

# Semantic Web Rules

## - Tools and Languages -

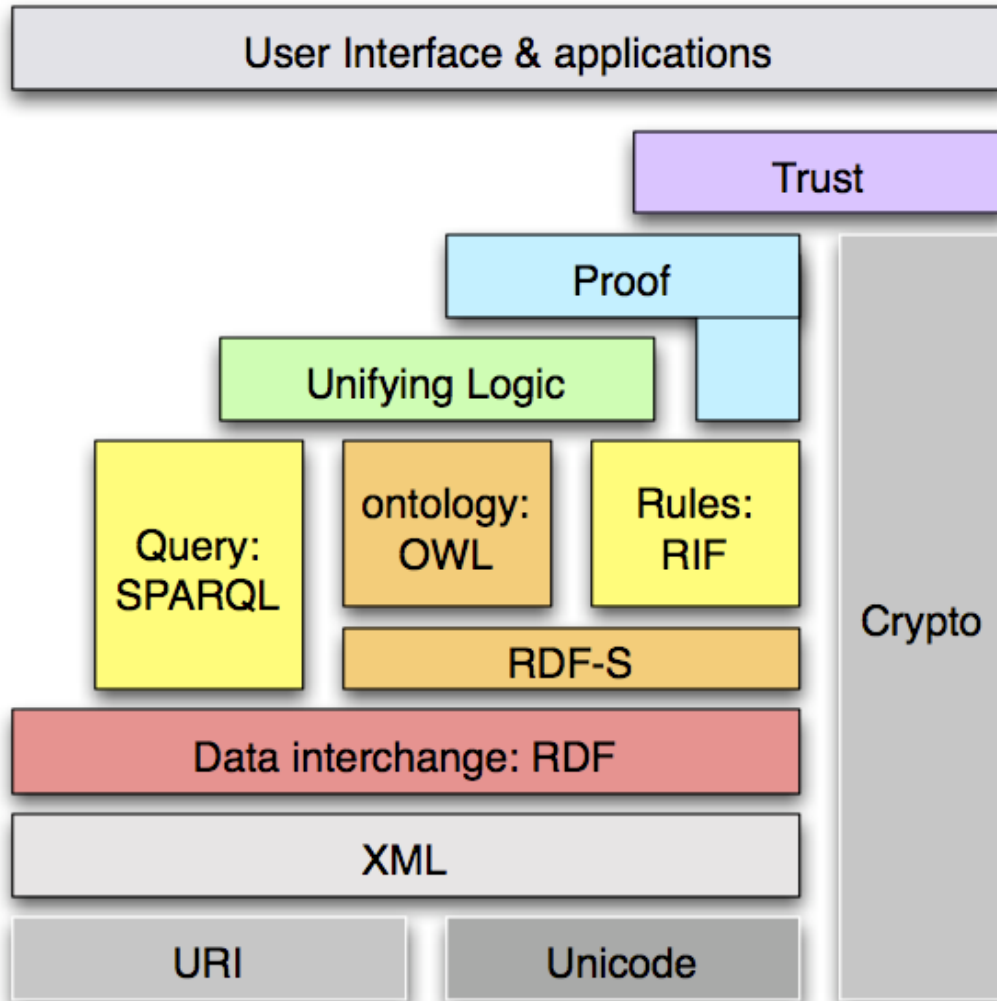
Tutorial at Rule ML 2006, Athens, GA  
Holger Knublauch



**TopQuadrant**

helping enterprises envision, architect and plan knowledge-based systems

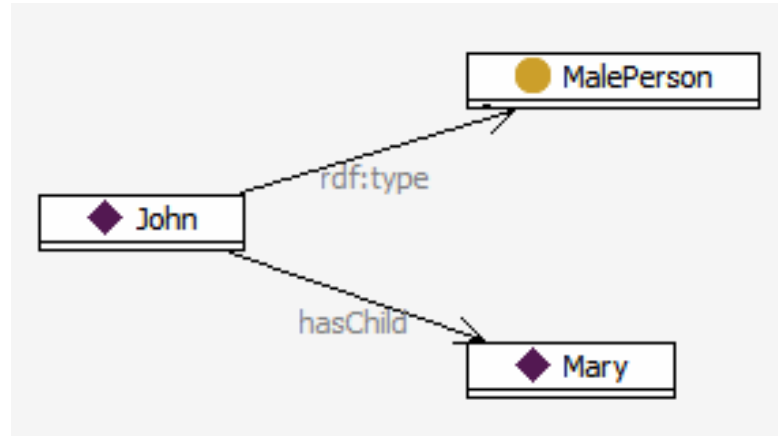
# Semantic Web



# Languages

- RDF Schema
  - OWL
  - SWRL
  - Jena Rules Language
  - SPARQL
- 
- RDF Triples are the common foundation

