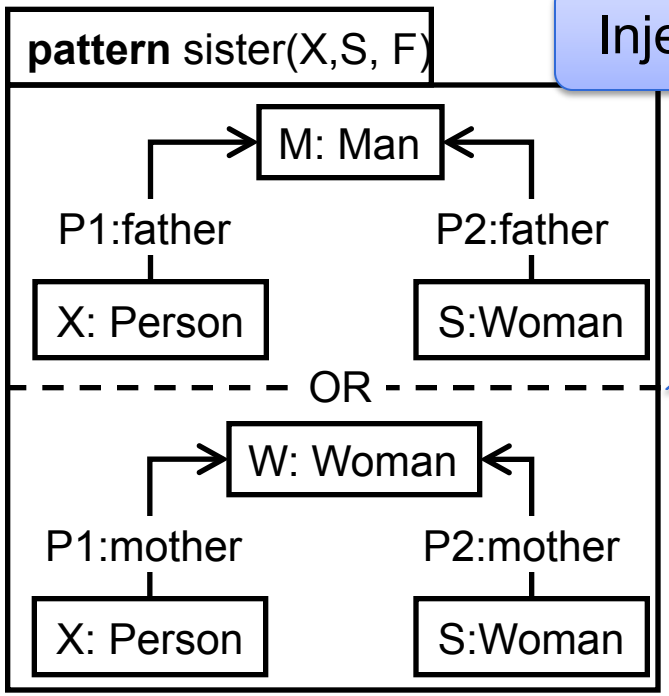
Gráf minták (VTCL)




```
// B is a brother of X
 pattern brother(X, B) =
 {
    person(X);
    person.parent(P1,X,P);
    person(P);
    person.parent(P2,B,P);
    man(B);
    check (X != B)
 }
```

```
// B is a brother of X
 shareable pattern
    brother(X, B) =
 {
    person(X);
    person.parent(P1,X,P);
    person(P);
    person.parent(P2,B,P);
    man(B);
}
```

