Mapping OCL as a Query and Constraint Language

Carolina Inés Dania Flores

PhD defense

Supervisors:
Manuel García Clavel - Marina Egea González

Universidad Complutense de Madrid, Madrid, Spain
30th of June, 2017.

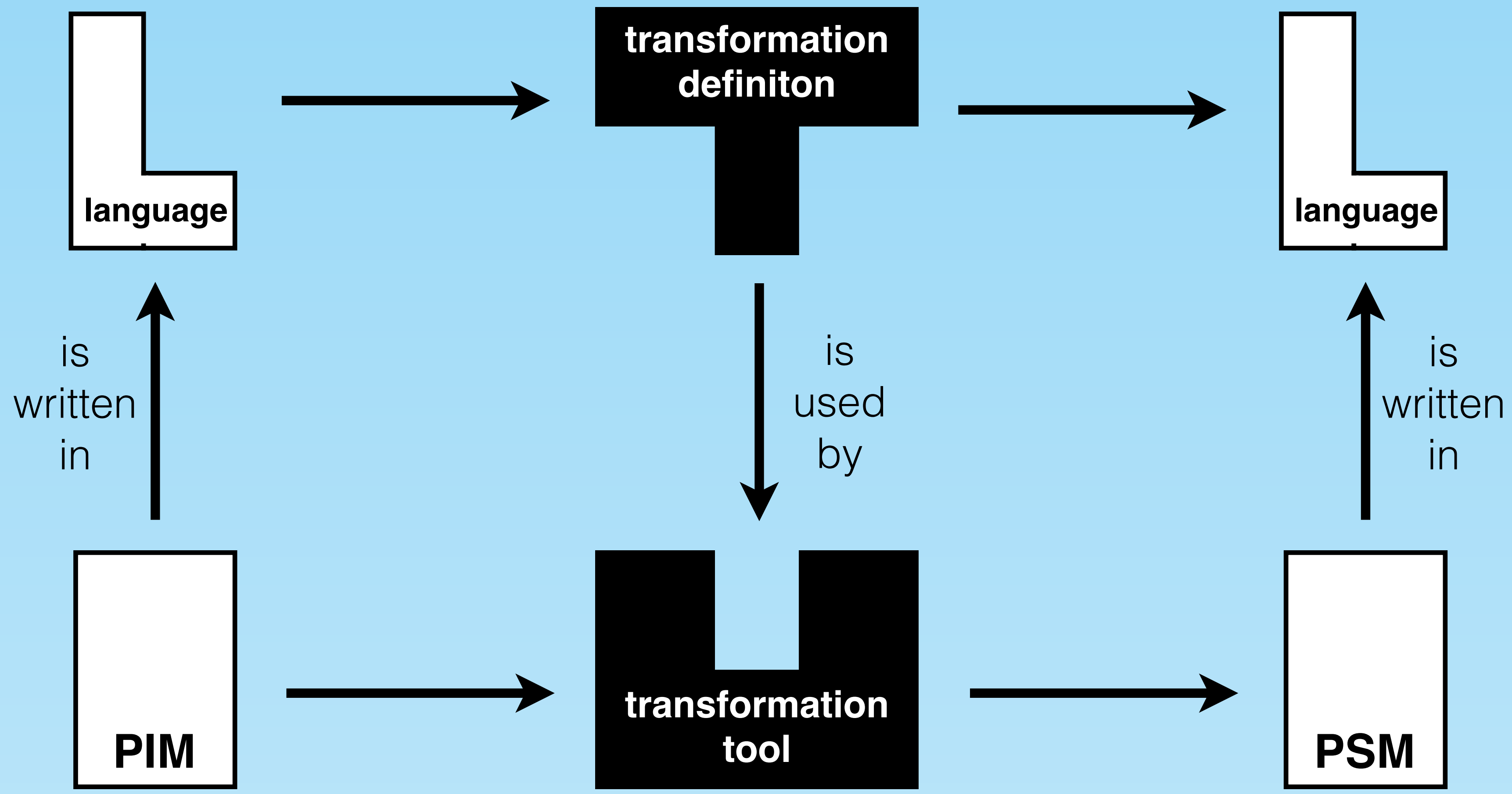
Outline

- Motivation
- Background
- Mapping OCL to SQL-PL
- Mapping OCL to MS-FOL
- Application domains:
  - checking model unsatisfiability
  - analysing security and privacy models
  - checking data invariants preservation across states

This research focused
on providing methods and tool support
which help building complex systems within
the Model Driven Architecture framework

MDA (Model Driven Architecture)

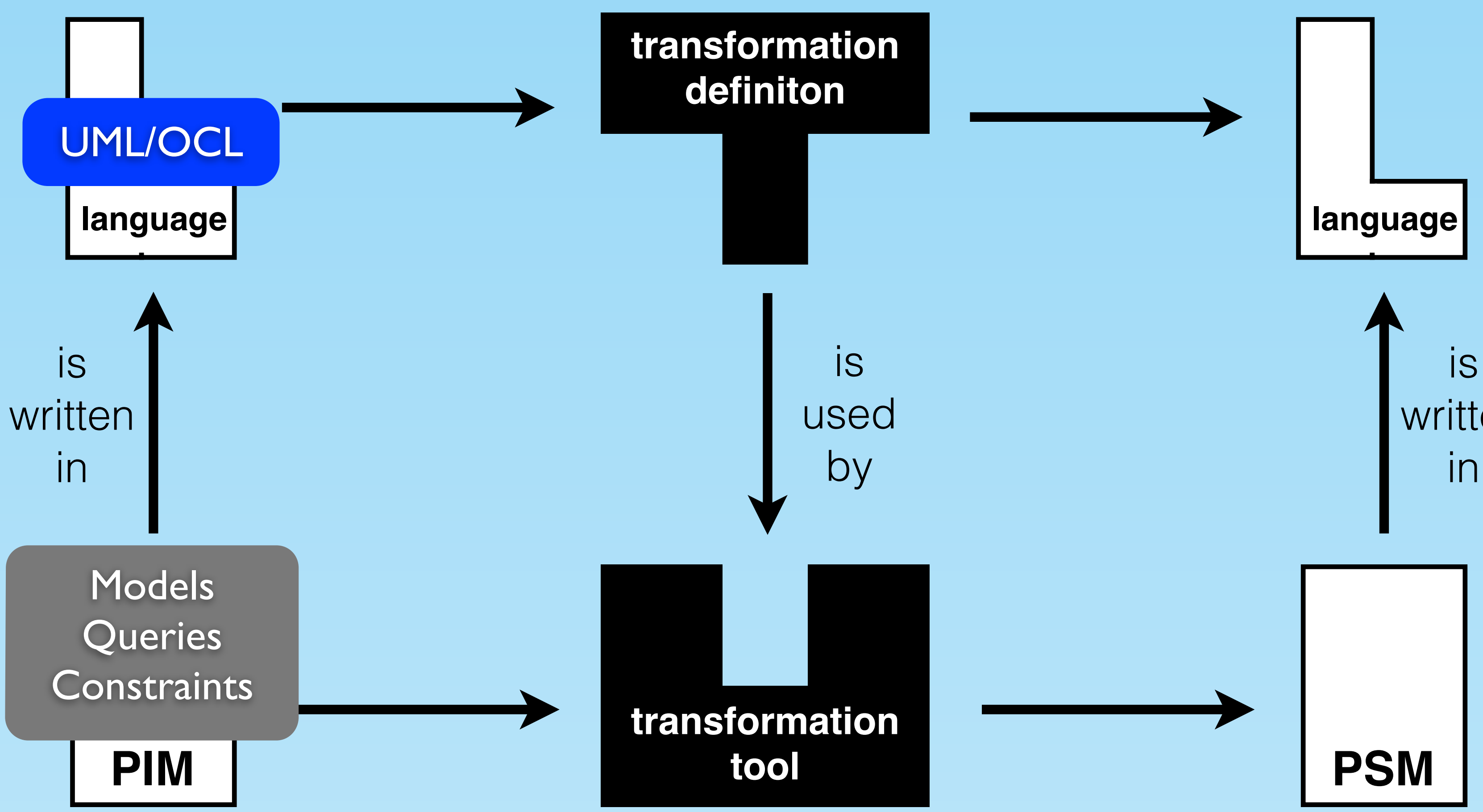
- It supports the development of complex systems by generating software from models.



PIM (Platform Independent Model), PSM (Platform Specific Model)

MDA (Model Driven Architecture)

- It supports the development of complex systems by generating software from models.



PIM (Platform Independent Model), PSM (Platform Specific Model)

Motivation
Why models?

We always create models

Motivation
Why models?

A model can be used in a different ways during the development process:

- for communication purposes to discuss design decisions.
- to provide a detailed specification of the system.
- to develop the system.

Motivation
Why UML?

- UML is the de-facto language for Object-Oriented analysis and design of information systems.
- UML is a standard of the Object Management Group (OMG) (1997), and it is also an ISO standard (2005).
- UML sustains many aspects of software engineering, but it does not provide enough level of precision.

Motivation
Why OCL?

OCL was born as a constraint language to add precision to UML like models an envolved as a query language.

It is a declarative language, and OMG and ISO standard.

Motivation

A variety of applications arises for OCL as a query language.

OCL as a constraint language helps to add precision to UML like models with detailed formal semantics.

OCL as a query language

The limitations of OCL as a query language can be solved by mapping it to the most commonly used query systems, i.e. databases

OCL as a constraint language

Our goal is provide a formal semantics that support automatic reasoning to a great extent so it can be used by software engineers.

Motivation

The quality of the generated code depends on the quality of the source models.

- About 90% of security software incidents are caused by known software defects.
- A study of 45 e-business applications showed that 70% of software failures are related to design.
- One million lines of code can have approximately between1000 and 5000 software defects in production.

We want to prevent, detect, and correct errors as early as possible.

Source: Team Software Process for Secure Systems Development. Software Engineering Institute. Carnegie Mellon

Motivation

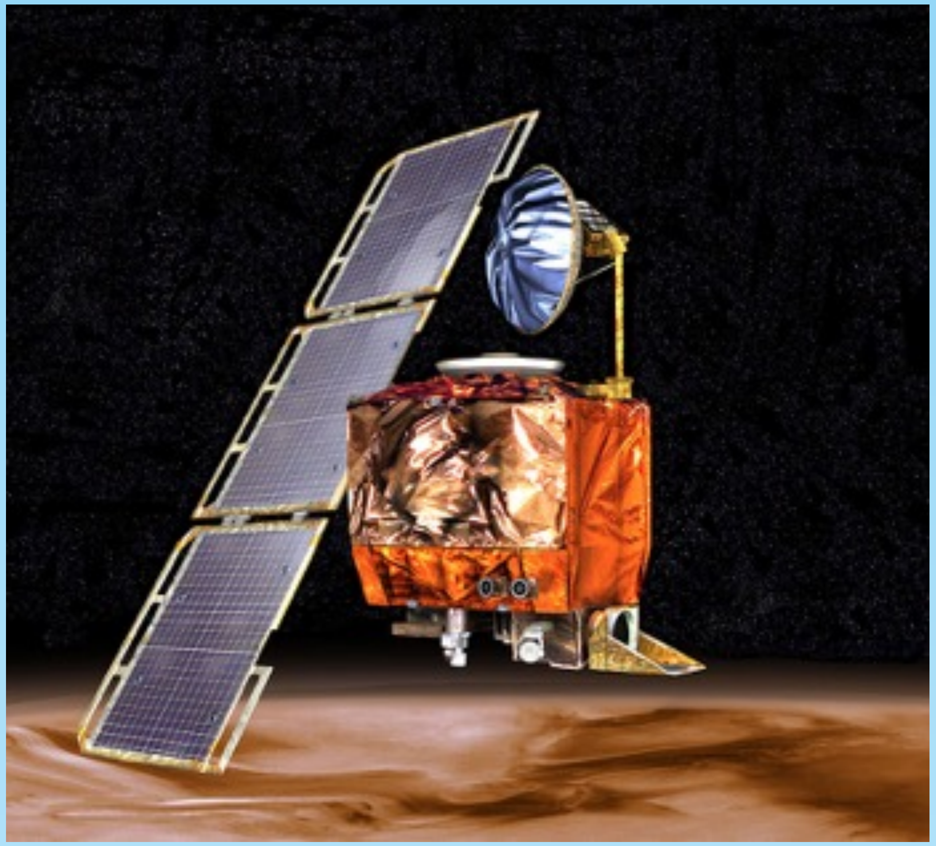
USS Yorktown, smartship

- Crew member entered 0 in a data field and cost a “divide by 0” error
- it down the propulsion
- ship was dead in the water for 2:45mins

Motivation

Mars Climate Orbiter (MCO)

