Department of Engineering Ferrara

Introduction to Semantic Web and Description Logics

Riccardo Zese

Outline

Outline

Our world

Representing information is useful in many event

- In every domain, objects naturally fall into categories and often into multiple categories.
- Different categories can be related to each other. For example, some categories can be more general (specific) than others.
- Objects can be composed or contain parts, that are other objects.

Representing relationships among objects and among categories is essential

Our world

We would like to represent:

- Objects AKA **individuals**: the objects of the domain
  - andrea, rex, ai_course
- Categories AKA **concepts**: describe basic category classes
  - Football_player, Dog, University_course
- Relations AKA **roles**: describe objects that are parts or attributes or properties of other objects
  - Plays_for, Age, Holds_by

Outline

Description Logics

- We would like to define some generalization relation
- We would like to represent complex concepts as the results of some **composition** of simpler concepts
- We would like to know if an individual **belongs** to some categories or not

We need a language that allows us to represent all these information but also to automatically **infer** generalization hierarchies as a consequence of the description we have made of concepts

Description Logics

Description Logics are a family of logics. Each logic is different depending on which **operators** are admitted in the logic.
Of course, more operators means:

- higher expressivity
- higher computational costs
- the logic might to be *undecidable*
- ...

Decidability: a theory is *decidable* if there is an effective method for determining whether arbitrary formulas are included in the theory. In other words, a DL is decidable if, given a generic KB, there exists a method or an algorithm that, given a generic subsumption axiom to prove, the algorithm can determine if the axiom holds or not.

Description Logics

At `http://www.cs.man.ac.uk/~ezolin/dl/` there is a nice application that shows complexity issues and decidability depending on which operator you decide to include/exclude in your logic

Outline

The Simplest Description Logics: $\mathcal{ALC}$

Grammar:

- Logical symbols (with a fixed meaning)
  - Concept-forming operators: `:and`, `:or`, `:some`, `:all`
  - Connectives: $\sqsubseteq$, $\equiv$
- Non-logical symbols (with a meaning domain dependent)
  - Individuals: andrea, rex, ai_course
  - Universal concept: *Thing* ($\top$)
  - Bottom concept: *Bottom* ($\bot$)
  - Atomic concepts: Football_player, Dog, University_course
  - Atomic Roles: Plays_for, Age, Taught_by

Complex Concepts

Complex concept:

- Every atomic concept is a concept
- if C is a concept, then $\neg C$ is a concept
- If $C_1, C_2, ..., C_n$ are concepts, then (`:and` $C_1, C_2, ..., C_n$) and (`:or` $C_1, C_2, ..., C_n$) are concepts
- If R is a role, then (`:some` R C) is a concept
- If R is a role and C is a concept, then (`:all` R C) is a concept

Axioms

An axiom is a proposition that can model an information of the domain There are several different axioms:

- If C_1 and C_2 are concepts, then $C_1 \sqsubseteq C_2$ is an axiom
- If C_1 and C_2 are concepts, then $C_1 \equiv C_2$ is an axiom
- If a is an individual and C is a concept, then $a : C$ is an axiom
- If a and b are individuals and R is a role, then $(a, b) : R$ is an axiom

Concept Forming Operators

A desired feature in a description logic is to *define complex concepts in term of more simpler concepts*. The *concept forming operators* allow the definition of complex concepts.

Until now we have seen only four operators but there are many other that we will see further.

:and

`:and` constructions represent conjoined concepts. Each individual is a member of all the concepts specified by the complex concept

- (`:and` Adult Male Person) would represent the concept of something that is at the same time an adult, a male, and a person.
- It could be seen as an *intersection* of the specified concepts

:or

`:or` constructions represent disjoined concepts. Each individual is a member of at least one of the concepts specified by the complex concept

- (`:or` Car Truck Van) would represent the concept of something that can be either a Car, a Truck, a Van or a combination of them
- It could be seen as a *union* of the specified concepts

:some

`:some` garantees that for each individual that belongs to this concept there will be at least one individual related by the role *R* that belongs to *C*

- (`:some` Child Thing) represents the concept of parent (that have at least one child)
- (`:some` HasPet Cat) represents the concept of owner of cats (that have at least one cat)

:all

`:all` garantees that for each individual that belongs to this concept are related by the role *R* **only** to individuals that belong to concept *C*

- (`:all` Child Doctor) represents the concept of someone all of whose children are doctors
- (`:all` Child Male) represents the concept of something that have **zero or more** children which are all male

Complement of Complex Concept

- $\neg(C \sqcap D) \Leftrightarrow \neg C \sqcup \neg D$
- $\neg(C \sqcup D) \Leftrightarrow \neg C \sqcap \neg D$
- $\neg(\forall R.C) \Leftrightarrow \exists R.\neg C$
- $\neg(\exists R.C) \Leftrightarrow \forall R.\neg C$

Axioms

$$C_1 \sqsubseteq C_2$$
C_1, C_2 concepts

Concept C_1 is subsumed by concept C_2, i.e. every individual that satisfies C_1 satisfies also C_2. In other words, every individual that belongs to C_1 belongs also to C_2

Warning: the vice-versa is not true. If an individual belongs to C_2, I cannot say that it belongs also to C_1

Example: $Cat \sqsubseteq Pet$

- Every cat is also a pet

Axioms

$$C_1 \equiv C_2$$
C_1, C_2 concepts

Concept C_1 is equivalent to concept C_2, i.e. the individuals that satisfy C_1 are precisely those that satisfy C_2. In other words, every individual that belongs to C_1 belongs also to C_2 and every individual of C_2 belongs also to C_1

Example: *Parent* $\equiv$ (`:and` *Person* (`:some` *Child*))

- A parent is a person that has at least one child

Axioms

$$a : C$$

a individual, C concept

The individual a satisfies the description expressed by the concept C, the individual belongs to C

Example: $Cat(tom)$

- The individual tom is a cat

Axioms

$$(a, b) : R$$

a, b individuals, R role

The individual a is linked to the individual b by the role R

Example: $Friends(tom, jerry)$

- The individual tom is a friend of the individual jerry

Axioms

Translation of each axiom into predicate logic.

Axiom	Translation
$C \sqsubseteq D$	$\forall x.\pi_x(C) \rightarrow \pi_x(D)$
$C \equiv D$	$\forall x.\pi_x(C) \leftrightarrow \pi_x(D)$
$a : C$	$C(a)$
$(a, b) : R$	$R(a, b)$

From now role and concept assertions will be written using their logic translation

Knowledge Base

A Knowledge Base (KB) is a collection of axioms concerning:

- individuals stand for *objects* in some application domain
- concepts stand for *classes or categories* of individuals
- roles stand for *binary relations* over those individuals

Knowledge Base

Formally, a Knowledge Base is a pair $(\mathcal{A}, \mathcal{T})$ where:

- $\mathcal{A}$ is called **ABox**, Assertional Box: a list of facts about individuals
- $\mathcal{T}$ is called **TBox**, Terminological Box: a list of axioms that describes the concepts

Reasoning On a KB

Reasoning on a KB means find implicit information starting from the explicit information contained in the knowledge base. The most important reasoning problems, in description logics, are the following.

