

Peter Winstanley

Semantic Arts UK Ltd



IS AN UPPER-LEVEL ONTOLOGY USEFUL?

ENDORSE.

THE EUROPEAN DATA CONFERENCE ON REFERENCE DATA AND SEMANTICS

A tale of two clothes stores - the wardrobe vs the drawer



What is in the drawer?

Perhaps t-shirts – not sure.

Can't tell how many

There is a blue one with leaf patterns, a black one with some wording, something green...

Basically, without lifting things out we can't really tell.



What's in the wardrobe?



- At least 4 pairs of trousers (1 x grey, 2 x fawn, 1 x blue)
- At least 10 shirts (2 x white, 2 x pink,....etc)
- Perhaps 4 suits or sports jackets
- One suit bag – no idea of the contents
- Two outer jackets
- Other unknown jacket-looking clothes

What is the special technology that provides insight here?

The hanger



Cheap, simple, easy
to use.

Which is easier to use in a systematic way?

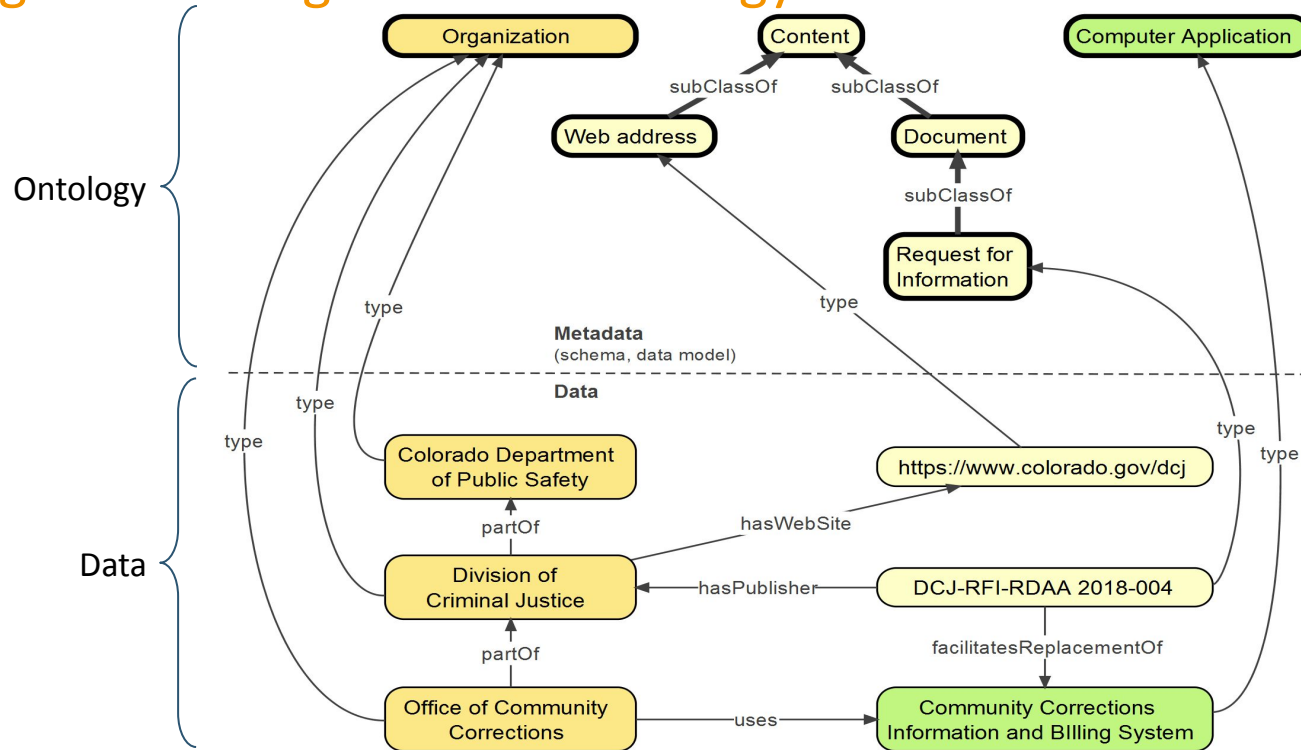
Easy!






