Ontology Summit 2014
Big Data and Semantic Web Meet Applied Ontology

Track C: Overcoming Ontology Engineering Bottlenecks

Pascal Hitzler, Matthew West, Krzysztof Janowicz
Track Co-Champions

Mission and Scope of Track C

The mission of track C is to **identify bottlenecks that hinder the large-scale development and usage of ontologies and identify ways to overcome them**.

Bottlenecks include

- Ontology engineering processes that are time consuming,
- Social, cultural, and motivational issues
- Modeling axioms or knowledge representation language fragments that cause difficulties in terms of an increase in reasoning complexity or reducing the reusability of ontologies
- The identification of areas and applications that would most directly benefit from ontologies but have not yet considered their use and development.

Potential Solutions include

- Tools and techniques,
- Research findings and methods, guidelines, documentation, and best practice,
- Automation
- The combination of inductive and deductive methods to scale the creation of axioms
- The development of a set of reusable patterns that can ease ontology development and alignment
- The identification of purpose-driven modeling granularities that provide sufficient semantics without over-engineering
- Lessons learned from ontologies that are seeing wide adoption
- The development of tutorials and other educational materials

Report from Track C Session I (2014/02/06)

Session I title: Strategies and Building Blocks

Speakers:

Prof. Werner Kuhn (University of California, Santa Barbara)
"Abstracting behavior in ontology engineering"

Prof. Aldo Gangemi (University Paris 13 and ISTC-CNR Rome)
"Knowledge Patterns as one means to overcome ontology design bottlenecks"

Mr. Karl Hammar (Jönköping University)
"Reasoning Performance Indicators for Ontology Design Patterns"

Track C Session I (2014/02/06)

Bottleneck focus of session I:

- Modeling axioms or knowledge representation language fragments that cause difficulties in terms of an increase in reasoning complexity or reducing the reusability of ontologies

Potential solutions focus of session I:

- The development of a set of reusable patterns that can ease ontology development and alignment
- The identification of purpose-driven modeling granularities that provide sufficient semantics without over-engineering

Some questions we wanted to address

- How do we arrive at reusable patterns?
- How many patterns are there?
- Are there types of patterns?
- Are all patterns domain-independent?
- Can we mine patterns from data?
- Who will develop and maintain these patterns?
- Are there measures or at least experience reports on the robustness and usefulness of patterns?
- Are there success stories of large-scale pattern usage?
- How to abstract from individual ontology designs?
- Do we need higher-level ontology modeling languages on top of knowledge representation languages?
- How to get community buy-in?
- How important is the selection of specific language constructs for the scalability and reuse of patterns?

Important findings from the talks

- A standardized and accepted knowledge representation language such as OWL does not necessarily replace the need for a knowledge modeling language (see Kuhn's talk)
- Behavioral abstraction (e.g., duck typing) may be one approach to support the development of more robust ontologies (Kuhn)
- Entity-centric, frame-oriented data science is required to ensure relevance of SW technologies and ontologies (Gangemi)
- Need for improved data-driven techniques to scale the development of patterns and ontologies without loosing reference frames (Gangemi)
- The usage of specific KR language constructs has direct consequences for reasoning complexity, tool support (e.g., CGI), and reusability (Hammer)

Some important findings from the chat

- There are an unlimited number of patterns
- We can mine patterns from data
- True patterns will mostly be discovered, rather than invented
- When you abstract patterns from ontology designs you are usually moving up the subtype/supertype hierarchy rather than moving out class-instance, so you should not normally need another language.
- Buy in comes from utility plus ease of availability and use.
- It is first of all important that the language constructs can support the requirements of the application, otherwise all is lost. Generating more efficient language forms from more understandable forms may be a way forward.

Some important findings from the Summit List discussion

What is it that takes a lot of time and effort?

- Education and team buy-in takes a lot of time.
- There are 2 tasks that are rather time-consuming;
  - the extraction of the knowledge from Subject-Matter experts, and
  - the explanation of the model to developers using it.

What is it that is very expensive?

