Ontology Summit 2014
Big Data and Semantic Web Meet Applied Ontology

Track C: Overcoming Ontology Engineering Bottlenecks Synthesis-I

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 reuseability 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"

Report from 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 reuseability 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

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

Questions we wanted to address during session I:

- How to 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?

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

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** 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)

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

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.

Report from Track C List Discussion

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.