Challenging!

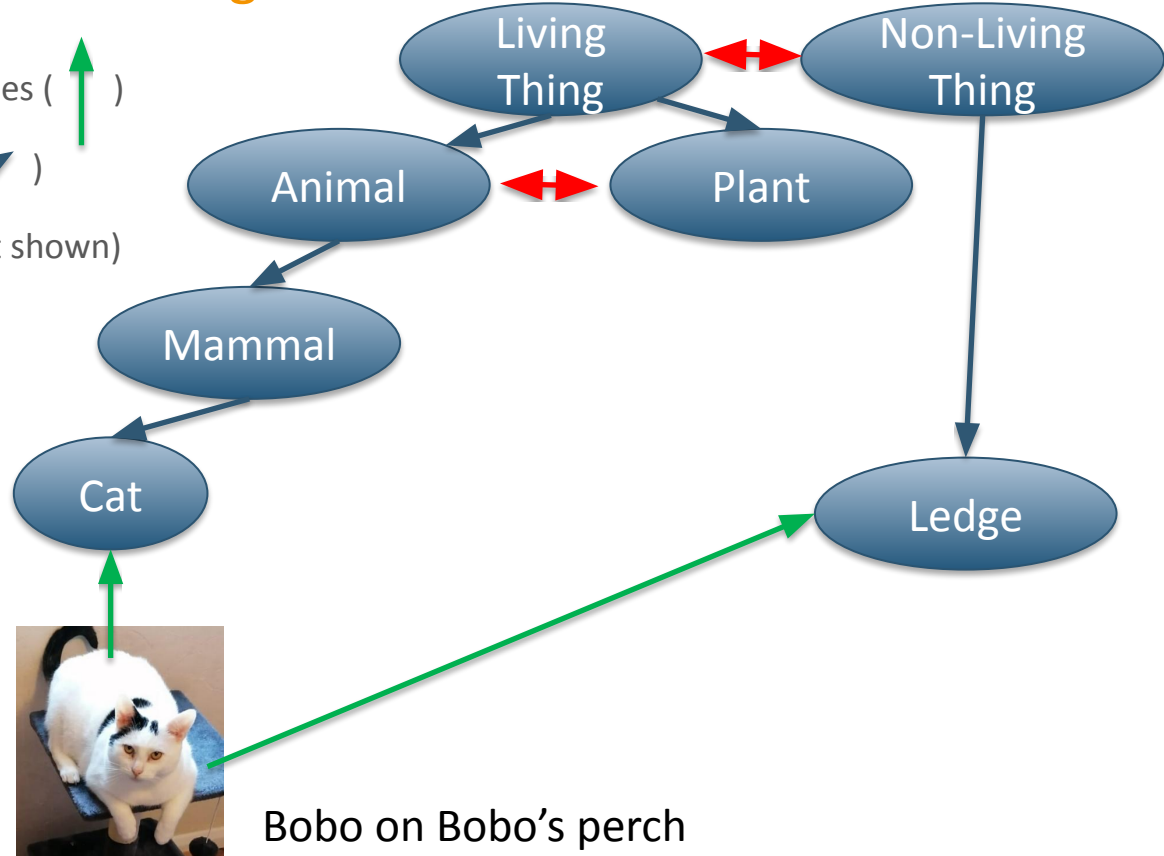


Data gets meaning from the ontology.



How does an ontology give meaning to the data?

