GALAXY EXPRESS

SWCLOS

Semantic Web Processing in CLOS

Seiji Koide

National Institute of Informatics

IHI Corporation

Case Sensitive Lisp or Modern Lisp

ACL8.0 or 7.x in Windows
Start → Program → Allegro CL → Modern ACL Images
→ Allegro CL (w IDE, Modern)
DO NOT SELECT ANSI image IN WINDOW

SWCLOS requires Case Sensitive Mode of ACL.

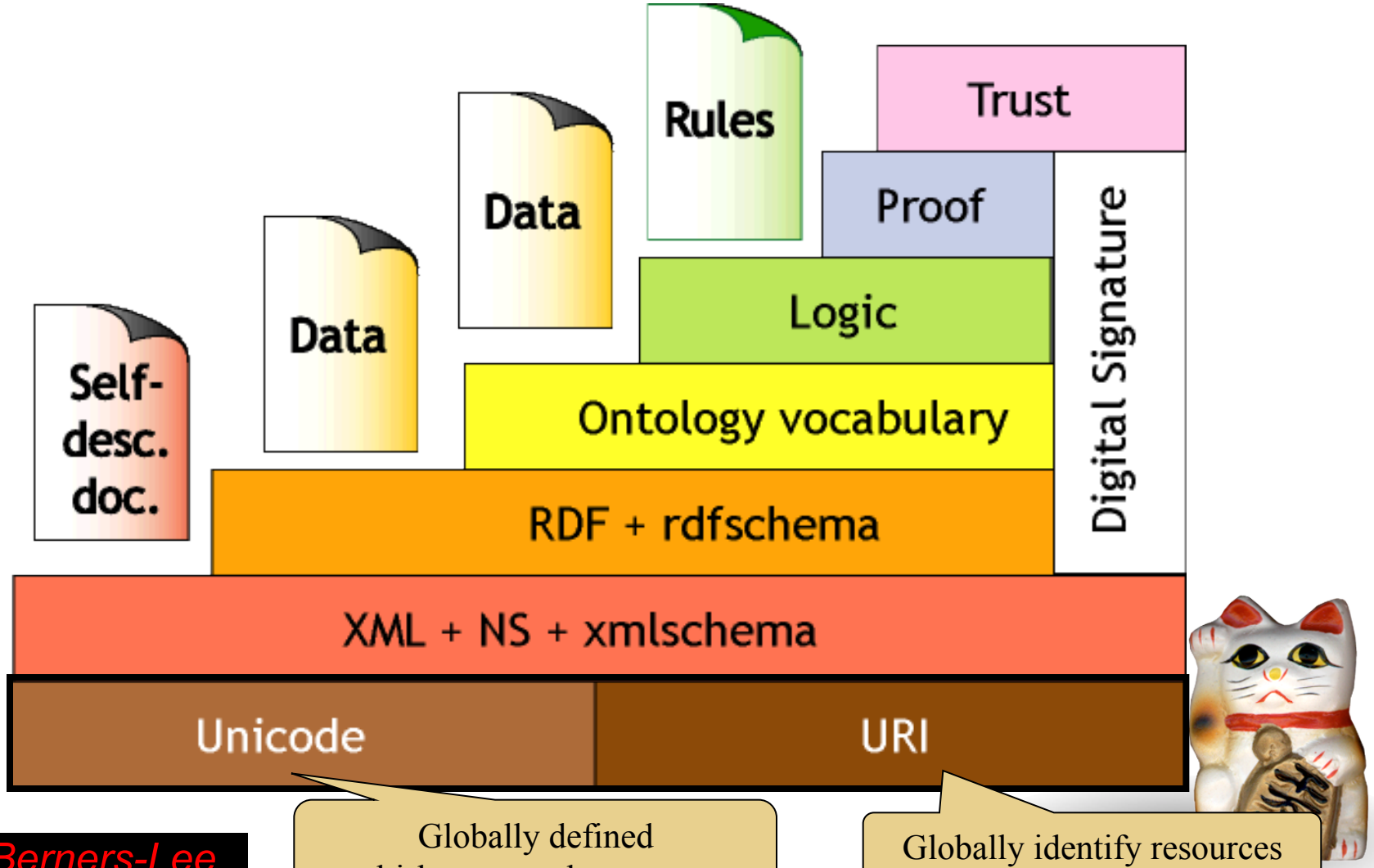
What is SWCLOS?

- Semantic Web Processor on CLOS
- Processing Ontology in RDFS and OWL
- Input Ontology in S-expression, RDF/XML, Triples
- Output Ontology in S-expression , RDF/XML, Triples
- An Amalgam of CLOS and RDFS/OWL
- Pros
  - A tool for CLOS programmers for Semantic Web programming
- Cons
  - Unavailable for C# and Java Programmers

Semantics in RDF/OWL

- Open World Assumption
  - `t` means true, but `cl:nil` does not mean false, it means either unknown or negation.
  - There may be another knowledge in the Web World.
- Monotonicity
  - The world is refined with assertions incrementally.
  - Impossible to retract asserted knowledge
- Class = Membership of Instances
  - An instance may belongs to multiple classes.
- Inherit slot-value type constraints, not slot values
  - A type facet value in the slot-definition is inherited.

Semantic Web Layer-cake



Tim Berners-Lee

UNICODE at Allegro Common Lisp

http://www.franz.com/support/documentation/7.0/doc/iacl.htm

Internally, all Lisp strings are represented as arrays of Unicode character codes. Each array element is exactly 16-bits wide, even if the string contains only 7-bit ASCII characters. This widening of strings causes a memory usage increase. However, since almost all initial Allegro CL strings are stored in memory-mapped files, the initial runtime memory usage difference between International Allegro CL and non-international Allegro CL is less than 5%.

```
Non-international (8-bit characters) Allegro CL:
 ) (char-code #\ł)
 163                    ;;; This is the (8-bit) Latin-2 code.

International Allegro CL:
 ) (char-code #\ł)
 321                    ;;; This is the Unicode code.
```

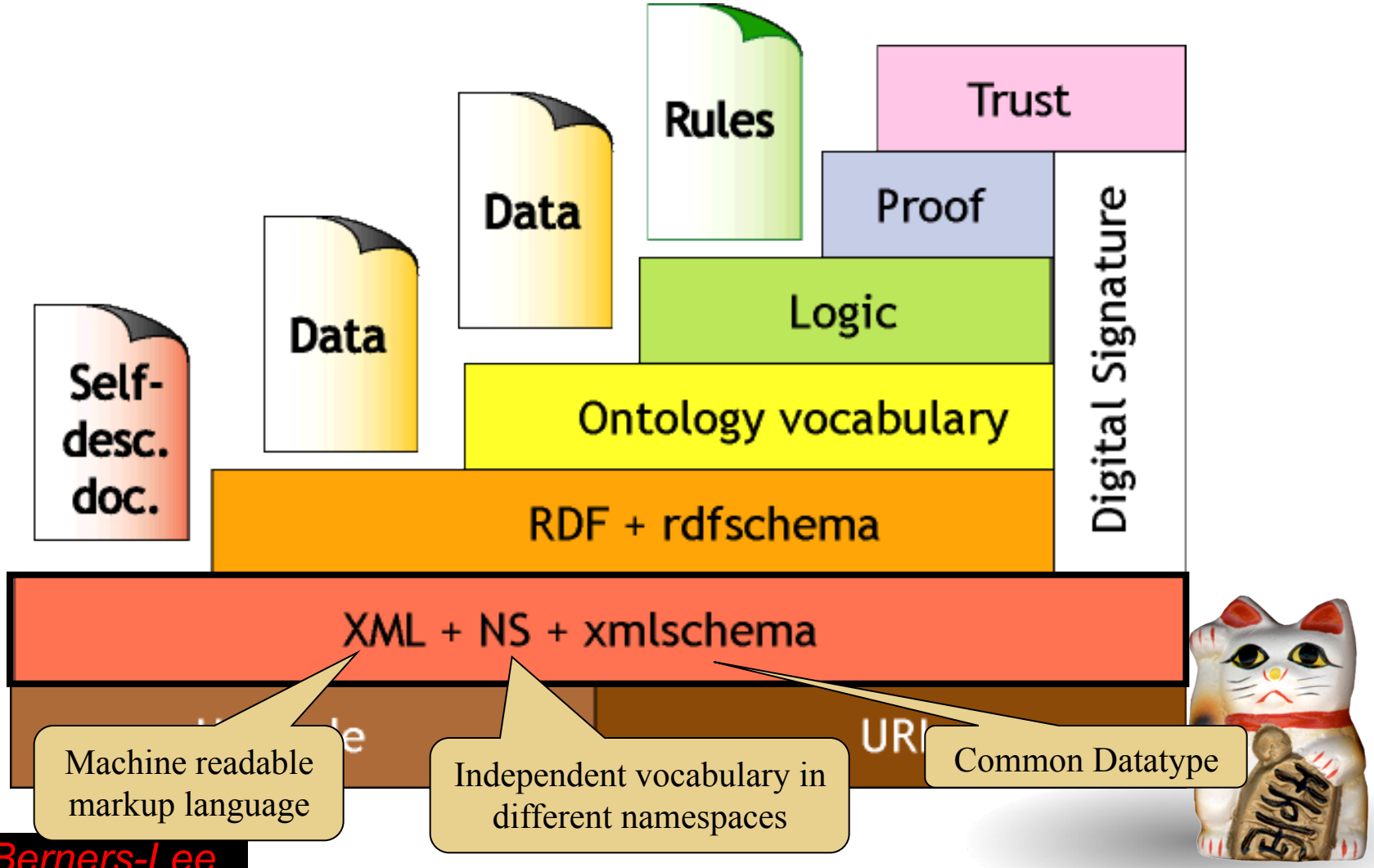
URI Library at Allegro Common Lisp

http://www.franz.com/support/documentation/7.0/doc/uri.htm

URI stands for *Universal Resource Identifier*. For a description of URIs, see RFC2396, which can be found in several places, including the IETF web site (http://www.ietf.org/rfc/rfc2396.txt) and the UCI/ICS web site (http://www.ics.uci.edu/pub/ietf/uri/rfc2396.txt).
We prefer the UCI/ICS one as it has more examples.

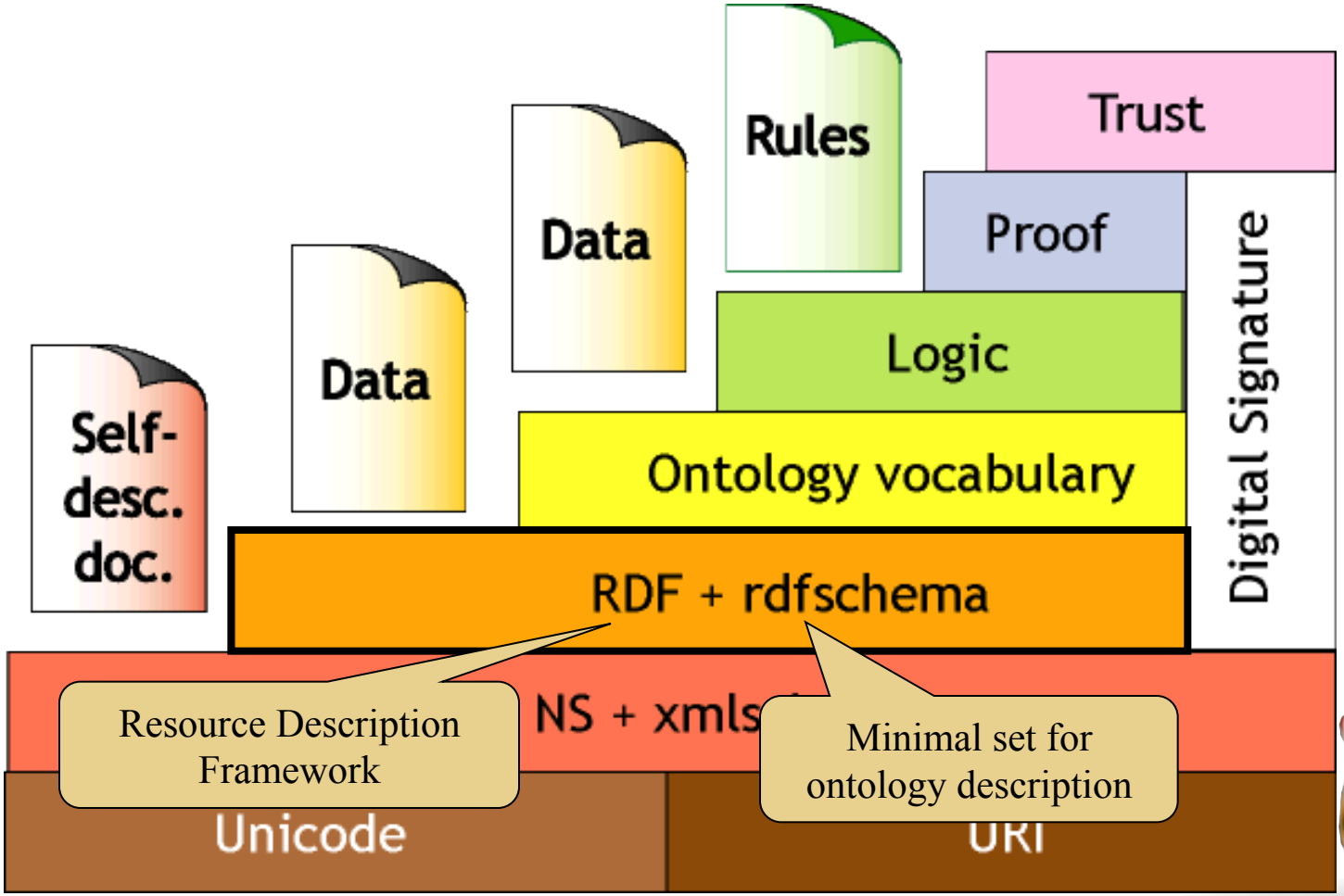
URIs are a superset in functionality and syntax to URLs (Universal Resource Locators) and URNs (Universal Resource Names). That is, RFC2396 updates and merges RFC1738 and RFC1808 into a single syntax, called the URI. It does exclude some portions of RFC1738 that define specific syntax of individual URL schemes.

