Ontology Engineering

CSE 595 – Semantic Web

Instructor: Dr. Paul Fodor

Stony Brook University

http://www3.cs.stonybrook.edu/~pfodor/courses/cse595.html

Lecture Outline

- Constructing Ontologies
- Reusing Existing Ontologies
- Semiautomatic Ontology Acquisition
- Ontology Mapping
- Exposing Relational Databases
- Semantic Web Application Architecture

Ontology Engineering

- ***Ontology Engineering*** are methodological issues that arise when building ontologies, in particular, constructing ontologies manually, reusing ontologies, and using semiautomatic methods (populate ontology instances from relational databases)
- Constructing Ontologies **main stages**:
    1. **Determine scope**
    2. **Consider reuse**
    3. **Enumerate terms**
    4. **Define taxonomy**
    5. **Define properties**
    6. **Define facets**
    7. **Define instances**
    8. **Check for anomalies**

statista.com

1. Determine Scope

- Developing an ontology of a domain is not a goal in itself
  - Define the set of data and its structure for **other programs to use**
  - An ontology is a model of a particular domain, **built for a particular purpose**
  - An ontology is by necessity **an abstraction of a particular domain**, and there are always multiple viable alternatives
    - What is included in this abstraction should be determined by the use to which the ontology will be put, and by future extensions that are anticipated
- Basic questions to be answered at this stage are:
  - ***What is the domain that the ontology will cover?***
  - ***For what we are going to use the ontology?***
  - ***For what types of questions should the ontology provide answers?***
  - ***Who will use and maintain the ontology?***

2. Consider Reuse

- With the spreading deployment of the Semantic Web, many ontologies, especially for common domains (social networks, medicine, geography), are **available for use**
- Thus, we rarely have to start from scratch when defining an ontology