# RDF Graphs and Triples



Subject	Predicate	Object
John	rdf:type	MalePerson
John	hasChild	Mary

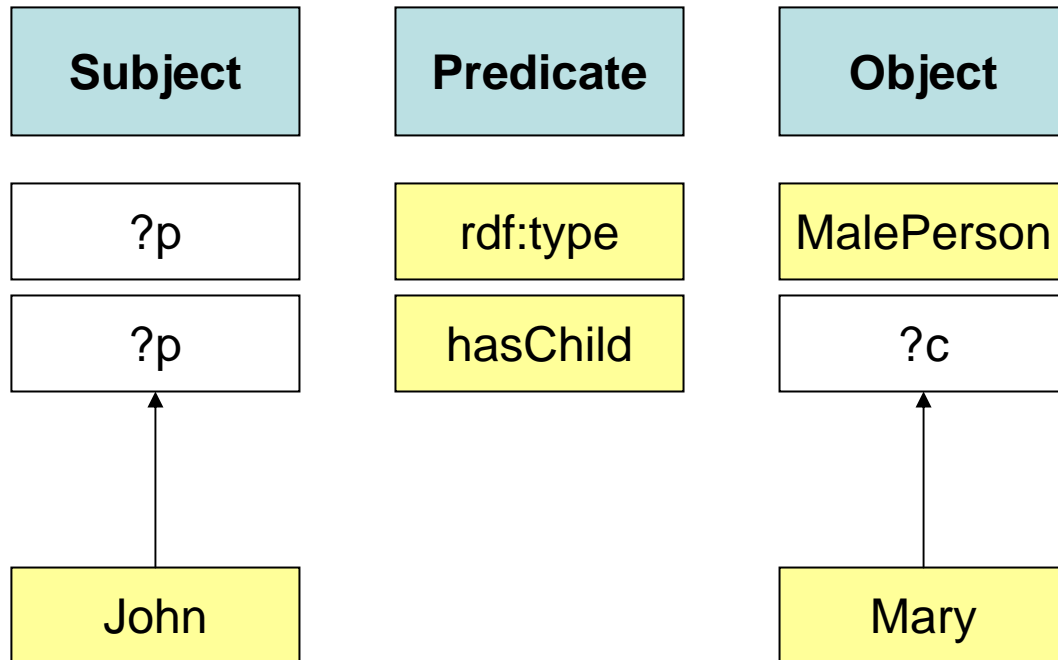
RDF/XML Serialization:

```
<MalePerson rdf:ID="John">  
  <hasChild rdf:resource="#Mary"/>  
</MalePerson>
```

N3/Turtle Serialization:

```
:John a      :MalePerson ;  
      :hasChild :Mary .
```

# Triple Pattern Matching



# Rules & Triples

- Execution of rules infers new triples

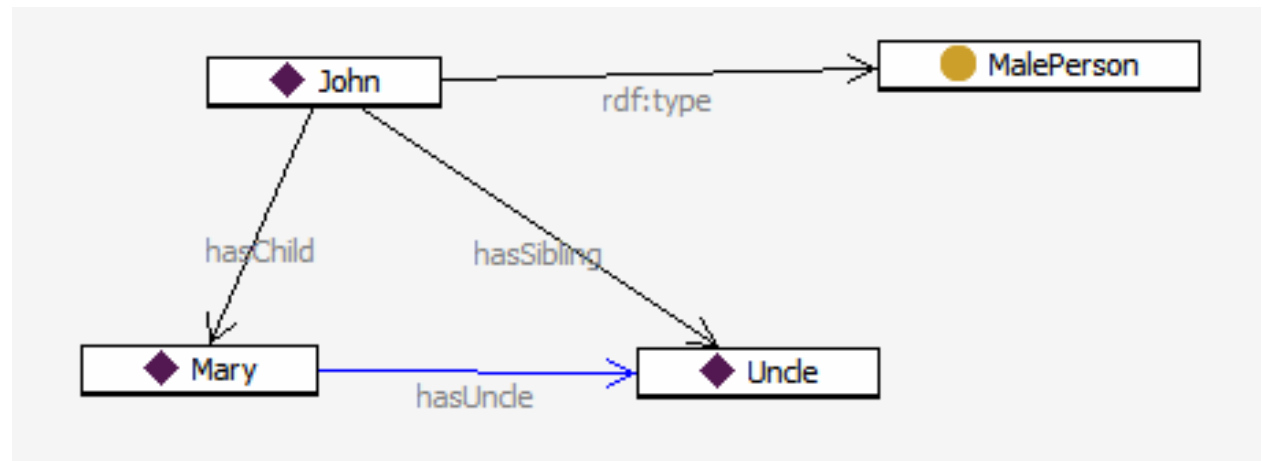
[defineUncle:

(?p :hasChild ?c)

(?p :hasSibling ?s)

(?s rdf:type :MalePerson)

-> (?c :hasUncle ?s)]



# Components of a Rule

- Triple patterns – like a triple, but with some named variables instead of fixed parts
  - ?company rdf:type :MajorCompany
  - Fortune500 :lists ?company
  - ?company :hasCEO ?person
- Rule “Body”
  - Set of triple patterns, all of which must match
  - Each variable must be ‘bound’ to the same item at every occurrence

HP rdf:type :MajorCompany .  
Fortune500 :lists HP .  
HP :hasCEO Fiorina .
- Rule “Head”
  - Set of triple patterns that will be asserted, when the body matches
  - Variables in these patterns have values that were bound in the body

# Demos

- Tools
  - Protégé + JESS
  - TopBraid Composer + Jena
- Example use cases
  - Family relationships
  - Real estate business
  - Ontology Mapping