- **Satisfiability**: a concept C is **satisfiable** w.r.t. a KB if it is possible for C to have individuals which belong to it
- **Subsumption**: a concept C_1 is **subsumed** by a concept C_2 w.r.t. the KB if all the individuals that belong to C_1 also belong to C_2
- **Equivalence**: a concept C_1 is **equivalent** to a concept C_2 w.r.t. the KB if all the individuals that belong to C_1 also belong to C_2 and vice-versa
- **Disjointnes**: a concept C_1 is **disjoint** with a concept C_2 w.r.t. the KB if NO individual belonging to C_1 also belongs to C_2

Reduction to Subsumption

Satisfiability, Equivalence and Disjointness can be computed by reducing them to the subsumption. The following proposition hold:

Reduction to Subsumption

For concepts C, C_1 and C_2:

- C is **unsatisfiable** iff C is *subsumed* by $\bot$
- C_1 and C_2 are **equivalent** iff C_1 is *subsumed* by C_2 and C_2 is *subsumed* by C_1
- C_1 and C_2 are **disjoint** iff $C_1 \cap C_2$ is *subsumed* by $\bot$

Reasoning With The ABox

Until now we have talked about reasoning on terminological information. But we would like to answer also to questions like: *does an individual a satisfies concept C?*
Also this kind of questions can be reduced to **subsumption**

Reduction to Satisfiability

A second approach exploits **reduction to satisfiability**

- Concept satisfiability
  KB $\not\models C \equiv \bot \Leftrightarrow$ exists x s.t. KB $\cup \{C(x)\}$ has model
- Subsumption
  KB $\models C \sqsubseteq D \Leftrightarrow$ KB $\cup \{(C \sqcap \neg D)(x)\}$ has no models
- Instance checking
  KB $\models C(a) \Leftrightarrow$ KB $\cup \{\neg C(a)\}$ has no models

Reasoning w.r.t. a Knowledge Base

Summarizing, whatever questions we would like to answer, we can resolve a subsumption problem.
There are two main families of algorithms for computing subsumption:

- based on structural comparisons between concept expressions. The key idea is that if the two concept expressions to be compared are made of subexpressions, one can compare separately one subexpression of a concept with all those of the others
- tableaux-based algorithms (which often exploit reduction to satisfiability)

Structural Comparison - Example

- *Adult* $\sqsubseteq$ (`:or` *Adult Male*)
- (`:and` *Adult Male Rich*) $\sqsubseteq$ (`:and` *Adult Male*)
- (`:some` *Child*) $\sqsubseteq$ (`:or` *Adult Male* (`:some` *Child*))
- (`:and` *Adult Male* (`:all` *Child Male*)) $\sqsubseteq$ (`:all` *Child Male*)
- (`:all` *Child* (`:and` *Adult Male*)) $\sqsubseteq$ (`:or` *Adult Male* (`:all` *Child Male*))

Structural Comparison - Example

- (`:and` *Adult Female*) $\not\sqsubseteq$ (`:and` *Adult Male*)
- (`:and` *Adult Male*) $\not\sqsubseteq$ (`:and` *Adult Male Rich*)
- (`:some` *Child*) $\not\sqsubseteq$ (`:and` *Adult Male* (`:some` *Parent*))
- (`:and` *Adult* (`:all` *Child Male*)) $\not\sqsubseteq$
  (`:and` *Adult Male* (`:all` *Child Female*))
- (`:all` *Child Male*) $\not\sqsubseteq$ (`:and` *Adult* (`:all` *Child* (`:and` *Adult Female*)))

Outline

Extending $\mathcal{ALC}$

What happens if we want, for example, to model the fact that a biped has exactly two legs?

$$\mathit{Biped} \sqsubseteq \geq\ 2\mathit{hasLeg} \sqcap \leq\ 2\mathit{hasLeg}$$

With the language described so far, we cannot model this concept. This logic is too poor.

We need to introduce at least two new operators:

- minimum cardinality: (`:at-least`), i.e. $\geq$
- maximum cardinality: (`:at-most`), i.e. $\leq$

Extending $\mathcal{ALC}$

It is possible to extend the presented description logic in several directions:

- by adding concept-forming operators, such as negation and disjunction
- by relating roles, and considering also complex roles
- by adding rules

Adding Concept-Forming Operators

We have already seen the operator `:some`. Thus, we could add the operator `:at-least`, where (`:at-least` *n R C*) describes individuals related by role *R* to *at-least n* individuals of *C*.

In the same way, we could add also the operator `:at-most`, where (`:at-most` *n R C*) describes individuals related by role *R* to *at-most n* individuals of *C*.

Adding Concept-Forming Operators

Examples:

- (`:at-least` 2 *hasChild Thing*) denotes all the parents that have two or more children
- (`:at-most` 1 *hasChild Male*) denotes all the parents that have at most only one male child

What about the following examples?

1. (`:and` (`:at-least` 4 *R C*) (`:at-most` 3 *R C*))
2. (`:and` (`:at-least` 3 *R C*) (`:at-most` 3 *R C*))

The first should be inconsistent, while the latter could inspire, for example, the definition of the operator `:exactly`, where (`:exactly` *n R C*) denotes individuals related by role *R* to *exactly n* individuals belonging to *C*

Adding Concept-Forming Operators

Unfortunately, usign qualified cardinality restriction increases the computational complexity of subsumption (entailment)
We can use unqualified version of these operators if we do not need to specify the concepts

We can use (`:at-least` *n R*), (`:at-most` *n R*) and (`:exactly` *n R*)

Adding Concept-Forming Operators

It would be nice to specify that a role can be filled only by individuals belonging to a certain set (without recurring to a concept). We could add the operator `:one-of`, where (`:one-of` a_1 ... a_n) is a concept satisfied only by a_i. Used in conjunction with `:all` would lead to a *restriction* on the individuals that could fill a certain role

Example

Beatles $\equiv$ (`:and` *BandMember* (`:one-of` *jhon paul george ringo*))

Adding Concept-Forming Operators

If we introduce new operators, enriching the expressivity, we can find some new problem. For example, the structural comparison algorithm cannot take into account some interaction between the negation and disjuction operators.

For example $C_1 \sqcup \neg C_1$ subsumes every concept, even if such a concept does not mention at all the atomic concept C_1

We must use other algorithms for computing subsumption

Relating the Roles

What if we want to relate certain role? Suppose you want to say that a daughter is a child
We can add the opertor (`:subrole-of` R_1 R_2). Similarly we can add also the operator (`:same-as` R_1 R_2), which equates fillers of roles R_1 and R_2.

