Formalising Metamodel Evolution based on Category Theory

Florian Mantz[1], Alessandro Rossini[2], Gabriele Taentzer[3], Yngve Lamo[1], Uwe Wolter[2]

[1]Department of Computer Engineering, Bergen University College, Norway

[2]Department of Informatics, University of Bergen, Norway

[3]Philipps-Universität Marburg, Germany

27 October 2011
Nordic Workshop on Programming Theory, Västerås, Sweden

HØGSKOLEN I BERGEN

Model-driven engineering

Model-driven engineering (MDE)

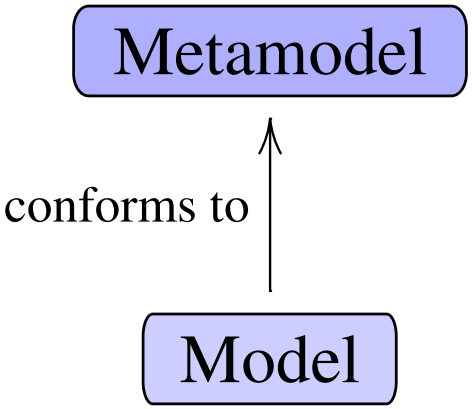
- (Diagrammatic) Models
  - reason at a higher level of abstraction
- Metamodels
  - definition of modelling languages
- Model transformation
  - automatic generation of systems

Challenges in MDE

- Complex evolution of models and metamodels
  - Need for techniques and tools to support this evolution

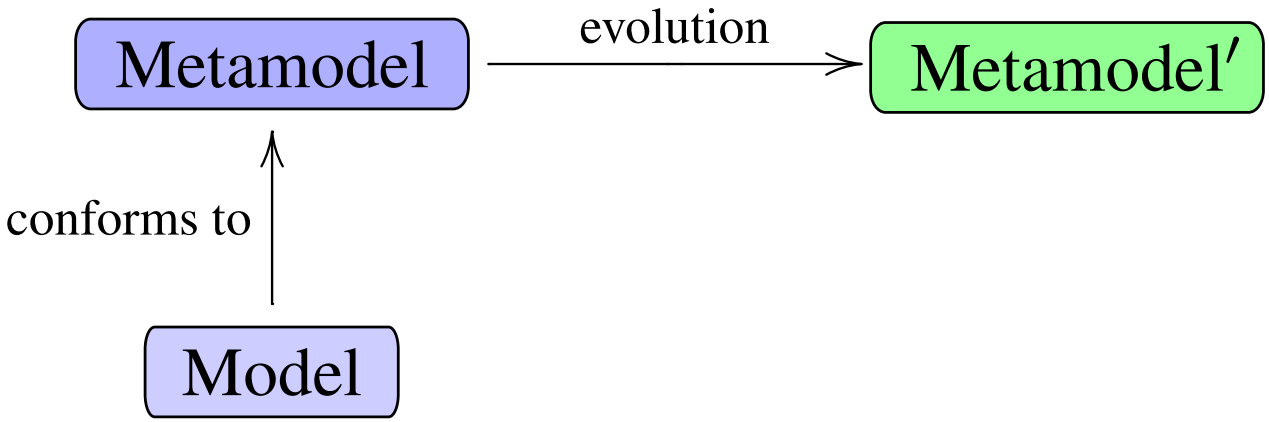
Metamodel evolution

Metamodel evolution



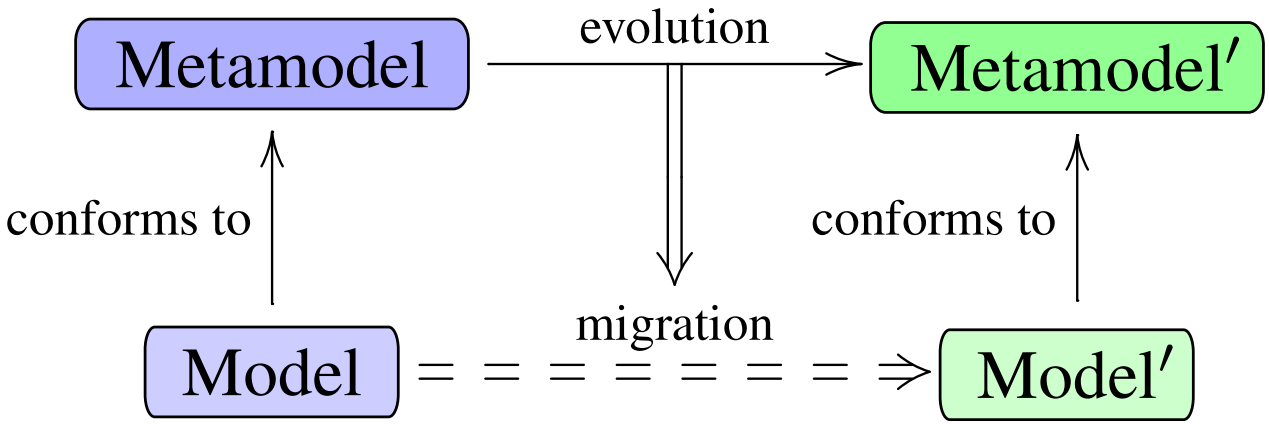
- Metamodels describe modelling languages
- Models conform to the metamodel

Metamodel evolution



- Metamodels evolve

Metamodel evolution



- Models have to be migrated
- Model migration should reflect metamodel evolution

Challenges in metamodel evolution

- Description of metamodel evolution
- Migration of dependent artefacts
  - Models
  - Model transformations
  - Documentation
  - ...
- Lack of formalisation

Diagram Predicate Framework

Diagram Predicate Framework (DPF)