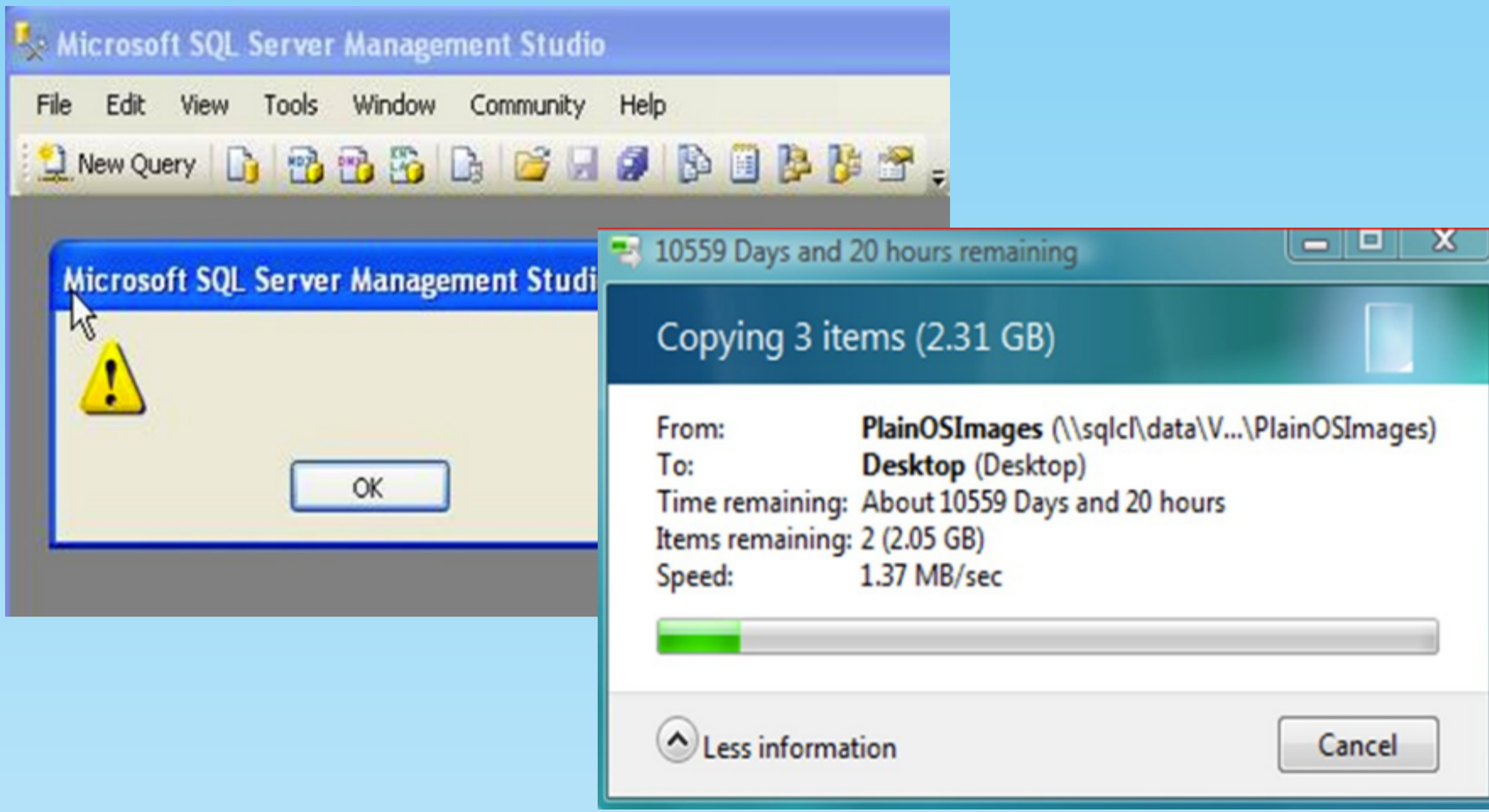
- NASA lost a $125 million
- Metric System Mixup (metrix vs imperial)



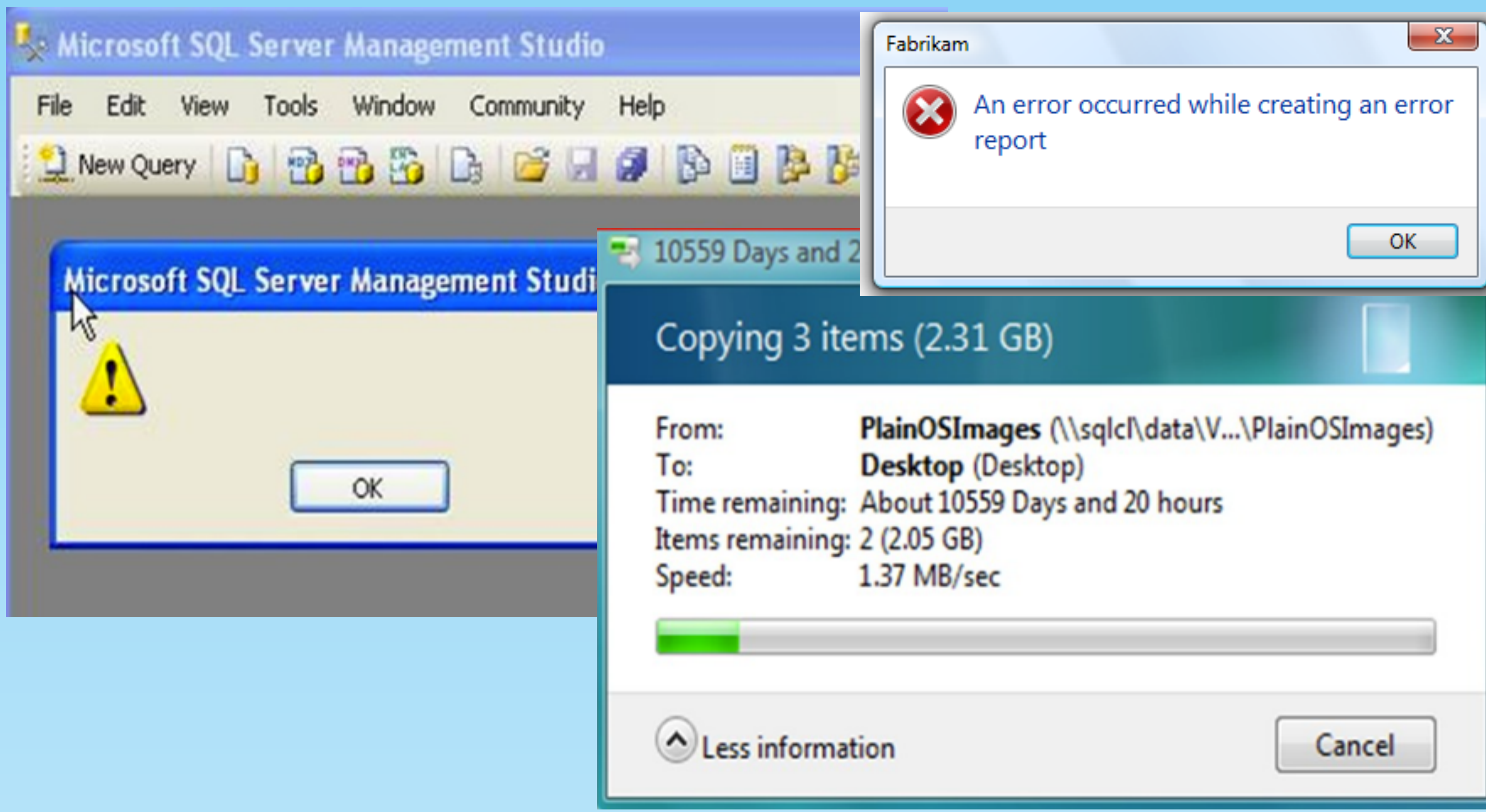
Motivation


Microsoft SQL Server Management Studio
File
Edit
View
Tools
Window
Community
Help
New Query
Microsoft SQL Server Management Studio
OK

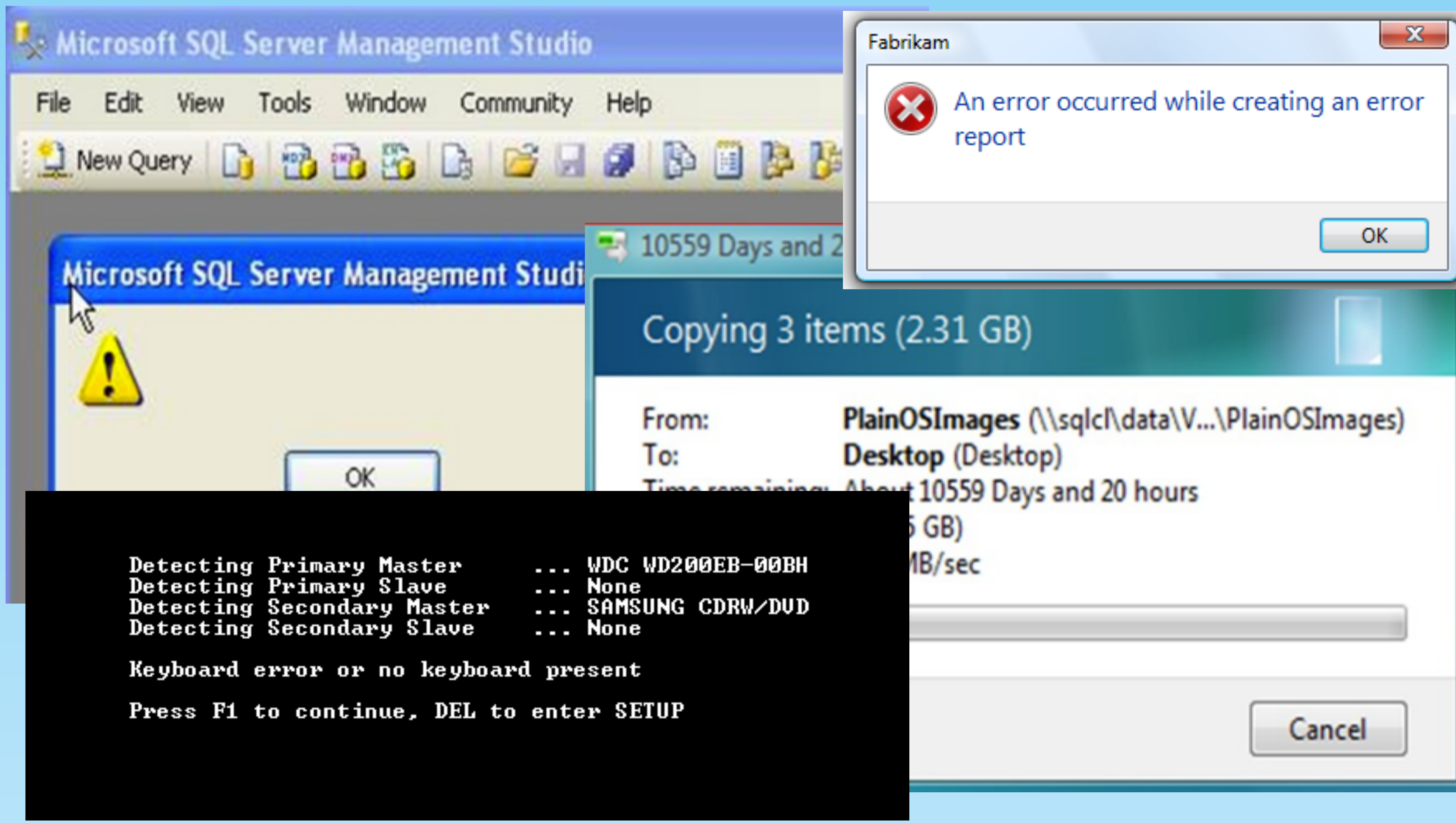
Motivation


Microsoft SQL Server Management Studio
File Edit View Tools Window Community Help
New Query
Microsoft SQL Server Management Studi
OK
10559 Days and 20 hours remaining
Copying 3 items (2.31 GB)
From: PlainOSImages (\\sqlcl\data\V...\PlainOSImages)
To: Desktop (Desktop)
Time remaining: About 10559 Days and 20 hours
Items remaining: 2 (2.05 GB)
Speed: 1.37 MB/sec
Less information
Cancel

Motivation


Microsoft SQL Server Management Studio
File Edit View Tools Window Community Help
New Query
Microsoft SQL Server Management Studi
OK
Fabrikam
An error occurred while creating an error report
OK
10559 Days and 2
Copying 3 items (2.31 GB)
From: PlainOSImages (\\sqlcl\data\V...\PlainOSImages)
To: Desktop (Desktop)
Time remaining: About 10559 Days and 20 hours
Items remaining: 2 (2.05 GB)
Speed: 1.37 MB/sec
Less information
Cancel

Motivation


Microsoft SQL Server Management Studio
File Edit View Tools Window Community Help
New Query
Microsoft SQL Server Management Studi
OK
10559 Days and 2
Copying 3 items (2.31 GB)
From: PlainOSImages (\\sqlcl\data\V...\PlainOSImages)
To: Desktop (Desktop)
About 10559 Days and 20 hours
Cancel
Fabrikam
An error occurred while creating an error report
OK
Detecting Primary Master ... WDC WD200EB-00BH
Detecting Primary Slave ... None
Detecting Secondary Master ... SAMSUNG CDRW/DVD
Detecting Secondary Slave ... None
Keyboard error or no keyboard present
Press F1 to continue, DEL to enter SETUP

Motivation

This doctoral dissertation aims

to help the current status of methodology and tools for building complex software systems

