On Metamodel Superstructures Employing UML Generalization Features

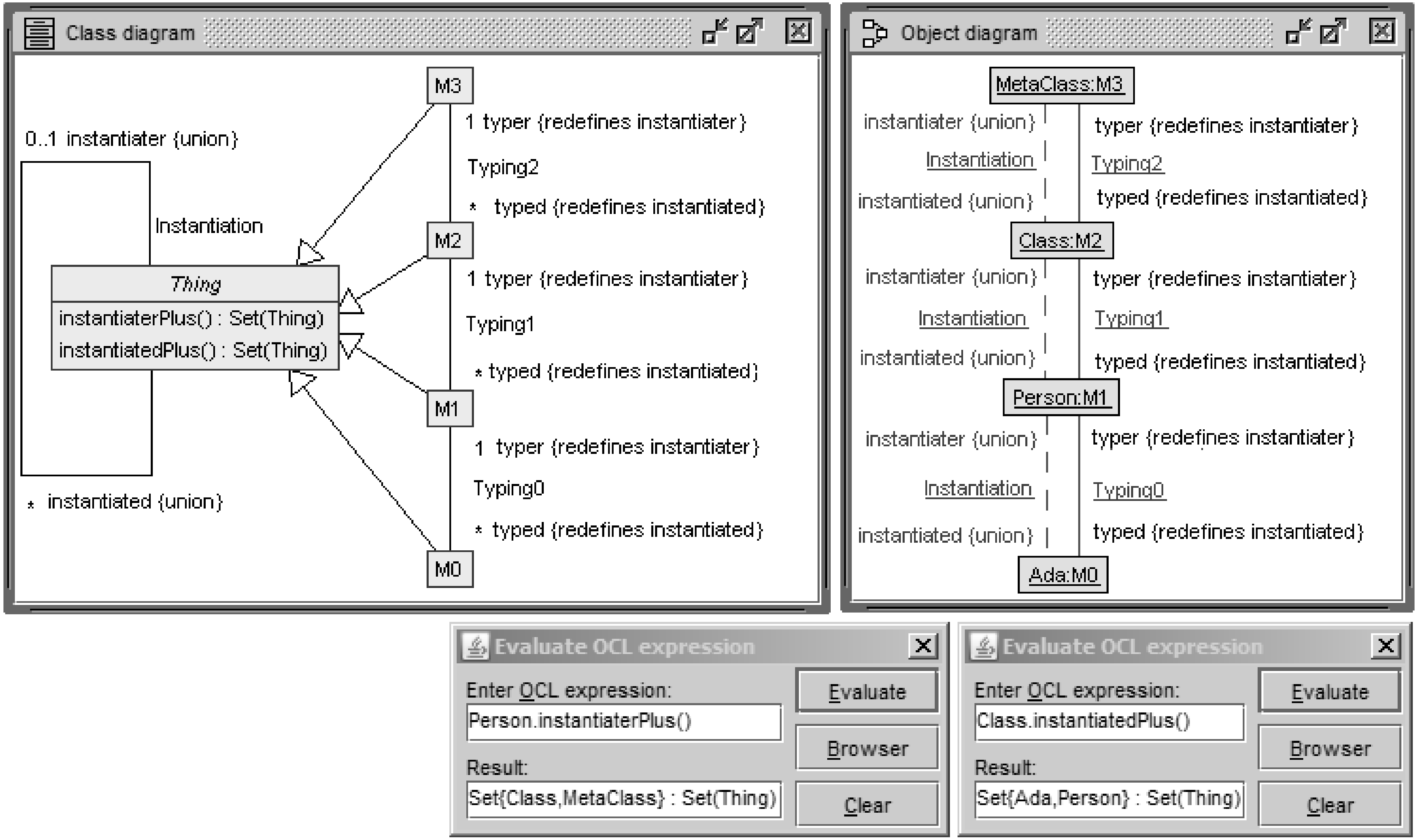
Martin Gogolla, Matthias Sedlmeier, Lars Hamann, Frank Hilken
University of Bremen, Database Systems

Motivation and context

- paper presents proposal for handling different metamodel levels in a uniform way
- in technical terms: represent different metamodel levels in ONE model, i.e. one class diagram including OCL constraints
- establish the connection between levels with associations and generalizations
- instanceOf relationship (usually between metamodel levels) can become a simple association with a precise meaning
- advantage: uniform employment of OCL
  - within each metamodel level,
  - for restricting the connection between the metamodel levels, and
  - for navigation between the metamodel levels

Example 1

Ada is a Person, Person is a Class, Class is MetaClass


Class diagram
0..1 instantiater {union}
Instantiation
Thing
instantiaterPlus() : Set(Thing)
instantiatedPlus() : Set(Thing)
* instantiated {union}
M3
1 typer {redefines instantiater}
Typing2
* typed {redefines instantiated}
M2
1 typer {redefines instantiater}
Typing1
* typed {redefines instantiated}
M1
1 typer {redefines instantiater}
Typing0
* typed {redefines instantiated}
M0
Object diagram
MetaClass:M3
instantiater {union}
Instantiation
instantiated {union}
typer {redefines instantiater}
Typing2
typed {redefines instantiated}
Class:M2
instantiater {union}
Instantiation
instantiated {union}
typer {redefines instantiater}
Typing1
typed {redefines instantiated}
Person:M1
instantiater {union}
Instantiation
instantiated {union}
typer {redefines instantiater}
Typing0
typed {redefines instantiated}
Ada:M0
Evaluate OCL expression
Evaluate
Browser
Clear
Evaluate OCL expression
Evaluate
Browser
Clear

Enter OCL expression:
Person.instantiaterPlus()

Result:
Set{Class,MetaClass} : Set(Thing)

Enter OCL expression:
Class.instantiatedPlus()