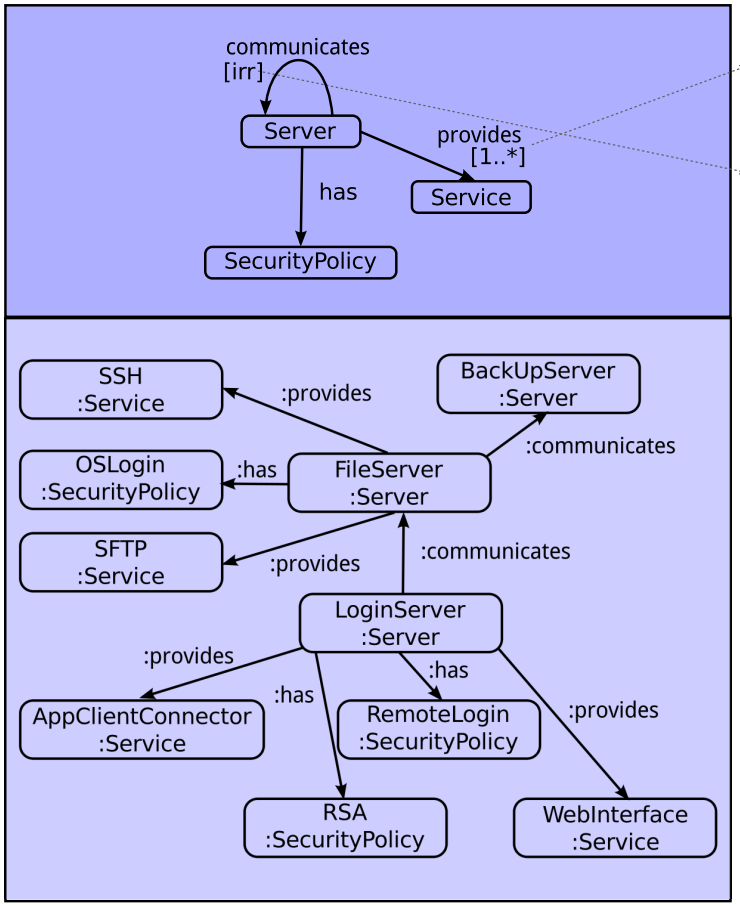
- Formal diagrammatic specification framework founded on category theory and graph transformation
  - Diagrammatic modelling
  - Metamodelling
  - Model transformation
  - Model versioning
  - Deep metamodelling

Metamodelling in DPF

- Models
  - Graph + set of atomic constraints
- Metamodels
  - Graph + set of atomic constraints
- Model-Metamodel Relation
  - Morphisms

Running example

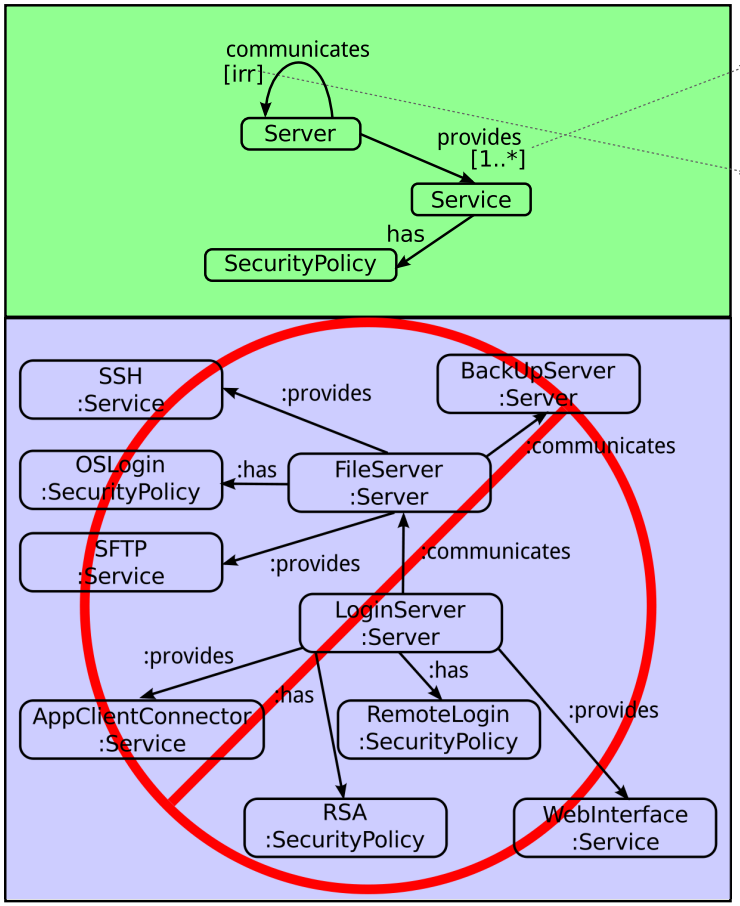
Servers and services



p	$\alpha^{\Sigma_2}(p)$	Proposed vis.	Intended semantics
`[mult(m,n)]`	$1 \xrightarrow{a} 2$	X $\xrightarrow[\text{[m..n]}]{\text{f}}$ Y	$\forall x \in X : m \leq \lvert f(x) \rvert \leq n$, with $0 \leq m \leq n$ and $n \geq 1$
`[irreflexive]`	$1 \circlearrowleft f$	[irr] X $\circlearrowleft$ f	$\forall x \in X : x \quad f(x)$

- Simple network metamodel and conforming model

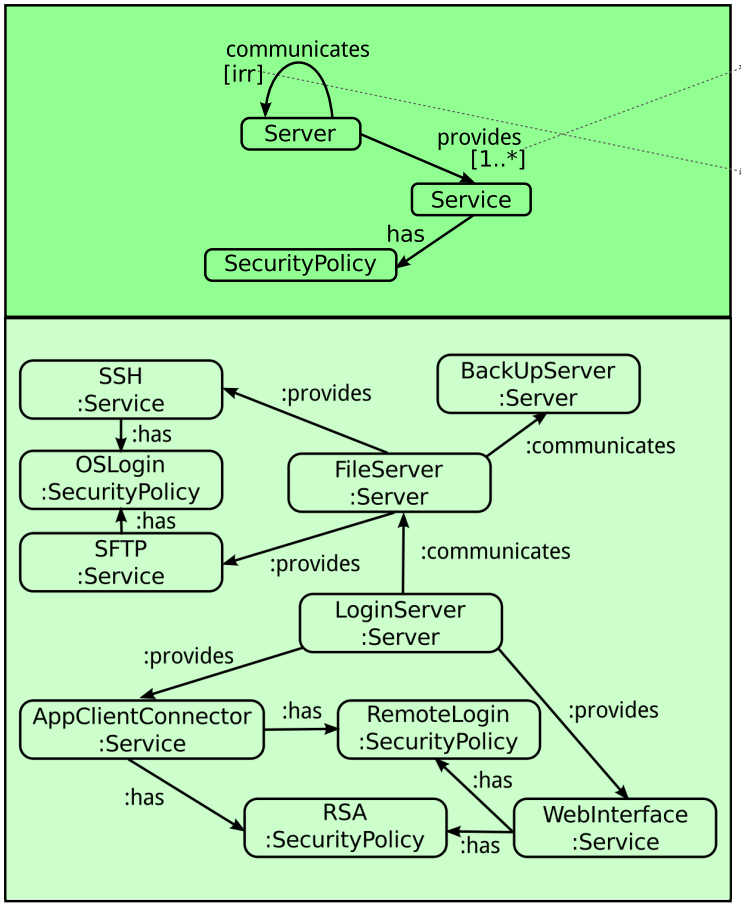
Servers and services



p	$\alpha^{\Sigma_2}(p)$	Proposed vis.	Intended semantics
`[mult(m,n)]`	$1 \xrightarrow{a} 2$	$\boxed{X} \xrightarrow[\text{[m..n]}]{f} \boxed{Y}$	$\forall x \in X : m \leq \lvert f(x) \rvert \leq n$, with $0 \leq m \leq n$ and $n \geq 1$
`[irreflexive]`	$1 \circlearrowleft f$	[irr] $\boxed{X} \circlearrowleft$ f	$\forall x \in X : x \quad f(x)$

