Ontology Authoring Inspired By Dialogue*

Artemis Parvizi[1], Yuan Ren[1], Markel Vigo[2], Kees van Deemter[1], Chris Mellish[1], Jeff Z. Pan[1], Robert Stevens[2] and Caroline Jay[2]

University of Aberdeen
[1]{a.parvizi, y.ren,k.vdeemter,c.mellish,jeff.z.pan}@abdn.ac.uk
University of Manchester
[2]{markel.vigo,robert.stevens,caroline.jay}@manchester.ac.uk

Abstract

This paper introduces a dialogue-based ontology authoring interface. The aim of this interface is to simplify the ontology authoring process for the users. The design of the interface is based on the insights that have emerged from research into human language and explorations of authoring activities in Protégé. We will discuss our initial findings regarding ontology authoring patterns and how we aim at modelling user's goals and intentions. We will also discuss the challenges arising whilst generating interesting and comprehensible feedback.

1 Introduction

The idea that human-computer interfaces can be seen as supporting a *dialogue* between a user and a system is well established; its origins are difficult to trace with certainty.[1] This idea has particular appeal in the case of knowledge authoring, as when a human author uses an interface such as Protégé to construct or modify a formal ontology. In knowledge authoring, the information conveyed by both the system and the user consists of potentially complex propositions instead of only button pushes or selections of an item from a menu, for example; this complexity makes the metaphor of a dialogue apt.

The present paper is an intermediate report on some parts of a project that takes the analogy between ontology authoring (OA) and human dialogue seriously by making use of insights that have emerged from research into human language. The aim is that by doing this, we will ultimately make knowledge authoring interfaces more effective, because they will offer users a better understanding of the ontologies that they produce during their interaction.

We start this exploration with an investigation of the dialogue patterns between an ontology author and their ontology in Protégé. By recording all actions at the Protégé user interface, along with the use of eye-tracking, we have a dataset that can be explored using techniques such as N-gram analysis. From this we obtain a set of typical patterns of authoring activity that are 'dialogues' between ontology author and Protégé. Based on the initial explorations of authoring activities with Protégé, we analysed the existing speech acts in Protégé and drafted a manual highlighting the potential speech acts for a dialogue-based ontology authoring interface. Section 3 will discuss this interface.

From this base, we then discuss the problem that a knowledge editing tool faces when it has to feed back to a user the effects of a knowledge editing action (e.g., the addition of an axiom to the ontology) in terms of the propositions entailed by the ontology (Section 4). We argue that this problem bears important similarities to what a Natural Language Generation system does when it selects, orders, and aggregates information for presentation to a reader, and this suggests ways in which the "entailment selection" problem in knowledge authoring might be solved.

*This work was supported by EPSRC grants EP/J014354/1 and EP/J014176/1.

[1]See e.g. various contributions in The Structure of Multimodal Dialogue, Volumes 1 and 2.

Natural language dialogues also allow the participants to discuss their current goals. In the context of ontology authoring, we interpret these as the natural language Competency Questions (CQs) that are used by many ontology authors. As questions, CQs are conversationally appropriate only when certain presuppositions (Beaver, 1997) in the CQ are satisfied by the ontology. For example, for the CQ "Which software implements some algorithm?" to be appropriate, in the ontology some software should be allowed to implement an algorithm, and some software should also be allowed to not implement an algorithm. Otherwise the answers will always be "None of them" or "Every one of them", no matter how software and algorithms are described. We analyse the patterns of CQs in practice, from which authoring tests corresponding to the presuppositions can be automatically deduced. The status of the tests (i.e., whether a given test is true or false given the ontology) can be fed back to the user in a manner that we expect to be helpful.

2 Ontology Authoring Patterns in Protégé

In order to support ontology authoring as a dialogue, we need an understanding of what speech acts are appropriate in this activity. To anchor this in reality we are carrying out an analysis of how people use an existing ontology authoring tool, even though this presents something rather different from natural language dialogue. The hope is that, at least at an abstract level, common activity patterns observed in current practice will give us ideas about functionalities to be supported by our speech acts.