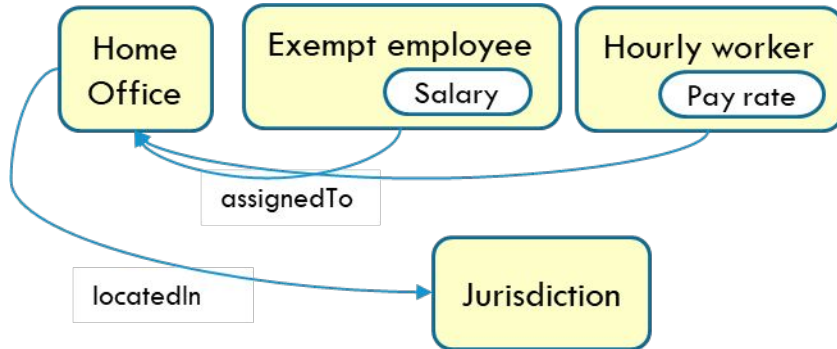
- via `rdf:type` (is-a) relationships to Classes ()
- via Class – Subclass relationships ()
- via Domain and Range constraints (not shown)
- via Disjoint statements ()



Modularity & Reuse Example

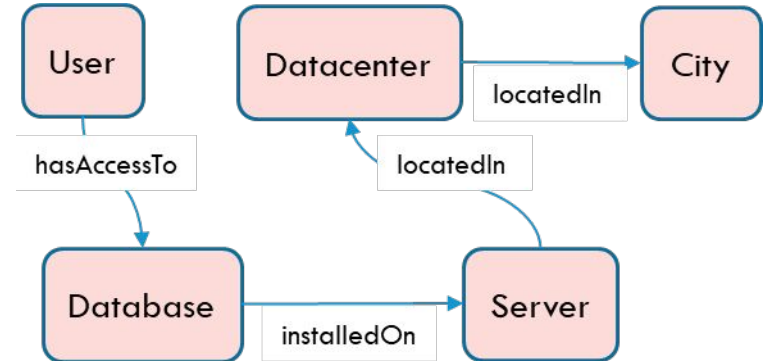
HR Department

A typical set of enterprise systems...



