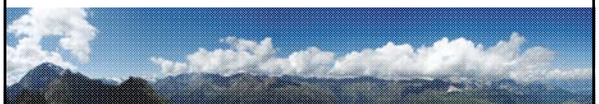
Applied Ontology Engineering

Seminar 2013 / 2014

Dr Anna Fensel

STI Innsbruck



1

Agenda

- Course overview
- Seminar organization
- Seminar topics presentation and assignments
- Tutorial: „How to Build an Ontology“
- Tutorial: „Protege and Collaborative Protege“

2

Course overview

- "An ontology is a formal representation of a shared conceptualization of a domain." Gruber, 1999.
- This seminar covers topics related to ontology engineering and the process of semantic content creation.
  - ontology building methodology
  - ontology matching
  - ontology evaluation
  - ontology learning
  - incentives for ontology building
  - technologies and tools for building ontologies
  - collaborative ontology building
  - annotations based on ontologies

3

Overall learning goal

- Get an overview of ontology engineering.
- Dive into a specific topic in detail.
- Small, hands-on ontology engineering project

.

Also:

- Presentation skills
- Working in teams
- Scientific writing

4

About the lecturer

Dr. Anna Fensel
University of Innsbruck
Technikerstr. 21a
6020 Innsbruck, Austria

e: anna.fensel@sti2.at

Office hours: only by appointment. Please contact me per e-mail at anna.fensel@sti2.at

5

Assignment

- Language: English
- Topic + state of the art survey about the topic.
- Designed ontology (whole semester)
  - Final ontology version / annotations should be published at the Open Source repositories or as Linked Data
- Paper
- Regular update and final presentations
- Grade requirements: successful participation assessment

  (a presentation, written seminar paper, and ontology)

6

Assignment

- Ontology drafts
- Introductory and further presentations
- Written Paper
  - Prepare a ca. 10-15 pages paper
  - Papers have to formatted in Springer LNCS format:
    http://www.springer.com/computer/lncs?SGWID=0-164-7-72376-0

First assignment guidelines

- Work in teams of 2
- Start to work immediately.
- Do not go to the presentation without having understood what you are presenting.
- This is a seminar where you should do scientific work.
- If you have questions / problems – do not hesitate to ask me.
- **DO NOT copy! PLAGIARISM IS NOT ACCEPTABLE!**
- **Whenever you cite somebody, cite properly.**
- **References will be part of your work!**
- When you use work of somebody else, re-phrase and summarize in your own words and **cite properly.**
- Scientific writing:
  - http://www.wit.at/events/peyton-jones/Giving%20a%20talk.pdf
  - http://www.columbia.edu/cu/biology/ug/research/paper.html

First assignment

- Form a team of 2 & select the topic – now!
- Literature pointers will be provided, but you should independently do literature research yourself.
- Once you have a list of sources, please drop me an Email to get feedback on the sources.

Administrative issues

- The course is structured as follows.
  - **Mon 07.10.2013 11.15 - 14.00 3W03**
  - **Then weekly – unless noted otherwise - till Monday, 02.12.2013**
- First session: introduction, topic assignment
- Second session: presentation of the paper and the ontology 1st version, ontology project refinement
- Intermediate sessions: ontology project evolution, submission of short papers
- Last session: ontology project assignment final submission

Topic 1: „Open Fridge“ Ontology Annotations

- Goal: Adapt and populate an ontology for an Internet of Things type of application, that collects and processes data about home appliances and devices, namely, the fridge, with the most possible complete and detailed data about the types and models of fridges.
- Think about possible data sources that are already available on the internet, and how to acquire knowledge from them (e.g. fridge specifications).
- Publish at the end the refridgerator annotations for the public use e.g. as linked open data cloud.

OpenFridge App Description – Usage Scenario

The user installs our application which integrates Plugwise system. Through our setup the system is configured so that the user can insert the data about the fridge, e.g., the type, how old it is, how many people are using it, is it stand alone or built in, …., the city, country, and then start and stop and also annotate the measurements. The local application controls when the data are sent to the back-end server and inserted into the database. It automatically stops the measurements after the two hours and notifies the user to annotate the measurements.

For example, the user runs local application to start the measurement (press "start" button). The local application opens the link for the portal where the user can annotate the measurement in advance, e.g., entering the cooling level (1,2,3,..) or temperature, or how full the fridge is, how often the doors are opened, is some hot container entered, etc.

