Meta-Circularity and MOP in Common Lisp for OWL Full

Seiji Koide
The Graduate Univ. Advanced Studies (SOKENDAI)

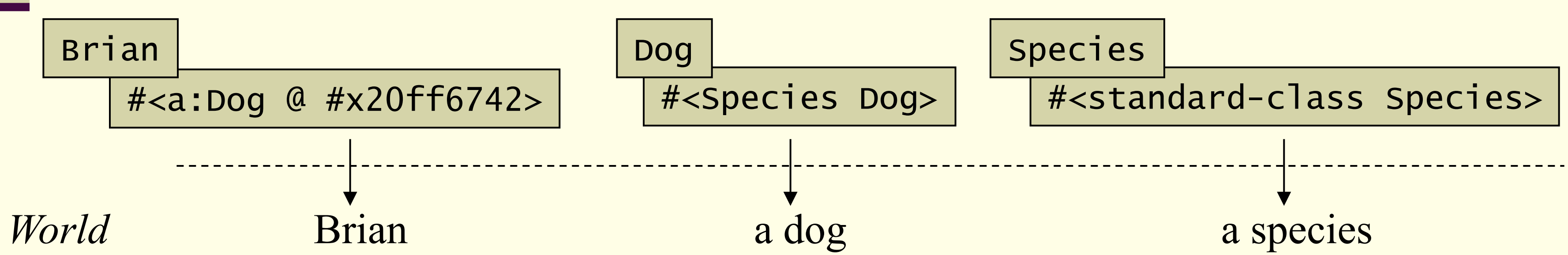
Hideaki Takeda
National Institute of Informatics and SOKENDAI

Computer Language Semantics

- Procedural Semantics
- Denotational Semantics
- Axiomatic Semantics

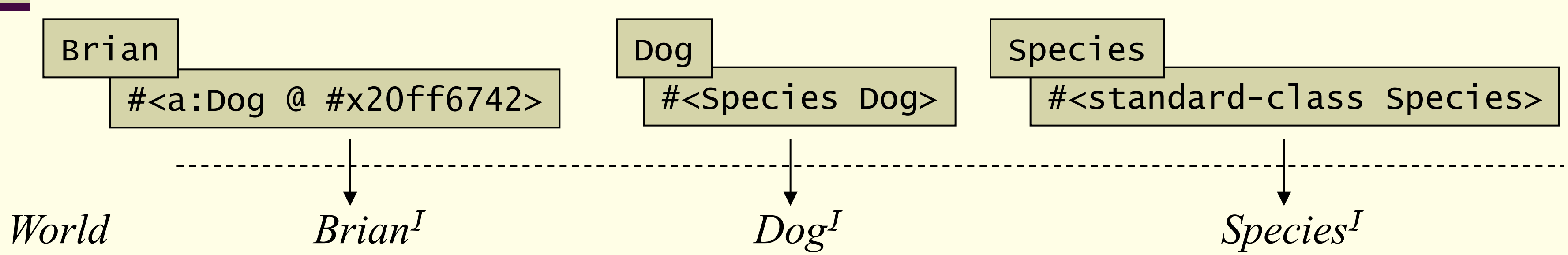
Denotational Semantics in CLOS

```
(defpackage :a
  (:export "Dog" "Brian" "Species"))
(defparameter a:Species
  (defclass a:Species (cl:standard-class) ()
    (:metaclass cl:standard-class)))
(defparameter a:Dog
  (defclass a:Dog () ()
    (:metaclass a:Species)))
(defparameter a:Brian
  (make-instance 'a:Dog))
```

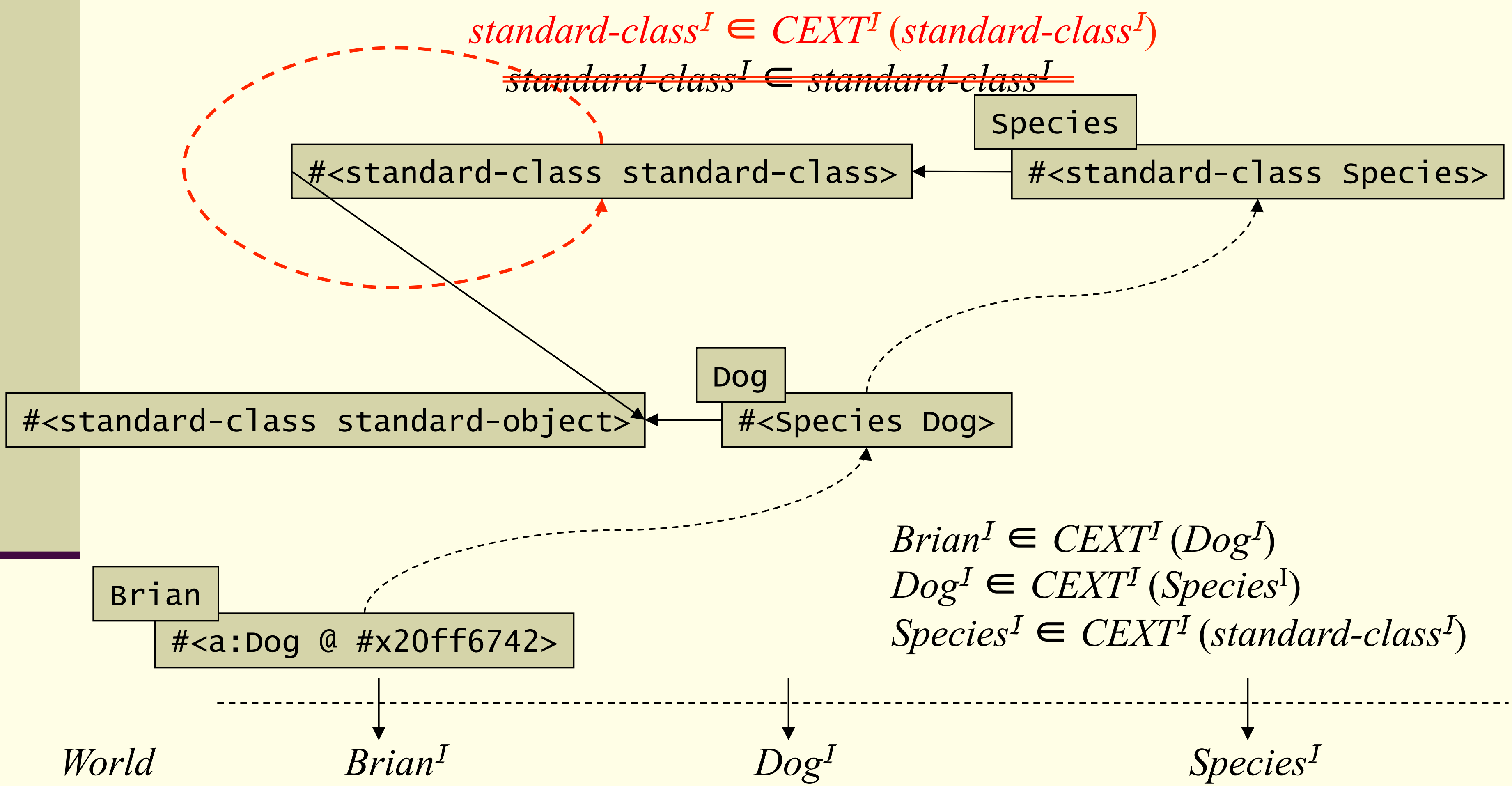


Denotational Semantics in CLOS

```
(defpackage :a
  (:export "Dog" "Brian" "Species"))
(defparameter a:Species
  (defclass a:Species (cl:standard-class) ()
    (:metaclass cl:standard-class)))
(defparameter a:Dog
  (defclass a:Dog () ()
    (:metaclass a:Species)))
(defparameter a:Brian
  (make-instance 'a:Dog))
```



Extensional Semantics in CLOS



Meta-Circularity of rdfs:Class

- Axiom1: Every class that is an instance of $rdfs{:}Class^{\mathcal{I}}$ is a subclass of $rdfs{:}Resource^{\mathcal{I}}$.
  - $x \in C^{\mathcal{I}} = CEXT^{\mathcal{I}}(rdfs{:}Class^{\mathcal{I}}) \Rightarrow x \sqsubseteq rdfs{:}Resource^{\mathcal{I}}$
- Axiom 2: Any class in super-subclass relation is an instance of $rdfs{:}Class^{\mathcal{I}}$.
  - $x \sqsubseteq y \Rightarrow x \in C^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}$
- Axiom 3: $rdfs{:}Class^{\mathcal{I}}$ is a subclass of $rdfs{:}Resource^{\mathcal{I}}$.
  - $rdfs{:}Class^{\mathcal{I}} \sqsubseteq rdfs{:}Resource^{\mathcal{I}} \Rightarrow rdfs{:}Class^{\mathcal{I}} \in C^{\mathcal{I}}$
    $\Rightarrow rdfs{:}Resource^{\mathcal{I}} \in C^{\mathcal{I}}$

Meta-Circularity of rdfs:Class

- Axiom1: Every class that is an instance of $rdfs{:}Class^{\mathcal{I}}$ is a subclass of $rdfs{:}Resource^{\mathcal{I}}$.
  - $x \in C^{\mathcal{I}} = CEXT^{\mathcal{I}}(rdfs{:}Class^{\mathcal{I}}) \Rightarrow x \sqsubseteq rdfs{:}Resource^{\mathcal{I}}$
- Axiom 2: Any class in super-subclass relation is an instance of $rdfs{:}Class^{\mathcal{I}}$.
  - $x \sqsubseteq y \Rightarrow x \in C^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}$
- Axiom 3: $rdfs{:}Class^{\mathcal{I}}$ is a subcla

Meta-Circularity

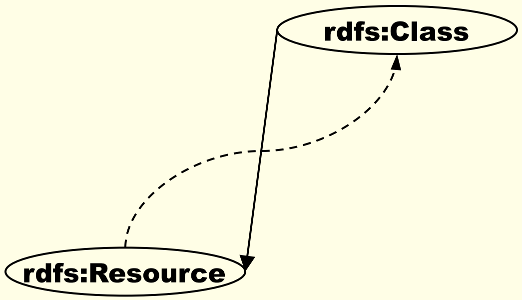
$rdfs{:}Resource^{\mathcal{I}}$.
  - $rdfs{:}Class^{\mathcal{I}} \sqsubseteq rdfs{:}Resource^{\mathcal{I}} \Rightarrow rdfs{:}Class^{\mathcal{I}} \in CEXT^{\mathcal{I}}(rdfs{:}C$
    $\Rightarrow rdfs{:}Resource^{\mathcal{I}} \in CEXT^{\mathcal{I}}(rd$
    $\Rightarrow rdfs{:}Resource^{\mathcal{I}} \sqsubseteq rdfs{:}Resou$

Meta-Circularity of rdfs:Class

- Axiom1: Every class that is an instance of $rdfs{:}Class^{\mathcal{I}}$ is a subclass of $rdfs{:}Resource^{\mathcal{I}}$.
  - $x \in C^{\mathcal{I}} = CEXT^{\mathcal{I}}(rdfs{:}Class^{\mathcal{I}}) \Rightarrow x \sqsubseteq rdfs{:}Resource^{\mathcal{I}}$
- Axiom 2: Any class in super-subclass relation is an instance of $rdfs{:}Class^{\mathcal{I}}$.
  - $x \sqsubseteq y \Rightarrow x \in C^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}$
- Axiom 3: $rdfs{:}Class^{\mathcal{I}}$ is a subclass of $rdfs{:}Resource^{\mathcal{I}}$.
  - $rdfs{:}Class^{\mathcal{I}} \sqsubseteq rdfs{:}Resource^{\mathcal{I}} \Rightarrow rdfs{:}Class^{\mathcal{I}} \in CEXT^{\mathcal{I}}(rdfs{:}C$
    $\Rightarrow rdfs{:}Resource^{\mathcal{I}} \in CEXT^{\mathcal{I}}(rd$
    $\Rightarrow rdfs{:}Resource^{\mathcal{I}} \sqsubseteq rdfs{:}Resou$