The activity patterns in the ontology authoring process signal the existence of common ways of addressing the authoring tasks. These patterns are of importance as they can potentially be mapped into recommendations for the design of a user interface, for the inclusion of new functionalities or to articulate innovative interaction modalities such as the speech acts in Section 3.

In order to seek these activity patterns we instrumented Protégé, which is the preferred tool of 74% of the ontology developers according to a recent survey (Warren, 2013). Our instrumented Protégé logs the events triggered by users when authoring ontologies. These events can be classified as interaction events (e.g. expanding the class hierarchy), authoring events (e.g. add a new class) and environment events (e.g. load an ontology). Sixteen participants carried out 3 authoring tasks with the instrumented version of Protégé. Additionally we used an eye-tracker to identify how attention was allocated to different areas of Protégé. The log files collected contained a mean of $\sim$7K rows of events, which accounted for 45 minutes of interaction.

In analysing these log files, we first removed those events that were not meaningful: i.e. the invocation of the reasoner generates several events that can be summarised into one, namely *reasoner invoked*. Second, those events that were infrequent and had been triggered anecdotally were removed (i.e. use of the search functionality). Third, we carried out an analysis of N-grams in order to find the patterns in these long sequences of events. We discovered that the most frequent N-grams were those repeating the same event: i.e. *entity select*, *entity select*, *entity select* .... To more readily reveal authoring patterns, we merged all the N-grams containing the same events. In the above example the merging led to new multiple events such as *multiple entity select*. After the merging, the higher-level activities include:

- The **exploration** activity describes how users navigate an unfamiliar ontology, and the different exploration patterns in the asserted and the inferred hierarchy. While in the former users try to find a specific location in the hierarchy in order to add or modify a class, in the latter the behaviour is more of an exploratory nature and is often related to checking the consequences of running the reasoner.
- The **editing** activity indicates how properties are added to classes and how restrictions are established on them by selecting an entity in the class hierarchy, looking at the description window and invoking the entity modification window.
- Often, saving the ontology initiates the **reasoning** activity, which is followed by the exploration of the inferred hierarchy to ascertain how the hierarchy has been updated by the automated reasoner.

The method we are using for this activity is essentially the same as using corpus analysis to inform

the design of a natural language dialogue system, the only difference being that the low level dialogue moves being observed are in this case button clicks and keyboard events, rather than natural language utterances.

3 A Dialogue Interface for Ontology Authoring

As a step towards the construction of a dialogue-based interface, the WHATIF gadget, we have drafted an authoring manual, a set of speech acts for Protégé, and a dialogue manual containing dialogue speech acts[2] (Parvizi et al., 2013). Based on these manuals and Nielsen's (Nielsen, 1986) proposed model of human computer interaction[3], we have developed a prototype of a dialogue system. In the latest version of the prototype, speech acts *a*) checking, *b*) observation, *c*) axiom addition, *d*) modelling element addition[4], *e*) axiom or modelling element deletion, *f*) axiom modification, and *g*) WHATIF question have been implemented. The system presents this list of speech acts to the users and requires them to select one. In addition, deletion and modification speech acts, the normal process of ontology authoring is carried out; the *checking* speech act will inform the user of the presence or absence of a specific modelling element or an axiom; the *observation* speech act provides a detailed account of the characteristic of a specific modelling element or an axiom; and importantly, the WHATIF speech act in which the user can ask a question about the logical consequences of an authoring action, and the system will perform a look-ahead search and provide a description of changes and future implications.

Based on the analysis done in Section 2, we can map *(1)* observation to exploration, *(2)* checking to a combination of reasoning and exploration *(3)* adding, deleting, and modifying to editing, and *(4)* WHATIF to a combination of reasoning and exploration activities.

The WHATIF gadget receives users' requests in Manchester syntax or OSE along with a speech act; the command is parsed and, based on the speech act, an appropriate response is generated. The next step is to use the coherence and relevance between various commands to provide an unambiguous and clear feedback or occasionally summarise the generated responses (see Section 4). We also aim at simplifying the ontology authoring process by understanding users' goals and intention (see Section 5).
Users interact with the system through the following panels:

Class hierarchy: panel displays the *inferred* hierarchy to the users. This hierarchy is updated after each editing task (adding, removing, and modifying), and the user can always view the current state of the ontology.