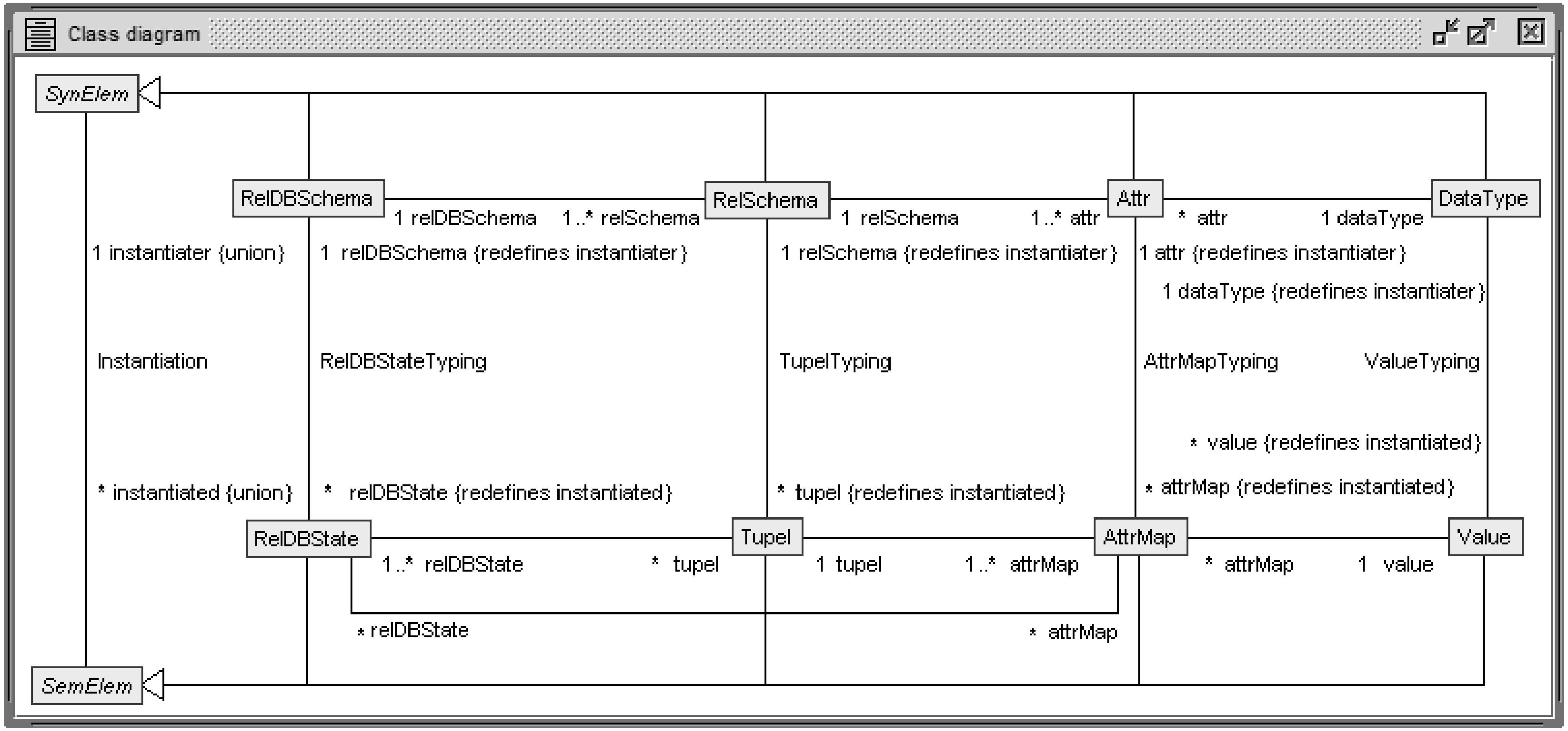
Result:
Set{Ada,Person} : Set(Thing)

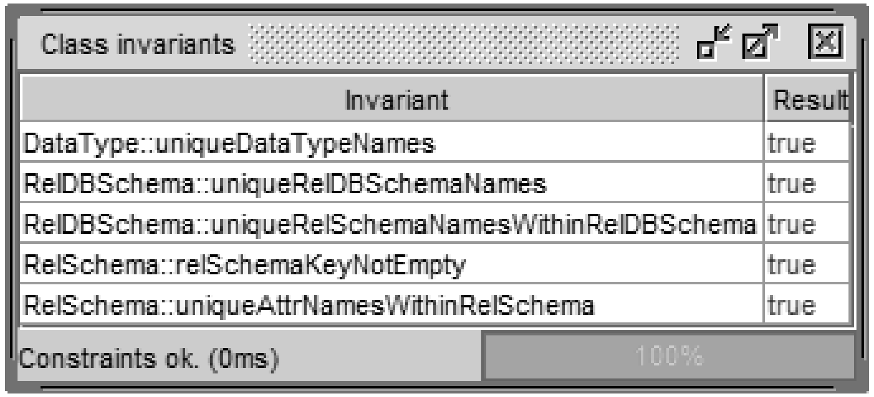
```
abstract class Thing
operations
  instantiatedPlus():Set(Thing)=
    self.instantiated->closure(t|t.instantiated)
  instantiaterPlus():Set(Thing)= ...

constraints
  inv acyclicInstantiation: self.instantiatedPlus()->excludes(self)
end
```

Example 2: Relational data model

Metamodel level 1 - Database schemata (Syntax)

