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

(PDF version: ruleml.org/talks/PSOAMetamodelGrailogWedding.pdf**)**

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

July 13, 2018
Update: Dec. 26, 2018

Introduction

- PSOA RuleML builds on a novel data systematics
- Slicing and dicing the *PSOA metamodel cube* (from PSOAPerspectivalKnowledge, Appendix A)
- Exemplify with 18 oidless/oidful, tupled/slotted/…, perspeneutral/perspectival/… wedding atoms
- Illustrate by Grailog visualization all kinds of atoms in *presentation syntax* realized by PSOATransRun
- Informal syntax templates and English semantics (formal in PSOAPerspectivalKnowledge, Sections 4 and 5)

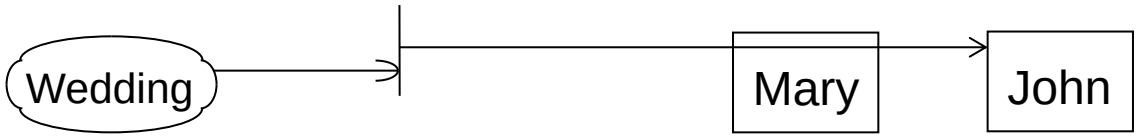
Slicing and Dicing the PSOA Metamodel Cube

- Via 3 (orthogonal) dimensions, the **full metamodel** cube systematizes 18 kinds of atoms that are contained in (3*3*2 =) 18 unit cubes (units) named p*xi* (*x*=n,v,p; *i*=1,…,6)
- By choosing one of the reductions PDO, DPO, or ODP, users can variously slice and dice the cube, in a kind of (meta)OLAP, initially reducing its 3 dimensions to slices of 2 dimensions:
- PDO reduction, via Perspectivity dimension, to 3 slices, each with 6 units structured by Descriptor-row and OID-column dimensions:
  - 6 perspeneutral units (*x*=n; *i*=1,…,6) vs. 6 perspectival units (*x*=v; *i*=1,…,6) vs. 6 perspeneutral+perspectival units (*x*=p; *i*=1,…,6)
- The **core metamodel** is an 8-unit subcube of the full metamodel cube, which can be reduced, PDO-style, to 2 Perspectivity slices: pn1-pn4 and pv1-pv4
  - Each includes a prominent unit: ***framepoint*** atoms (pn4) and ***relationship*** atoms (pv1)
- DPO reduction (e.g., for full metamodel), via Descriptor dimension, to 3 slices, each with 6 units structured by Perspectivity-row and OID-column dimensions:
  - 6 tupled+slotted units (*x*=n,v,p; *i*=5,6) vs. 6 slotted units (*x*=n,v,p; *i*=3,4) vs. 6 tupled units (*x*=n,v,p; *i*=1,2)
- ODP reduction (e.g., for full metamodel), via OID dimension, to 2 slices, each with 9 units structured by Descriptor-row and Perspectivity-column dimensions:
  - 9 oidful units (*x*=n,v,p; *i*=2,4,6) vs. 9 oidless units (*x*=n,v,p; *i*=1,3,5)

Exemplifying the Perspectivity Slices

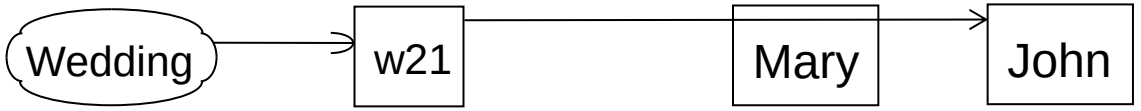
Core oidless/oidful, tupled/slotted atoms that are **p**erspe**n**eutral:

pn1. single-tuple: shelfships



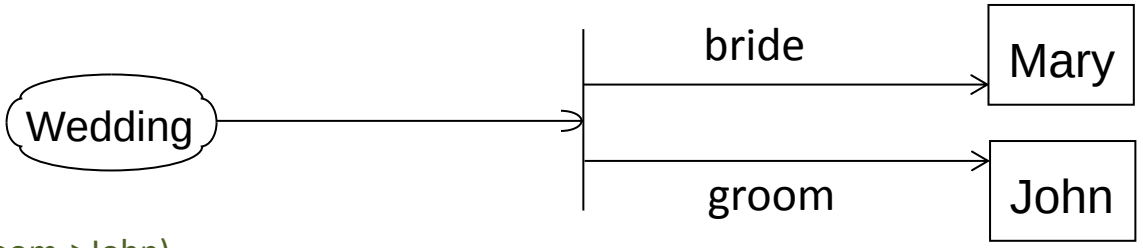
Wedding(-[Mary John])