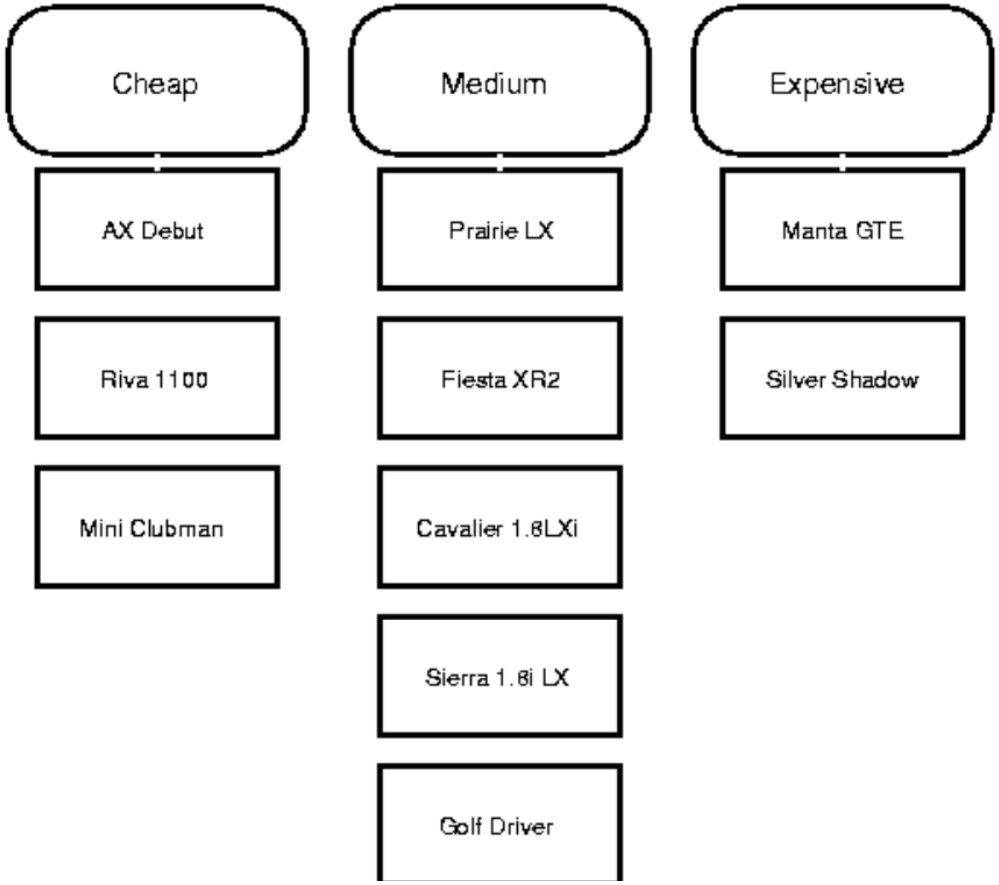
3. Enumerate Terms

- Write down in an unstructured list all the relevant terms that are expected to appear in the ontology
  - nouns form the basis for class names
  - verbs (or verb phrases) form the basis for property names (e.g., is part of, has component)
- Traditional knowledge engineering tools such as laddering and grid analysis can be productively used at this stage to obtain both the set of terms and an initial structure for these terms
  - Laddering involve the construction, reviewing modification and validation of hierarchical knowledge, often in the form of ladders (i.e. tree diagrams)
    - The expert and knowledge engineer both refer to a ladder presented on paper or a computer screen, and add, delete, rename or update
  - Grid KE = tabular representation for what column solution is applicable to which problem (e.g. timelines)

statista.com

3. Enumerate Terms

- Grid:



http://www.aiai.ed.ac.uk/project/ix/project/kingston/phdjkk.pdf

statista.com

4. Define Taxonomy

- After the identification of relevant terms, these terms must be organized in a taxonomic (subclass) hierarchy in a top-down or a bottom-up fashion
- A is a **`rdfs:subClassOf`** of B, then every instance of A must also be an instance of B

5. Define Properties

- Attach properties to the highest class in the hierarchy to which they apply
  - Interleaved with the previous step
- While attaching properties to classes, provide statements about the domain and range of these properties
- There is a methodological tension here between generality and specificity
  - It is attractive to give properties as general a domain and range as possible, enabling the properties to be used (through inheritance) by subclasses
  - On the other hand, it is useful to define domain and range as narrowly as possible, enabling us to detect potential inconsistencies in the ontology by spotting domain and range violations

6. Define Facets

- Enrich the previously defined properties with facets:
  - **Cardinality**: specify for as many properties as possible whether they are ***allowed*** or ***required*** to **have a certain number of different values**
    - Often, occurring cases are "*at least one value*" (i.e., required properties) and "*at most one value*" (i.e., single-valued properties)
  - **Required values** can be specified in OWL, using **`owl:hasValue`** or (less stringent, a property is required to have some values from a given class and not necessarily a specific value) **`owl:someValuesFrom`**
  - **Relational characteristics** of properties: symmetry, transitivity, inverse properties, and functional values

6. Define Facets

- After this step in the ontology construction process, it will be possible to check the ontology for internal inconsistencies
  - This is not possible before this step, simply because RDF Schema is not rich enough to express inconsistencies
  - Examples of often occurring inconsistencies are:
    - Incompatible domain and range definitions for transitive, symmetric, or inverse properties
    - Cardinality properties
    - Property values that can conflict with domain and range restrictions

7. Define Instances

- Use ontologies to organize or create sets of instances
- Typically, the number of instances is many orders of magnitude larger than the number of classes from the ontology
  - Ontologies vary in size from a few hundred classes to tens of thousands of classes
  - The number of instances varies from hundreds to hundreds of thousands, or even larger
    - Because of these large numbers, populating an ontology with instances is typically not done manually
    - Often, instances are retrieved from legacy data sources such as databases
    - Another often used technique is the automated extraction of instances from a text corpus

statista.com

8. Check for Anomalies

- An important advantage of using OWL rather than RDF Schema is the possibility of detecting inconsistencies in the ontology itself, or in the set of instances that were defined to populate the ontology
  - Check again for the instances:
    - Cardinality properties
    - Property values that can conflict with domain and range restrictions