The local application collects the data and sends the data to the back end server, which is there stored as the raw data in a data base. After the measurement, the analytics module processes the data of the measurement and created information (e.g., performance parameters) acquired from the measurements, which is then inserted, together with the annotation information, in the OWLIM knowledge base.

We will define a number of SPARQL queries which will correspond to requests such as “how efficient is my fridge as compared to average data of all fridges of the same type” … These queries shall result in answers with attractive graphical presentations.

For example the user will be able to see visualization of the fridge consumption in some comparisons, information about the cycles, presentation of the total energy used for some standardized period – e.g. extrapolated to 24 hours, dependency on the temperature, etc.

Some information from this community of refrigerators will be public, but the users whose refrigerators are in the community will have special space and capabilities.

With installing and running the app, the user sees a power profile logging experiments for his fridge in the duration of e.g. 1-2 hours. During this period a detailed profile with measurements for every second is collected. This profile shows the real consumption patterns of the refrigerator, comprising the periods of cooling, periods of inactivity, etc.

The detailed profiles offer the ground to compare: different refrigerators, refrigerators of the same type, the consumption of the same fridge, at different environmental conditions, fridge loadings, impact of opening the doors, inserting warm dishes, impact of aging, and impact of installation.

These comparisons would be of interest to end users, and, due to the collected data, also to the vendors of devices.

Topic 2: Semantic workflows

- Goal: overview existing works in semantic workflow modeling, and scenarios for their adaption, and derive the "best" model i.e. suitable to most of the typical existing scenarios.
  - e.g. in manufacturing and similar disciplines
- Outcome: potential research paper.

Topic 3: "OntoHealth" Medical Informatics Ontologies

- Goal: create an ontology that could be part of a system helping the doctors with push and pull of the relevant information for making decisions. Take a narrow domain, namely, diabetes disease.
- Pointers to start:
  - Integrating the healthcare enterprise (IHE)
  - clinical document architecture (CDA)
  - SNOMED, including its existing OWL versions
- Outcome: published ontology, maybe also the paper.

Assignments principles

- Build teams of 2-3 people
- Choose topic & let me know the group members + topic
- Download + install Protégé OWL (4.x?) (alternatively you can set up collaborative Protégé) (http://protégé.stanford.edu)
- Build ontology collaboratively
- Document well!
- Competency questions! (ALSO for evaluation)
- Instances!
- Visualize!
- Presentation incl. ontology project, online version of ontology
  - Explain your choice of tool
  - Visualize ontology (MEANINGFULLY)
  - Explain usage scenarios
  - Explain which data sources could be used for creating this part of the data set

Mandatory reading

- Read about Ontology 101, Stanford University (Google)
- Christian Bizer, Tom Heath and Tim Berners-Lee. Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems, Special Issue on Linked Data.

On ontologies

- Slides on „Ontologies" of Semantic Web lecture (see STI website).
- Fabien Gandon: Ontologies in a nutshell (http://www.slideshare.net/fabien_gandon/presentations) - more interesting presentations.
- Tom Gruber: "Ontology" in the *Encyclopedia of Database Systems*, Ling Liu and M. Tamer Özsu (Eds.), Springer-Verlag, 2009. (http://tomgruber.org/writing/ontology-definition-2007.htm)

How to build an ontology

Natalya F. Noy and Deborah L. McGuinness. ``Ontology Development 101: A Guide to Creating Your First Ontology''. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.

Step 1: Determine the domain and scope of the ontology

- What is the domain that the ontology will cover?
- For what we are going to use the ontology?
- For what types of questions the information in the ontology should provide answers?
- Who will use and maintain the ontology?

Competency Questions

- A set of queries which place demands on the underlying ontology.
- Ontology must be able to represent the questions using its terminology and the answers based on the axioms
- Ideally, in a staged manner, where consequent questions require the input from the preceeding ones.
- A rationale for each competency question should be given.

Step 2: Consider reusing existing ontologies

- Reuse ensures interoperability and reduces costs
- VOCAMP initiative
- Ontology libraries and tools for customization are required for this step
- Sub-steps
  - Discover potential reuse candidates
  - Evaluate their usability
  - Customize ontologies to be reused
  - Integrate and merge to the target ontology
- Dr. Watson (Open University, Knowledge Media Institute)
- SWOOGLE

Step 3: Enumerate important terms in the ontology

- What are the terms we would like to talk about?
- What properties do those terms have?
- What would we like to say about those terms?

