

# Data Systematics: The PSOA RuleML Metamodel Illustrated by Grailog Visualization of Wedding Atoms

(PDF version: [ruleml.org/talks/PSOAMetamodelGrailogWedding.pdf](http://ruleml.org/talks/PSOAMetamodelGrailogWedding.pdf))

Harold Boley

University of New Brunswick  
Faculty of Computer Science  
Fredericton, NB, Canada

July 13, 2018

Update: Oct. 15, 2019

# Introduction

- [PSOA RuleML](#) builds on a novel data systematics
- Slicing and dicing the *PSOA metamodel cube* (from [PSOAPerspectivalKnowledge](#), Appendix A)
- Exemplify with oidless/oidful, tupled/slotted/combined, independent/dependent/combined atoms ( $2*3*3 = 18$ )
- Illustrate all kinds of atoms by [Grailog](#) visualization, realizing them in *presentation syntax* by [PSOATransRun](#)
- Informal syntax templates and English semantics (formal in [PSOAPerspectivalKnowledge](#), Sections 4 and 5)
- Experience full metamodel dynamically by online [PSOAMetaViz](#) visualization, realized in JavaScript/JSON

# Slicing and Dicing the PSOA Metamodel Cube

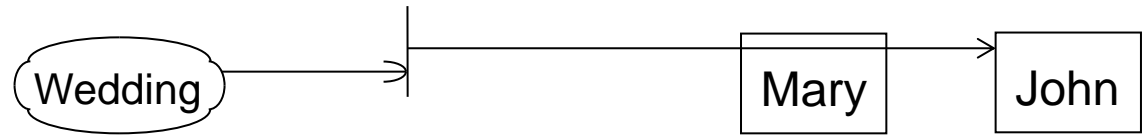
- Via 3 (orthogonal) dimensions, the **full metamodel** cube systematizes 18 kinds of atoms that are contained in 18 unit cubes (units) named  $inj$ ,  $dej$ ,  $idj$  ( $j=1,\dots,6$ )
- Choosing one of the reductions DVO, VDO, or OVD (s. below), users can slice and dice the cube, in a kind of (meta)[OLAP](#), initially reducing its 3 dimensions to slices of 2 dimensions:
- **DVO** reduction, via **Dependency** dimension, to 3 slices, each with 6 units structured by **Variety**-row and **OID**-column dimensions:
  - 6 **independent** units **inj** ( $j=1,\dots,6$ ) vs. 6 **dependent** units **dej** ( $j=1,\dots,6$ ) vs. 6 combined independent+dependent units **idj** ( $j=1,\dots,6$ )
- The **core metamodel** is an 8-unit subcube of the full metamodel cube, which can be reduced, DVO-style, to 2 Dependency slices:  $in1-in4$  and  $de1-de4$ 
  - Each includes a 'landmark' unit: **framepoint** atoms ( $in4$ ) and **relationship** atoms ( $de1$ )
- **VDO** reduction (e.g., for full metamodel), via **Variety** dimension, to 3 slices, each with 6 units structured by **Dependency**-row and **OID**-column dimensions:
  - 6 tupled+slotted units  $inj$ ,  $dej$ ,  $idj$  ( $j=5,6$ ) vs. 6 slotted units  $inj$ ,  $dej$ ,  $idj$  ( $j=3,4$ ) vs. 6 tupled units  $inj$ ,  $dej$ ,  $idj$  ( $j=1,2$ )
- **OVD** reduction (e.g., for full metamodel), via **OID** dimension, to 2 slices, each with 9 units structured by **Variety**-row and **Dependency**-column dimensions:
  - 9 oidful units  $inj$ ,  $dej$ ,  $idj$  ( $j=2,4,6$ ) vs. 9 oidless units  $inj$ ,  $dej$ ,  $idj$  ( $j=1,3,5$ )

# Exemplifying the Dependency Slices

Core oidless/oidful, tupled/slotted atoms that are **independent**:

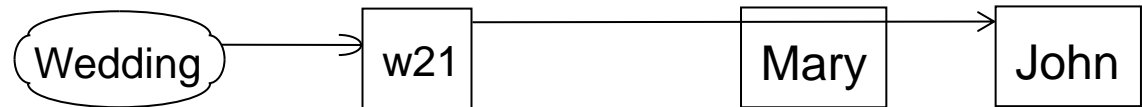
*Grailog:*

in1. single-tuple:  
shelfships



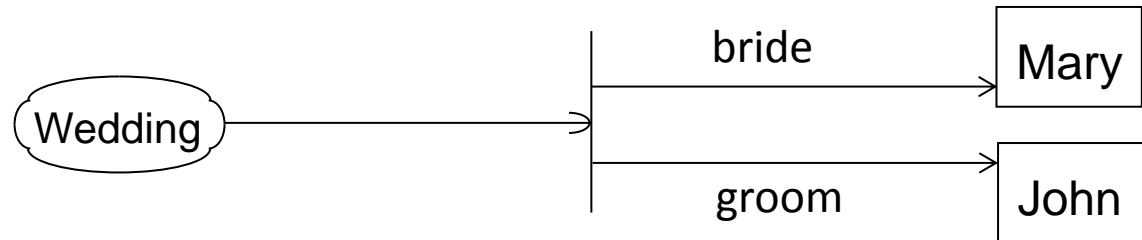
`Wedding(-[Mary John])`

in2. single-tuple:  
shelfpoints



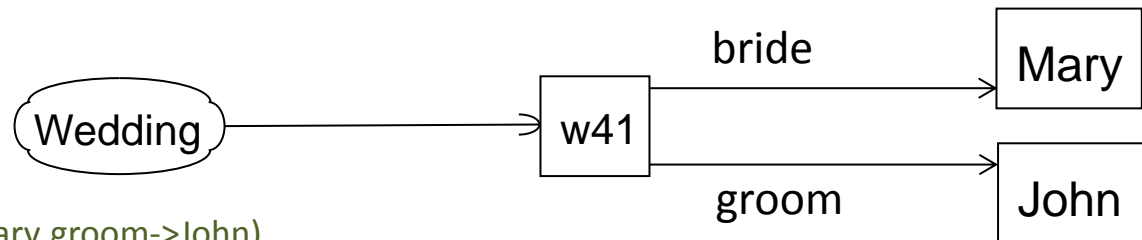
`w21#Wedding(-[Mary John])`

in3. frameships



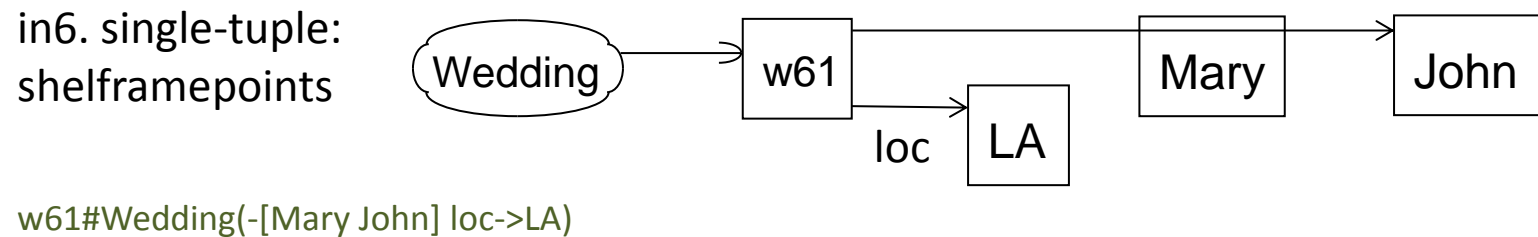
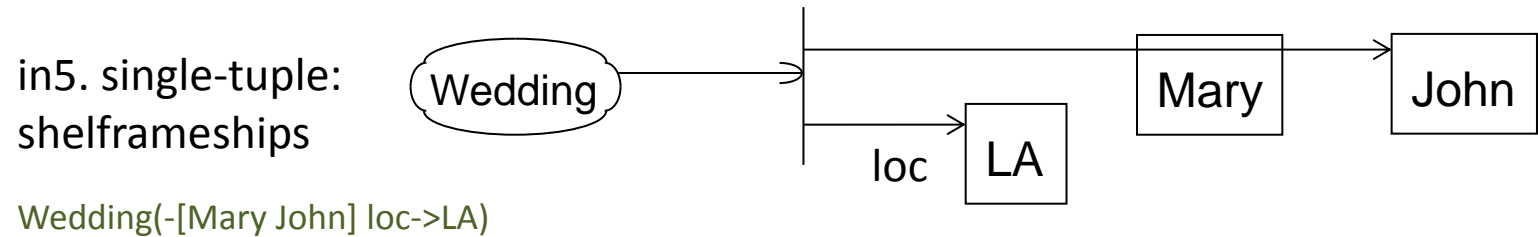
`Wedding(bride->Mary groom->John)`