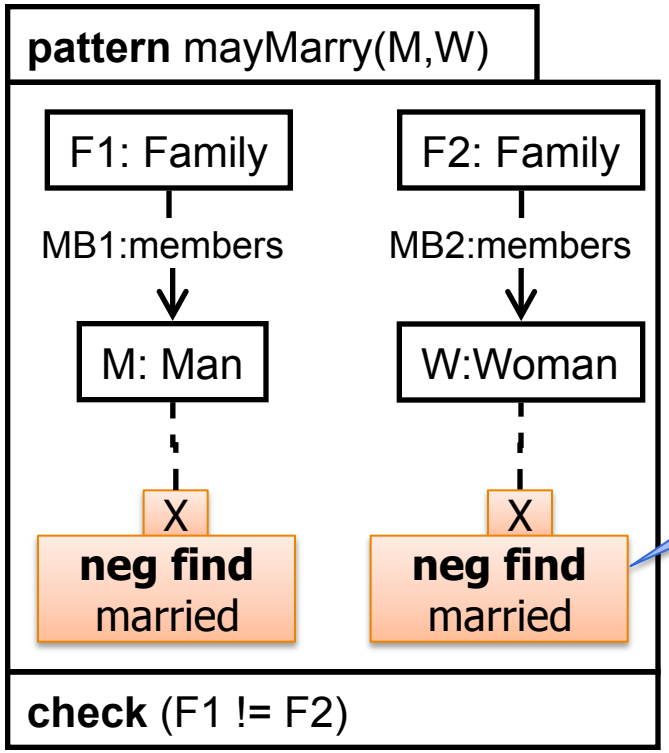
Injectivity

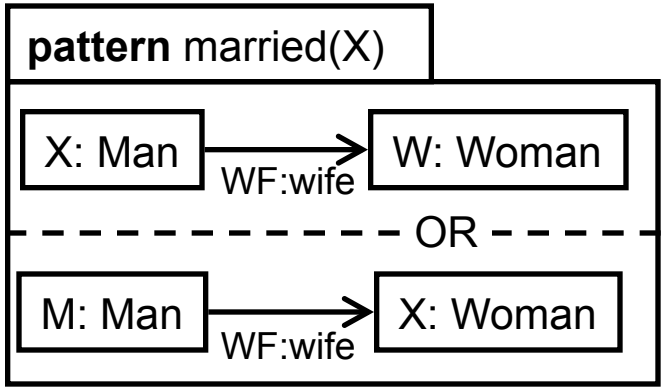


OR pattern

```
// S is a sister of X
 pattern sister(X, S, F) = {
    person(X) in F;
    person.father(P1, X, M);
    man(M) in F;
    person.father(P2, S, M);
    woman(S) in F;
}   or {
    person(X) in F;
    person.mother(P1, X, W);
    woman(W) in F;
    person.mother(P2, S, W);
    woman(S) in W;
}
```

Negatív minták (VTCL)



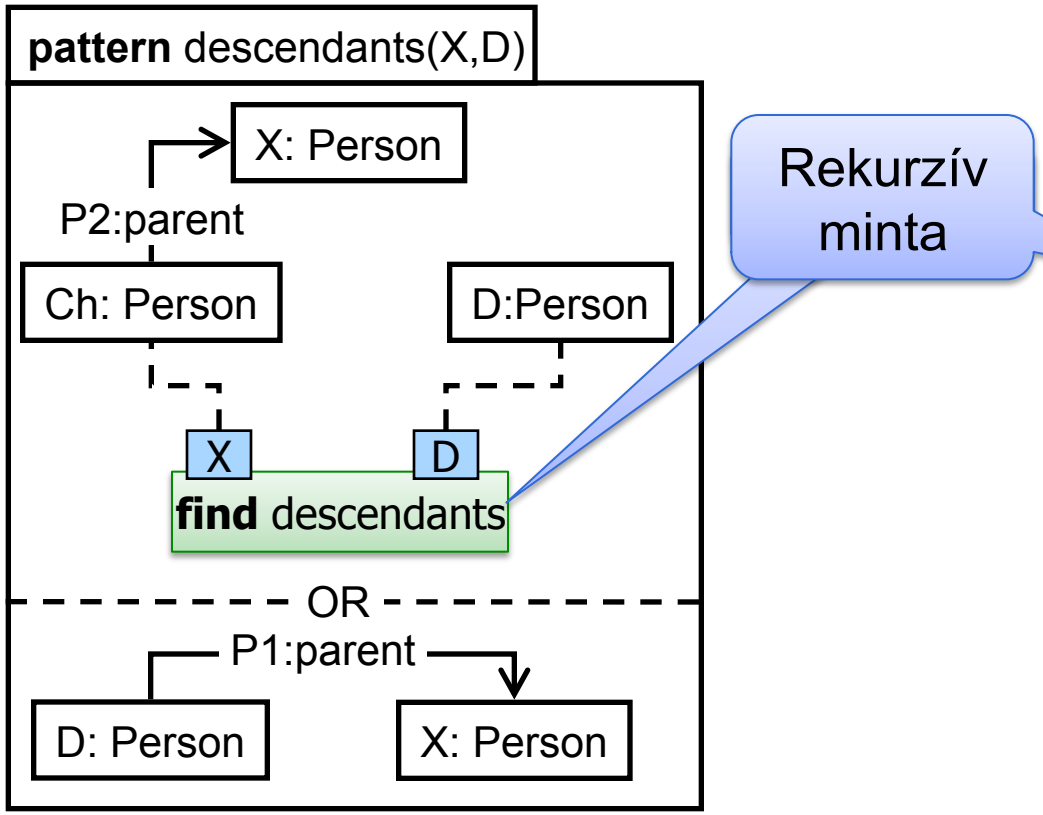


Negatív minta

```
shareable
pattern mayMarry(X, D) = {
    man(M);
    family(F1);
    family.members(MB1, F1, M);
    woman(W);
    family(F2);
    family.members(MB2, F2, W);
    neg find married(M);
    neg find married(W);
    F1 =/= F2;
}

pattern married(X) =
{
    man(X);
    man.wife(WF, X, W);
    woman(W);
  } or {
    woman(X);
    man.wife(WF, M, X);
    man(M);
  }
```