Step 4: Define classes and class hierarchy

- A top-down development process starts with the definition of the most general concepts in the domain and subsequent specialization of the concepts.
- A bottom-up development process starts with the definition of the most specific classes, the leaves of the hierarchy, with subsequent grouping of these classes into more general concepts.
- Middle-out approach: define the more salient concepts first and then generalize and specialize them appropriately.

Step 4: Define classes and class hierarchy (ii)

- From the list created in Step 3, select the terms that describe objects having independent existence rather than terms that describe these objects.
  - These terms will be classes in the ontology.
- Organize the classes into a hierarchical taxonomy by asking if by being an instance of one class, the object will necessarily (i.e., by definition) be an instance of some other class.
  - *If a class A is a superclass of class B, then every instance of B is also an instance of A.*
- Classes as unary predicates—questions that have one argument. For example, “Is this object a wine?”
  - Later: binary predicates (or slots)—questions that have two arguments. For example, “Is the flavor of this object strong?” “What is the flavor of this object?”

- Concepts (classes)
- Datatype properties (string, int etc.)
  - Domain and range
  - Person hasBirthdate int
- Object properties (2 classes)
  - Person isMarried Person
- Instances
- Cardinality
- Templates

How to find classes

- Interview: talk to subject matter experts.
- Documentation: read what experts have written about the subject matter, read the requirements documentation, read proposals and invitations to tender.
- Observation and reflection.

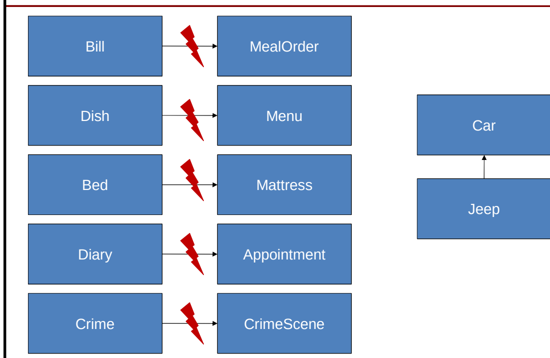
- Typical candidates for classes: NOUNS.
  - But: actors of use cases do not necessarily correspond to classes.

- Other terms as well
  - Verbs: an association which starts to take on attributes and associations of its own turns into an entity: „Officer arrests suspect".
  - Verbs: events: „Illness episode".
  - Passive form: re-formulate in active form.

Class hierarchy

- A subclass of a class represents a concept that is a "kind of" the concept that the superclass represents.
- Classes represent concepts in the domain and not the words that denote these concepts. Synonyms for the same concept do not represent different classes.
- All the siblings in the hierarchy (except for the ones at the root) must be at the same level of generality.
- If a class has only one direct subclass there may be a modeling problem or the ontology is not complete.
- If there are more than a dozen subclasses for a given class then additional intermediate categories may be necessary.
- Subclasses of a class usually (1) have additional properties that the superclass does not have, or (2) restrictions different from those of the superclass, or (3) participate in different relationships than the superclasses.

Examples



Step 5: Define attributes and relationships

- Step 4 selected classes from the list of terms we created in Step 3.
  - Most of the remaining terms are likely to be properties of these classes.
  - For each property in the list, we must determine which class it describes.
- Types of properties
  - Attributes (Data properties, OWL)
  - Relationships (Object properties, OWL)
- Properties are inherited and should be attached to the most general class in the hierarchy.

Characterizing classes

- Two types of principal characteristics:
  - Measurable properties: attributes.
  - Inter-class connections: relationships.

  - Color of an image as attribute vs. class.

How to find attributes

- Interview: talk to subject matter experts.
- Documentation: read what experts have written about the subject matter, read the requirements documentation, read proposals and invitations to tender.
- Observation and reflection.
- Nouns in „-ness"
  - Velocity-ness, job-ness, arrested-ness…
- „How much, how many" test.
  - If you evaluate this, then it is probably an attribute.
  - If you enumerate these, it is probably an entity.

Relationships

- Are defined on sets of instances.
- Properties: reflexivity, cardinality, many-to-many, all values from, some values of, transitivity, symmetry etc.
- Arity.