Class description: this panel will be activated by clicking on any class in the class hierarchy. The panel will display the characteristics of the selected class in natural language. This panel essentially works as a verbaliser.

Input panel: this panel allows the users to enter their commands either in Manchester Syntax or in OSE. The interaction is governed by the speech acts mentioned above.

Task list: this panel contains the goals and intentions of the users, formulated as a set of competency questions (CQ). Based on each CQ, the system will generate a set of appropriate authoring tests (AT) (see Section 5). The state of each AT, pass or fail, is known by the colour of the icon, green or red.

History panel: this panel records the entire interaction between the user and the system. All the system's written feedback will appear in this panel. Also, the written status of the CQs will appear in this panel. The biggest issue is the length of the feedback provided, which will be discussed in Section 4.

[2]The dialogue manual was generated from an examination of existing dialogue systems and their speech acts, and a manual exploration of Protégé.

[3]Nielsen's interaction model is a dual model of the user and the computer at the same time. In this model, the interaction between the two participants, human and computer, has been viewed as a dialogue. However, unlike a human-human interaction, the two participants do not have the same communication skills. The user model has to be psychologically valid, whilst the system model has to follow some design principles.

[4]The users need to define the modelling elements (i.e classes and individuals) before they can add an axiom to the ontology.

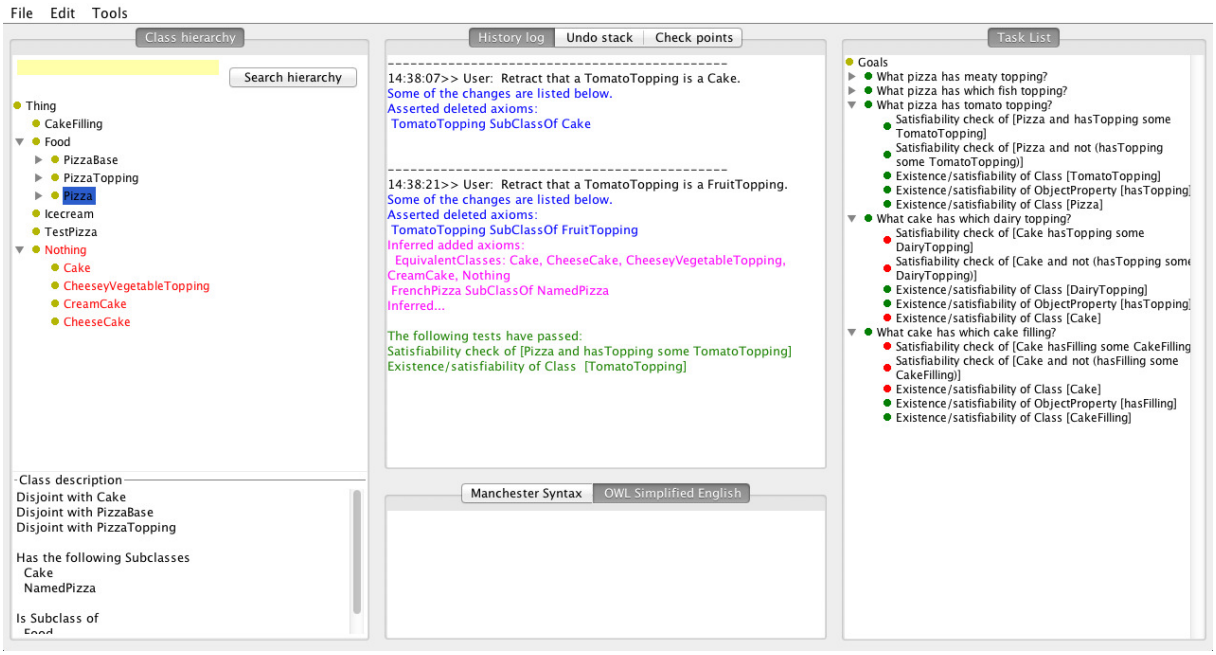

Figure 1: The WHATIF gadget

4 Giving Feedback

A real dialogue, rather than a monologue, has significant contributions from both participants. Given the knowledge-rich nature of ontology authoring, there is an opportunity to go beyond the user-driven monologues of most user interfaces by having the system provide *feedback*. In particular, an ontology author needs to be made aware of the consequences of their authoring actions. If the author has just added or deleted an axiom from the ontology, the system should tell them about the new *entailments* (the new propositions that follow from the revised axioms, and also those propositions that are no longer entailed). In some families of OWL, this set of entailments can be *infinite*, and in others large. At this step, most of the ontology authoring tools or reasoners, based on some predefined criteria, decide to show only a *subset* of these entailments. We have categorised entailment selection in previous work into the following categories:

Syntax-driven: selection is based on the explicit structure of the entailments. For instance, a syntactic approach might select axioms of the form $A \sqsubseteq B$ where A is a named concept. Such an approach is quite common in traditional ontology authoring interfaces such as Protégé which displays information based on the structural forms that the user selects. In the Rabbit to OWL Ontology authoring (ROO) editor (Denaux et al., 2012) the existence of an infinite entailment set has been acknowledged and is tackled by including only new and lost entailments of the form $A \sqsubseteq B$, $\top \sqsubseteq B$, $A \sqsubseteq \top$ and $A(a)$, where A and B are concept expressions that appear in some axiom and a is a named individual. This is probably the least restrictive syntactic approach in current use.