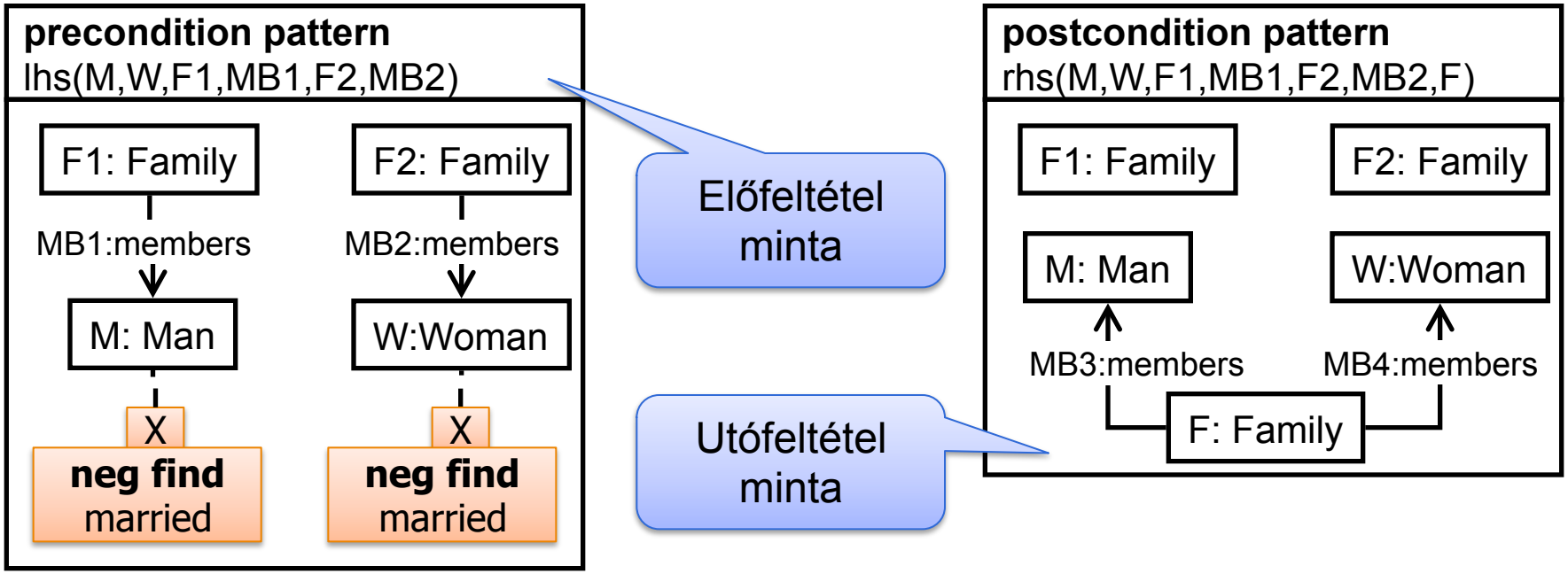
Rekurzív minták (VTCL)



Rekurzív minta

```
pattern descendants(X, D) =
    {
    person(X);
    person.parent(P2, Ch, X);
    person(Ch);
    find descendants(Ch, D)
    person(D);
} or {
    person(X);
    person.parent(P1, D, X);
    person(D);
 }
```