How to find relationships

- Interview: talk to subject matter experts.
- Documentation: read what experts have written about the subject matter, read the requirements documentation, read proposals and invitations to tender.
- Observation and reflection.
- Verbs, verbal phrases and things that could have been verbs.
  - *„The buttler murdered the duchess"*

Step 6: Define the restrictions of the properties

- Refine the semantics of the properties
  - Cardinality
  - Domain and range
    - When defining a domain or a range for a slot, find the most general classes or class that can be respectively the domain or the range for the slots .
    - Do not define a domain and range that is overly general
- Example
  - Class Dog
  - Attribute Age
  - Relationship hasOwner
  - Dog hasAge Integer
  - Dog hasOwner Person
  - Domain: Dog in both cases
  - Range: XML data type: integer; Class Person

Step 7: Create instances

- Define an individual instance of a class requires
  - choose a class
  - create an individual instance of that class
  - filling in the values of the properties

Protege and Collaborative Protege

Overview*

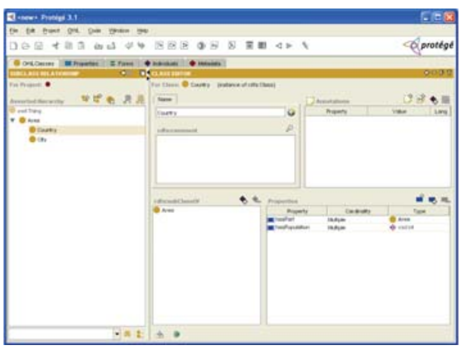
- Protege is a free, open-source platform to construct domain models and knowledge-based applications with ontologies.
- Go to http://protege.stanford.edu/doc/owl/getting-started.html to download Protege (version 4.x)

* Based on the Protege OWL Tutorial on the Protege website

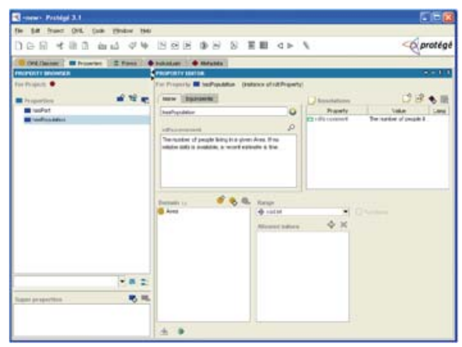
Features

- Protege supports the following ways of OE:
  - Frame-based
  - OWL
- Protege Frames editor: enables users to build and populate ontologies that are frame-based, in accordance with OKBC (Open Knowledge Base Connectivity Protocol).
  - Classes
  - Slots for properties and relationships
  - Instances for class
- Protege OWL editor:
  - Classes
  - Properties
  - Instances
  - Axioms

Interface



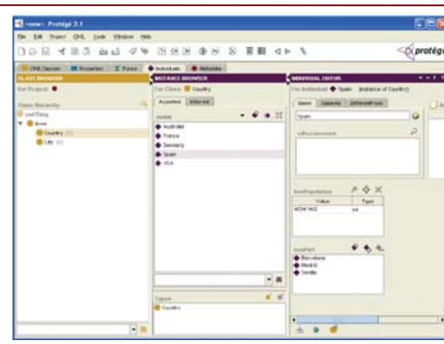
Properties tab



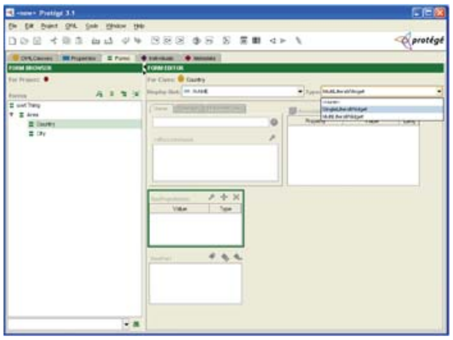
Properties

- Object properties
  - Domain: Class
  - Range: Class
  - E.g.: Dog hasOwner Person
- Datatype properties
  - Domain: Class
  - Range: Datatype
  - E.g. Dog hasBirthyear integer
- Property hierarchies

Individuals tab



Forms tab



Importing ontologies

- Re-use of ontologies
- Local or on the Web

Collaborative Protege

- Extension of Protege
- Supports:
  - Annotating ontology components and changes in the ontology
  - Dicussion threads (live chat)
  - Proposal and voting
  - Searching and filtering
  - Defining users, groups, policies
- Works with Protege OWL and Frames
- http://protege.stanford.edu/doc/collab-protege/

Collaborative Protege