- **SecurityPolice** is moved from **Server** to **Service**
- Model does not conform any more to its metamodel

Servers and services



p	$\alpha^{\Sigma_2}(p)$	Proposed vis.	Intended semantics
`[mult(m,n)]`	$1 \xrightarrow{a} 2$	X $\xrightarrow[\text{[m..n]}]{\text{f}}$ Y	$\forall x \in X : m \leq \lvert f(x) \rvert \leq n$, with $0 \leq m \leq n$ and $n \geq 1$
`[irreflexive]`	$1 \circlearrowleft f$	[irr] X $\circlearrowleft$ f	$\forall x \in X : x \quad f(x)$

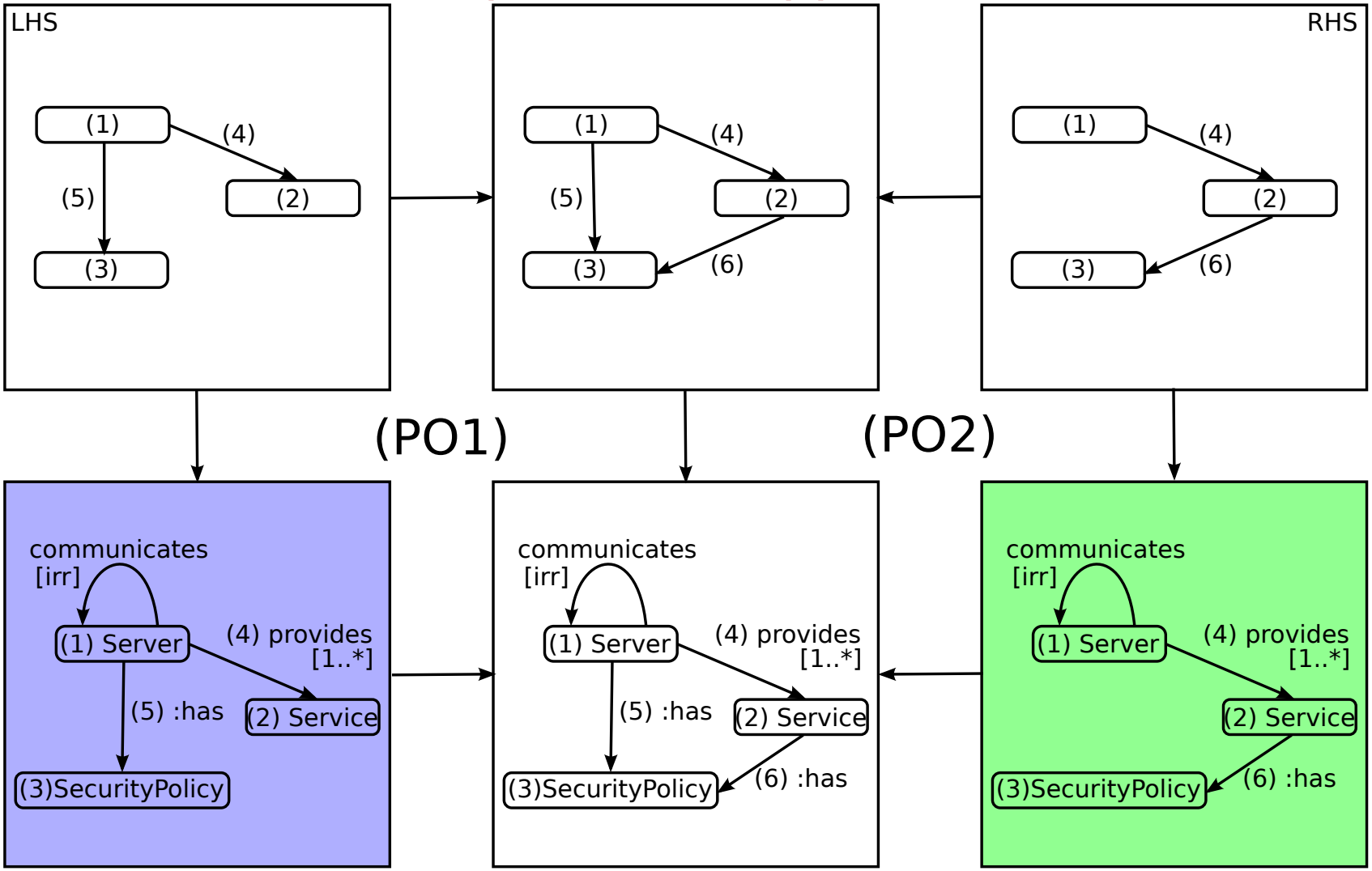
- Model has to be migrated

Formal approach

Idea

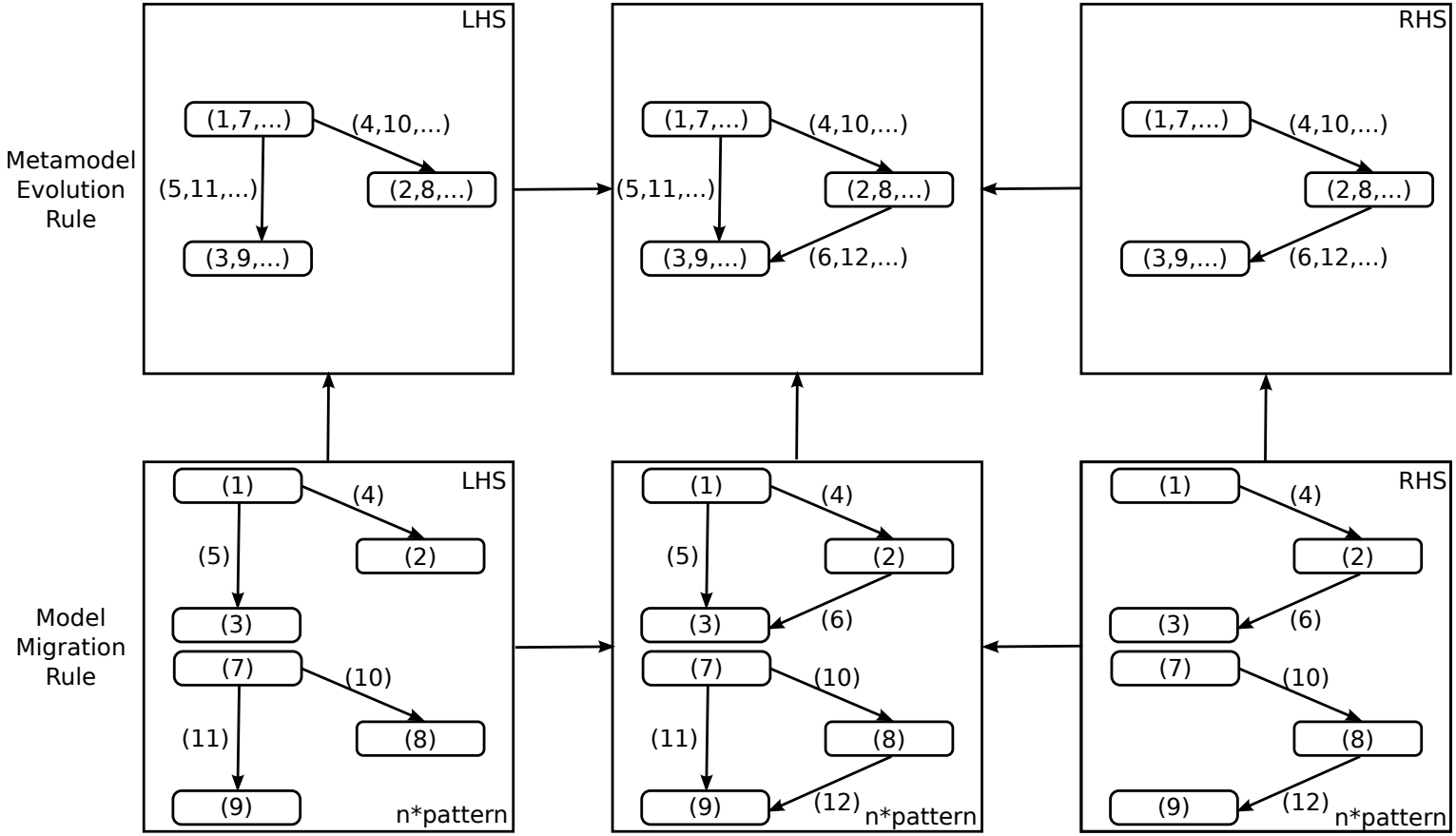
1. Specify metamodel evolution change as rule
2. Derive the metamodel migration rule from this rule

Cospan DPO approach



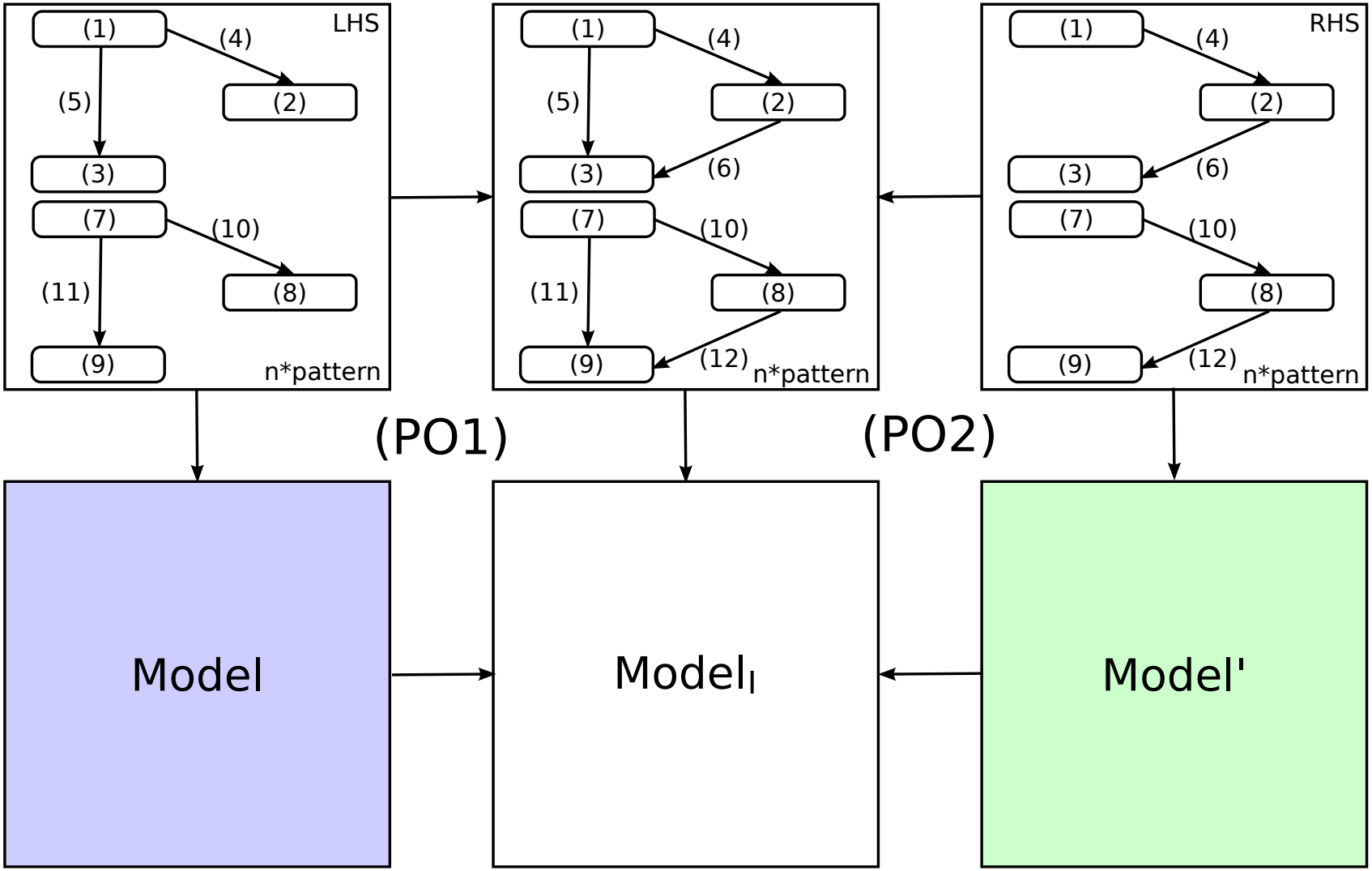
- Metamodel evolution rule as graph transformation rule