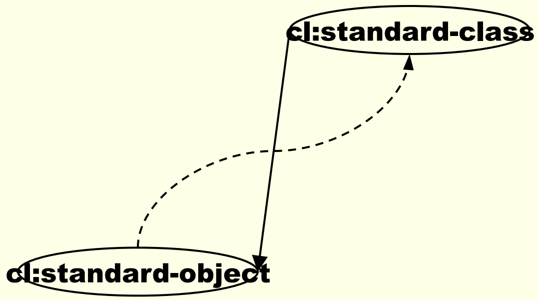
Twist Relation of rdfs:Resource and rdfs:Class

- $rdfs{:}Class^{\mathcal{I}} \sqsubseteq rdfs{:}Resource^{\mathcal{I}} \wedge rdfs{:}Resource^{\mathcal{I}} \in CEXT^{\mathcal{I}}(rdfs{:}Class^{\mathcal{I}})$



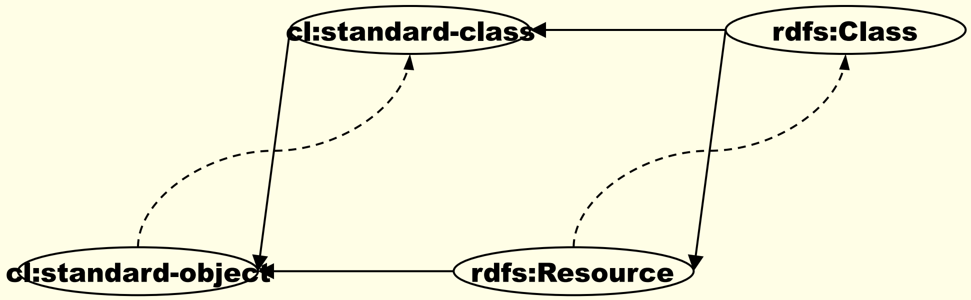
Twist Relation of cl:standard-object and cl:standard-class

- $cl{:}standard\text{-}class^{\mathcal{I}} \sqsubseteq cl{:}standard\text{-}object^{\mathcal{I}} \wedge cl{:}standard\text{-}object^{\mathcal{I}} \in CEXT^{\mathcal{I}}(cl{:}standard\text{-}class^{\mathcal{I}})$



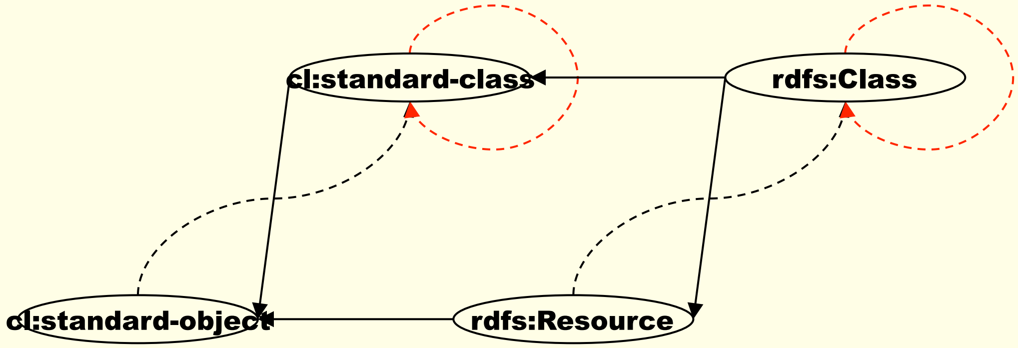
Twist Relation of rdfs:Resource and rdfs:Class

- $cl{:}standard\text{-}class^{\mathcal{I}} \sqsubseteq cl{:}standard\text{-}object^{\mathcal{I}} \wedge cl{:}standard\text{-}object^{\mathcal{I}} \in CEXT^{\mathcal{I}}(cl{:}standard\text{-}class^{\mathcal{I}})$
- $rdfs{:}Class^{\mathcal{I}} \sqsubseteq rdfs{:}Resource^{\mathcal{I}} \wedge rdfs{:}Resource^{\mathcal{I}} \in CEXT^{\mathcal{I}}(rdfs{:}Class^{\mathcal{I}})$



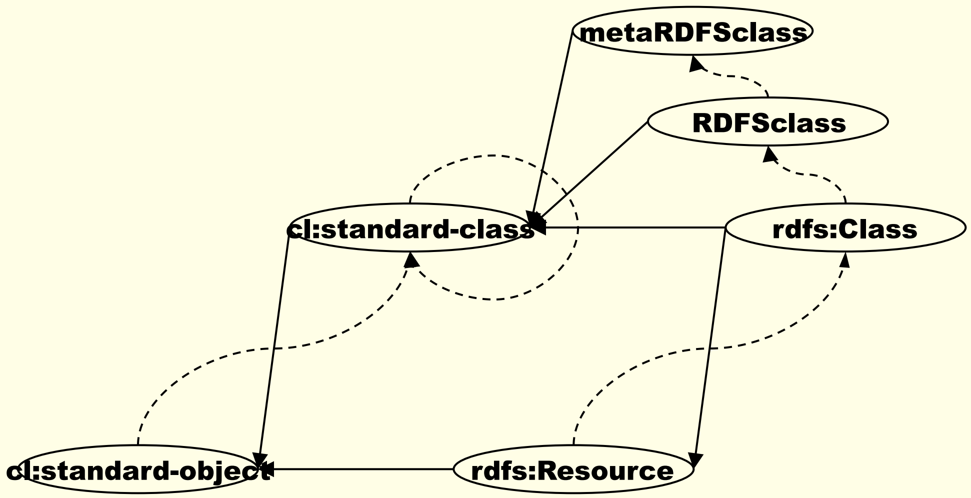
Twist Relation of rdfs:Resource and rdfs:Class

- $cl{:}standard\text{-}class^I \sqsubseteq cl{:}standard\text{-}object^I \wedge cl{:}standard\text{-}object^I \in CEXT^I(cl{:}standard\text{-}class^I)$
- $rdfs{:}Class^I \sqsubseteq rdfs{:}Resource^I \wedge rdfs{:}Resource^I \in CEXT^I(rdfs{:}Class^I)$



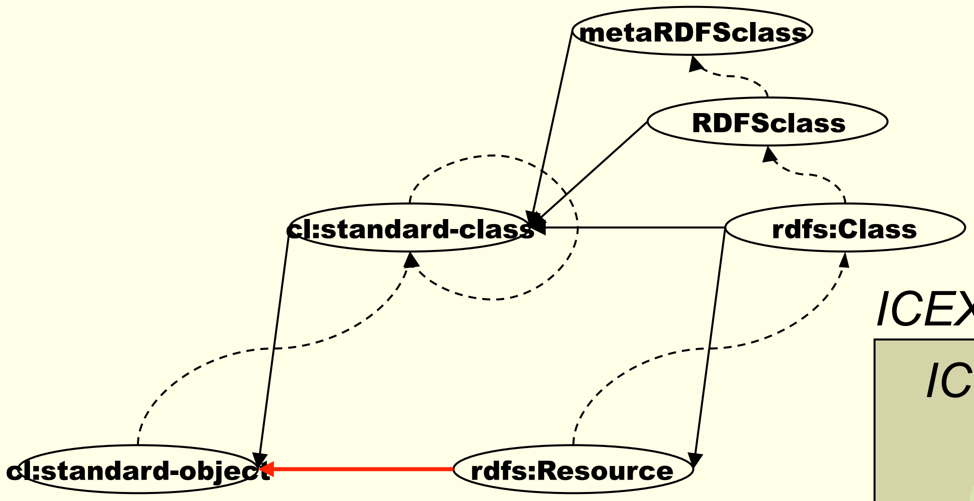
Twist Relation of rdfs:Resource and rdfs:Class

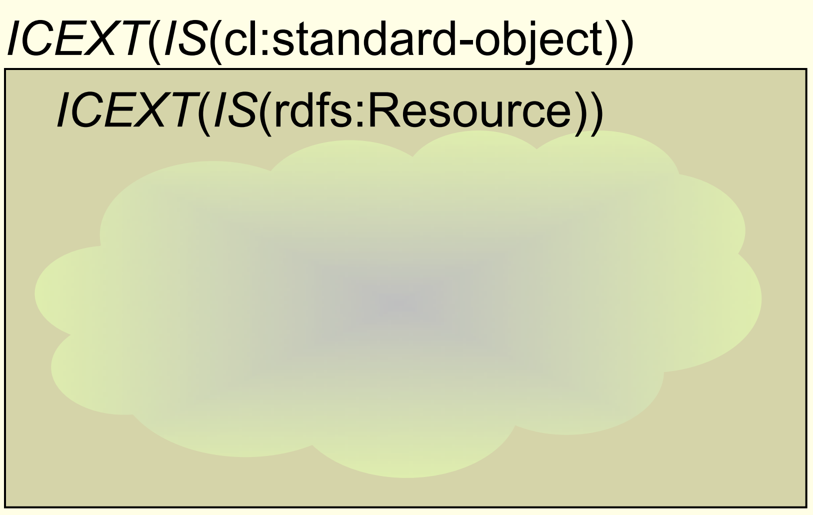
- $cl{:}standard\text{-}class^{\mathcal{I}} \sqsubseteq cl{:}standard\text{-}object^{\mathcal{I}} \wedge cl{:}standard\text{-}object^{\mathcal{I}} \in CEXT^{\mathcal{I}}(cl{:}standard\text{-}class^{\mathcal{I}})$



Twist Relation of rdfs:Resource and rdfs:Class

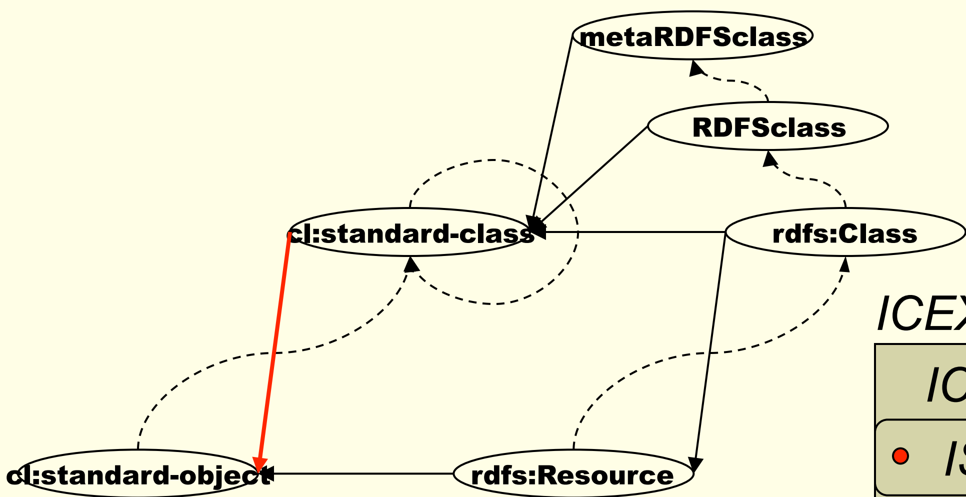
- $cl{:}standard\text{-}class^{I} \sqsubseteq cl{:}standard\text{-}object^{I} \wedge cl{:}standard\text{-}object^{I} \in CEXT^{I}(cl{:}standard\text{-}class^{I})$

