Formal Support for QVT-Relations with Coloured Petri Nets

Juan de Lara

Univ. Autónoma de Madrid (Spain)

Esther Guerra

Univ. Carlos III de Madrid (Spain)

MODELS'2009
Denver, Colorado, USA

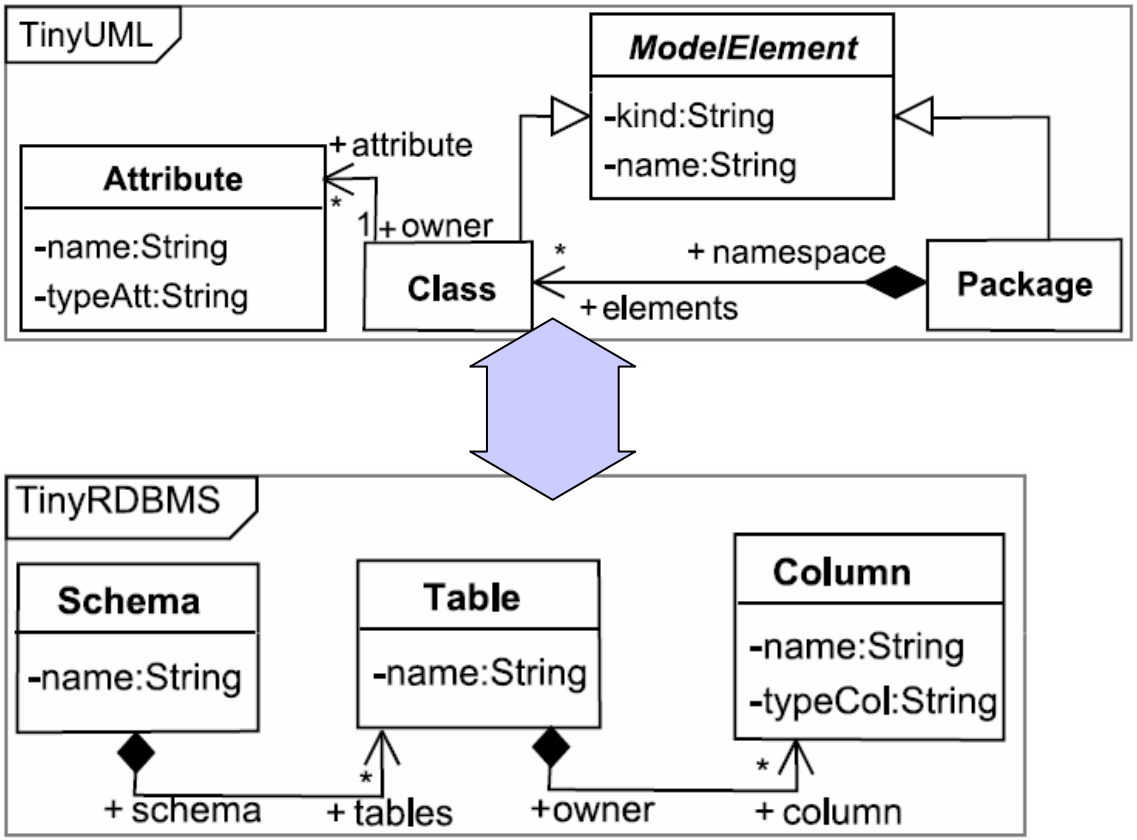
Motivation

- Model-to-Model (M2M) transformation is a key technology in Model-Driven Development.
- QVT is the OMG standard transformation language.
  - QVT-Relations (QVT-R) is the highest-level of abstraction part of the specification.
  - Its semantics is only semi-formally specified.
  - Tool support is still scarce.
- Give formal semantics to QVT-R through its compilation into Coloured Petri nets (CPNs).
  - Enable the analysis of QVT-R specifications.
  - Tool support by *CPNTools*.
  - Allow execution and debugging of QVT-R specifications.

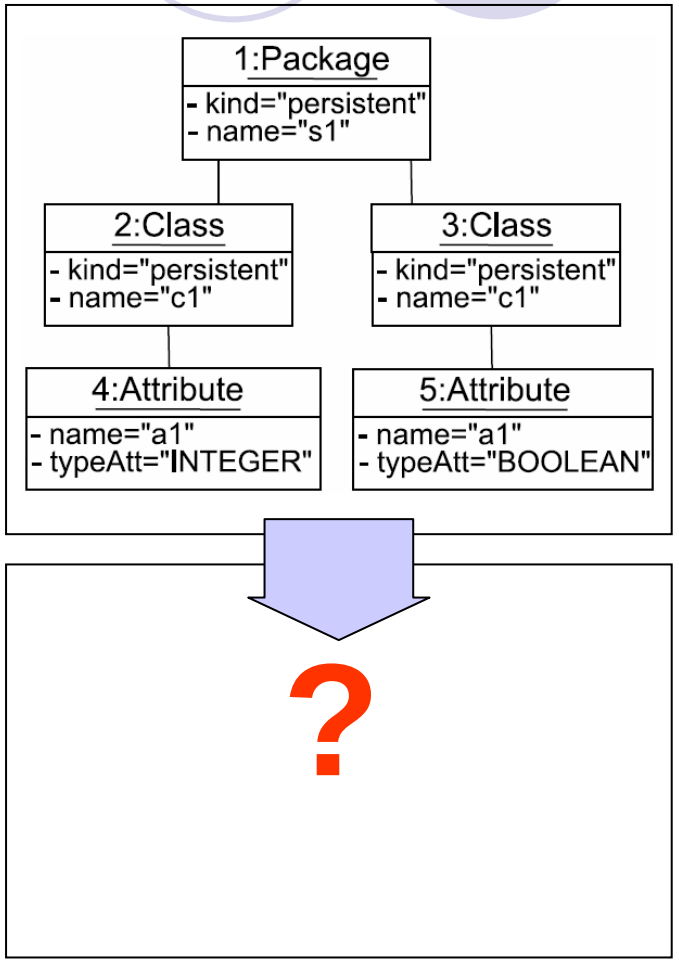
Index

- **Introduction**
  - **Case Study.**
  - **QVT-Relations.**
  - **Coloured Petri Nets.**
- Compiling QVT-R into CPNs.
- Analysis of QVT-R transformations.
- Conclusions and Future Work.

Case Study



Meta-model ***Model***



QVT-Relations

- QVT-R specifications are made of a set of relations, each having two or more domains.
  - Declarative.
  - Bidirectional.
- Domains are described by patterns.
- Enforced vs. Checkonly domains.

```
transformation umlToRdbms(uml:TinyUML, rdbms:TinyRDBMS) {
  top relation PackageToSchema { // maps each package to a schema
    pn: String;
    checkonly domain uml p:Package {name=pn, kind='persistent'};
    enforce domain rdbms s:Schema {name=pn};
  }
… }
```

QVT-Relations

- When and where clauses:
  - **when**: Conditions under which the relation needs to hold.
  - **where**: Extra conditions that need to hold if the relation holds.

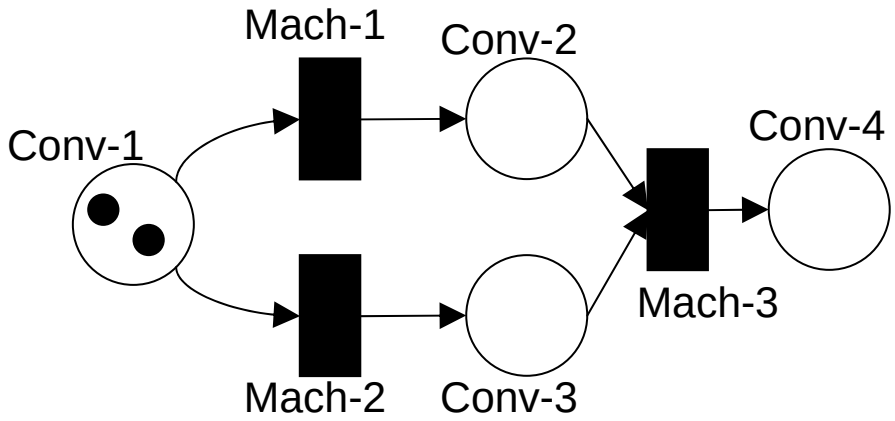
```
top relation ClassToTable {// maps each persistent class to a table
    cn, prefix: String;
    checkonly domain uml c:Class {namespace=p:Package {},
                                  kind='persistent', name=cn};
    enforce domain rdbms t:Table {  schema=s:Schema {}, name=cn};
    when {PackageToSchema(p, s);}
    where {prefix=''; AttributeToColumn(c, t, prefix); }
}
```

- Transformation into a lower-level language called *QVT-core*.

- **Check-Before Enforce Semantics**: before creating an element, check if an already existing one can be reused.