Lecture Outline

- Constructing Ontologies
- Reusing Existing Ontologies
- Semiautomatic Ontology Acquisition
- Ontology Mapping
- Exposing Relational Databases
- Semantic Web Application Architecture

Reusing Existing Ontologies

- Some ontologies are carefully crafted by a large team of experts over many years:
  - The cancer ontology from the National Cancer Institute in the United States

https://bioportal.bioontology.org/ontologies/NCIT

  - The Art and Architecture Thesaurus (AAT) (125,000 terms)

http://www.getty.edu/research/tools/vocabularies/aat

http://www.getty.edu/research/tools/vocabularies/index.html

    - The Getty Thesaurus of Geographic Names (TGN) (1 million entries)
    - The Union List of Artist Names (ULAN) (220,000 entries on artists)
    - The Cultural Objects Name Authority (CONA)

statista.com

Reusing Existing Ontologies

- ***Integrated Vocabularies***:
  - Sometimes attempts have been made to merge a number of independently developed vocabularies into a single large resource
  - The prime example of this is the Unified Medical Language System (UMLS), which integrates 100 biomedical vocabularies and classifications

https://www.nlm.nih.gov/research/umls/

  - The UMLS meta-thesaurus alone contains 750,000 concepts, with over 10 million links between them

statista.com

Reusing Existing Ontologies

- ***Upper-Level Ontologies***:
  - Whereas the preceding ontologies are all highly domain-specific, some attempts have been made to define very generally applicable ontologies (known as ***upper-level ontologies***)
  - Examples:
    - Cyc http://www.opencyc.org with 60,000 assertions on 6,000 concepts
    - Suggested Upper Merged Ontology (SUMO): intended as a foundation ontology for a variety of computer information processing systems

Reusing Existing Ontologies

- ***Topic Hierarchies***:
  - sets of terms, loosely organized in specialization hierarchies that mix different specialization relations, such as is-a, part-of, or contained-in => good starting point for general ontologies

Reusing Existing Ontologies

- ***Linguistic Resources***:
  - Classical WordNet with over 90,000 word sense definitions

  https://wordnet.princeton.edu (Prolog)

  RDF version: http://semanticweb.cs.vu.nl/lod/wn30/
  - VerbNet: grammatical and semantical patterns

  https://verbs.colorado.edu/~mpalmer/projects/verbnet.html
  - PropBank

  https://propbank.github.io
    - corpus of text annotated with information about basic semantic propositions
  - Linguistic Data Consortium (LDC): https://www.ldc.upenn.edu
  - BabelNet with over 300 languages

  http://babelnet.org

Reusing Existing Ontologies

- ***Encyclopedic Knowledge***:
  - Wikipedia: the community-generated encyclopedia
  - DBpedia extracts knowledge from Wikipedia and exposes it as Linked Data using RDF and OWL

http://wiki.dbpedia.org

  - Yago: https://github.com/yago-naga/yago3 leverages Wikipedia, WordNet and GeoNames
  - Wikidata leverages Wikipedia, Wikivoyage, Wikisource

https://www.wikidata.org/wiki/Wikidata:Main_Page

  - Babelnet: http://babelnet.org

statista.com

Reusing Existing Ontologies

- ***Ontology Libraries***:
  - http://owl.cs.manchester.ac.uk/tools/repositories/
  - http://dumontierlab.com/ontologies.php
  - BioPortal: comprehensive repository of biomedical ontologies

  http://bioportal.bioontology.org/
  - Open Biological and Biomedical Ontology (OBO) Foundry

  http://www.obofoundry.org/
    - Chemical Entities, Human Disease Ontology, Gene Ontology, Phenotype And Trait Ontology, PRotein Ontology (PRO), Anatomical Entity Ontology, Antibiotic Resistance Ontology, Biological Spatial Ontology, Clinical measurement ontology, Cell ontology, Drug-drug Interaction and Drug-drug Interaction Evidence Ontology
  - https://protegewiki.stanford.edu/wiki/Protege_Ontology_Library#OWL_ontologies