Semantic Web Layer-cake



Tim Berners-Lee

Semantic Web Layer-cake



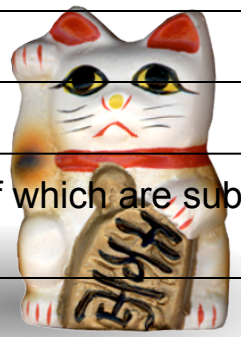
Tim Berners-Lee

RDF Vocabulary Description Language 1.0: RDF Schema

W3C Recommendation 10 February 2004

RDF Classes

Class name	comment
rdfs:Resource	The class resource, everything.
rdfs:Literal	The class of literal values, e.g. textual strings and integers.
rdf:XMLLiteral	The class of XML literals values.
rdfs:Class	The class of classes.
rdf:Property	The class of RDF properties.
rdfs:Datatype	The class of RDF datatypes.
rdf:Statement	The class of RDF statements.
rdf:Bag	The class of unordered containers.
rdf:Seq	The class of ordered containers.
rdf:Alt	The class of containers of alternatives.
rdfs:Container	The class of RDF containers.
rdfs:ContainerMembershipProperty	The class of container membership properties, rdf:_1, rdf:_2, ..., all of which are sub-properties of 'member'.
rdf:List	The class of RDF Lists.



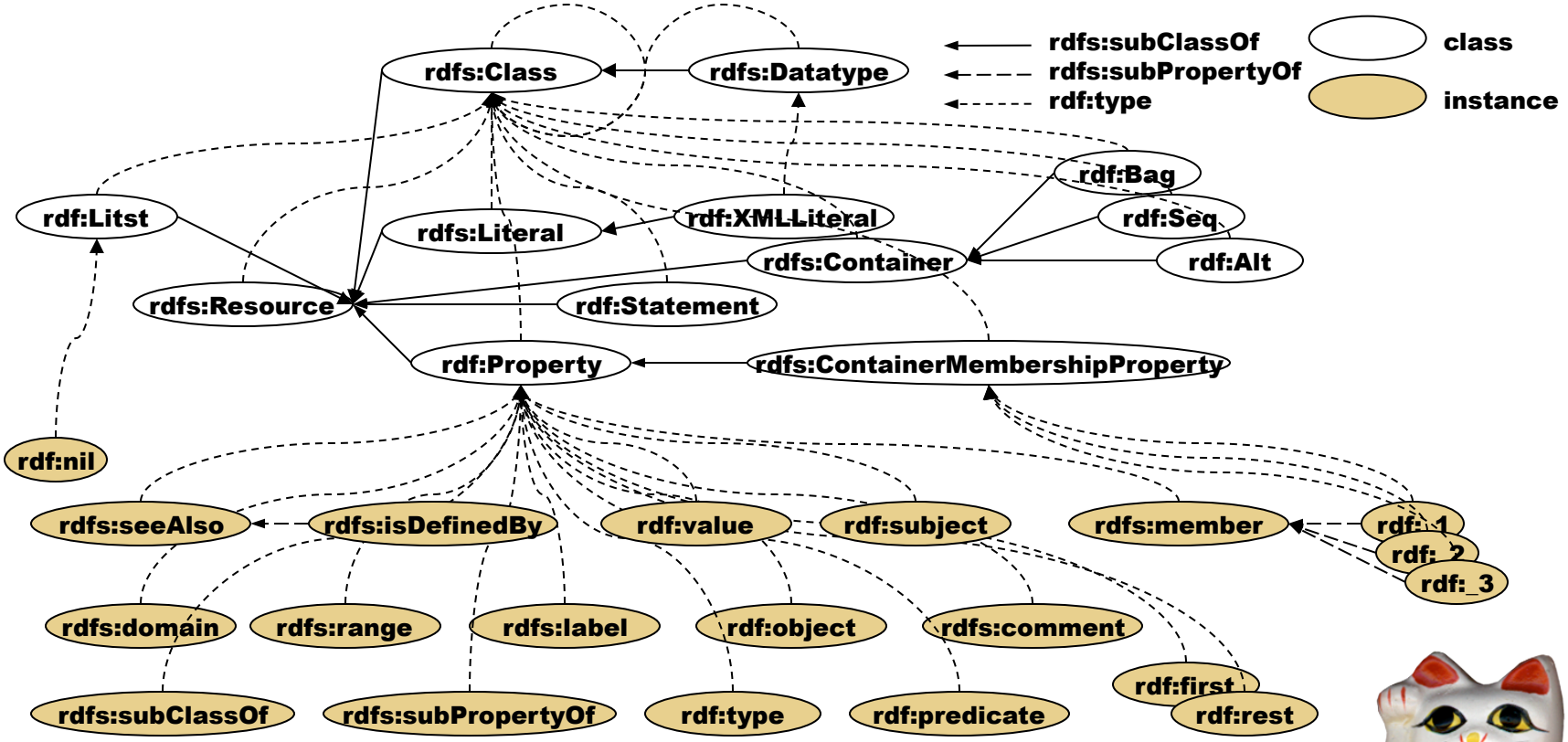
RDF Vocabulary Description Language 1.0: RDF Schema

W3C Recommendation 10 February 2004

RDF Properties

Property name	comment	domain	range
rdf:type	The subject is an instance of a class.	rdfs:Resource	rdfs:Class
rdfs:subClassOf	The subject is a subclass of a class.	rdfs:Class	rdfs:Class
rdfs:subPropertyOf	The subject is a subproperty of a property.	rdf:Property	rdf:Property
rdfs:domain	A domain of the subject property.	rdf:Property	rdfs:Class
rdfs:range	A range of the subject property.	rdf:Property	rdfs:Class
rdfs:label	A human-readable name for the subject.	rdfs:Resource	rdfs:Literal
rdfs:comment	A description of the subject resource.	rdfs:Resource	rdfs:Literal
rdfs:member	A member of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:first	The first item in the subject RDF list.	rdf:List	rdfs:Resource
rdf:rest	The rest of the subject RDF list after the first item.	rdf:List	rdf:List
rdfs:seeAlso	Further information about the subject resource.	rdfs:Resource	rdfs:Resource
rdfs:isDefinedBy	The definition of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:value	Idiomatic property used for structured values (see the RDF Primer for an example of its usage).	rdfs:Resource	rdfs:Resource
rdf:subject	The subject of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:predicate	The predicate of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:object	The object of the subject RDF statement.	rdf:Statement	rdfs:Resource

Hierarchical Structure of RDFS



Entailment Rules

http://www.w3.org/TR/rdf-mt/

Rule Name	If E contains:	then add:
rdfs1	uuu aaa lll. where lll is a plain literal (with or without a language tag).	_:nnn rdf:type rdfs:Literal . where _:nnn identifies a blank node allocated to lll by rule rule lg.
rdfs2	aaa rdfs:domain xxx . uuu aaa yyy .	uuu rdf:type xxx .
rdfs3	aaa rdfs:range xxx . uuu aaa vvv .	vvv rdf:type xxx .
rdfs4a	uuu aaa xxx .	uuu rdf:type rdfs:Resource .
rdfs4b	uuu aaa vvv.	vvv rdf:type rdfs:Resource .
rdfs5	uuu rdfs:subPropertyOf vvv . vvv rdfs:subPropertyOf xxx .	uuu rdfs:subPropertyOf xxx .
rdfs6	uuu rdf:type rdf:Property .	uuu rdfs:subPropertyOf uuu .
rdfs7	aaa rdfs:subPropertyOf bbb . uuu aaa yyy .	uuu bbb yyy .
rdfs8	uuu rdf:type rdfs:Class .	uuu rdfs:subClassOf rdfs:Resource .

Entailment Rules

http://www.w3.org/TR/rdf-mt/

Rule Name	If E contains:	then add:
rdfs9	**uuu rdfs:subClassOf xxx .** **vvv rdf:type uuu .**	**vvv rdf:type xxx .**
rdfs10	uuu rdf:type rdfs:Class .	uuu rdfs:subClassOf uuu .
rdfs11	**uuu rdfs:subClassOf vvv .** **vvv rdfs:subClassOf xxx .**	**uuu rdfs:subClassOf xxx .**
rdfs12	uuu rdf:type rdfs:ContainerMembershipProperty .	uuu rdfs:subPropertyOf rdfs:member
rdfs13	uuu rdf:type rdfs:Datatype .	uuu rdfs:subClassOf rdfs:Literal .

Subsumption Rule

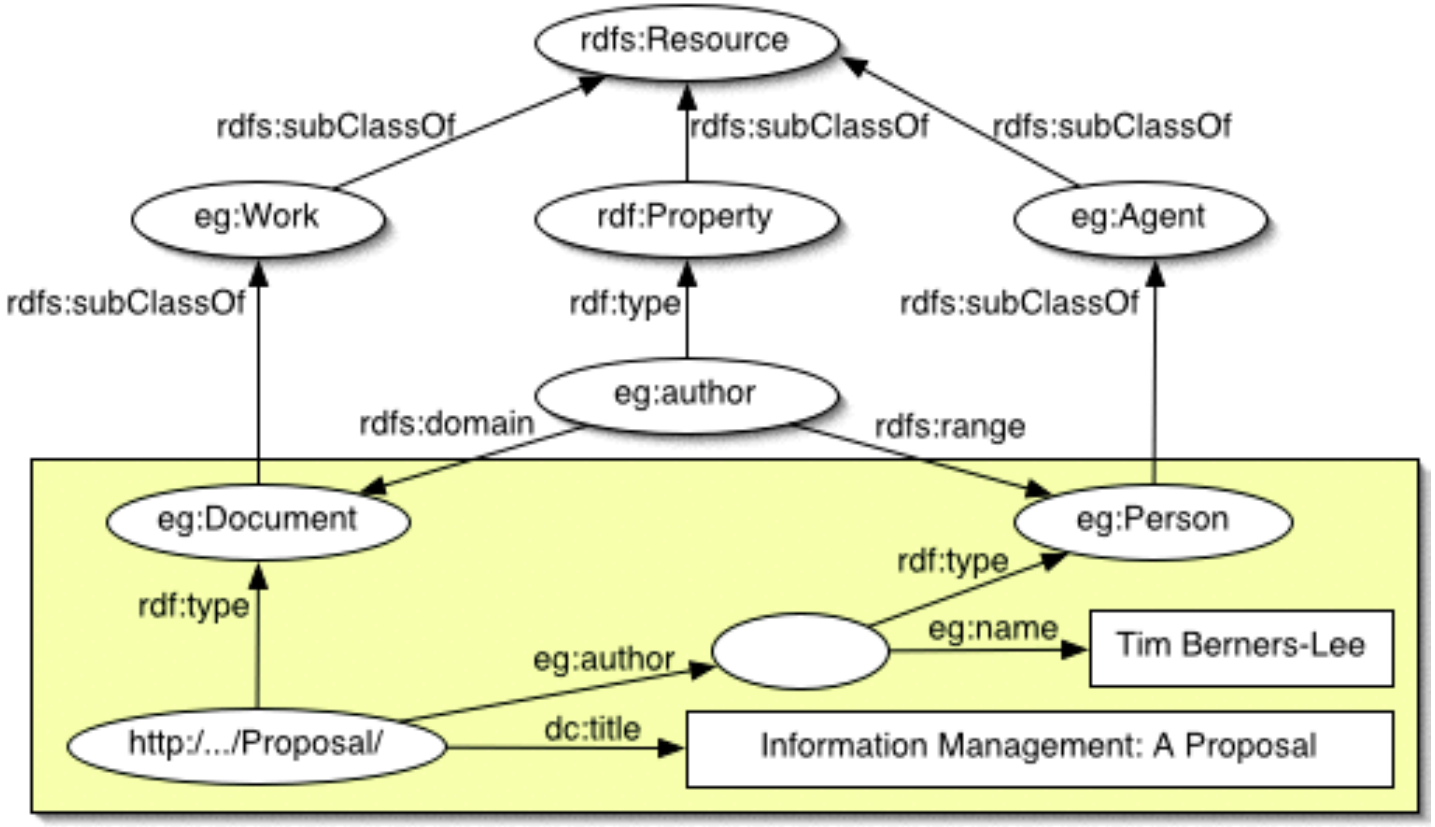
Transitivity Rule

```
(defclass xxx ( ) ( ) )
(setq vvv (make-instance (defclass uuu (xxx) ( ))))
(typep vvv xxx) → true
(defclass xxx ( ) ( ))
(defclass vvv (xxx) ( )) (defclass uuu (vvv) ( ))
(subtypep 'uuu 'xxx) → true
```