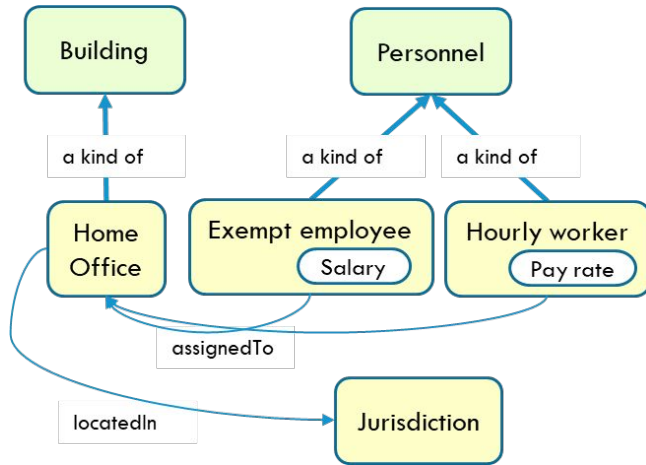
IT Department

...converted into departmental ontologies

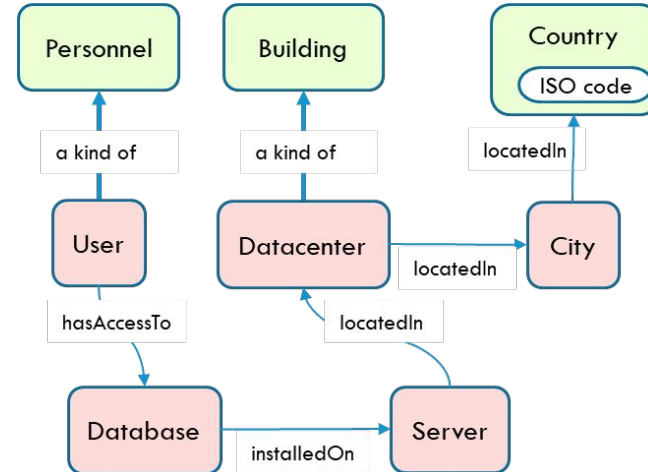


Common Meaning Extracted

HR Department

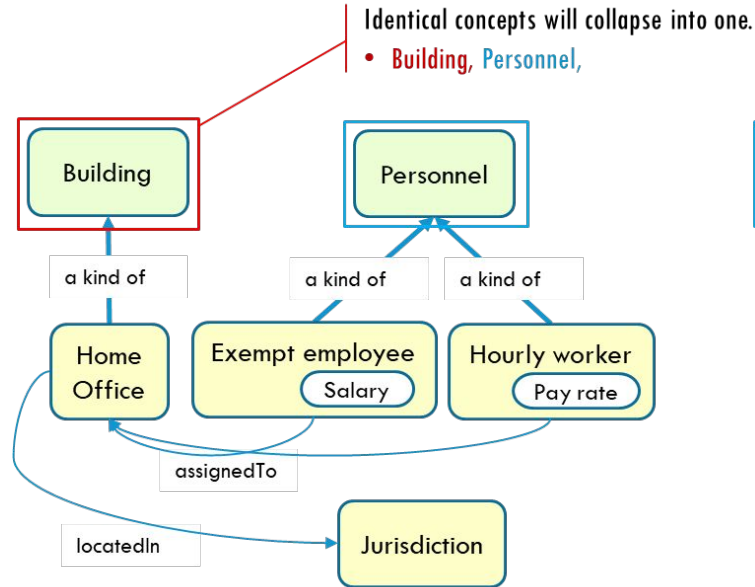


IT Department

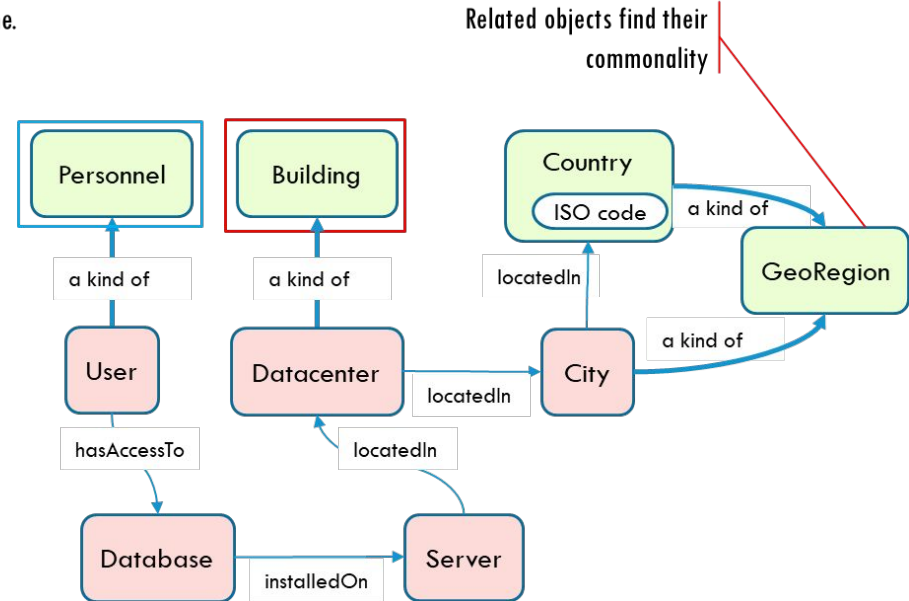


Linking Commonality with a General Ontology

HR Department

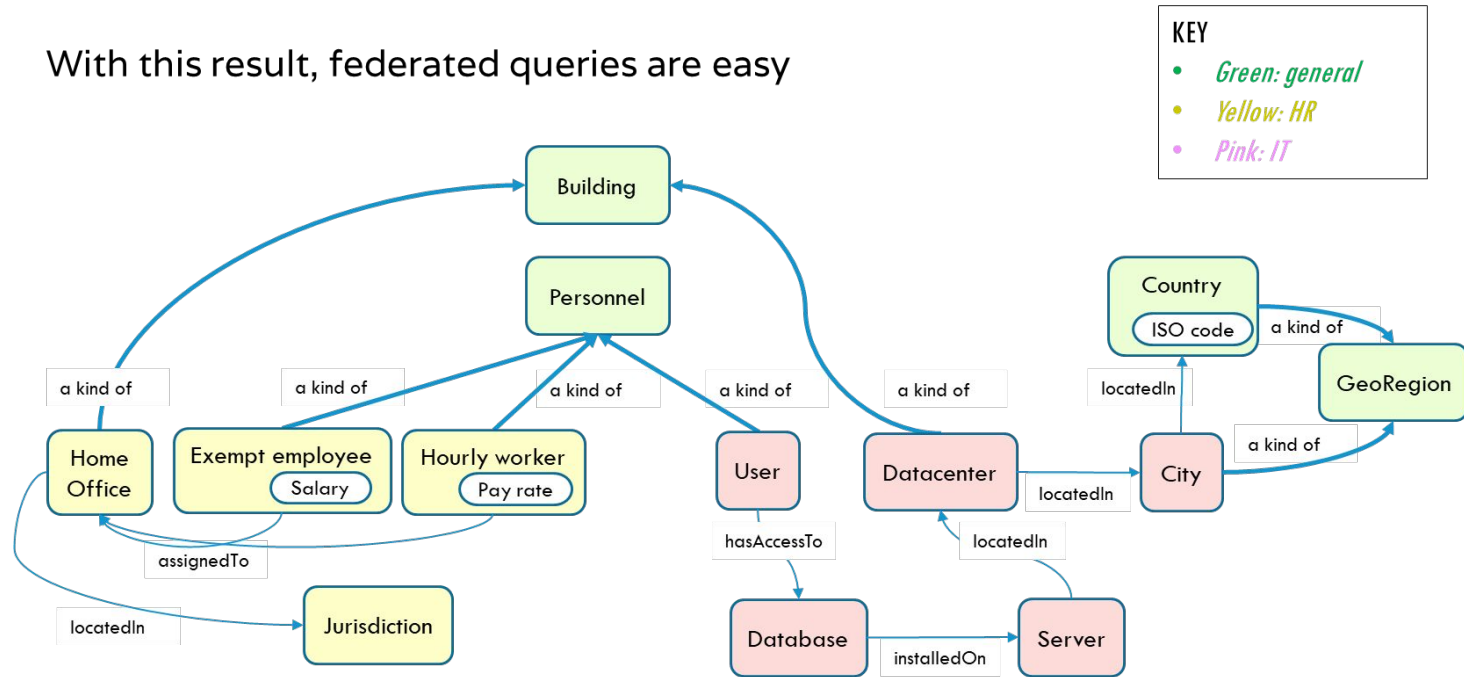


IT Department



Data Snaps Together

With this result, federated queries are easy



Modularity & Reuse—Review

Started with two separate data sets

- Typical, uncoordinated application data repositories

Created a general ontology