Derivation of migration rule

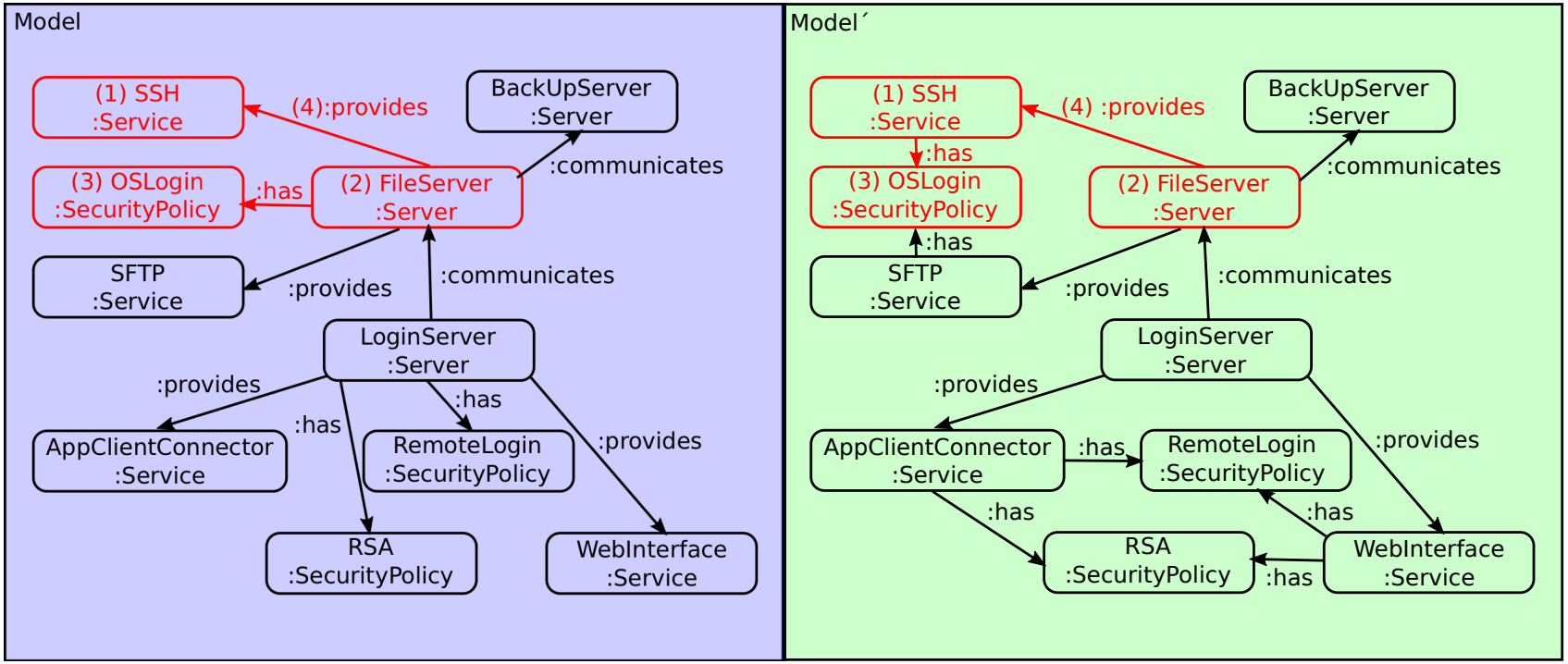


- Generated model migration rule as an amalgamated rule consisting of a set of isomorphic pattern of the metamodel evolution rule