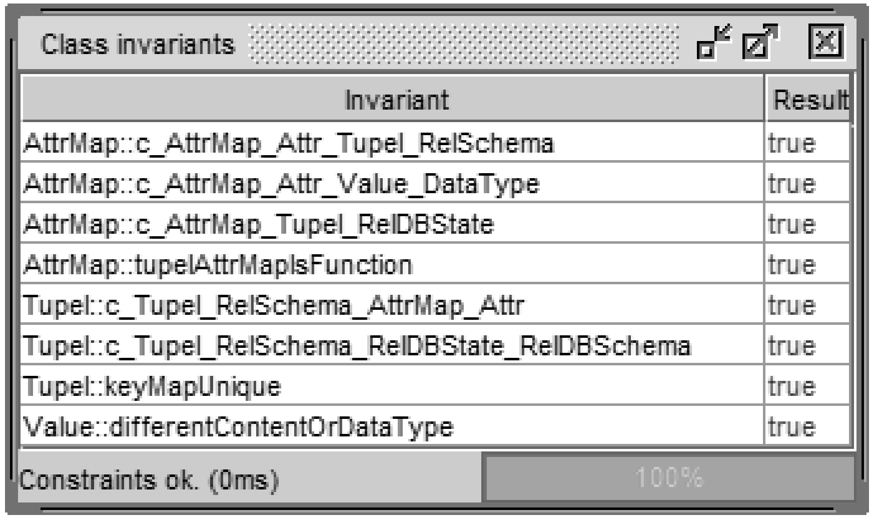
Metamodel level 0 - Database states (Semantics)




Class invariants

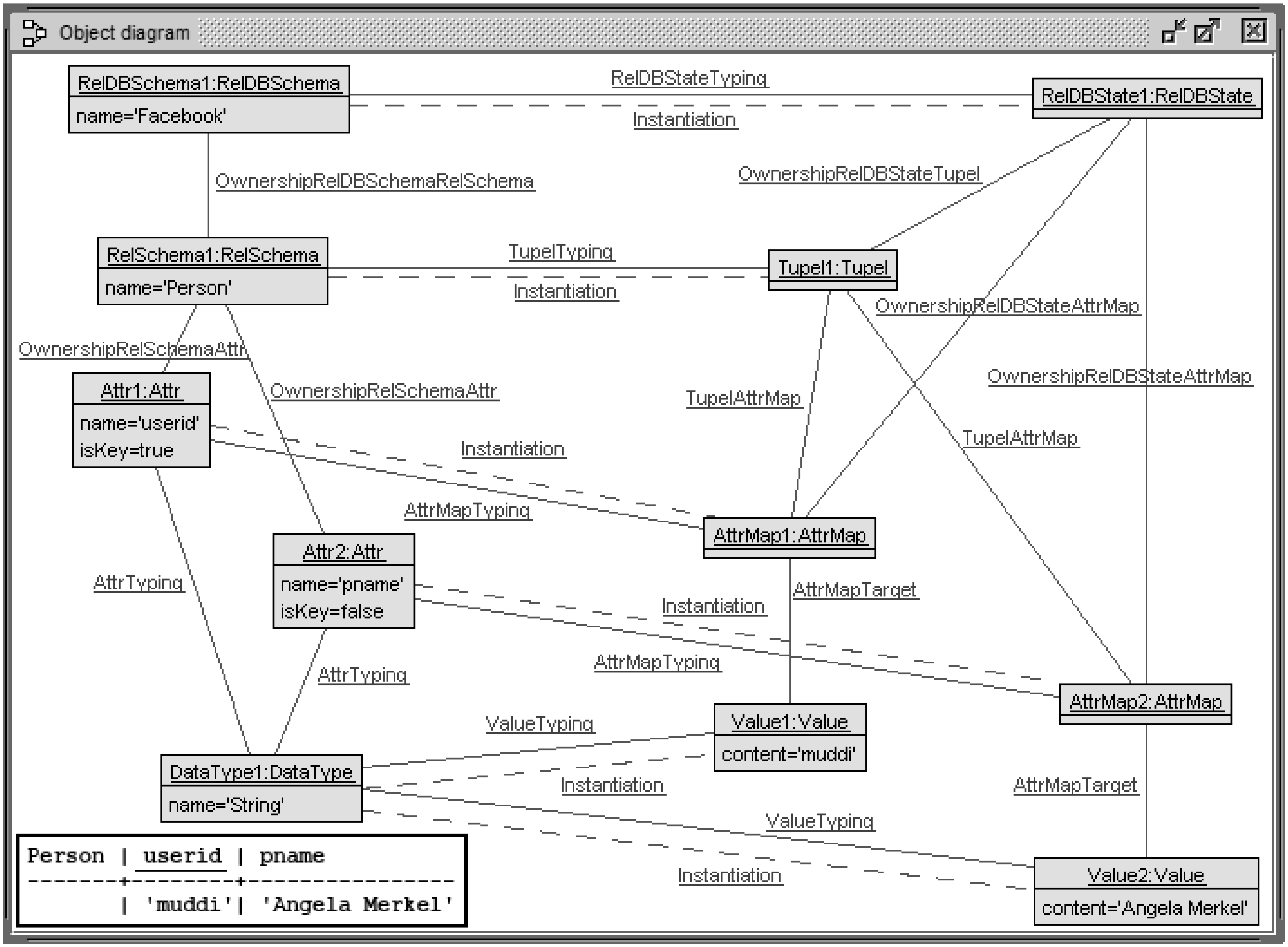
Invariant	Result
DataType::uniqueDataTypeNames	true
RelDBSchema::uniqueRelDBSchemaNames	true
RelDBSchema::uniqueRelSchemaNamesWithinRelDBSchema	true
RelSchema::relSchemaKeyNotEmpty	true
RelSchema::uniqueAttrNamesWithinRelSchema	true