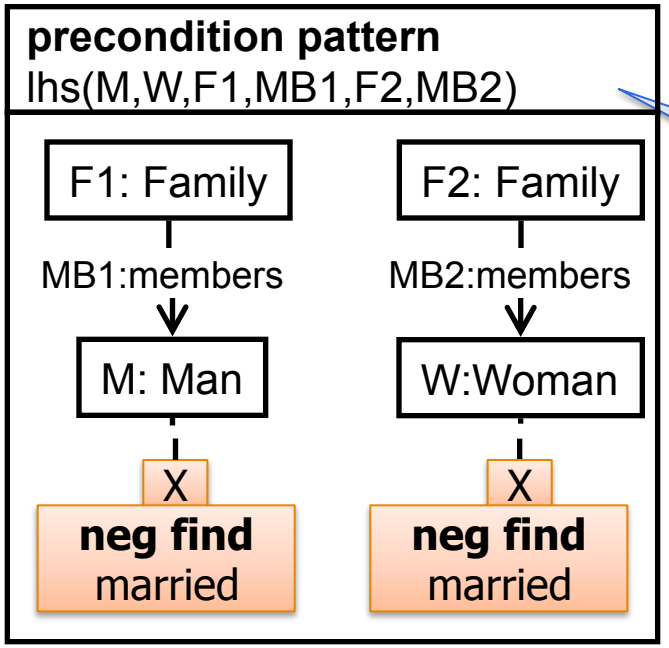
Gráf transzformációs szabályok (VTCL)




```
gtrule marry(in M, in W, out F) =
precondition pattern lhs(M,W,F1,MB1,F2,MB2) =
    {
    family(F1);
    family.members(MB1, F1, M);
    man(M);
    family(F2);
    family.members(MB2, F2, W);
    woman(W);
    neg find married(M);
    neg find married(W);
}
```

```
postcondition pattern
rhs(M,W,F1,MB1,F2,MB2,F) = {
    family(F1);
    man(M);
    family(F2);
    woman(W);
    family(F);
    family.members(MB3, F, M);
    family.members(MB4, F, W);
}
```

Gráf transzformációs szabályok (VTCL)



Előfeltétel minta

Akció rész

```
gtrule marry(in M, in W, out F) =
precondition pattern lhs(M,W,F1,MB1,F2,MB2) =
    {
    family(F1);
    family.members(MB1, F1, M);
    man(M);
    family(F2);
    family.members(MB2, F2, W);
    woman(W);
    neg find married(M);
    neg find married(W);
}
```

```
action {
  delete(MB1);
  delete(MB2);
  new(family(F));
  new(family.members(MB3, F, M));
  new(family.members(MB4, F, W));
  new(man.wife(WF1, M, W));
}
```

Absztrakt állapotgépek

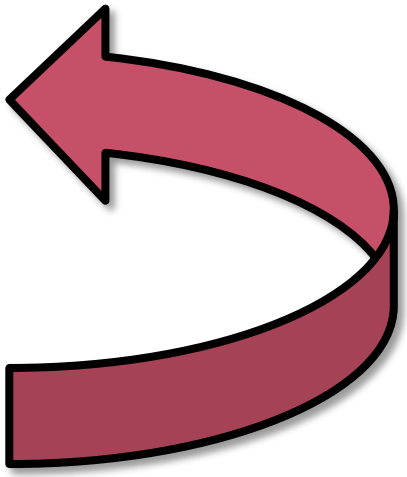
- ASM: magas szintű programozási nyelv
  - Vezérlési szerkezet
  - GT-vel integrálva
- Példák
  - **update** *location* = *term*;
  - **parallel** {…} / **seq** {…}
  - **let** *var* = *term* **in** *rule*;
  - **if** (*formula*) *rule1*; **else** *rule2*;
  - **iterate** *rule;*
  - **forall/choose** *variables* **with find** *pattern* **do** *rule*;
  - **forall/choose** *variables* **with apply** *gtrule* **do** *rule*;

```
forall X below people.models,
    B below people.models
    with find brother(X, B) do seq {
      print(name(X) + "->" + name(B));
    }
```

```
let X = people.models.Varro1.Daniel,
    Y = people.models.Gyapay1.Szilvia,
    F = undef, F2 = undef in
choose F below people.models
with apply marry(X, Y, F) do seq {
    rename(F, "Varro2");
    move(F, people.models);
    iterate
      choose M below people.models,
        W below people.models
          with apply marry(M, W, F2)
            do move(F2, people.models);
}
```

Transzformációk életciklusa

- Transzformációk is programok
  - (Életciklus hasonló szoftverhez)
  - Specifikáció
  - Fejlesztés
    - Tesztelés, eszköztámogatás
  - Felhasználás
    - Szoftver fejlesztés során
  - Javítások



