Design Insights for the Next Wave Ontology Authoring Tools

Markel Vigo
University of Manchester
School of Computer Science
markel.vigo@manchester.ac.uk

Caroline Jay
University of Manchester
School of Computer Science
caroline.jay@manchester.ac.uk

Robert Stevens
University of Manchester
School of Computer Science
robert.stevens@manchester.ac.uk

ABSTRACT
Ontologies have been employed across scientific and business domains for some time, and the proliferation of linked data means the number and range of potential authors is set to increase significantly. Ontologies using the Web Ontology Language (OWL) are complex artefacts, however: the authoring process requires not only knowledge of the application domain, but also skills in programming and logics. To date, there has been no systematic attempt to understand the effectiveness of existing tools, or explore what users really require to build successful ontologies. Here we address this shortfall, presenting insights from an interview study with 15 ontology authors. We identify the problems reported by authors, and the strategies they employ to solve them. We map the data to a set of design recommendations, which describe how tools of the future can support ontology authoring. A key challenge is dealing with information overload: improving the user's ability to navigate, populate and debug large ontologies will revolutionise the engineering process, and open ontology authoring up to a new generation of users.

Author Keywords
Ontologies; semantic web; authoring tools;

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous; I.2.4 Knowledge Representation Formalisms and Methods: Semantic networks

INTRODUCTION
The use of ontologies to capture the knowledge or semantics of a field of interest has grown over the past two decades as a consequence of the needs of applications to query, integrate and analyse data on the basis of the nature of the entities being described in those data. Ontologies have been widely used in the biomedical and healthcare domains, as well as many other areas of science, and commerce. The production of linked data has also led to a widespread use of ontologies to describe its entities and their relationships.

An ontology attempts to represent knowledge of the entities in a field of interest. Entities are typically described as classes of individuals, and the relationships between those individuals. For example, we may describe the class of all Person as a set of objects all of which has a Sex, a Father and a Mother, with the latter two entities also being classes of individuals of class Person that have a specified Sex and are parents of a Person. These classes of objects may be related via relationships such as hasSex, hasFather and hasMother.

Many ontologies are represented using the Web Ontology Language (OWL) [4]. OWL has its roots in description logics, a decidable fragment of first order logic. An ontology written in OWL is a collection of axioms describing the classes, individuals and the relationships between them. A consequence of OWL's strict semantics is that an OWL ontology can be subjected to automated reasoning. Thus an OWL ontology can be a complex artefact, with thousands of axioms from which implications can be deduced. The strict semantics, potentially numerous axioms – sometimes in complex patterns – from which deductions can be drawn means that an OWL ontology is difficult both to build and understand.

There are a range of tools that support the ontology authoring lifecycle. Typically, these tools assist authors in carrying out particular tasks including ontology building, reasoning or debugging. In such a fragmented landscape, the few tools that aim to support the chore processes of ontology authoring are consequently extremely popular, Protégé being a notable example [2]. Considering the potential complexity of ontologies, little is known about how well the mentioned tools help ontology authors to accomplish their tasks.

Previous work on ontology authoring dynamics addresses the socio-technical issues of distributed collaborative authoring, highlighting that reaching consensus on the use, purpose and scope of a given ontology may generate tensions among the authors [3]. In contrast to collaborative ontology building, axiom addition is carried out individually. To gain a better understanding of the individual ontology authoring process and how tools support it, we interviewed 15 ontology authors about the problems they encounter and the strategies they employ. Our results suggest that the next generation of ontology authoring tools should provide support to navigate and populate large ontologies, be able to reason on-the-fly and aid the debugging process by identifying the most relevant problems.

STUDY

Method
Ontology authors were recruited through convenience/snowball sampling. Participants worked in a wide variety of disciplines: genetics, anatomy, archaeology,

CHI 2014, Apr 26–May 1, Toronto, ON, Canada
ACM 978-1-4503-2473-1/14/04.
http://dx.doi.org/10.1145/2556288.2557284

amino acids and the food industry. The semi-structured interviews, which were conducted either face-to-face or via video-conferencing, followed this script:

– Can you describe the authoring tasks you perform?
– How do you use tools to support you in these tasks?
– What sort of problems do you encounter?

The mean interview time was 41 minutes and 31 seconds (sd = 15 minutes, 20 seconds). The interviews were recorded and transcribed.

Participants

We interviewed 15 ontology authors (11 female participants) with an mean age of 37 years (sd = 7). Thirteen of these were based in the UK and 2 were based in the US. Nine were computer scientists, 4 were biologists and the remaining 2 had a background in science. All of them had more than 4 years of experience working with ontologies. Participants were awarded with a £10 book store gift certificate.

The participants' interaction with ontologies can be classified into 3 different groups (5 participants in each): ontology researchers (R), curators (C) and ontology developers (D). Ontology researchers are typically computer scientists investigating the properties of ontologies, building test ontologies or building reasoning engines. Curators are individuals with a deep knowledge about a domain and maintain large ontologies (a magnitude of 100,000 axioms). They receive requests from ticketing systems or a consortia to add terms, update entities or add branches into existing ontologies. Ontology developers are computer scientists that work very closely with domain experts in order to model and validate a specific domain. The ontologies they build are normally used by applications such as search engines or web applications.

The tools participants use are: Protégé (14 users), OWL API (6), OBO-Edit (4), Bioportal (3), AmiGO (1), ArrayExpress (1), Corona (1), Ontodog (1), Ontofox (1), Ontogene (1), OLS (1), QuickGO (1), Topbraid (1) and Zooma (1).

Analysis

The transcriptions of the interviews were uploaded to the qualitative data analysis software, Dedoose 4.5. Transcripts were thematically analysed in an open coding fashion following established analysis methods [1]: (1) familiarising with data, (2) generating the initial codes, (3) searching for themes and (4) iteratively reviewing themes. The generated codebook was agreed between the authors. Then, an independent coder was given the codebook and transcripts in order to establish coding reliability.

RESULTS

The principal emerging themes that were identified are the tasks ontology authors typically tackle, the problems they encounter and the strategies they employ to overcome such difficulties. The inter-coder reliability analysis yielded a Cohen's $\kappa = 0.61$, which is considered a substantial agreement.

Sensemaking, exploration and searching

Sensemaking problems relate to the loss of contextual awareness and to the difficulties grasping the current state of the tool and the ontology: "*You just lose the overview. If you have a small ontology I know every single axiom I put in there, I can remember it, whereas with big ontologies I wouldn't be able to remember every single existential restriction.*" (P4, R).

Exploration problems describe the difficulties encountered when navigating and traversing the ontology: "*We have classes and instances about clinical statements; moving from instances to classes is not so good because instances appear as a plain list [. . .] As the number increases it is more difficult to track back which instance belongs to which class.*" (P7, D).

Search problems are about unsatisfactory searching capabilities of tools or because tools cannot search in multiple ontologies. The latter is crucial because before adding new terms to ontologies, authors check the existence of similar terms in other ontologies in order to reuse them. "*I forgot what the primary label was; then it is very difficult because you cannot search on synonyms or other metadata.*" (P14, C).

Strategies (15 instances)

Authors get an overview from looking at the top element of the hierarchy, especially if authors are not familiar with the domain: "*The first thing would be to start from the top level of genes and just investigate the hierarchy and try to learn about it.*" (P1, R).

If the author is familiar with the domain and the ontology, finding one's way through the ontology is done by navigating the class hierarchy, which is shaped as an expandable tree: "*If you edit an ontology full-time you get very familiar with it [. . .] all of us can drill-down the ontology and find it quite easily.*" (P12, C).

Resorting to domain specific information retrieval engines allows authors to obtain information about terms and finding whether a given term exists in another ontology. "*I would look at the ontology directly in Bioportal to see whether a term is in the hierarchy and also to figure out what it means.*" (P10, C).

Ontology building

One of the foremost problems encountered by authors is efficiently building and populating an ontology with many entities "*If you want an ontology with 100-1000 nodes Protégé is useful for the higher levels and abstract classes.*" (P7, D).

Ontologies do not necessarily have to be shaped as trees. In theory, ontologies are a set of axioms, which is a view mostly supported by computer scientists and logicians: "*An ontology is a set of axioms, end of story.*" (P2, R). We call the particular style of building ontologies as a set of axioms and letting the reasoner build the relationships between nodes definition-oriented ontology development. However, existing tools do not support definition-oriented ontologies well because the interfaces drive authors to build hierarchies by hand. "*I would say that this interface makes us make use of the hierarchy feature the main structuring block for classes.*" (P5, R). This has a number of inconveniences including maintenance problems. "*Definition-oriented development is better for main-*

tenance because the hierarchies are not anymore subject to human decision [. . .] you don't have to decide upfront where you want to locate your thing." (P5, R).

Strategies (17 instances)
Reusing existing ontologies is embedded into the ethos of the semantic web community. Consequently and in theory, only a few ontologies have to be built from scratch. "*We only add new content to the ontology if no other ontology contains that.*" (P6, C). To reuse and import existing terms, authors try to match or map their terms with those described in other ontologies (which may use a different term to refer to the same concept). Some authors resort to Bioportal, which can be used as an information retrieval system that searches over the annotations and metadata of some ontologies on the life sciences domain. "*There is so far no available tool but we do use some ontology mapping tool which is Bioportal.*" (P11, D). Those who are computer scientists use programming libraries such as the OWL API to efficiently populate ontologies. "*If it is 2 classes I do it manually, if it is 10 classes in Protégé and if it is 100 I do it programmatically.*" (P7, D). On the other hand, curators and those who work closely with domain experts have template based tools that are given to domain experts to fill out. These templates are then automatically processed and their content is placed into ontologies. "*We had biologists filling out the templates, then we instantiate lots of axioms with these templates.*" (P6, C).

Reasoning
The size of the ontology and the complexity of the axioms cause problems when running the reasoner. "*I don't like to work with ontologies that are so big or complicated that I can't reason reasonably quickly.*" (P15, C).

Strategies
The following strategies are often used to speed-up the otherwise lengthy reasoning process.

- Externalising reasoning by employing continuous integration (3 instances). "*Reasoning is so memory intensive we have to do all the reasoning separately.*" (P12, C).
- Complex entities are taken out of the ontology and reasoning is done separately (3 instances). "*The disjoints are kept in a separate file and they are all combined in this set of checks that run every time*" (P12, C).
- Using low expressivity languages has some positive consequences (2 instances): these tend to be more human readable, seen as less error prone and faster to reason.

Debugging
Debugging is one of the most challenging tasks due to the lack of relevant information and appropriate instructions to fix problems. "*If the ontology is big enough, it is very difficult to work out why particular axioms are not working.*" (P8, D).

Strategies
In a debugging scenario, ontology authors exhibit particularly elaborate strategies.

- Injecting test axioms or using description logic (DL) queries (3 instances). "*You build the test classes so you can break down a class, you use a DL query to check which one stops working.*" (P6, C).
- Frequent reasoning not to accumulate problems over time (8 instances). "*Every axiom change, every class addition, every refactor I always run the reasoner.*" (P8, D).
- Fixing first those entities which are more prone to propagate problems (2 instances). If these are removed high number of problems disappear. "*Go to most general inconsistent concept and try to figure out most common concepts of inconsistencies.*" (P9, D).
- Users try to fix the ontology by making changes in a trial and error fashion, almost randomly (3 instances). "*It's a little bit trial and error, we don't really have a methodology.*" (P9, D).

Evaluation
In the context of ontology authoring, evaluation entails checking if the ontology does what is expected. This quality assurance stage is poorly supported either by tools or processes "*We don't have a formal mechanism for checking. We rely on the fact that the ontology is being used.*" (P14, C). Only those who work in larger teams where the ontology authoring process is carried out in a more systematic way employ evaluation procedures.

Strategies (7 instances)
Sitting next to experts and using domain specific databases are the main strategies employed for evaluation purposes. "*It's reasonable to have experts to confirm what is correct or what is wrong.*" (P1, R). Natural language statements about what is expected from an ontology are also formalised as competency questions.

Resorting to Protégé
Protégé is intended to be a tool that gives support at every step of the ontology authoring process. Conversely, we found that a number of users employ Protégé only in particular situations and for very specific tasks:

- To convert and classify non-OWL ontologies (5 instances). "*I transform my document into OWL and then I will load it with Protégé and classify it.*" (P6, C).
- To have an overview and visualise the effect of actions (7 instances). "*If I need to check a small piece of code that I have in mind I just pull up Protégé and I quickly do it.*" (P13, D).
- To show ontologies to domain experts (2 instances). "*We show the ontology to the clinicians. If you give them the whole ontology that's not really useful for them.*" (P11, D).

DISCUSSION
Results indicate that tools aiming to support the chore processes of ontology building (e.g. Protégé) fail to do so. These tools are primarily used to build small ontologies and to visualise medium-sized ontologies that have been built using other means. Assistance for populating, reasoning, debugging and evaluation is limited and not sufficiently supported by tools.

From the strategies users employ when carrying out the authoring tasks we derive a number of design insights. Following these will enable tools to better support the ontology authoring process.

Sensemaking, exploration and search

Provide situational awareness

Existing tools do not clearly show the consequences of the actions taken (i.e. the *undo* command). When the modification of a single axiom can affect the underlying logic and structure of an entire ontology that contains hundreds of thousands of classes, support for situational awareness is crucial. This may be obtained through textual and visual feedback: the former would inform about the problems that may have arisen (background reasoning would be necessary for this), while the latter can highlight the updated elements.

Overview, filtering and linking features

Navigating ontologies is prone to information overload and disorientation. This is especially true of the larger and the heavier axiomatised ontologies. Implementing fundamental principles of information visualisation by providing overviews and filters can alleviate this problem. Overviews may help those who are not familiar with the domain, whereas filters will speed up the navigation of domain experts. Considering how intertwined the elements of an ontology are, hyperlinking the class hierarchy, entities and axioms may help authors to use the information scent in links to traverse the ontology.

Ontology building

Support for ontology retrieval and reuse

Extending the search capabilities of tools to well known ontology search engines (e.g. Bioportal), established terminologies (e.g. the Gene Ontology) or manually aggregated sources would allow authors not only to retrieve those resources that match a query, but also to select and incorporate these into the working ontology. Support for dealing with discrepancies between the working ontology and existing ones should provide mechanisms for mapping, aligning and merging ontologies.

Efficient ontology population methods

Current GUI tools fail to support the massive number of additions and modifications that need to be done in large ontologies, which calls for efficient population methods. Templates, which are mostly used by curators, are especially relevant if domain experts are involved in the building process.

Support for definition driven ontologies

Whilst definition-driven ontologies have several advantages over hand-crafted deep trees, current tools drive authors to build the latter type of ontologies and do not support the former. Tools should cater for different building styles without imposing one in particular: for those who are not familiar with description logics, tree structured ontologies may be easier to understand.

Debugging and evaluation

On-the-fly reasoning

The complexity of ontologies leads some authors to invoke the reasoner at every modification they make or when complex axioms are added. Some adopt this strategy in order to avoid the propagation of problems. Considering how demanding reasoning is, tools and reasoners should allow authors to reason over subsets of ontologies and axioms to speed up the process. Ideally, authoring tools should provide background reasoning capabilities, as do the compilers of software development IDEs.

Error filtering

Debugging is often dealt with by identifying first the most relevant problems among highly overloaded explanations. It should be relatively straightforward for tools to identify these problems, and removing these initially will minimise the number of explanations.

Incorporating unit tests

Having pre-built axioms to test the ontology helps to unearth problems. When these axioms are shaped as competency questions the authors can evaluate whether the ontology meets the initial requirements in terms of coverage and capabilities.

CONCLUSION

We provide design recommendations to ease the authoring process of large logical artefacts that are potentially complex and are becoming more pervasive, i.e. ontologies. The possibilities of ontology use are opening up to a wide range of people, including those who may never have encountered this form of knowledge representation before. A consequence of this is that ontology authoring will move from being a specialist enterprise that is the preserve of experts, to being a tackled by mainstream – even amateur – developers. Adopting these design insights will help to ensure this evolution in ontology development realises its full potential.

ACKNOWLEDGMENTS

Research funded by the EPSRC project: *WhatIf: Answering "What if..." Questions for Ontology Authoring*. EPSRC reference EP/J014176/1.

The authors would like to thank Samantha Bail for her help in analysing the data.

REFERENCES

1. Braun, V., and Clarke, V. Using thematic analysis in psychology. *Qualitative Research in Psychology 3*, 2 (2006), 77–101.
2. Cardoso, J. The semantic web vision: Where are we? *IEEE Intelligent Systems 22*, 5 (2007), 84–88.
3. Randall, D., Procter, R., Lin, Y., Poschen, M., Sharrock, W., and Stevens, R. Distributed ontology building as practical work. *International Journal of Human-Computer Studies 69*, 4 (2011), 220 – 233.
4. W3C OWL Working Group. OWL 2 Web Ontology Language Document Overview, 2012. `http://www.w3.org/TR/owl2-overview/`.