Example
(`:subrole-of` *Daughter Child*)

Unfortunately, also these simple extensions increase the computational complexity of entailment, and if allowed with *role chains*, can lead to undecidability

Knowledge Base - a more precise definition

Formally, a Knowledge Base is a triple composed of $(\mathcal{A}, \mathcal{T}, \mathcal{R})$ where:

- $\mathcal{A}$ is called **ABox**, Assertional Box: a list of facts about individuals
- $\mathcal{T}$ is called **TBox**, Terminological Box: a list of axioms that describes the concepts
- $\mathcal{R}$ is called **RBox**, Relational Box: a list of axioms that describes the roles

Relating the Roles

Until now we treated roles and primitive concepts. What if, for example, we want to consider a role defined as *conjunction* of more roles?

What if we want to add *inverse* of a role?

Rules

In the language presented here, it could be difficult to say, for example, that all instances of one complex concept have a certain property

- we could do this by adding definition of the type $C_1 \equiv C_2$, where C_1 and C_2 are complex concepts with their own definition
- but, expecially with complex concepts, this could lead to several classification problems, when computing subsumption

Some description logics allows the introduction of rules, like for example:

$$\textbf{If } C_1(a) \textbf{ and } C_2(b) \textbf{ then } R(a, b)$$

The rule above means that every combination of individuals of class C_2 and of class C_2 are R-connected

Outline

Description Logics

Description Logics are a family of logics. Each logic is different depending on which **operators** are admitted in the logic.
A description logics' name depends on the operators included.

- $\mathcal{ALC}$ stand for *Attributive Language with Complement*
- $\mathcal{S}$ is a synonim of $\mathcal{ALC}$ augmented with *transitive roles*

Description Logics

Following this line, a set of letters indicate the expressivity of the logic and name the logic:

- $\mathcal{F}$ Functional Properties
- $\mathcal{E}$ Full Existential Qualification
- $\mathcal{U}$ Concept Union (`:or`)
- $\mathcal{C}$ Complex Concept Negation
- $\mathcal{H}$ Role Hierarchy

Description Logics

- $\mathcal{R}$ Complex role inclusion axioms
- $\mathcal{O}$ Nominals (`:one-of`)
- $\mathcal{I}$ Inverse Properties
- $\mathcal{N}$ (Unqualified) Cardinality Restriction ((`:at-least` *R*), (`:at-most` *R*), (`:exactly` *R*))
- $\mathcal{Q}$ Qualified Cardinality Restriction ((`:at-least` *R C*), (`:at-most` *R C*), (`:exactly` *R C*))
- ($\mathcal{D}$) Use of datatypes properties

Outline

Semantic Web

- **Semantic Web**
  - Aims at making information available in a form that is understandable by machines
  - The W3C Working group developed the Web Ontology Language (OWL)
    - OWL is a family of knowledge representation formalisms for defining knowledge bases
    - Based on Description Logics. Which logic?

Outline

OWL

OWL defines three different sublanguages:

- *OWL-Lite*, based on $\mathcal{SHIF}(\mathbf{D})$
- *OWL DL*, based on $\mathcal{SHOIN}(\mathbf{D})$
- *OWL Full*, highly expressive, can be also high-order logic

OWL-Lite

- Limited support for certain features (ex.: cardinality)
- Good for thesauri or hierarchies
- Intended to be easily computable
- Tools development is difficult as for the other languages

OWL DL

- Good expressiveness
- Computationally complete
- Decidable
- Availability of practical reasoning algorithms

OWL DL

OWL provides concept constructors and axioms

Constructor	DL Syntax	Example
`intersectionOf`	$C_1 \sqcap \ldots \sqcap C_n$	$Human \sqcap Male$
`unionOf`	$C_1 \sqcup \ldots \sqcup C_n$	$Doctor \sqcap Lawyer$
`complementOf`	$\neg C$	$\neg Male$
`oneOf`	$\{x_1, \ldots, x_n\}$	$\{jhon, mary\}$
`allValuesFrom`	$\forall r.C$	$\forall hasChild.Doctor$
`someValuesFrom`	$\exists r.C$	$\exists hasChild.Lawyer$
`hasValue`	$\exists r.\{x\}$	$\exists citizenOf.\{USA\}$
`minCardinality`	$(\geq n\ r)$	$(\geq 2\ hasChild)$
`maxCardinality`	$(\leq n\ r)$	$(\leq 1\ hasChild)$
`inverseOf`	r^-	$hasChild^-$

OWL DL Constructors

- `intersectionOf`: a concept is defined as conjunction (AND) of other concepts. E.g.: $Human \sqcap Male$
- `unionOf`: a concept is defined as the disjunction (OR) of other concepts. E.g.: $Doctor \sqcap Lawyer$
- `complementOf`: a concept is defined as all the individuals that are not members of a given concept. E.g.: $\neg Male$ stands for all the females
- `oneOf`: a concept is defined as a specified set of individuals. E.g.: $Beatles \equiv \{ jhon, paul, george, ringo \}$

OWL DL Constructors

- `allValuesFrom`:all the individuals that are in relation r only with individuals of a certain class. E.g.: $\forall hasChild.Male$
- `someValuesFrom`: all the individuals that are in relation r with *at least* one individual of a certain class. E.g.: $\exists hasChild.Male$
- `hasValue`: all the individuals that are in relation r only with a certain individual. E.g.: $\exists hasPet.\{tom\}$

OWL DL Constructors

- **`minCardinality`**: all the individuals that are in relation *r* with *at least n* individuals. E.g.: all the parents that have at least 2 children ($\geq$ 2 *hasChild*)
- **`maxCardinality`**: all the individuals that are in relation *r* with *at most n* individuals. E.g.: all the parents that have at most 2 children ($\leq$ 2 *hasChild*)
- **`inverseOf`**: relations have a direction from a *domain* to a *range*. It is common to define also the property from the range to the domain. It is useful to define that a relation is the inverse of another one. E.g.: *hasChild inverseOf hasParent*. If we have (*mario hasChild luigi*), then it also holds (*luigi hasParent mario*)

OWL DL Axioms

In OWL DL a property (role) can be defined as **functional**, which means that there can be only one value y for each x in relation with r. There cannot be two *distinct* y_1 and y_2 such that we have $r(x, y_1)$ and $r(x, y_2)$.