- Refining the ontology during development to satisfy logical consistency
- The extraction of the knowledge is expensive

What is it that is held up because of a lack of scarce resources?

- We need new and better ways to discover, express and process ontologies.

Why is it that ontological approaches are not taken when they could/should be?

- Time constraint on the delivery of the ontological artifacts mean that the model and its implementation are generally not separated.
- Current ontological approaches are too primitive.

Ontology Engineering Bottlenecks – Session II

Oscar Corcho (Universidad Politecnica de Madrid)

10 basic rules to overcome ontology engineering deadlocks in collaborative ontology engineering tasks

Dhaval Thakker (University of Leeds)

Modeling Cultural Variations in Interpersonal Communication for Augmenting User Generated Content

Peter Haase (Fluid Operations)

Developing Semantic Applications with the Information Workbench – Aspects of Ontology Engineering

Some Key Problem Areas

- Ontologies are perceived as costly
- There is confusion over the level of expressiveness needed
- Who will develop the shared ontologies?
- How do we do quality control?
- What level of semantics is needed?
- What tools to use?
- How to reuse successfully?
- Why are ontologies in English?

Ontologies are perceived as costly

- Ontologies may be hard to develop, but taken as a proportion of the overall project (at a business improvement level) they are a part of, they are generally a relatively small proportion of the total cost
- Timely and appropriate ontology development will reduce overall project costs, whereas no (implicit) ontology or late development of ontologies will lead to higher overall project costs
- You need to be able to make this case

There is confusion over the level of expressiveness needed

Different applications will require different levels of expressiveness:

- Expressiveness is important for descriptive ontologies, where the queries are not known at design time
- When you do know the queries to be answered, it is often possible to construct a more restrictive ontology that will answer those queries with improved performance
- You may need multiple ontologies (or a master and subsets) to meet all needs in a domain

Who will develop the shared ontologies?

- Many ontologies will be private – some ontologies will be public
  - Accounts – private
  - Product Catalogue – public
  - Public administration data, and standards - public
- The authoritative source should develop the shared ontology
  - Avoids replication
  - Sources need to be aware of obligation
- Examples
  - Governments should develop ontologies related to their laws
  - Standards bodies should develop ontologies
- Alternatively a multitude of individual, non-authoritative ontologies can be integrated via ontology-alignment.

How do we do quality control?

- Test, test, test
- It is just the same as software development quality control and data quality control
- Inferencing tools can help with logical consistency, but there are many more errors that can be made beyond logical consistency

What level of semantics is needed?

- *Identity* (same name – same thing) has a high priority
- Level of semantics required varies from application to application and domain to domain.
- Examples
  - Engineering : quite a lot.
  - Geosciences (due to their diversity): quite a lot.
  - Life sciences : medium.
  - Publishing : very little as they focus on vocabularies

What tools to use?

- It is sensible to start with lightweight tools like Excel
- You cannot manage large/complex ontologies with lightweight tools

How to reuse successfully?

- Select for re-use
  - Reuse is generally not an objective in itself
  - Determine requirements before reviewing candidate ontologies to reuse
  - Reuse is successful when it reduces costs and increases quality
- Design for re-use
  - Reuse is unlikely to be achieved unless it is a design objective
  - Common errors are:
    - The ontology is over-constrained
    - Range and domain are set at too low a level of abstraction – the highest level in a particular domain, rather than across domains
    - Local rand and domain constraints may sometimes be more suitable

Why are ontologies in English?

- Many ontologies intended for reuse are designed in English and it is assumed all users will use English – this is not valid
- It is pragmatic that IDs should be in the language of the developer, since this helps the development and debugging process
- IDs should be hidden from end users, who should be able to choose the language for the labels they see

Reflections

- Bottlenecks and barriers to the use of ontologies in Big Data and the Semantic Web are many and various
- Alignment and reuse of ontologies and ontology patterns offers promise in overcoming development bottlenecks, but comes with its own bottlenecks and barriers
- Automation of tedious and repetitive tasks is demonstrated to be effective, but there is a need for more tools that deliver this automation