pn2. single-tuple: shelfpoints



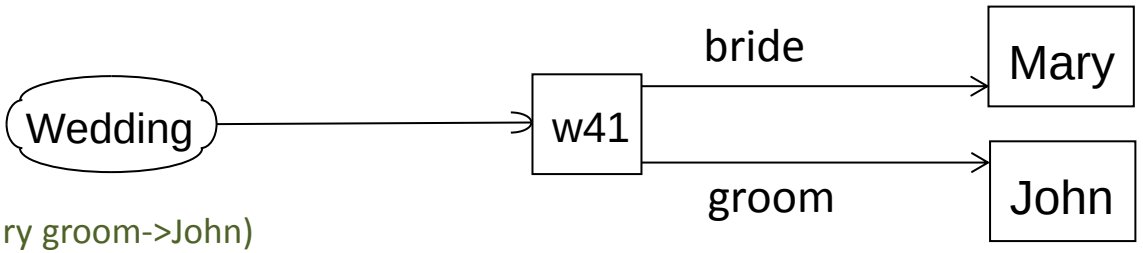
w21#Wedding(-[Mary John])

pn3. frameships



Wedding(bride->Mary groom->John)

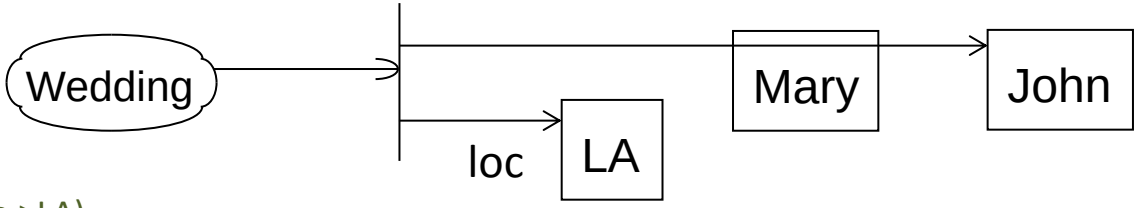
pn4: **framepoints**



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

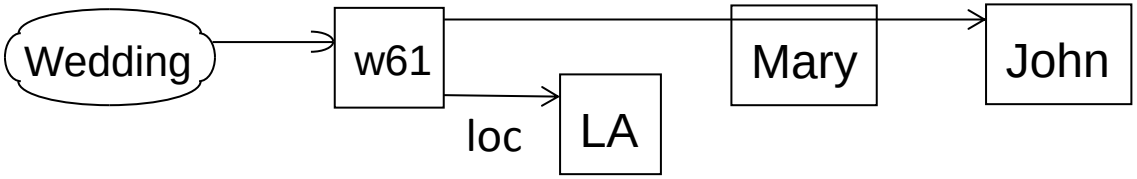
Extra oidless/oidful, combined tupled+slotted atoms that are **p**erspe**n**eutral:

pn5. single-tuple: shelframeships



Wedding(-[Mary John] loc->LA)

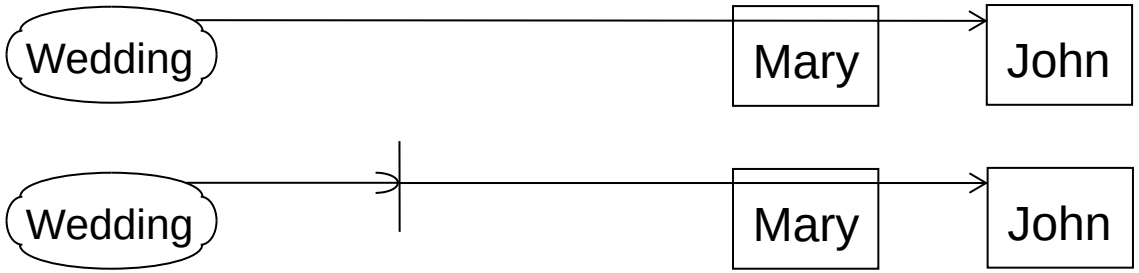
pn6. single-tuple: shelframepoints



w61#Wedding(-[Mary John] loc->LA)