Example consider the relation *childOfFather*, and consider the kid *luca*. If we have $childOfFather(luca, f_1)$ and $childOfFather(luca, f_2)$, then we can conclude:

1. f_1 and f_2 are the same person, i.e. the father of *luca*
2. if I stated previously that $f_1 \neq f_2$, then my KB is inconsistent

Is also possible to define **transitive** roles and hierarchy relations between roles

OWL Full

- Has different, highly expressive, semantics (classes as both collections of individuals and individuals)
- Not decidable
- Complete reasoning support is unlikely

OWL 1.1 and OWL 2

The W3C has updated the recommendation of OWL, introducing OWL 1.1 and OWL 2, both based on $\mathcal{SROIQ}(\mathbf{D})$ description logics

In particular, OWL 2 introduces "profiles" such as

- *OWL 2-EL* fragment with polynomial time reasoning complexity
- *OWL 2-QL* simplifies support to queries
- *OWL 2-RL* OWL subset meant to handle rules

Outline

Tools

The definiton of OWL paved the way for the definition of a large set of tools which allow to handle and manage the information of the Semantic Web.

- KB editor, e.g. Protégé
- reasoners, e.g. Pellet, RacerPro, FaCT++

OWL DL is not the only choice. For example, the onology SnoMed is based on $\mathcal{EL}$ with additional role properties

Outline

Semantic Web: What Is It?

We have spoken about Semantic Web. But what is the Semantic Web? What does it means?

"The Semantic Web is about **two things**. It is about **common formats for integration and combination of data** drawn from diverse sources, where on *the original Web mainly concentrated on the interchange of documents*. It is also about **language for recording how the data relates to real world objects**. That allows a person, or a machine, to start off in one database, and then move through an unending set of databases which are connected not by wires but by being about the same thing."

SOURCE: W3C Sematic Web Initiative

Semantic Web: What Is It?

The Semantic Web objective is to use and reason upon all the available data on the internet **automatically**

How?

Web

The information are represented by means of

- natural language
- images, multimedia, etc.

The Web was built around human users, which can easily exploit all these means for deducting facts from partial information, merging information from different sources, creating mental associations between, e.g., fact and images, text and videos, ...
A human user uses *different communication channels* at the same time

Web

The content of web pages is published with the principal aim of being **human readable**

Standard HTML is focused on **how** to represent the content, not on *what* is represented

There are some tags (`<title>`) that provide an implicit semantics but their content is not structured and their use is not really standardized

Web

Il vero rivale di Google è Amazon. Lo dice il presidente Schmidt

Non sono i motori di ricerca come Bing o Yahoo! a far tremare Mountain View. È piuttosto il colosso dell'ecommerce che, proprio come Big G, risponde alle richieste degli utenti. E amplia il proprio servizio di consegne, che diventa a pagamento

Lo leggo dopo | 14 ottobre 2014

35 Consiglia Condividi 11 Tweet 2 +1

AMAZON risponde alle domande e alle richieste degli utenti. Proprio come Google. Per questo Eric Schmidt, ex amministratore delegato del colosso di Mountain View e ora presidente del consiglio di amministrazione, considera il colosso dell'ecommerce Amazon come il suo principale rivale tra i motori di ricerca. "Molti" dice Schmidt

```
<h1>
  <b> Il vero rivale di
  Google &egrave; Amazon.
  Lo dice il presidente
  Schmidt </b>
</h1>
```

We can identify the title by means of its representation (<h1>, <b>), but if the designer changes the format of the web pages?

Web

Web pages contain also links to other pages, but...

- no information on the link itself
  - what does it represent?
  - what does the linked page/resource represent?

Example In my home page there are links to other home pages

- which ones link to colleagues?
- which ones link to friends?

Web

$$\textit{Web} = \textbf{Layout} + \textbf{Routing}$$

The problem is that it is impossible to *automatically reason* about the data

- The Web can be seen as an immense database, from where we would like to find all the information we need
- Every day the Web is queried by millions of users that use *search engines* to find information by specifing *keywords*
- A succesful search depends on many parameters, such as, the quality of the indexing and search algorithm, the number of total pages that have been indexed, the (meta-)content of the pages

Web

- Every page can link to anything
- Anyone can publish anything on the web, about everything
  - *Distribution* of the information
  - *Inconsistency* of the information
  - *Incompleteness* of the information

Web 2.0

- Term coined in 1999 by Darcy DiNucci, a consultant on electronic information design, and was popularized by Tim O'Reilly at the O'Reilly Media Web 2.0 Conference
- A new way of using the web, where users can do more than just search and read information
  - comment on published information, add and share content
  - create user account and profile for increasing the social participation by using blog, wikis, social networks, folksonomies, ...
  - use net-distributing application, e.g., user interface, web services, file storage, cloud computing, web-office

How To Allow The Use Of Semantic Web...

By *extending* the current web with **knowledge** and **semantic information** about the content of the pages...

... but preserving the characteristics of the web

- globality
- information distribution
- information inconsistency
- information incompleteness

Adding Information About The Content

Adding information is not enough

- Information should be structured: classification, *ontologies*
- There is the need of some inference mechanism, but for which logic?
- We should be able to find proofs for the inferred new knowledge, in order to understand what originated such information

Semantic Web Applications

The Semantic Web has application fields in every possible domain

To cite some:

- search engines
- intelligent assistant
- database integration
- digital libraries
- data analysis and recommendation services
- web services and cloud computing

Search Engines

Mid-size and big industries need to index and easily access and retrieve all the documentation

Intelligent Assistant

Original scenario prosed by Tim Berners-Lee

Two Semantic Web Agent (running on a smartphone):

- Synchronize the agenda
- Generate and agree upon a plan
- Delegate tasks each other
- Exploit business contracts to support cost-based decisions

Database Integration

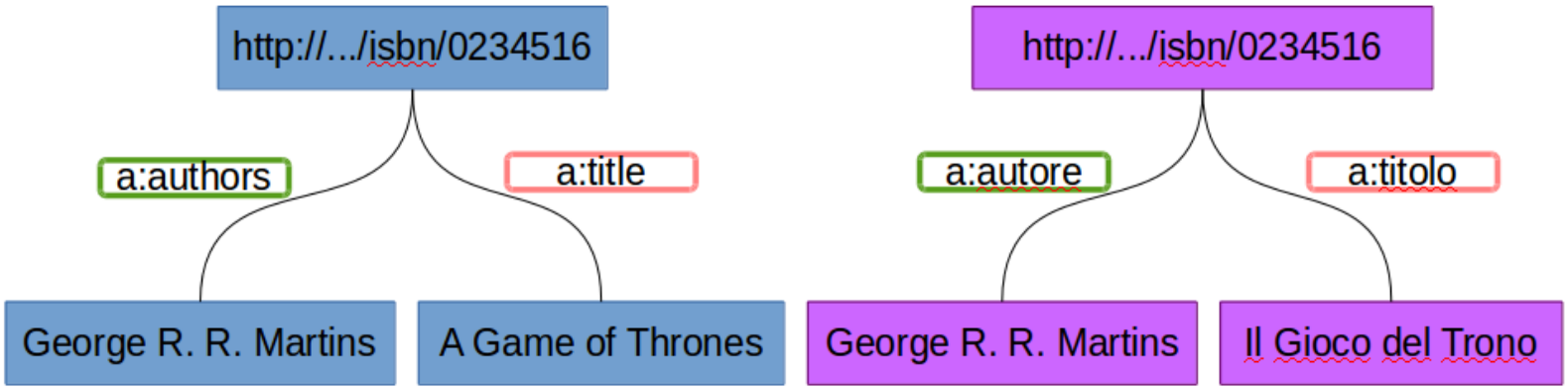
Frequently, we need to integrate several different databases

