Dynamic Web Development

Lecture 6

The Semantic Web 1

http://www.linkeddatatools.com/semantic-web-basics

The Semantic Web

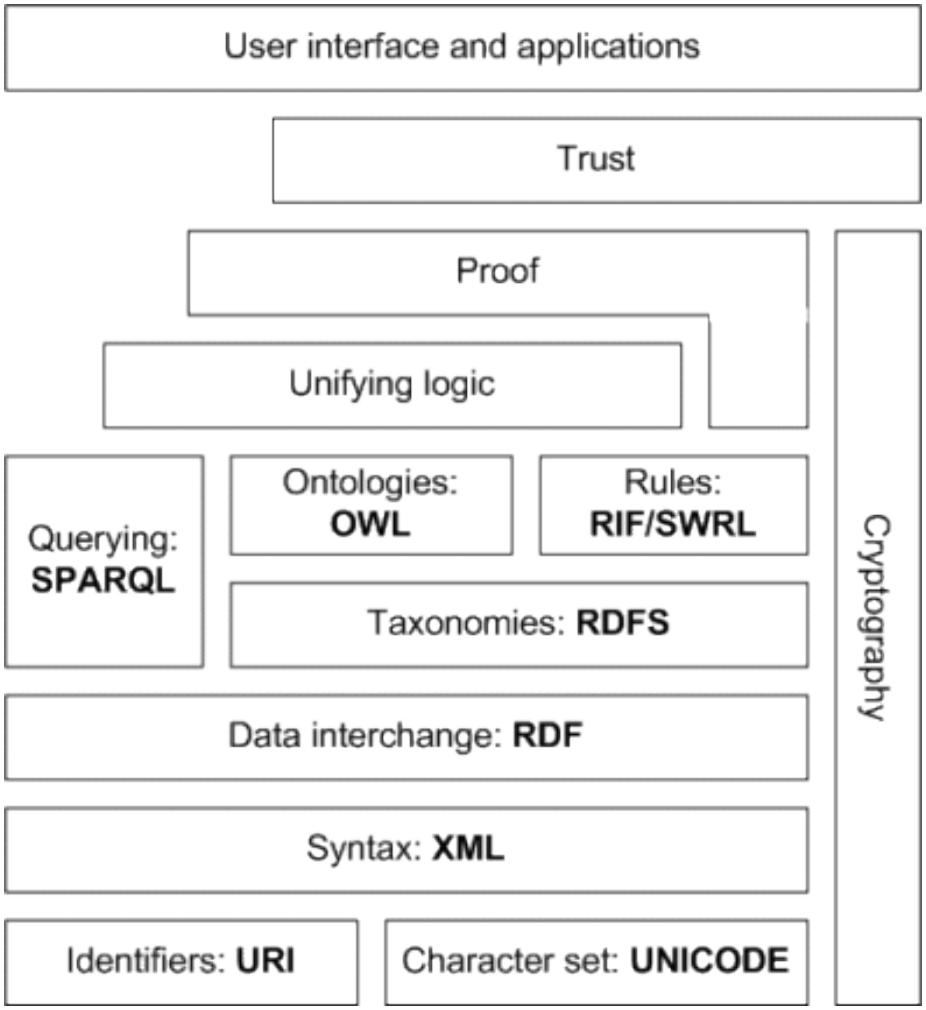
"In addition to the classic “Web of documents” W3C is helping to build a technology stack to support a “Web of data,” the sort of data you find in databases. "

"The ultimate goal of the Web of data is to enable computers to do more useful work and to develop systems that can support trusted interactions over the network. The term “Semantic Web” refers to W3C’s vision of the Web of linked data. "

"Semantic Web technologies enable people to create data stores on the Web, build vocabularies, and write rules for handling data. Linked data are empowered by technologies such as RDF, SPARQL, OWL, and SKOS."

www.w3.org/standards/semanticweb

Semantic Web Diagram



RDF
Resource Description Framework

RDFS
RDF Schema

OWL
Web Ontology Language

RIF
Rule Interchange Format

SPARQL
An RDF query language

http://www.obitko.com/tutorials/ontologies-semantic-web/semantic-web-architecture.html

This Lecture

- Graph Data
- Introducing RDF
- Semantic Modelling

Next Lecture

- RDFS and OWL
- Querying Semantic Data

A Collection of Data Items

One way of storing them is in a relational database.