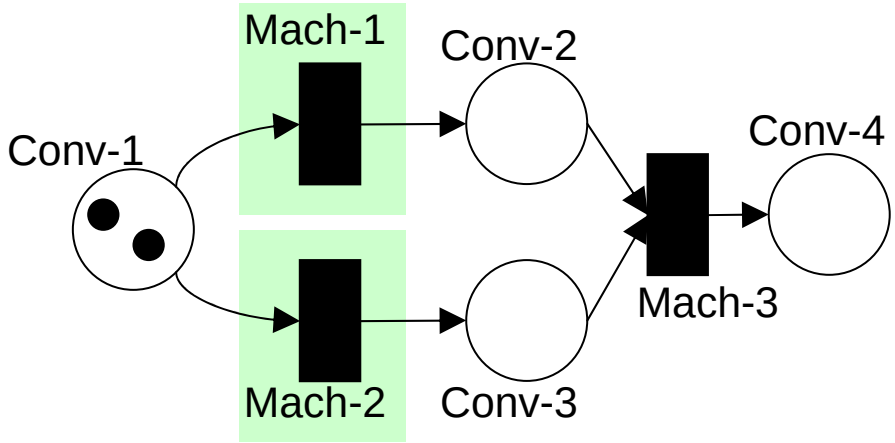
Petri nets

- Popular formalism to model and analyse concurrent systems.



Petri nets

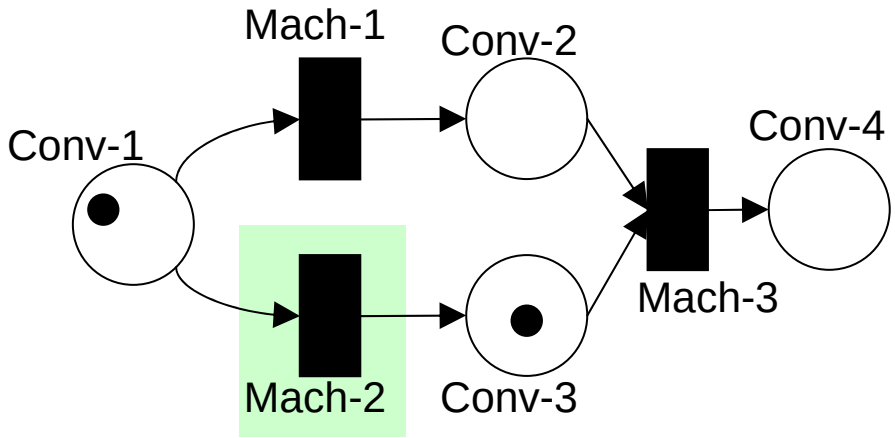
- Popular formalism to model and analyse concurrent systems.



Both Mach-1 and Mach-2 enabled

Petri nets

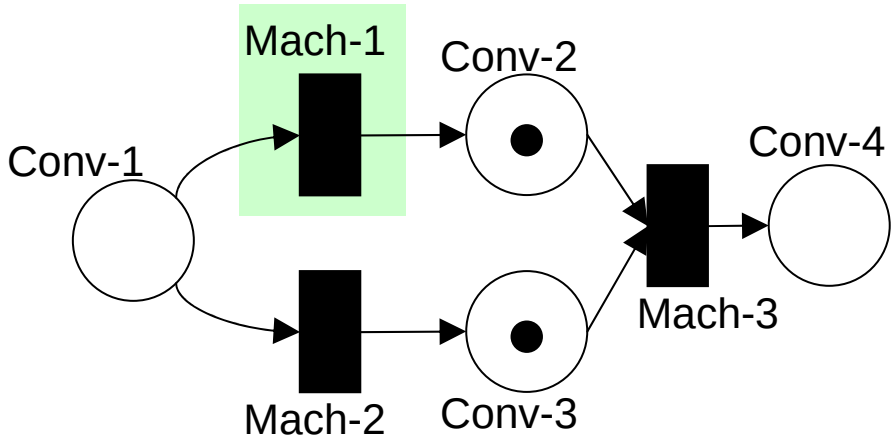
- Popular formalism to model and analyse concurrent systems.



One of them can fire

Petri nets

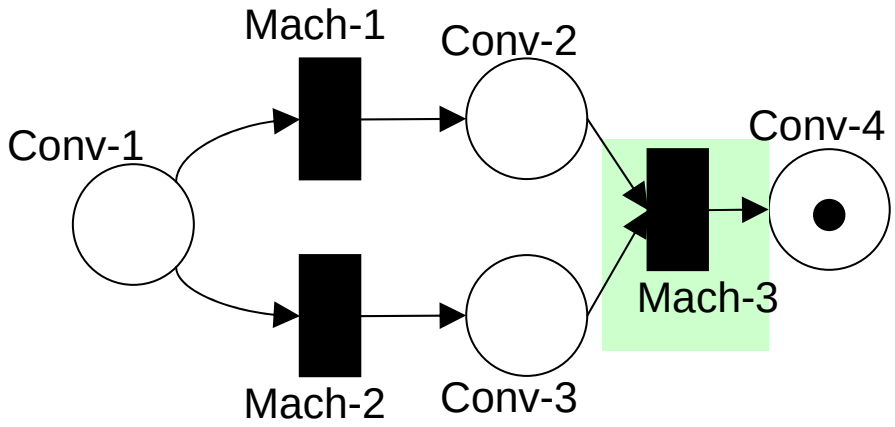
- Popular formalism to model and analyse concurrent systems.



Mach-1 fires

Petri nets

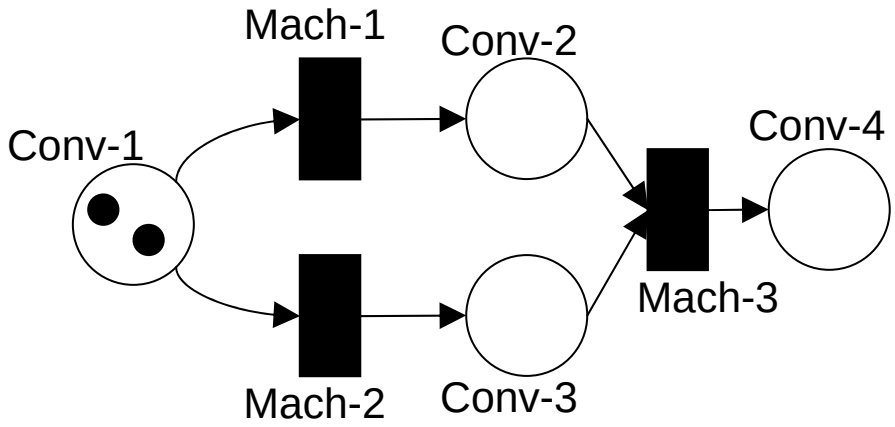
- Popular formalism to model and analyse concurrent systems.

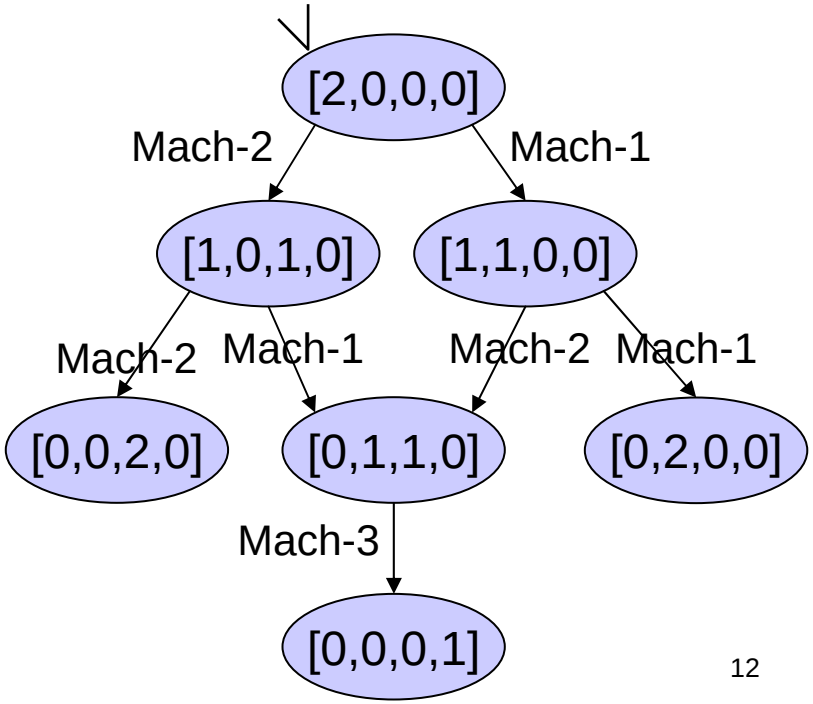


Mach-3 fires

Petri nets

- Reachability Graph: graph representation of the set of reachable markings (states).