TODO

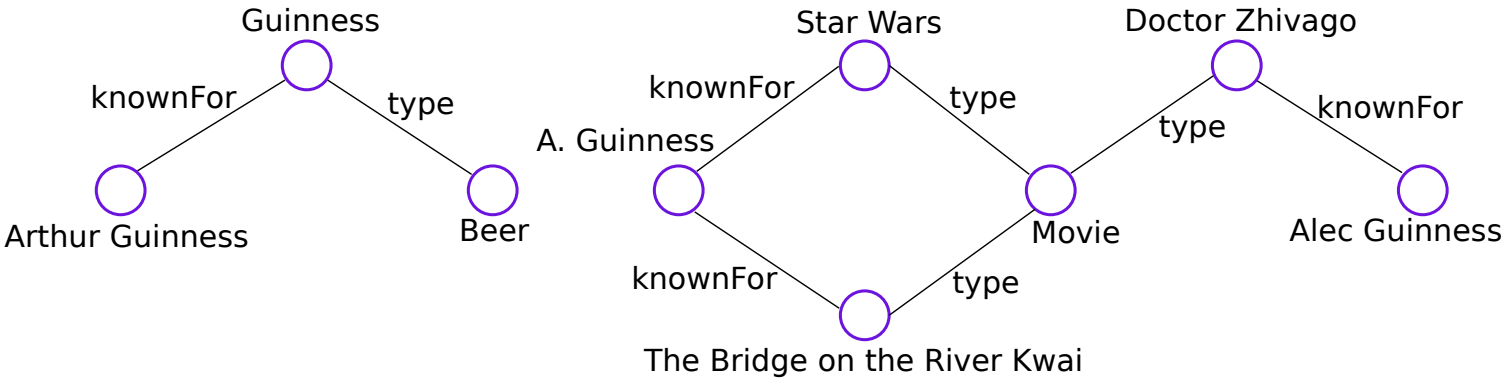
- Define new, more abstract data structures to capture the data heterogeneity
- Merge on these abstract representations
  - Resolve name conflicts
- Execute more complex and more expensive queries

Database Integration

The merge is succesfull if the data abstraction correctly maps data equivalence



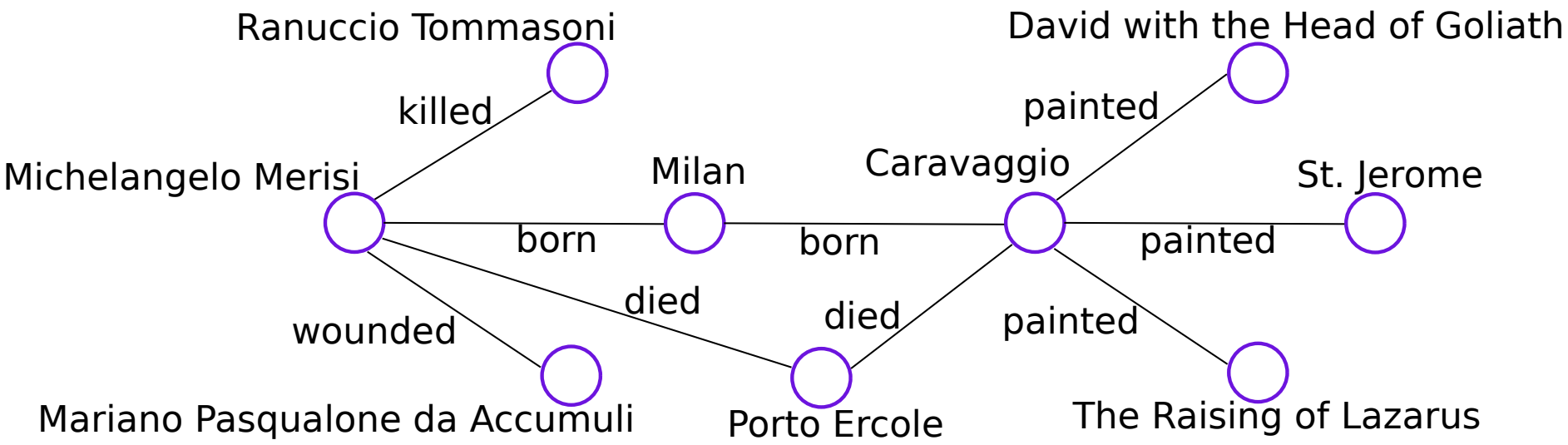
Example



Nodes referring to “A. Guinness” and “Alec Guinness” refer to the identical entity, the actor Alec Guinness, who is a different person than Arthur Guinness, the founder of the Guinness brewery

- The name “A. Guinness” alone is not be sufficient to correctly match the right entity.
- A relational approach can perform the correct matching using links revealing the occupations of the nodes

Example



Nodes referring to “Michelangelo Merisi” and “Caravaggio” seem to refer to different people, a criminal and a painter.