statista.com

Reusing Existing Ontologies

- ***Ontology Libraries***:
  - http://prefix.cc/ lists the most commonly used namespace prefixes used on the Semantic Web
  - http://swoogle.umbc.edu
  - **Linked Open Vocabularies (LOV):** http://lov.okfn.org/dataset/lov/
    - Latest insertions:
      - imo - The IMGpedia Ontology 2018-03-13
      - eepsa - EEPSA (Energy Efficiency Prediction Semantic Assistant) Ontology 2018-02-25
      - vocals - VoCaLS: A Vocabulary and Catalog for Linked Streams 2018-02-25
      - bto - BOT: Building Topology Ontology 2018-02-19
      - mv - MobiVoc: Open Mobility Vocabulary 2018-01-25

statista.com

Lecture Outline

- Constructing Ontologies
- Reusing Existing Ontologies
- Semiautomatic Ontology Acquisition
- Ontology Mapping
- Exposing Relational Databases
- Semantic Web Application Architecture

Semiautomatic Ontology Acquisition

- There are two core challenges for putting the vision of the Semantic Web into action:
  - support the reengineering task of semantic enrichment for building the web of metadata
    - metadata should be produced at high speed and low cost
      - the task of merging and aligning ontologies for establishing semantic interoperability may be supported by machine learning techniques
  - a means for maintaining and adopting the machineprocessable data that are the basis for the Semantic Web
    - we need mechanisms that support the dynamic nature of the web
- Ontology acquisition remains a time-consuming, expensive, highly skilled, and sometimes cumbersome task that can easily result in a knowledge acquisition bottleneck

statista.com

Semiautomatic Ontology Acquisition

- Tasks that can be supported by machine learning techniques:
  - Extraction of ontologies from existing data on the web
  - Extraction of relational data and metadata from existing data on the web
  - Merging and mapping ontologies by analyzing extensions of concepts
  - Maintaining ontologies by analyzing instance data
  - Improving Semantic Web applications by observing users
- An important requirement for ontology representation is that ontologies must be symbolic, human-readable, and understandable
  - symbolic learning algorithms that make generalizations and to skip other methods like neural networks and genetic algorithms

Semiautomatic Ontology Acquisition

- Machine learning provides a number of techniques that can be used to support these tasks:
  - Clustering
  - Incremental ontology updates
  - Support for the knowledge engineer
  - Improving large natural language ontologies
  - Pure (domain) ontology learning

Semiautomatic Ontology Acquisition

- ***Natural language ontologies (NLOs)*** contain lexical relations between language concepts
  - They are large in size and do not require frequent updates
  - Usually they represent the background knowledge of systems and are used to expand user queries
  - NLO ***learning***: general-purpose techniques for automatically or semi-automatically construction and enrichment of domain-specific NLOs
    - Automated Discovery of Relations
      - Lexico/Syntactic Patterns for Hyponymy
      - Discovery of New Patterns

statista.com

Semiautomatic Ontology Acquisition

- Domain Ontologies capture knowledge of one particular domain, such as pharmacological or printer knowledge
  - Provide a detailed description of the domain concepts in a restricted domain
  - Usually, they are constructed manually, but different learning techniques can assist the (especially the inexperienced) knowledge engineer
    - find statistically valid dependencies in the domain texts and suggest them to the knowledge engineer

Semiautomatic Ontology Acquisition

- Ontology Instances can be generated automatically and frequently updated (e.g., a company profile in the Yellow Pages will be updated frequently) while the ontology remains unchanged
  - The task of learning of the ontology instances fits nicely into a machine learning framework, and there are several successful applications of machine learning algorithms for this (populate the markup without relating to any domain theory)