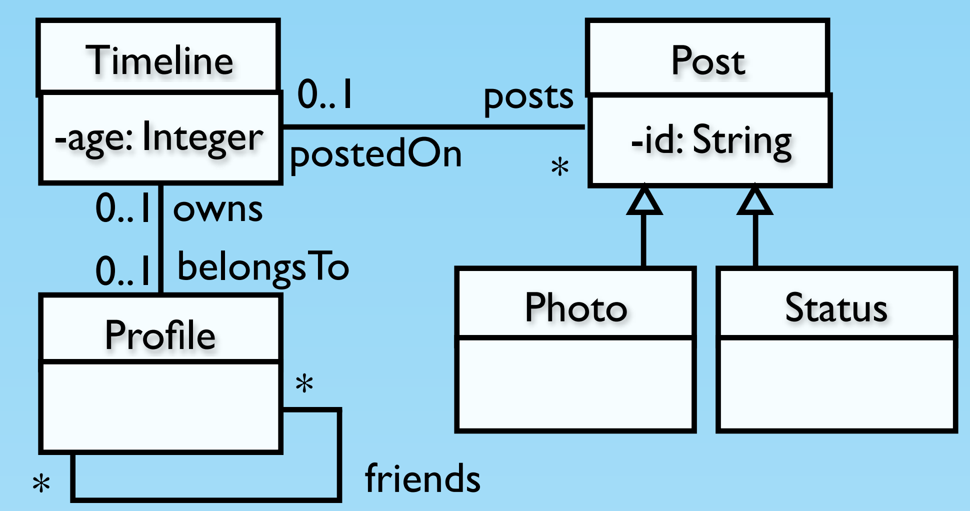
Background

UML (Unified Modeling Language)

Ex. Social Network

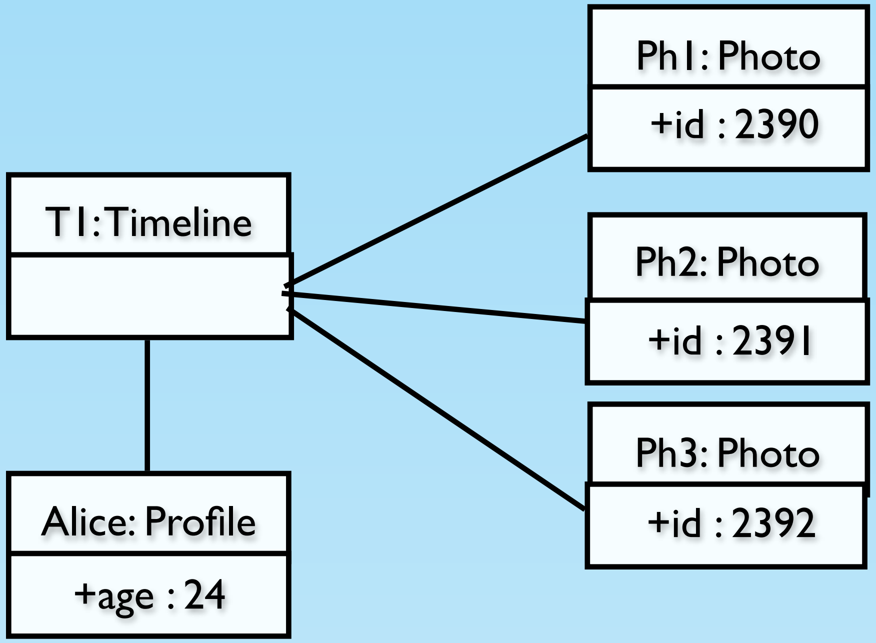
Class diagram

- classes
- attributes
- associations (association-ends)
- inheritance



Object diagram

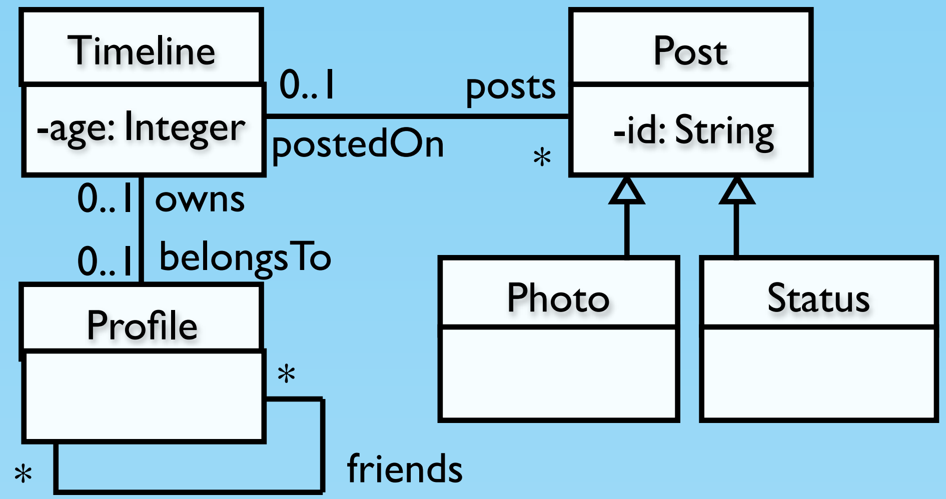
- objects
- values
- links



OCL (Object Constraint Language)

- It is a general-purpose (textual) formal language that allows:
  - retrieve objects and their values
  - navigate through related objects
- It supports a set of types with a set of operations over them, and
  - primitive types (Integer, String, Boolean), and
  - collection types (Set, Bag, OrderedSet, and Sequence), and
  - operators like: +, -, >, <, size, isEmpty, notEmpty, characters, and
  - iterators like: forAll, exists, collect

OCL (Object Constraint Language)



- All instances of Timeline

  Timeline.allInstances()

- Number of instances

  Timeline.allInstances()–>size()

- Every profile is older than 18 years old

  Profile.allInstances()–>forAll(p|p.age > 18)

- There isn't any profile older than 18

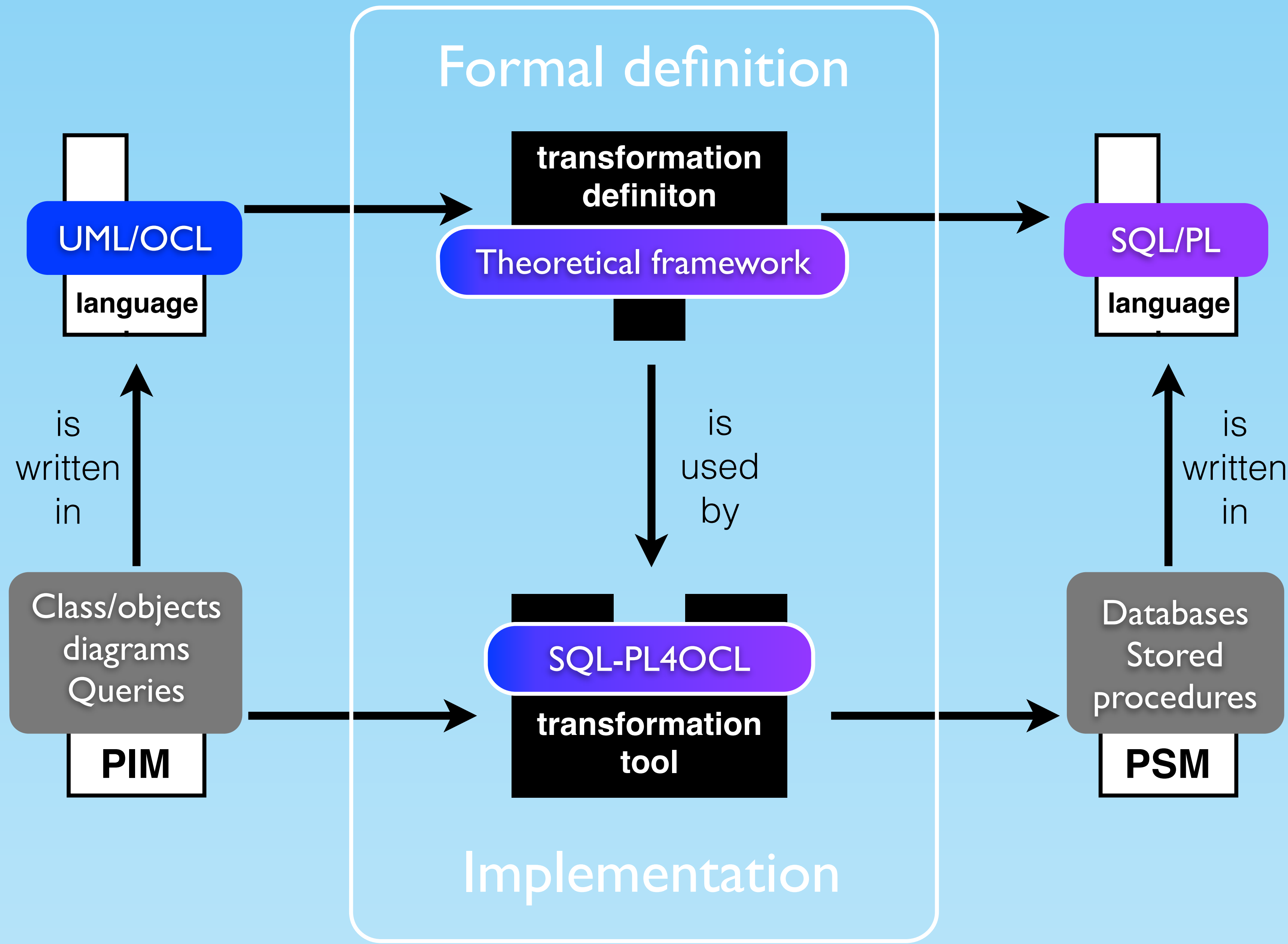
  Profile.allInstances()–>select(p|p.age > 18)–>isEmpty()

- Convert the string 'hi' in a sequence of characters

  'hi'.characters()

Mapping OCL to SQL-PL

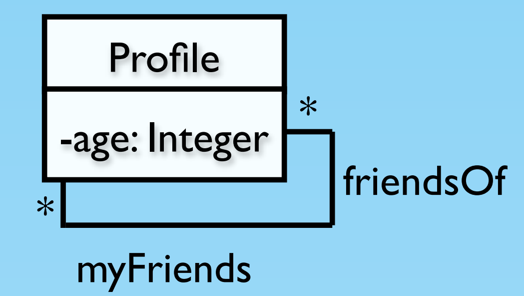
Mapping OCL to SQL-PL



M. Egea, C. Dania, M. Clavel: MySQL4OCL: A Stored Procedure-Based MySQL Code Generator for OCL. ECEASST 36 (2010).
M. Egea, C. Dania. SQL-PL4OCL: an automatic code generator from OCL to SQL procedural language. Software & Systems Modeling, 2017, p. 1-23.

From OCL to SQL-PL

Mapping data/object models.



Data model

- a table with a column for each class
- a column for each attribute
- a table with two columns for each association

Object model

- a row for each object in the table associated with the class
- a row for each link in the corresponding table

table: Profile

pk	age
1	18
2	10

table: friendship

myFriends	friendsOf
1	2

From OCL to SQL-PL
Mapping OCL expressions

Every expression is mapped into a stored procedure

```
create procedure name
begin

    OCL to SQL-PL expression

end;//
call name()//
```

Depending on the complexity of the OCL expressions, they are mapped:

- into a SQL query
- into a SQL query and need an auxiliary block definition

From OCL to SQL-PL
Mapping OCL expressions (cont.)

- Expressions that are mapping into a SQL query

Timeline.allInstances()

```
select Timeline.pk as val
from Timeline
```

```
create procedure name
begin

;

end;//
call name();//
```

From OCL to SQL-PL
Mapping OCL expressions (cont.)

- Expressions that are mapping into a SQL query

Timeline.allInstances()

```
create procedure name
begin


    select Timeline.pk as val
    from Timeline                ;
end;//
call name();/ /
```

From OCL to SQL-PL
Mapping OCL expressions (cont.)

- Expressions that are mapping into a SQL query

Timeline.allInstances()

select Timeline.pk as val
from Timeline

Timeline.allInstances()–>size()

select count(t1.val) as val
from
(
) as t1

create procedure name
begin

;

end;//
call name();/ /

From OCL to SQL-PL
Mapping OCL expressions (cont.)

- Expressions that are mapping into a SQL query

Timeline.allInstances()

```
create procedure name
begin
```

Timeline.allInstances()−>size()

```
select count(t1.val) as val
from
  ( select Timeline.pk as val
    from Timeline ) as t1
```

```
;
end;//
call name(); / /
```

From OCL to SQL-PL
Mapping OCL expressions (cont.)

- Expressions that are mapping into a SQL query

Timeline.allInstances()

Timeline.allInstances()->size()

```
create procedure name
begin
  select count(t1.val) as val
  from
   (select Timeline.pk as val
    from Timeline  ) as t1      ;
end;//
call name();/ /
```

From OCL to SQL-PL

Mapping OCL expressions (cont.)