Core oidless/oidful, tupled/slotted atoms that are **p**erspecti**v**al:

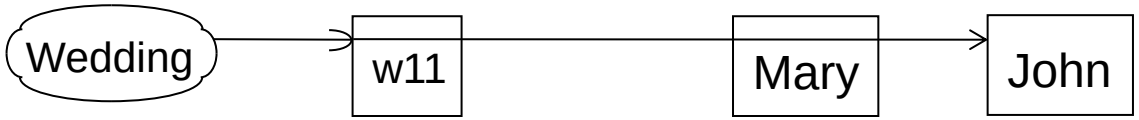
pv1. single-tuple: **relationships**



or

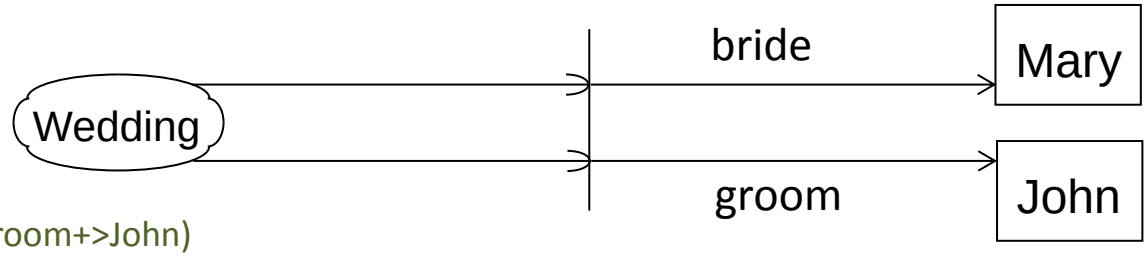
Wedding(Mary John) *or* Wedding(+[Mary John])

pv2. single-tuple: relationpoints



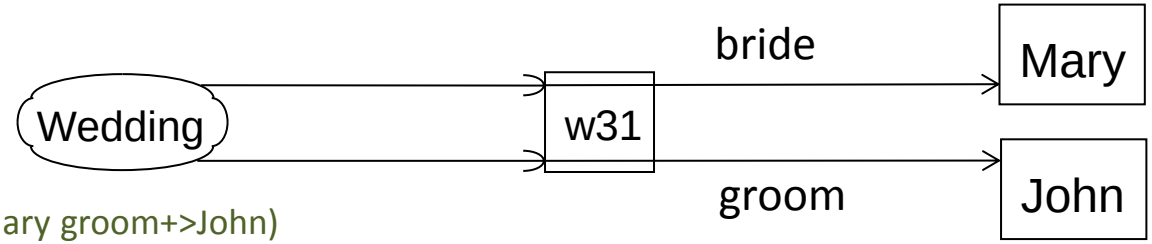
w11#Wedding(+[Mary John])

pv3: pairships



Wedding(bride+>Mary groom+>John)

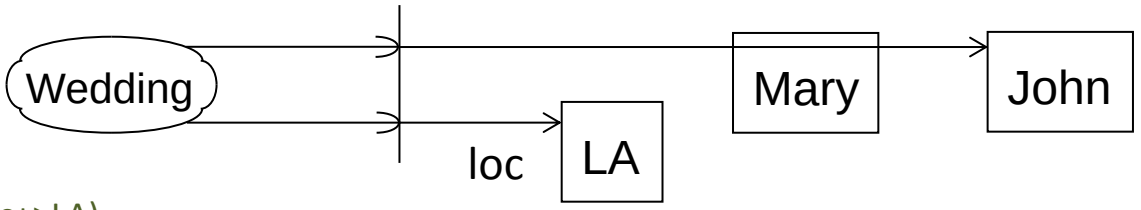
pv4. pairpoints



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

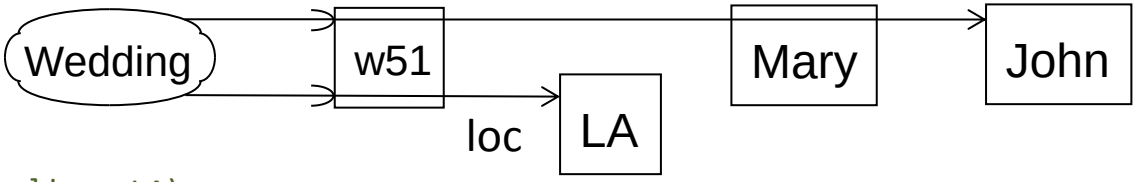
Extra oidless/oidful, combined tupled+slotted atoms that are **p**erspecti**v**al:

pv5. single-tuple: relpairships



Wedding(+[Mary John] loc+>LA)