But relational databases were invented before the age of networks – they have been extended to create ‘distributed databases’.

ID	Title	Author	Medium	Year
1	As You Like It	Shakespeare	Play	1599
2	Hamlet	Shakespeare	Play	1604
3	Othello	Shakespeare	Play	1603
4	Sonnet 78	Shakespeare	Poem	1609
5	Astrophil and Stella	Sir Phillip Sidney	Poem	1590
6	Edward II	Christopher Marlowe	Play	1592
7	Hero and Leander	Christopher Marlowe	Poem	1593
8	Greensleeves	Henry VIII Rex	Song	1525

Could be Stored This Way

7	Hero and Leander	Christopher Marlowe	Poem	1593
3	Othello	Shakespeare	Play	1603

1	*As You Like It*	Shakespeare	Play	1599

4	Sonnet 78	Shakespeare	Poem	1609
6	Edward II	Christopher Marlowe	Play	1592

What do they all need to agree on?

Which column is which?
Needs a shared schema (metadata) - and everyone has to agree on it!

Or They Could be Stored This Way

Title
As You Like It
Hamlet
Othello
"Sonnet 78"
Astrophil and Stella
Edward II
Hero and Leander
Greensleeves

What do they all need to agree on this time?

Need a <u>shared</u> way to identify individual records - which things are we talking about?

Year	Medium
1599	Play
1604	Play
1603	Play
1609	Poem
1590	Poem
1592	Play
1593	Poem
1525	Song

Author
Shakespeare
Shakespeare
Shakespeare
Shakespeare
Sir Phillip Sidney
Christopher Marlowe
Christopher Marlowe
Henry VIII Rex

The Most Flexible Way is This

	Author
Row 4	Shakespeare

	Medium
Row 7	Poem

	Title
Row 2	*Hamlet*

	Year
Row 2	1604

	Medium
Row 6	Play

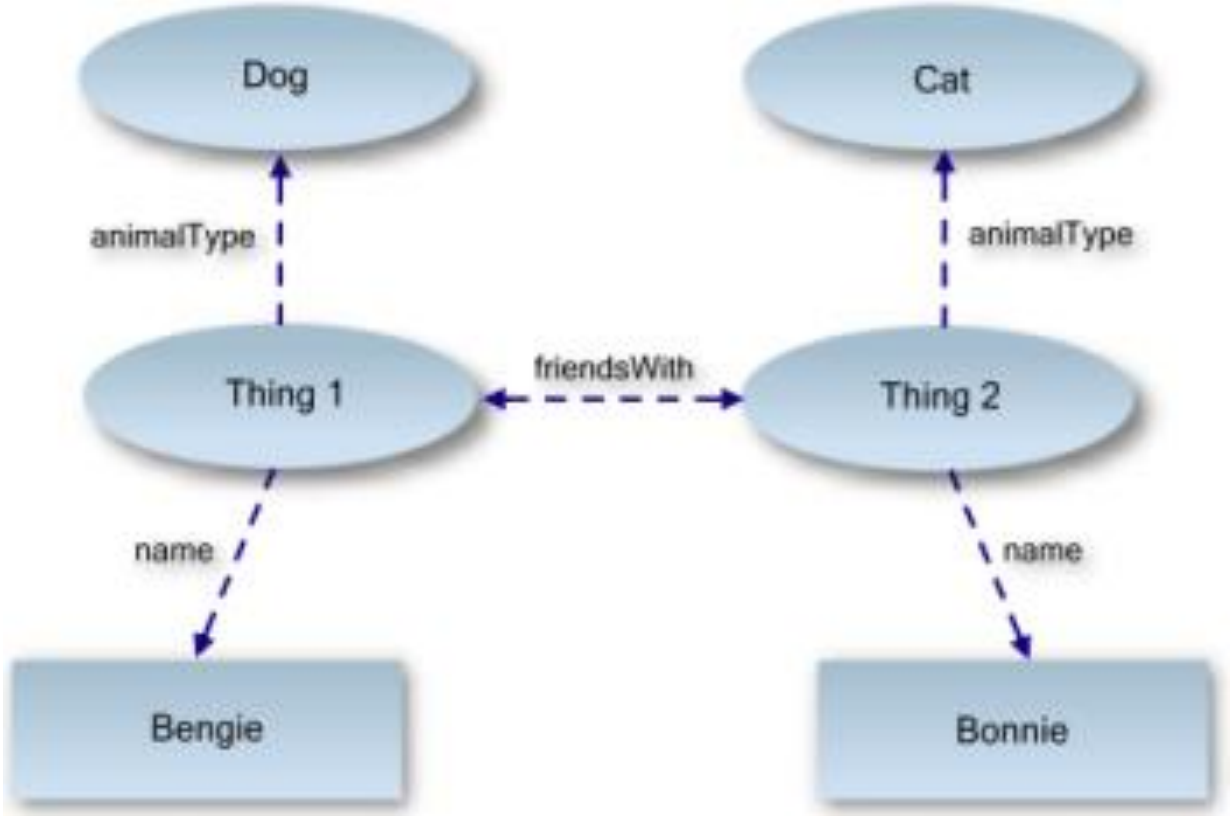
Need to use a <u>shared</u> schema and entity ID system.