VIATRA, MINT ECLIPSE BŐVÍTMÉNY

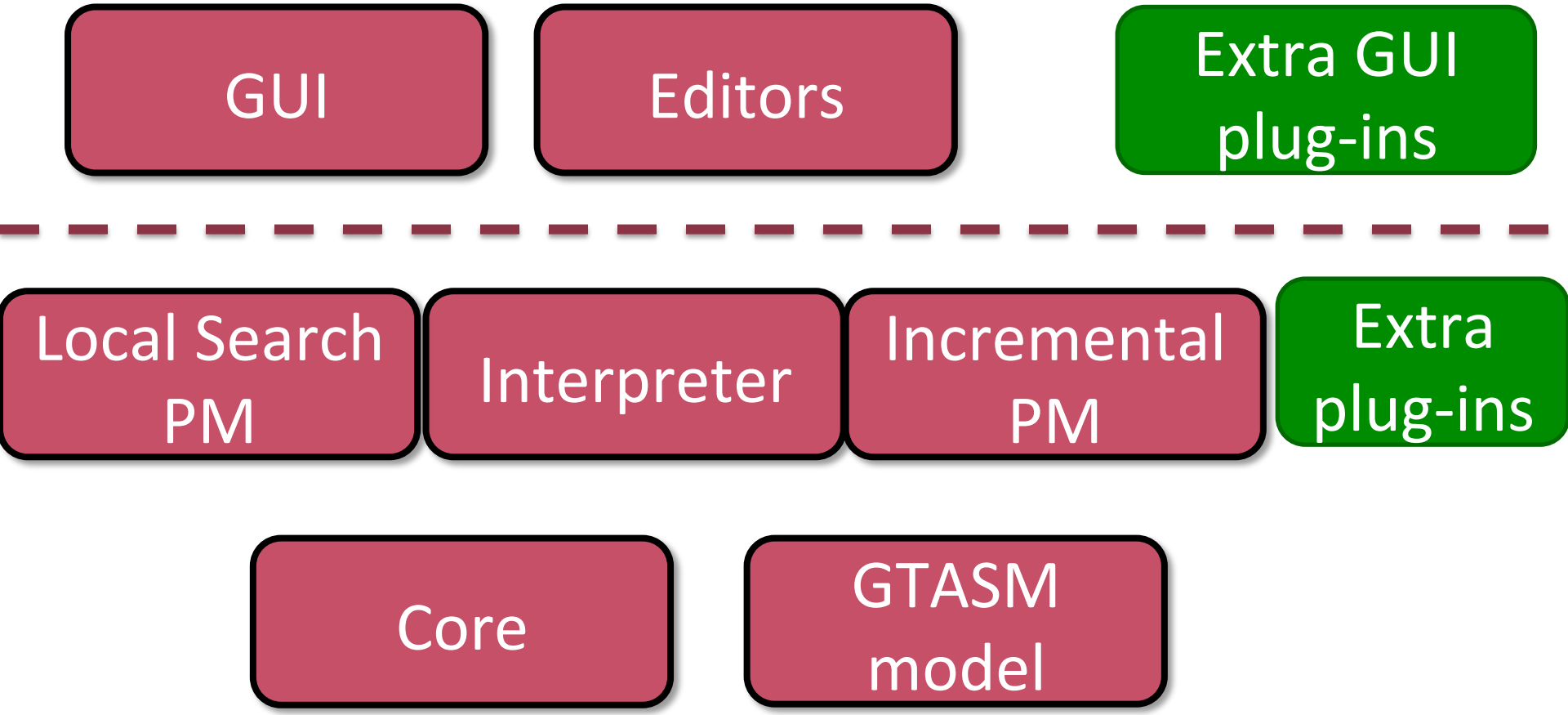
VIATRA fejlesztés

- Sok fejlesztő
- 10 év
- Core
  - 19 projekt
  - 140 kLOC (ebből 55kLOC generált) kód
- Sok kiegészítő projekt

Fejlesztők

- Fejlesztők
  - 5 csoporttag
    - Főleg a rendszer magja
  - Hallgatók ☺
    - Tipikusan kiegészítő plug-inek
- Felhasználás
  - Projekteken belül
    - Partnerek

Architektúra



KUTATÁSI IRÁNYOK

Kutatási irányok

- Hatékony mintaillesztés
  - Inkrementális mintaillesztő
- Modell evolúció
  - Élő transzformációk
- Tervek optimalizálása
  - CSP(M) megoldó MT alapokon
- Domain-specifikus modellezés
  - Jobb eszköztámogatás beépített MT eszközzel

Kutatási irányok

- **Hatékony mintaillesztés**
  - **Inkrementális mintaillesztő**
- Modell evolúció
  - Élő transzformációk
- Tervek optimalizálása
  - CSP(M) megoldó MT alapokon
- Domain-specifikus modellezés
  - Jobb eszköztámogatás beépített MT eszközzel

Inkrementális mintaillesztő

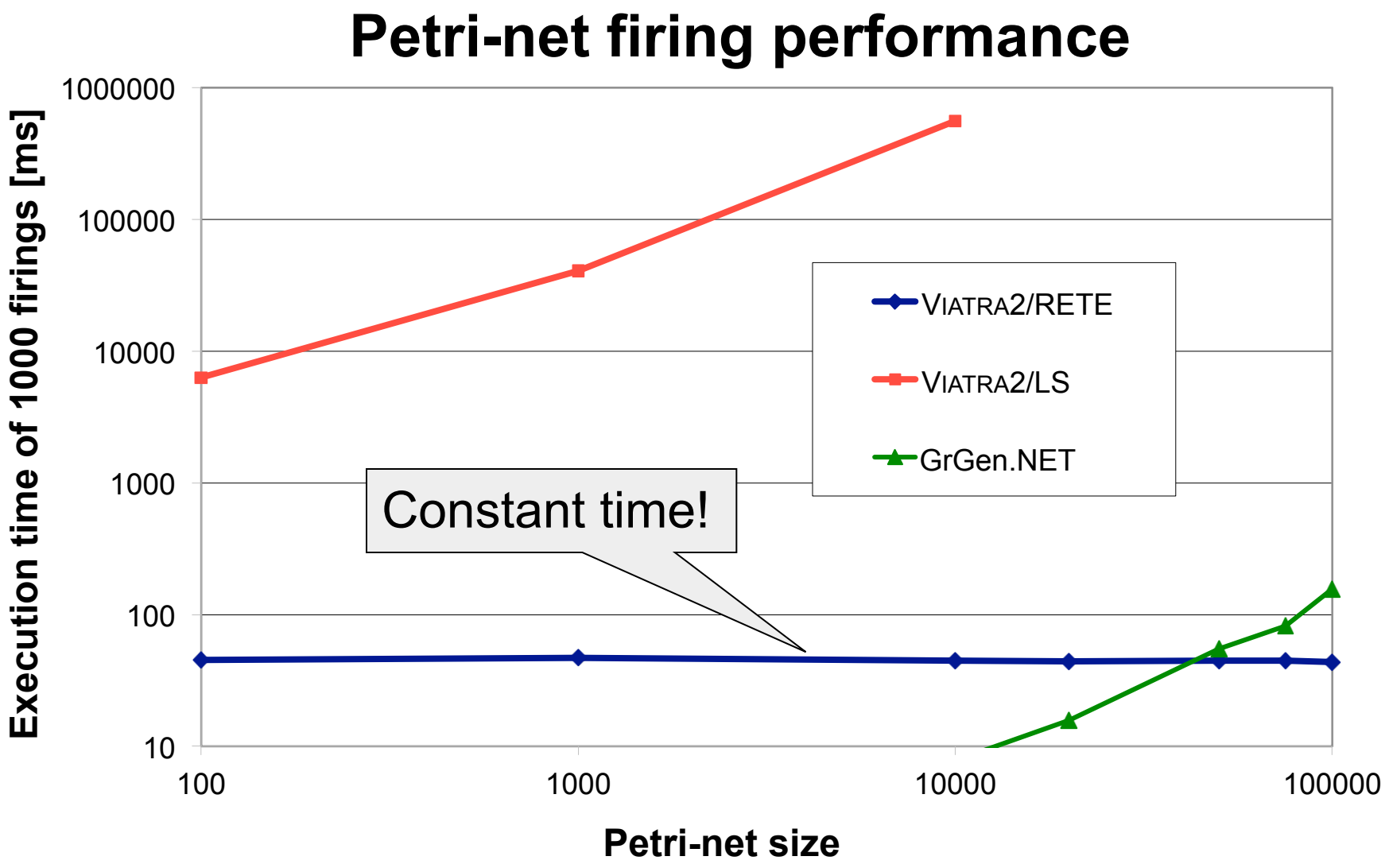
- Cél
  - Találat halmazok tárolása
  - Inkrementális frissítés
  - Gyors válasz

```
@incremental
pattern brother(X, B) =
 {
        person(X);
        person.parent(P1, X, P);
        person(P);
        person.parent(P2, B, P);
        man(B);
}
```