- Expressions that are mapped into a SQL query and need an auxiliary block definition

pos	val
1	h
2	i

’hi’.characters()

```
create procedure name
begin
   begin
      drop table if exists wchars;
      create temporary table wchars (pos int not null auto increment,
         val varchar(250), primary key(pos));
      insert into wchars(val) (select ’h’ as val);
      insert into wchars(val) (select ’i’ as val);
   end;
   select val from wchars order by pos;
end;//
```

From OCL to SQL-PL

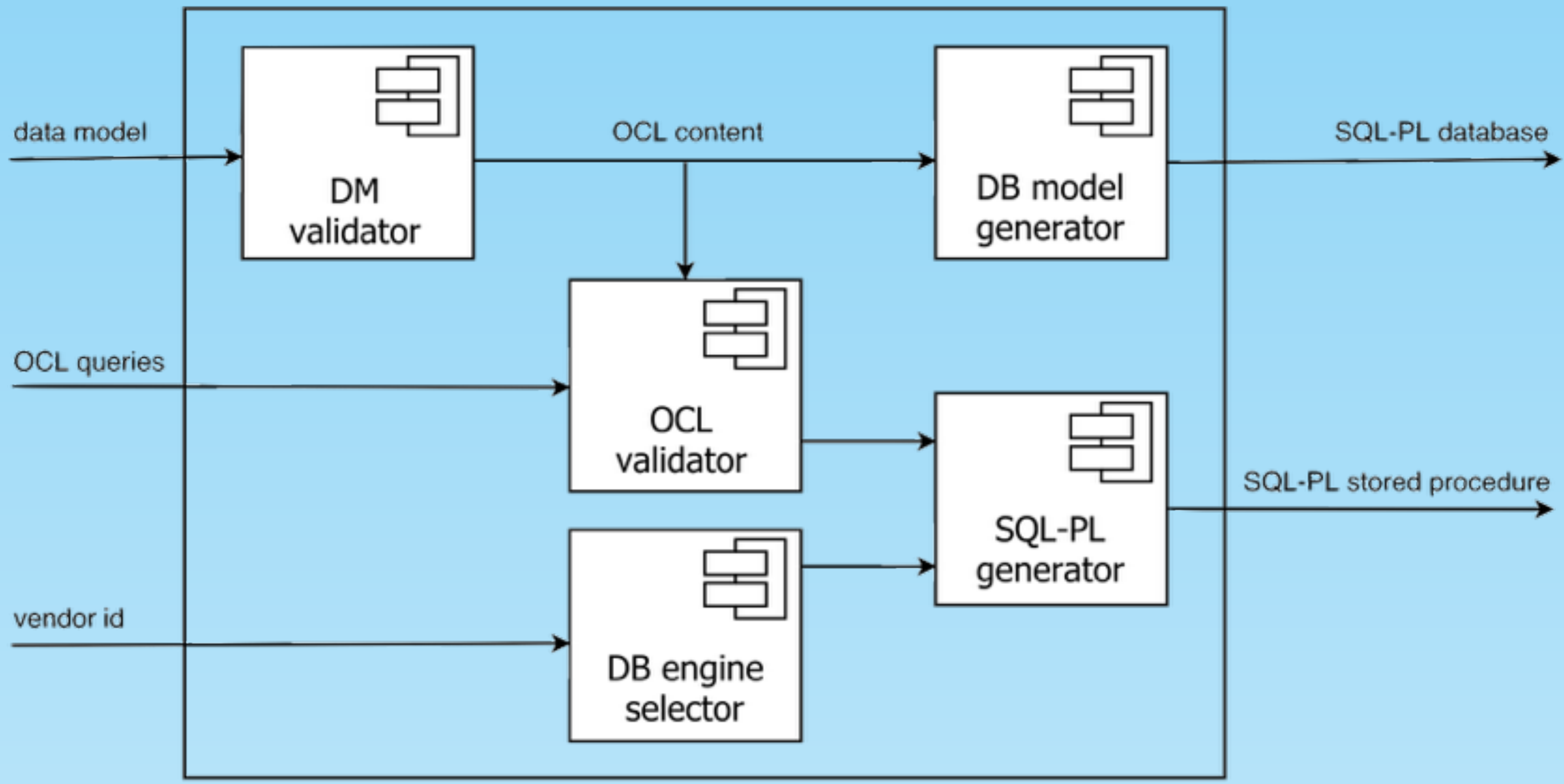
Iterators

src—>it(body)

```
begin
  declare done int default 0;
  declare var;
  declare crs cursor for (cursor-specific type - src );
  declare continue handler for sqlstate '02000' set done = 1;
  drop table if exists blq_name;
  create temporary table blq_name (value-specif type )
  open crs;
  repeat
    fetch crs into var;
    Iterator-specific body query
    if not done then
      Iterator-specific processing code
    end if;
  until done end repeat;
  close crs;
end;
```

SQL-PL4OCL

tool component architecture



SQL-PL4OCL
Benchmark

- Vendor specific supported: MySQL/MariaDB, PostgreSQL, SQL Server DBMS
- MariaBD works faster in most of the cases

	MySQL	MariaDB	PostgreSQL	MSSQL
Q1	0.19s	0.13s	**0.10s**	0.12s
Q2	0.25s	**0.20s**	0.33s	0.28s
Q3	0.36s	0.35s	0.27s	**0.26s**
Q4	**0.04s**	**0.04s**	**0.04s**	0.05s
Q5	0.55s	**0.40s**	0.40s	0.42s
Q6	1.05s	**0.55s**	1.06s	1.03s
Q7	2.07s	**1.56s**	1.99s	2.08s
Q8	50.02s	**43.08s**	57.04s	53.47s
Q9	9.14s	**8.00s**	8.18s	8.89s
Q10	0.05s	**0.04s**	0.07s	0.05s
Q11	49.56s	**40.02s**	40.10s	43.46s
Q12	59.58s	**51.23s**	51.25s	54.82s
Q13	**1.67s**	1.98s	2.35s	1.90s
Q14	59.52s	**54.33s**	63.35s	58.33s

Related work
(comparison with OCL2SQL-DresdenOCL)

OCL pattern

context: Class

inv: OCL boolean expression

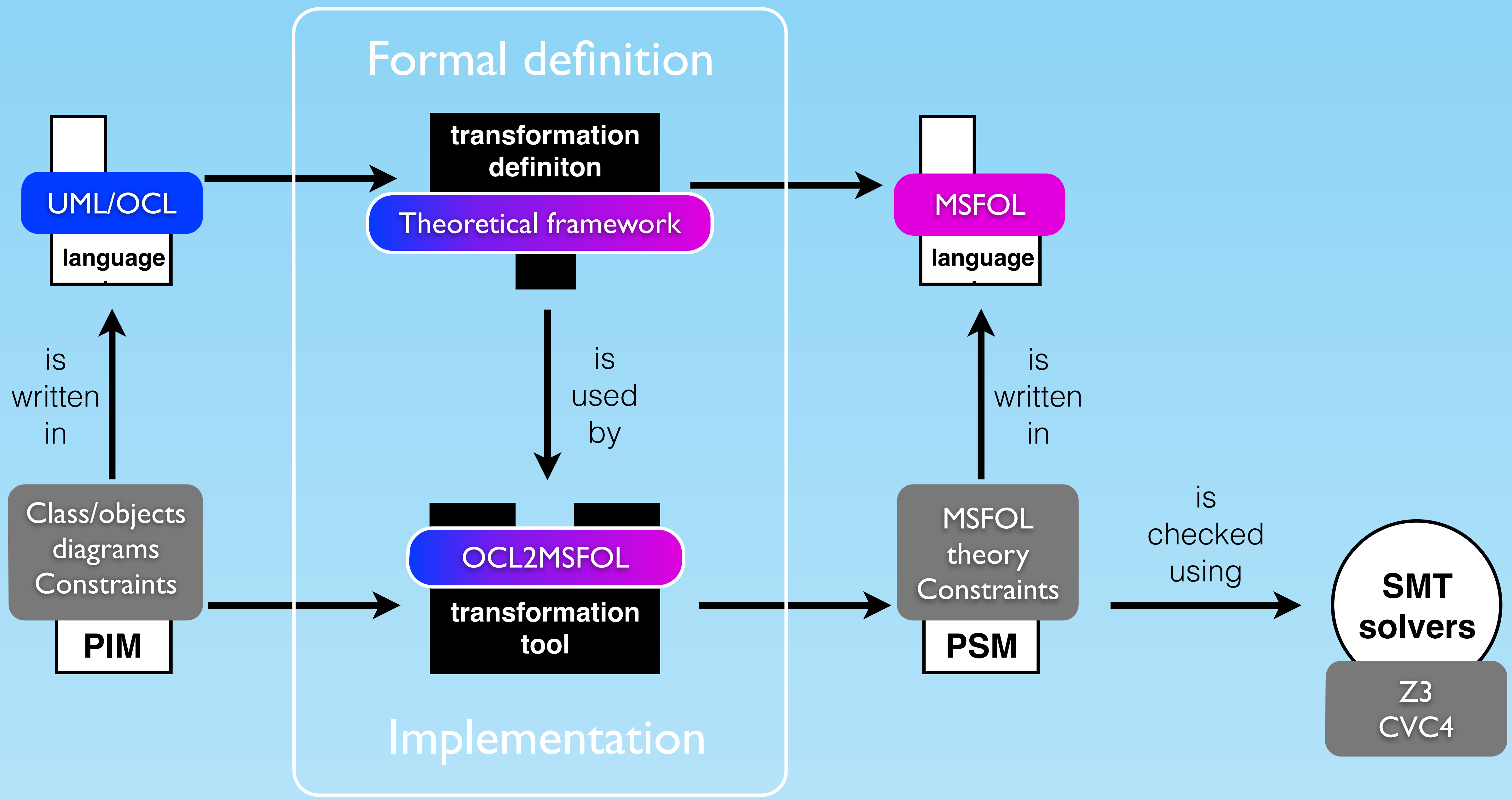
MySQL pattern

select *
from Class
where not OCL2SQL(OCL boolean expression)

OCL2SQL mapping is based on patterns and it does not support iterators.

Mapping OCL to MSFOL

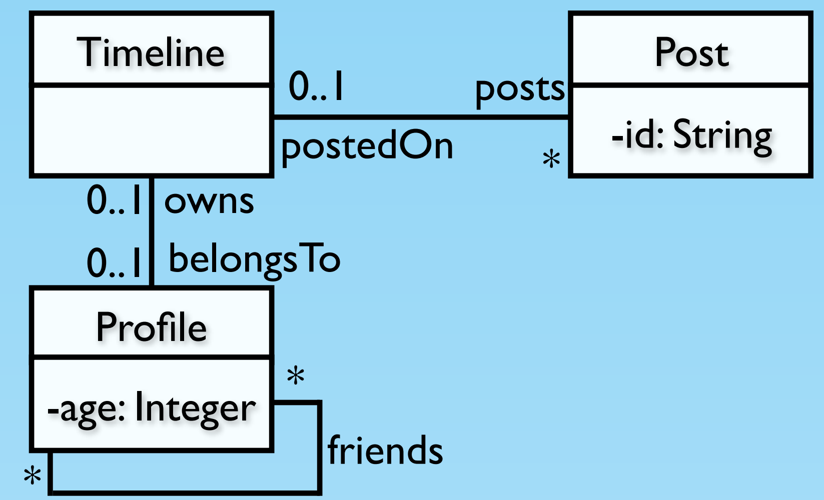
Mapping OCL to MSFOL



C. Dania, M. Clavel: OCL2FOL+: Coping with Undefinedness. OCL@MoDELS 2013: 53-62
C. Dania, M. Clavel. OCL2MSFOL: a mapping to many-sorted first-order logic for efficiently checking the satisfiability of OCL constraints. MoDELS 2016: 65-75

From OCL to MSFOL
Mapping data models

- sorts: Int, $String$ and $Classifier$.
  (null and invalid for each sort)



- a predicate for each class.
  $Timeline : Classifier \rightarrow Bool$
- a function for each attribute.
  $age : Classifier \rightarrow Int$
- one/two function(s)/predicate(s) for each association.
  $friends : Classifier \times Classifier \rightarrow Bool$

+ Set of axioms:

$$\forall (x : Classifier)(Profile(x) \Rightarrow \neg(Timeline(x) \vee \ldots \vee Post(x)))$$

$$\neg(Profile(nullClassifier) \vee Profile(invalClassifier))$$

From OCL to MSFOL
Mapping OCL expressions

- (Sub-)expressions of type **Boolean (Integer)** are translated into formulas (terms)
  - not, and, or, implies, =, >, <, forAll, exists, one, isEmpty, notEmpty, includes, excludes, +, -, ….