This is an example of Graph Data

Subject	Predicate	Object
Item2	hasTitle	Hamlet
Item2	hasAuthor	Shakespeare
Item2	hasMedium	Play
Item2	YearCreated	1604
Item7	hasTitle	Hero and Leander
Item7	hasAuthor	Christopher Marlowe
Item7	hasMedium	Poem
Item7	YearCreated	1593

Data arranged this way is sometimes referred to as a set of 'triples'.

Another Example of a Graph Data



Other Types of Database

There are databases that are designed to store graph data.

See: http://en.wikipedia.org/wiki/Graph_database

There are also more specialised ones known as **Triplestores** designed to hold Subject-Predicate-Object triples.

Bigdata, IBM DB2, Redland, Sesame

See: http://en.wikipedia.org/wiki/Triplestore

What we need is an XML based language (good for transmitting data over a network) which is designed to encode data triples.

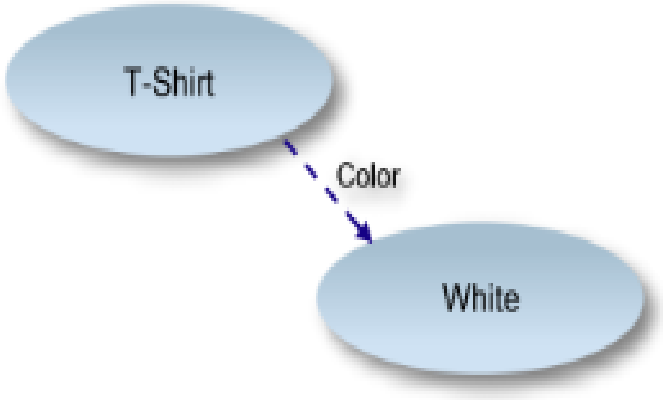
This Lecture

- Graph Data
- Introducing RDF
- Semantic Modelling

Next Lecture

- RDFS and OWL
- Querying Semantic Data

RDF - Resource Description Framework



An RDF statement has three parts:

Subject	T-shirt
Predicate (property)	Color
Object	white

What is supplying the shared semantic meaning of these terms at the moment?

How is it done when it is just computers, doing the sharing, not people?

Partly by using RDFS (RDF Schema) and OWL (Web Ontology Language).

Written as an RDF statement

```
<?xml version="1.0" encoding="UTF-8"?>

<rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
          xmlns:feature="http://www.useful-metadata.com/clothing-features#">

  <rdf:Description rdf:about="http://en.wikipedia.org/wiki/T-shirt">
      <feature:size>12</feature:size>
  </rdf:Description>
</rdf:RDF>
```

Note the use of a URL to refer to the subject.

This gives a way of referring to entity definitions that can be shared.

Note that the predicate (size) is not part of RDF

It belongs to the namespace **feature**

The RDF Document Root Tag

```
<rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

<!-- Body Code Omitted -->

</rdf:RDF>
```

Note the standard W3.org namespace
http://www.w3.org/1999/02/22-rdf-syntax-ns#.

This namespace tells any machine reader that the enclosing document is an RDF document, and that the rdf:RDF tag resides in this namespace.

This namespace, and the RDF node, forms the root of all RDF documents.