Semiautomatic Ontology Acquisition

- Ontology creation from scratch by the knowledge engineer
  - machine learning assists the knowledge engineer by suggesting the most important relations in the field or checking and verifying the constructed knowledge bases
- Ontology schema extraction from web documents
  - machine learning systems take the data and metaknowledge (like a meta-ontology) as input and generate the ready-to-use ontology as output with the possible help of the knowledge engineer.
- Extraction of ontology instances populates given ontology schemas and extracts the instances of the ontology presented in the web documents
  - This task is similar to information extraction and page annotation, and can apply the techniques developed in these areas

Semiautomatic Ontology Acquisition

- Ontology integration and navigation deal with reconstructing and navigating in large and possibly machine-learned knowledge bases
  - For example, the task can be to change the propositional-level knowledge base of the machine learner into a first-order knowledge base
- An ontology maintenance task is updating some parts of an ontology that are designed to be updated (like formatting tags that have to track the changes made in the page layout)
- Ontology enrichment (or ontology tuning) includes automated modification of minor relations into an existing ontology
  - This does not change major concepts and structures but makes an ontology more precise

Semiautomatic Ontology Acquisition

- Potentially applicable algorithms:
  - Propositional rule learning algorithms learn association rules or other forms of attribute-value rules
  - Bayesian learning is mostly represented by the Naive Bayes classifiers - based on the Bayes theorem and generates probabilistic attribute-value rules based on the assumption of conditional independence between the attributes of the training instances
  - First-order logic rules learning induces the rules that contain variables, called first-order Horn clauses
  - Clustering algorithms group the instances together based on the similarity or distance measures between a pair of instances defined in terms of their attribute values

statista.com

Lecture Outline

- Constructing Ontologies
- Reusing Existing Ontologies
- Semiautomatic Ontology Acquisition
- Ontology Mapping
- Exposing Relational Databases
- Semantic Web Application Architecture

Ontology Mapping

- It will rarely be the case that a single ontology fulfills the needs of a particular application; more often multiple ontologies will have to be combined
- With reuse rather than development-from-scratch becoming the norm for ontology deployment, ***ontology integration*** (also called ***ontology alignment*** or ***ontology mapping***) is an increasingly urgent task
  - Various linguistic, statistical, structural, and logical methods

Linguistic Methods

- Exploit the linguistic labels attached to the concepts in source and target ontology in order to discover potential matches
  - Stemming
  - Calculating Hamming distances
  - Use specialized domain knowledge
    - Example: the difference between Diabetes Melitus type I and Diabetes Melitus type II is not a negligible difference to be removed by a small Hamming distance

Statistical Methods

- Use **instance data** to determine correspondences between concepts
  - If there is a significant statistical correlation between the instances of a source concept and a target concept, there is reason to believe that these concepts are strongly related by:
    - An equivalence relation OR
    - A subsumption relation
- These approaches rely on the **availability** of a sufficiently large corpus of instances that are classified in both the source and the target ontologies

Structural Methods

- Since ontologies have internal structure, **exploit the graph structure** of the source and target ontologies and try to determine similarities between these structures (graph isomorphism)
- Can be used in conjunction with the previous methods
  - If a source concept and a target concept have similar **linguistic labels**, then the dissimilarity of their graph neighborhoods could be used to detect homonym problems where purely linguistic methods would falsely declare a potential mapping

Logical Methods

- Ontologies are "***formal*** *specifications of a shared conceptualization*" (R. Studer) and we exploit the logical formalization of both source and target structures
- A serious limitation of this approach is that many practical ontologies are **semantically rather lightweight** and thus do not carry much logical formalism with them