Profile.allInstances()−>forAll(p|p.age > 18)

$$\forall(x : Classifier)(Profile(x) \wedge (age(x) > 18 \wedge \neg(nullInt = age(x) \vee invalInt = age(x))))$$

Axiom: $\neg(nullInt = 18 \vee invalInt = 18)$

From OCL to MSFOL
Mapping OCL expressions

- (Sub-)expressions of type **Set** (or **Primitive types** that require definition) are translated into predicates formulas (functions), whose (fresh) predicate (function) symbols satisfy the corresponding axioms (also generated by the mapping)
  - select, reject, including, excluding, collect (follow by asSet),
  - any, max, min

Profile.allInstances()->select(p|p.age > 18)->isEmpty()

Select

$$\mathbf{Select} : Classifier \rightarrow Bool$$

$$\forall(x : \mathit{Classifier})(\mathbf{Select}(x) \Leftrightarrow (\mathit{Profile}(x) \wedge (age(x) > 18 \wedge \neg(\mathit{nullInt} = age(x) \vee \mathit{invalInt} = age(x)))))$$

$$\forall(x : \mathit{Classifier})(\neg\mathbf{Select}(x))$$

Checking unsatisfiability

Data model $\mathcal{D}$. Set of $\mathcal{D}$-constraints $\mathcal{I}$. A Boolean OCL expression ***expr***

Then, ***expr*** evaluates to true in every valid instance of $\mathcal{D}$ if and only if :

$$\mathrm{o2f}_{\mathrm{data}}(\mathcal{D}) \cup \left(\bigcup_{inv \in \mathcal{I}} \mathrm{o2f}_{\mathrm{def}}(inv) \right) \cup \left(\bigcup_{inv \in \mathcal{I}} \{\mathrm{o2f}_{\mathrm{true}}(inv)\} \right) \cup \mathrm{o2f}_{\mathrm{def}}(expr) \cup \{\mathrm{o2f}_{\mathrm{false}}(expr)\}.$$

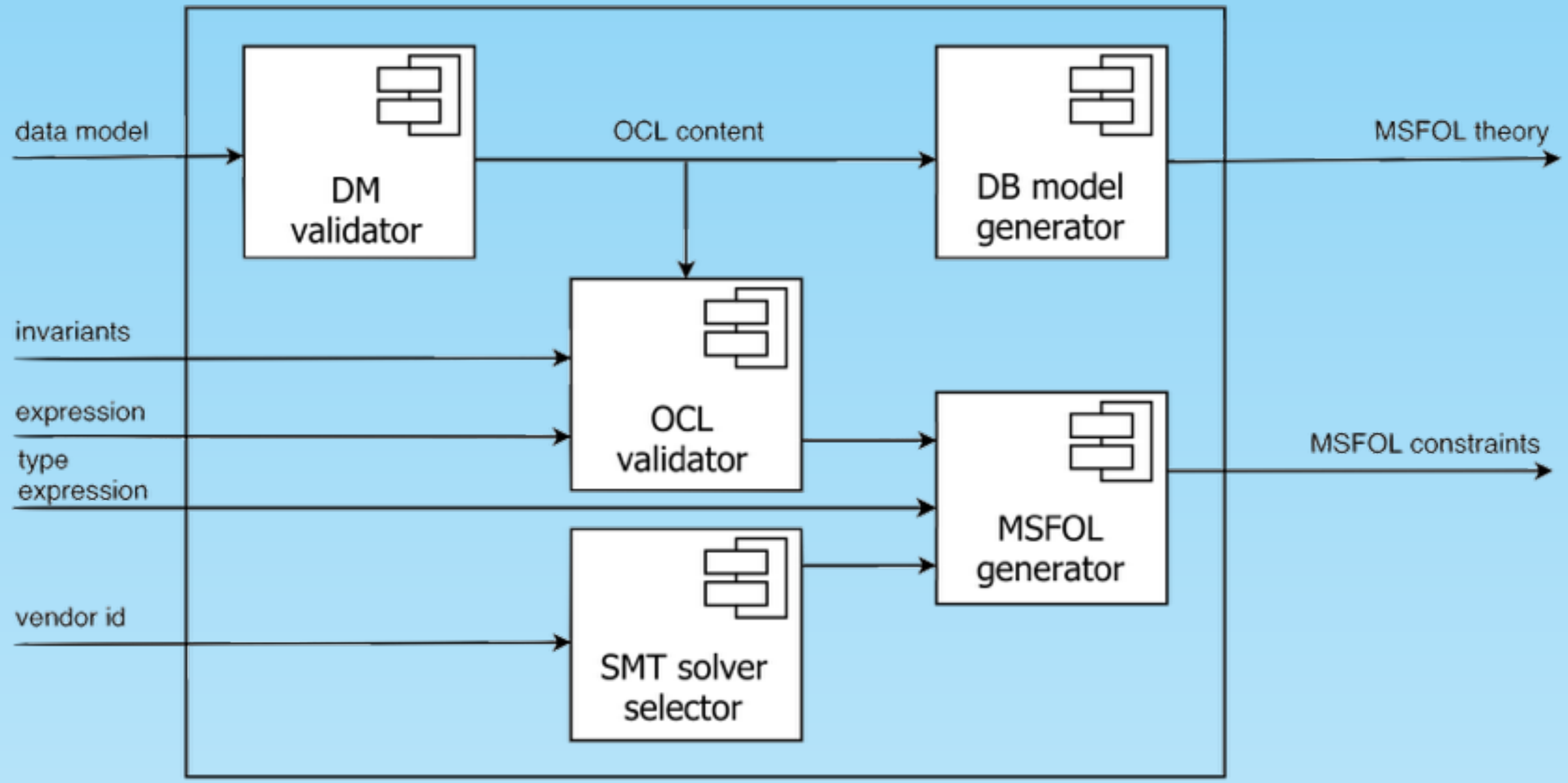
is unsatisfiable.

Satisfiability Module theories (SMT) solvers

We can expect: **sat** (there exists at least one valid instance of the model),
unsat (no valid instance of the model exists),
unknown (check is inconclusive).

SMT solvers cannot be complete when dealing with quantifiers (undecidability)

OCL2MSFOL tool component architecture



OCL2MSFOL

Benchmark

		CVC4	Z3	CVC4fm
{1,2}	unsat	161	24	48
{1,3}	unsat	173	13	22
{2,3}	sat	—	16	25
{4}	unsat	138	15	27
{5}	sat	—	17	22
{5,6}	unsat	172	13	30
{1,7}	unsat	237	14	30
{1,8}	sat	—	18	25
{1,6,8}	unsat	198	16	26
{1,9}	sat	—	18	25
{1,6,9}	unsat	200	19	29
{1,10}	unsat	203	18	30
{12}	sat	—	169	27
{11,12,13}	sat	—	24	174

Undefinedness-related (times in ms)

		CVC4	Z3	CVC4fm
{14,20}	sat	—	105	28
{16,20}	sat	—	466	32
{17,20}	sat	—	14	22
{14,17,20}	unsat	239	13	26
{16,19}	unsat	168	16	28
{21}	sat	—	17	27
{22}	sat	—	199	24
{16,22}	unsat	149	18	25
{16,23}	unsat	148	16	26
{15,17,18,24}	unsat	250	15	35
{25}	unsat	63	58	24
{11,12,13,18}	sat	—	—	27
{6,27}	sat		—	26
{11,12,13,18,26}	unsat	352	13	25

Generalization-related (times in ms)

Related work
Other mappings from UML/OCL to other formalisms

	Mapping	Target formalism
G1 (do not support OCL constraints)	FiniteSAT	System of Linear Inequalities
	DL	Description Logics, CSP
	MathForm	Mathematical Notation
G2 (support OCL constraints)	UMLtoCSP	CSP
	EMFtoCSP	CSP
	AuRUS	FOL
	OCL2FOL	FOL
	OCL-Lite	Description Logics
	BV-SAT	Relation Logic
	PVS	HOL
	CDOCL-HOL	HOL
	KeY	Dynamic Logic
	Object-Z	Object-Z
	UML-B	B
G3 (support OCL constraints and OCL null)	UML2Alloy	Relation Logic
	USE	Relation Logic
G4 (support OCL constraints and OCL null and invalid)	HOL-OCL	HOL
	OCL2FOL+	FOL

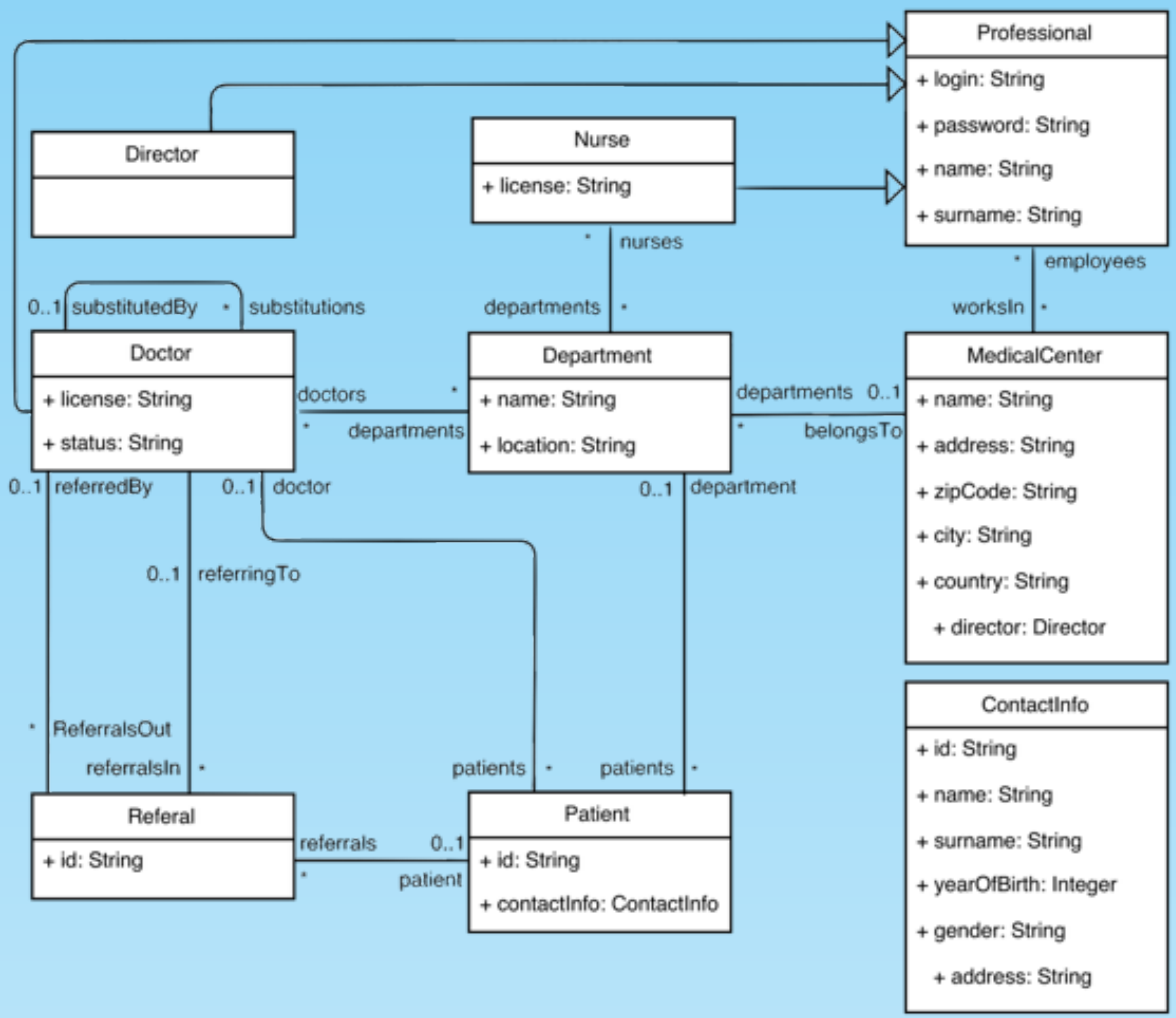
Application domains

Checking model satisfiability

Case study: eHealth Record Management System

Data models

- 9 classes
- 3 generalisations
- 24 attributes
- 10 associations



M.A. García de Dios, C. Dania, D. Basin, M. Clavel: Model-Driven Development of a Secure eHealth Application. Engineering Secure Future Internet Services and Systems 2014: 97-118

Checking model satisfiability
Case study: eHealth Record Management System

- 38 invariants

There must be at least one medical center
MedicalCenter.allInstances()->notEmpty()

Every medical center should have at least one employee.
MedicalCenter.allInstances()->forAll(m|m.employees->notEmpty())

Each patient is treated by a doctor who works in the department where the patient is treated.
Patient.allInstances()->forAll(p|
p.doctor.departments->exists(d|d=p.department))