Data Integration

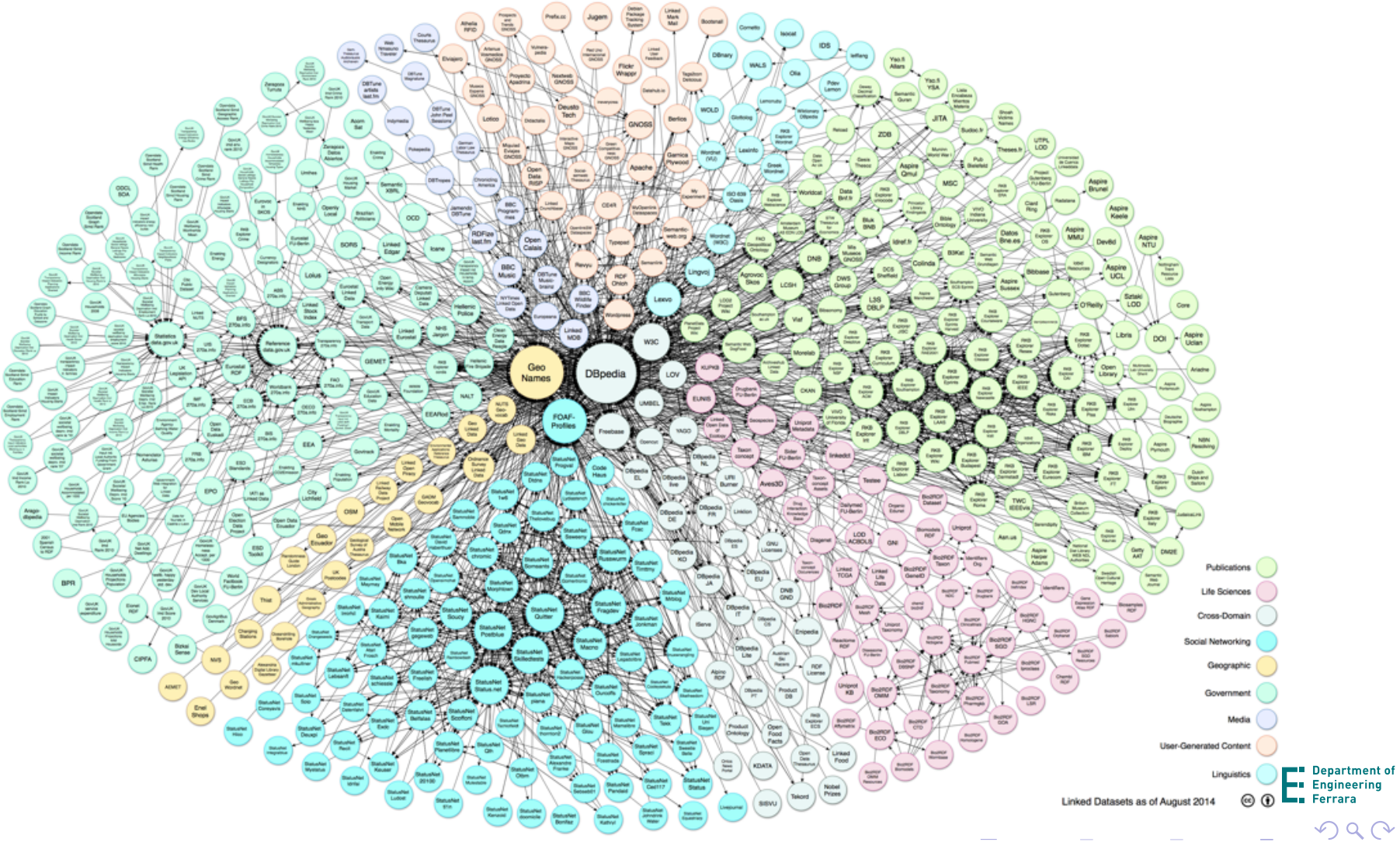
The semantic web intrisecally supports such data abstraction process

Difference:

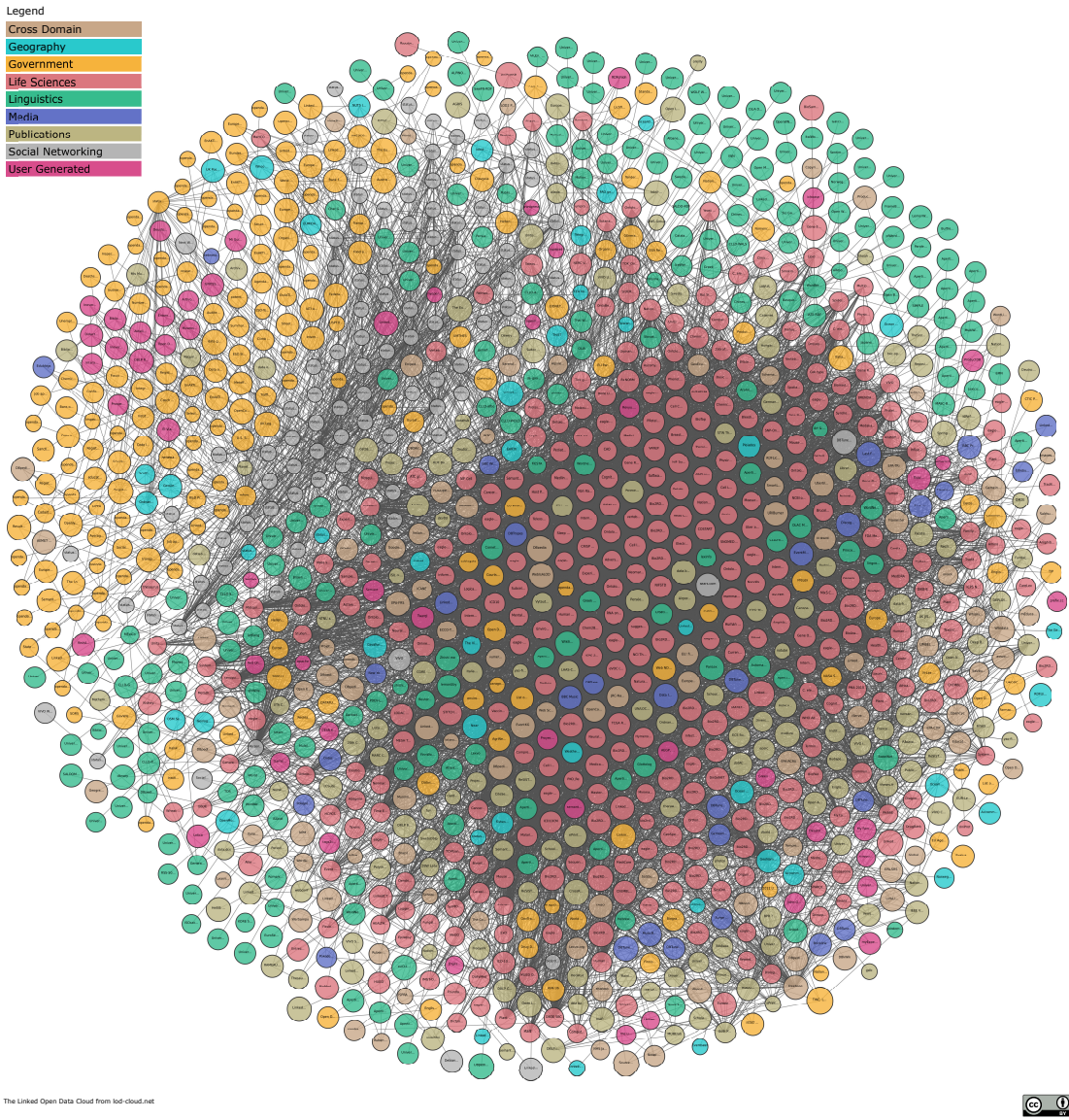
- The entire web is the background
- data are considered as distributed
- Use of ontologies (more expressivity of the ER model)

Open Linked (Big) Data

Big Data (August 2014 - https://lod-cloud.net))



Big Data (Now - https://lod-cloud.net)



Digital Libraries

Without meta information there are no guarantees about a resource is about what we search
Before them, if I would like to download a certain song or album, I had many chances to download something different

- Universal Digital Library
- Spotify

Data Analysis and Recommendation Services

We need to analyze large amount of data in order to manage them and find usefull information

- Crif
  - Italian company that does analysis of credit and financing based on an ontology modeling the Italian language
- Google Play Music, Last.fm, ...
  - Use ontology for modeling information about music, digital libraries and other services and analyze your liking for doing recommendation of what to listen
- Google AdWords and AdSense, Amazon Advertising, ...