Mapping Implementations

- Frameworks for ontology mapping:
  - R2R Framework: http://wifo5-03.informatik.uni-mannheim.de/bizer/r2r/
    - enables Linked Data applications which discover data on the Web, that is represented using unknown terms, to search the Web for mappings and apply the discovered mappings to translate Web data to the application's target vocabulary
  - Limes: http://aksw.org/Projects/LIMES.html
    - link discovery based on the characteristics of metric spaces
  - http://sameas.org collects and exposes owl:sameAs mappings from several different sources
- The research community has run the Ontology Alignment Evaluation Initiative http://oaei.ontologymatching.org to encourage the creation of accurate and comprehensive mappings
  - assessing strengths and weaknesses of alignment/matching systems
  - comparing performance of techniques
  - increase communication among algorithm developers
  - improve evaluation techniques

statista.com

Lecture Outline

- Constructing Ontologies
- Reusing Existing Ontologies
- Semiautomatic Ontology Acquisition
- Ontology Mapping
- Exposing Relational Databases
- Semantic Web Application Architecture

Exposing Relational Databases

- Most websites today are dynamically generated from data stored in relational databases
  - Mapping Terminology:
    - A ***table*** (also called a ***relation***) consist of series of columns named ***attributes***
    - Each of the rows of the table is called a ***tuple***

Homes

HomeId	City	Price (Euros)
1	Amsterdam	100 000
2	Utrecht	200 000

    - Each table in the database can be considered a **class**
    - Each attribute can be considered a property and each tuple can be considered an **instance**

statista.com

Exposing Relational Databases

- A main difference between relational databases and RDF is that RDF uses URIs to identify entities, which means that everything has a globally unique identifier
  - Relational databases have identifiers that are unique only within the local scope of the given database
  - When performing a mapping one must also create URIs for each of the entities
    - Use the primary key for the URIs of each instance, AND
    - Prepend a namespace to the beginning of the attribute or table name

Conversion Tools

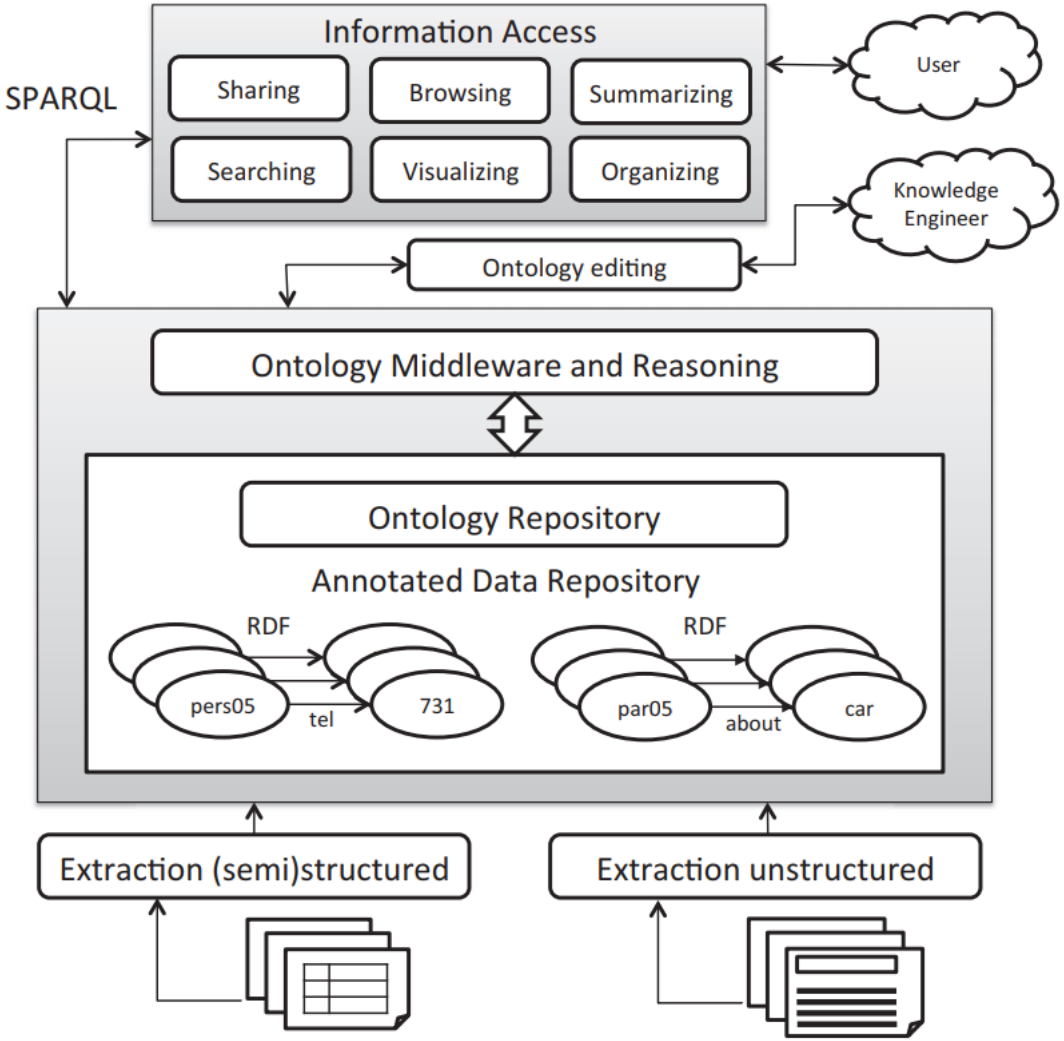
- There are several tools available, as identified by the W3C Relational Database to RDF Incubator Group
  - Most of these tools work by analyzing the structure of the relational database and then generating almost complete RDF
  - The user is then required to modify configuration files in order to specify more appropriate URIs as well as link to existing ontologies
- Conversion tools are often used in two capacities:
  - Convert in bulk a database to RDF, which can then be uploaded to a triple store, OR
  - Expose a relational database directly as a SPARQL endpoint http://d2rq.org/d2r-server