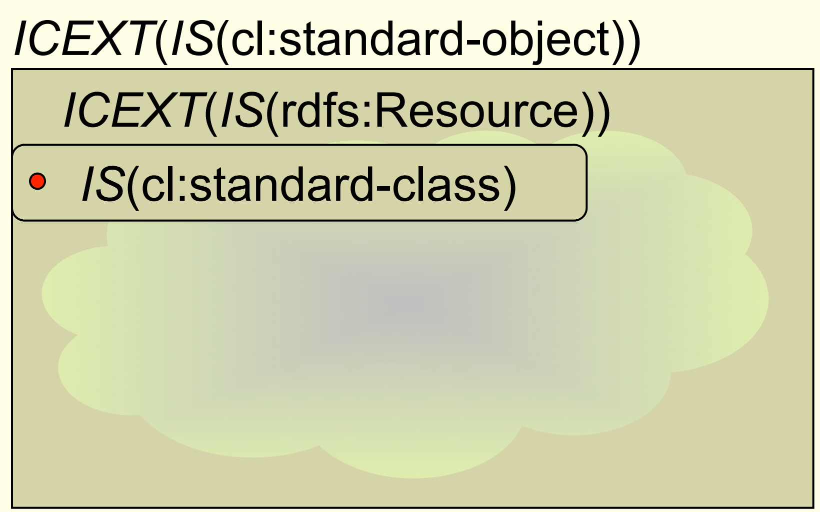




Twist Relation of rdfs:Resource and rdfs:Class

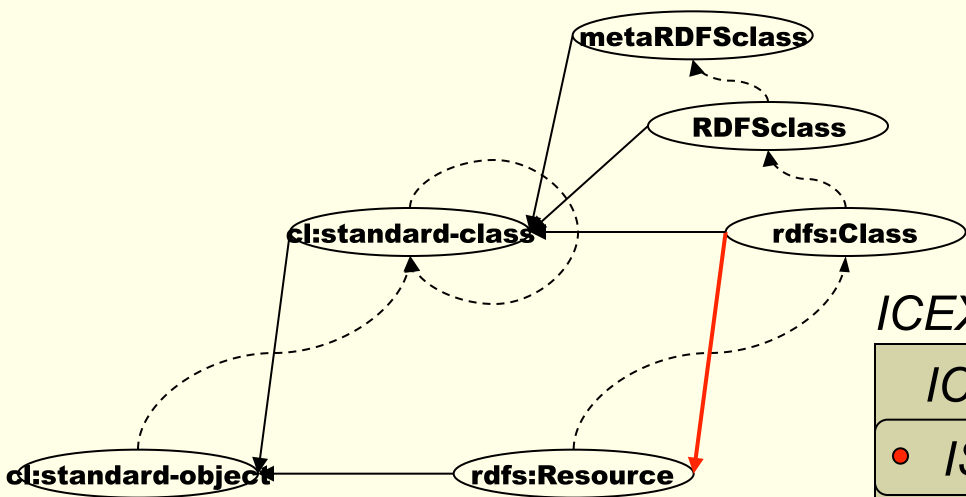
- $cl{:}standard\text{-}class^{\mathcal{I}} \sqsubseteq cl{:}standard\text{-}object^{\mathcal{I}} \wedge cl{:}standard\text{-}object^{\mathcal{I}} \in CEXT^{\mathcal{I}}(cl{:}standard\text{-}class^{\mathcal{I}})$

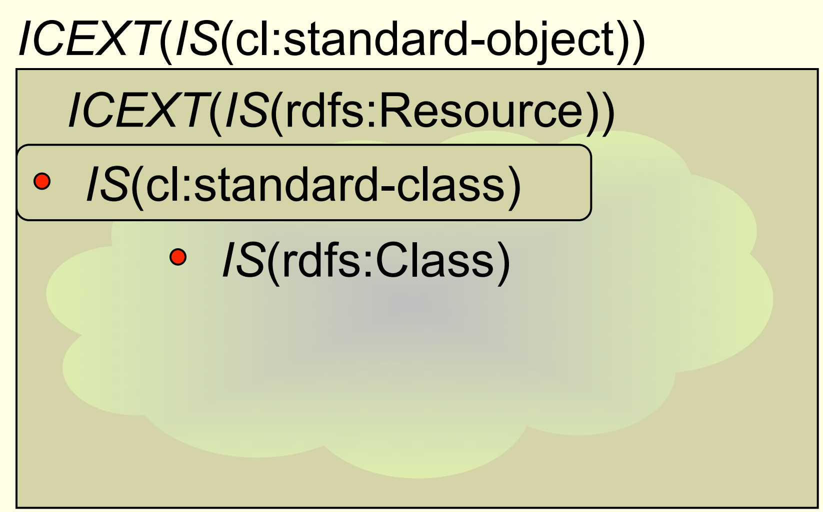




Twist Relation of rdfs:Resource and rdfs:Class

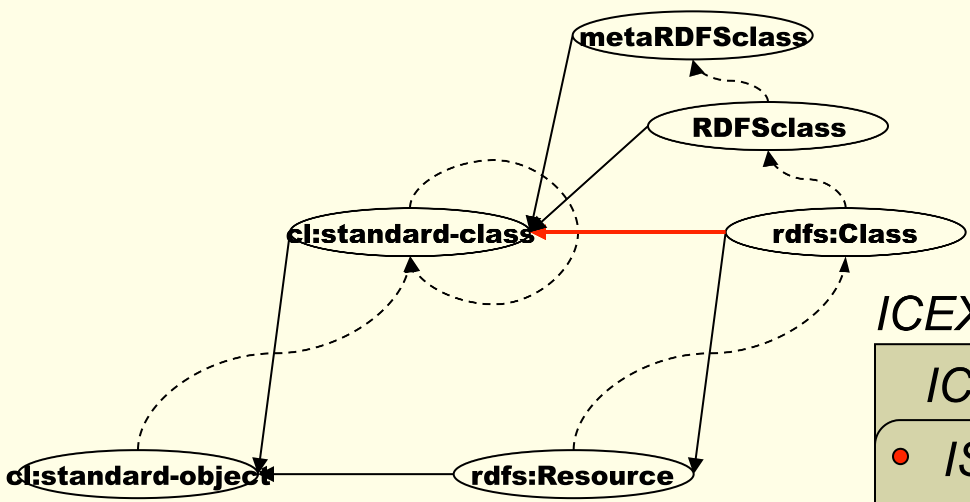
- $cl{:}standard\text{-}class^{\mathcal{I}} \sqsubseteq cl{:}standard\text{-}object^{\mathcal{I}} \wedge$
  $cl{:}standard\text{-}object^{\mathcal{I}} \in CEXT^{\mathcal{I}}(cl{:}standard\text{-}class^{\mathcal{I}})$

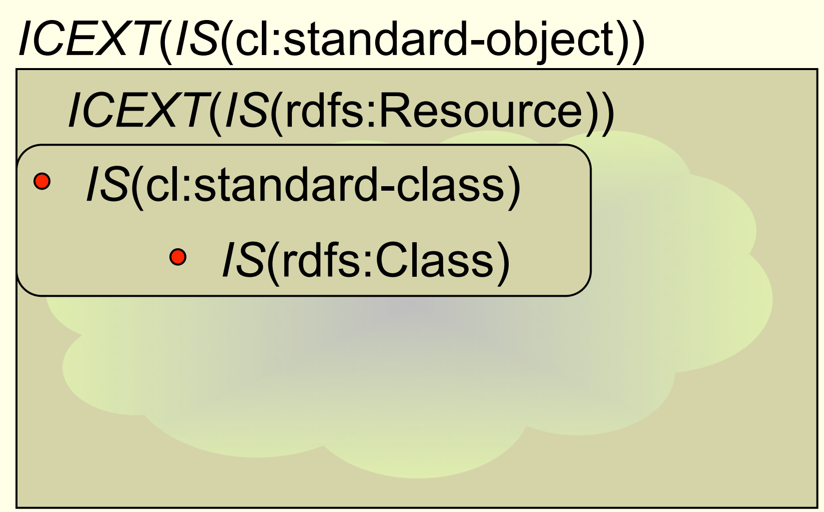




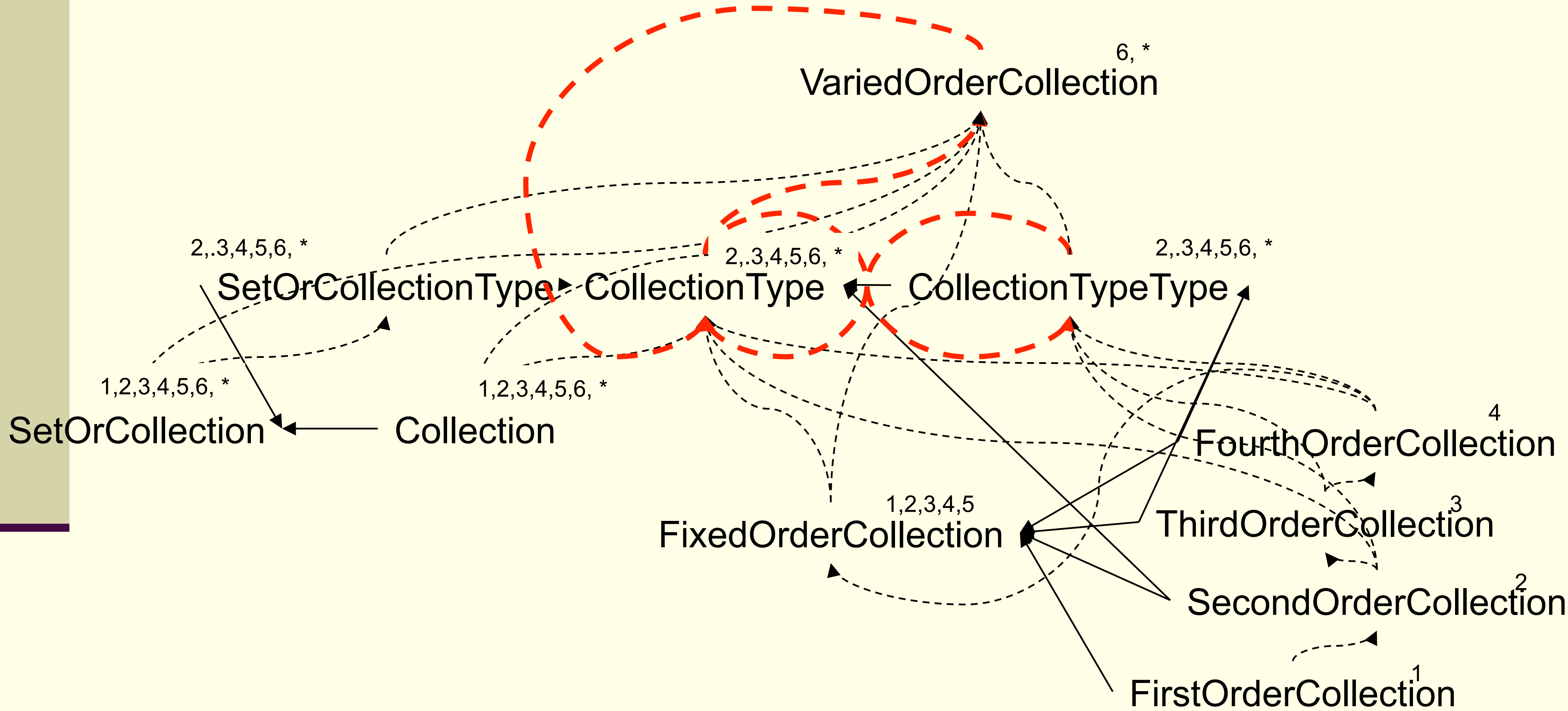
Twist Relation of rdfs:Resource and rdfs:Class