Add A Statement

An RDF document can contain more than one statement. For simplicity, we'll only add one. Start by adding a an rdf:Description tag, which in RDF/XML can contain one or more statements about the same subject:

```
<rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

  <rdf:Description rdf:about="http://en.wikipedia.org/wiki/T-shirt">

    <!-- Statement Code Omitted -->

  </rdf:Description>

</rdf:RDF>
```

The rdf:Description tag simply means "I'm going to describe something (a *subject*) and I'm giving it the unique ID **"http://en.wikipedia.org/wiki/T-shirt"**.

Add Predicates

RDF statements describe the characteristics of their subjects using properties, or *predicates* in RDF terminology.

For simplicity, let's start by adding one predicate: the size of our T-shirt.

```
<rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
          xmlns:feature="http://www.useful-metadata.com/clothing-features#">

  <rdf:Description rdf:about="http://en.wikipedia.org/wiki/T-shirt">
     <feature:size>12</feature:size>
  </rdf:Description>

</rdf:RDF>
```

This simply says "The subject has a property with name **feature:size** which has the literal value 12".

Finally, let's add one more predicate: the color of the T-shirt.

```
<rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
          xmlns:feature="http://www.useful-metadata.com/clothing-features#">

  <rdf:Description rdf:about="http://en.wikipedia.org/wiki/T-shirt">
     <feature:size>12</feature:size>
     <feature:color rdf:resource="http://en.wikipedia.org/wiki/White"/>
  </rdf:Description>

</rdf:RDF>
```

You'll notice this one isn't quite the same as the last one. Whereas the last one had the *literal value* 12, this one is referring to the subject (ID) of another statement.

This allows to make use of a shared definition of the term 'white'.

I don't have to hope that someone uses exactly the same semantic definition as I do.

Breaking Down The Statement

Now we've looked at a simple example of an RDF document, let's formalize what we've learned and break the statement into its component parts:

```
<rdf:Description rdf:about="subject">
   <namespace:predicate rdf:resource="object" />
   <namespace:predicate>literal value</namespace:predicate>
<rdf:Description>
```

Here you see the *subject* of the statement (what the statement is about), and the two forms of predicates (*literal values* and *resources*, which reference other RDF statements).

Note even the attributes have namespaces.

A More Thorough Example

```
<?xml version="1.0" encoding="UTF-8"?>

<rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
          xmlns:dc="http://purl.org/dc/elements/1.1/"
          xmlns:region="http://www.country-regions.fake/">

  <rdf:Description rdf:about="http://en.wikipedia.org/wiki/Oxford">
    <dc:title>Oxford</dc:title>
    <dc:coverage>Oxfordshire</dc:coverage>
    <dc:publisher>Wikipedia</dc:publisher>
    <region:population>10000</region:population>
    <region:principaltown rdf:resource="http://www.country-regions.fake/oxford"/>
  </rdf:Description>

</rdf:RDF>
```

See if you can identify on the RDF document above the:

Subject of the statement
Predicates of the statement - including whether they are *resources* or *literals*
Objects referenced by the resource predicates

A Quick Recap Of URIs And XML Namespaces

Note the use of the predicates with the prefix dc on the previous example.

That prefix belongs to the namespace:

http://purl.org/dc/elements/1.1/

which means that the predicates are defined elsewhere. They are not part of RDF.

RDF is a framework which allows you to make use of sets of predicates that have been defined elsewhere.