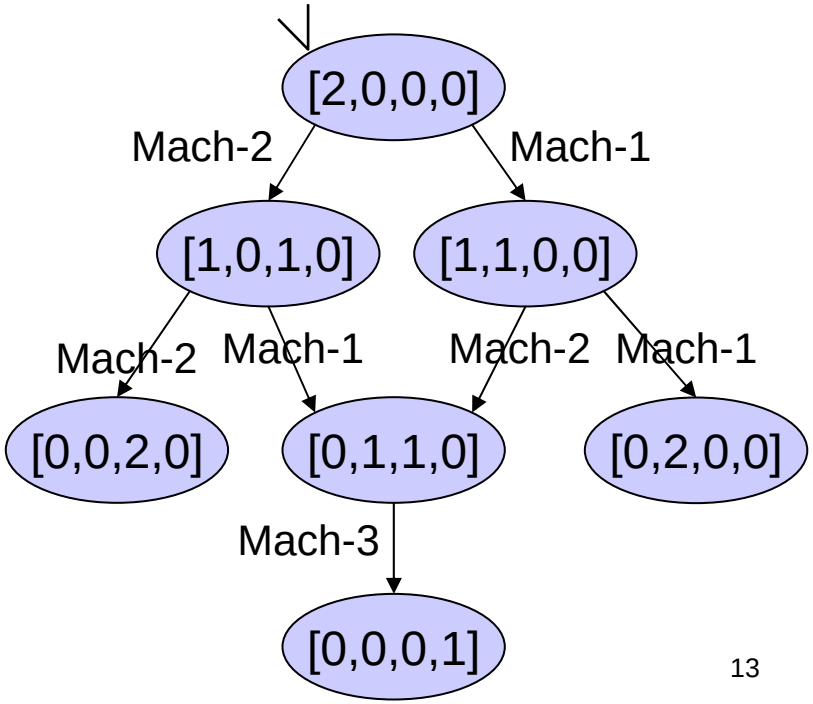




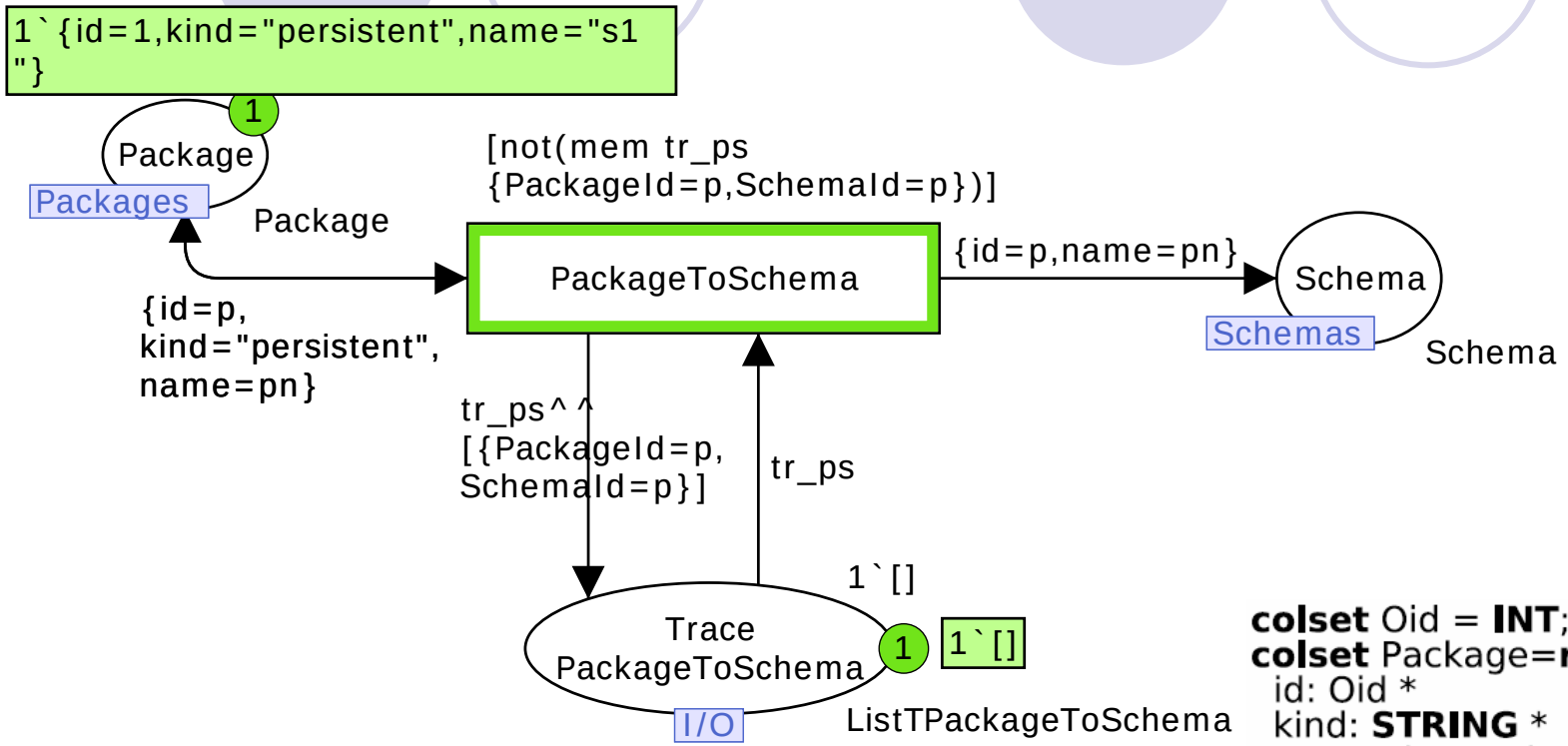
Petri nets

- Reachability Graph: graph representation of the set of reachable markings (states).

- Allows investigating:
  - reachability.
  - deadlocks.
  - invariants.
  - reversibility.
  - persistence.
  - …



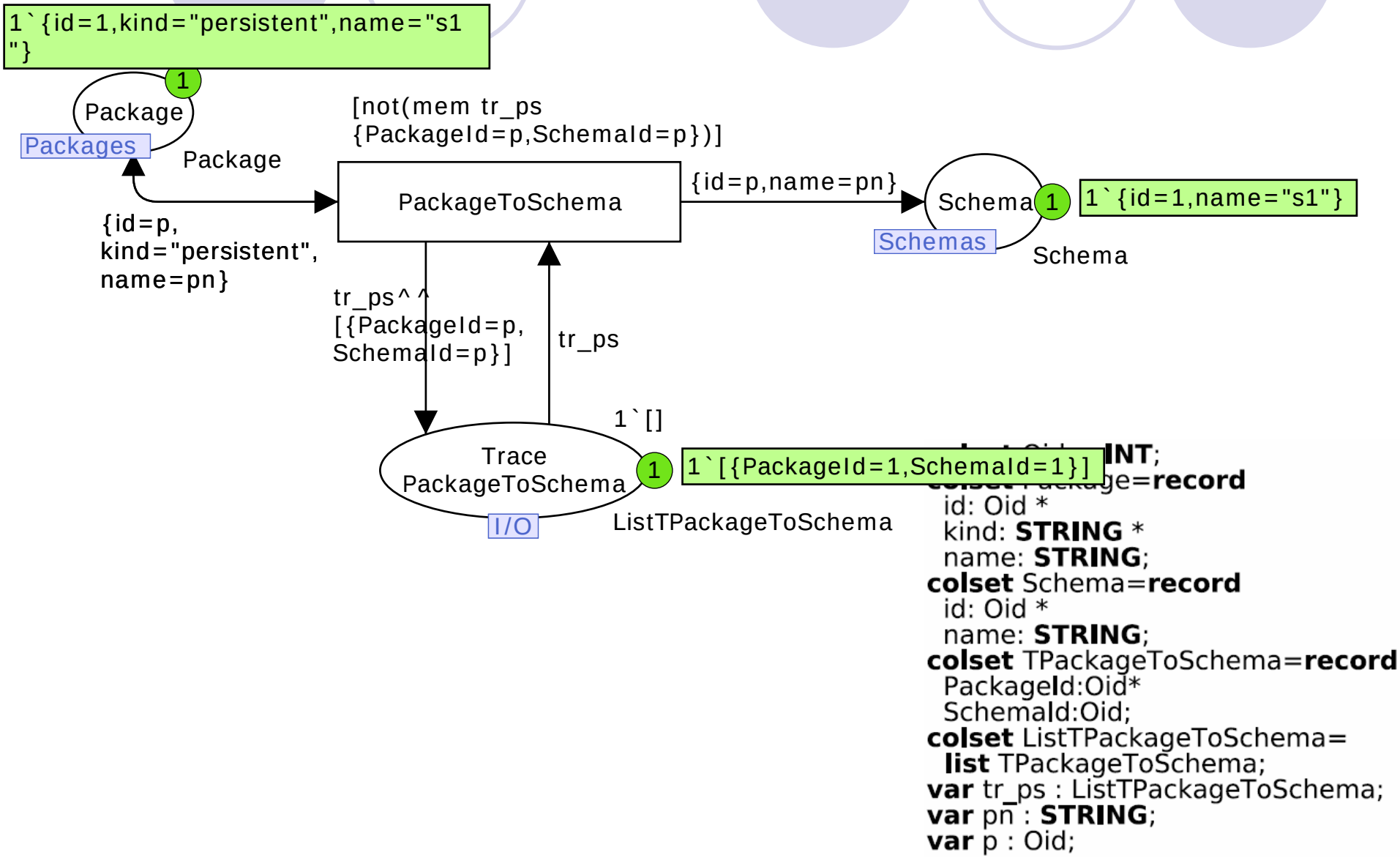
Coloured Petri nets




```
colset Oid = INT;
colset Package=record
  id: Oid *
  kind: STRING *
  name: STRING;
colset Schema=record
  id: Oid *
  name: STRING;
colset TPackageToSchema=record
  PackageId:Oid*
  SchemaId:Oid;
colset ListTPackageToSchema=
  list TPackageToSchema;
var tr_ps : ListTPackageToSchema;
var pn : STRING;
var p : Oid;
```