Introductory Example

Obsolete RDFS Document

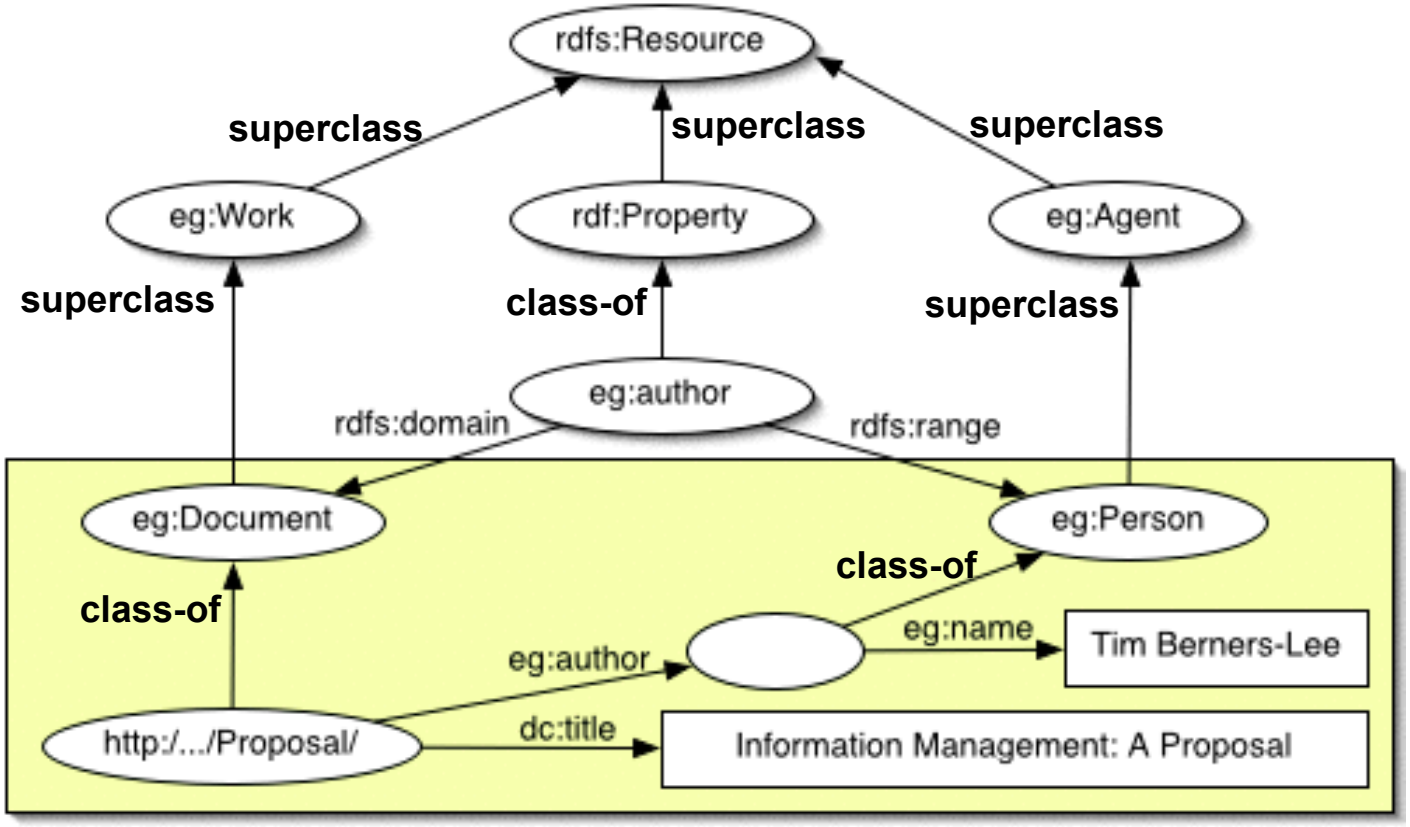
http://www.w3.org/TR/2002/WD-rdf-schema-20021112/



Introductory Example (Pure CLOS)

Obsolete RDFS Document

http://www.w3.org/TR/2002/WD-rdf-schema-20021112/



Introductory Example (Pure CLOS)

Obsolete RDFS Document

http://www.w3.org/TR/2002/WD-rdf-schema-20021112/

```
(defpackage rdf
  (:documentation "http://www.w3.org/1999/02/22-rdf-syntax-ns"))
(defpackage rdfs
  (:documentation "http://www.w3.org/2000/01/rdf-schema"))
(defpackage eg
  (:documentation "http://somewhere-for-eg/eg"))
(defpackage dc
  (:documentation
   "http://dublincore.org/2002/08/13/dces"))

(defclass rdfs::Resource ( ) ((rdf::about :initarg :about)))
(defclass eg::Work (rdfs::Resource) ( ))
(defclass eg::Agent (rdfs::Resource) ( ))
(defclass eg::Person (eg::Agent)
  ((eg::name :initarg :name)))
(defclass eg::Document (eg::Work)
  ((eg::author :initarg :author :type eg::Person)
   (dc::title :initarg :title)))
```

Introductory Example (Pure CLOS)

Obsolete RDFS Document

http://www.w3.org/TR/2002/WD-rdf-schema-20021112/

```
(defclass rdfs::Resource ( ) ((rdf::about :initarg :about)))
(defclass eg::Work (rdfs::Resource) ( ))
(defclass eg::Agent (rdfs::Resource) ( ))
(defclass eg::Person (eg::Agent)
  ((eg::name :initarg :name)))
(defclass eg::Document (eg::Work)
  ((eg::author :initarg :author :type eg::Person)
   (dc::title :initarg :title)))

(setq eg::Proposal
    (make-instance 'eg::Document
     :author (make-instance 'eg::Person :name "Tim Berners-Lee")
     :title "Information Management: A Proposal"
     :about "http:/…/Proposal/"))
(describe eg::Proposal)
```

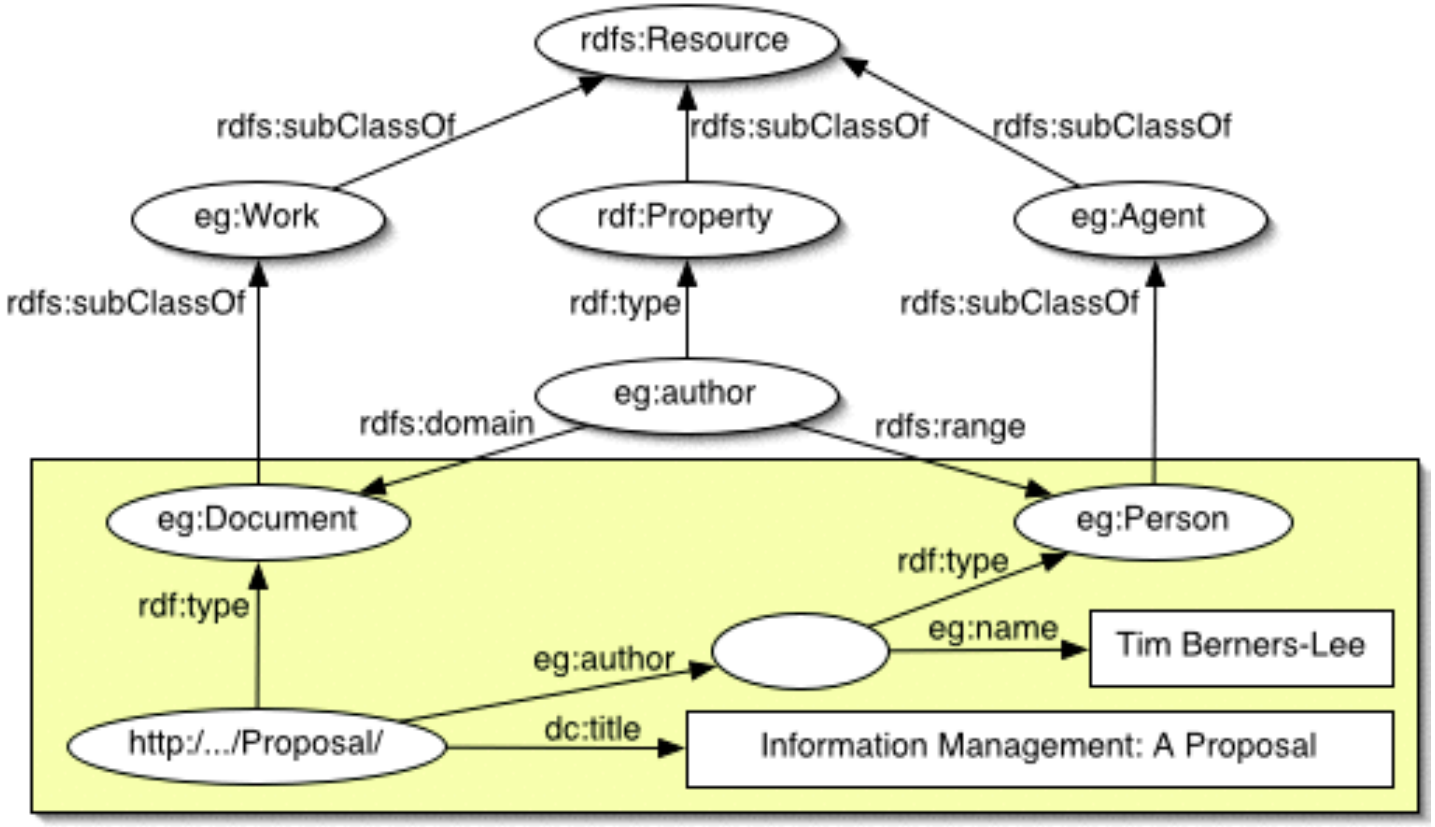
Bind to the Name Symbol

Lisp Native Function

Introductory Example

Obsolete RDFS Document

http://www.w3.org/TR/2002/WD-rdf-schema-20021112/



C:¥Documents and Settings¥koide¥My Documents¥SWOOP¥Example.rdf

ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)

戻る 検索 お気に入り メディア

アドレス(D) C:¥Documents and Settings¥koide¥My Documents¥SWOOP¥Example.rdf 移動

```
<?xml version="1.0" ?>
<!DOCTYPE rdf:RDF (View Source for full doctype...)>
- <rdf:RDF xmlns="http://galaxy-express.co.jp/semantic-web/example#"
    xmlns:eg="http://galaxy-express.co.jp/semantic-web/example#"
    xmlns:dc="http://dublincore.org/documents/2003/06/02/dces#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  - <rdf:Property rdf:ID="name">
      <rdfs:domain rdf:resource="#Person" />
      <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal" />
    </rdf:Property>
  - <rdf:Property
      rdf:about="http://dublincore.org/documents/2003/06/02/dces#title">
      <rdfs:domain rdf:resource="#Document" />
      <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal" />
    </rdf:Property>
  - <rdf:Property rdf:ID="author">
      <rdfs:domain rdf:resource="#Document" />
      <rdfs:range rdf:resource="#Person" />
    </rdf:Property>
  - <rdfs:Class rdf:ID="Person">
      <rdfs:subClassOf rdf:resource="#Agent" />
    </rdfs:Class>
  - <rdfs:Class rdf:ID="Document">
      <rdfs:subClassOf rdf:resource="#Work" />
    </rdfs:Class>
  - <eg:Document rdf:about="http:/.../Proposal/">
    - <eg:author>
      - <eg:Person>
          <eg:name>Tim Berners-Lee</eg:name>
        </eg:Person>
      </eg:author>
      <dc:title>Information Management: A Proposal</dc:title>
    </eg:Document>
</rdf:RDF>
```

