

Knowledge Discovery Framework for the Virtual Observatory

Brian Thomas, Edward Shaya, Zenping Huang,
Peter Teuben

Department of Astronomy,
University of Maryland,
College Park, MD

URL of Talk & Materials: <http://archive.astro.umd.edu/VOFramework>

The Problem

One of the fundamental reasons for the Virtual Observatory (VO) is that it federates many different data. Unfortunately:

Search for and fusion of VO data for scientific use still requires a human.

Diversity of description of these data make it difficult to remove the human from the loop of data discovery and fusion. Data may be held in various DB schema, file formats and semantics may differ.

An example of what the scientist is thinking is different from the actual process of locating and downloading and combining the data.

Finding Stars with measured IR magnitudes which observed inside a Spiral Galaxy arm.

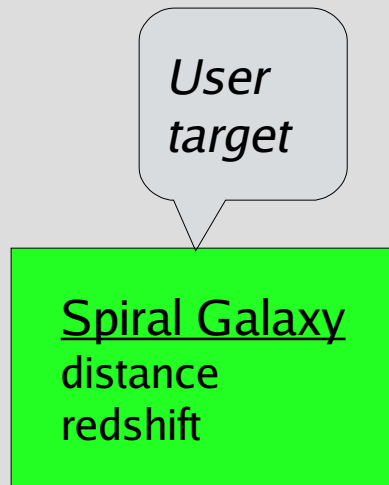
The Solution : Semantic Interoperability

Semantic interoperability : data have associated semantic (informational) meaning which may be ‘understood’ by the machine for the human.

Machine understanding implies the computer may:

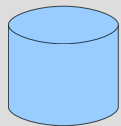
- ***Discover/infer information.***
 - * planetary data are about ‘pluto’ using an ephemeris;
 - * a star has properties which make it a ‘flare star’
- ***Extract/manipulate information.***
 - * Calculate new properties like line ratio, magnitudes, distance, from existing property sets.

Example of Semantic Interoperability



Desire: User wants to determine value of Hubble Constant for Spiral Galaxies.