Constraints ok. (0ms) 100%

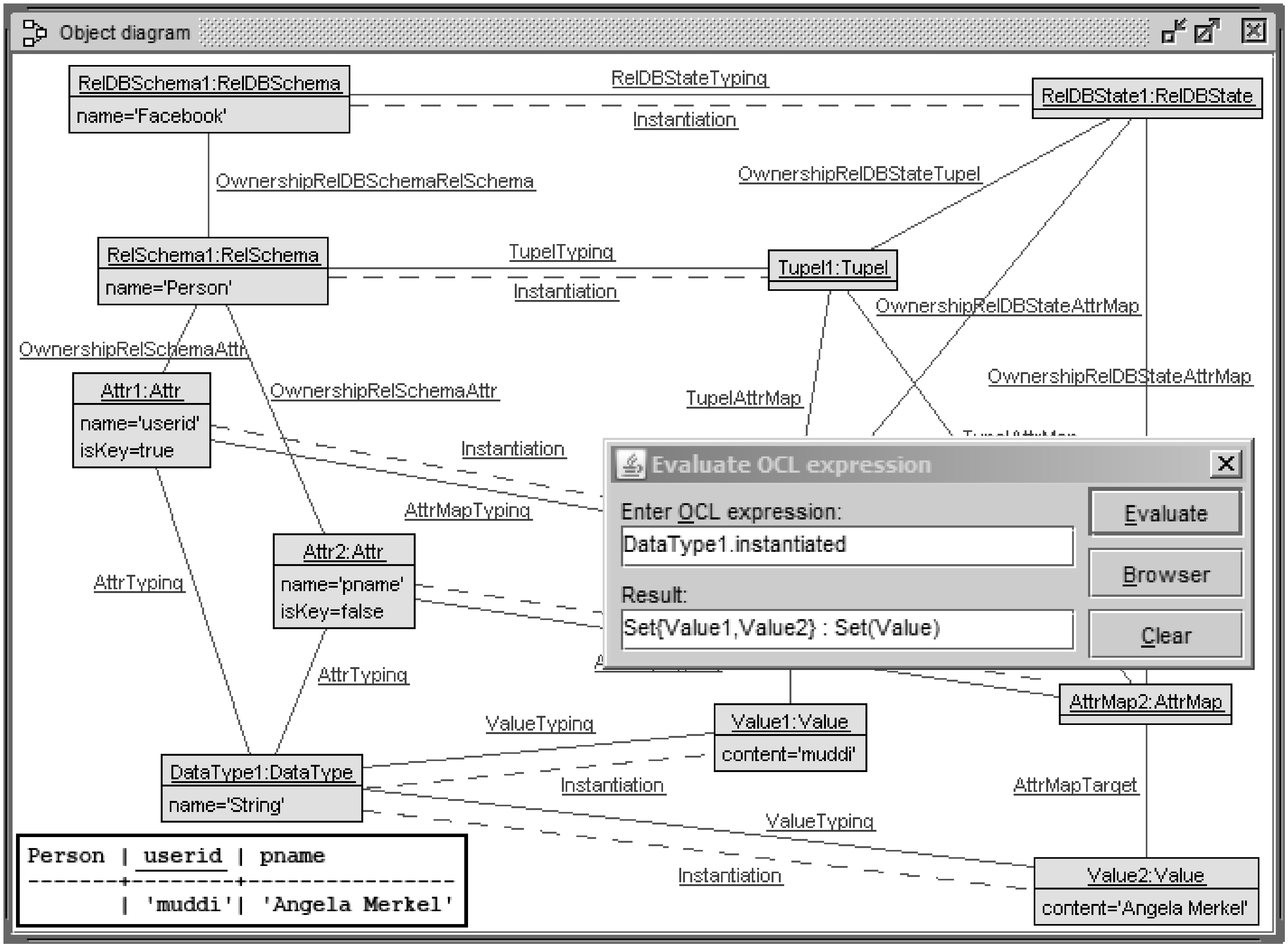

Class invariants

Invariant	Result
AttrMap::c_AttrMap_Attr_Tupel_RelSchema	true
AttrMap::c_AttrMap_Attr_Value_DataType	true
AttrMap::c_AttrMap_Tupel_RelDBState	true
AttrMap::tupelAttrMapIsFunction	true
Tupel::c_Tupel_RelSchema_AttrMap_Attr	true
Tupel::c_Tupel_RelSchema_RelDBState_RelDBSchema	true
Tupel::keyMapUnique	true
Value::differentContentOrDataType	true

Constraints ok. (0ms) 100%



```
Person | userid | pname
-------+--------+-----------------
       | 'muddi'| 'Angela Merkel'
```



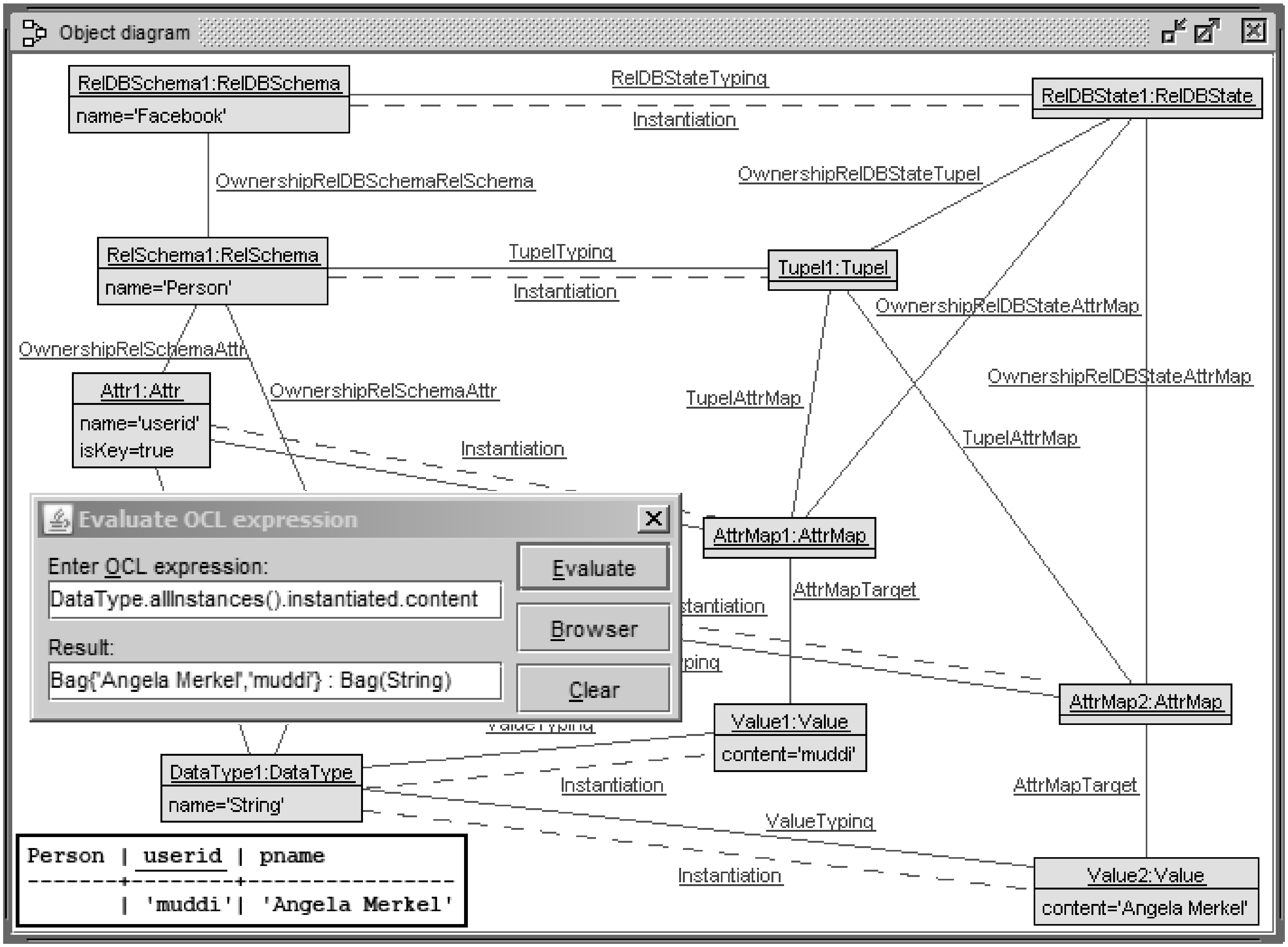
Enter OCL expression:

```
DataType1.instantiated
```

Result:

```
Set{Value1,Value2} : Set(Value)
```

```
Person | userid | pname
-------+--------+-----------------
       | 'muddi'| 'Angela Merkel'
```



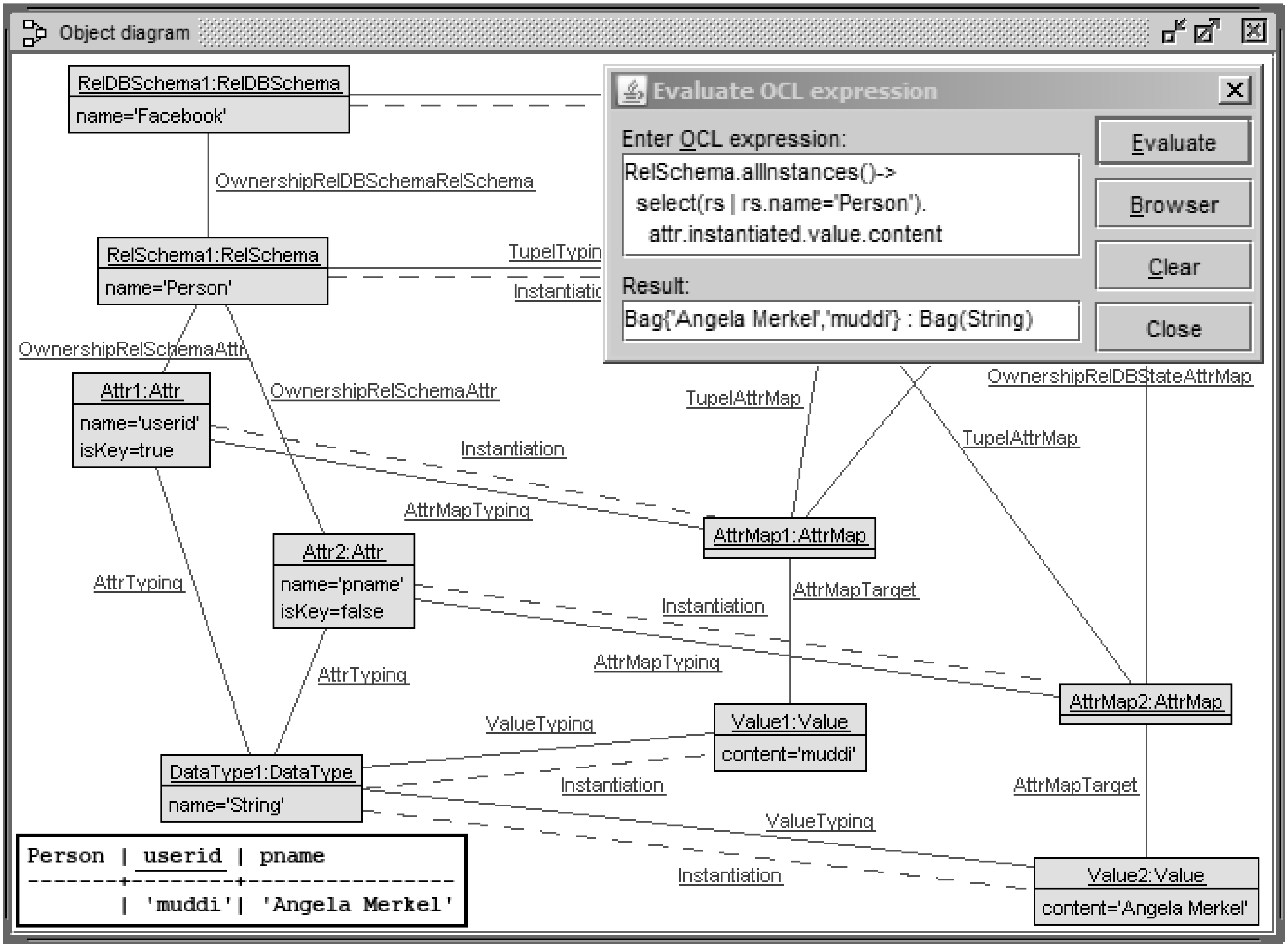
Enter OCL expression:

```
DataType.allInstances().instantiated.content
```

Result:

```
Bag{'Angela Merkel','muddi'} : Bag(String)
```

```
Person | userid | pname
-------+--------+----------------
       | 'muddi'| 'Angela Merkel'
```



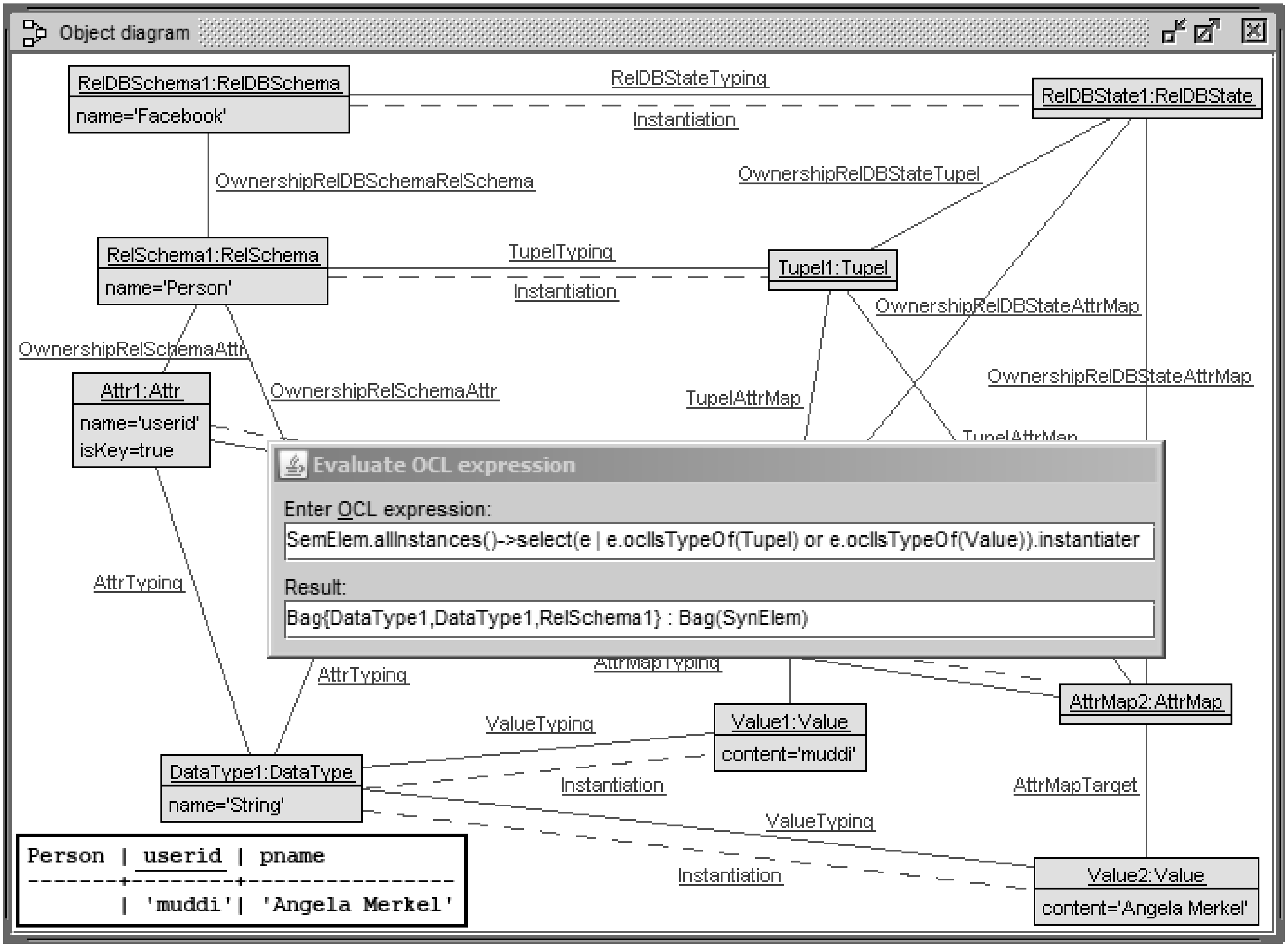
Enter OCL expression:

```
RelSchema.allInstances()->
  select(rs | rs.name='Person').
    attr.instantiated.value.content
```

Result:

```
Bag{'Angela Merkel','muddi'} : Bag(String)
```

```
Person | userid | pname
-------+--------+----------------
       | 'muddi'| 'Angela Merkel'
```



Evaluate OCL expression

Enter OCL expression:

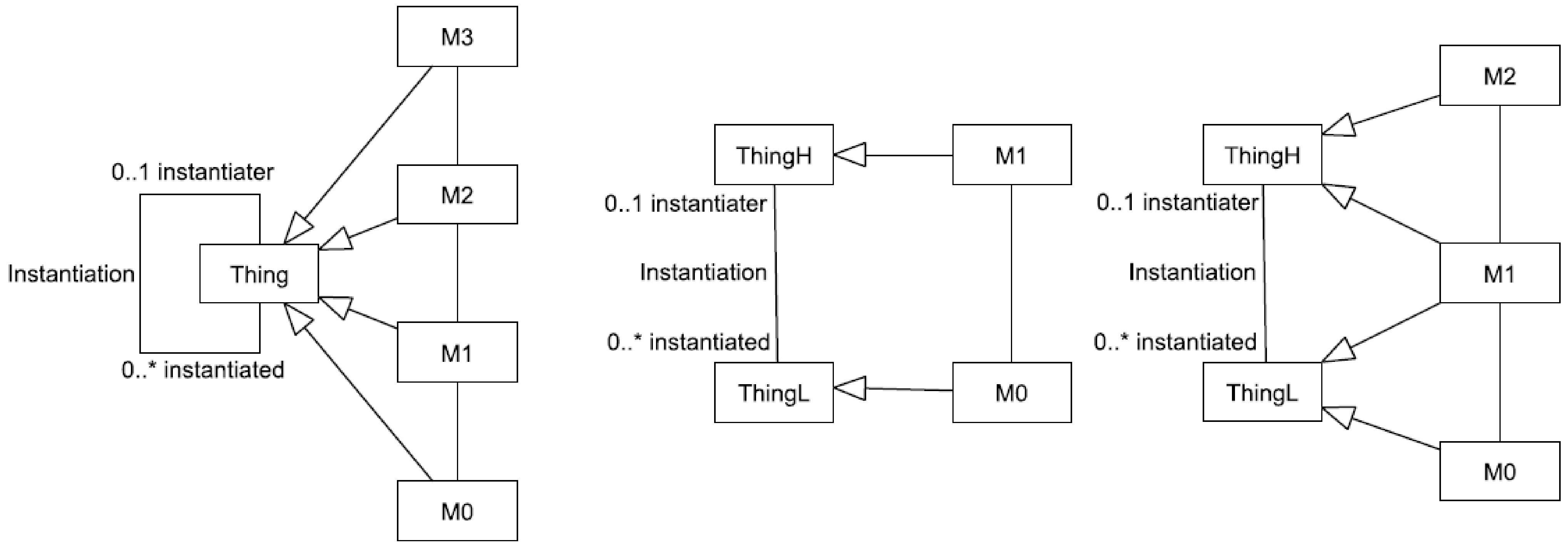
```
SemElem.allInstances()->select(e | e.oclIsTypeOf(Tupel) or e.oclIsTypeOf(Value)).instantiater
```

Result:

```
Bag{DataType1,DataType1,RelSchema1} : Bag(SynElem)
```

```
Person | userid | pname
-------+--------+-----------------
       | 'muddi'| 'Angela Merkel'
```

Different metamodel structures



Conclusion and summary

- presented an approach for incorporating different metamodel levels into a single model

- used associations, generalizations and OCL for restricting the connection between metamodel levels

Future work

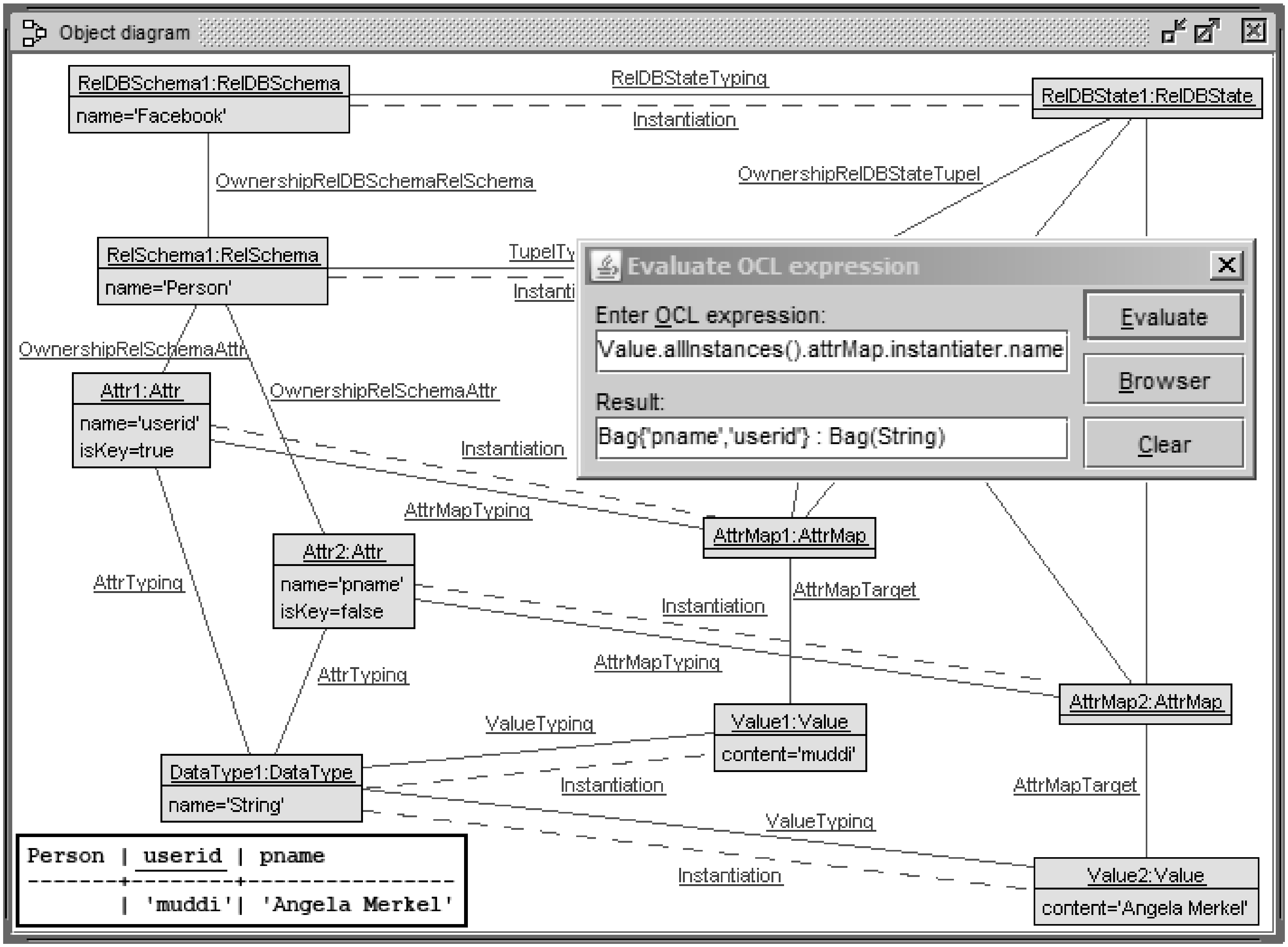
- discover connections to and formalize notions like clabject, potency, powertype