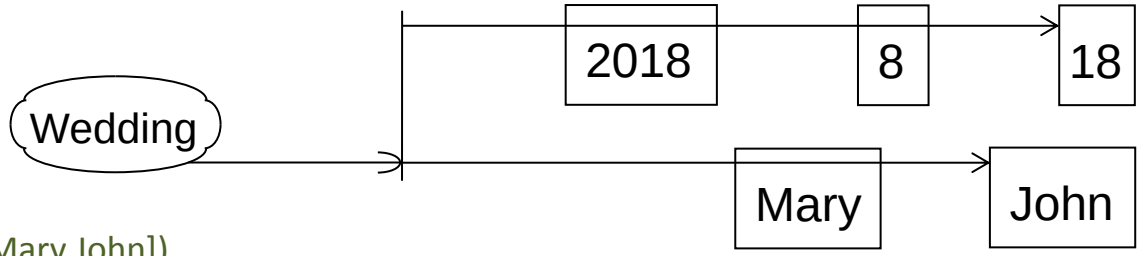
pv6. single-tuple: relpairpoints



w51#Wedding(+[Mary John] loc+>LA)

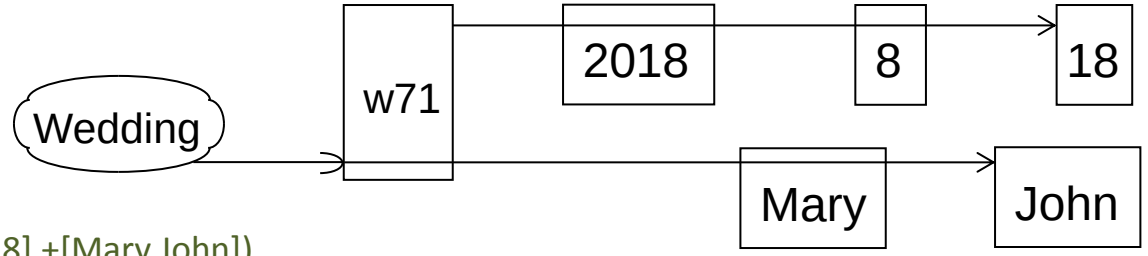
Adding oidless/oidful, tupled/slotted, combined **p**erspeneutral+**p**erspectival atoms:

pp1



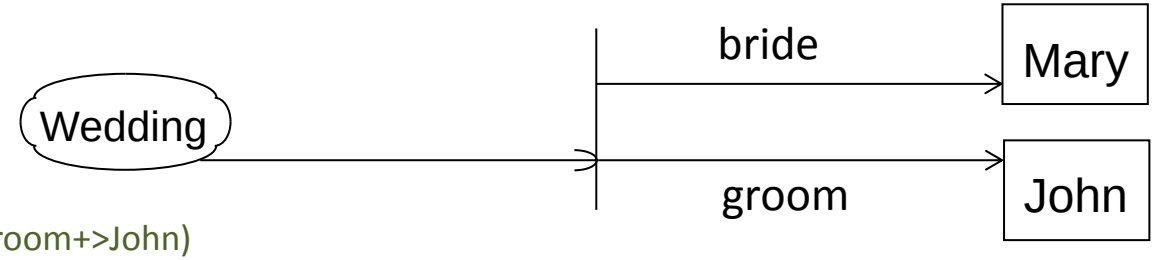
Wedding(-[2018 8 18] +[Mary John])

pp2



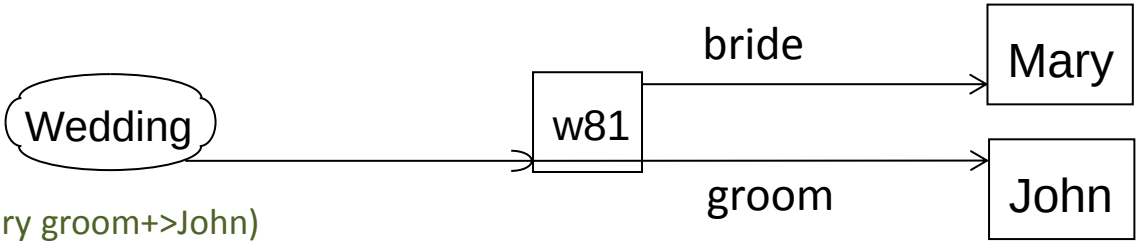
w71#Wedding(-[2018 8 18] +[Mary John])

pp3



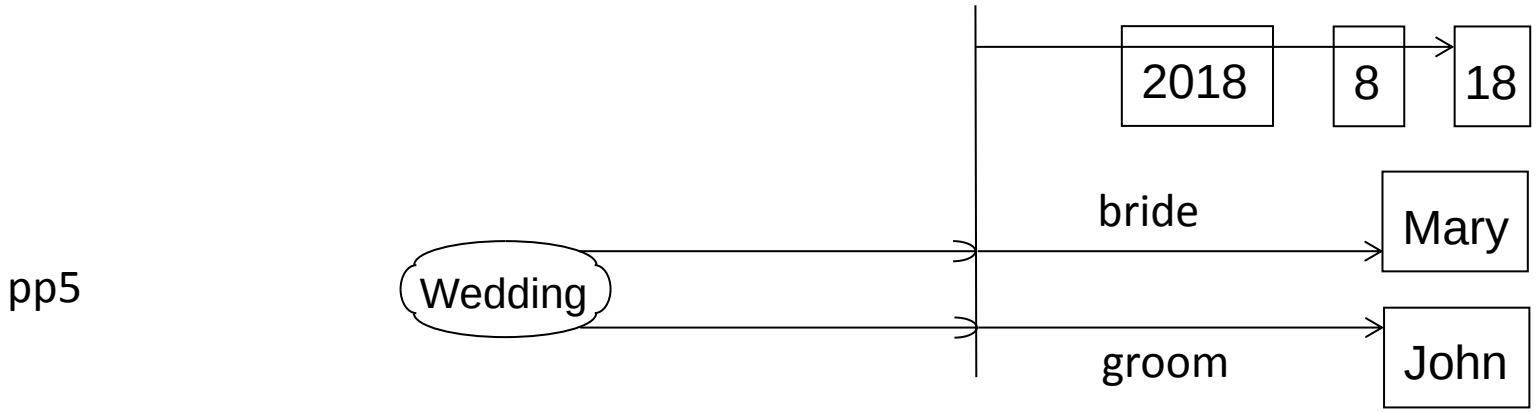
Wedding(bride->Mary groom+>John)

pp4

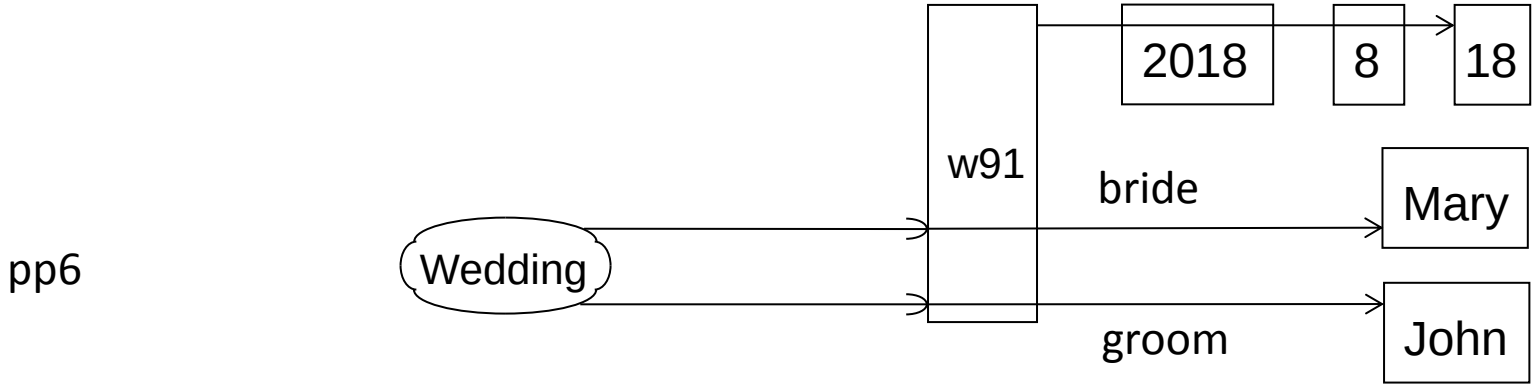


w81#Wedding(bride->Mary groom+>John)