- Places are assigned data structures.
- Tokens carry data.
- Arcs contain expressions with variables, to be bound with tokens.
- Transitions have guards.

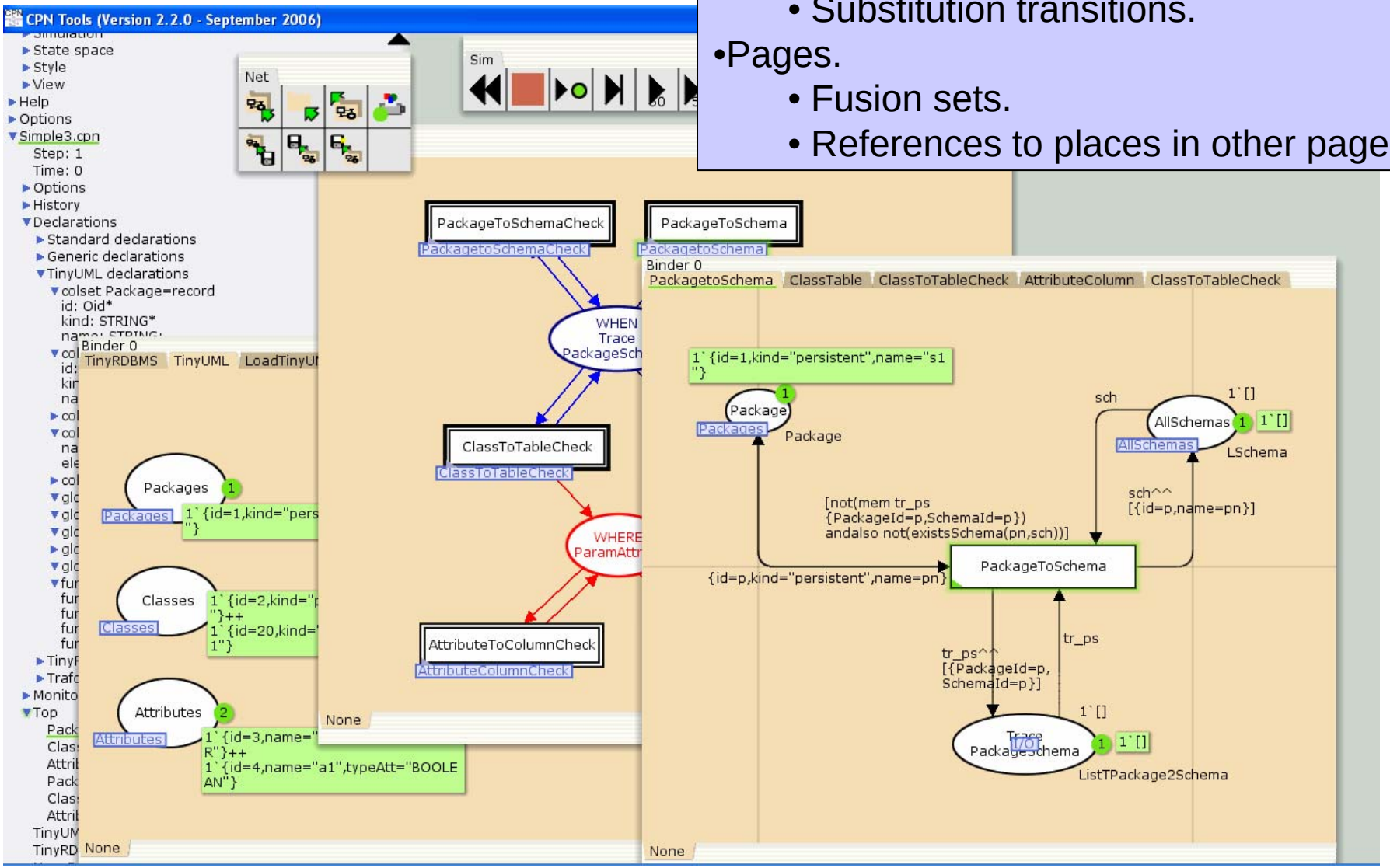
Coloured Petri nets



Coloured Petri nets

CPNTools

- Hierarchy.
  - Substitution transitions.
- Pages.
  - Fusion sets.
  - References to places in other pages.



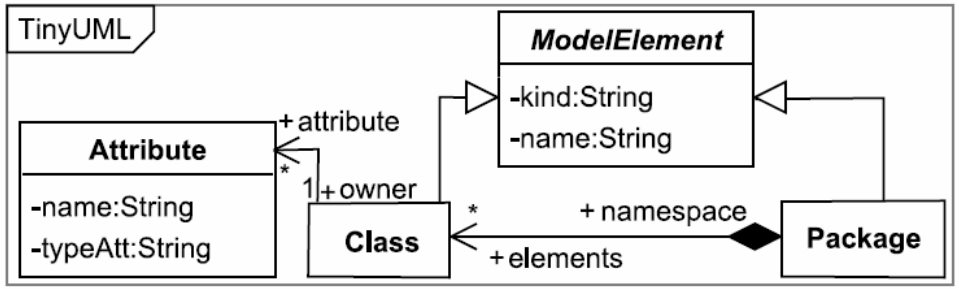
Index

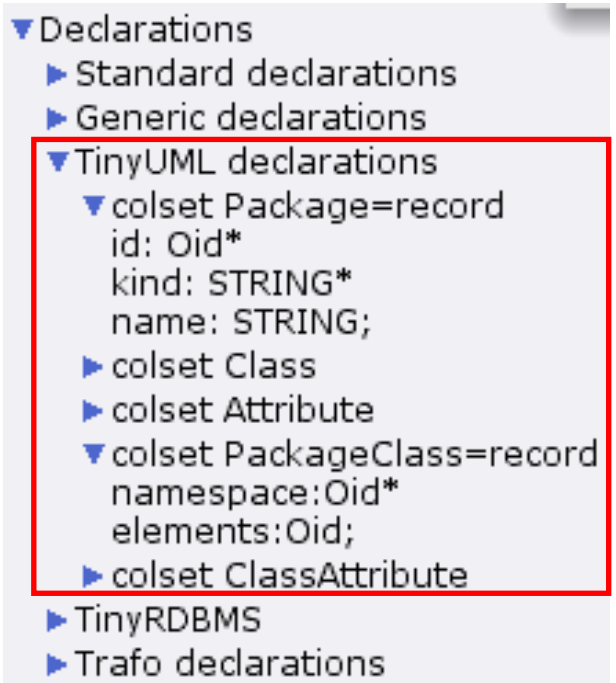
- Introduction.
- **Compiling QVT-R into CPNs.**
  - **Compiling the meta-models and the initial model.**
  - **Compiling the relations.**
  - **The When and Where clauses.**
  - **High-level view.**
  - **CBE Semantics.**
- Analysis of QVT-R transformations.
- Conclusions and Future Work.

Compilation into CPNs

Meta-models

- Compile meta-models into colour sets declarations.
- Transitive closure of inheritance:
  - copy attributes and relations from ancestors to children classes.
- A record for each class and association.
  - Identifiers.