Application of migration rule

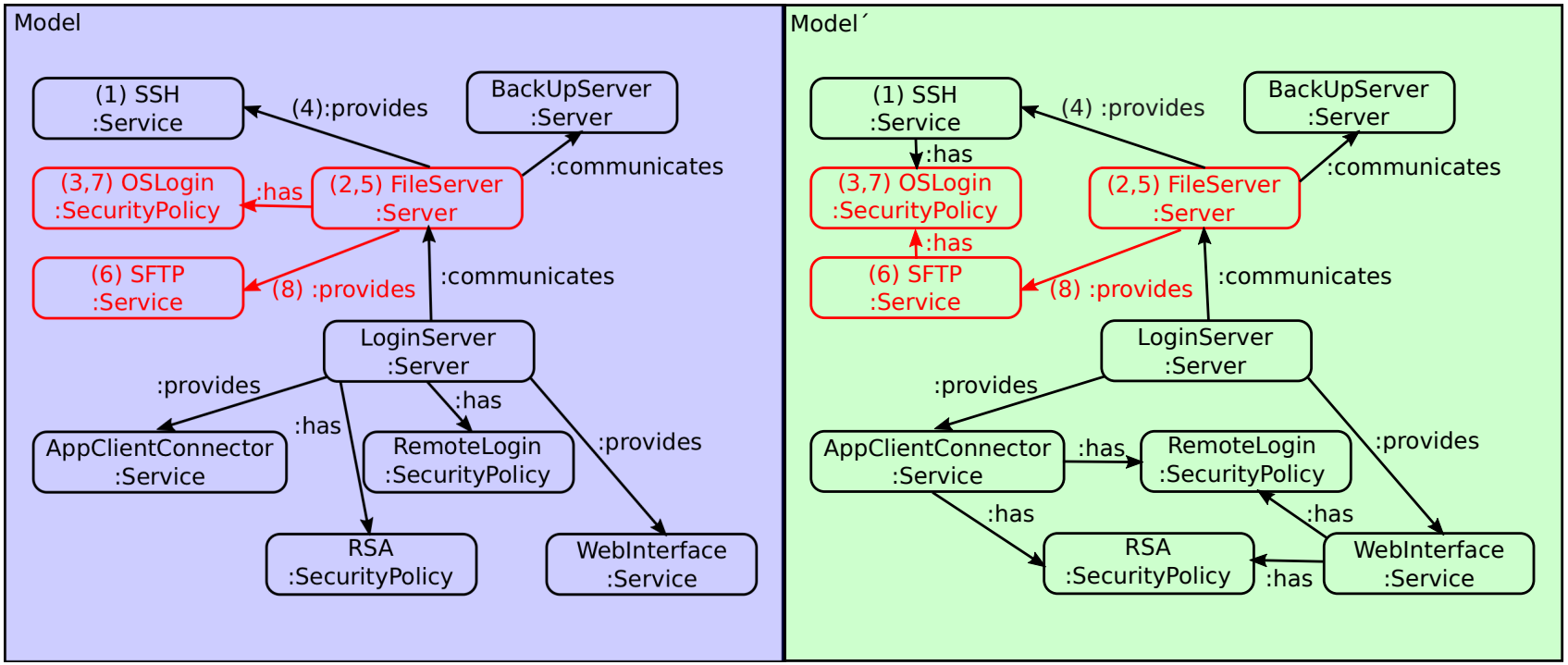


Application of migration rule



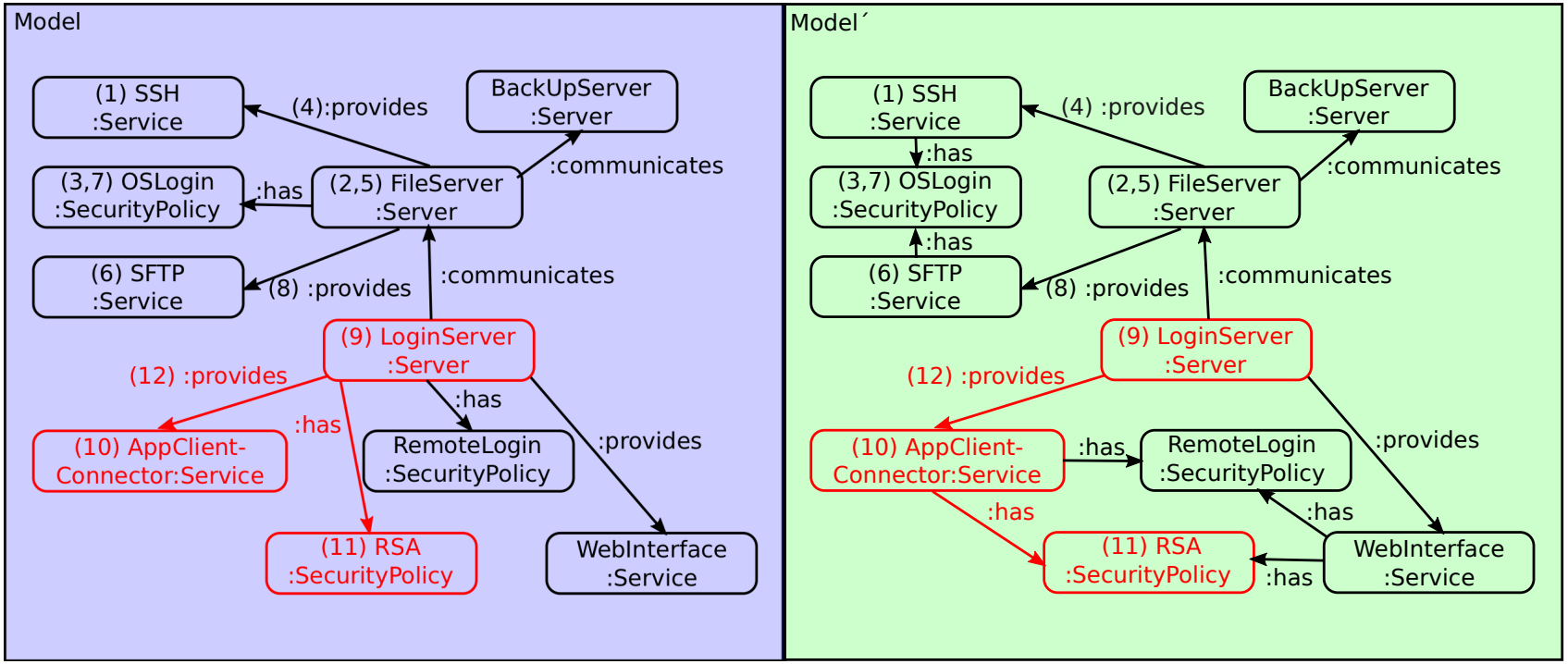
- One match of a sub-pattern of the migration rule
- There are as many patterns as possible matches