- Jó teljesítményt mértünk, **ha**:
  - Gyakori mintaillesztés
  - Kis változások
- Benchmarkokon kijön
- RETE algoritmus használata

```
@localsearch
pattern brother(X, B) =
 {
        person(X);
        person.parent(P1, X, P);
        person(P);
        person.parent(P2, B, P);
        man(B);
}
```

Benchmark: Petri Net Simulation



Kutatási irányok

- Hatékony mintaillesztés
  - Inkrementális mintaillesztő
- **Modell evolúció**
  - **Élő transzformációk**
- Tervek optimalizálása
  - CSP(M) megoldó MT alapokon
- Domain-specifikus modellezés
  - Jobb eszköztámogatás beépített MT eszközzel

Frissülő modellek transzformációja

Frissülő modellek transzformációja

- Kihívás: **frissülő** modellek
  - Állandó változások/módosítások
  - Hatékonyság? Nyomon követés?
- Megoldás: **inkrementalitás**
  - Változások azonosítása, és
  - leképzése és alkalmazása a célmodellre
- **Live Transformation**
  - Cél folyton szinkronban van a forrással
  - Forrás változásaira reagálunk: **események**
- Szükséges: **esemény-vezérelt transzformációk**

Event-driven live transformations

Event-driven live transformations

- Eseményvezérelt transzformációs szabályok
  - Formalizmus: MIKOR és MIT csinálunk/reagálunk
- Mi megközelítésünk: események megadhatóak, mint egy minta illeszkedéshalmazának változásai.
  - ~általánosítás
  - inkrementális mintaillesztő ezt adja!
- **Gráf triggerek**
  - Figyel a megjelenő/eltűnő mintákra
  - Magas szintű, deklaratív specifikáció
  - Tetszőleges (deklaratív/imperatív) reakció

Eseményvezérelt transzformációk: VIATRA

```
machine HelloTriggerWorld {
  @Trigger(priority='100',
   mode='always',
   sensitivity='rise',
   startup='passive',
   execution='iterate')
  gtrule myTrig(inout E) = {
   // This pattern triggers the
   // action sequence.
    precondition pattern lhs(E) = {
      entity(E);
    }
    // The action sequence.
    action {
      println( „Triggered by " + E );
    }
  }
  //Main rule, the startTrigger()
  // method invokes the engine.
  rule main() = seq {
    println(startTrigger("myTrig"));
  }
}
```

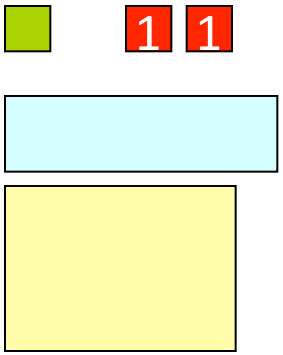
Annotations :

- **Priority** (integer):
  - If two or more triggers are ready to run at the same time, the trigger with the highest priority will run first.
  - Of the triggers with the same priority one is selected non-deterministically.
- **Mode** (once | <u>always</u>): default value underlined
  - **Always**: the trigger runs in the background until it is stopped by the user.
  - **Once**: the trigger will only run once.
- **Sensitivity** (<u>rise</u> | fall):
  - **Rise**: the action is executed with every new match of the trigger's condition pattern,
  - **Fall**: the action is executed with every disappearance of a previously existing pattern match.
- **Startup** (active | passive):
  - **Active**: trigger action is executed with all pattern matches that are present in the model at the trigger's startup.
  - **Passive**: all current matches that are present in the model space when the trigger is added to the framework, are discarded. The default setting of a trigger is active startup.
- **Execution** (forall | iterate):
  - **Forall**: the action will be executed with all pattern substitutions in a pseudo-parallel way.
  - **Iterate**: the pattern substitutions are executed in separate steps as long as possible.
- **AutoStart** (true | false):
  - **true**: the trigger will start automatically after loading
  - **false**: the trigger will not start automatically after loading