ページが表示されました マイ コンピュータ

Intr

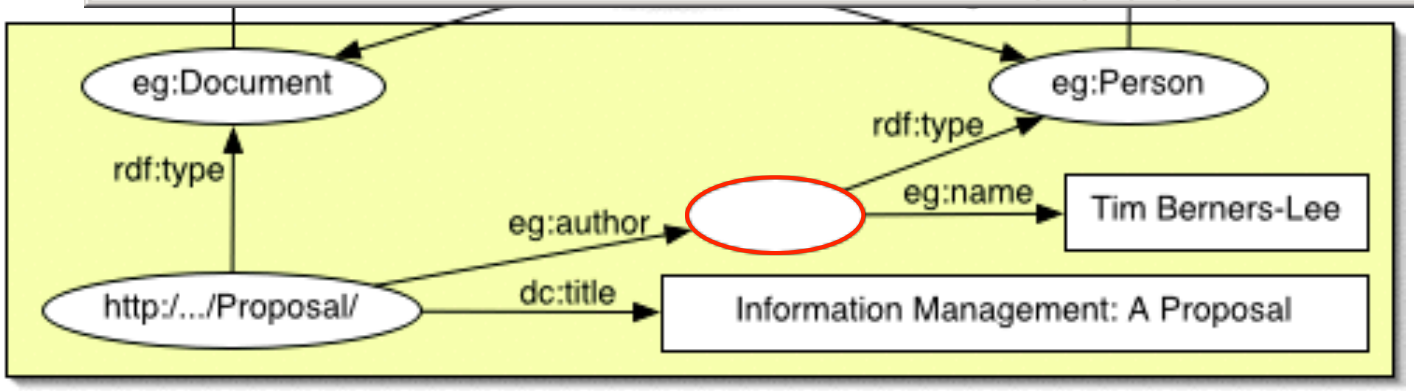
Obse

http:/

```
    rdf:about="http://dublincore.org/documents/2003/06/02/dces#title">
    <rdfs:domain rdf:resource="#Document" />
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal" />
  </rdf:Property>
- <rdf:Property rdf:ID="author">
    <rdfs:domain rdf:resource="#Document" />
    <rdfs:range rdf:resource="#Person" />
  </rdf:Property>
- <rdfs:Class rdf:ID="Person">
    <rdfs:subClassOf rdf:resource="#Agent" />
  </rdfs:Class>
- <rdfs:Class rdf:ID="Document">
    <rdfs:subClassOf rdf:resource="#Work" />
  </rdfs:Class>
- <eg:Document rdf:about="http:/.../Proposal/">
  - <eg:author>
    - <eg:Person>
        <eg:name>Tim Berners-Lee</eg:name>
      </eg:Person>
    </eg:author>
    <dc:title>Information Management: A Proposal</dc:title>
  </eg:Document>
</rdf:RDF>
```

ページが表示されました　マイ コンピュータ

rdfs:su



Introductory Example

Obsolete RDFS Document

http://www.w3.org/TR/2002/WD-rdf-schema-20021112/

Reboot Lisp and Load SWCLOS RDFS

Already defined in SWCLOS

```
(in-package gx-user)
rdfs:Resource → #<rdfs:Class rdfs:Resource>
(defpackage eg
  (:documentation "http://somewhere-for-eg/eg"))
(defpackage dc
  (:documentation
   "http://dublincore.org/2002/08/13/dces"))

(defResource eg::Work (rdfs:subClassOf rdfs:Resource))
(defResource eg::Agent (rdfs:subClassOf rdfs:Resource))
(defResource eg::Person (rdfs:subClassOf eg:Agent))
(defResource eg::Document (rdfs:subClassOf eg:Work))

(defProperty eg::author
  (rdfs:domain eg:Document)
  (rdfs:range eg:Person))
```

Double colons needed firstly

Single colon accepted secondly and after

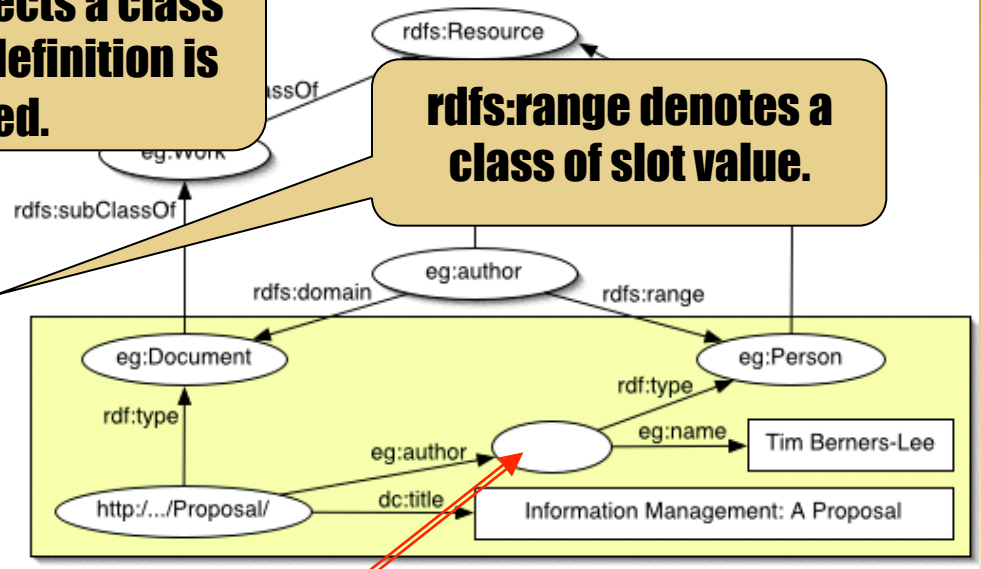
Introductory Example

Obsolete RDFS Document

http://www.w3.org/TR/2002/WD-rdf-schema-20021112/

rdfs:domain directs a class where the slot definition is attached.

rdfs:range denotes a class of slot value.




```
(defProperty eg::name
  (rdfs:domain eg:Person)
  (rdfs:range rdfs:Literal))
(defProperty dc::title
  (rdfs:domain eg:Document)
  (rdfs:range rdfs:Literal))
```

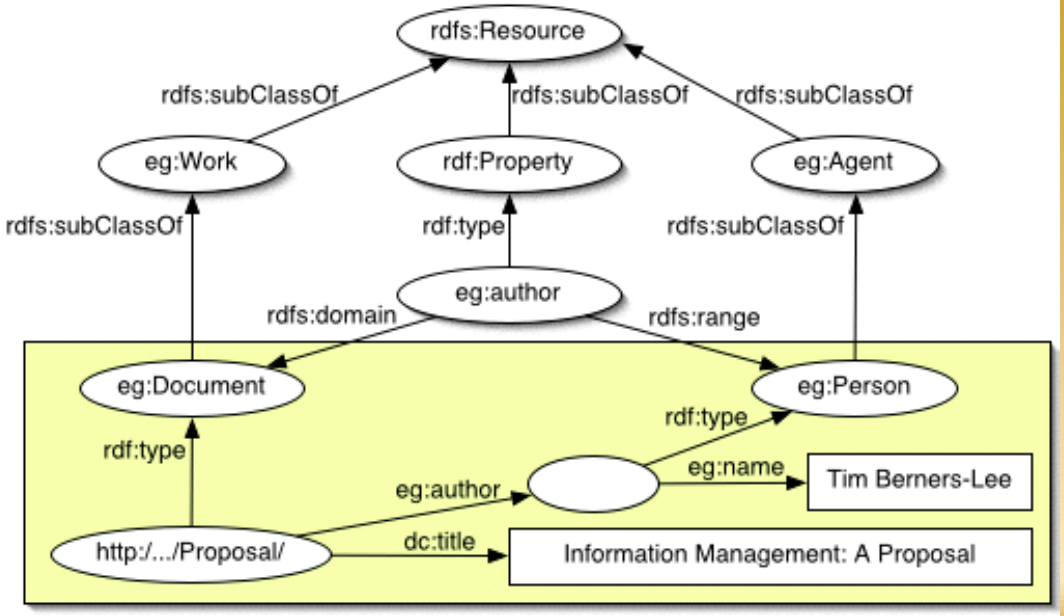
```
(defIndividual eg::Proposal (rdf:type eg::Document)
  (eg:author (eg:Person (eg:name "Tim Berners-Lee")))
  (dc:title "Information Management: A Proposal")
  (rdf:about "http:/…/Proposal/"))
```

A bnode is automatically created on demand.

Introductory Example

Obsolete RDFS Document

http://www.w3.org/TR/2002/WD-rdf-schema-20021112/




```
eg:Person →
eg:author →
eg:Proposal →

(describe eg:Proposal) →
(get-form eg:Proposal) →

(write-xml eg:Proposal)
```

The RDF/XML form of `eg:Proposal` is printed.

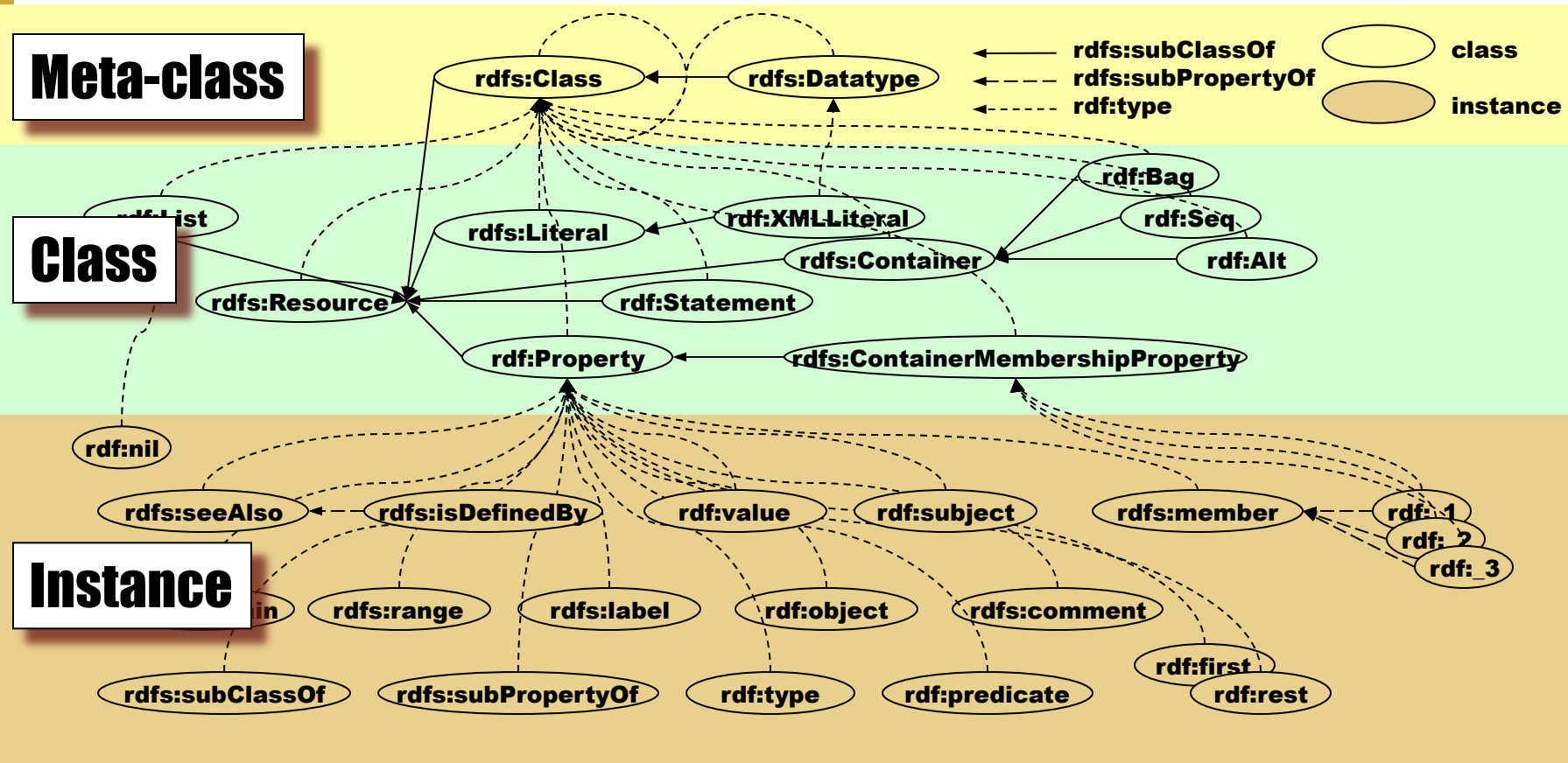
```
(-> eg:Proposal dc:title) →
(-> eg:Proposal eg:author eg:name)  →
(-> eg:Proposal eg:author rdf:type) →
```

Start from 1st parameter and travel the graph along with the path

Summary of Introduction

- Case sensitive lisp
- rdf:type to `class-of` (`type-of`) mapping
- rdfs:subClassOf to superclass mapping
- A QName is represented as an exported lisp symbol.
- A resource of RDF is created as a CLOS object, and bound to the QName.
- Define Macro `defResource`, `defIndividual`, and `defProperty`
- rdfs:domain indicates a class where the slot definition is attached.
- rdfs:range denotes a class of slot value.