Logic-driven: selection is based on logical behaviour or preference (Siberski et al., 2006) of entailments. Such an approach can only make choices between axioms which are logically distinct. For example, selecting the *most specific* entailments (Mellish and Pan, 2008).

Mellish and Pan (2008) considered various logical principles based on the Gricean maxims (Grice, 1970) of cooperative conversation, places where a reader is likely to draw false implicatures where words such as “all” and “not” are used. This is beginning to go beyond a purely logical problem to a problem where the mode of presentation (here, natural language) also plays a role. That is, the maxims only apply if the interaction is regarded as a cooperative conversation of the kind conducted by people. Given our context of authoring inspired by natural dialogue, this extra assumption is especially relevant. Indeed, it suggests that there may also be useful criteria of the following kinds:

Discourse-driven: selection is based on the structure of the dialogue. Instead of viewing each interaction in isolation, we can analyse a *sequence of utterances*, and based on some criteria such as the Gricean maxim of relevance or the user's previous focus of attention, select a subset of entailments. Unlike ours, most ontology editors do not have an explicit linear discourse within which relevance and coherence arise. However, verbalisers often focus on grouping and ordering of textual material generated from ontologies. It might be possible to transform these grouping and ordering strategies into selection strategies.

Pragmatics-driven: selection is based on the user's goals and intentions. Entailment selection must potentially be customised to consider the user's aims during the authoring process. In the context of a dialogue, we might expect the user to tell us something of their goals. To this end, we introduce the notion of Competency Questions, which will be discussed in detail in Section 5.

We can see abstract similarities in design between existing authoring tools that attempt to provide feedback or generate summaries and natural language generation (NLG) systems. NLG systems are frequently based on a pipeline such as: *a*) content selection *b*) aggregation *c*) grouping and ordering Following the NLG pipeline, after content selection, we must consider grouping and ordering. Often, in ontology authoring interfaces, the order in which the entailed axioms are presented is arbitrary. Regardless of how well the entailment selection strategy has functioned, a poor ordering and grouping policy can cancel the effect of selection. We can learn a lot from verbalisers such as OntoVerbal (Liang et al., 2011), and NaturalOWL (Androutsopoulos et al., 2013). In OntoVerbal, a distinction between direct and indirect axioms is made, and in NaturalOWL, coherence and complexity of axioms play a role. But there are complex issues here, especially if entailments involving complex classes are considered.