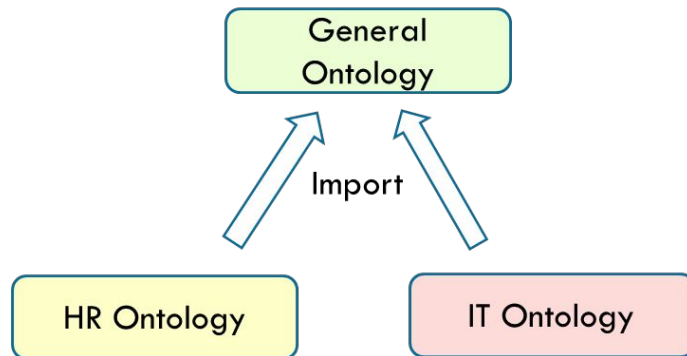
- Extracted shared, reusable concepts from each

Adjusted the HR ontology

- Now specializes concepts from HR perspective

Adjusted the IT ontology

- Now specializes concepts from IT perspective



Don't let anyone tell you it's easy – it's not!

It's a lot of work to retrofit

Agreeing on terminology between teams, or else agreeing to adopt the external upper-level ontology

Agreeing on, or adopting minting patterns for URIs (both ontology & instance)

Updating schema to link to upper

Better to start off with an upper-level ontology

Ontological Commitment

Be careful!