Hierarchical Structure of RDFS



Straight-forward Mapping

- RDFS Class to CLOS Class
- Resource Instance to CLOS Instance
- rdfs:Class & rdfs:Datatype to CLOS meta-class

Gain:

- Lisp native `cl:typep` and `cl:subtypep` is available
- Linear search in the class precedence list
- Reflective programming by meta-class

RDFS Axioms on CLOS

cf. http://www.w3.org/TR/rdf-mt/

```
(type-of rdf:type)          → rdf:Property
(type-of rdf:subject)       → rdf:Property
(type-of rdf:predicate)     → rdf:Property
 .. ..
(type-of rdf:nil)           → rdf:List

(rdfs:domain rdf:type)      → #<rdfs:Class rdfs:Resource>
(rdfs:domain rdfs:domain)   → #<rdfs:Class rdf:Property>
(rdfs:domain rdfs:range)    → #<rdfs:Class rdf:Property>
(rdfs:range rdf:type)       → #<RDFSclass rdfs:Class>
(rdfs:range rdfs:domain)    → #<RDFSclass rdfs:Class>
(rdfs:range rdfs:range)     → #<RDFSclass rdfs:Class>

(type-of rdfs:Resource)     → rdfs:Class
(type-of rdfs:Class)        → rdfs:Class
(type-of rdfs:Literal)      → rdfs:Class
```

Entailment Rules

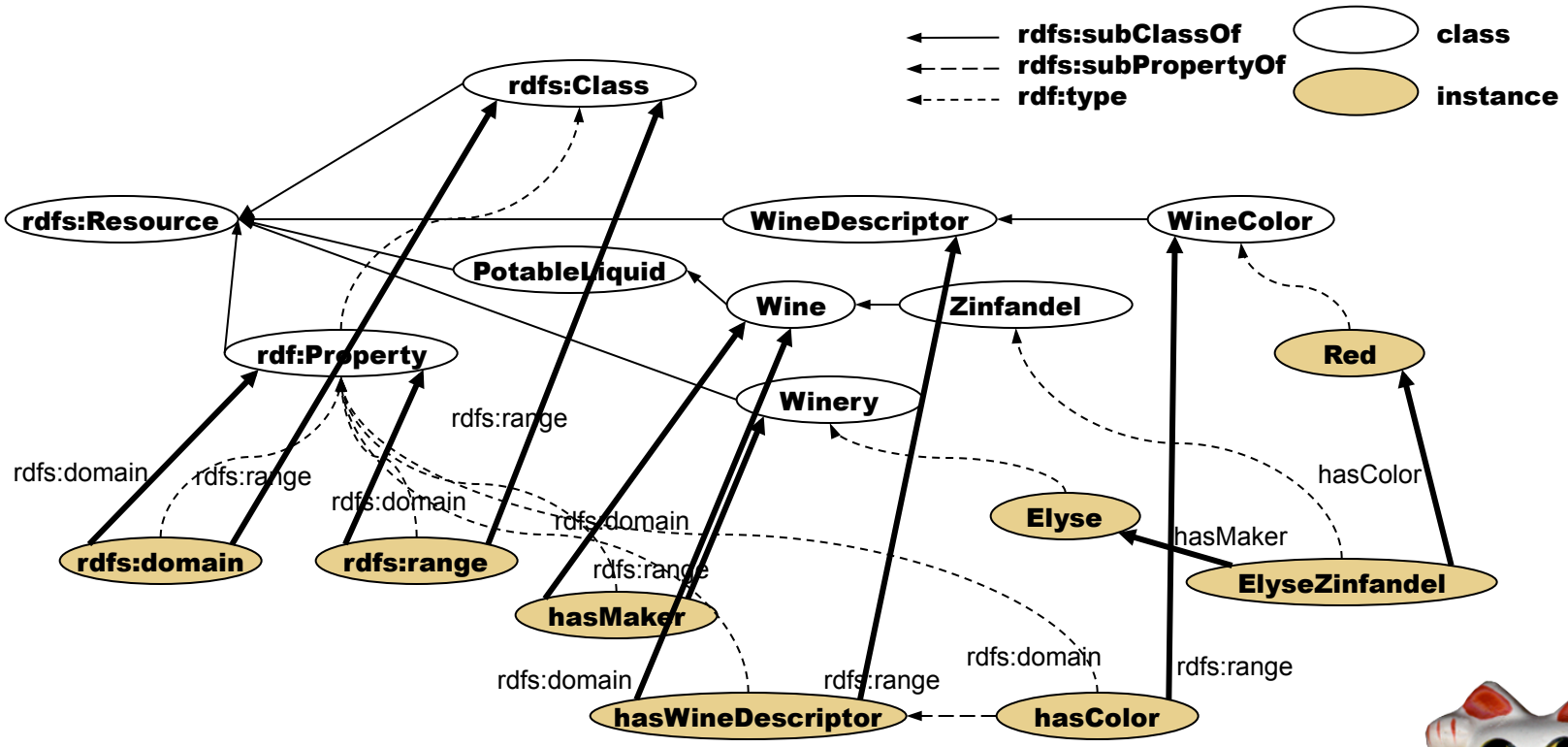
http://www.w3.org/TR/rdf-mt/

Rule Name	If E contains:	then add:
rdfs1	uuu aaa lll. where lll is a plain literal (with or without a language tag).	_:nnn rdf:type rdfs:Literal . where _:nnn identifies a blank node allocated to lll by rule rule lg.
rdfs2	**aaa rdfs:domain xxx .** **uuu aaa yyy .**	**uuu rdf:type xxx .**
rdfs3	**aaa rdfs:range xxx .** **uuu aaa vvv .**	**vvv rdf:type xxx .**
rdfs4a	uuu aaa xxx .	uuu rdf:type rdfs:Resource .
rdfs4b	uuu aaa vvv.	vvv rdf:type rdfs:Resource .
rdfs5	uuu rdfs:subPropertyOf vvv . vvv rdfs:subPropertyOf xxx .	uuu rdfs:subPropertyOf xxx .
rdfs6	uuu rdf:type rdf:Property .	uuu rdfs:subPropertyOf uuu .
rdfs7	aaa rdfs:subPropertyOf bbb . uuu aaa yyy .	uuu bbb yyy .
rdfs8	uuu rdf:type rdfs:Class .	uuu rdfs:subClassOf rdfs:Resource .

Domain Entailment

Range Entailment

Domain & Range in Wine Ontology (RDFS)



rdfs:domain: classes that are reachable from a subject node with an path of rdf:type→rdfs:subClassOf
rdfs:range: classes that are reachable from a object node with an path of rdf:type→rdfs:subClassOf

Domain & Range in Wine Ontology (RDFS)

```
(defpackage vin)
(defpackage food)
(in-package gx-user)

(defResource vin::Wine (rdfs:subClassOf food::PotableLiquid))
(defProperty vin::hasWineDescriptor (rdfs:domain vin:Wine))
(defProperty vin::hasColor
  (rdfs:subPropertyOf vin:hasWineDescriptor))

(rdfs:domain vin:hasWineDescriptor)  →
(rdfs:domain vin:hasColor)*          →
(domain-value vin:hasWineDescriptor) →
(domain-value vin:hasColor)          →
(get-domain vin:hasWineDescriptor)   →
(get-domain vin:hasColor)            →

(defProperty vin:hasWineDescriptor
  (rdfs:range vin::WineDescriptor))
(get-range vin:hasColor)             →
```

rdfs:domain method causes an error for unbound value

get-domain inherits the value from super-properties

Piecewisely add the rage information later

Domain & Range in Wine Ontology (RDFS)

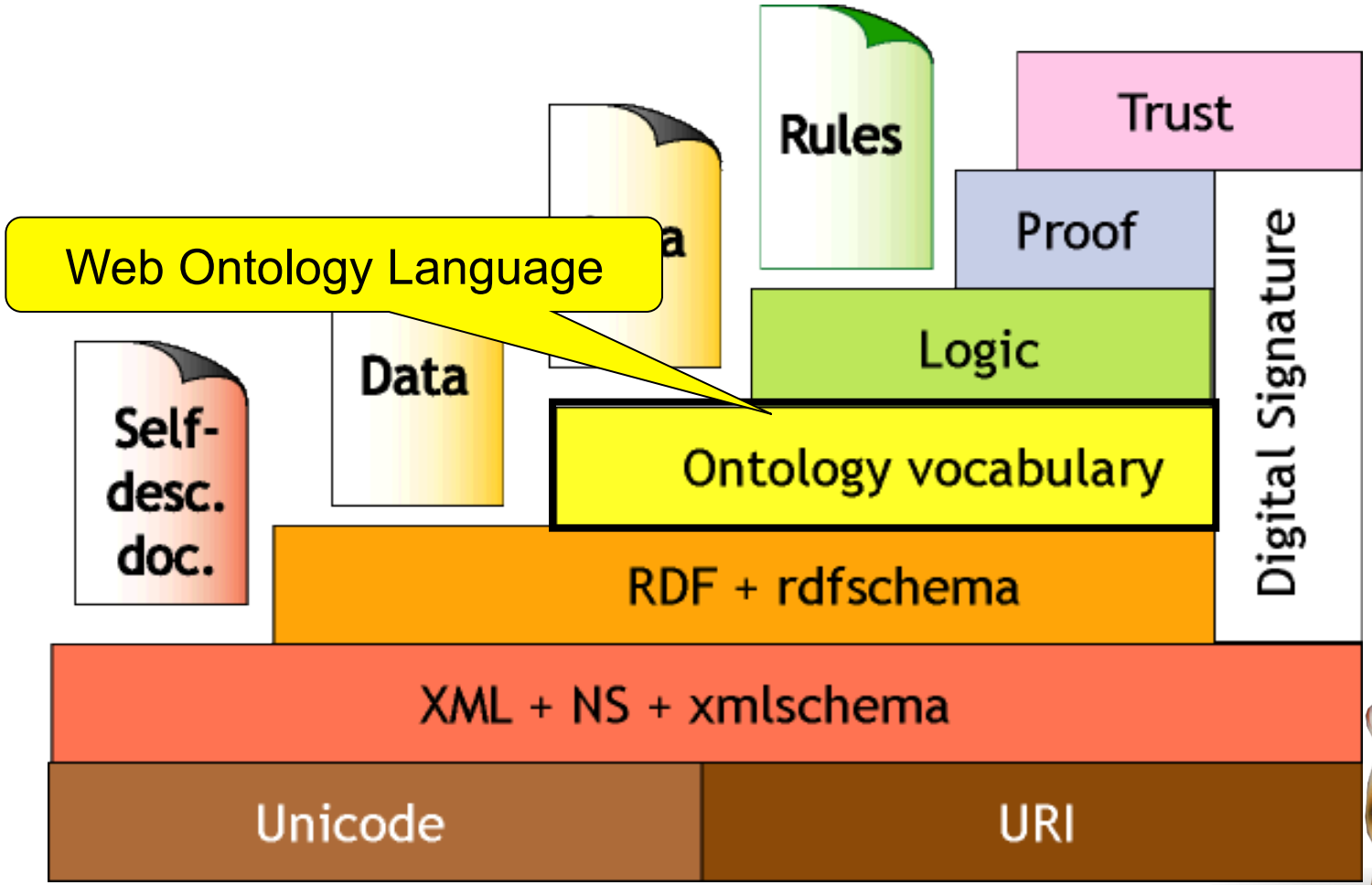
```
(defProperty vin::hasColor (rdfs:range vin::WineColor))
(defResource vin:WineColor (rdfs:subClassOf vin:WineDescriptor))

(defIndividual vin::ElyseZinfandel (rdf:type vin::Zinfandel)
  (vin:hasColor vin::Red))
;;Entailment by domain and range, making multiple classes
(type-of vin:ElyseZinfandel)      →
(cl:type-of vin:ElyseZinfandel)   →
(subclasses-of vin:Zinfandel)     →
(subclasses-of vin:Wine)          →
(type-of vin:Red)                 →

(defIndividual vin::NumericWine (vin:hasColor 123))*

(defProperty vin::hasMaker
  (rdfs:domain vin:Wine)
  (rdfs:range vin::Winery))
(defIndividual vin:ElyseZinfandel (vin:hasMaker vin::Elyse))
(type-of vin:Elyse)               →
(get-form vin:ElyseZinfandel)     →
```