From our survey of the literature, we conclude that there are few good logic-driven approaches to entailment selection. Therefore, for our interface we plan to investigate the syntactic selection method of Denaux et al. (2012), together with a preference ordering based on linear discourse structure and the user's goals, as represented by "Competency Questions" (Section 5).

5 Being Sensitive to the User's Goals

In natural language dialogues, the participants are aware of and sensitive to one anothers' goals. We believe this sensitivity is something that could benefit ontology authoring. Although many real world ontologies, including some of the biggest ones, are constructed manually by human authors, manual ontology authoring remains a challenging task for ontology engineers (Rector et al., 2004; Dzbor et al., 2006). A large part of the difficulty is that authors cannot easily express their requirements for the ontology and, even where this is possible, it is unclear how to check whether the requirements are fulfilled.

To tackle this problem, we are incorporating the technique of Competency Question-driven Ontology Authoring (CQOA) (Ren et al., 2014) into our dialogue system. This new technique takes "Competency Questions" as requirements for ontologies and uses them to automatically generate authoring tests for ensuring the quality of the ontology.

(Informal) Competency Questions (CQs) are expressions of natural language questions that an ontology must be able to answer (Uschold et al., 1996). Below is a typical CQ:

Which processes implement an algorithm? (1)

Obviously one can think of many other CQs with similar syntactic forms, such as "Which pizza has tomato topping?", "Which animal has a tail?". In fact, they all have the following semi-formal pattern:

$$Which\ [CE1]\ [OPE]\ [CE2]? \tag{2}$$

where $CE1$ and $CE2$ are class expressions (or individual expressions as a special case) and OPE is a binary object property expression. Given a CQ of a particular pattern, its elements and their features can be identified.

The ability to answer a CQ meaningfully can be regarded as a *functional requirement* that must be satisfied by the ontology. We argue that for a CQ to be meaningful, its presuppositions (Beaver, 1997) must be satisfied by the ontology when the query is eventually asked. Otherwise the CQ or its answers will be trivial. For example, in order for question (1) to be meaningfully asked, the ontology must satisfy the following presuppositions:

1. Classes $Process$, $Algorithm$ and property $implements$ occur in the ontology;
2. The ontology allows the possibility of $Process$es implementing *Algorithm*s;
3. The ontology allows the possibility of $Process$es *not* implementing $Algorithm$s.

Particularly, if case 2 was not satisfied, the ontology could never have any $Process$ implementing any $Algorithm$ and the answer to the CQ is always "none". This would be exactly the kind of uncooperative answer looked at by the previous work on cooperative question-answering (Gaasterland et al., 1992). It is hard to imagine an ontology author really wanting to retrieve this information. Rather, this can be taken as evidence of possible design problems in the ontology. If case 3 was not satisfied, the answer to all the $Algorithm$s would be a list of all the $Process$es. This would mean that the questions would be similarly uninteresting to the ontology author, again signalling a possible problem in the ontology.

With the corresponding features and elements, the presuppositions of a certain CQ pattern can be formally represented and verified. For example, the presuppositions shown above for CQ pattern (2) can be verified automatically: (1) $CE1$, $CE2$ and OPE should occur in the ontology; (2) The class expression $CE1$ *and (*OPE *some* $CE2$*)* should be satisfiable in the ontology; (3) The class expression $CE1$ *and not (*OPE *some* $CE2$*)* should also be satisfiable in the ontology. Such kind of tests that can be derived from CQs are called *Authoring Tests* (ATs). All ATs in CQOA can be automatically tested.

This CQOA pipeline has been integrated in the interface presented in Section 3. CQs are either imported into the interface, or are entered by the users. The system will analyse CQs and identify their elements and patterns, based on which corresponding ATs will be generated and tested against the ontology. The status of ATs are constantly being monitored by the reasoner, and reported to the user. As seen in Figure 3, a traffic light approach for pass or fail status in the "task list" panel, and a written feedback in the "history log" panel will inform user of the status of the ATs. For the sake of conciseness of the interface, one can provide feedback only when the status of an AT has changed.

6 Discussion and Outlook

The WHATIF project explores what we take to be a few big ideas.