Lecture Outline

- Constructing Ontologies
- Reusing Existing Ontologies
- Semiautomatic Ontology Acquisition
- Ontology Mapping
- Exposing Relational Databases
- Semantic Web Application Architecture

Semantic Web Application Architecture

- Building the Semantic Web involves using the new languages described in this course plus ontology engineering plus service



Knowledge Acquisition

- Tools that use surface analysis techniques to obtain content from unstructured natural language documents or structured and semi-structured documents (such as databases, HTML tables, and spreadsheets)
  - For unstructured documents, the tools typically use a combination of statistical techniques and shallow natural language technology to extract key concepts from documents
  - For more structured documents, use database conversion tools
    - Induction and pattern recognition techniques can be used to extract the content from more weakly structured documents.

Knowledge Storage

- The output of the analysis tools is:
  - a set of concepts (organized in a concept hierarchy), and
  - instance data
- The repository will store both the ontology (class hierarchy, property definitions) and the instances of the ontology (specific individuals that belong to classes, pairs of individuals between which a specific property holds)
- Besides storing the knowledge produced by the extraction tools, the repository must provide the ability to retrieve this knowledge using a structured query language such as SPARQL
- RDF Schema repository will also support the RDF model theory: domain and range definitions, derivation of the transitive closure of the subClassOf relationship

Knowledge Maintenance

- A practical Semantic Web repository provides functionality for ***managing and maintaining the ontology***: change management, access and ownership rights, and transaction management
- Besides lightweight ontologies that are automatically generated from unstructured and semi-structured data, there must be support for **human engineering** of much more knowledge-intensive ontologies
  - Sophisticated editing environments can be used to retrieve ontologies from the repository, allow a knowledge engineer to manipulate them, and place them back in the repository

statista.com

Applying the Architecture

- ***Syntactic interoperability*** is achieved because all components communicate in RDF
- ***Semantic interoperability*** is achieved because all semantics are expressed using RDF Schema
- ***Physical interoperability*** is achieved because all communications between components are established using HTTP connections
- Frameworks using this architecture:
  - Drupal content management system added semantic support: http://www.drupal.com
  - Jena: http://jena.apache.org
  - Sesame: http://www.openrdf.org