Application of migration rule



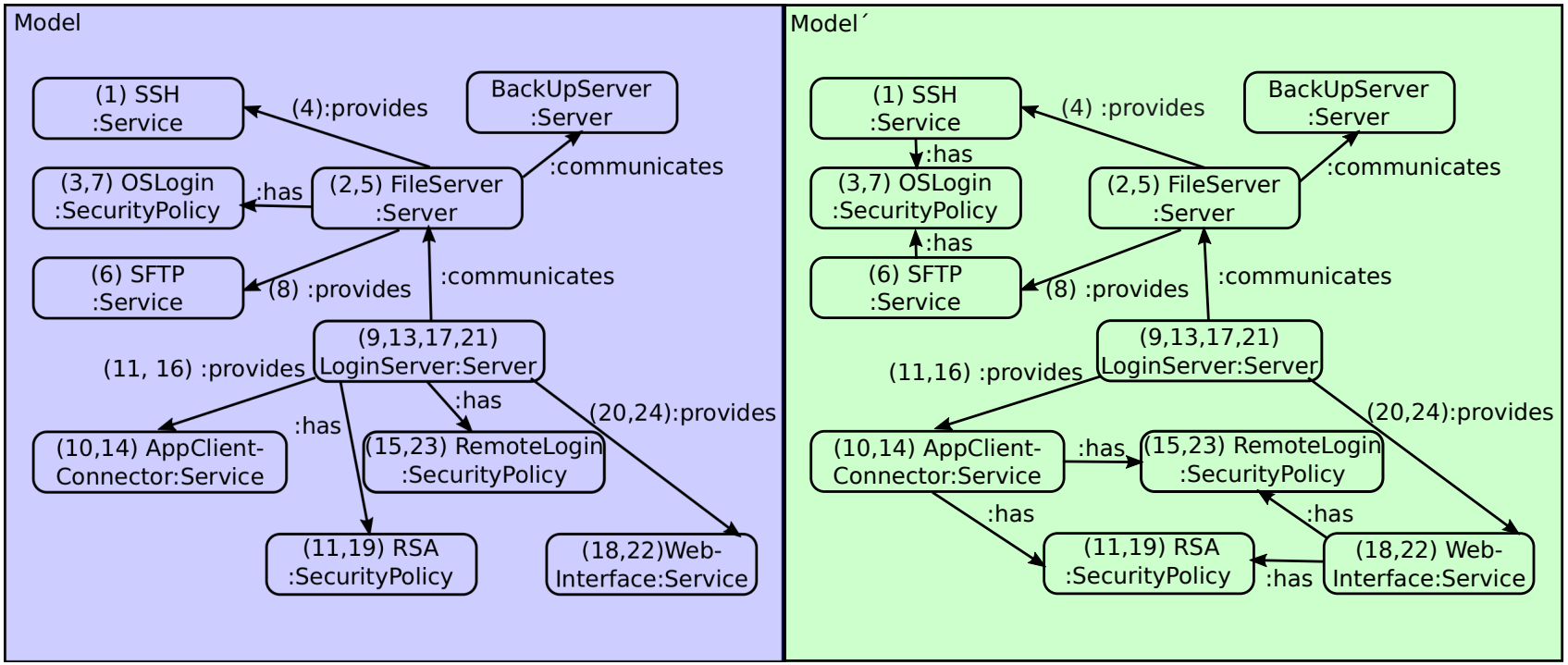
- A second match

Application of migration rule



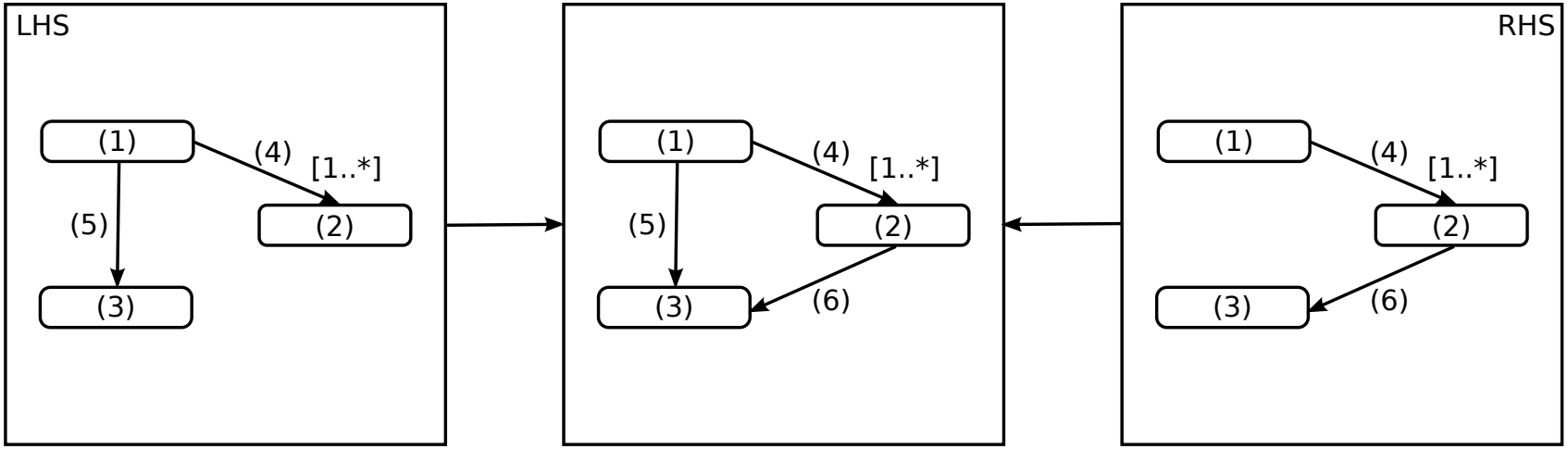
- A third match

Application of migration rule



- Complete match indicated by mapping numbers

Extension of metamodel evolution rule



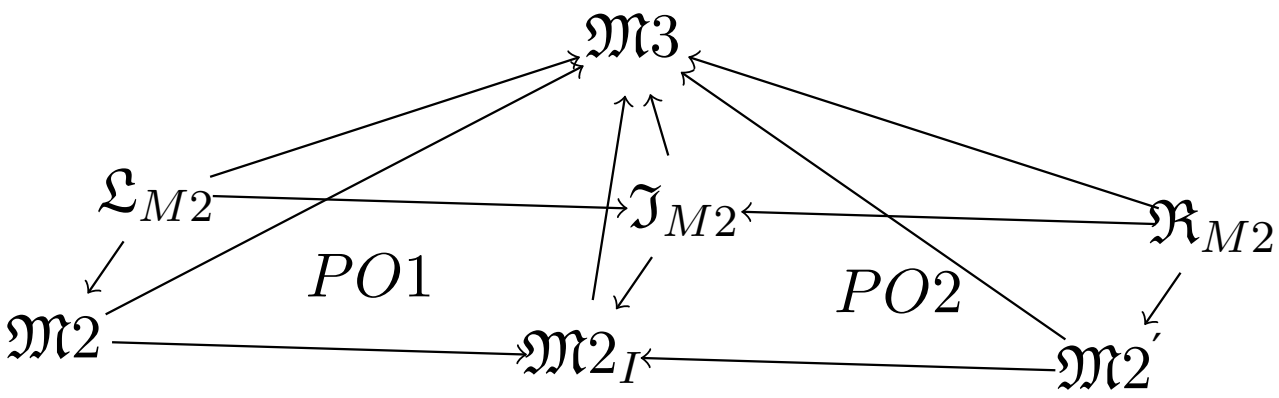
- Arrow (5) is deleted in the metamodel
- All corresponding arrows must be deleted in the model
- The migration rule must migrate all elements deleted by the metamodel evolution rule
- The metamodel evolution rule application is restricted using constraints

Formal approach

$$
\begin{tikzcd}
\mathfrak{L}_{M2} \arrow[rr] \arrow[d] \arrow[dr, phantom, "PO1"] & & \mathfrak{I}_{M2} \arrow[d] \arrow[dr, phantom, "PO2"] & & \mathfrak{R}_{M2} \arrow[ll] \arrow[d] \\
\mathfrak{M}2 \arrow[rr] & & \mathfrak{M}2_I & & \mathfrak{M}2' \arrow[ll]
\end{tikzcd}
$$

- We create a metamodel evolution rule using the Cospan DPO approach

Formal approach



- The rule conforms to a specification $\mathfrak{M}_3$

Formal approach

$$
\begin{tikzcd}
\mathfrak{L}_{M2} \arrow[rr] \arrow[d] \arrow[drr, phantom, "PO1"] & & \mathfrak{I}_{M2} \arrow[d] & & \mathfrak{R}_{M2} \arrow[ll] \arrow[d] \arrow[dll, phantom, "PO2"] \\
\mathfrak{M}2 \arrow[rr] & & \mathfrak{M}2_I & & \mathfrak{M}2' \arrow[ll]
\end{tikzcd}
$$

- From this rule we derive a model migration rule

Formal approach

$$
\begin{tikzcd}
\mathfrak{L}_{M2} \arrow[rr] \arrow[d] \arrow[dr, phantom, "PO1"] & & \mathfrak{I}_{M2} \arrow[d] \arrow[dr, phantom, "PO2"] & & \mathfrak{R}_{M2} \arrow[ll] \arrow[d] \\
\mathfrak{M}2 \arrow[rr] & & \mathfrak{M}2_I & & \mathfrak{M}2' \arrow[ll] \\
\overline{L}_{M1} \arrow[rr] \arrow[d] \arrow[dr, phantom, "PO1"] \arrow[uu] & & \overline{I}_{M1} \arrow[d] \arrow[dr, phantom, "PO2"] \arrow[uu] & & \overline{R}_{M1} \arrow[ll] \arrow[d] \arrow[uu] \\
M1 \arrow[rr] \arrow[uu] & & M1_I \arrow[uu] & & M1' \arrow[ll] \arrow[uu]
\end{tikzcd}
$$