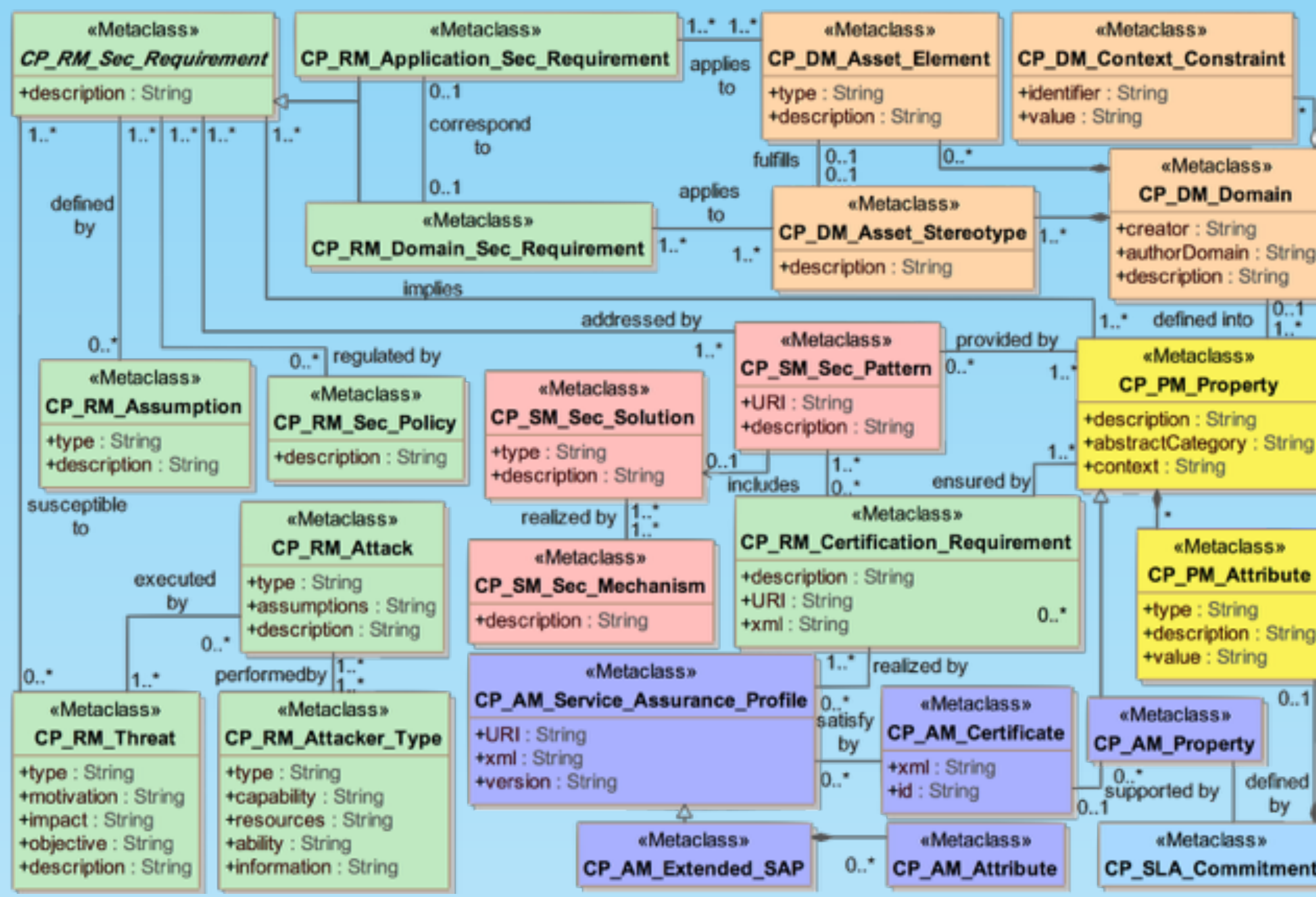
1. CVC4 Finite Model returns **sat** in 7 seconds.
2. If we add 1 more constraint.
   CVC4 Finite Model returns **unsat** in 4 seconds.

Validating and instantiating models
A Security Metamodel

Data models

- 24 classes
- 3 generalisations
- 47 attributes
- 22 associations

33 invariants

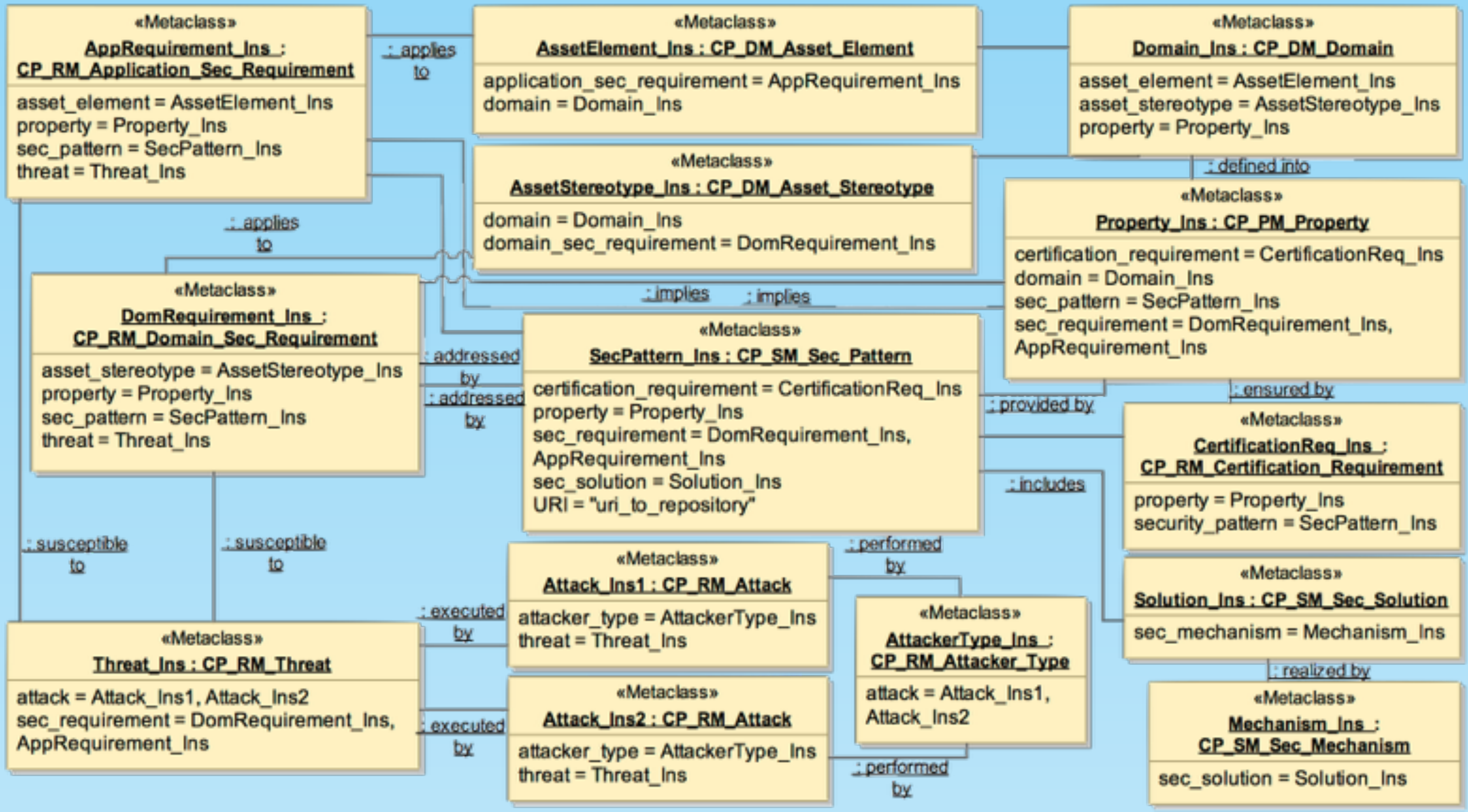


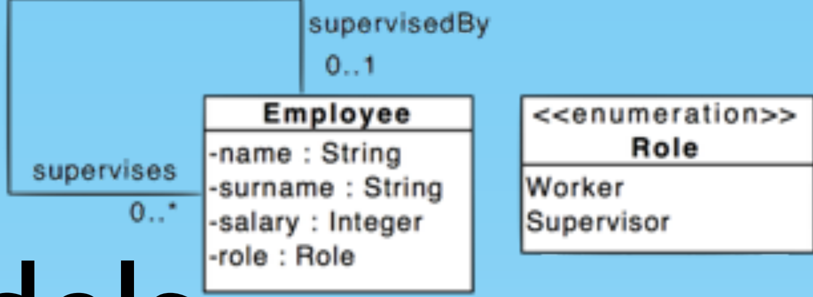
M. Arjona, C. Dania, M. Egea, A. Maña, Validation of a Security metamodel for Development of Cloud Applications. OCL@MoDELS 2014: 33-42

Validating and instantiating metamodels

A Security Metamodel

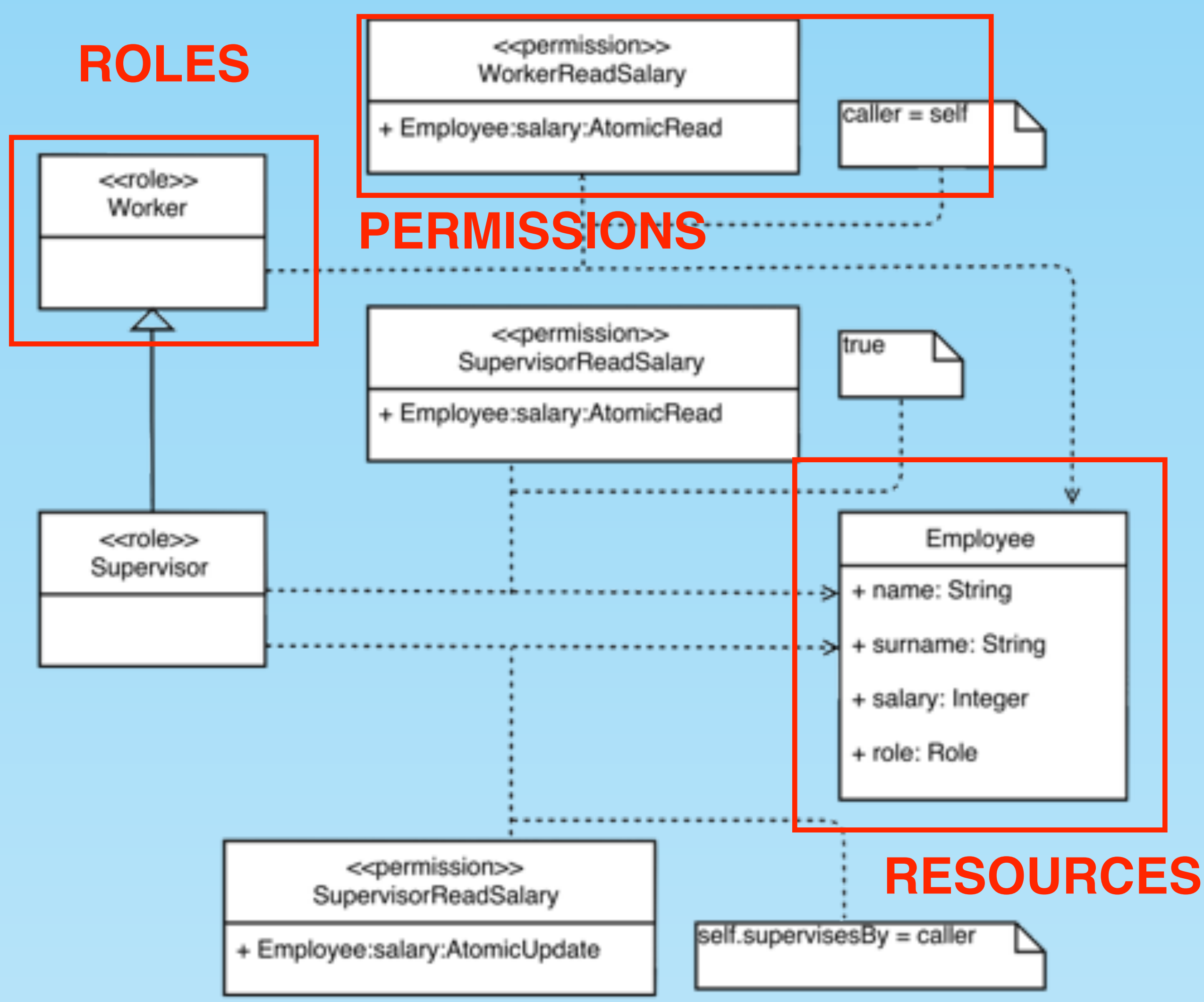
CVC4 Finite Model returns **sat** + one instance.





Analysing security models

- SecureUML is a modeling language for specifying fine-grained access control policies for actions on protected resources.



Auth(**Worker,** update(salary)) =
false

Auth(**Supervisor**, update(salary)) =
self.supervisedBy = caller or false

Auth(**Worker**, read(salary)) =
caller = self

Auth(**Supervisor**, read(salary)) =
caller = self or true

M.A. García de Dios, C. Dania, M. Clavel: Formal Reasoning about Fine-Grained Access Control Policies. APCCM 2015: 91-100

Analysing security models

Auth(**Worker,** update(salary)= **false**

Auth(**Supervisor**, update(salary) =
self.supervisedBy = caller or false

Auth(**Worker**, read(salary)) = **caller = self**

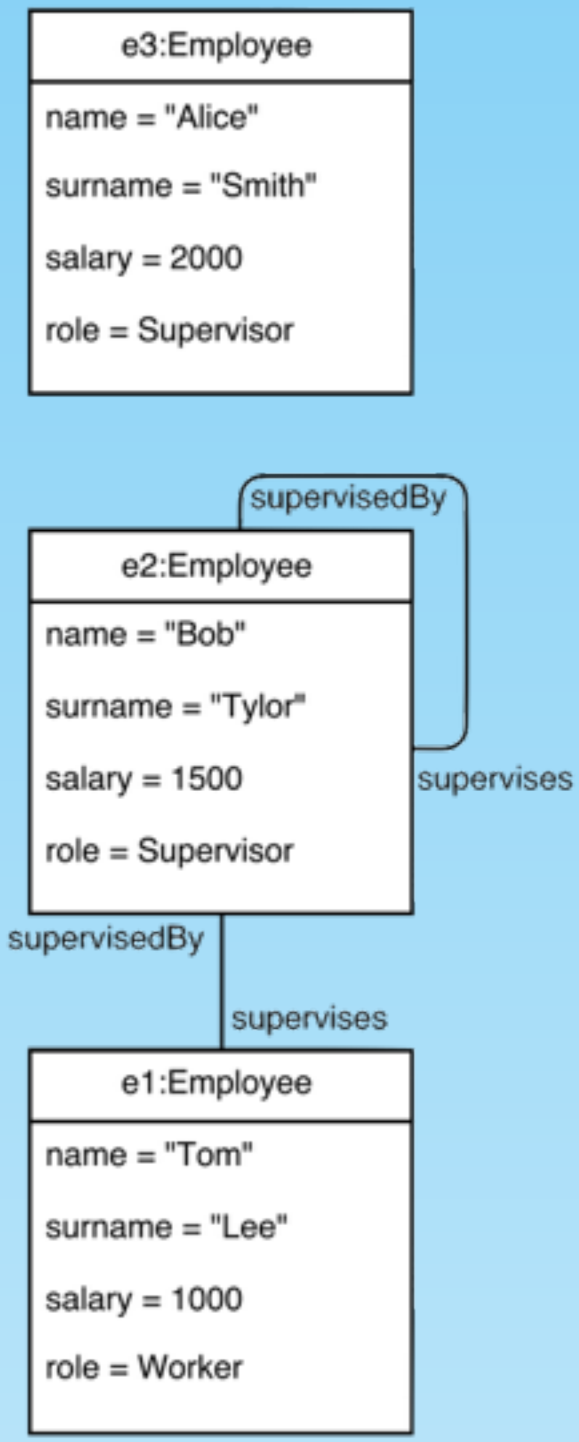
Auth(**Supervisor**, read(salary) = **caller = self or true**

Can Bob read Alice's salary?

Data model $\mathcal{D}$. SecureUML model $\mathcal{S}$.

A role r. An action act.

$$\text{o2f}_{\text{data}}(\mathcal{D}) \cup \{\exists(caller)\exists(self)\ (\text{o2f}_{\text{true}}(caller.\text{role} = r) \wedge \text{o2f}_{\text{true}}(\text{Auth}(\mathcal{S}, r, act)))\}$$



Analysing security models

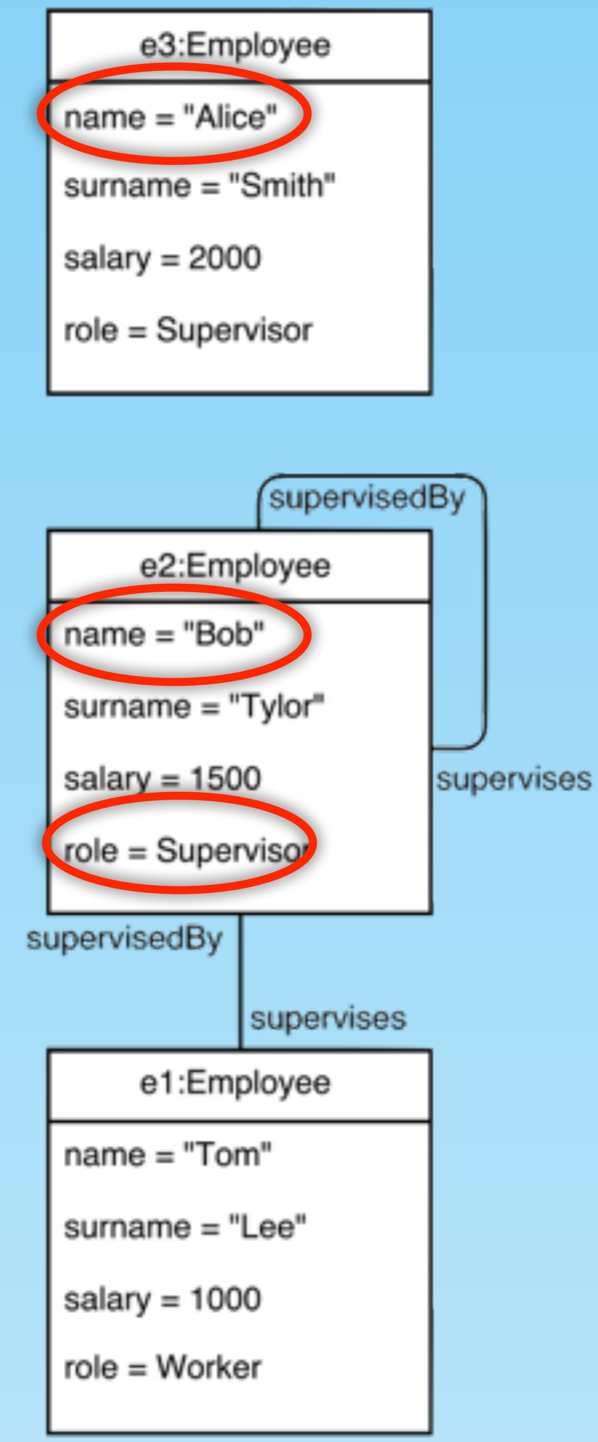
Auth(**Worker,** update(salary)= **false**

Auth(**Supervisor**, update(salary) =
self.supervisedBy = caller or false

Auth(**Worker**, read(salary)) = **caller = self**

Auth(**Supervisor**, read(salary) = **caller = self or true**

Can Bob read Alice's salary? ✓



Data model $\mathcal{D}$. SecureUML model $\mathcal{S}$.

A role r. An action *act*.

$$\mathrm{o2f}_{\mathrm{data}}(\mathcal{D}) \cup \{\exists(caller)\exists(self)\ (\mathrm{o2f}_{\mathrm{true}}(caller.\mathrm{role} = r) \wedge \mathrm{o2f}_{\mathrm{true}}(\mathrm{Auth}(\mathcal{S}, r, act)))\}$$

Analysing security models

Auth(**Worker,** update(salary)= **false**

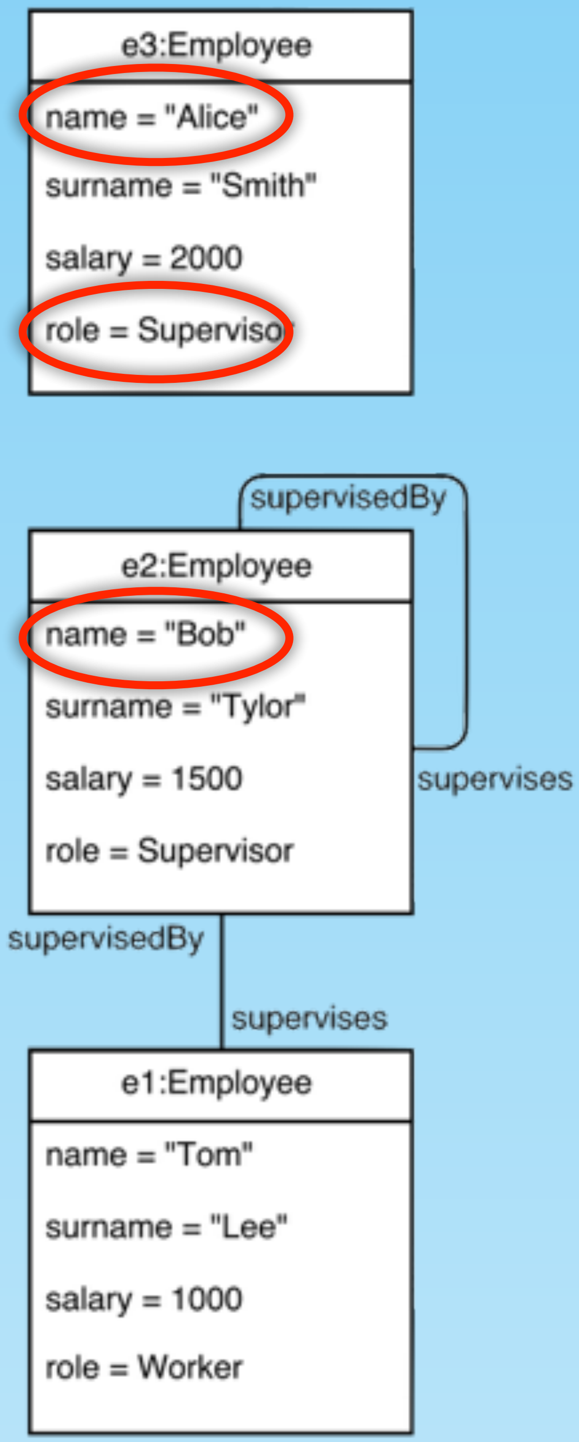
Auth(**Supervisor**, update(salary) =
self.supervisedBy = caller or false

Auth(**Worker**, read(salary)) = **caller = self**

Auth(**Supervisor**, read(salary) = **caller = self or true**

Can Bob read Alice's salary? ✓

Can Alice update Bob's salary? ✗



Data model $\mathcal{D}$. SecureUML model $\mathcal{S}$.

A role r. An action act.

$$\text{o2f}_{\text{data}}(\mathcal{D}) \cup \{\exists(caller)\exists(self)(\text{o2f}_{\text{true}}(caller.\text{role} = r) \land \text{o2f}_{\text{true}}(\text{Auth}(\mathcal{S}, r, act)))\}$$

Related work
Security models

Many proposals exist for reasoning about RBAC policies, each one using a different logic or formalism

Lithium: framework for specifying and reasoning about FGAC policies.

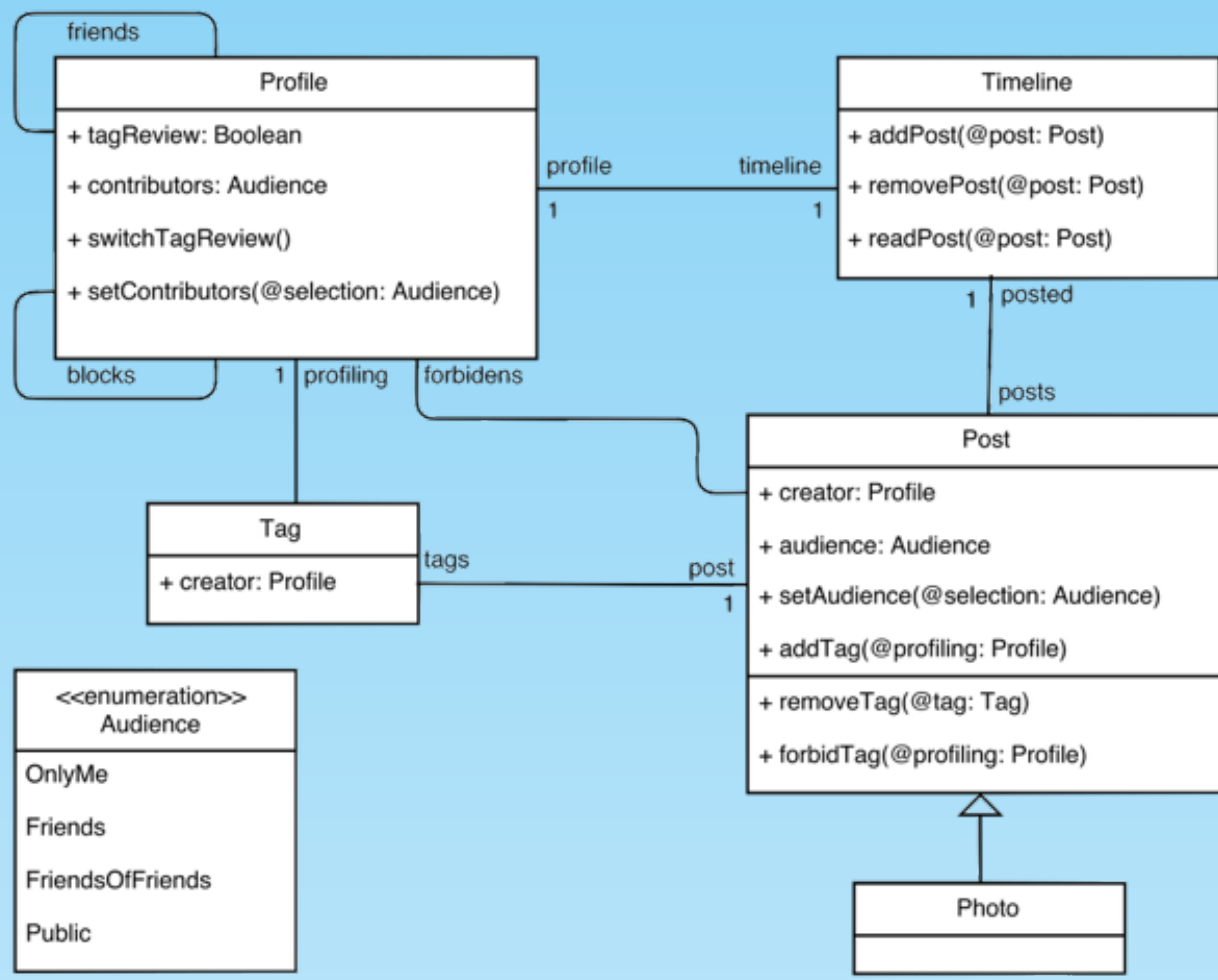
It is based on a decidable fragment of (multi-sorted) first-order logic. In contrast to OCL, this logic does not consider undefined values.

Kuhlmann et al: Employing UML and OCL for designing and analysing role-based access control models.

Analysing privacy models

Facebook: posting and tagging

- Who owns the timeline where the post is posted?
  - Who are his/her friends?
  - Who are his/her friends' friends?
- Who posted the post?
  - Who is tagged in the post? | Who are his/her friends?
  - Who are his/her friends' friends?
- Audience selected by the timeline's owner for a post that is posted in his/her timeline.



C Dania, M Clavel: Modeling Social Networking Privacy. TASE 2014: 50-57

Analysing privacy models

Facebook: posting and tagging

Alice posts a photo of herself, Bob and Ted in her timeline, and sets its audience to Friends. Then, Alice tags Bob in this photo.

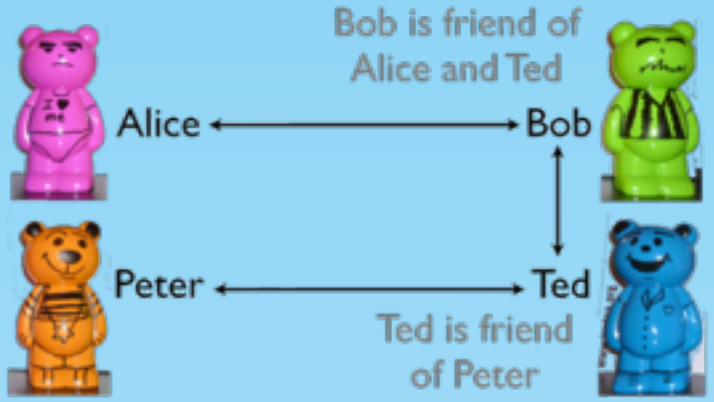
Can Bob see the photo in Alice's timeline? ✓

Alice has set her default audience to Friends.

post.audience= Friends

Bob is a friend of Alice.

self.profile.friends−>includes(caller)



Method:
readPost(post)

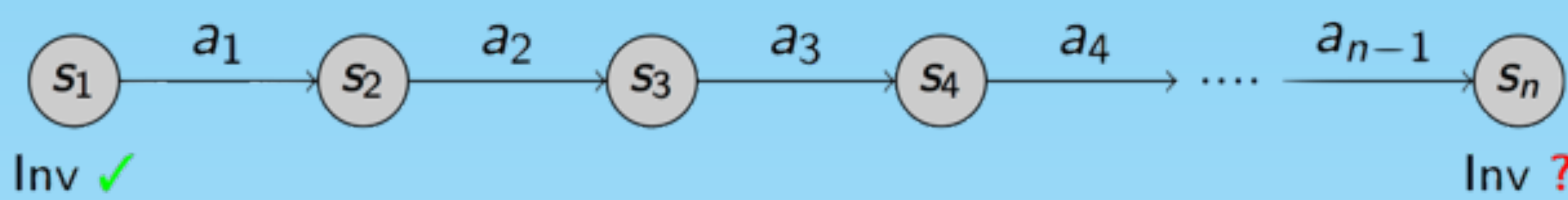
anybody can read any post that has its audience selected to 'Friends' and was created by the owner of the timeline, if he or she is a friend of somebody tagged on the post, unless he or she is blocked by the owner of the timeline.

(post.audience = 'Friends' and post.creator = self.profile
and post.tags.profiling.friends−>includes(caller)
and self.profile.blocks−>excludes(caller))

Checking data invariants preservation

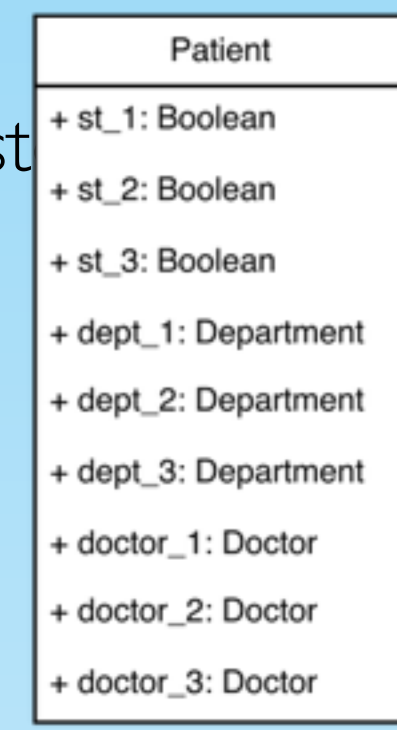
Steps

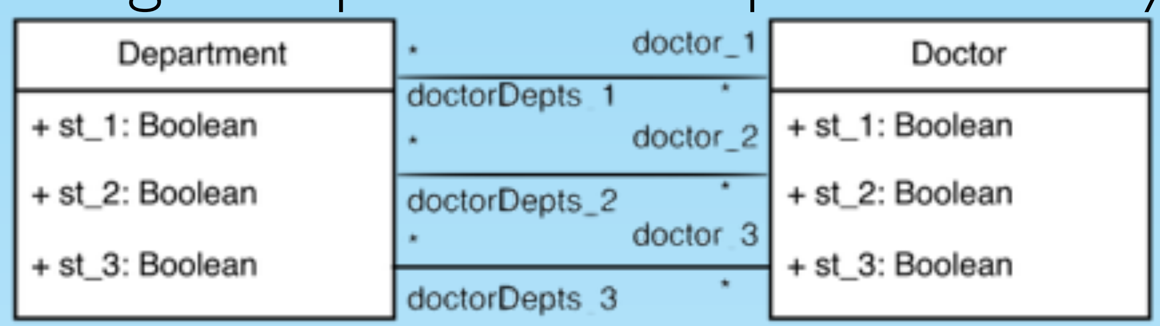
Preservation of the application's data invariants.



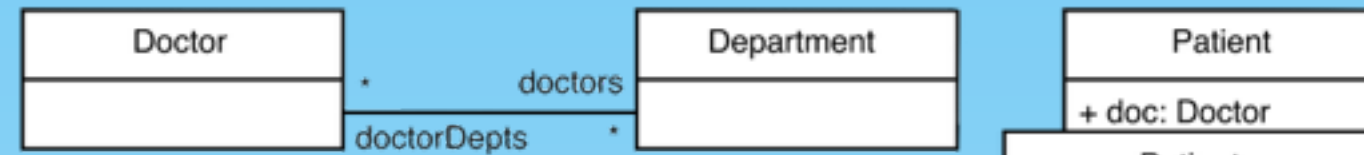
It consists in 3 steps:

Step 1: Modelling sequences of states (Film, Project).
A filmstrip is a way of encoding a sequence of snapshots of a syst

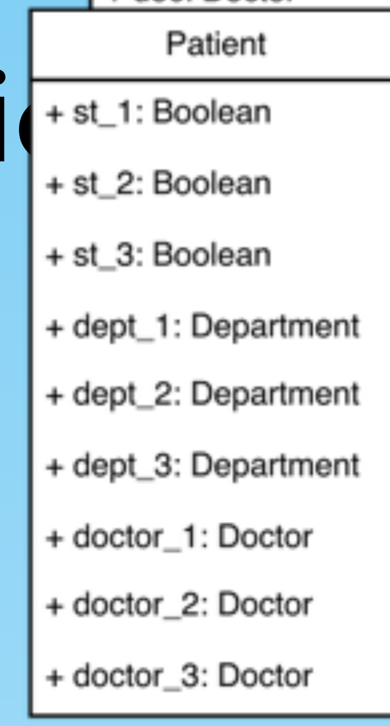




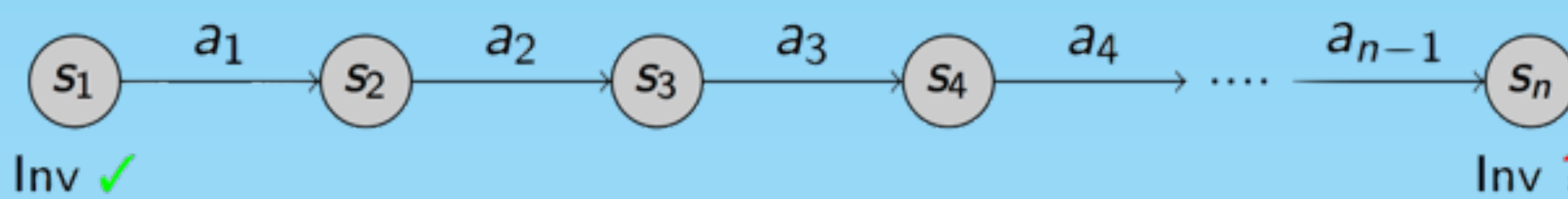
C. Dania, M. Clavel: Model-Based Formal Reasoning about Data-Management Applications. FASE 2015: 218-232
M.A. García de Dios, C. Dania, D. Basin, M. Clavel: Model-Driven Development of a Secure eHealth Application. Eng. Sec. Future Internet Services and Systems 2014: 97-118



Checking data invariants preservati



Preservation of … ion's data in…



It consists in 3 steps:

Step 1: Modelling sequences of states (Film, Project).
A filmstrip is a way of encoding a sequence of snapshots of a system.

Step 2: Modelling sequences of data actions (Execute)

Update(doctor, o1, i+1, 'Bob')

o1.doctor(i+1) = 'Bob'

Step 3: Proving invariants preservation.

C. Dania, M. Clavel: Model-Based Formal Reasoning about Data-Management Applications. FASE 2015: 218-232
M.A. García de Dios, C. Dania, D. Basin, M. Clavel: Model-Driven Development of a Secure eHealth Application. Eng. Sec. Future Internet Services and Systems 2014: 97-118

Checking data invariants preservation

Data model $\mathcal{D}$ with invariants Φ

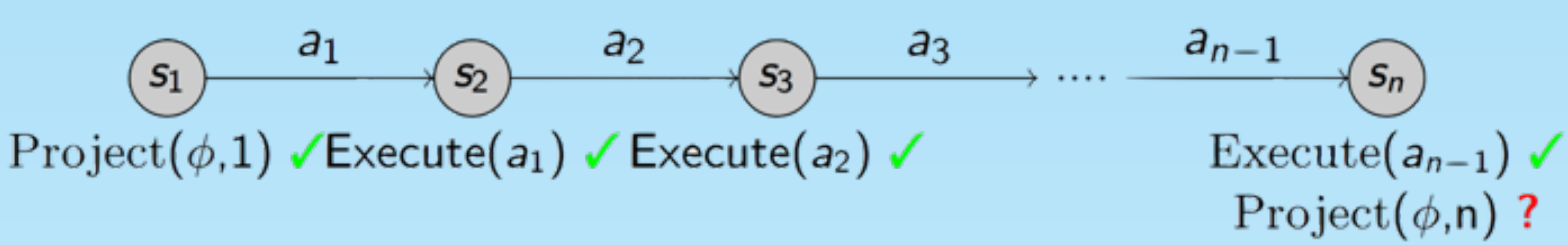
A sequence of actions $\mathcal{A} = \langle act_1, act_2, \ldots, act_n \rangle$

We say that ϕ preserves an invariant Υ if and only if:

$$\mathrm{o2f}_{\mathrm{data}}(\mathrm{Film}(\mathcal{D}, n)) \cup \{\mathrm{o2f}_{\mathrm{true}}(\gamma) \mid \gamma \in \bigcup_{i=1}^{n-1} \mathrm{Execute}(\mathcal{D}, act_i, i)\}$$

$$\cup\ \mathrm{o2f}_{\mathrm{true}}(\textsf{not}(\mathrm{Project}(\mathcal{D}, \bigwedge_{\psi \in \Phi} (\psi), 1)\ \textsf{implies}\ \mathrm{Project}(\mathcal{D}, \phi, n))).$$

is unsatisfiable.



Checking data invariants preservation

Case study: eHealth Record Management System

The data model contains18 entities, 40 attributes, and 48 association-ends.

	Acts.	Conds.	Invariants			Time		
			affected	preserved	violated	min.	max.	avge.
Create an administrative	8	9	18	18	0	0.03s	0.20s	0.50s
Create a nurse	10	11	22	22	0	0.03s	0.22s	0.06s
Create a doctor	11	12	25	24	1	0.03s	27.00s	0.07s
Reassing a doctor	2	6	2	2	0	6.88s	11.10s	8.94s
Reassing a nurse	2	6	2	1	1	0.10s	17.01s	8.55s
Register patient	30	6	28	26	2	0.03s	0.20s	0.05s
Move a patient	2	3	3	3	0	0.03s	0.03s	0.03

Related work. Gogolla et al. From Application Models to Filmstrip Models: An Approach to Automatic Validation of Model Dynamics.

Conclusions

- Code-generator from OCL queries to the procedural language extensions of SQL (SQL-PL)
  - each OCL expression is mapped to a single stored procedure
  - temporary tables are used
  - the three-valued evaluation semantics of OCL is considered
- Mapping from OCL to many-sorted FOL
  - our results depend of our formalization of UML/OCL in MSFOL and the heuristics implemented in the SMT solver (finite model finder),
  - the four-valued evaluation semantics of OCL is considered.
- Application domain:
  - checking consistency, analysing security and privacy properties, and checking data invariants preservation across states

Future work

- Look for the integration of developed tools into CASE tools
- Emprirical validation of the usefulness of the approach for a software engineering team.

Questions?

http://software.imdea.org/~dania/

publications + tools + case studies