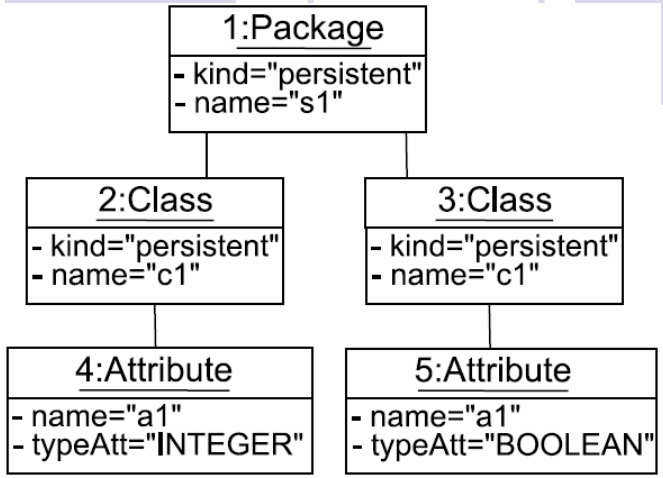


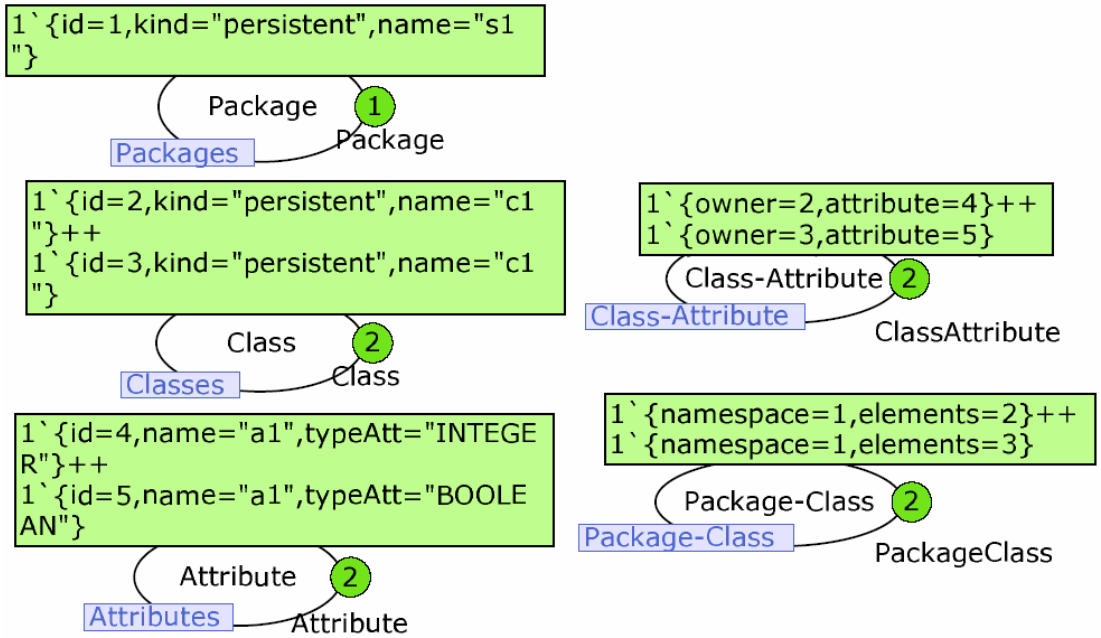


Compilation into CPNs

Initial model

- One place for each element of the meta-model.
- Tokens for each element of the model.



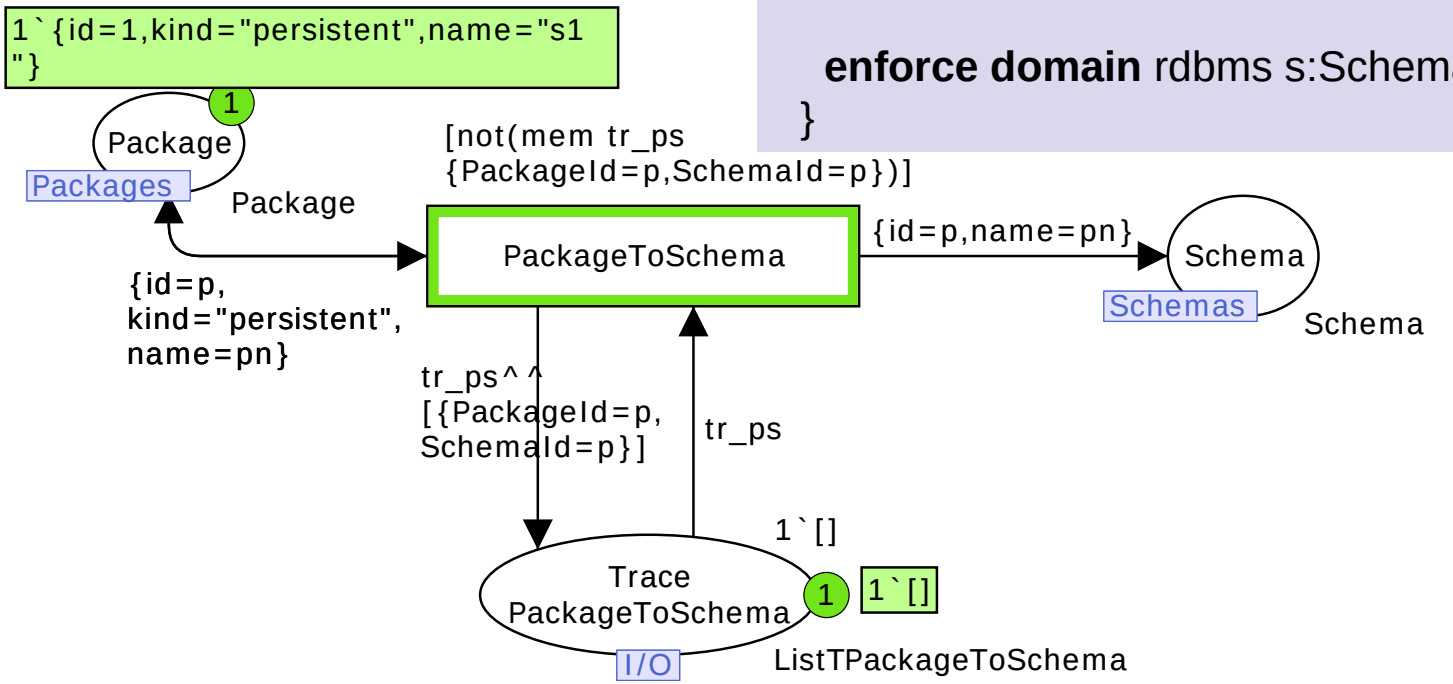


Compilation into CPNs

The relations

- One transition for each relation and one place for each type in each domain.
  - Read arcs, or write arcs depending on check/enforce and the direction of execution.
  - Arcs with variables acc
- A place for the traces,

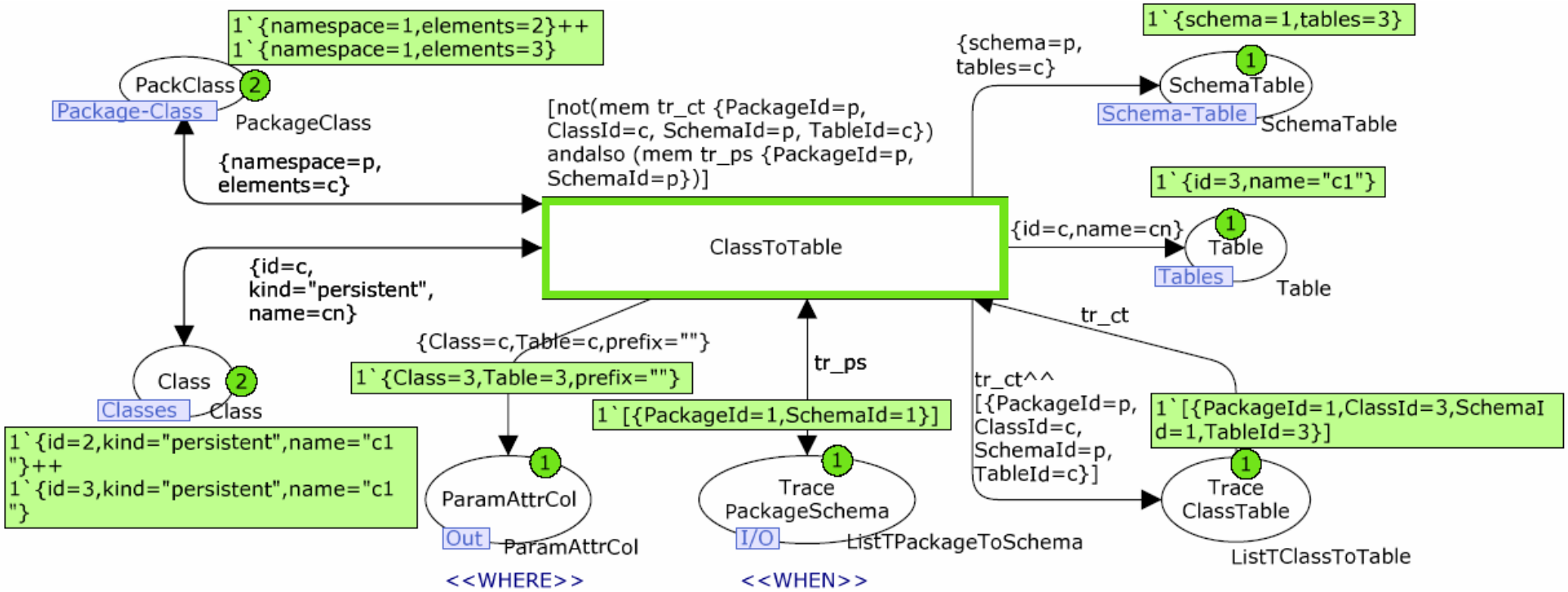
```
top relation PackageToSchema {
    pn: String;
    checkonly domain uml p:Package {  name=pn,
                                      kind='persistent'};
    enforce domain rdbms s:Schema  {  name=pn};
}
```