Web Services

"A Web service is a software system designed to **support interoperable machine-to-machine interaction** over a network. It has an **interface** described in a machine-processable format (specifically WSDL). Other systems **interact** with the Web service in a manner prescribed by its description using SOAP **messages**, typically conveyed using **HTTP with an XML serialization** in conjunction with other **Web-related standards**"

SOURCE: Web Services Architecture

`http://www.w3.org/TR/ws-arch/`

Web Services

Through WSDL it is possible to dynamically retrieve location and interface of a service, but, it is not possible to retrieve information on *what* the service does

Example: invoke a service called *sum* that accepts two *integers* an return an *integer*.
How do we know that such service calculate the sum? It could, for example, convert to string the values of the integer, concatenate them and return the integer value of the so concatenated resulting string

Web Services

We need extra information, given by the semantics of the Web Service

- Semantic information of the functionality offered by the service (precondition, input, output, effects)
- Constraints about the data
- ...

Outline

Uniform Resource Identifier

Semantic Web applications need to uniquely identify concepts

Use of *Uniform Resource Identifier* (URI)

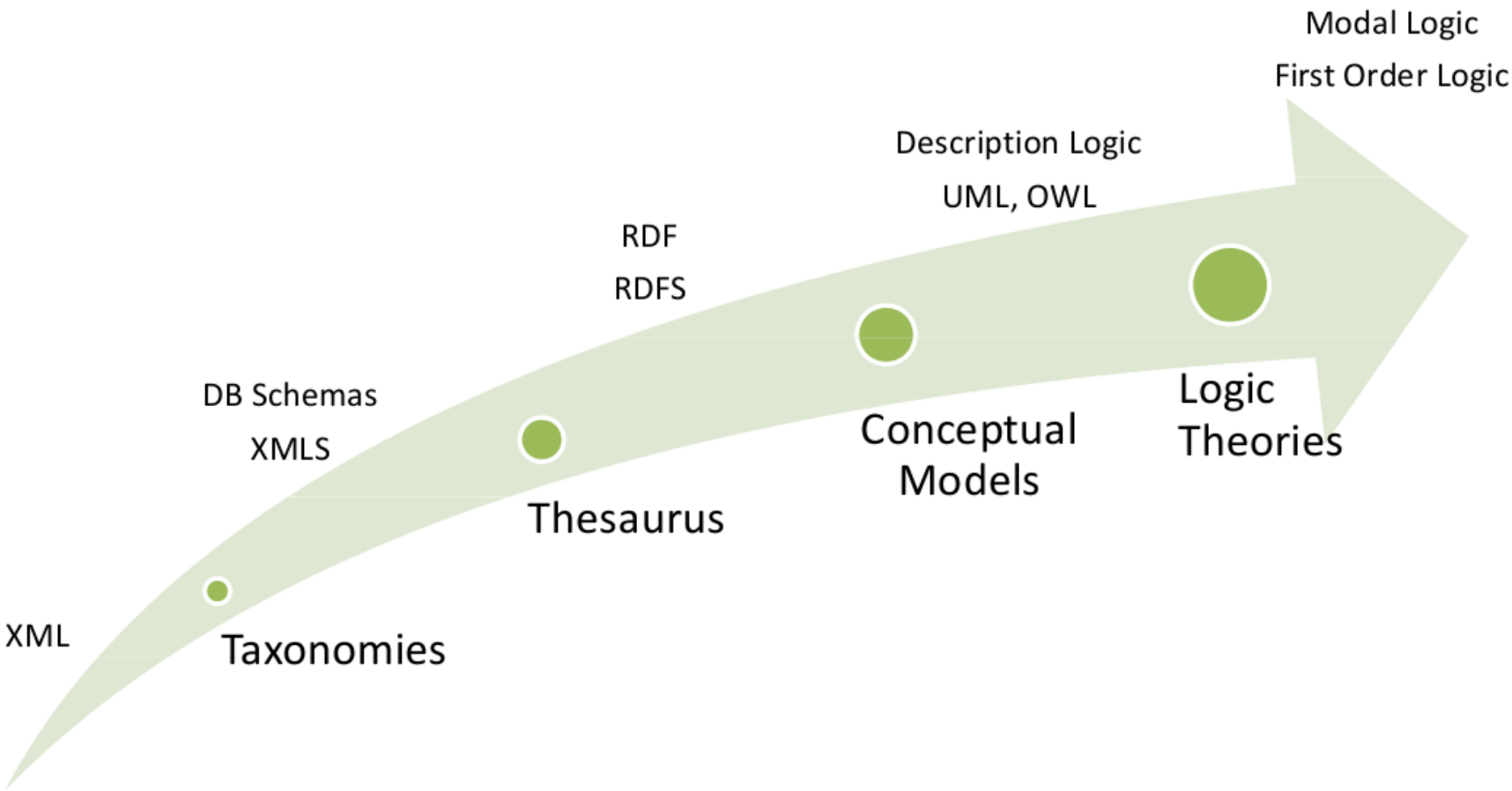
- By definition, URI guarantees unicity of the names
- To each URI correspond one and only one concept but more URI can refer to the same concept
- it is not necessary that to each URI corresponds some content

Semantic Models

All the applications above need semantic information
How to represent them?

Which language and expressivity?
Reasoning is needed? What about performances?

Semantic Models



Ontologies

A **formal**, **explicit description** of a **domain** of interest

- Classes (Concepts)
- Semantic relation between classes (roles)
- Properties associated to a concept (restrictions, ...)
- Logic (axioms, inference rules)

Ontologies


Role
Symbol
Religious-Symbol
Political-Symbol
National-Symbol
Economic-Symbol
Person-Role
Leader
Religious-Leader
Church-Reformer
Ecclesiarch
Other-Religious-Leader
Pope
Clerical-Leader
Secular-Leader
Head-of-State
King
Emperor
Sultan
Prince
Queen
Magnate
Head-of-Government
Other-Religious-Person
Scientist
Philosopher
Discoverer
Saint
Military-Person
Functionary
Minister
Politician
Diplomat
Cleric
Inventor
Philanthropist
Creator
Engineer
Artist
Composer
Painter
Sculptor
Publisher
Architect
Author
is-a

Ontologies

- An ontology provides a structured model of a domain
- Solves term ambiguity

Examples

- Dublin Core, focussed on document
- WordNet
- Gene Ontology, genomic
- Protein Ontology, proteomic
- SnoMed, a very important ontology in the medical field

- 35 Case Studies and 13 Use Cases available at `http://www.w3.org/2001/sw/sweo/public/UseCases/`
- a number of ontologies is available at `http://info.slis.indiana.edu/~dingying/Teaching/S604/OntologyList.html`

Ontologies - Is XML Enough?