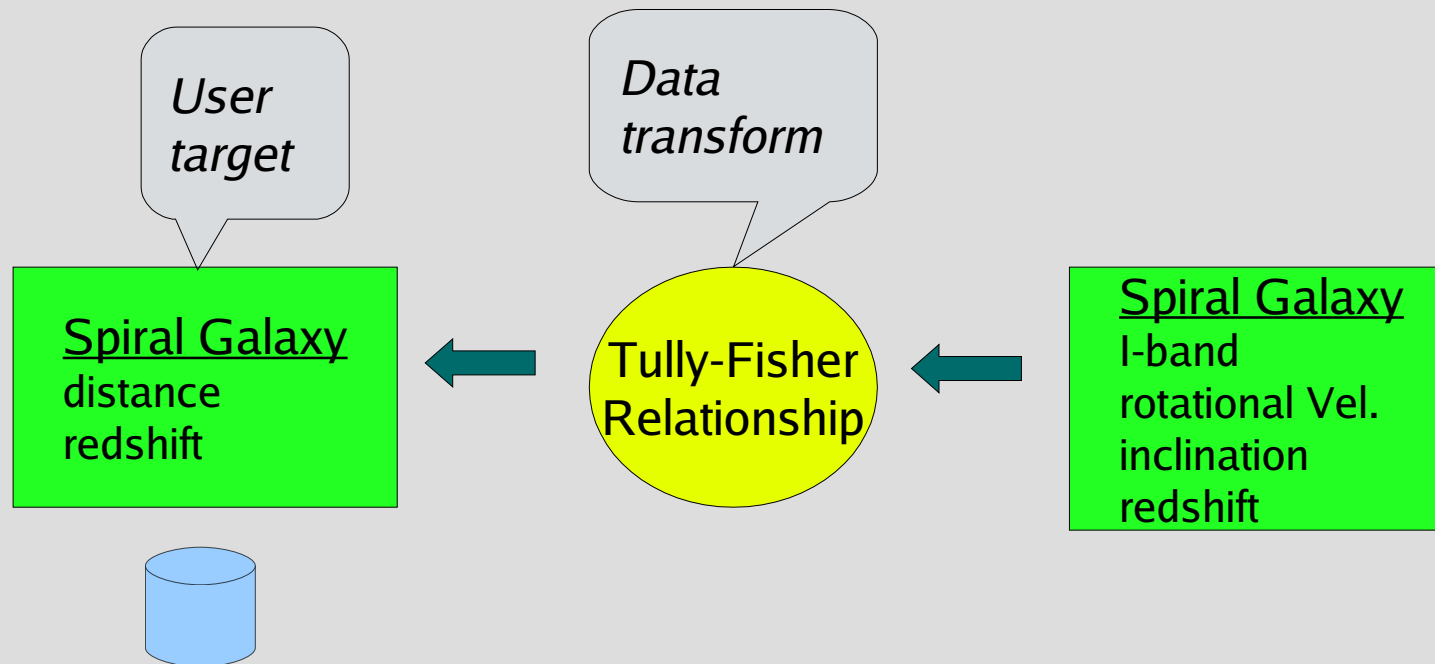
Example of Semantic Interoperability



These data are located
at only 1 archive.

**Problem: Data are unlikely to be in desired form,
only a few data repositories will directly match.**

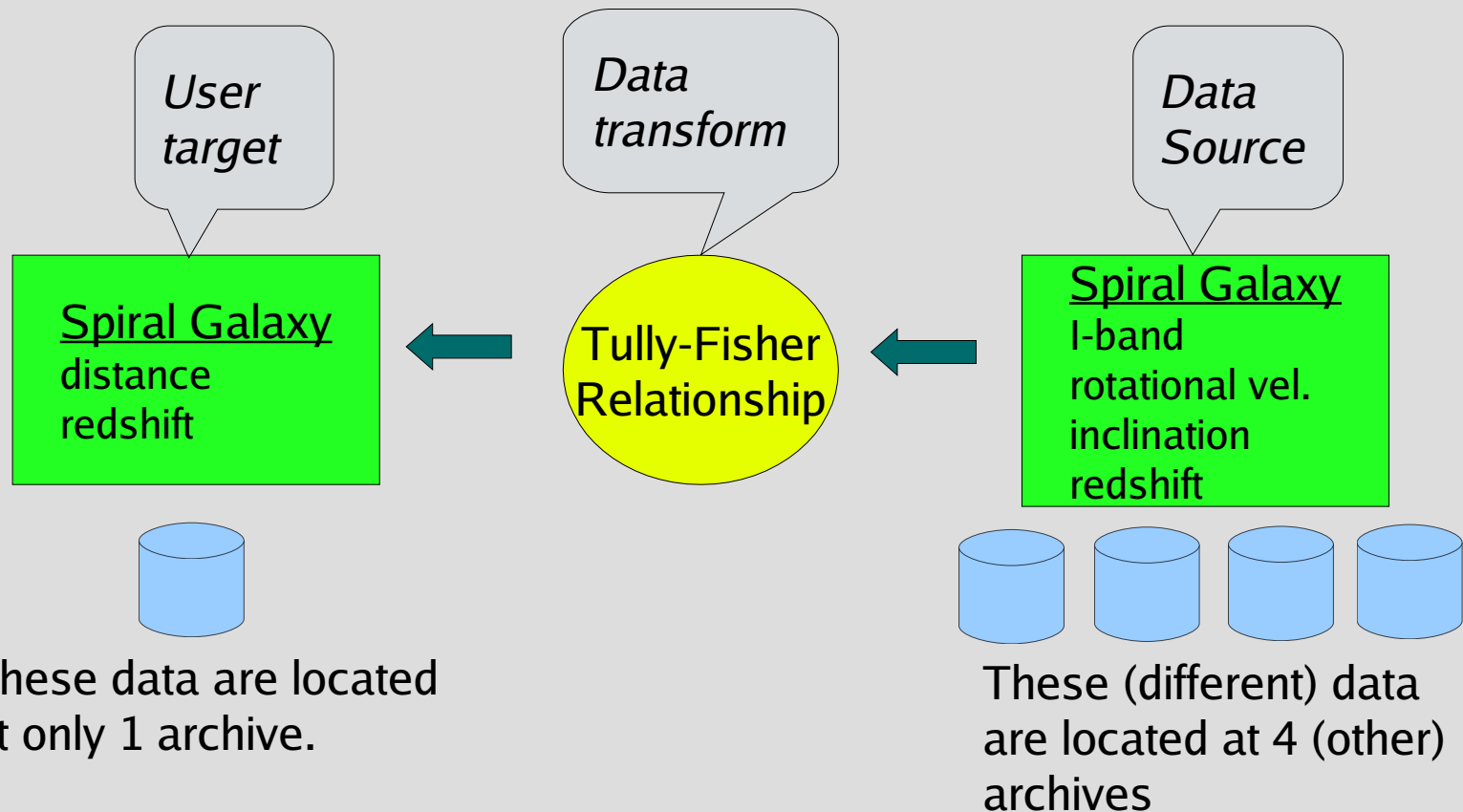
Example of Semantic Interoperability



These data are located
at only 1 archive.

**More data should be available: Information is held,
but in a different semantic form.**

Example of Semantic Interoperability



Outcome: Data are transformed *semantically*, and the user has increased dataset available.

Building blocks of a knowledge discovery framework

- ◆ Ontology and inference tools: allow the machine to understand relationships and information.
- ◆ Information layer:
 - Informational search
 - Description/model linking data <-> information
 - Informational transformation based on relationships in Ontology
- ◆ User tool: to aid in search and transformation. Ideally both graphical and scripting capabilities.

Ontology and Inference Tools

Good News: software has been written by others and is *available now and works.*

Ontology :(> 1000 objects, <http://archive.astro.umd.edu/ont>)

serialized in **OWL**

- * OWL based on first-order logic:
If then, sub-set, subclass, inheriting properties
- * OWL is W3C recommendation :

<http://www.w3.org/2004/OWL/>

Code support: in Java

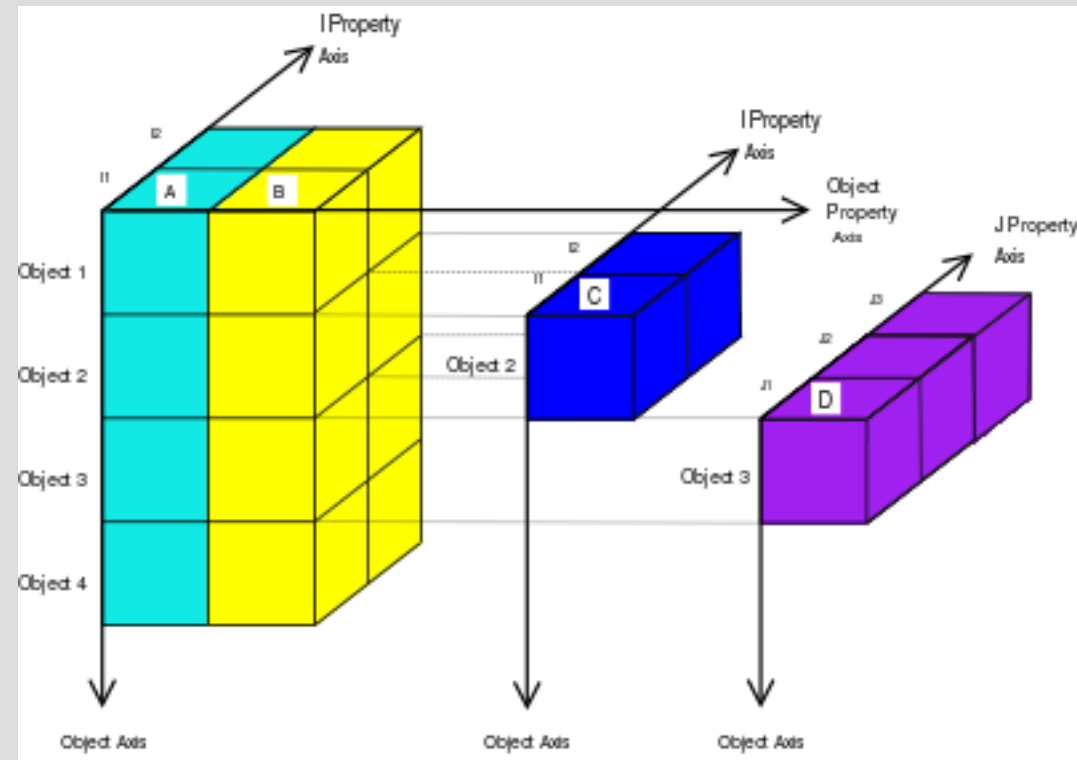
- * Java Framework : **Jena**
<http://jena.sourceforge.net>

- * Inference engine : **Pellet**
<http://www.mindswap.org/2003/pellet/>

Informational Layer : Data

◆ Bridging Data Model

- Maps between Ontology to existing DB, File data
- Can hold hierarchy of objects
- Can hold multidimensional properties
- Can hold a multitude of instances

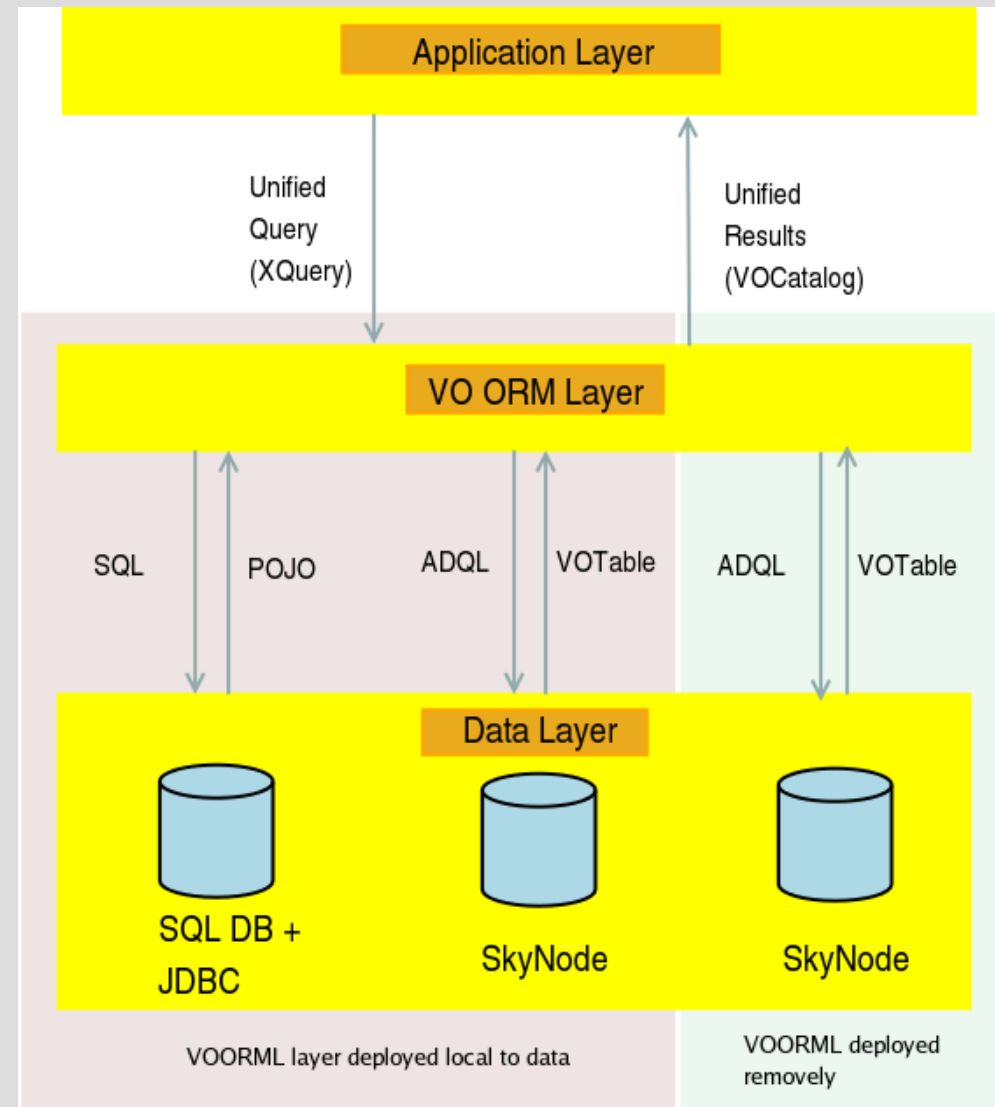


Implementation: JQML (Java Quantity) + Collection

Informational Layer : Query

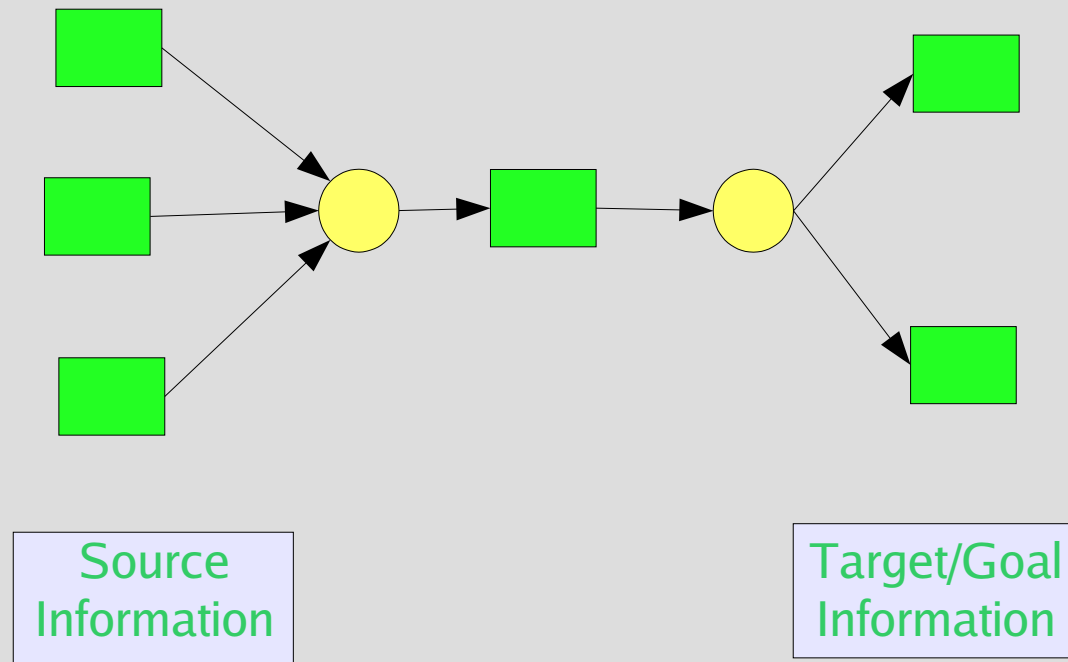
- ◆ Object-Relational Mapping Layer
 - Allows uniform query across many archives
 - Can leverage ADQL and VO infrastructure
 - Requires Bridging data model

Implementation: VO – ORM Layer (VOORML)



Informational Layer : Transform

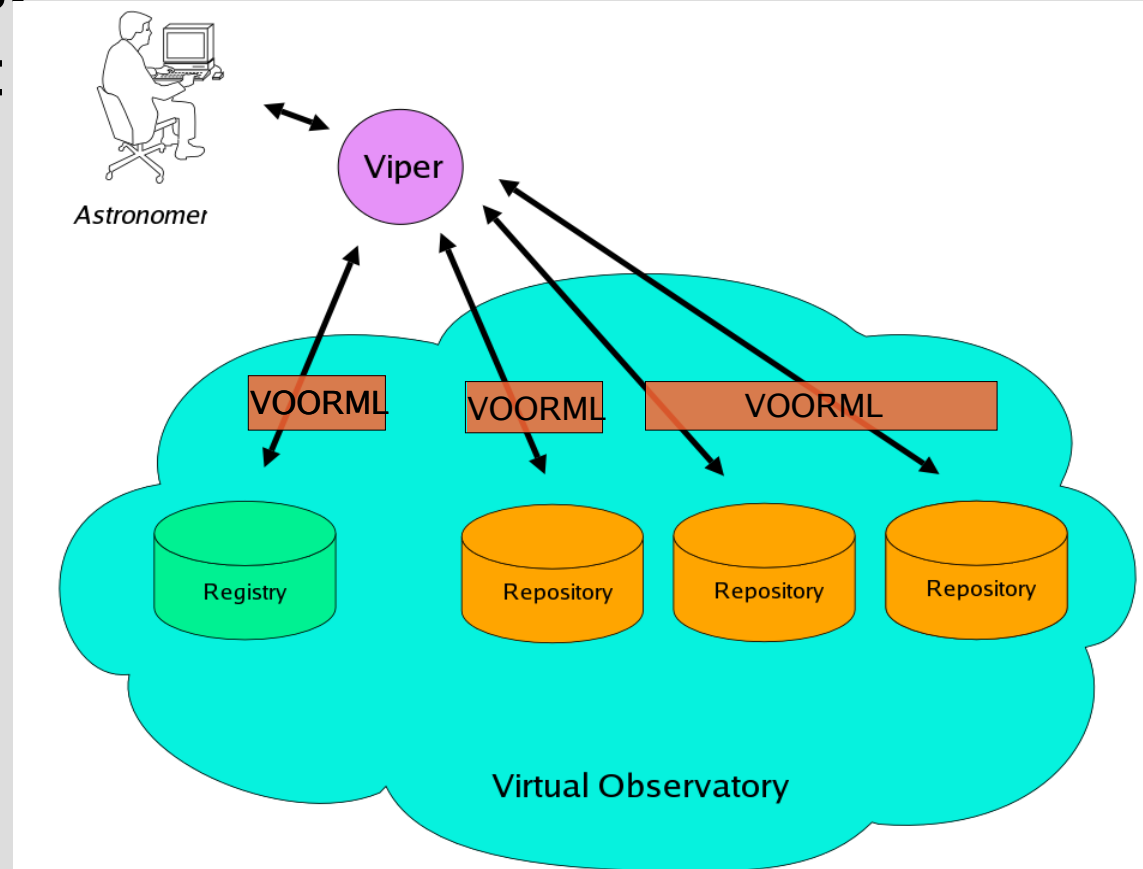
- ◆ Operates on semantic objects
- ◆ Wrap existing software
 - Both local libraries and web services/Grid
- ◆ Ops are mapped to relationships in ontology
 - Not just Astronomical
- ◆ Form a workflow



Implementation: Java Transform Package (JTP)

User Tool

- ◆ Provides an easy to use interface for user so that:
 - Can *make inferences*
 - Can *initiate searches*
 - Can *build complex workflows* which transform information