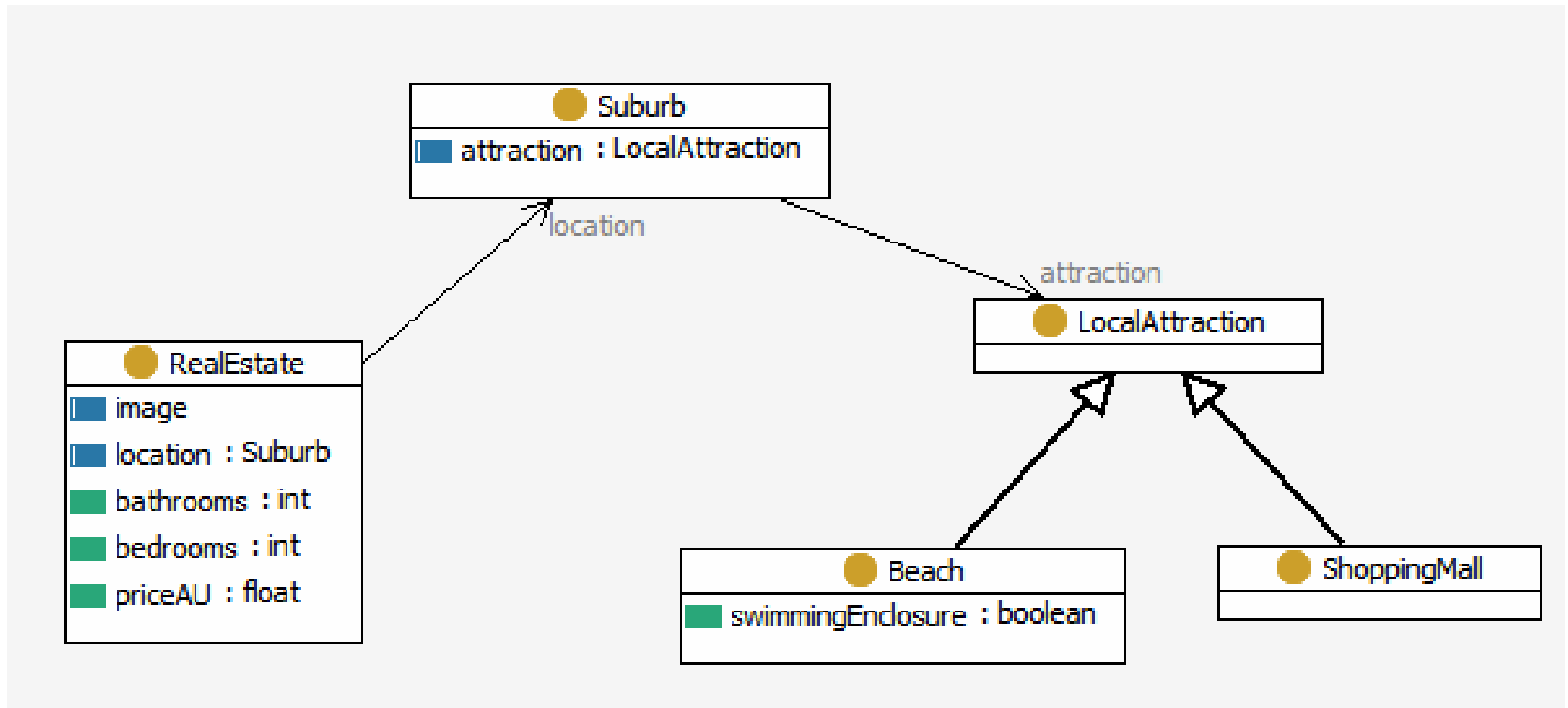
# Example Scenario

- Real Estate agents
  - “Database” of available properties
  - Properties are updated continuously
  - Customers have specific search patterns
  - The rule system shall notify the agent if a matching property has been added

# Design

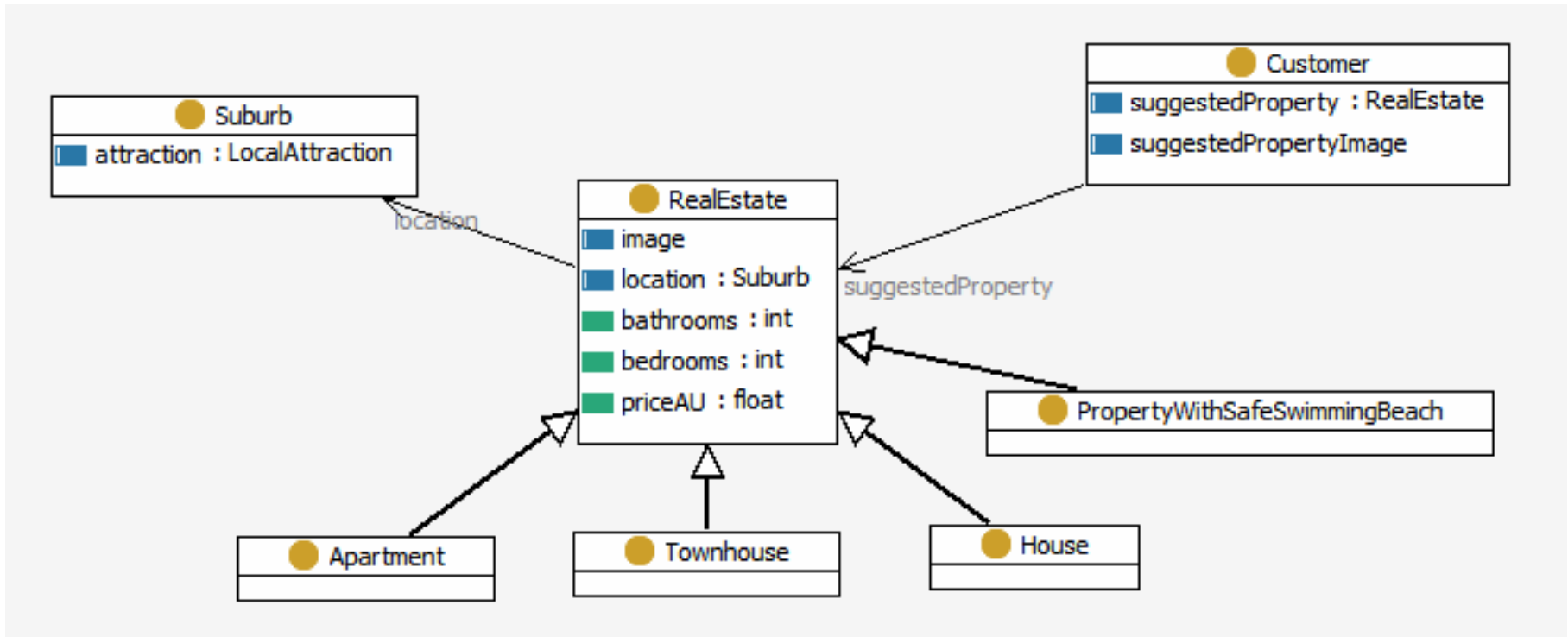
- OWL Ontology with domain concepts
  - Real Estate Properties
  - Characteristics of properties
  - Suburbs
  - Local attractions of the suburbs
- (Jena) Rules to drive matching

# Ontology Overview (1)



- Real Estate properties are located in Suburbs
- Suburbs have local attractions (Beaches etc)

# Ontology Overview (2)



- We have various types of Real Estate properties
- Properties are suggested to Customers

# Instance Database

The screenshot displays the TopBraid Eclipse SDK interface. The main window shows the 'Class Form' for the 'House' class, which is a subclass of 'RealEstate'. The 'Class Form' includes fields for Name, Annotations, Class Axioms (showing 'rdfs:subClassOf' with 'RealEstate' selected), and Other Properties. The 'Classes' view on the left shows a hierarchy of classes including 'owl:Thing (18)', 'Customer (3)', 'LocalAttraction (5)', 'owl:Nothing', 'RealEstate (5)', 'Apartment', 'House (5)', 'PropertyWithSafeSwimming', 'Townhouse', and 'Suburb (5)'. The 'Navigator' view at the bottom left lists various RDF files. The 'Geography' view on the right shows a map of Kewarra Beach with several house instances marked by red pins. The 'Instances' view at the bottom right shows a table of instances.

[Resource]	rdf:type	rdfs:label	rdfs:comment
◆ Smithfield-3-1	House		
◆ TrinityBeach-DesignerH...	House		
◆ TrinityBeach-TimberHou...	House		
◆ TrinityBeach-TimberHou...	House		
◆ TrinityPark-3-1	House		

# Example Instance

The screenshot displays the Eclipse IDE with the TopBraid plugin. The main workspace shows the instance details for 'TrinityBeach-TimberHouse-4-3'. The instance has the following properties:

- bathrooms**: 3
- bedrooms**: 4
- image**: <http://www.realestate.com.au/objects/props/6131/103386131mm1161144018.jpg>

The image property is visualized with a photograph of a modern timber house with a swimming pool. To the right, a map shows the location of the house in Trinity Beach, with a red pin marking the specific property. The map includes street names such as Victoria Close, Anderson St, and Moore St.

The bottom of the interface shows a table of instances:

[Resource]	rdf:type	rdfs:label	rdfs:comment
Smithfield-3-1	House		
TrinityBeach-DesignerH...	House		
TrinityBeach-TimberHou...	House		
TrinityBeach-TimberHou...	House		
TrinityPark-3-1	House		

# Rule 1: Convert Currencies

- Property prices are in Australian Dollars
- Customers may ask for prices in \$US

```
[convertAU2USDollar:  
    (?p :priceAU ?aud)  
    product(?aud 0.7745 ?usd)  
-> (?p :priceUS ?usd)]
```

# Rule 2: Simple Matching

- Customer Mike Turner is looking for a three-bedroom house

```
[findMatchesForMikeTurner:  
    (?p rdf:type :House)  
    (?p :bedrooms 3)  
-> (:MikeTurner :suggestedProperty ?p)]
```



# Rule 3: Matching

- Rebecca is looking for a property close to a shopping mall

[findMatchesForRebeccaSmith:

(?p :location ?l)

(?l :attraction ?a)

(?a rdf:type :ShoppingMall)

-> (:RebeccaSmith :suggestedProperty ?p)]

# Rule 4: Classification

- Find all properties that are located in a suburb that has a beach with a swimming enclosure

[findSafeSwimmingInstances:

(?p rdf:type :RealEstate)

(?p :location ?s)

(?s :attraction ?a)

(?a rdf:type :Beach)

(?a :swimmingEnclosure "true"^^xsd:boolean)

-> (?p rdf:type :PropertyWithSafeSwimmingBeach) ]

# Rule 5: Complex Matching

- John Doe is looking for a property with a safe swimming beach, at least 4 bedrooms and less than US\$ 900,000

[findMatchesForJohnDoe:

(?p rdf:type :PropertyWithSafeSwimmingBeach)

(?p :priceUS ?usd)

(?p :bedrooms ?b)

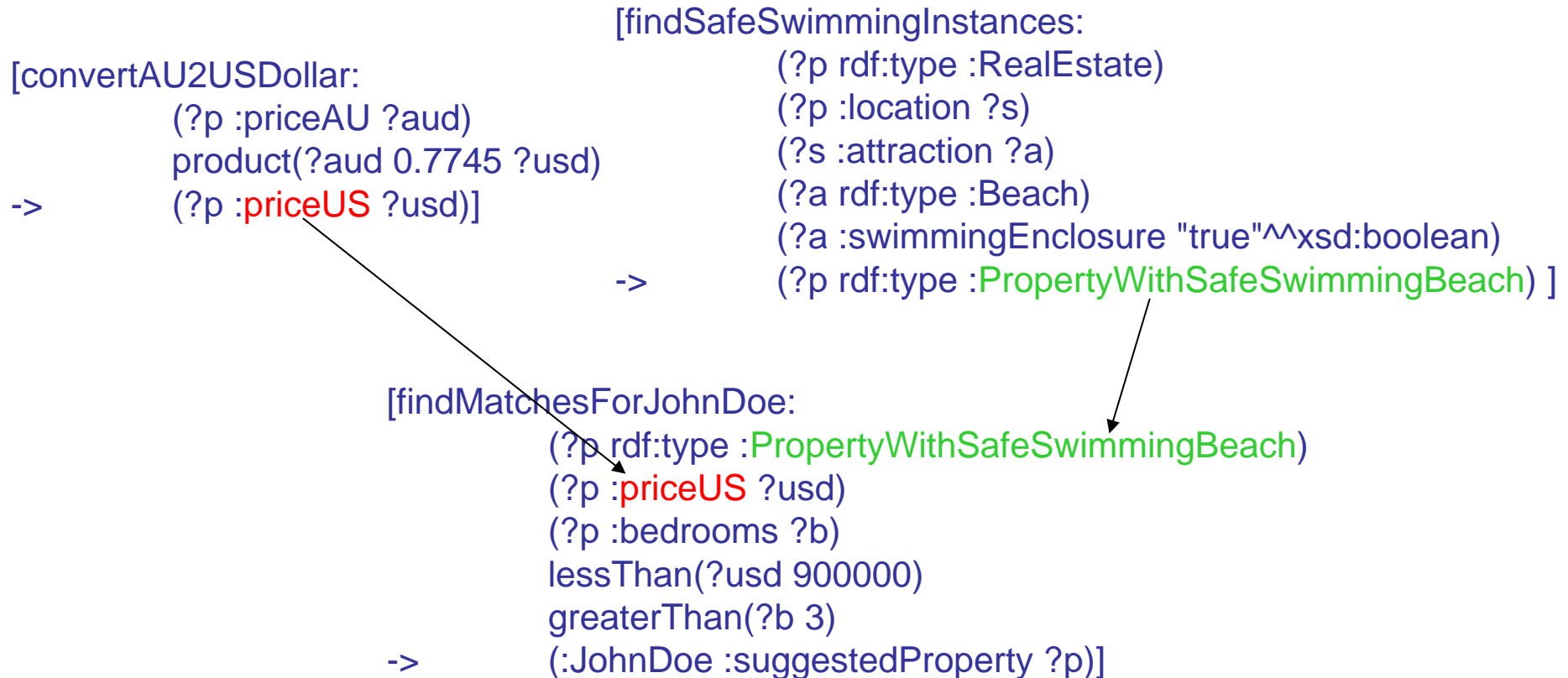
lessThan(?usd 900000)

greaterThan(?b 3)

-> (:JohnDoe :suggestedProperty ?p)]

# Rule Chaining

- Rule 5 depends on Rules 1 & 4



# Rule 6: Copying Values

- Whenever something is a suggestedProperty, then we want to copy its image into suggestedPropertyImage

[copyImages:

(?c :suggestedProperty ?p)

(?p :image ?i)

-> (?c :suggestedPropertyImage ?i)]

# Executing Rules

The screenshot displays the Eclipse IDE with the TopBraid plugin. The main window shows the 'realEstate.owl' ontology. The 'Resource Form' for 'JohnDoe' is open, showing a jenaRule that filters properties based on price, bedrooms, and swimming beach safety. The 'Inferences' table on the right lists the results of rule execution, showing various properties like 'suggestedProperty' and 'priceUS' for different entities. The 'Rules' table at the bottom lists the active rules and their expressions.

**Resource Form: JohnDoe**

**Annotations:**

```
jenaRule
[findMatchesForJohnDoe:
(?p rdf:type :PropertyWithSafeSwimmingBeach)
(?p :priceUS ?usd)
(?p :bedrooms ?b)
lessThan(?usd 900000)
greaterThan(?b 3)
-> (:JohnDoe :suggestedProperty ?p)]
```

**rdfs:comment:**

This customer is interested in a 3 (or more) bedroom property for less than \$US 900,000 and a safe swimming beach close by.

**Other Properties:**

- TrinityBeach-TimberHouse-4-2
- TrinityBeach-TimberHouse-4-3

**Inferences Table:**

[Subject]	Predicate	Object
JohnDoe	suggestedProperty	TrinityBeach-Timb...
JohnDoe	suggestedPropert...	<http://www.real...
JohnDoe	suggestedPropert...	<http://www.real...
JohnDoe	suggestedProperty	TrinityBeach-Timb...
MikeTurner	suggestedProperty	Smithfield-3-1
MikeTurner	suggestedPropert...	<http://www.real...
MikeTurner	suggestedPropert...	<http://www.real...
MikeTurner	suggestedProperty	TrinityBeach-Desig...
MikeTurner	suggestedPropert...	<http://www.real...
MikeTurner	suggestedProperty	TrinityPark-3-1
RebeccaSmith	suggestedPropert...	<http://www.real...
RebeccaSmith	suggestedProperty	Smithfield-3-1
Smithfield-3-1	priceUS	309800.00495910...
TrinityBeach-Design...	priceUS	413730.9361227751
TrinityBeach-Design...	rdf:type	PropertyWithSafe...
TrinityBeach-Timber...	rdf:type	PropertyWithSafe...
TrinityBeach-Timber...	priceUS	526660.008430481
TrinityBeach-Timber...	rdf:type	PropertyWithSafe...
TrinityBeach-Timber...	priceUS	772951.0123729706
TrinityPark-3-1	priceUS	294310.0047111511
dc:date	rdf:type	owl:DatatypeProp...
delanguage	rdf:type	owl:DatatypeProp...

**Rules Table:**

Name	Rule Expression
findMatchesForMikeTurner	(?p rdf:type :House) (?p :bedrooms '3'^^http://www.w3.o...
copyImages	(?c :suggestedProperty ?p) (?p :image ?i) -> (?c :suggeste...
convertAU2USDollar	(?p :priceAU ?aud) product(?aud '0.7745'^^http://www.w...
findMatchesForJohnDoe	(?p rdf:type :PropertyWithSafeSwimmingBeach) (?p :price...
findMatchesForRebeccaSmith	(?p :location ?l) (?l :attraction ?a) (?a rdf:type :ShoppingM...
findSafeSwimmingInstances	(?p rdf:type :RealEstate) (?p :location ?s) (?s :attraction ?...

# Browsing Suggestions

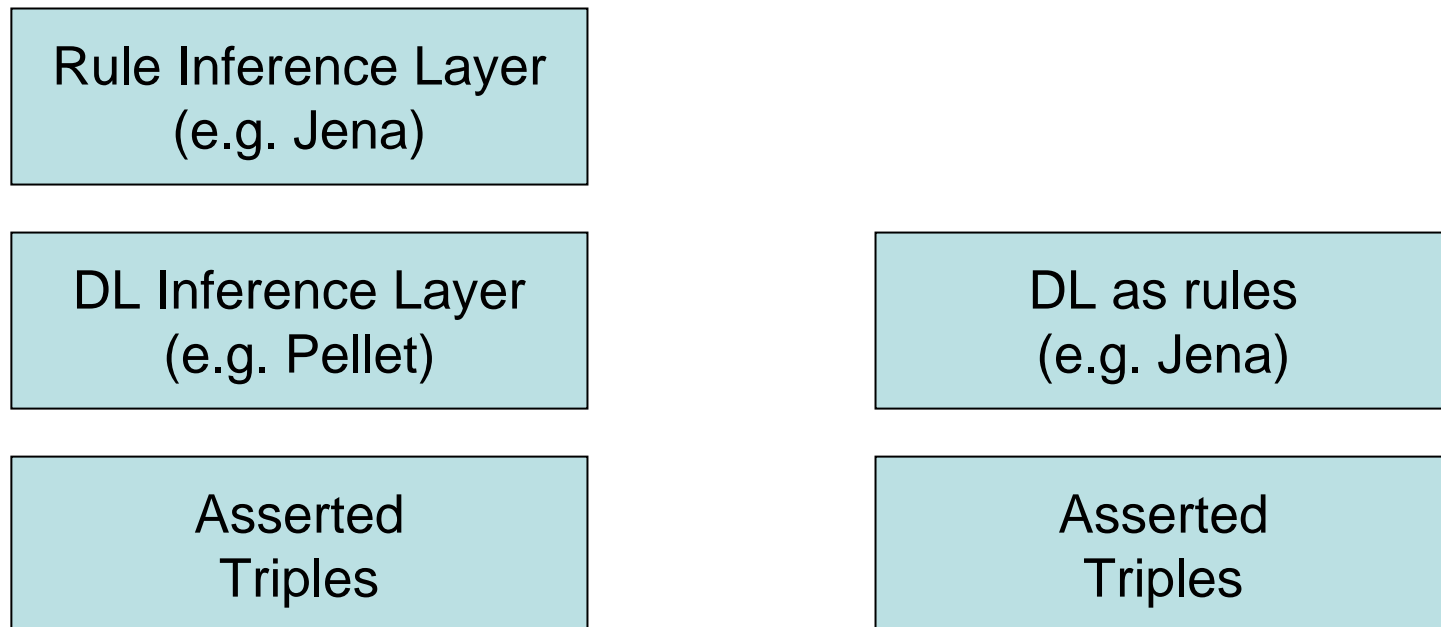
The screenshot displays the TopBraid Eclipse SDK interface. The main workspace is divided into several panes:

- Classes:** A tree view showing the ontology structure, including `owl:Thing` (18) and subclasses like `Customer` (3), `geo:Point`, `geo:SpatialThing`, `LocalAttraction` (5), `owl:Nothing`, `RealEstate` (5), `Apartment`, `House` (5), `PropertyWithSafeSwimming`, `Townhouse`, and `Suburb` (5).
- Navigator:** A list of RDF files, including `AtomOwl.rdf`, `foaf.rdf`, `FuGO.owl`, `galen-default.owl`, `icecream.owl`, `images.owl`, `oneOf.owl`, `reificationTest.owl`, `Thesaurus.owl`, `Thesaurus2.owl`, and `undeRule.owl`.
- realEstate.owl:** The main ontology editor, showing two images of a property: a house with a swimming pool and a house with a balcony.
- Map:** A Google Map of Trinity Beach, Australia, with several red location pins. The map is powered by Google and includes a search bar.
- Properties, Geography, Inferences:** Tabs for managing the application's data and logic.
- Rules:** A table listing various rules and their expressions.
- Basket:** A list of suggested properties, including `TrinityBeach-TimberHouse-4-2` and `TrinityBeach-TimberHouse-4-3`.

Name	Rule Expression
<input checked="" type="checkbox"/> findMatchesForMikeTurner	<code>(?p rdf:type :House) (?p :bedrooms '3'^^http://www.w3.o...</code>
<input checked="" type="checkbox"/> copyImages	<code>(?c :suggestedProperty ?p) (?p :image ?i) -&gt; (?c :suggeste...</code>
<input checked="" type="checkbox"/> convertAU2USDollar	<code>(?p :priceAU ?aud) product(?aud '0.7745'^^http://www.w...</code>
<input checked="" type="checkbox"/> findMatchesForJohnDoe	<code>(?p rdf:type :PropertyWithSafeSwimmingBeach) (?p :price...</code>
<input checked="" type="checkbox"/> findMatchesForRebeccaSmith	<code>(?p :location ?l) (?l :attraction ?a) (?a rdf:type :ShoppingM...</code>
<input checked="" type="checkbox"/> findSafeSwimmingInstances	<code>(?p rdf:type :RealEstate) (?p :location ?s) (?s :attraction ?...</code>

# OWL DL and Rules

- Rules can be executed “on top of” DL
- DL can be implemented by Rules





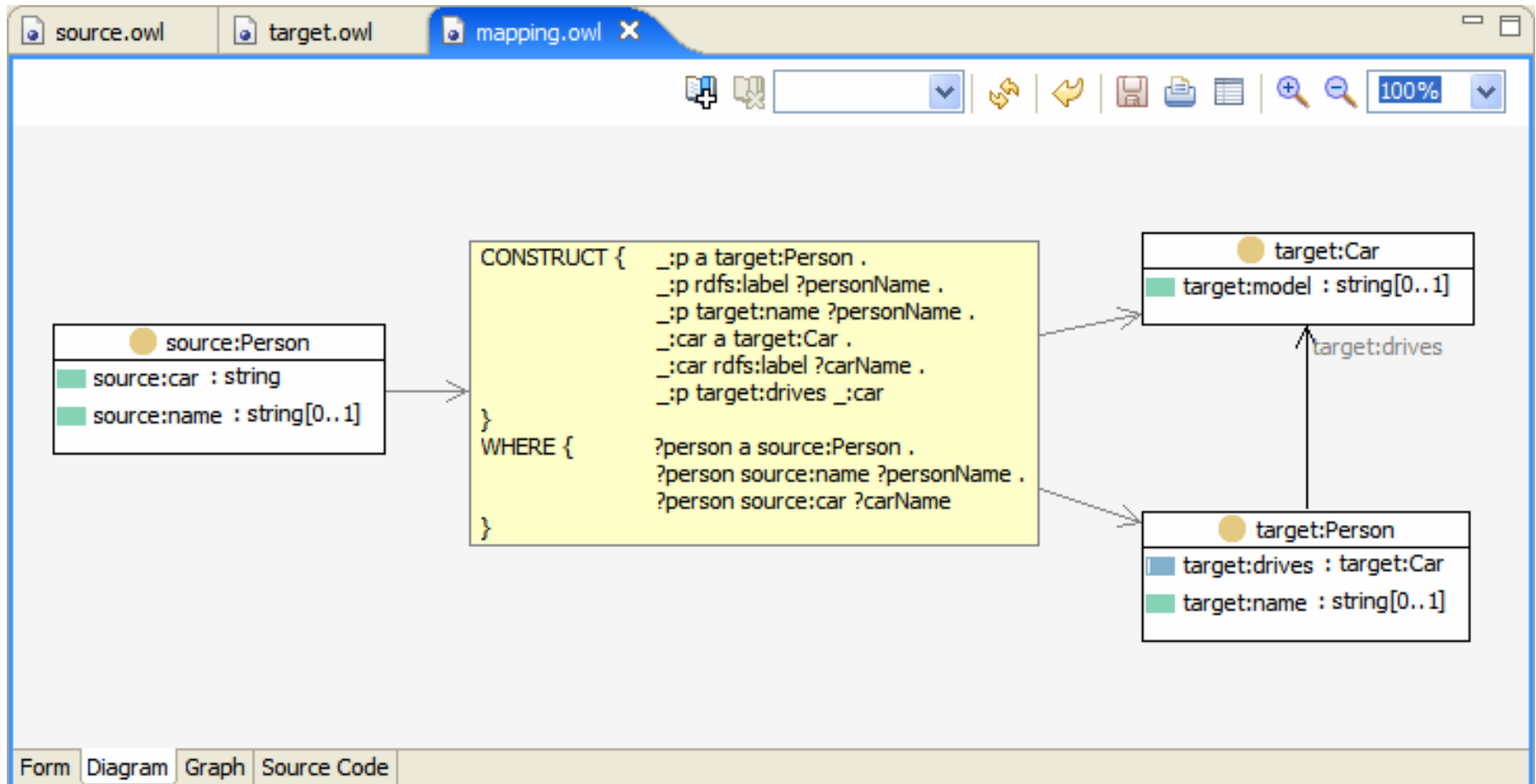
# OWL vs. Rules

<b>OWL</b>	<b>SWRL / RIF</b>
W3C Recommendation	Standard in Progress
Recent implementations	>20 years technology
Formal decidability	Possibility of Spaghetti code
Restriction language highly constrained	Powerful pattern language

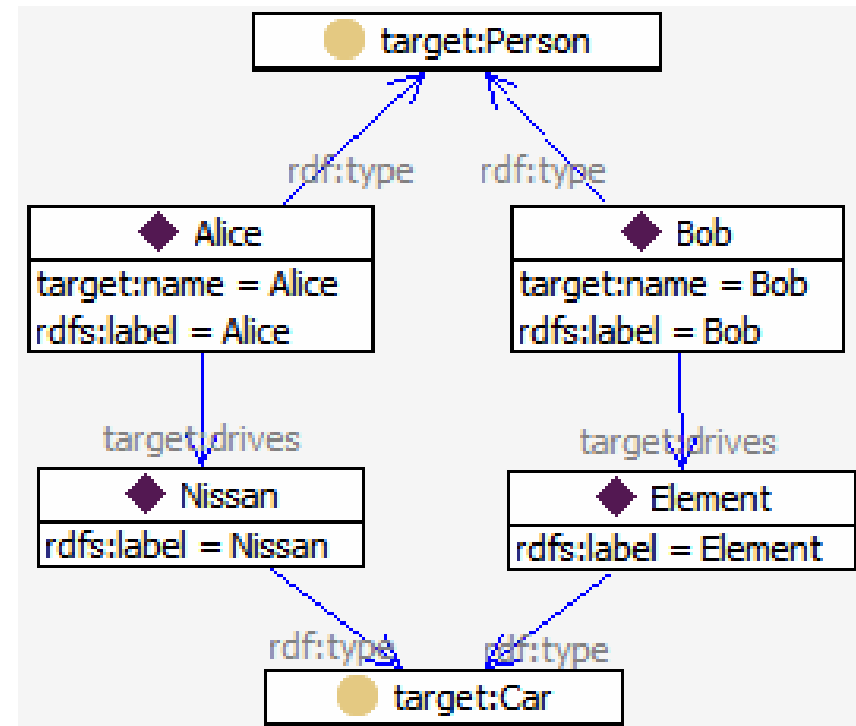
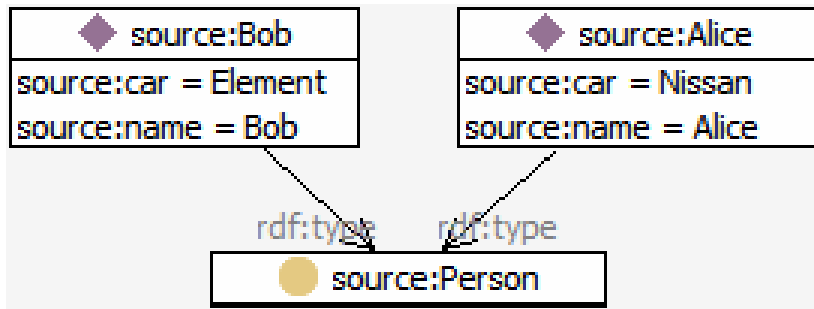
# SPARQL

- Not designed as a rule language
- W3C Standard query language for RDF
- Triple matching
  - SELECT
  - CONSTRUCT
- “Pragmatic” rule language

# Schema Mapping with SPARQL



# Schema Mapping (2)



# SPARQL and RULES

SPARQL	RULES (SWRL)
Complex patterns with ?variables	Complex patterns with ?variables
Defaults, options, boolean operations	AND only
Filters with math	SWRLb built-ins for math
Run under user/program control	chaining opportunistically
Optimized for a single query	Optimized for groups of rules