SWCLOS checks the range

Summary of RDFS

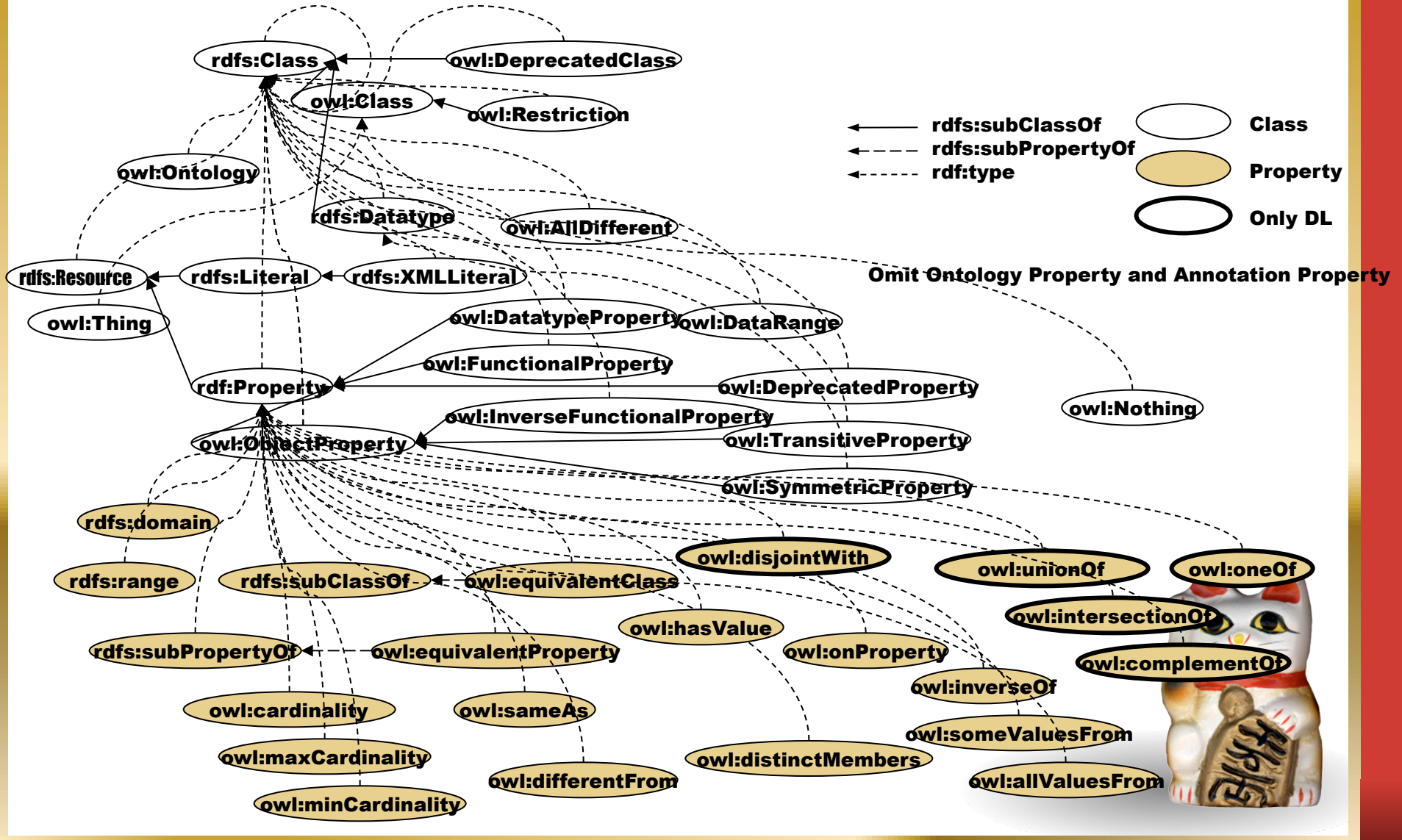
- Meta-classes, classes, and instances
- Instances of rdf:Property has super-sub concept.
- Domain and range value is inherited.
- RDFS axioms
- rdfs:domain & rdfs:range constraint
- rdfs:domain & rdfs:range entailment
- Piecewise ontology adding

Semantic Web Layer-cake



Tim Berners-Lee

Web Ontology Language - OWL



Questions on Implementation of OWL

- Unknown Complete Set of Entailments in OWL
- How to Realize Proactive Alarming
  - Lisp is interactive.
  - One by one satisfiable check and entailment
  - Tableau algorithm looks like for Query.
  - What query should we make?
  - Proactive Entailment needs Entailment Rules
  - Extension of RDFS implementation by MOP

Local Range Restrictions

- owl:allValuesFrom
  - the values of the property are all members of the class indicated
- owl:someValuesFrom
  - at least one value of the property is an instance of the class indicated
- Cardinality
  - owl:maxCardinality to specify an upper bound
    owl:minCardinality to specify a lower bound
    owl:cardinality to specify an exact number of elements
- owl:hasValue
  - a class of all individuals for which the property concerned has at least one value semantically equal to the value (it may have other values as well)

Cardinality

Reboot Lisp and Load SWCLOS OWL

```
(in-package gx-user)
(read-rdf-file #'addRdfXml "wine.rdf") or
(read-rdf-file #'addRdfXml) and indicate wine.rdf file
(write-xml vin:Wine)
```

RDF/XML form on vin:Wine

```
vin:Elyse     → #<vin:Winery vin:Elyse>
vin:Bancroft → #<vin:Winery vin:Bancroft>

(describe vin:ElyseZinfandel)

#<vin:Zinfandel.53 vin:ElyseZinfandel> is an instance of #<gx::shadow-class vin:Zinf
 The following slots have :instance allocation:
  mclasses                  (#<owl:Class vin:Wine> #<owl:Class vin:Zinfandel>)
  member                    <unbound>
  name                      vin:ElyseZinfandel
  about                     <unbound>
  lang                      common-lisp:nil
  label                     <unbound>
  isDefinedBy               <unbound>
  comment                   <unbound>
  locatedIn                 #<vin:Region vin:NapaRegion>
  differentFrom             <unbound>
  sameAs                    <unbound>
```

Cardinality

```
(setf (slot-value vin:ElyseZinfandel 'vin:madeFromGrape)
      vin:Bancroft)

CARDINALITY CHECK ERROR: THE OPERATION IS NOT EFFECTIVE:
 #<vin:Zinfandel vin:ElyseZinfandel> vin:hasMaker <- #<vin:Winery vin:Bancroft>
```

(get-form vin:Wine) →

```
(owl:Class vin:Wine (rdfs:label (rdf:en "wine") (rdf:fr "vin"))
 (rdfs:subClassOf
  food:PotableLiquid
   (owl:Restriction (owl:onProperty vin:hasMaker)
                    (owl:cardinality 1))
   (owl:Restriction (owl:onProperty vin:hasMaker)
                    (owl:allValuesFrom vin:Winery))
   (owl:Restriction (owl:onProperty vin:madeFromGrape)
                    (owl:minCardinality 1))
   (owl:Restriction (owl:onProperty vin:hasSugar)
                    (owl:cardinality 1))
   (owl:Restriction (owl:onProperty vin:hasFlavor)
                    (owl:cardinality 1))
   (owl:Restriction (owl:onProperty vin:hasBody)
                    (owl:cardinality 1))
   (owl:Restriction (owl:onProperty vin:hasColor)
                    (owl:cardinality 1))
   (owl:Restriction (owl:onProperty vin:locatedIn)
                    (owl:someValuesFrom vin:Region))))
```

Already
(vin:hasMaker
vin:Elyse)

Attempt to set a new value

Not overwrite, just add monotonically

Cardinality

```
(defResource BlendedWine (rdf:type owl:Class)
  (rdfs:subClassOf
    vin:Wine
    (owl:Restriction (owl:onProperty vin:madeFromGrape)
                     (owl:minCardinality 2))
    (owl:Restriction (owl:onProperty vin:hasColor)
                     (owl:minCardinality 2))))

(defIndividual MyBlendedWine (rdf:type BlendedWine)
  (vin:hasColor vin:Red vin:White))
```

```
Error: Unsatisfiability by cardinality:BlendedWine vin:hasColor
```

The minimum maxcardinality and the maximum mincardinality so far are effective.

$$\text{minCardinality} \leq \text{maxCardinality}$$

owl:allValuesFrom

```
(defResource House (rdf:type owl:Class))
(defIndividual MyHouse (rdf:type House))
(defIndividual MyHomeMadeWine (rdf:type vin:Wine)
 (vin:hasMaker MyHouse))

Warning: Range entail4:change class of #<House MyHouse> to #<owl:Class
vin:Winery>.
Warning: #<House MyHouse> is additionally classified to (owl:Class vin:Winery).

MyHouse → #<House.17 MyHouse>
(type-of MyHouse) → (vin:Winery House)
(get-form vin:Wine) →
(owl:Class vin:Wine (rdfs:label (rdf:en "wine") (rdf:fr "vin"))
 (rdfs:subClassOf
  food:PotableLiquid
   (owl:Restriction (owl:onProperty vin:hasMaker)
                    (owl:cardinality 1))
   (owl:Restriction (owl:onProperty vin:hasMaker)
                    (owl:allValuesFrom vin:Winery))
   (owl:Restriction (owl:onProperty vin:madeFromGrape)
                    (owl:minCardinality 1))
   (owl:Restriction (owl:onProperty vin:hasSugar)
                    (owl:cardinality 1))
   (owl:Restriction (owl:onProperty vin:hasFlavor)
                    (owl:cardinality 1))
   (owl:Restriction (owl:onProperty vin:hasBody)
```

multiple classes

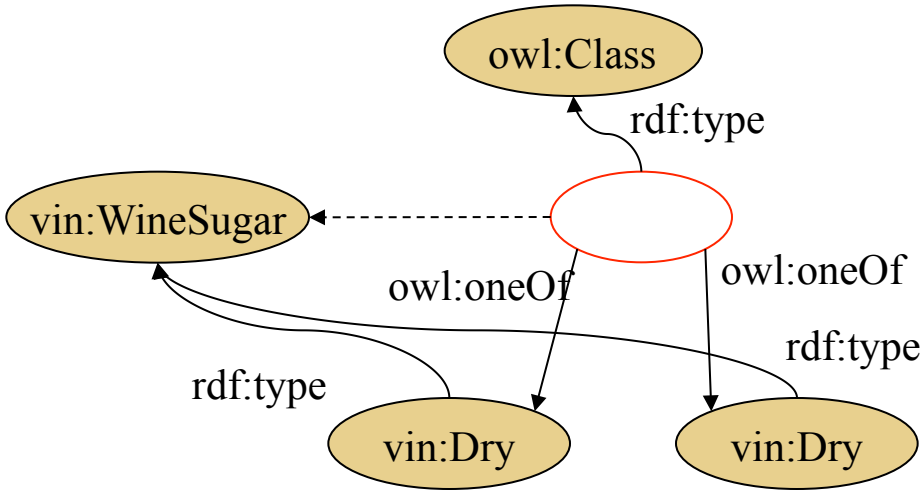
owl:someValuesFrom

```
(get-form vin:Wine) →
(owl:Class vin:Wine (rdfs:label (rdf:en "wine") (rdf:fr "vin"))
 (rdfs:subClassOf food:PotableLiquid
                  (owl:Restriction (owl:onProperty vin:hasMaker)
                   (owl:cardinality 1))
                  (owl:Restriction (owl:onProperty vin:hasMaker)
                   (owl:allValuesFrom vin:Winery))
                  (owl:Restriction (owl:onProperty vin:madeFromGrape)
                   (owl:minCardinality 1))
                  (owl:Restriction (owl:onProperty vin:hasSugar)
                   (owl:cardinality 1))
                  (owl:Restriction (owl:onProperty vin:hasFlavor)
                   (owl:cardinality 1))
                  (owl:Restriction (owl:onProperty vin:hasBody)
                   (owl:cardinality 1))
                  (owl:Restriction (owl:onProperty vin:hasColor)
                   (owl:cardinality 1))
                  (owl:Restriction (owl:onProperty vin:locatedIn)
                   (owl:someValuesFrom vin:Region))))
```

This statement claims that there must be vin:locatedIn slot in Wine instances,
And one of the slot values must be a region.
However, you cannot entail if you find something because of Open World Assumption.

owl:hasValue

```
(pprint (get-form vin:EarlyHarvest)) →
(owl:Class vin:EarlyHarvest
 (rdfs:subClassOf
   vin:Wine
   (owl:Restriction (owl:onProperty vin:hasSugar)
     (owl:allValuesFrom
      (owl:Class (owl:oneOf vin:Dry vin:OffDry))))))
(get-range vin:hasSugar)
  → #<owl:OneOf vin:WineSugar{vin:Sweet vin:OffDry vin:Dry}>
```



Equality and Difference/Disjoint

- owl:sameAs
- owl:differentFrom
- owl:AllDifferent - owl:distinctMembers

Individual

- owl:equivalentClass
- owl:disjointWith

For Class

Different/Same Individuals

```
(owl-different-p vin:Dry vin:OffDry)   → true
(owl-different-p vin:OffDry vin:Sweet) → true
(owl-different-p vin:Sweet vin:Dry)    → true

(get-form vin:Dry) →
(vin:WineSugar vin:Dry (rdf:about "#Dry"))

(get-form vin:Sweet) →
(vin:WineSugar vin:Sweet (rdf:about "#Sweet")
 (owl:differentFrom vin:Dry))

(get-form vin:OffDry) →
(vin:WineSugar vin:OffDry (rdf:about "#OffDry")
 (owl:differentFrom vin:Dry vin:Sweet))
```

Different/Same Individuals

```
(owl-different-p vin:SonomaRegion vin:NapaRegion)
 → common-lisp:nil
(owl-same-p vin:SonomaRegion vin:NapaRegion) → common-lisp:nil
```

cl:nil don't means false in open world.

```
(get-form vin:SonomaRegion) →
(vin:Region vin:SonomaRegion
  (vin:locatedIn vin:CaliforniaRegion))

(get-form vin:NapaRegion) →
(vin:Region vin:NapaRegion
  (vin:locatedIn vin:CaliforniaRegion))

(collect-all-extensions-of owl:sameAs)   → common-lisp:nil
```

collect-all-extensions-of collects all pairs on the relation of the property in memory.

Different/Same Individuals

```
(owl-different-p vin:Red vin:White)  → true
(owl-different-p vin:White vin:Rose) → true
(owl-different-p vin:Rose vin:Red)   → true
(get-form vin:Red) → (vin:WineColor vin:Red (rdf:about "#Red"))
(get-form vin:White) →
(vin:WineColor vin:White (rdf:about "#White"))
(collect-all-extensions-of owl:distinctMembers) →
((#<owl:AllDifferent common-lisp:nil>
  (#<vin:Winery vin:Bancroft> ...))
 (#<owl:AllDifferent common-lisp:nil>
  (#<vin:WineSugar vin:Sweet> #<vin:WineSugar vin:OffDry>
   #<vin:WineSugar vin:Dry>))
 (#<owl:AllDifferent common-lisp:nil>
  (#<vin:WineFlavor vin:Delicate> ...))
 (#<owl:AllDifferent common-lisp:nil>
  (#<vin:WineBody vin:Light> #<vin:WineBody vin:Medium>
   #<vin:WineBody vin:Full>))
 (#<owl:AllDifferent common-lisp:nil>
  (#<vin:WineColor vin:Red> #<vin:WineColor vin:White>
   #<vin:WineColor vin:Rose>)))
```

Equivalent/Disjoint Class

```
(collect-all-extensions-of owl:disjointWith) →
((#<owl:Class vin:LateHarvest> #<owl:Class vin:EarlyHarvest>))

(disjoint-p vin:LateHarvest vin:EarlyHarvest) → true
(get-form vin:LateHarvest) →
(owl:Class vin:LateHarvest
 (rdfs:subClassOf
   vin:Wine
   (owl:Restriction (owl:onProperty vin:hasSugar)
                    (owl:hasValue vin:Sweet))
   (owl:Restriction (owl:onProperty vin:hasFlavor)
                    (owl:allValuesFrom (owl:Class #))))
 (owl:disjointWith vin:EarlyHarvest))

(collect-all-extensions-of owl:equivalentClass) →
common-lisp:nil
```

Do not share instances

No equivalentClasses
in Wine Ontology

Equivalent/Disjoint Class

```
(read-rdf-file #'addRdfXml "food.rdf")

(collect-all-extensions-of owl:equivalentClass) →
((#<owl:Class food:Wine> #<owl:Class vin:Wine>))

(owl-equivalent-p food:Wine vin:Wine) → true
(get-form food:Wine) →
(owl:Class food:Wine (owl:equivalentClass vin:Wine))

(owl-equivalent-p vin:DryWine vin:TableWine) → true
(get-form vin:DryWine) →
(owl:Class vin:DryWine
 (owl:intersectionOf vin:Wine
  (owl:Restriction (owl:onProperty vin:hasSugar)
                   (owl:hasValue vin:Dry))))
(get-form vin:TableWine)→
(owl:Class vin:TableWine (rdf:about "#TableWine")
 (owl:intersectionOf vin:Wine
  (owl:Restriction (owl:onProperty vin:hasSugar)
                   (owl:hasValue vin:Dry))))
```

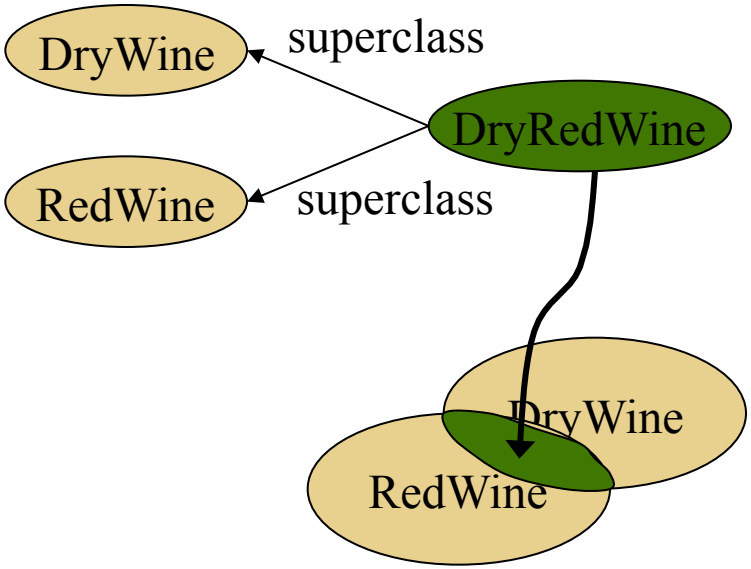
intersectionOf is descriptive and definitive.

Set Calculation of Classes

- owl:intersectionOf
- owl:unionOf
- owl:complementOf

Intersection of Classes

```
(get-form vin:DryRedWine) →
(owl:Class vin:DryRedWine
  (owl:intersectionOf vin:DryWine vin:RedWine))
(subsumed-p vin:DryRedWine vin:DryWine) → true
(subsumed-p vin:DryRedWine vin:RedWine) → true
```



Intersection of Classes

```
(get-form vin:RedWine) →
(owl:Class vin:RedWine
 (owl:intersectionOf vin:Wine
  (owl:Restriction (owl:onProperty vin:hasColor)
                   (owl:hasValue vin:Red))))

(get-form vin:RedTableWine)→
(owl:Class vin:RedTableWine
 (owl:intersectionOf vin:TableWine
  (owl:Restriction (owl:onProperty vin:hasColor)
                   (owl:hasValue vin:Red))))
(subsumed-p vin:RedTableWine vin:TableWine) → true
(subsumed-p vin:RedTableWine vin:RedWine)   → true
```

vin:TableWine ⊆ vin:Wine

vin:RedTableWine ⊆ vin:RedWine

Intersection of Classes

```
(get-form vin:MariettaOldVinesRed) →
(vin:RedTableWine vin:MariettaOldVinesRed
 (vin:hasMaker vin:Marietta)
 (vin:hasFlavor vin:Moderate)
 (vin:hasBody vin:Medium)
 (vin:locatedIn vin:SonomaRegion)
 (vin:hasColor vin:Red)
 (vin:hasSugar vin:Dry))

(typep vin:MariettaOldVinesRed vin:RedWine) → true

(get-form vin:RedWine) →
(owl:Class vin:RedWine
 (owl:intersectionOf vin:Wine
  (owl:Restriction (owl:onProperty vin:hasColor)
                   (owl:hasValue vin:Red))))
```

Intersection of Classes

```
(get-form vin:MariettaOldVinesRed) →
(vin:RedTableWine vin:MariettaOldVinesRed
 (vin:hasMaker vin:Marietta)
 (vin:hasFlavor vin:Moderate)
 (vin:hasBody vin:Medium)
 (vin:locatedIn vin:SonomaRegion)
 (vin:hasColor vin:Red)
 (vin:hasSugar vin:Dry))

(typep vin:MariettaOldVinesRed vin:DryWine) → true

(get-form vin:DryWine) →
(owl:Class vin:DryWine
 (owl:intersectionOf
  vin:Wine
   (owl:Restriction (owl:onProperty vin:hasSugar)
                    (owl:hasValue vin:Dry))))
```

Intersection of Classes

```
(get-form vin:MariettaOldVinesRed) →
(vin:RedTableWine vin:MariettaOldVinesRed
 (vin:hasMaker vin:Marietta)
 (vin:hasFlavor vin:Moderate)
 (vin:hasBody vin:Medium)
 (vin:locatedIn vin:SonomaRegion)
 (vin:hasColor vin:Red)
 (vin:hasSugar vin:Dry))

(typep vin:MariettaOldVinesRed vin:TableWine) → true

(get-form vin:TableWine)→
(owl:Class vin:TableWine (rdf:about "#TableWine")
 (owl:intersectionOf
  vin:Wine
   (owl:Restriction (owl:onProperty vin:hasSugar)
                    (owl:hasValue vin:Dry))))
```

Intersection of Classes

```
(typep vin:ElyseZinfandel vin:RedWine)   → true
(typep vin:ElyseZinfandel vin:TableWine) → true

(get-form vin:ElyseZinfandel) →
(vin:Zinfandel vin:ElyseZinfandel
 (vin:hasMaker vin:Elyse)
 (vin:hasSugar vin:Dry)
 (vin:hasFlavor vin:Moderate)
 (vin:hasBody vin:Full)
 (vin:locatedIn vin:NapaRegion)
 (vin:hasColor vin:Red)
 (vin:madeFromGrape vin:ZinfandelGrape))

(get-form vin:RedWine) →
(owl:Class vin:RedWine
 (owl:intersectionOf vin:Wine
  (owl:Restriction (owl:onProperty vin:hasColor)
                   (owl:hasValue vin:Red))))
```

Intersection of Classes

```
(subsumed-p vin:Zinfandel vin:RedWine)   → true
(subsumed-p vin:Zinfandel vin:TableWine) → true

(get-form vin:Zinfandel) →
(owl:Class vin:Zinfandel (rdf:about "#Zinfandel")
 (rdfs:subClassOf (owl:Restriction (owl:onProperty vin:hasColor)
                                   (owl:hasValue vin:Red))
                  (owl:Restriction (owl:onProperty vin:hasSugar)
                                   (owl:hasValue vin:Dry))
                  (owl:Restriction (owl:onProperty vin:hasBody)
                                   (owl:allValuesFrom (owl:Class #)))
                  (owl:Restriction (owl:onProperty vin:hasFlavor)
                                   (owl:allValuesFrom (owl:Class #))))
 (owl:intersectionOf vin:Wine
  (owl:Restriction (owl:onProperty vin:madeFromGrape)
                   (owl:hasValue vin:ZinfandelGrape))
  (owl:Restriction (owl:onProperty vin:madeFromGrape)
                   (owl:maxCardinality 1))))
```

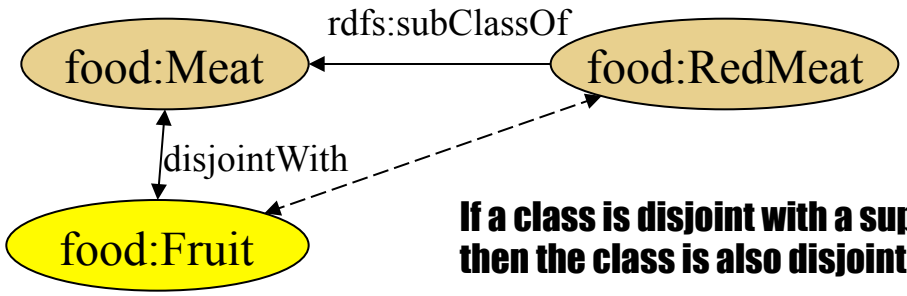
Take care of rdfs:subClassOf and 1st parameter of subtypep

Intersection of Classes (Complex)

```
(subsumed-p vin:WhiteNonSweetWine vin:DryWhiteWine)→ cl:nil
(subsumed-p vin:DryWhiteWine vin:WhiteNonSweetWine)→ true

(get-form vin:DryWhiteWine)→
(owl:Class vin:DryWhiteWine
 (owl:intersectionOf vin:DryWine vin:WhiteWine))

(get-form vin:WhiteNonSweetWine)→
(owl:Class vin:WhiteNonSweetWine
 (owl:intersectionOf vin:WhiteWine
  (owl:Restriction (owl:onProperty vin:hasSugar)
                   (owl:allValuesFrom
                    (owl:Class
                     (owl:oneOf vin:Dry vin:OffDry))))))
(get-form vin:DryWine) →
(owl:Class vin:DryWine
 (owl:intersectionOf vin:Wine
  (owl:Restriction (owl:onProperty vin:hasSugar)
                   (owl:hasValue vin:Dry))))
```

Satisfiability Checking

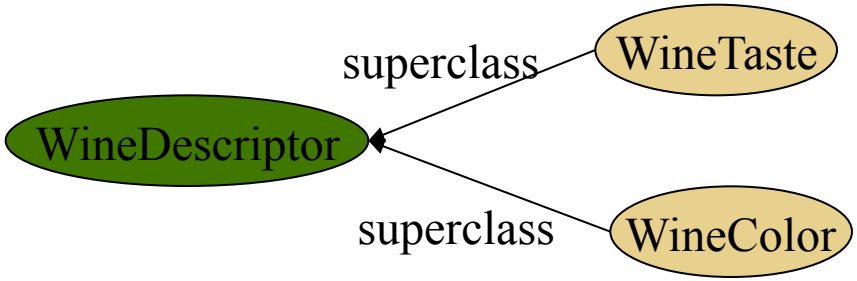
```
(defResource TheSpecialCourse (rdf:type owl:Class)
  (owl:intersectionOf
    food:RedMeatCourse
    (owl:Restriction (owl:onProperty food:hasFood)
                     (owl:allValuesFrom food:Fruit))))

(defIndividual No1SpecialCourse (rdf:type TheSpecialCourse)
               (food:hasFood food:Meat food:Bananas))
Error: Unsatisfiable by disjont pair in
       (#<owl:Class food:Fruit> #<owl:Class food:RedMeat>) for
       TheSpecialCourse food:hasFood
```



If a class is disjoint with a superclass,
then the class is also disjoint with the subclasses.