Compilation into CPNs

Th

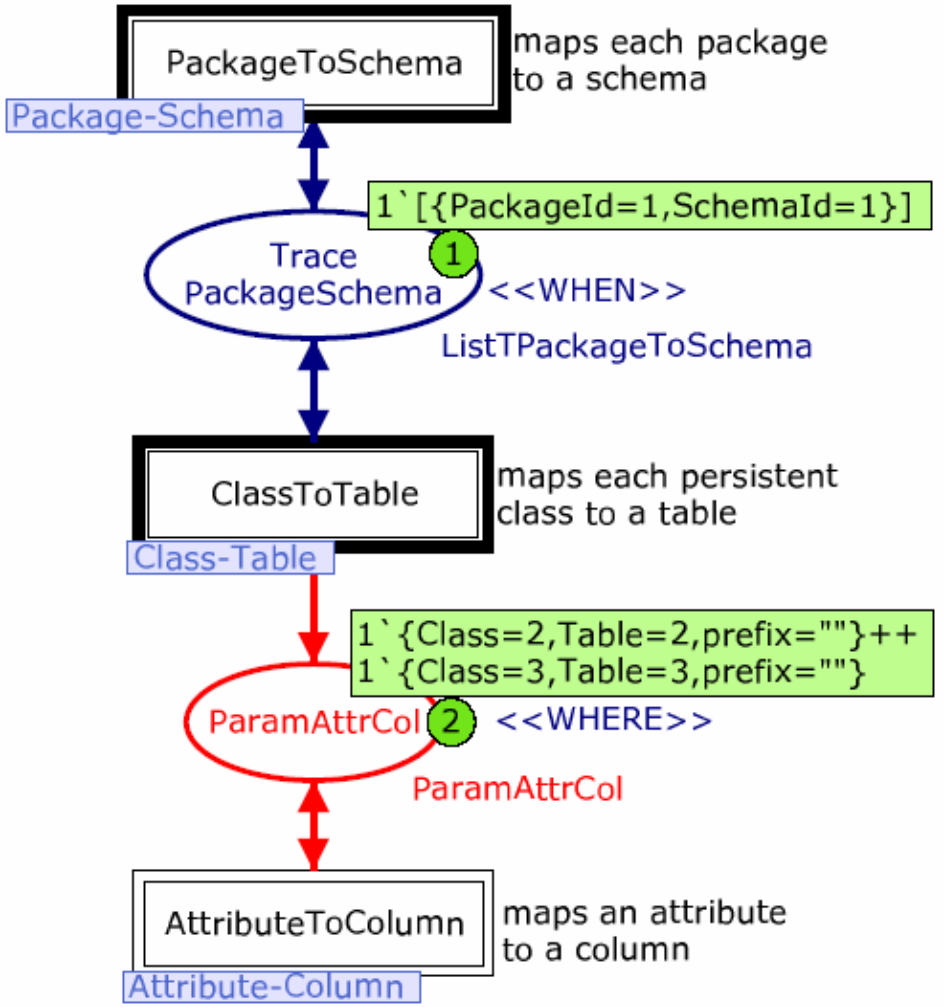
```
top relation ClassToTable {// maps each persistent class to a table
    cn, prefix: String;
    checkonly domain uml c:Class {namespace=p:Package {},
                                  kind='persistent', name=cn};
    enforce domain rdbms t:Table {  schema=s:Schema {}, name=cn};
    when {PackageToSchema(p, s);}
    where {prefix=''; AttributeToColumn(c, t, prefix); }
}
```



Compilation into CPNs

High-Level View

- Use of the hierarchical capabilities of CPNs.
- Facilitates debugging:
  - Observe the parameter flow of the different relations, the information in the traces, etc.



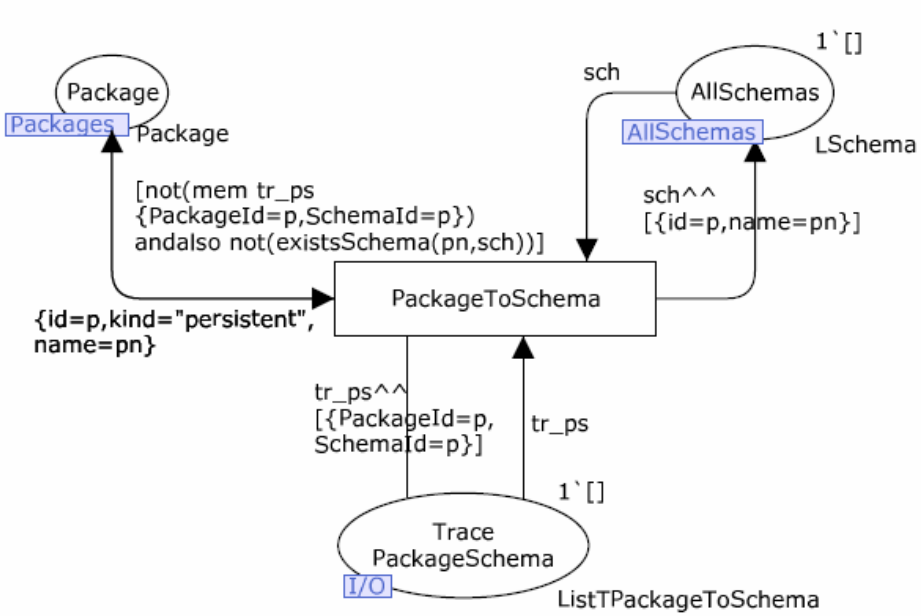
Compilation into CPNs

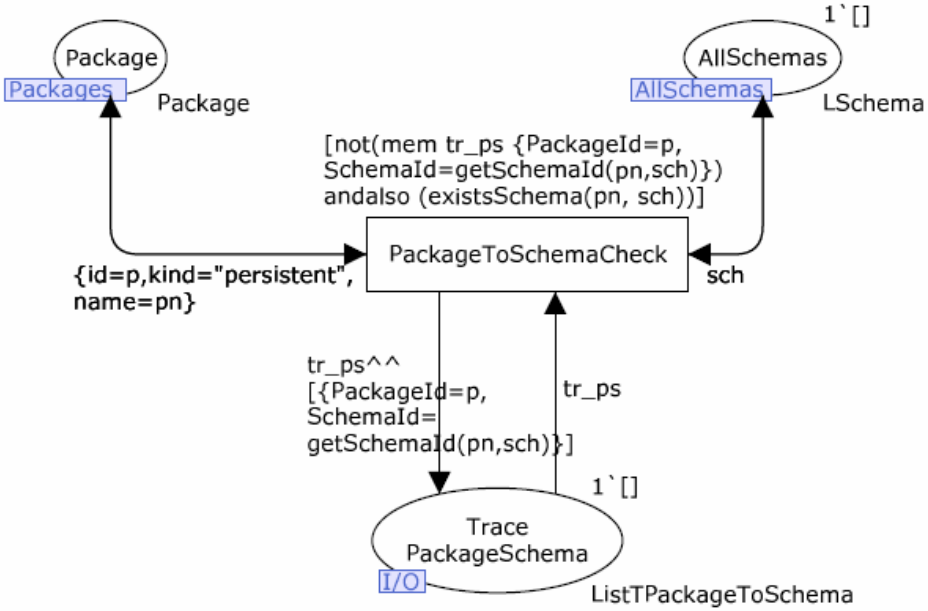
CBE Semantics

- Check if a suitable object exists before creating a new one.
- Keys allow specifying when two objects are considered equal.
- Generate several transitions per relation:
  - Mutually exclusive.
  - Each try to reuse increasingly bigger parts of the enforced domain.
- We need to test if objects are not present:
  - Inhibitor arcs are not supported by CPNs.
  - Use lists instead of “discrete” tokens in enforced domain.
  - Each place contains a list, initially empty.