"XML is only the first step to ensuring that computers can communicate freely. XML is an alphabet for computers and as everyone who travels in Europe knows, knowing the alphabet doesn't mean you can speak Italian or French"

Business Week, March 18, 2002

From XML ...

eXtensible Markup Language (XML) was created for supporting data exchange between heterogeneous systems

- No presentation information
- Human and machine readable

Extensible, so that anyone can represent any type of data **Hierarchically structured** by means of tags

... Through RDF ...

Resource Description Framework (RDF) is a standard W3C

- XML-based
- Used for representing knowledge
- Based on the concept of triples

<subject, predicate, object>
<resource, attribute, value>

RDF - Representations

- XML representation

```
<rdf:RDF
    xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
    xmlns:contact=http://www.w3.org/2000/10/swap/pim/contact#
>

    <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
        <contact:fullName>Eric Miller</contact:fullName>
        <contact:mailbox rdf:resource="mailto:em@w3.org"/>
        <contact:personalTitle>Dr.</contact:personalTitle>
    </contact:Person>

</rdf:RDF>
```

RDF - Representations

- Graph representation

RDF Schema

RDF can be intended also as a description of resource attributes and of the values of such attributes
RDFS is a set of classes with certain properties which provide basic element for the description of ontologies

- type
- subClassOf
- subPropertyOf
- range
- domain

... To OWL

Ontology Web Language (OWL) is a standard W3C

- RDF/RDFS-based
- Formal Semantics (Description Logic Fragments)
- Three level of expressivity/complexity
  - OWL-Lite, decidable
  - OWL DL, decidable and more expressive
  - OWL Full, not decidable, highly expressive, minimal compatibility with RDFS

OWL - Features

- Classes (categories): subClassOf, intersectionOf, unionOf, complementOf, enumeration, equivalence, disjoint
- Properties (Roles, Relations): symmetric, transitive, functional, inverse Functional, range, domain, subPropertyOf, inverseOf, equivalentProperty
- Instances (Individuals): sameIndividualAs, differentFrom, allDifferent

OWL - Representations

- Functional-Style syntax
- RDF/XML syntax
- Manchester syntax
- Turtle syntax
- OWL XML syntax

OWL - Tools

Many tools for OWL

- Editors (37 listed at `http://www.w3.org/2001/sw/wiki/Category:Editor`)
- Reasoners (39 listed at `http://www.w3.org/2001/sw/wiki/Category:Reasoner` other more at `http://owl.cs.manchester.ac.uk/tools/list-of-reasoners/`)

Quite often integrated in a comprehensive framework
We will see

- the editor Protégé `http://protege.stanford.edu/`
- the reasoner Pellet `https://github.com/stardog-union/pellet`

Outline

Sematic Web - Problems

- RDF adoption
  - Adding semantic content is expensive
- Ontologies
  - To produce a new one is highly expensive and time-demanding
  - An ontology changes in time
    - Updating costs
    - Managing costs
  - Use of machine learning approaches

Sematic Web - Problems

- Which use of the data?
  - Censorship problems and freedom
  - Privacy problems
- Data are already available on the web, it is sufficient to add metadata to and/or extract them
  - But how to add metadata? Wich format?
    - Avoiding wrong content
    - Avoiding the addition of metacrap
    - The output dimension and the time needed for creating contents will probably double because there would need to be two formats for one piece of data: one for human viewing and one for machines
    - Google Bombing: “miserable failure”, “more evil than Satan himself” and many others
  - But how to extract and then represent data?
  - Again.. use of machine learning approach

Sematic Web - Problems

- Computationally expensive
  - adoption of a fragment of Description Logic
  - we do not need all the expressive power in every application

Conclusions

- Semantic Web: adding semantic information to web resources
- Tim Berners-Lee has described the semantic web as a component of ”Web 3.0”
  In 2006, he said: “People keep asking what Web 3.0 is. I think maybe when you’ve got an overlay of scalable vector graphics - everything rippling and folding and looking misty - on Web 2.0 and access to a semantic Web integrated across a huge space of data, you’ll have access to an unbelievable data resource ...”
  We are still far from that moment... but not so far as one can think

Interesting Sites

- DBPedia, structured information from Wikipedia
  `https://wiki.dbpedia.org/`
- GoPubmed, an improved search in Pubmed, is a knowledge based search engine via ontologies for biomedical text
  `https://www.gopubmed.org`
- FOAF, Friend Of A Friend, a vocaboulary used to describe the relationships people have with other people and the ”things” around them. To start with if you want to add semantic content to your home page
  `http://www.foaf-project.org/`
  `http://xmlns.com/foaf/spec/`

Interesting Sites

- eagle-i.net, an open source, semantic web platform for entering and publishing information about resources used in biomedical research
  `https://www.eagle-i.net/`
- NextBio, a database consolidating life sciences experimental data tagged and connected via biomedical ontologies
  `http://www.nextbio.com/`

Interesting Sites

- MusicBrainz, an open music encyclopedia that collects music metadata and makes it available to the public.
  `https://musicbrainz.org/`
- NELL (Never-Ending Language Learning), attempts to create a computer system that learns over time to read the web
  `http://rtw.ml.cmu.edu/rtw/`

Readings

- Knowledge Representation and Reasoning, R. J. Brachman and H.J. Levesque, Morgan Kaufmann 2004.
- Semantic Web Primer, 2nd edition in 2008, G. Antoniu and F. van Harmelen
- Semantic Web for the Working Ontologist, D. Allemang and J. Hendler, 2008
- Probabilistic Semantic Web: Reasoning and Learning, R. Zese, IOS Press 2017

Readings

- The W3C activity
  `http://www.w3.org/2013/data/`
  `http://www.w3.org/2001/sw/wiki/Main_Page`
- OWL Guide
  `http://www.w3.org/standards/techs/owl`
- RDF Primer
  `http://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/`
- DL Website
  `http://www.dl.kr.org/`

Disclaimer

A large part of this presentation has been taken from and inspired by the Descritpion Logics Website (`http://www.dl.kr.org/`), and by “F. Chesani, Introduction to Description Logic(s)” and “F. Chesani, Introduction to Semantic Web”

Thanks.

Questions?