- build more case studies in order to obtain more insights into advantages and drawbacks

- extend our tool USE to cope with (at least) three modeling levels

```
- class diagram
- object diagram = class diagram
-                  object diagram
```

Thanks for your attention!



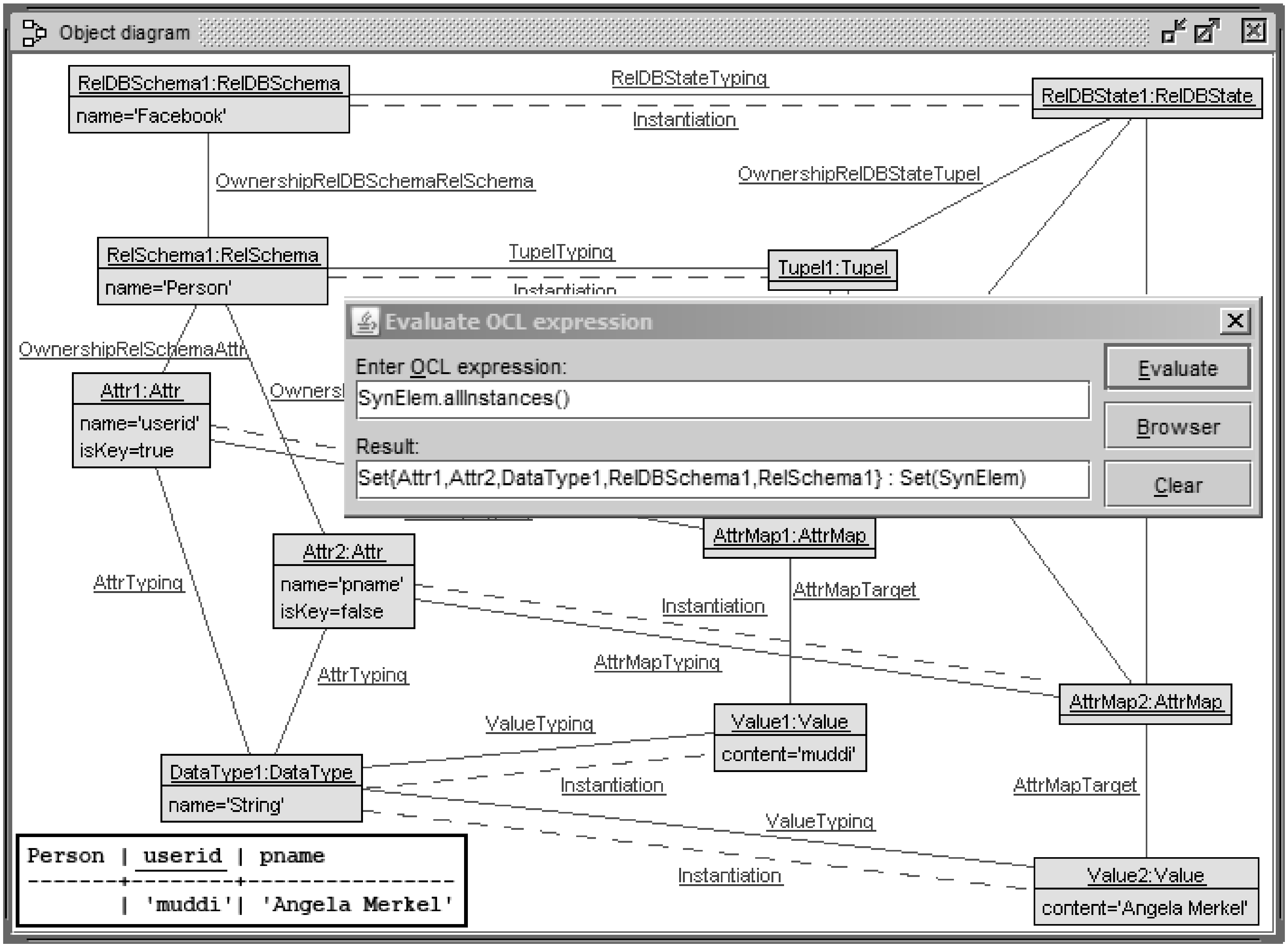
Enter OCL expression:

```
Value.allInstances().attrMap.instantiater.name
```

Result:

```
Bag{'pname','userid'} : Bag(String)
```

```
Person | userid | pname
-------+--------+----------------
       | 'muddi'| 'Angela Merkel'
```



Enter OCL expression:

```
SynElem.allInstances()
```

Result:

```
Set{Attr1,Attr2,DataType1,RelDBSchema1,RelSchema1} : Set(SynElem)
```

```
Person | userid | pname
-------+--------+-----------------
       | 'muddi'| 'Angela Merkel'
```