Also oidless/oidful, combined tupled+slotted, combined **p**erspeneutral+**p**erspectival:



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 **p**erspe**n**eutral:

pn1. single-tuple: shelfships

f(-[t ... t] . . . -[t ... t])

Implicit existential OID; tuples -[t ... t] independent from predicate f

pn2. single-tuple: shelfpoints

o#f(-[t ... t] . . . -[t ... t])

o#f(-[t ... t])

Explicit OID o; tuples -[t ... t] independent from predicate f

pn3: frameships

f(p->v . . . p->v)

Implicit existential OID; slots p->v independent from predicate f

pn4: **framepoints**

o#f(p->v . . . p->v)

Explicit OID o; slots p->v independent from predicate f

Extra oidless/oidful, combined tupled+slotted atoms that are **p**erspe**n**eutral:

pn5. single-tuple:
shelframeships

f(-[t … t] . . . -[t … t] p->v . . . p->v)

Implicit existential OID; descriptors independent from predicate f

pn6. single-tuple:
shelframepoints

o#f(-[t … t] . . . -[t … t] p->v . . . p->v)

o#f(-[t … t] p->v . . . p->v)

Explicit OID o; descriptors independent from predicate f

Core oidless/oidful, tupled/slotted atoms that are **p**erspecti**v**al:

pv1. single-tuple:
relationships

f(+[t ... t] . . . +[t ... t])

f(t ... t) *or* f(+[t ... t])

Implicit existential OID; tuples +[t ... t] dependent on predicate f

pv2. single-tuple:
relationpoints

o#f(+[t ... t] . . . +[t ... t])

o#f(t ... t) *or* o#f(+[t ... t])

Explicit OID o; tuples +[t ... t] dependent on predicate f

pv3: pairships

f(p+>v . . . p+>v)

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

pv4: pairpoints

o#f(p+>v . . . p+>v)

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

Extra oidless/oidful, combined tupled+slotted atoms that are **p**erspecti**v**al:

pv5. single-tuple:
relpairships

f(+[t … t] . . . +[t … t] p+>v . . . p+>v)

f(+[t … t] p+>v . . . p+>v)

Implicit existential OID; descriptors dependent on predicate f

pv6. single-tuple:
relpairpoints

o#f(+[t … t] . . . +[t … t] p+>v . . . p+>v)

Explicit OID o; descriptors dependent on predicate f

Adding oidless/oidful, tupled/slotted, combined **p**erspeneutral+**p**erspectival atoms:

pp1

```
f(+[t ... t] . . . +[t ... t]
  -[t ... t] . . . -[t ... t])
```

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

pp2

```
o#f(+[t ... t] . . . +[t ... t]
      -[t ... t] . . . -[t ... t])
```

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

pp3

```
f(p+>v . . . p+>v
  p->v . . . p->v)
```

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

pp4

```
o#f(p+>v . . . p+>v
      p->v . . . p->v)
```

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

Also oidless/oidful, combined tupled+slotted, combined **p**erspeneutral+**p**erspectival:

pp5

```
f(+[t ... t] . . . +[t ... t]
  -[t ... t] . . . -[t ... t]
  p+>v . . . p+>v
  p->v . . . p->v)
```

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

pp6

```
o#f(+[t ... t] . . . +[t ... t]
     -[t ... t] . . . -[t ... t]
     p+>v . . . p+>v
     p->v . . . p->v)
```

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

Conclusions

- PSOA metamodel cube visualized (dynamically by PSOAMetaViz) and atoms (e.g., data facts) in Grailog, significantly facilitate learning PSOA RuleML
- Facts complemented by (interoperation, …) rules:
  http://wiki.ruleml.org/index.php/PSOA_RuleML_Bridges_Graph_and_Relational_Databases
  (includes core interoperation path pv1-pv3-pv4-pn4, e.g. abridged to one PSOA rule)
- PSOA RuleML being standardized by Relax NG schemas for XML-serialized facts and rules: http://wiki.ruleml.org/index.php/PSOA_RuleML#Syntax
- PSOA metamodel transferrable to other languages
- Also see: http://wiki.ruleml.org/index.php/PSOA_RuleML_Bridges_Graph_and_Relational_Databases#Conclusions