Before you import, be sure that you agree with the whole of the ontology “tree” you’re connecting to, not just part of it.

This means you agree with or can live with all its assertions (i.e., triples) and inferences.

This is called “Committing to an Ontology”

The best approach is not to retrofit, but to start with the upper-level ontology

But there are lots of benefits

The 'hanger' could be any ontology – having an explicit set of semantics is a great benefit

The 'hanger' as an upper ontology makes it easier to group and sort information types across multiple sources

The 'hanger' makes it easier to bring in new information into your stores

The 'hanger' helps you to develop modular, reusable architectural components

Data becomes more readily interoperable

Queries can be written so that they can be more easily federated

If upper ontologies are so helpful, which should I use?

Upper and Middle:

- Cyc: Gigantic scale
- SUMO: Large scale
- schema.org (lightweight, growing)

Upper Only: small scale

- DOLCE-Lite
 - Basic Formal Ontology (BFO)
 - gist: business focus
 - IO: Italian Government
- } academic,
philosophical underpinning

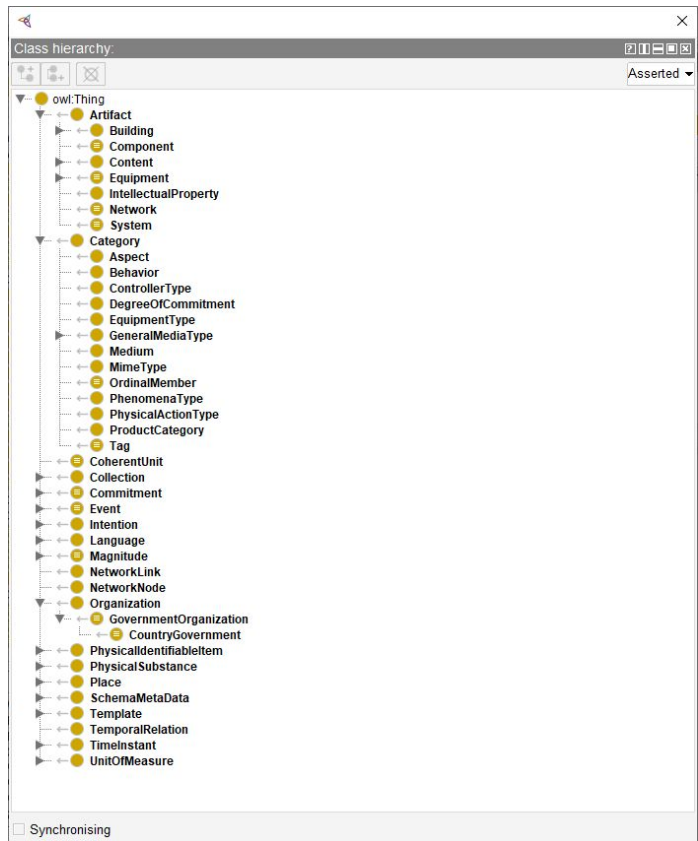
There is some choice

Pick something that is appropriately scaled to your problem domain

I work in enterprises (public and private sector) so **gist** is designed to suit my work.



gist – an upper ontology for business information models



gist

143 Classes

108 Object Properties

1692 Axioms

25 Domain constraints

46 Range constraints

<https://www.semanticarts.com/gist/>

gist is our minimalist upper ontology. It is designed to have the maximum coverage of typical business ontology concepts with the fewest number of primitives and the least amount of ambiguity. Our gist ontology is free (as in free speech and free beer--it is covered under the Creative Commons 3.0 attribution share-alike license). You can use as you see fit for any purpose, just give us attribution.