in4: **framepoints**



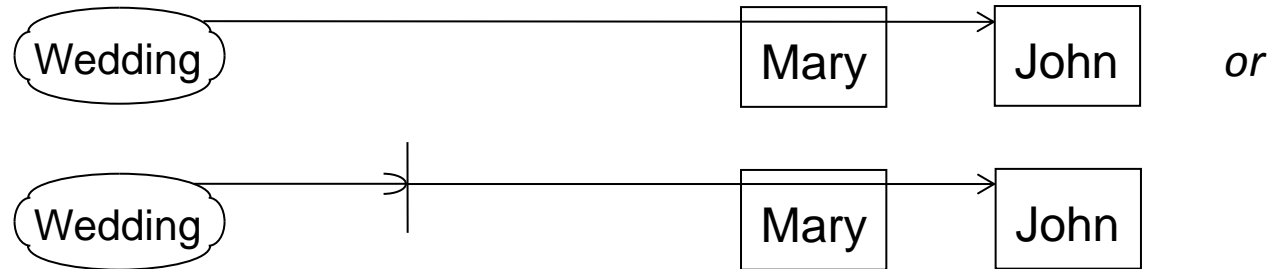
`w41#Wedding(bride->Mary groom->John)`

Extra oidless/oidful, combined tupled+slotted atoms that are **independent**:



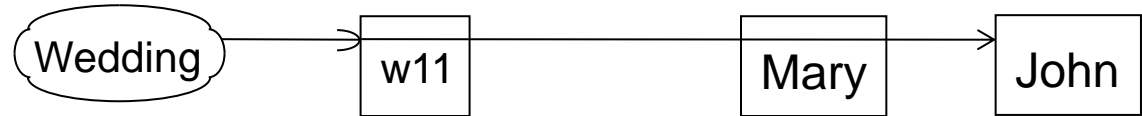
Core oidless/oidful, tupled/slotted atoms that are **dependent**:

de1. single-tuple:  
**relationships**



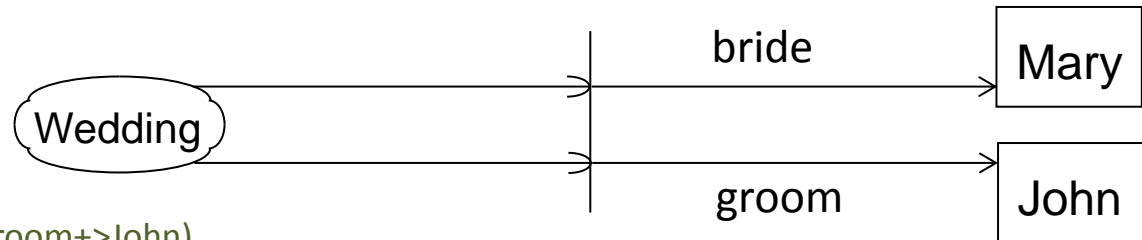
*Wedding(Mary John) or Wedding(+[Mary John])*