Compilation into CPNs

CBE Semantics





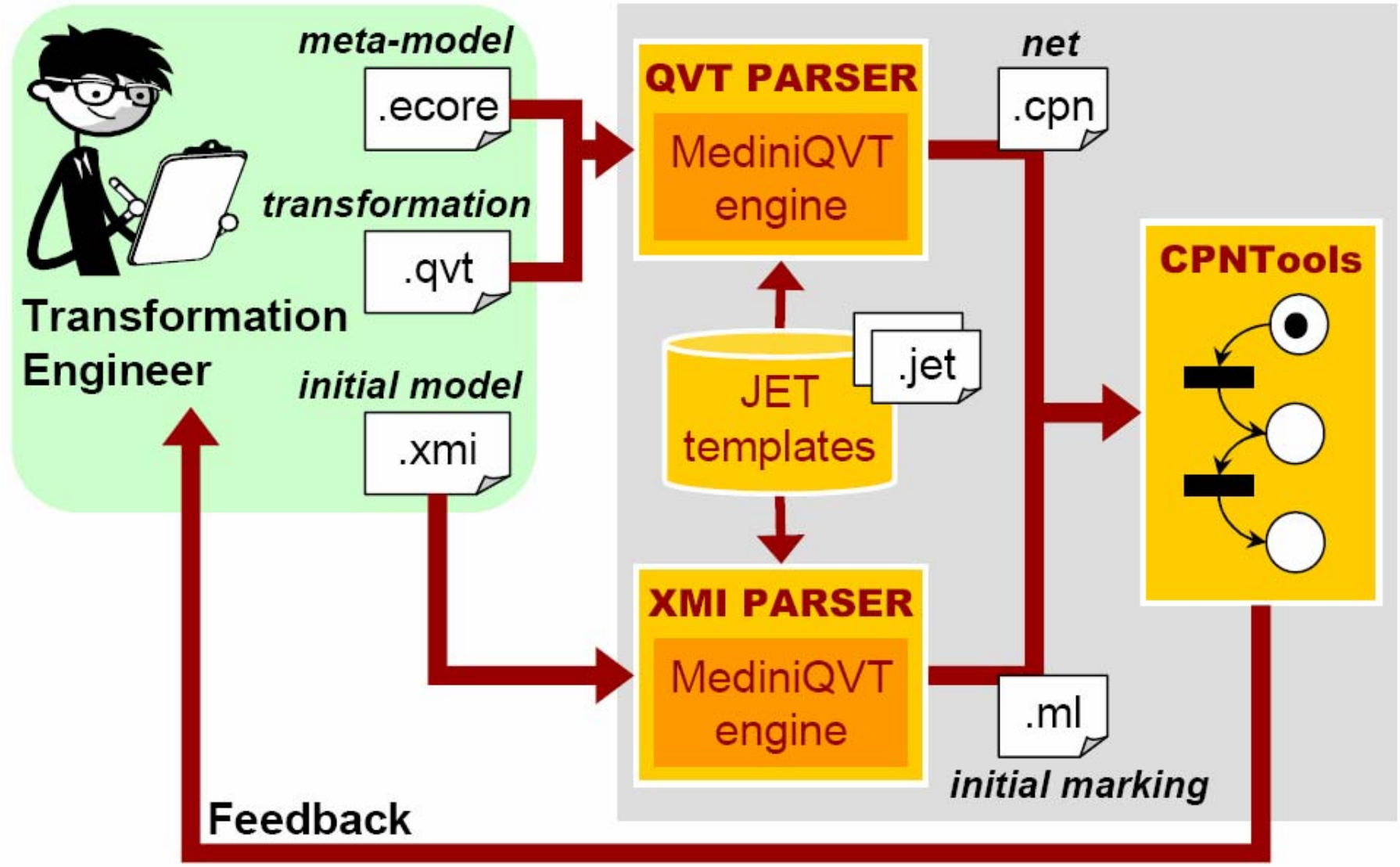
- Creates a new Schema, if no suitable one exists.
- Reuses the existing schema.

(the key of the Schema class is just its name)

Index

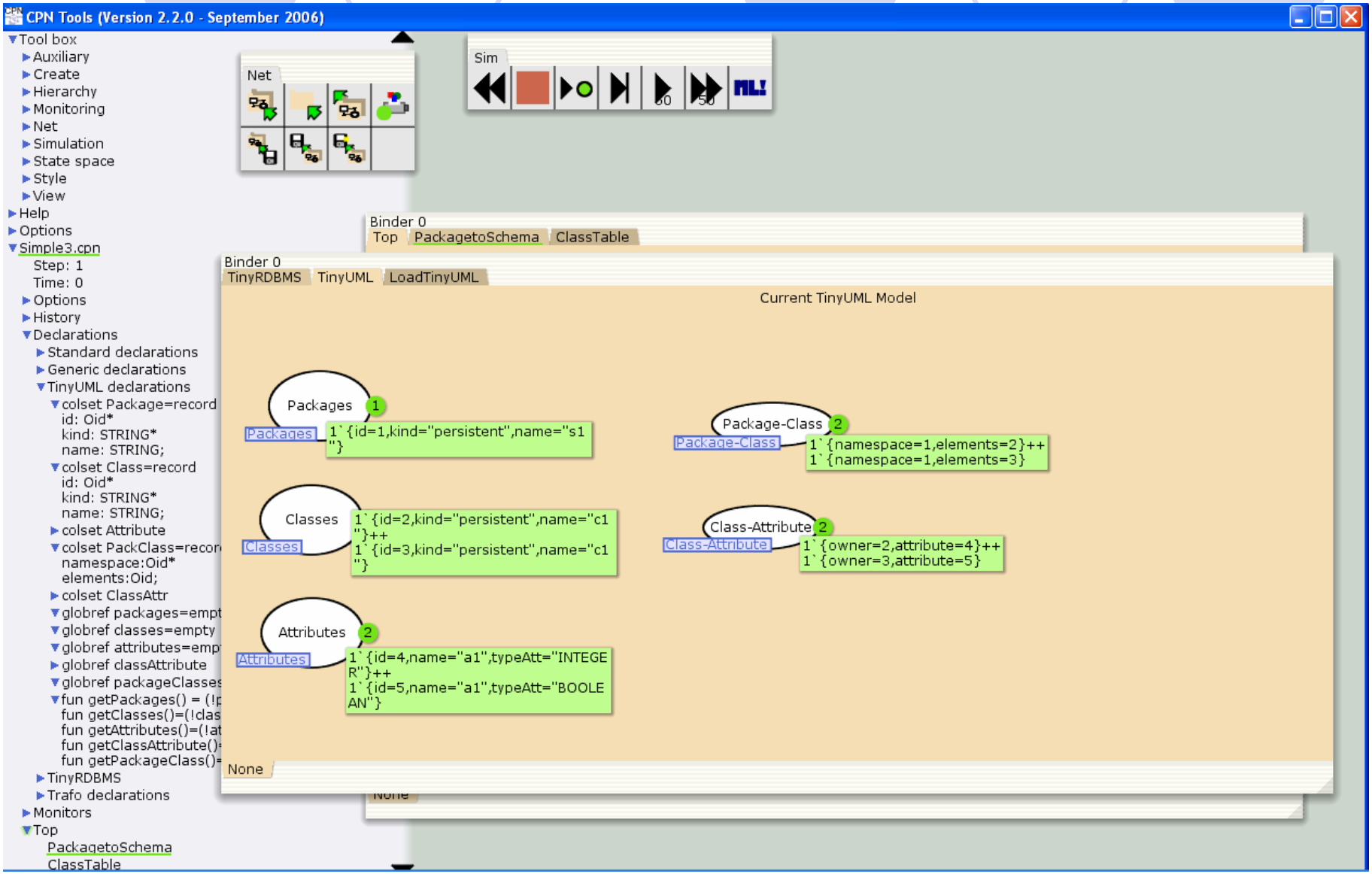
- Introduction.
- Compiling QVT-R into CPNs.
- **Analysis of QVT-R transformations.**
  - **Tool Support.**
  - **Validation.**
  - **Verification.**
- Conclusions and Future Work.