The Dublin Core

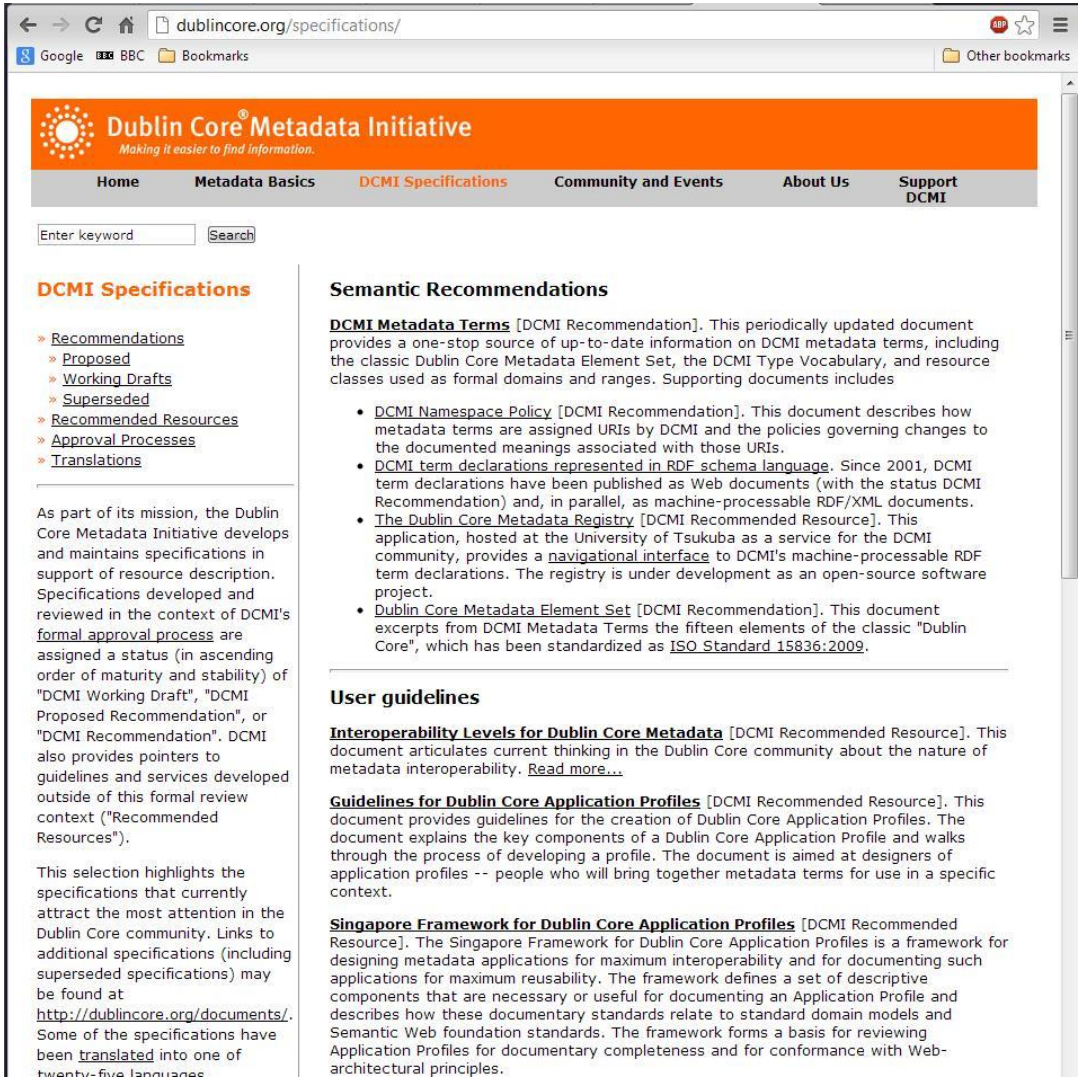
RDF is metadata (data about data). RDF is used to describe information resources.

The Dublin Core is a set of predefined predicates for describing documents.

The first Dublin Core properties were defined at the **Metadata Workshop in Dublin, Ohio** in 1995 and is currently maintained by the Dublin Core Metadata Initiative.

Property	Definition
Contributor	An entity responsible for making contributions to the content of the resource
Coverage	The extent or scope of the content of the resource
Creator	An entity primarily responsible for making the content of the resource
Format	The physical or digital manifestation of the resource
Date	A date of an event in the lifecycle of the resource
Description	An account of the content of the resource
Identifier	An unambiguous reference to the resource within a given context
Language	A language of the intellectual content of the resource
Publisher	An entity responsible for making the resource available
Relation	A reference to a related resource
Rights	Information about rights held in and over the resource
Source	A Reference to a resource from which the present resource is derived
Subject	A topic of the content of the resource
Title	A name given to the resource
Type	The nature or genre of the content of the resource

Dublin Core Metadata



Metadata Initiatives

Standard vocabularies, or formal ontologies representing terms within a domain of knowledge, are already available freely from various organisations dedicated to creating standard vocabularies for a range of subjects - for example media terms, or biomedical terms, or scientific terms. Below are some examples:

Dublin Core Metadata Initiative (DCMI)

- Creates ontologies for a range of subjects, particularly focusing on common, every day terms and terms important in media and document storage.