- **First of all**, we envision that understanding dialogue patterns between ontology authors and their editing tools could help improve ontology authoring tools, in particular for those providing a dialogue interface (Section 3). Our research indicates the existence of activity patterns for ontology authors using Protégé, a well known ontology editing tool (Section 2).
- **Secondly**, we advocate test-driven ontology authoring. An ontology should not just contain OWL files, but also other artefacts, such as competency questions and authoring tests. Research suggests that there exist a limited number of syntactic patterns for competency questions and these questions can be used as ontology requirements to generate authoring tests (Section 5). This means that ontology authors can use some controlled natural languages to specify their competency questions.
- **Thirdly**, we envision that dialogue based ontology authoring can get benefits from research into human language. For example, the 'entailment selection' problem in ontology authoring bears important similarities to what a Natural Language Generation system does for information presentation to a reader (Section 4).

As for future work, we plan to perform further evaluations. In an experiment that we plan to perform soon, participants will be asked to fulfil a task that involves over and underspecified parts in an ontology. We will measure the performance of users in the presence or absence of authoring tests. We will also measure the usefulness of the visual/written feedback given to the users. In a separate evaluation we will also evaluate the axiom selection mechanism during a predefined set of authoring tasks.

References

Androutsopoulos, I., G. Lampouras, and D. Galanis (2013). Generating natural language descriptions from OWL ontologies: the NaturalOWL system. *Journal of AI Research 48*, 671–715.

Beaver, D. (1997). Presupposition. In J. van Benthem and A. ter Meulen (Eds.), *The Handbook of Logic and Language*, pp. 939–1008. Elsevier.

Denaux, R., D. Thakker, V. Dimitrova, and A. G. Cohn (2012). Interactive Semantic Feedback for Intuitive Ontology Authoring. In *FOIS*, pp. 160–173.

Dzbor, M., E. Motta, J. M. Gomez, C. Buil, K. Dellschaft, O. Görlitz, and H. Lewen (2006, August). D4.1.1 Analysis of user needs, behaviours & requirements wrt user interfaces for ontology engineering. Technical report, Intelligent Software Components (ISOCO).

Gaasterland, T., P. Godfrey, and J. Minker (1992). An overview of cooperative answering. *Journal of Intelligent Information Systems 1*(2), 123–157.

Grice, H. P. (1970). *Logic and conversation*. Harvard Univ.

Liang, S. F., R. Stevens, D. Scott, and A. Rector (2011). Automatic verbalisation of SNOMED classes using ontoverbal. In *Artificial Intelligence in Medicine*, pp. 338–342. Springer.

Mellish, C. and J. Z. Pan (2008). Natural language directed inference from ontologies. *Artificial Intelligence 172*(10), 1285–1315.

Nielsen, J. (1986). A virtual protocol model for computer-human interaction. *International Journal of Man-Machine Studies 24*(3), 301–312.

Parvizi, A., C. Jay, C. Mellish, J. Z. Pan, Y. Ren, R. Stevens, and K. van Deemter (2013). A pilot experiment in knowledge authoring as dialogue. In *ISWC, Potsdam, Germany*.

Rector, A., N. Drummond, M. Horridge, J. Rogers, H. Knublauch, R. Stevens, H. Wang, and C. Wroe (2004). OWL pizzas: Practical experience of teaching OWL-DL: Common errors & common patterns. pp. 63–81. Springer.

Ren, Y., A. Parvizi, C. Mellish, J. Z. Pan, K. van Deemter, and R. Stevens. (2014). Towards competency question-driven ontology authoring. In *Proc. of ESWC 2014*.

Siberski, W., J. Z. Pan, and U. Thaden (2006). Querying the Semantic Web with Preferences. In *In Proc. of the 5th International Semantic Web Conference (ISWC2006)*, pp. 612 – 624.

Uschold, M., M. Gruninger, et al. (1996). Ontologies: Principles, methods and applications. *Knowledge engineering review 11*(2), 93–136.

Warren, P. (2013). Ontology users' survey – summary of results. Technical Report Technical Report KMI-13-1, Knowledge Media Institute.