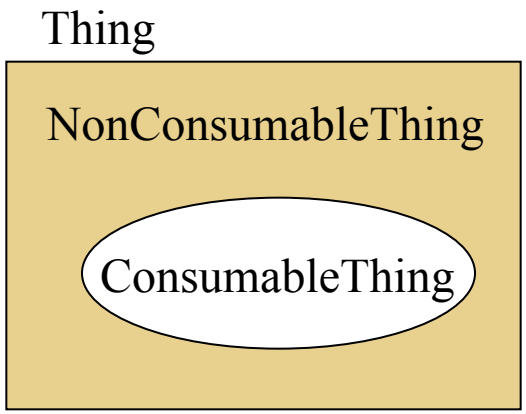
Union of Classes

```
(get-form vin:WineDescriptor) →
(owl:Class vin:WineDescriptor
 (rdfs:comment "Made WineDescriptor unionType of tastes and color")
 (owl:unionOf vin:WineTaste vin:WineColor))

(subtypep vin:WineTaste vin:WineDescriptor) → true
(subtypep vin:WineColor vin:WineDescriptor) → true

(typep vin:Red vin:WineDescriptor) → true
```



Complement of a Class

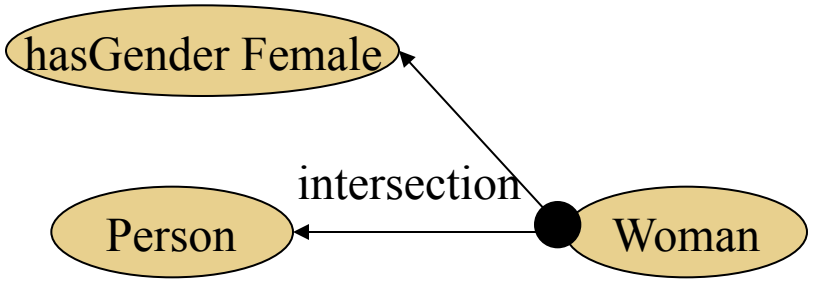
```
(collect-all-extensions-of owl:complementOf) →
((#<owl:Class food:NonConsumableThing> #<owl:Class food:ConsumableThing>)
 (#<owl:Class owl:Nothing> #<owl:Class owl:Thing>))

(owl-disjoint-p food:NonConsumableThing food:ConsumableThing)
 → true
(superclasses-of food:ConsumableThing) → (#<owl:Class owl:Thing>)
(superclasses-of food:NonConsumableThing) →
(#<owl:Class owl:Thing>)
```



Family Ontology

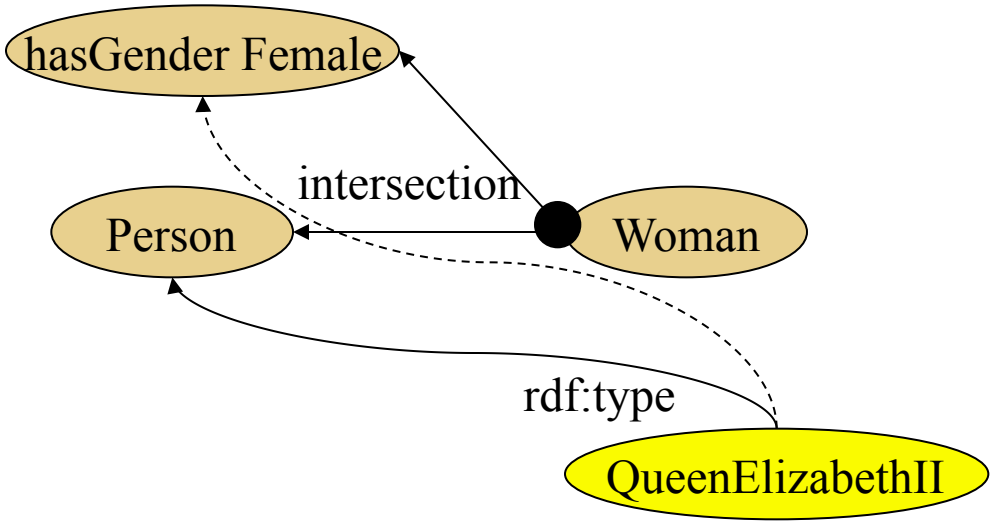
```
(defIndividual Female (rdf:type Gender)
  (owl:differentFrom Male))
Warning: Range entailX3 by owl:differentFrom: Male rdf:type owl:Thing

(defResource Person (rdf:type owl:Class)
  (owl:intersectionOf
    Human
    (owl:Restriction (owl:onProperty hasGender)
                     (owl:cardinality 1))))
Warning: Range entailX3 by owl:onProperty: hasGender rdf:type rdf:Propert

(defResource Woman (rdf:type owl:Class)
  (owl:intersectionOf
    Person
    (owl:Restriction (owl:onProperty hasGender)
                     (owl:hasValue Female))))
```

Family Ontology

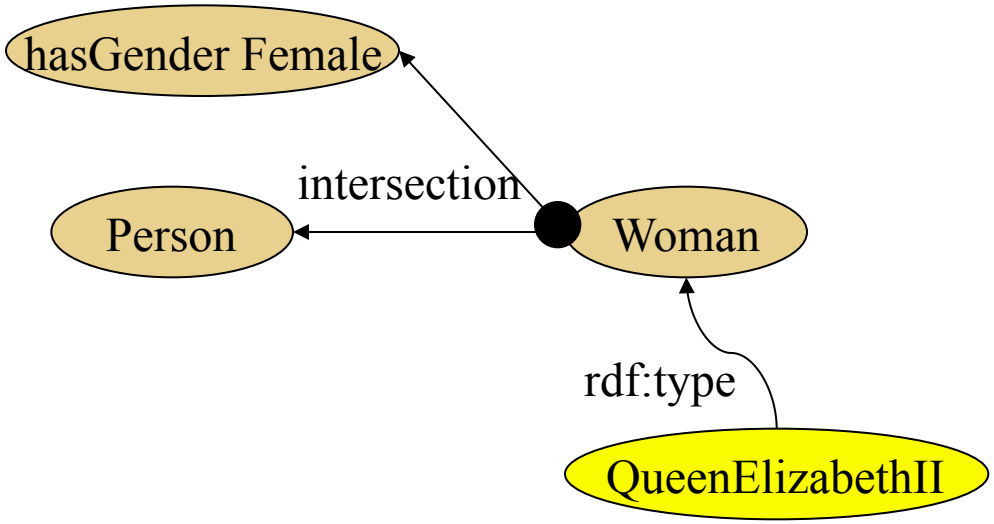



```
(defIndividual QueenElizabethII (rdf:type Person)
  (hasGender Female))
Warning: Entailed in refining: #<Person QueenElizabethII> to Woman.
```

Family Ontology



Family Ontology



Functional Property

```
(get-form vin:hasColor) →
((owl:ObjectProperty owl:FunctionalProperty) vin:hasColor
 (rdfs:domain vin:Wine) (rdfs:range vin:WineColor)
 (rdfs:subPropertyOf vin:hasWineDescriptor))
(-> vin:ChateauMargaux vin:hasColor)
→ #<vin:WineColor vin:Red>
(setf (slot-value vin:ChateauMargaux 'vin:hasColor) '赤)
Warning: Entail by range: 赤 rdf:type vin:WineColor.
Warning: #<vin:WineColor 赤> should be one of
         (#<vin:WineColor vin:White> #<vin:WineColor
vin:Rose> #<vin:WineColor vin:Red>).
Warning: Entailing same individuals (#<vin:WineColor 赤>
#<vin:WineColor vin:Red>) by functional property
         vin:hasColor
→ 赤
(owl-same-p 赤 vin:Red) → true
```

Transitive Property

```
(get-form vin:locatedIn) →
(owl:TransitiveProperty vin:locatedIn
 (rdfs:domain owl:Thing) (rdfs:range vin:Region))

(subsumed-p vin:MargauxRegion vin:MedocRegion)→ true
(subsumed-p vin:MedocRegion vin:BordeauxRegion)→ true
(subsumed-p vin:MargauxRegion vin:BordeauxRegion)→ true

(subsumed-p vin:Margaux vin:Medoc)→ true
(subsumed-p vin:Medoc vin:Bordeaux)→ true
(subsumed-p vin:Margaux vin:Bordeaux)→ true

(get-form vin:Margaux)→
```

Miscellaneous Utilities

```
(collect-all-instances vin:Wine) →
(#<vin:Beaujolais vin:ChateauMorgonBeaujolais> #<vin:Margaux vin:ChateauMargaux>
 #<vin:Pauillac vin:ChateauLafiteRothschildPauillac>
 #<vin:Sauternes vin:ChateauDYchemSauterne>
 #<vin:StEmilion vin:ChateauChevalBlancStEmilion>
 #<vin:CotesDOr vin:ClosDeVougeotCotesDOr>
 #<vin:WhiteBurgundy vin:PulignyMontrachetWhiteBurgundy>
 #<vin:WhiteBurgundy vin:CortonMontrachetWhiteBurgundy>
 #<vin:Meursault vin:ChateauDeMeursaultMeursault>
 #<vin:CabernetFranc vin:WhitehallLaneCabernetFranc> ...)

(dah vin:Wine) →
(vin:Wine (vin:AlsatianWine) (vin:AmericanWine)
 (vin:Beaujolais vin:ChateauMorgonBeaujolais)
 (vin:Bordeaux
  (vin:Medoc (vin:Margaux vin:ChateauMargaux)
   (vin:Pauillac vin:ChateauLafiteRothschildPauillac))
  (vin:RedBordeaux) (vin:Sauternes vin:ChateauDYchemSauterne)
  (vin:StEmilion vin:ChateauChevalBlancStEmilion) (vin:WhiteBordeaux))
 (vin:Burgundy (vin:RedBurgundy (vin:CotesDOr vin:ClosDeVougeotCotesDOr))
  (vin:WhiteBurgundy (vin:Meursault vin:ChateauDeMeursaultMeursault)))
 (vin:CabernetFranc vin:WhitehallLaneCabernetFranc)
 (vin:CabernetSauvignon vin:SantaCruzMountainVineyardCabernetSauvignon
  vin:PageMillWineryCabernetSauvignon vin:MariettaCabernetSauvignon
  vin:FormanCabernetSauvignon)
 (vin:TexasWine) (vin:CaliforniaWine)
 (vin:Chardonnay vin:PeterMccoyChardonnay vin:MountadamChardonnay
  vin:MountEdenVineyardEdnaValleyChardonnay vin:FormanChardonnay vin:BancroftChardonnay)
```

Conclusions

- Semantic Web Processor SWCLOS is demonstrated with hands-on materials.
- SWCLOS is not completed yet. It is still evolving.
- Allegro Prolog is available in coupling with SWCLOS.
- Open Source
  - Everything is permitted except change package-name `gx`, and class name `galaxy` and `galaxy-class`

Email: SeijiKoide@aol.com