- $cl{:}standard\text{-}class^{I} \sqsubseteq cl{:}standard\text{-}object^{I} \wedge cl{:}standard\text{-}object^{I} \in CEXT^{I}(cl{:}standard\text{-}class^{I})$





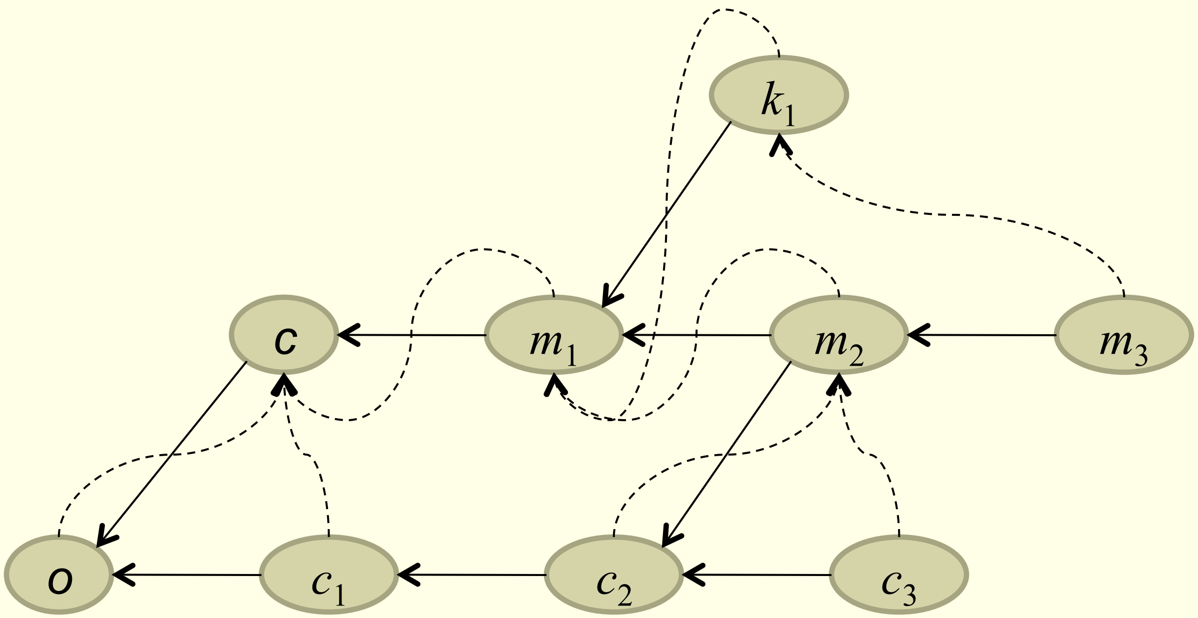
Cyc Ontology



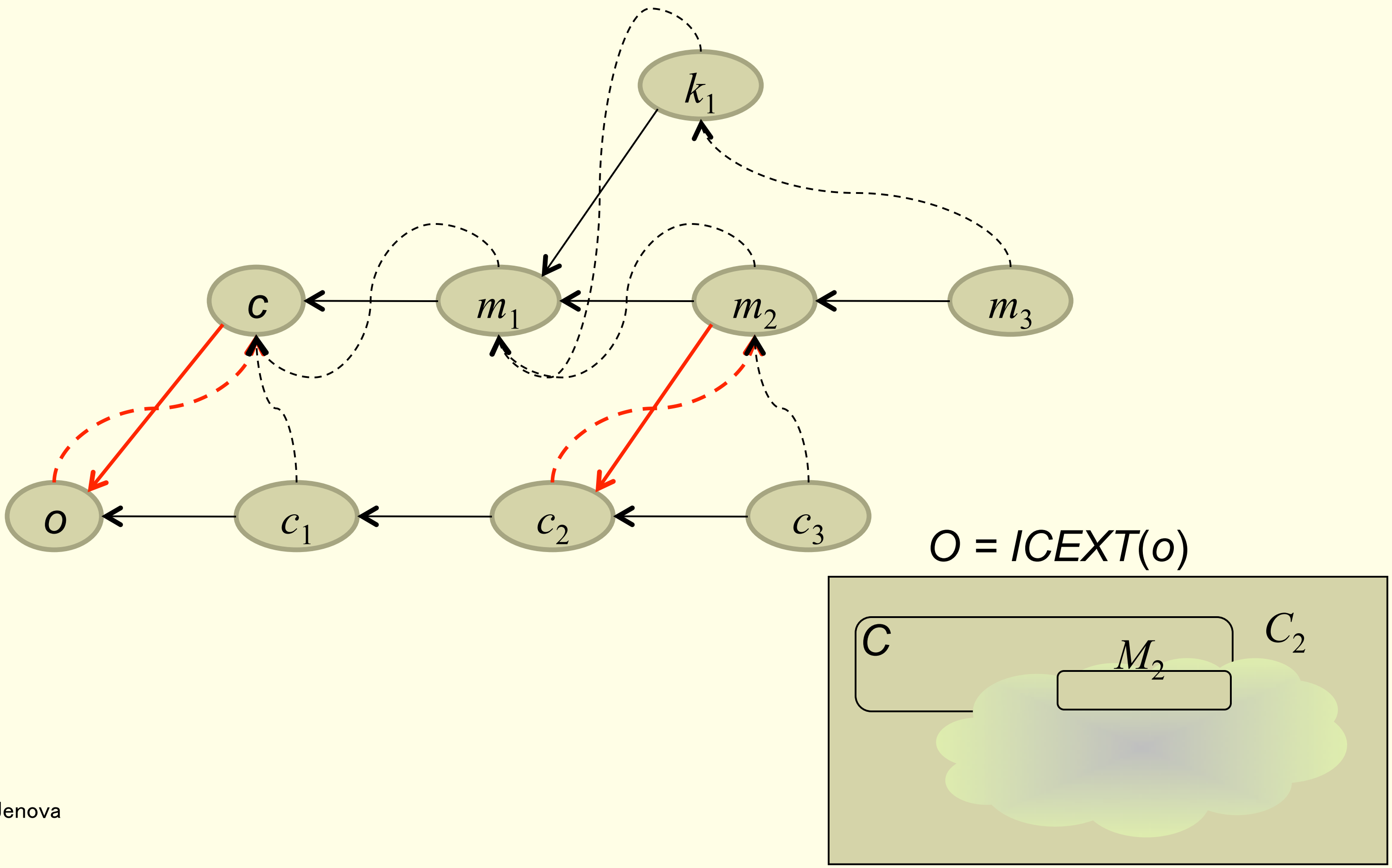
CLOS-Clean Ontology

- No class can have any direct and indirect cyclic loop in subclass relation.
- No class can have any direct and indirect membership loop but the direct loop of cl:standard-class.
- A parallel relation of subclass and membership can exist anywhere among metaclasses.
- A twist relation of subclass and membership can exist anywhere among classes and metaclasses.