Kutatási irányok

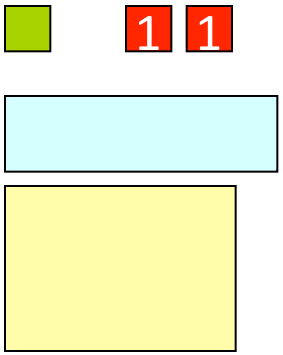
- Hatékony mintaillesztés
  - Inkrementális mintaillesztő
- Modell evolúció
  - Élő transzformációk
- **Tervek optimalizálása**
  - **CSP(M) megoldó MT alapokon**
- Domain-specifikus modellezés
  - Jobb eszköztámogatás beépített MT eszközzel

Tervoptimalizáció MT eszközzel



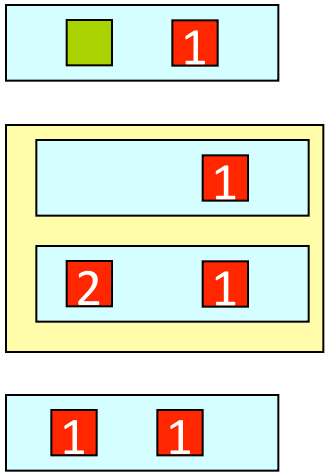
Tervoptimalizáció MT eszközzel

- Integrated modular avionics (IMA) system
- Composed of
  - **Jobs**; Simple Job, Critical Job
  - **Partitions**; executing jobs
  - **Modules**; hosting partitions



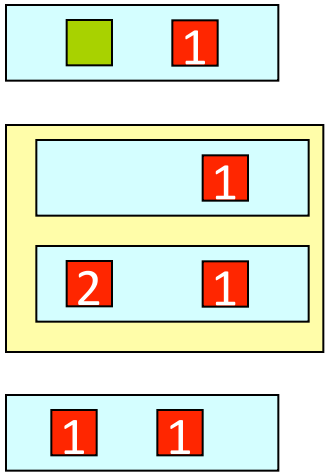
- Task
  - Allocate predefined Jobs on predefined Partitions using minimal number of Modules

Design Optimization with MT



Design Optimization with MT

- Constraints
  - Partition → one criticality level
  - Critical job's redundant instances → different Partitions and Modules
  - Free memory of partition ≥ 0
    - Attribute constraint



- Constraint satisfaction over models: CSP(M)
  - Problem defined by graph transformation rules
  - Solver based on VIATRA2
    - uses incremental pattern matching

Kutatási irányok

- Hatékony mintaillesztés
  - Inkrementális mintaillesztő
- Modell evolúció
  - Élő transzformációk
- Tervek optimalizálása
  - CSP(M) megoldó MT alapokon
- **Domain-specifikus modellezés**
  - **Jobb eszköztámogatás beépített MT eszközzel**

DSM és transzformációk

DSM és transzformációk

- Ötlet: MT a domain-specifikus szerkesztőkben (DSM)
- Integrációs ötletek
  - Modellek közötti szinkronizáció
    - Rugalmas absztrakt-konkrét szintax közti elkülönítés
  - Azonnali hibajelzések
  - Modellanimáció és dinamikus szintaxisváltoztatás
  - Kódgenerálás
    - Modell-kód szinkronizáció

ÖSSZEFOGLALÁS

VIATRA2 mint transzformációs eszköz

- Eszköz integráció MT-vel lehetséges
  - Szolgáltatás-orientált rendszerek
  - Megbízható beágyazott rendszerek
- MT támogatás
  - Fejlesztés, végrehajtás, integráció
- VIATRA2 környezet
  - Sikeres alkalmazások több környezetben
  - Több partner használja (akadémiai/ipari)
  - Kutatási és oktatási eszköz