Friend Of A Friend (FOAF)

- focuses on developing a standard vocabulary/ontology for social networking purposes.

Semantically-Interlinked Online Communities (SIOC)

- For linking blogs, forums, mailing lists etc.

Simple Knowledge Organisation System (SKOS)

- For representing thesauri, classification schemes, taxonomies and subject heading systems

Description of a Project (DOAP)

- Vocabulary for describing software projects.

OpenCyc / UMBEL

- An ontology of everyday, common sense terms.

"**The entire Cyc ontology** containing **hundreds of thousands of terms**, along with **millions of assertions** relating the terms to each other, forming an upper ontology whose domain is all of human consensus reality."

www.opencyc.org

This Lecture

- Graph Data
- Introducing RDF
- Semantic Modelling

Next Lecture

- RDFS and OWL
- Querying Semantic Data

Whilst RDF offers a flexible, graph-based model for recording data that is interchangable globally, it doesn't offer any means to record *semantics* or *meaning*.

Why Include Semantics In Data? Knowledge Integration

There's no point in adding semantics to your data if it does not provide significant benefits. One of the primary benefits of adding semantic meaning to your data is that it can be branched across **domains of knowledge automatically.**

In our example, two websites are started independently from each other. One site hosts information on current and historic Oscar winning films; the other a large database of biographies of Hollywood actors and actresses.

Both contain complementary information in their website databases. We will cover firstly how information sharing between these sites could happen without the use of semantics. Then, we will describe how the same information can be shared between the two sites - and potentially beyond - with the use of semantics.

Sharing Without Semantic Modeling

FilmID	Name	Year	Actor1	Actor2
F001	Dirty Harry	1970	Clint Eastwood	Andy Robinson
F002	Goldfinger	1964	Sean Connery	Honor Blackman
F003	Blade Runner	1984	Harrison Ford	Rutger Hauer

ActorID	Name	Year	Town
A001	Clint Eastwood	1930	San Francisco
A002	Sean Connery	1930	Edinburgh
A003	Harrison Ford	1942	Chicago

What do we need to agree on, if we want to share data?

Problems with sharing data

- Tables designed independently.
- Different Primary keys and Foreign Keys.
- Different database server systems.
- **Things with the same name might not have the same meaning**

Would need to create a common schema - a common way of describing film and actor data – and convert data into that schema every time it is moved from one database to the other.

Costs time and money.

It also requires humans to understand – and agree – on the meaning of the data so that they can agree on these common schemas.

Sharing With The Semantic Web Model

Vocabulary

A collection of terms given a well-defined meaning that is consistent across contexts.

i.e A standard set of predicates.

Ontology

Allows you to define the *contextual relationships* behind a defined vocabulary.

It is the cornerstone of defining a knowledge domain.

A formal syntax for defining ontologies is:

OWL (Web Ontology Language) which is an extension to:

RDFS (RDF Schema).

Sharing With Semantic Modeling

Shared Ontology
Which shows relationships between predicates

A film has a director
A film has several actors
Each actor has a birth year and town

Shared Set of Predicates
Film_ID, Film_Name, Actor_Name, Birth_Year etc

FilmID	Film_Name	Release_ Year	Actor1	Actor2
F001	Dirty Harry	1970	Clint Eastwood	Andy Robinson
F002	Goldfinger	1964	Sean Connery	Honor Blackman
F003	Blade Runner	1984	Harrison Ford	Rutger Hauer

ActorID	Actor_Name	Birth_ Year	Town
A001	Clint Eastwood	1930	San Francisco
A002	Sean Connery	1930	Edinburgh
A003	Harrison Ford	1942	Chicago

What do we need to agree on, if we want to share data?

If both sites are using a shared set of predicates, and a shared ontology for defining contextual relationships, the two sites can now query each other using the same terms automatically - without human interaction.

The Oscar Winning Movies site can now query the actor names on the Actor Biographies site and gain more detail about a specific actor or actress that has starred in a movie.

The Actor Biographies site can now query the film plots on the Oscar Winning Movies site and gain more detail about films an actor has starred in.

With the contextual relationships defined in a formal web ontology, further related information about the actors or films, e.g. film locations, or films made by the same director, may be found via the linked standard terminology without the user even imagining that information initially existed.