de2. single-tuple:  
relationpoints



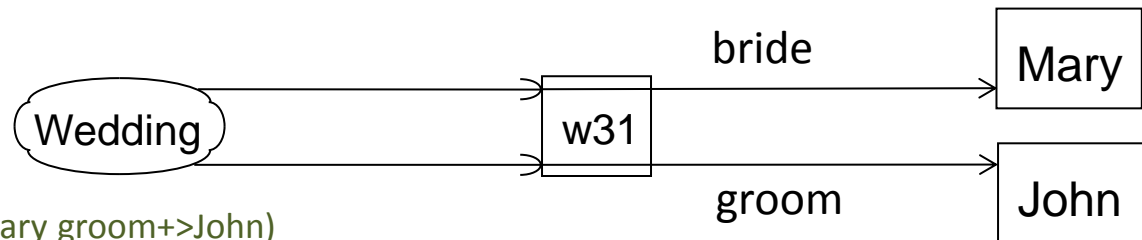
*w11#Wedding(+[Mary John])*

de3: pairships



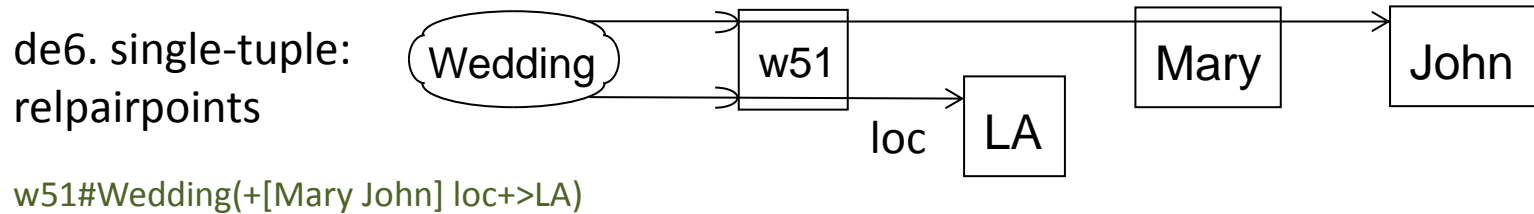
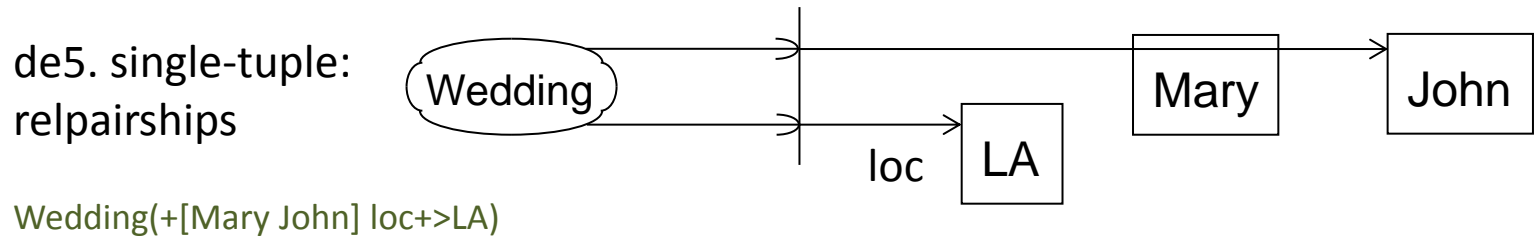
*Wedding(bride+>Mary groom+>John)*

de4. pairpoints

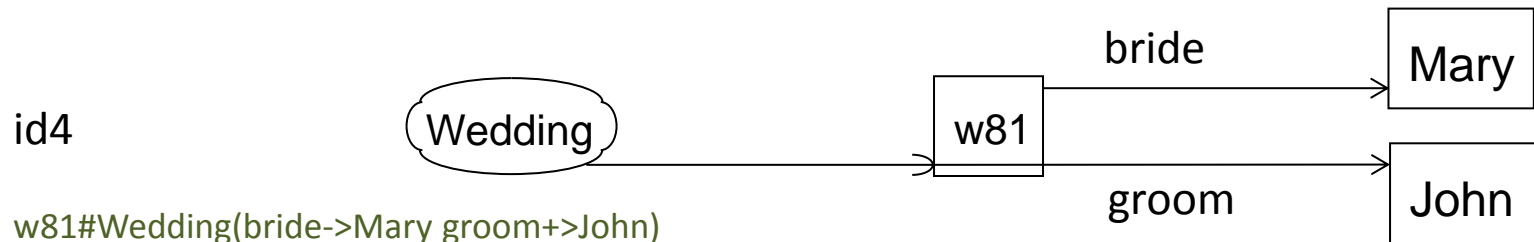
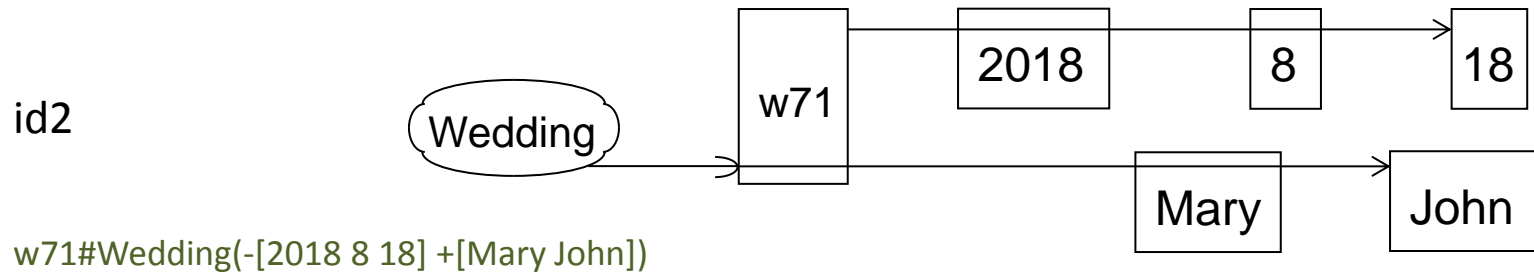
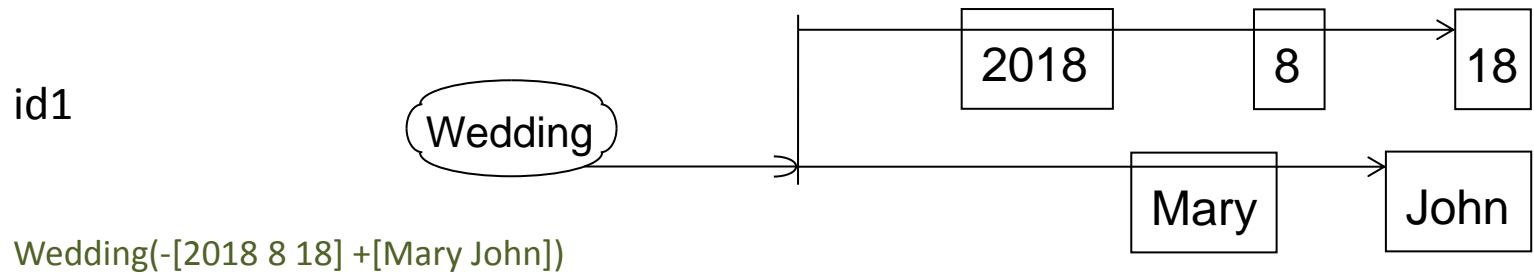


*w31#Wedding(bride+>Mary groom+>John)*

Extra oidless/oidful, combined tupled+slotted atoms that are **dependent**:

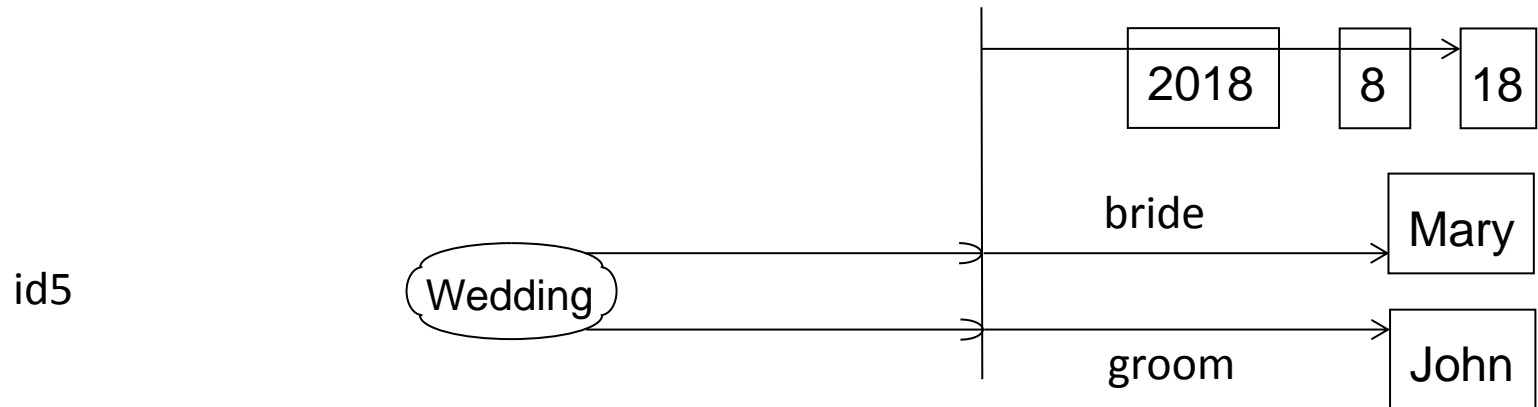


Adding oidless/oidful, tupled/slotted, combined independent+dependent atoms:

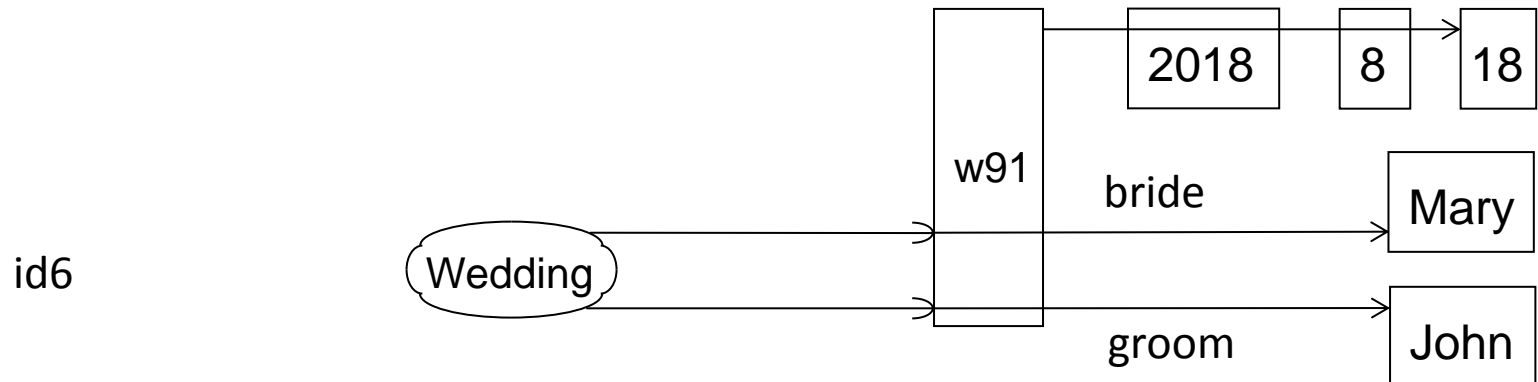




Also oidless/oidful, combined tupled+slotted, combined independent+dependent:



Wedding(-[2018 8 18] bride+>Mary groom+>John)



w91#Wedding(-[2018 8 18] bride+>Mary groom+>John)

# Syntax and Semantics of Atoms

Core oidless/oidful, tupled/slotted atoms that are **independent**:

in1. single-tuple:

shelfships

$f(-[t \dots t] \dots -[t \dots t])$

Implicit existential OID; tuples  $-[t \dots t]$  independent from predicate  $f$

in2. single-tuple:

shelfpoints

$o\#f(-[t \dots t] \dots -[t \dots t])$

$o\#f(-[t \dots t])$

Explicit OID  $o$ ; tuples  $-[t \dots t]$  independent from predicate  $f$

in3: frameships

$f(p \rightarrow v \dots p \rightarrow v)$

Implicit existential OID; slots  $p \rightarrow v$  independent from predicate  $f$

in4: **framepoints**

$o\#f(p \rightarrow v \dots p \rightarrow v)$

Explicit OID  $o$ ; slots  $p \rightarrow v$  independent from predicate  $f$

Extra oidless/oidful, combined tupled+slotted atoms that are **independent**:

in5. single-tuple:  
shelframeships

$f(-[t \dots t] \dots -[t \dots t] p \rightarrow v \dots p \rightarrow v)$

Implicit existential OID; descriptors independent from predicate  $f$

in6. single-tuple:  
shelframepoints

$o\#f(-[t \dots t] \dots -[t \dots t] p \rightarrow v \dots p \rightarrow v)$

$o\#f(-[t \dots t] p \rightarrow v \dots p \rightarrow v)$

Explicit OID  $o$ ; descriptors independent from predicate  $f$

Core oidless/oidful, tupled/slotted atoms that are **dependent**:

de1. single-tuple:  
**relationships**

$f(+[t \dots t] \dots +[t \dots t])$

$f(t \dots t) \text{ or } f(+[t \dots t])$

Implicit existential OID; tuples  $+[t \dots t]$  dependent on predicate  $f$

de2. single-tuple:  
**relationpoints**

$o\#f(+[t \dots t] \dots +[t \dots t])$

$o\#f(t \dots t) \text{ or } o\#f(+[t \dots t])$

Explicit OID  $o$ ; tuples  $+[t \dots t]$  dependent on predicate  $f$

de3: pairships

$f(p+>v \dots p+>v)$

Implicit existential OID; slots  $p+>v$  dependent on predicate  $f$

de4: pairpoints

$o\#f(p+>v \dots p+>v)$

Explicit OID  $o$ ; slots  $p+>v$  dependent on predicate  $f$

Extra oidless/oidful, combined tupled+slotted atoms that are **dependent**:

de5. single-tuple:  
relpairships

$f(+[t \dots t] \dots +[t \dots t] p+>v \dots p+>v)$

Implicit existential OID; descriptors dependent on predicate  $f$

$f(+[t \dots t] p+>v \dots p+>v)$

de6. single-tuple:  
relpairpoints

$o\#f(+[t \dots t] \dots +[t \dots t] p+>v \dots p+>v)$  Explicit OID  $o$ ; descriptors dependent on predicate  $f$

Adding oidless/oidful, tupled/slotted, combined independent+dependent atoms:

id1

$f(+[t \dots t] \dots +[t \dots t]$   
 $-[t \dots t] \dots -[t \dots t])$

Implicit existential OID; both in/dependent tuples w.r.t. predicate  $f$

id2

$o\#f(+[t \dots t] \dots +[t \dots t]$   
 $-[t \dots t] \dots -[t \dots t])$

Explicit OID  $o$ ; both in/dependent tuples w.r.t. predicate  $f$

id3

$f(p \rightarrow v \dots p \rightarrow v$   
 $p \rightarrow v \dots p \rightarrow v)$

Implicit existential OID; both in/dependent slots w.r.t. predicate  $f$

id4

$o\#f(p \rightarrow v \dots p \rightarrow v$   
 $p \rightarrow v \dots p \rightarrow v)$

Explicit OID  $o$ ; both in/dependent slots w.r.t. predicate  $f$

Also oidless/oidful, combined tupled+slotted, combined independent+dependent:

id5

$f(+[t \dots t] \dots +[t \dots t]$   
 $-[t \dots t] \dots -[t \dots t]$   
 $p+>v \dots p+>v$   
 $p->v \dots p->v)$

Implicit existential OID; both in/dependent descriptors w.r.t. predicate  $f$

id6

$o\#f(+[t \dots t] \dots +[t \dots t]$   
 $-[t \dots t] \dots -[t \dots t]$   
 $p+>v \dots p+>v$   
 $p->v \dots p->v)$

Explicit OID  $o$ ; both in/dependent descriptors w.r.t. predicate  $f$

# Conclusions

- Full PSOA metamodel cube visualized dynamically by [PSOAMetaViz](#), and atoms (e.g., data facts) in Grailog, to significantly facilitate learning PSOA RuleML
- Facts augmented by (interoperation) rules:  
[http://wiki.ruleml.org/index.php/PSOA\\_RuleML\\_Bridges\\_Graph\\_and\\_Relational\\_Databases](http://wiki.ruleml.org/index.php/PSOA_RuleML_Bridges_Graph_and_Relational_Databases)  
(includes core interoperation path de1-de3-de4-in4, e.g. abridged to one PSOA rule)
- Core path augmented to roundtrip between wedding atoms:  
[http://wiki.ruleml.org/index.php/Exploring\\_the\\_PSOA\\_RuleML\\_Space\\_of\\_Core\\_Atoms](http://wiki.ruleml.org/index.php/Exploring_the_PSOA_RuleML_Space_of_Core_Atoms)
- PSOA RuleML 1.03 being standardized by Relax NG schemas for XML-serialized facts and rules:  
[http://wiki.ruleml.org/index.php/PSOA\\_RuleML#Syntax](http://wiki.ruleml.org/index.php/PSOA_RuleML#Syntax)
- PSOA metamodel transferrable to other languages