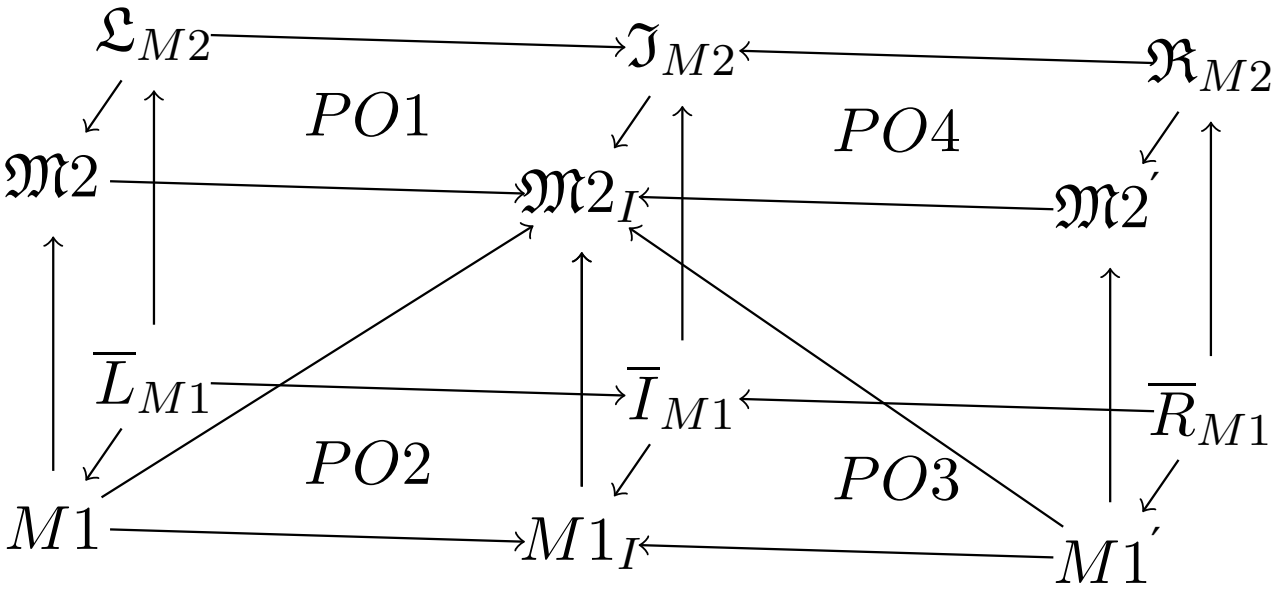
- The migration rule is also a Cospan DPO rule

Formal approach

$$
\begin{tikzcd}
\mathfrak{L}_{M2} \arrow[rr] \arrow[d] & & \mathfrak{I}_{M2} \arrow[d] & & \mathfrak{R}_{M2} \arrow[ll] \arrow[d] \\
\mathfrak{M}2 \arrow[rr] \arrow[u] \arrow[rr, phantom, shift left=4ex, "PO1"] & & \mathfrak{M}2_I \arrow[u] \arrow[rr, phantom, shift left=4ex, "PO4"] & & \mathfrak{M}2' \arrow[ll] \arrow[u] \\
\overline{L}_{M1} \arrow[rr] \arrow[d] \arrow[uu] & & \overline{I}_{M1} \arrow[d] \arrow[uu] & & \overline{R}_{M1} \arrow[ll] \arrow[d] \arrow[uu] \\
M1 \arrow[rr] \arrow[rr, phantom, shift left=4ex, "PO2"] & & M1_I \arrow[rr, phantom, shift left=4ex, "PO3"] & & M1' \arrow[ll]
\end{tikzcd}
$$

- We apply the metamodel evolution rule and the model migration rule interleaved to keep models permanently well-typed
- First we extend the metamodel, then we migrate the model and finally we reduce the metamodel

Formal approach



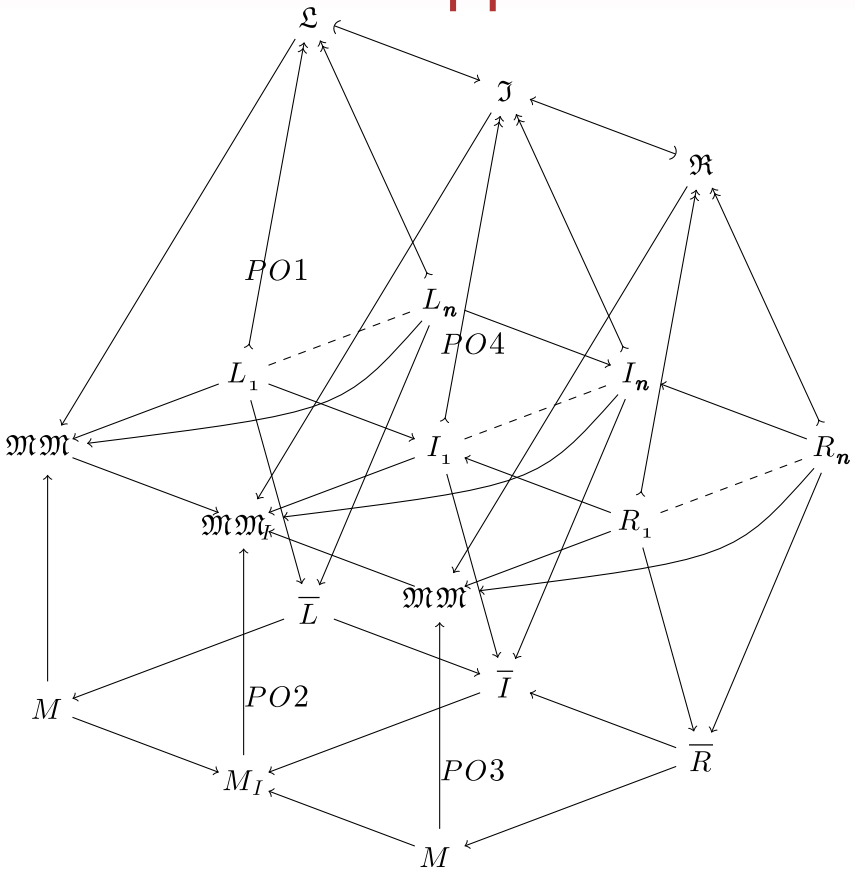
- Model $M1$ stays well-typed during the whole migration process

Formal approach

$$
\begin{tikzcd}
\mathfrak{L}_{M2} \arrow[rr] & & \mathfrak{I}_{M2} & & \mathfrak{R}_{M2} \arrow[ll] \\
L_n \arrow[u, tail, dashed] \arrow[rr] \arrow[d] & & I_n \arrow[u, two heads, dashed] \arrow[d] & & R_n \arrow[ll] \arrow[u, two heads, dashed] \arrow[d] \\
L_1 \arrow[u, tail, dashed] \arrow[rr] \arrow[dr] & & I_1 \arrow[u, two heads, dashed] \arrow[dr] & & R_1 \arrow[ll] \arrow[u, two heads, dashed] \arrow[dl] \\
\overline{L}_{M1} \arrow[rr] & & \overline{I}_{M1} & & \overline{R}_{M1} \arrow[ll]
\end{tikzcd}
$$

- The migration rule is represented by an amalgamated rule resembling the metamodel evolution rule

Formal approach



- All morphisms between metamodel, model, evolution- and migration rule

Conclusion

Future Work

- Improve rule derivation strategy
- Consider further constraints on the metamodel level
- Guide rule derivation where alternative migrations are possible

Thank you!

Questions?

For more information visit: `http://dpf.hib.no/`