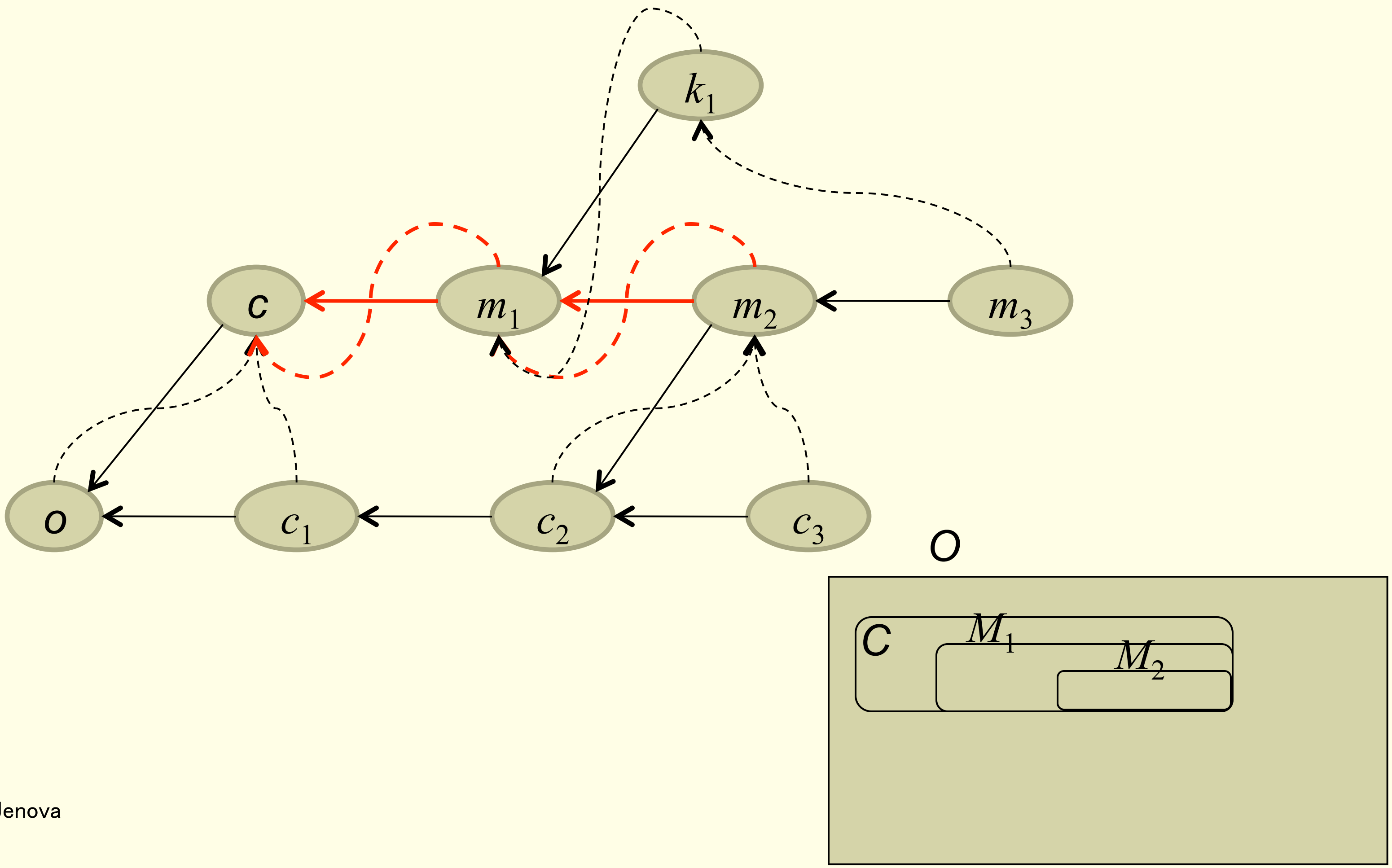
CLOS Clean Ontology Pattern



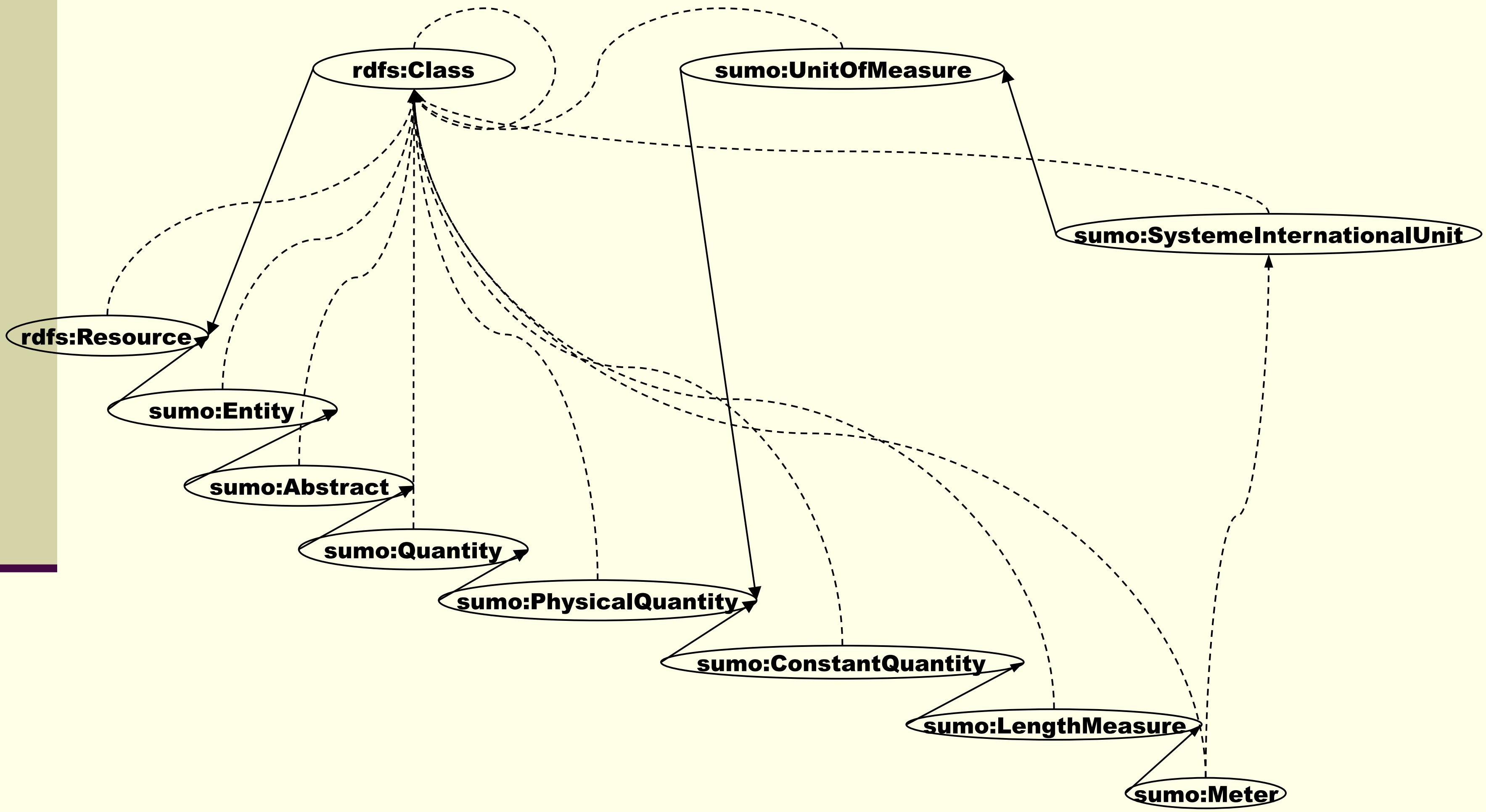
CLOS Clean Ontology Pattern



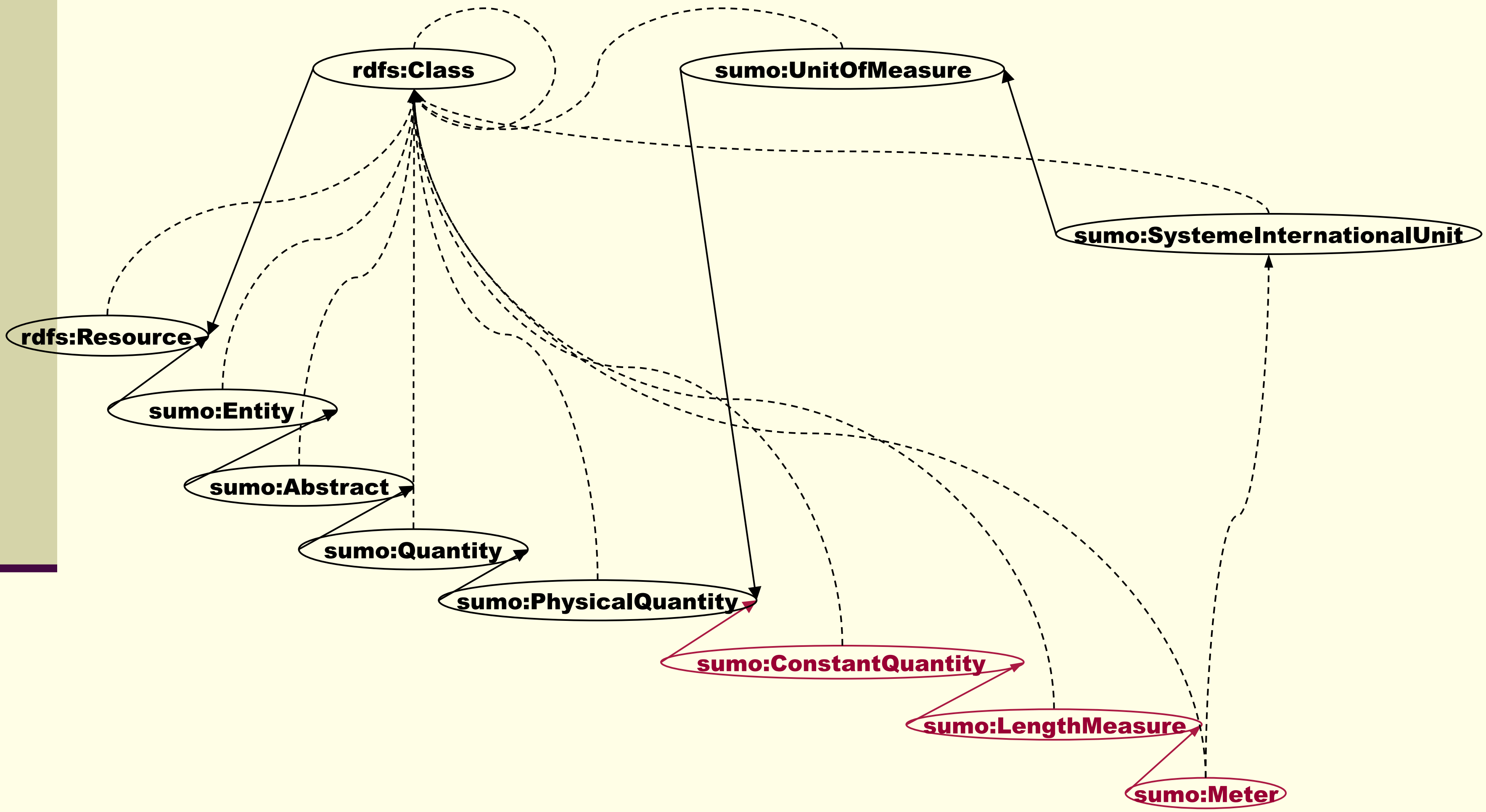
CLOS Clean Ontology Pattern



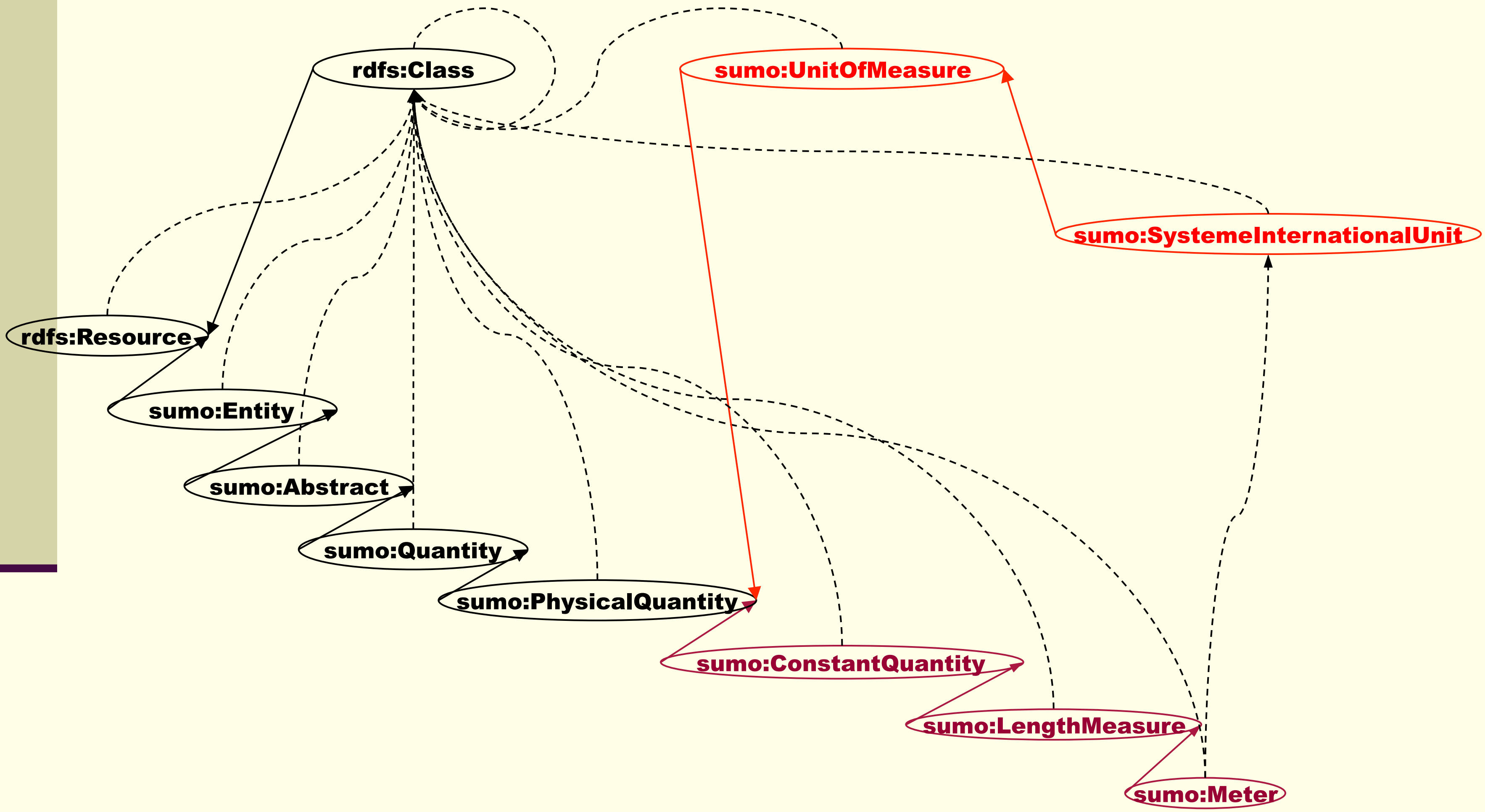
Naive Hierarchical Structure of SUMO



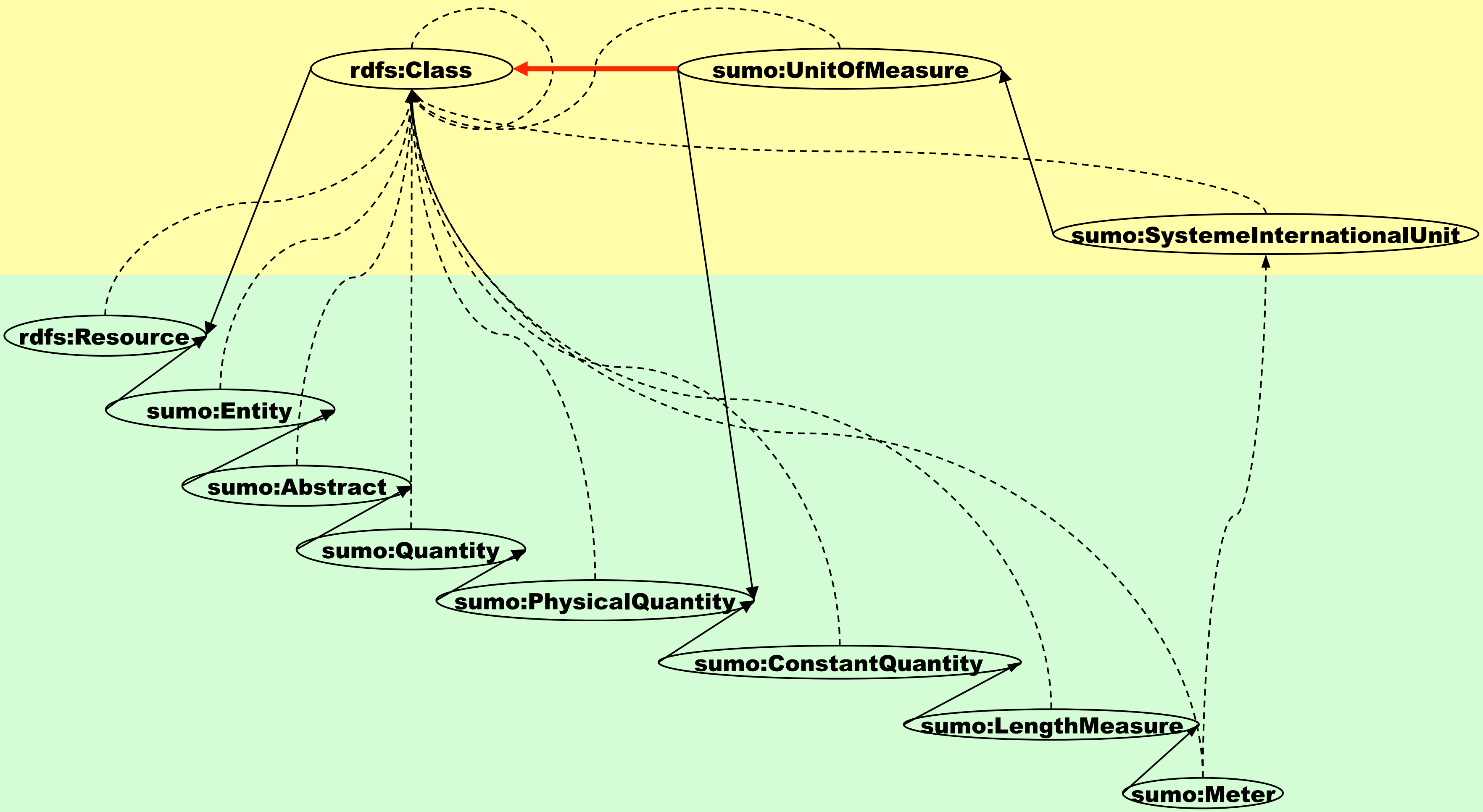
Naive Hierarchical Structure of SUMO



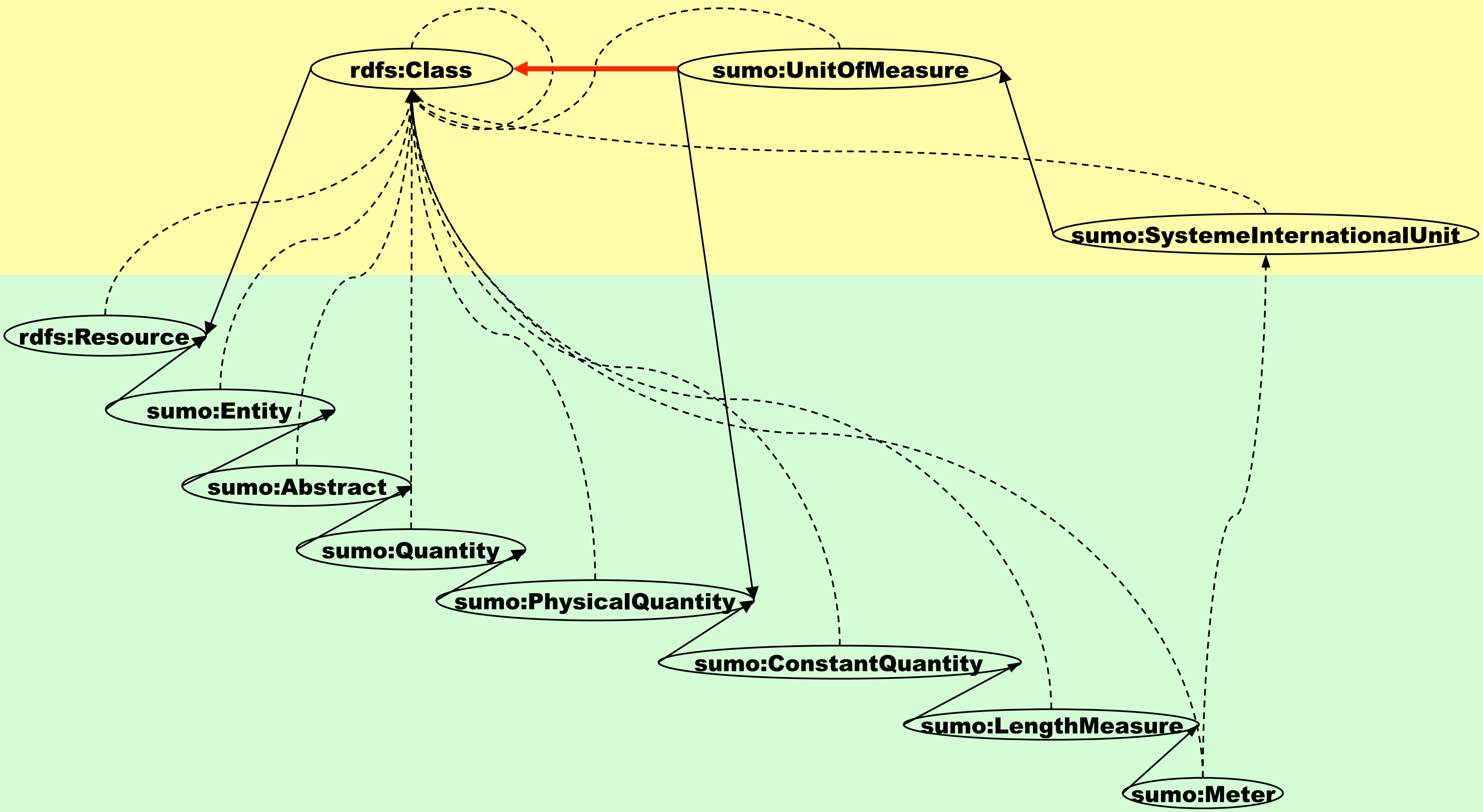
Naive Hierarchical Structure of SUMO



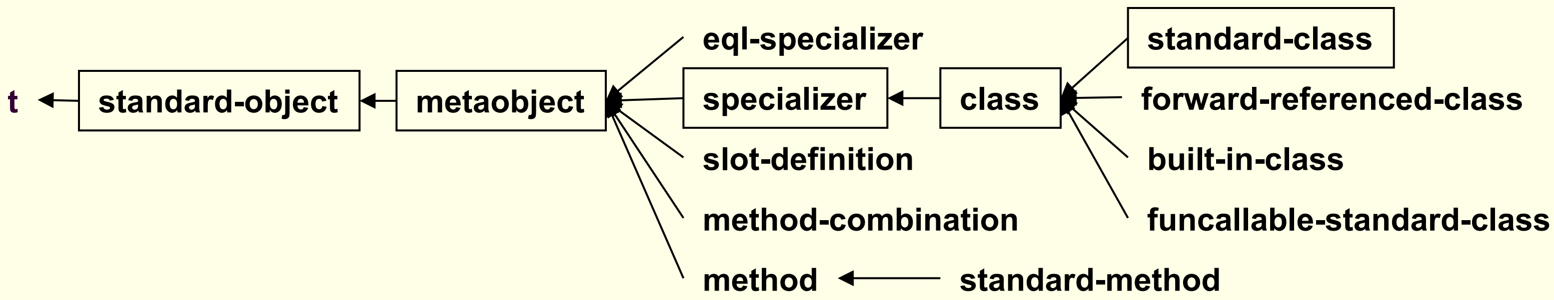
Naive Hierarchical Structure of SUMO

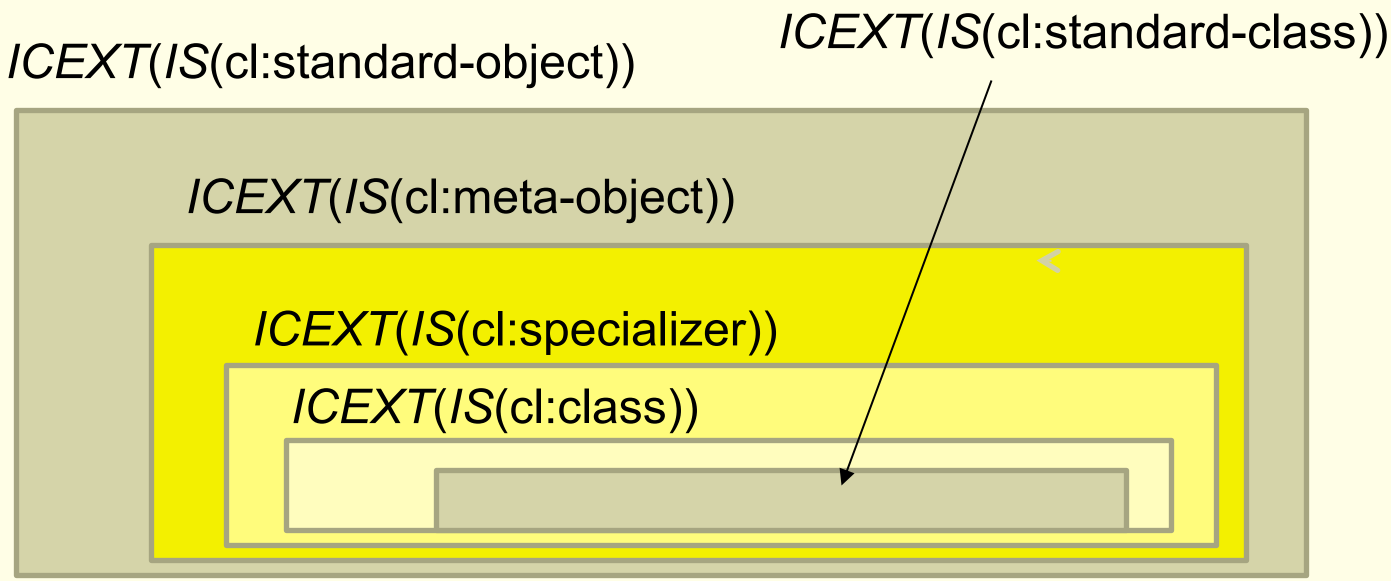


Hierarchical Structure of SUMO



Domain Inclusiveness of CLOS





Compound Type Specifieres

ANSI Common Lisp specifies the compound type specifieres;

- And
  - denotes the set of all objects of the type determined by the intersection of the type specs
- Or
  - denotes the set of all objects of the type determined by the union of the type specs
- Not
  - denotes the set of all objects that are not of the type specs

Compound Type Specifieres

ANSI Common Lisp specifies the compound type specifieres;

- And
  - denotes the set of all objects of the type determined by the intersection of the ~~type specs~~ class extensions

  determin?
- Or
  - denotes the set of all objects of the type determined by the union of the ~~type specs~~ class extensions

  determin?
- Not
  - denotes the set of all objects that are not of the ~~type specs~~ class extensions

Ternary Truth Table

- cl:subtypep
  - In ANSI Common Lisp returns two values. Then, ⟨t, t⟩ or ⟨nil, t⟩ when it is determined but ⟨nil, nil⟩ when it is not determined.