A screenshot of the Gist ontology metrics table. The table is titled 'Ontology metrics' and lists various metrics and their counts. The metrics are grouped into 'Metrics', 'Class axioms', 'Object property axioms', 'Data property axioms', and 'Individual axioms'. The 'Metrics' section includes counts for Axioms (1692), Logical axiom count (491), Declaration axioms count (290), Class count (143), Object property count (108), Data property count (22), Individual count (17), and Annotation Property count (7). The 'Class axioms' section includes counts for SubClassOf (68), EquivalentClasses (91), DisjointClasses (100), GCI count (0), and Hidden GCI Count (8). The 'Object property axioms' section includes counts for SubObjectPropertyOf (26), EquivalentObjectProperties (0), InverseObjectProperties (20), DisjointObjectProperties (1), FunctionalObjectProperty (3), InverseFunctionalObjectProperty (3), TransitiveObjectProperty (8), SymmetricObjectProperty (0), AsymmetricObjectProperty (0), ReflexiveObjectProperty (0), IrreflexiveObjectProperty (0), ObjectPropertyDomain (25), ObjectPropertyRange (46), and SubPropertyChainOf (0). The 'Data property axioms' section includes counts for SubDataPropertyOf (2), EquivalentDataProperties (0), DisjointDataProperties (0), FunctionalDataProperty (1), DataPropertyDomain (15), and DataPropertyRange (17). The 'Individual axioms' section includes counts for ClassAssertion (31) and ObjectPropertyAssertion (16). The table also has a 'Synchronising' checkbox at the bottom left.

Ontology metrics	
Metrics	
Axiom	1692
Logical axiom count	491
Declaration axioms count	290
Class count	143
Object property count	108
Data property count	22
Individual count	17
Annotation Property count	7
Class axioms	
SubClassOf	68
EquivalentClasses	91
DisjointClasses	100
GCI count	0
Hidden GCI Count	8
Object property axioms	
SubObjectPropertyOf	26
EquivalentObjectProperties	0
InverseObjectProperties	20
DisjointObjectProperties	1
FunctionalObjectProperty	3
InverseFunctionalObjectProperty	3
TransitiveObjectProperty	8
SymmetricObjectProperty	0
AsymmetricObjectProperty	0
ReflexiveObjectProperty	0
IrreflexiveObjectProperty	0
ObjectPropertyDomain	25
ObjectPropertyRange	46
SubPropertyChainOf	0
Data property axioms	
SubDataPropertyOf	2
EquivalentDataProperties	0
DisjointDataProperties	0
FunctionalDataProperty	1
DataPropertyDomain	15
DataPropertyRange	17
Individual axioms	
ClassAssertion	31
ObjectPropertyAssertion	16

ENDORSE. IS AN UPPER-LEVEL ONTOLOGY USEFUL?

A Selection of gist Top Classes

Artifact

An intentional, person-made thing, which could be physical or content

Category

A concept or label used to categorize other instances informally. Things that can be thought of as types are usually Categories.

Collection

Any identifiable grouping of instances. For instance, a jury is a collection of people.

Organization

A generic organization that can be formal or informal, legal or non-legal. It can have members, or not.

Commitment

An obligation (possibly unilateral).

Place

Union of all the geo classes

Event

Something happening over some period of time, often characterized as some kind of activity being carried out by some person, organization, or software application.

Physical Identifiable Item

You could, at least in principle, put an RFID tag on members of this class. Physical things are made of something. E.g., statues are made of bronze.

Intention

Goal, desire, aspiration. This is the "teleologic" aspect of the system that indicates things are done with a purpose.

So, Is an upper-level ontology useful?

You stand on the shoulders of giants – much heavy lifting has already been done.

Many ontologies and derived application profiles are built with the primary purpose being to directly structure and add meaning to specific kinds of data.

That's fine if they are just going to be standalone items, but most times they are not because we want to integrate data from multiple sources and schemas.

So, for portfolios of systems, an ontology whose primary purpose is to organize and unify other ontologies is very useful.

So “**yes!**”, an UPPER-LEVEL ONTOLOGY is very useful.

Practice Recommendation:

Start your ontology development by extending from an upper-level ontology.

Which one? Any ... just use one.

ENDORSE.

THE EUROPEAN DATA CONFERENCE ON REFERENCE DATA AND SEMANTICS