Implementation: Viper

VIPER Screen Shot

The screenshot displays the VIPER software interface. On the left, a tree view shows the ontology structure for `:TullyFisherRelation`. The `TullyFisherCode` node is highlighted in red. On the right, the `Query` tab is active, showing a query diagram with three stacked boxes connected by downward arrows. The top box is for `ast:SpiralGalaxy` with properties `ast:Inclination` and `ast:Galaxy.HI_LineWidth`. The middle box is for `:TullyFisherRelation` with properties `:TullyFisherCode`, `:SpiralGalaxyWithRotationVelocity.Maximum`, and `:SpiralGalaxyWithInclination`. The bottom box is for `ast:SpiralGalaxy` with properties `ast:Galaxy.HI_LineWidth`, `ast:Inclination`, and `ast:AbsoluteBrightnessOfGalaxy`.

File Help

Ont Class Ont Prop

:TullyFisherRelation

- relates
 - SpiralGalaxyWithLuminosity
- relates
 - SpiralGalaxyWithRotationVelocity.Maximum
- hasCode
 - TullyFisherCode**
- relates
 - SpiralGalaxyWithInclination
- hasPoleDirection
- hasLocation
- description
- hasProvenance
- name
- relates
 - hasDomain
 - hasCode
 - memberOf

Query Logic Data

Query Infer Save

ast:SpiralGalaxy

- ast:Inclination
- ast:Galaxy.HI_LineWidth

:TullyFisherRelation

- :TullyFisherCode
- :SpiralGalaxyWithRotationVelocity.Maximum
- :SpiralGalaxyWithInclination

ast:SpiralGalaxy

- ast:Galaxy.HI_LineWidth
- ast:Inclination
- ast:AbsoluteBrightnessOfGalaxy

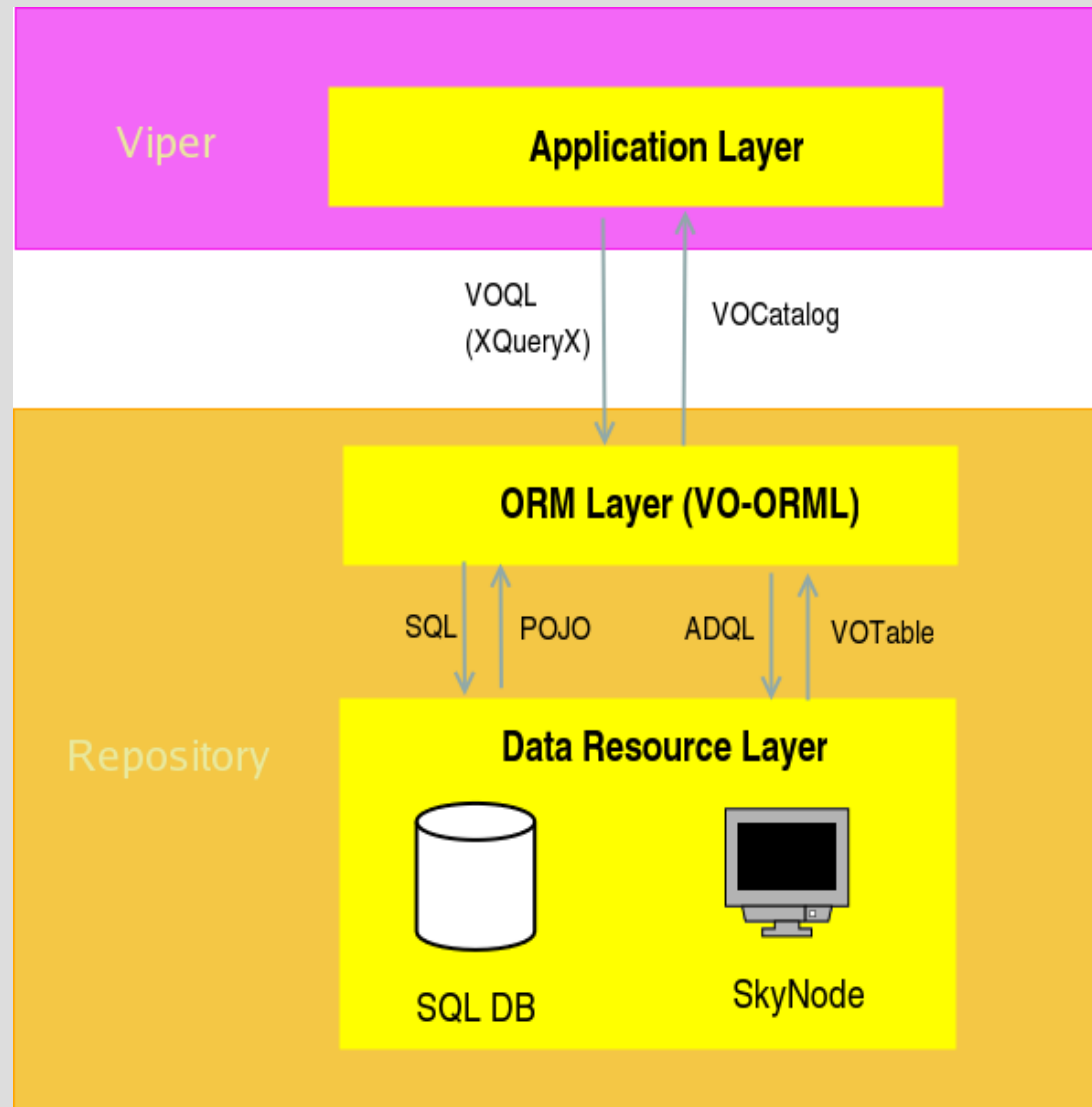
First select a tree node by clicking on it

Summary

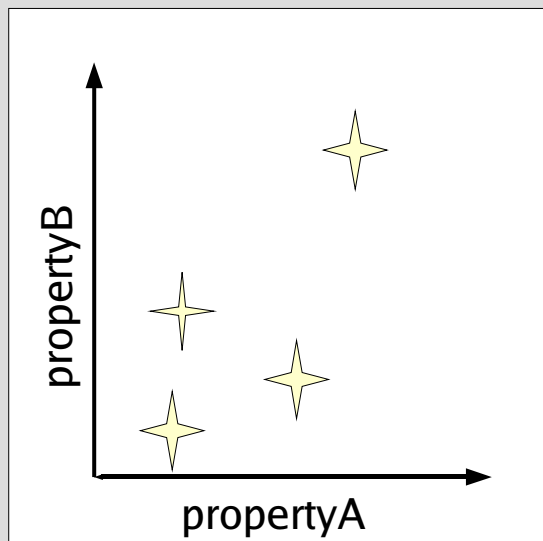
- ♦ ***We have been working on design and implementation of framework for KD in VO***
 - ***Basics only, but extensible.*** We can only provide basic configuration.
 - ***Progress of Components?***
 - ***Ontology*** – prototype working, needs refactor
 - ***VOORML*** - prototype working, needs refactor, astronomical functions, SQL only needs SkyNode
 - ***Data Model*** - prototype working, needs refactor
 - ***Transform Package*** – prototype **Dec 2006**, in design
 - ***Viper*** – limited prototype **Jan 2007**, alpha release for **Spring/Summer 2007**.

Spare Slides..

Viper-VOORML



Representation



*Graphical
Representation
(point == Object)*

Class of Object = 'Star'
propertyA = '0.321'
propertyB = '102.1'

*Semantic
Representation*

propertyA	propertyB
0.321	102.1
2.12	121.8
1.3233	99.3
9.325	2145.1

*Tabular
Representation
(row == Object)*

Both graphical and tabular data are 'objects' with properties.

Viper Framework