True

False

Unknown

value1	value2	meaning
t	t	Type1 is definitely a subtype of type2.
nil	t	Type1 is definitely not a subtype of type2.
nil	nil	Subtypep could not dtermin the relationship, so type1 might or might not be a subtype of type2

Ternary Truth Table

Conjunction

	True	Unknown	False
True	True	Unknown	False
Unknown	Unknown	Unknown	False
False	False	False	False

Disjunction

	True	Unknown	False
True	True	True	True
Unknown	True	Unknown	Unknown
False	True	Unknown	False

Negation

x	True	Unknown	False
(not x)	False	Unknown	False

NII

Ternary Truth Table

Rewriting Rules for Inclusiveness

If E	then E'
$C \subseteq (A \wedge B)$	$(C \subseteq A) \wedge (C \subseteq A)$
$C \subseteq (A \vee B)$	$(C \subseteq A) \vee (C \subseteq B)$
$(A \wedge B) \subseteq C$	$(A \subseteq C) \vee (B \subseteq C)$
$(A \vee B) \subseteq C$	$(A \subseteq C) \wedge (B \subseteq C)$
$(A \vee B) \subseteq (C \wedge D)$	$(A \subseteq C) \wedge (A \subseteq D) \wedge (B \subseteq C) \wedge (B \subseteq D)$
$(A \wedge B) \subseteq (C \vee D)$	$(A \subseteq C) \vee (A \subseteq D) \vee (B \subseteq C) \vee (B \subseteq D)$
$(A \wedge B) \subseteq (C \wedge D)$	$((A \subseteq C) \wedge (A \subseteq D)) \vee ((B \subseteq C) \wedge (B \subseteq D))$
$(A \vee B) \subseteq (C \vee D)$	$((A \subseteq D) \vee (A \subseteq D)) \wedge ((B \subseteq C) \vee (B \subseteq D))$
$\neg A \subseteq \neg B$	$B \subseteq A$

Simple Query for Compound Types

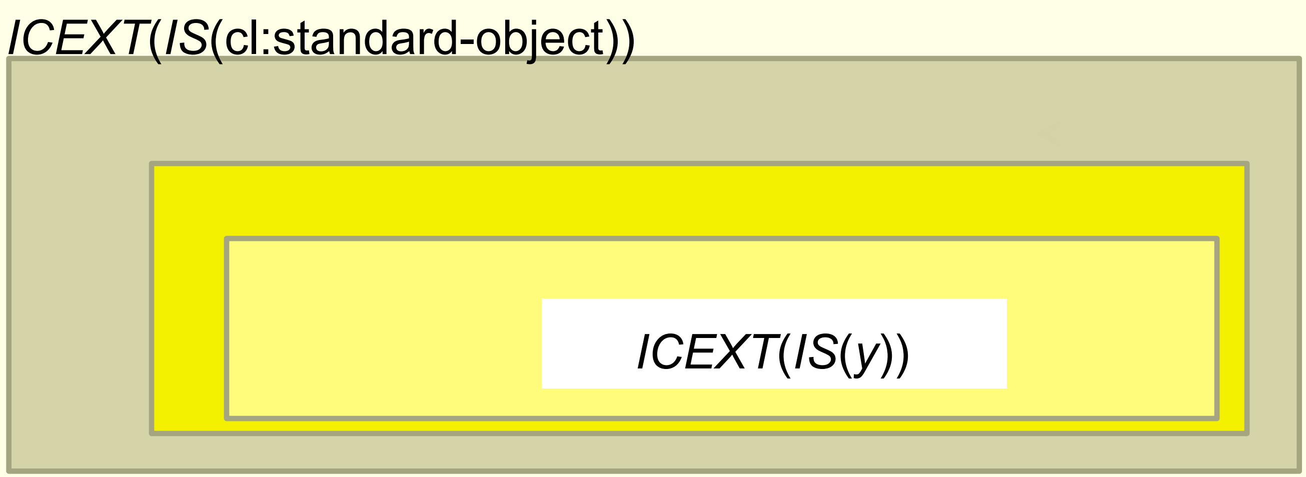
```
(defparameter a (defclass a () ()))
(defparameter b (defclass b () ()))

Query 1: (subtypep '(and a b) a)
Query 2: (subtypep a '(and a b))
Query 3: (subtypep '(or a b) a)
Query 4: (subtypep a '(or a b))
Query 5: (subtypep '(not a) a)
Query 6: (subtypep a '(not a))
```

system	Q1	Q2	Q3	Q4	Q5	Q6
A	True	False	False	True	Unknown	Unknown
B	False	False	False	True	False	False
C	True	False	False	True	False	False
D	True	False	False	True	False	False

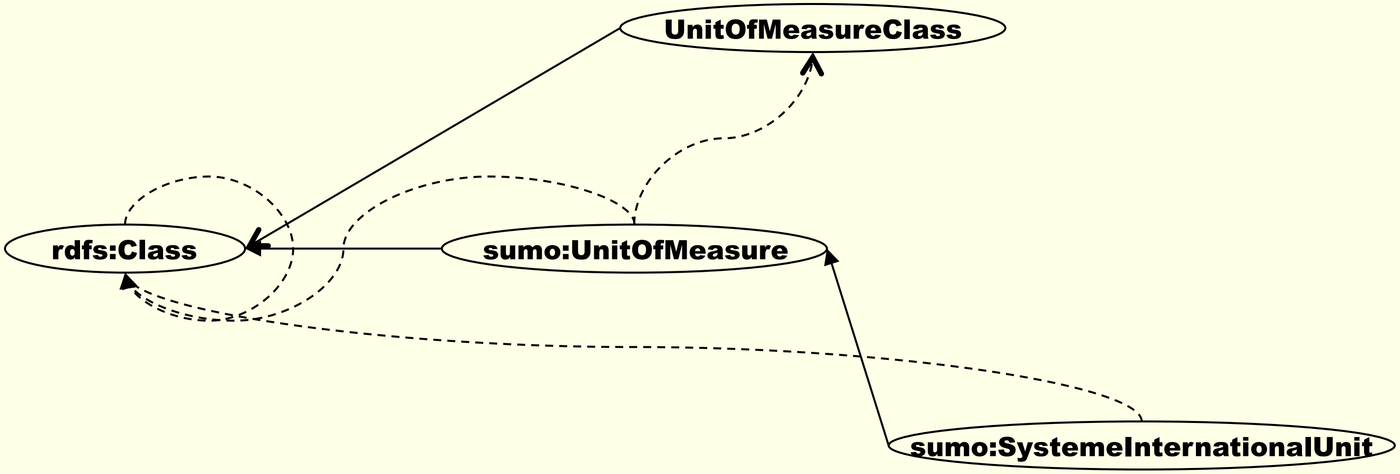
Simple Query for Compound Types

- And
  - $CEXT^{\mathcal{I}}(x \wedge y) \equiv CEXT^{\mathcal{I}}(x) \cap CEXT^{\mathcal{I}}(y)$
- Or
  - $CEXT^{\mathcal{I}}(x \vee y) \equiv CEXT^{\mathcal{I}}(x) \cup CEXT^{\mathcal{I}}(y)$
- Not
  - $\neg CEXT^{\mathcal{I}}(x) \equiv CEXT^{\mathcal{I}}(cl{:}standard\text{-}object) \setminus CEXT^{\mathcal{I}}(y)$



Meta-modeling and Reflection

- Meta-modeling in SWCLOS



Meta-modeling and Reflection

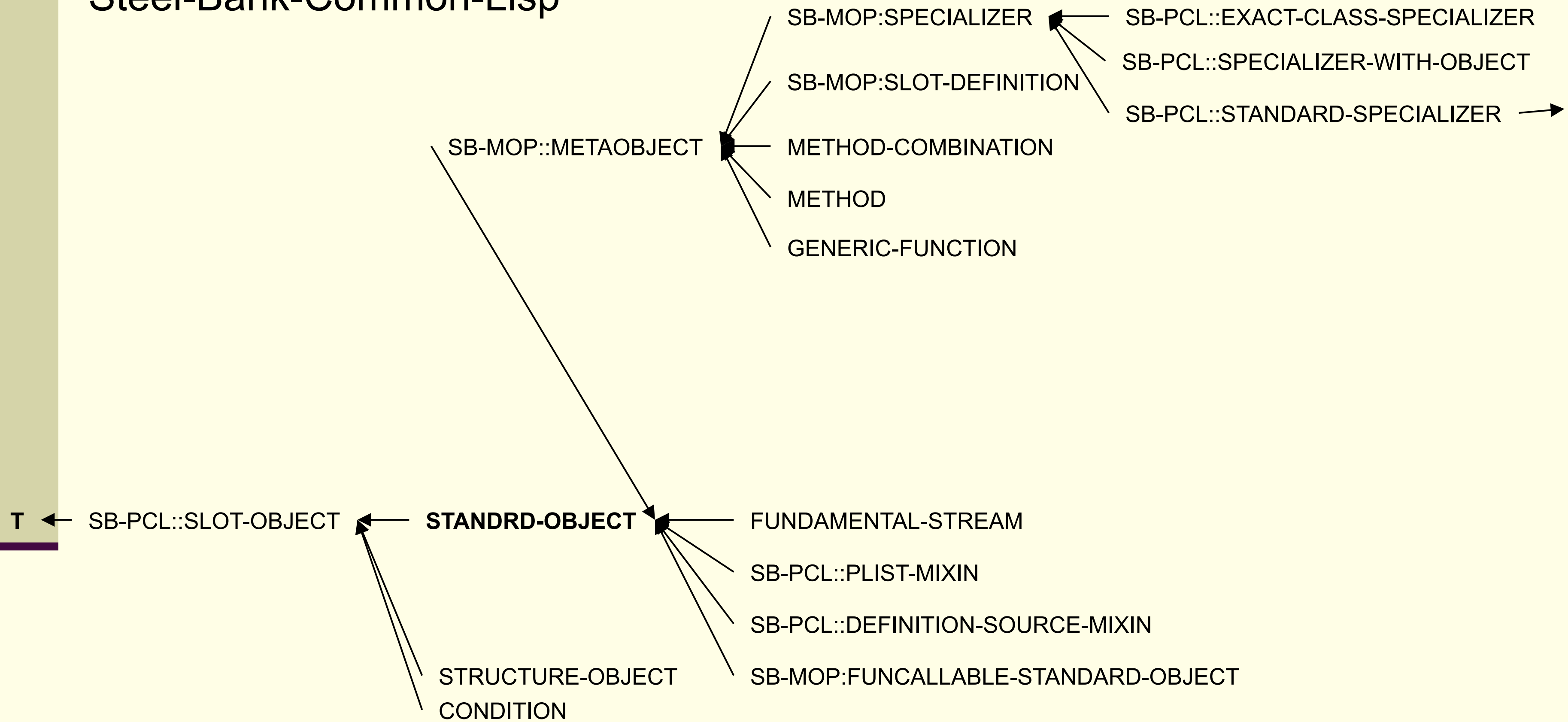
- Meta-modeling in SWCLOS

```
(defConcept UnitOfMeasureClass (rdfs:type rdfs:Class)
  (rdfs:subClassOf rdfs:Class))
(defProperty p1
  (rdfs:domain UnitofMeasureClass))
```