Tool Support



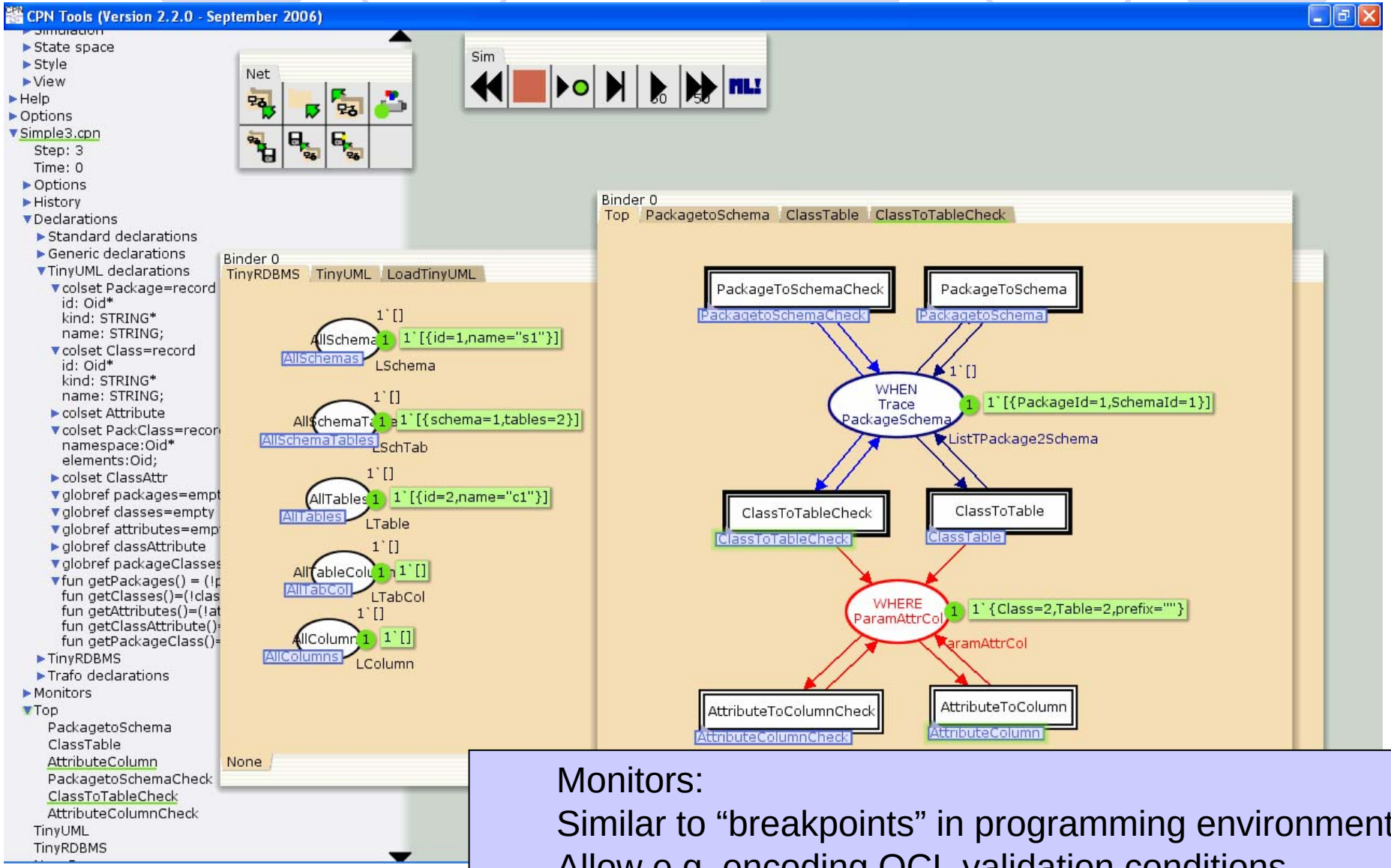
Tool Support

Validation with CPNTools



Tool Support

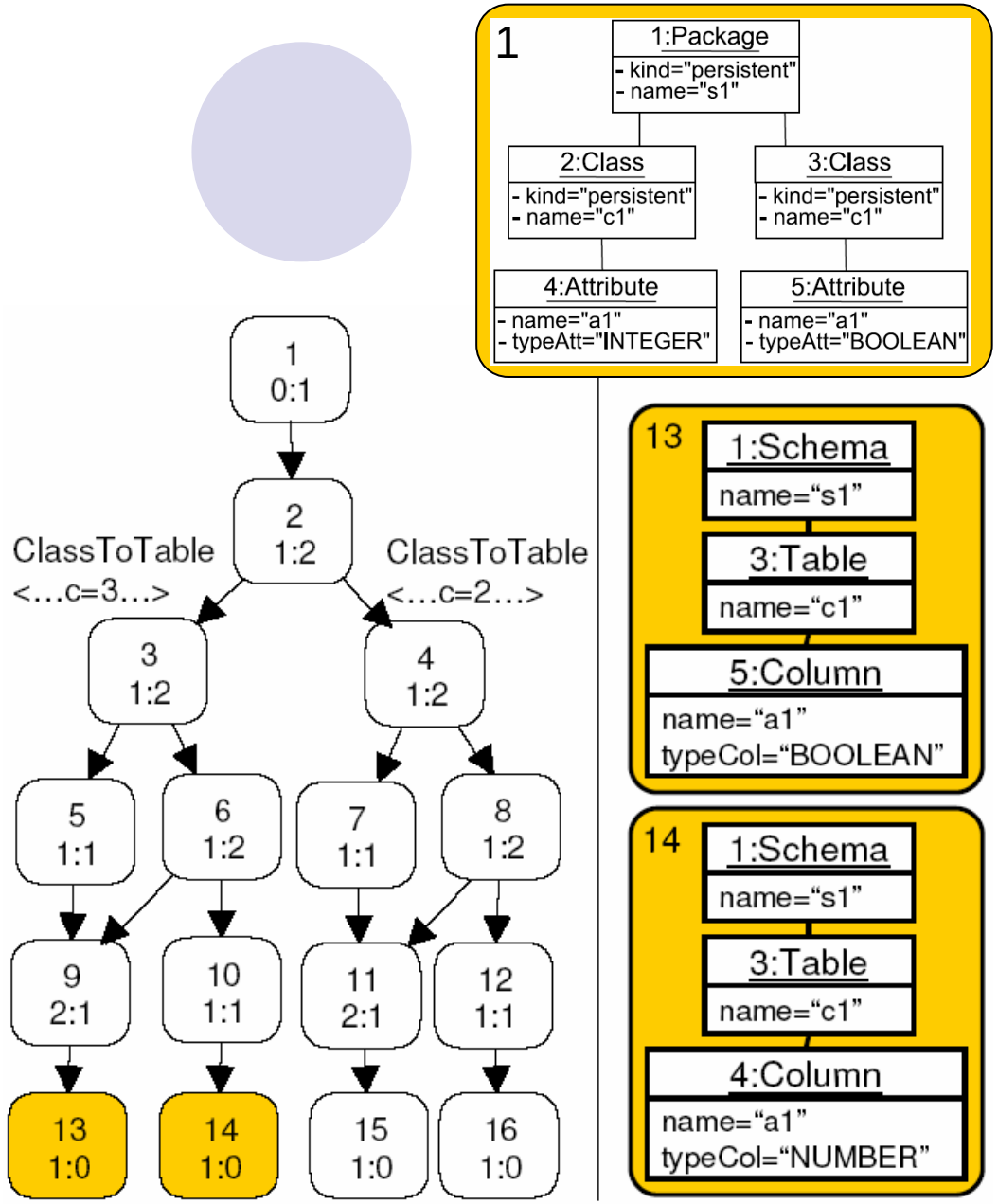
Validation with CPNTools



Monitors:
Similar to “breakpoints” in programming environments
Allow e.g. encoding OCL validation conditions.

Verification

- Occurrence graph.
- Graph representation of all possible reachable states.
- Allows investigating:
  - Confluence.
  - Termination.
  - Relation conflicts (through transition persistence).



(assumming that the key for columns includes the name and table, but not its type)

Verification

- Other properties:
  - **Boundedness and Invariants**.
    - Non-creation of certain types of elements in enforced domains.
    - E.g. whether in a class diagram where classes do not have attributes, it is an invariant that tables do not have columns.
  - **Model checking**.
    - Whether some structure can be produced in the enforced domain.
    - E.g. whether all transformation paths produce a column of type BOOLEAN:

```
eval node INV(POS(NF("Has Bool Column", hasColumn))) InitNode
```

    - … or whether we always obtain as many columns as attributes, the same number of tables as classes, etc.

Index

- Introduction
- Compiling QVT-R into CPNs.
- Analysis of QVT-R transformations.

- **Conclusions and Future Work.**

Conclusions

- Formal semantics for QVT-R through its compilation into CPNs.
- Allows applying the theory of Petri nets to model-to-model transformation:
  - Execution, validation and verification.
  - Confluence, termination, conflicts, boundedness, invariants, model-checking.
- “Low-cost” tool support for QVT-R through CPNTools.
- Main limitation: full support for OCL.

Future Work

- Improve tool support. More experimentation.
- Back-annotation (hide CPNs).
- User friendly ways to express verification properties.
- Higher-level formal semantics of QVT-R by our “pattern-based” approach to model-to-model transformation.

Thanks!

Questions?

Formal Support for QVT-R with Coloured Petri Nets

Juan de Lara (Juan.deLara@uam.es) *Univ. Autónoma de Madrid*
Esther Guerra *Univ. Carlos III de Madrid*