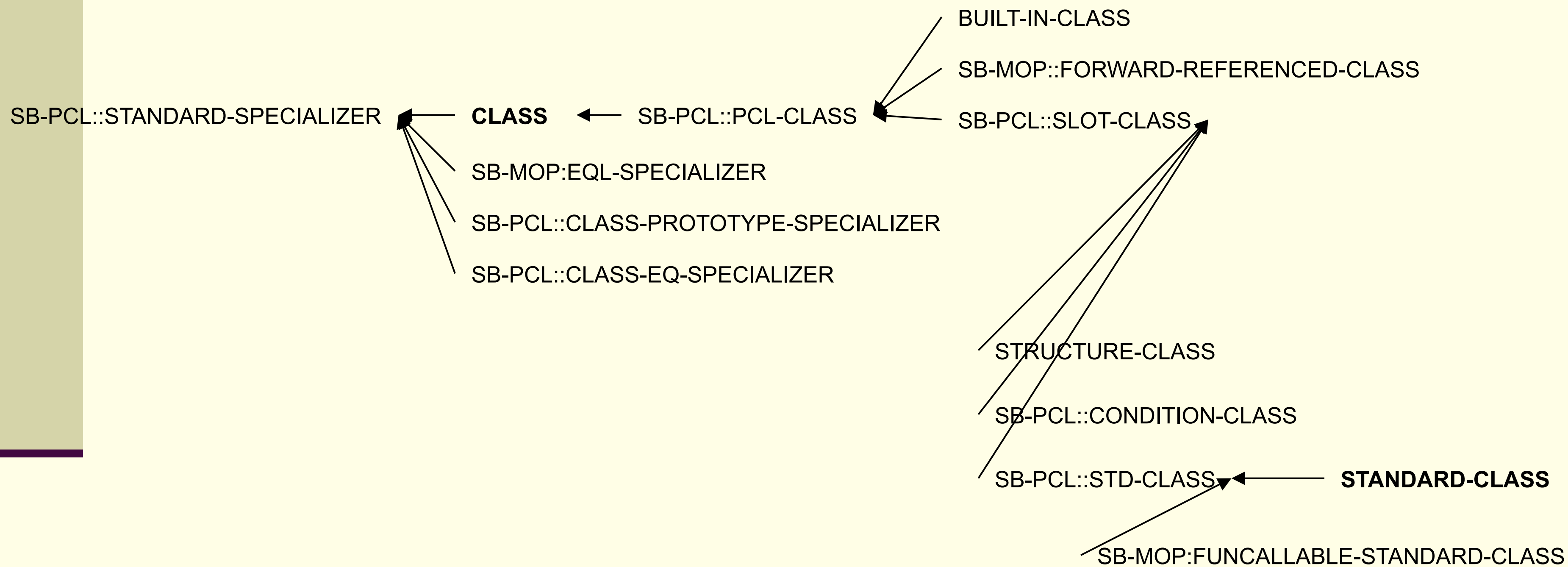
Conclusion

- Explained meta-circularity and meta-modeling in CLOS in terms of denotational and extensional semantics, which are obtained from the W3C document of RDF Semantics Recommendation.
- The membership loop of the cl:standard-class and the twist relation between cl:standard-class and cl:standard-object are similar to those of the rdfs:Class and rdfs:Resource.
- Addressed the CLOS clean meta-modeling for ontology construction.

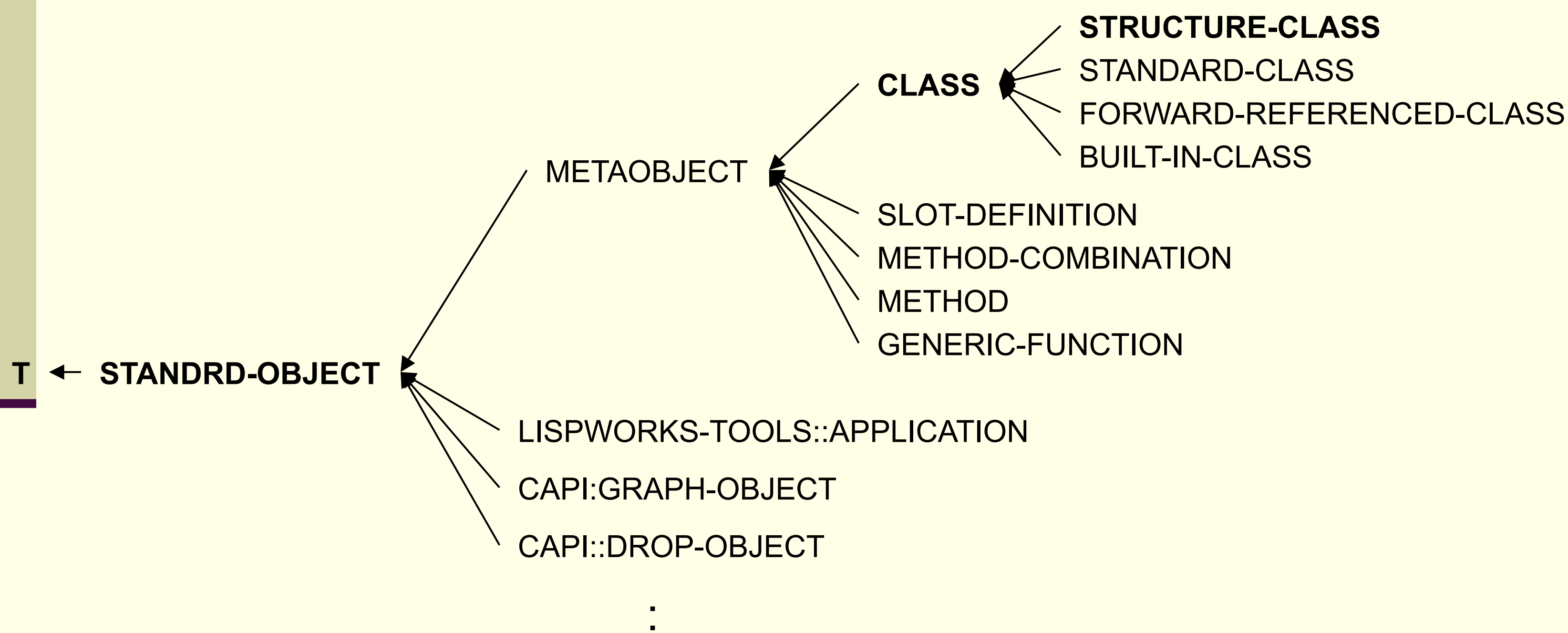
Steel-Bank-Common-Lisp



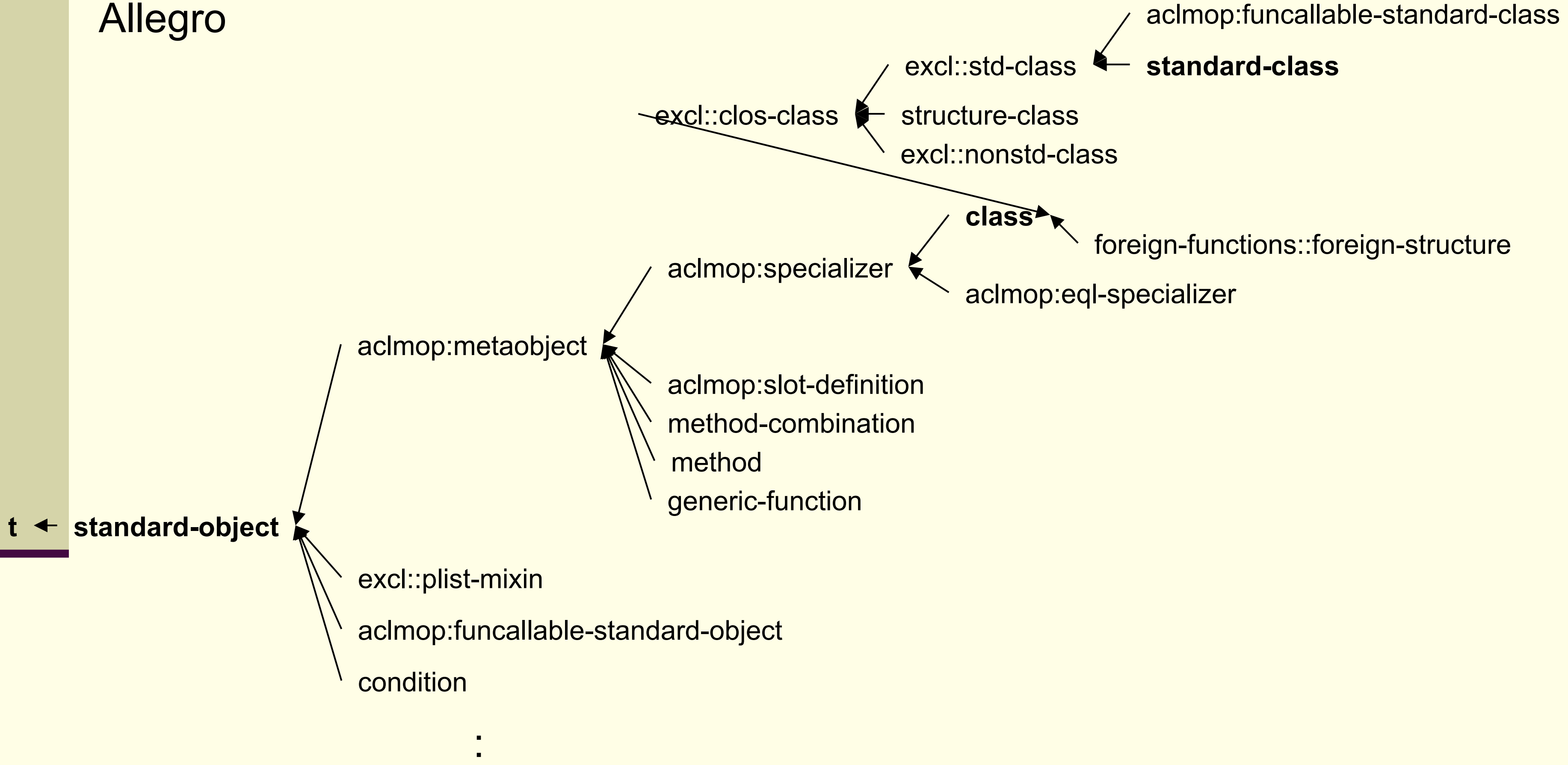
Steel-Bank-Common-Lisp



LispWorks


T
STANDRD-OBJECT
METAOBJECT
CLASS
STRUCTURE-CLASS
STANDARD-CLASS
FORWARD-REFERENCED-CLASS
BUILT-IN-CLASS
SLOT-DEFINITION
METHOD-COMBINATION
METHOD
GENERIC-FUNCTION
LISPWORKS-TOOLS::APPLICATION
CAPI:GRAPH-OBJECT
CAPI::DROP-OBJECT
:

Allegro



aclmop:funcallable-standard-class
excl::std-class
standard-class
excl::clos-class
structure-class
excl::nonstd-class
class
foreign-functions::foreign-structure
aclmop:specializer
aclmop:eql-specializer
aclmop:metaobject
aclmop:slot-definition
method-combination
method
generic-function
t
standard-object
excl::plist-mixin
aclmop:funcallable-standard-object
condition
: