Constructing Conceptual Knowledge Artefacts: Activity Patterns in the Ontology Authoring Process

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 are complex knowledge representation artefacts used widely across biomedical, media and industrial domains. They are used for defining terminologies and providing metadata, especially for linked open data, and as such their use is rapidly increasing, but so far development tools have not benefited from empirical research into the ontology authoring process. This paper presents the results of a study that identifies common activity patterns through analysis of eye-tracking data and the event logs of the popular authoring tool, Protégé. Informed by the activity patterns discovered, we propose design guidelines for bulk editing, efficient reasoning and increased situational awareness. Methodological implications go beyond the remit of knowledge artefacts: we establish a method for studying the usability of software designed for highly specialised complex domains.

Author Keywords
Ontologies; knowledge representation; semantic web; activity patterns; authoring tools; complex domains.

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
Ontologies define a domain of interest by describing entities and their logical relationships. A number of fields including healthcare, genetics, chemistry and geography have successfully adopted ontologies to represent the entities in these domains [11]. We can illustrate how an ontology describes a field of interest by showing an excerpt from the Wine ontology [27]; according to this ontology we can say that wine is a potable liquid made in a region using at least one variety of grape. Using the Web Ontology Language, OWL [28], authors can formalise the knowledge stated in that natural language sentence using logical axioms that establish relationships between classes of objects (Wine, PotableLiquid, Region and WineGrape) using OWL constructs (subClassOf) and properties (locatedIn and madeFromGrape) as shown in Figure 1. The strict semantics of OWL, in conjunction with an automated reasoner, allow implicit knowledge to be inferred from the asserted ontology. These facilities help ontology builders construct robust and flexible knowledge representations for use in a range of application settings [11].

```
Class: Wine
    SubClassOf: food:PotableLiquid,
    locatedIn some Region,
    madeFromGrape min 1 owl:Thing
```

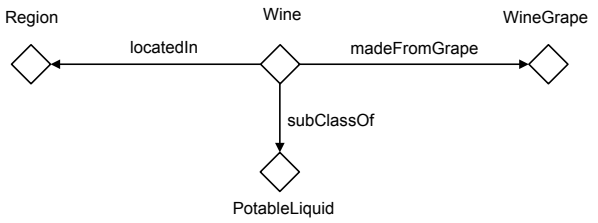

Figure 1. OWL syntax of the entities that define *Wine* and their visual representation below. Diamonds indicate individuals of a class.

Ontologies are employed for a variety of purposes, but are particularly useful for the definition of terminologies, which are frequently used in the biomedical domain for data descriptions. An exemplar is the SNOMED Clinical Terms terminology [20], which defines more than 300 000 medical concepts, and is mandated for use in around 60 countries. Similarly, the Gene Ontology [21] has three sub-ontologies containing more than 40 000 classes. It has been used to describe the functional attributes of gene products of many species, amounting to millions of annotations. The Gene Ontology also drives many analysis tools in biology [14]. These large, often complex, logical descriptions of a field of interest are key to much data analysis and are increasingly used to describe the wealth of Linked Open Data [2]. For instance, ontologies constitute a core technology of the British Broadcasting Corporation (BBC) website [3], which delivers information —including archival audio content— to around 40 million unique users per week. To support this activity, we must develop tools that reliably support the creation, maintenance and use of ontologies, by a wide range of users.

Despite their success and significance, ontologies are known to be complex to create and maintain, which hinders adoption, particularly by novices. This complexity has a number of dimensions:

CHI 2015, April 18 - 23 2015, Seoul, Republic of Korea

ACM 978-1-4503-3145-6/15/04 ...$15.00.
http://dx.doi.org/10.1145/2702123.2702495

– Cognitive complexity: description logics are difficult to understand [19] and consequently a steep learning curve is required to sufficiently master them [31].
– Size: ontologies can be large artefacts containing hundreds of thousands of axioms. Exploring, navigating and adding even a single axiom to an ontology can cause disorientation and confusion to a user [24].
– Expertise: ontologies are often used to model specific phenomena such as genes, nanoparticles, or amino acids. This requires domain experts to work together with computer scientists —a common practice, but one that can generate tensions between those playing different roles [18].
– Reasoning: because of the complexity of description logics, the logical justifications used to explain inferred knowledge or consistency problems in the ontology are often difficult to understand, let alone to correct [10].

One of the key factors contributing towards the acceptance of a new technology is the availability of tools to create high quality artefacts. Ontology authoring tools have seen many improvements over recent years, and Protégé in particular has become a popular integrated ontology development tool [5, 30]. However, primarily because little is known about how ontology authors tackle their tasks, the ontology engineering community has thus far largely ignored the question of whether existing tools are adequate to support authors. As a result, and as suggested by the analysis of the literature below, the interfaces of existing tools might not be well-suited to the intended tasks.

Background

The literature covering the usability of ontology authoring tools is scarce: seminal work in the realm of knowledge representation systems indicates that adequate reporting of errors and explanation of inferences are key to building usable tools [15]. Heuristic evaluation has often been used to assess the suitability of ontology authoring tools. These heuristics comprise questions to check whether particular types of functionality are implemented, e.g. "Is it possible to use multiple inheritance?", or cover aspects of usability, e.g. "Is there a good overview of the ontology?" [7]. Other work suggests that implementing faceted browsing, faceted viewing and inline editing would make semantic authoring tools more usable [12]. In a previous paper we proposed a framework [23], constructed around the strategies reported by ontology authors in an interview study, to evaluate ontology authoring tools based on a number of design recommendations, such as the inclusion of situational awareness mechanisms, efficient ontology population methods and search capabilities, amongst others. The conclusion of both this and previous studies using heuristics and quality frameworks, is that current tools have severe usability problems.

The direct participation of end users in the evaluation of knowledge engineering tools is also scarce. In a study of non-expert users, participants were asked to carry out basic tasks including ontology loading, and entity addition, modification and removal [13]. Tool efficiency, user attitude and aspects of learnability were measured using questionnaires. Another study, which involved authors tackling more difficult tasks including, adding subsumptions, equivalence and range axioms, also used questionnaires to measure the effectiveness, efficiency, and user experience of tools [8]. In an interview study with expert ontology authors we found that individuals employ sophisticated strategies to mitigate the weaknesses of tools [24]. Based on these strategies, we proposed a set of design insights to improve sensemaking, exploration, ontology building, debugging and evaluation tasks. The studies above involving users agree that tools are difficult to use and that no tool meets all the functional requirements of users. However, the results need to be handled with care. Ideally, they should be complemented (and corroborated) with objectively collected data to mitigate the cognitive biases that might have been introduced due to self-reporting.

To date, there is little work investigating how ontology authoring tools are actually used. In a recent study, Wang *et al*. analysed the ontology editing history of the online version of Protégé, using data mining techniques to predict the authoring event that is going to occur next [29]. In this instance, however, little is said about the tasks or activities of users.

A key contribution of this paper is the method we employ to identify the activity patterns of users. *A priori* we know very little about how users deal with conceptual knowledge artefacts and we ignore how the activities are exhibited. We collect event data via an instrumented Protégé, combine this with eye-tracking data, and use a data-driven approach to isolate core activities and their implementation details. We argue that this approach is generalisable to other situations in which individuals must deal with data of a complex nature, including exploration of big linked data, spreadsheet use and other forms of programming. The remaining contributions of the paper are as follows:

– We build a version of Protégé that logs interaction data to enable posterior analysis. The tool is openly available for the community —see §*Instrumenting Protégé*.
– We run the first study in which interaction and gaze data are used together to identify the authoring activities. This methodological contribution goes beyond the construction of knowledge artefacts, with potential application to studying the usability of highly specialised complex software whose users are typically experts —see §*Study*.
– We provide details about how the three main activities of editing, reasoning and exploring are realised. This study establishes a baseline for the exploration of how these activities are linked across different settings, tasks, types of users and ontologies —see §*Results*.
– We articulate the core activities and behaviours as design guidelines that authoring tools should support to improve the ontology engineering process. —see §*Discussion*.

INSTRUMENTING Protégé

Surveys indicate that Protégé is the preferred authoring tool of the vast majority of ontology authors, with reported uptake figures of around 68% in 2007 [5] and 74% in 2013 [30]. While we must be very careful about generalising our results beyond the realm of Protégé, it is safe to say that the outcomes of using Protégé in user studies are relevant to a majority of ontology authors.

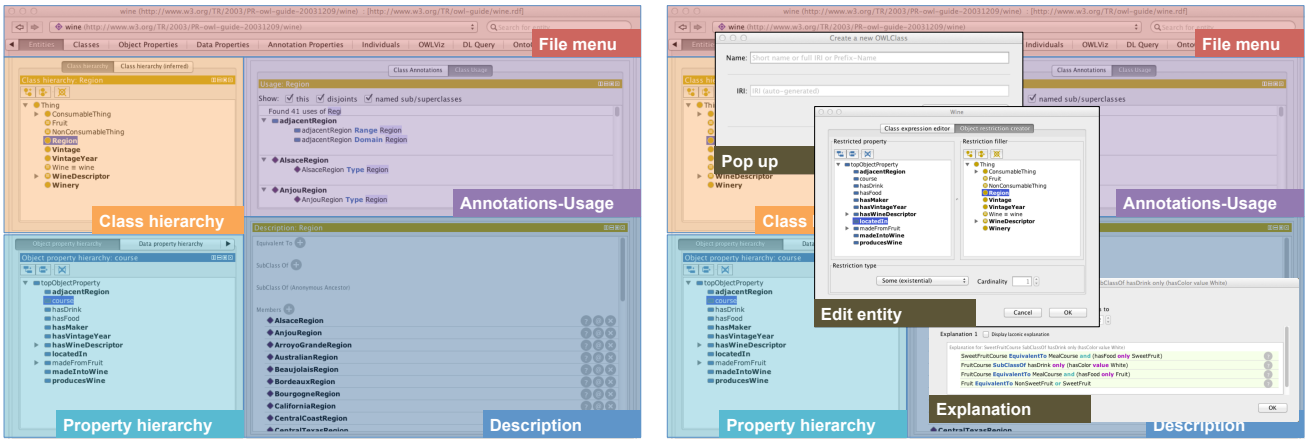

Figure 2. Core areas of interest defined in Protégé (left); Pop-up areas of interest defined in Protégé (right).

We modified the source code of Protégé 4.3 so that the tool logged the events triggered by users[1]. We classify these as *interaction events*, *authoring events* and *environment events*. Events occur in different areas of the Protégé user interface as highlighted by Figure 2 (left). At the top, the *file menu* contains a number of functionalities that can be reached by navigating the menus. In addition to running the reasoner and customising the views of Protégé, these drop-down menus typically provide redundant functionalities to those offered in the remaining areas. This area also contains a number of tabs that display different views of the ontology. On the left side, the *class hierarchy* and the *property hierarchy* contain expandable treeviews that represent the hierarchy of classes and properties as defined by subsumption axioms. On the top-right side, in the annotations-usage area, *annotations* are used to provide metadata for ontologies and their entities, and often show comments, label synonyms or authoring metadata about items in the hierarchies. *Usage* shows a list of axioms in which the latest selected entity appears. At the bottom right, the *description* area contains a list of the axiomatic descriptions for the latest item selected in either of the hierarchies; in addition, a number of functionalities to create and modify such entities can be invoked, for instance, a pop-up to set restrictions (see the *Edit entity* dialogue in Figure 2, right).

Interaction Events

Interaction events are logged as applying to either asserted hierarchies, entities and descriptions, or to inferred ones. The former are explicitly authored by the user, whereas the latter are built by the reasoner.

- *Hierarchy expanded*[2] causes the class hierarchy and property hierarchy to display child nodes of the clicked item.
- *Hierarchy collapsed* hides all the displayed subsumed classes or properties of a particular item.
- *Entity selected* indicates the selection of an item in any of the hierarchies.
- *Description selected* means an item contained in the description area is selected.

[1]The UI of the following versions including Protégé 5.0 is consistent with the version we instrumented.

[2]Protégé provides different ways of accomplishing certain events. For brevity, we describe how they most often occur in the study.

Authoring Events

- *Class/property addition* is typically carried out by typing the name of a new entity in the textbox of the *Pop-up* window —see the leftmost dialogue in Figure 2, right.
- *Entity edited:start* records that the dialogue for establishing relationships between entities, or modifying existing ones, has appeared. If we want to assert that all objects of the Wine class belong to a particular Region, the *Edit entity* dialogue provides two options: one, adding OWL statements using Manchester syntax in a textbox (writing locatedIn some Region in the class expression editor); or two, selecting, in the hierarchy, the restricted property (locatedIn), the restriction filler (Region) and setting existential (∃) as the type of quantifier.
- *Entity edited:finish* indicates that the editing is finished.
- *Entity renamed* is the action of modifying the original name of an entity in a hierarchy by typing the new name in a pop-up text box.
- *Entity deleted* removes the selected entity.
- *Entity dragged* moves an entity to another location.
- *Convert into defined class* turns a given class's restrictions from necessary into *necessary and sufficient* conditions.
- *Set property* defines the characteristics of a property by clicking on a set of checkboxes.

Environment Events

- *Run reasoner* invokes an automated reasoner. The results show whether any class is unsatisfiable or the ontology is inconsistent (i.e. there is a logical contradiction in the ontology), and any inferred knowledge is exposed.
- *Get explanation* shows the axioms that led to a given inference, including the unsatisfiability or inconsistency of any constructs —see the rightmost dialogue in Figure 2, right.
- *Load ontology* loads an OWL file into the environment.
- *Back* moves the focus to the previous view.
- *Undo* reverts the ontology to the state before the last modification was made.
- *Save* saves the current state of the ontology.

We report a thorough description of our instrumented version of Protégé, which includes more events and functionalities in addition to the ones above [25]. Generated log files (see an

excerpt below) contain a timestamp, the name of the event, and the object of the event.

```
1: 1390228276585,Element edited:finish,StEmilion subclass of:
Bordeaux and madeFromGrape value CabernetSauvignonGrape
2: 1390228277786,Save ontology,wine.owl
3: 1390228280204,Reasoner invoked,HermiT 1.3.8
4: 1390228280647,Mouse entered,Class hierarchy
5: 1390228282910,I_Entity hovered,Burgundy
6: 1390228283049,I_Entity selected,Burgundy
7: 1390228283661,I_Hierarchy expanded,Burgundy
```

This excerpt shows that the user added a property madeFromGrape and a filler to that property CabernetSauvignonGrape to the class of StEmilion wines (line 1); that, the ontology is saved (line 2); and the reasoner is invoked (line 3). When the reasoner is finished the user enters the class hierarchy (line 4), selects an inferred class (line 6) and finally expands the inferred hierarchy under Burgundy.

STUDY

Participants

Sixteen participants (11 male) between 22 and 47, median age 32.5, took part in our study. They had a background in computer science, and worked both in academia and industry. We used snowball sampling to contact possible candidates and invited any who reported to be knowledgeable about OWL and Protégé to participate. Participants completed a questionnaire containing 5-point Likert scales on the following: "Assess your expertise with OWL"; "Assess your expertise with Protégé"; "Assess your knowledge about potato varieties" (to ascertain the familiarity of participants with the domain of the tasks), where 1 indicated 'Novice' and 5, 'Expert'.

Experimental Design

The overall goal was to construct an ontology of potatoes to drive a 'potato finder' application. Participants were asked to carry out the following tasks of increasing difficulty:

1. Classify the potatoes by cropping times.
2. Import a file containing descriptions of potato yields. Represent the yields of each kind of potato and classify by combinations of yield and cropping time.
3. Add in a representation of culinary role (i.e. preferred way of cooking). Build at least two classes that combine the three axes (culinary role, yield and cropping time).

Participants were told not to start from scratch, but to adopt the persona of a 'jobbing' ontologist given an OWL ontology to extend and maintain. No explicit instructions were given on how to accomplish the tasks, but because participants were experts, we expected them to exhibit a repertoire of activities, including the invocation of the reasoner and the establishment of restrictions on classes. Participants were given a printed table with the necessary information —cropping time, yield or skin colour— to build the potato ontology. We also provided them with an OWL file containing 13 subclasses of various potato varieties and another one with a small hierarchy of cropping times for potatoes, which also removed some of the burden of large amounts of editing. There was no fixed time specified in which to complete the tasks, but participants were free to stop at any time. Participants filled out a post-test questionnaire about the perceived difficulty of each task using a 5-point Likert scale, where 1 indicated 'easy' and 5, 'difficult'.

Apparatus

The instrumented Protégé was deployed on a Windows 7 laptop with a Tobii X2-60 eye-tracker installed. The areas of interest (AOIs) correspond to those defined in Figure 2. It must be noted that the AOIs are not necessarily static as their arrangement and layout changes depending on the view being used, and users do often switch views. There are 8 different views on the default Protégé installation. While there is a general view (the one shown in Figure 2), the other views specialise in particular aspects of the ontology: classes, properties or queries. Thus, when users switch to another view or a pop-up dialogue appears, the AOIs are redefined accordingly. Video and audio were also recorded; completion times were logged during observation of the video recordings.

RESULTS

A median of 4 was obtained for both the questions regarding OWL and Protégé expertise, which confirms that participants were tending towards being experts, whereas a median of 1.5 for expertise about potatoes suggests little knowledge about the domain. Participants perceived an increasing level of difficulty in the tasks: $Mdn = 1$ and $M = 1.4$ for task 1, $Mdn = 2$ and $M = 2.1$ for task 2, and $Mdn = 2$ and $M = 2.5$ for task 3. A Kruskal-Wallis test indicates that these differences are significant, $\chi^2 = 12.19, p < 0.005$. Median completion times were 10min 42s for task 1, 12min 11s for task 2 and 18min 37s for task 3. Again, the Kruskal-Wallis test suggest that these differences are significant, $\chi^2 = 10.65, p < 0.005$. Reasoning time was included in completion times; the impact of delays was negligible as the aggregated median time for reasoning across participants was just 560 ms.

Log Data Analysis

The logs of participants' interaction with Protégé show 9 210 events of the types mentioned in §*Instrumenting Protégé* and around 55K mouse hovering events. Figure 3, on the left hand side, shows the distribution of frequencies of the events. The top three events alone (*Entity selected*, *Description selected* and *Entity edited:start*) account for the vast majority of events (56%). On the other hand, the 8 events that are less frequently triggered —those at the bottom of the long tailed distribution— account for just 3.5% of all the events. If we analyse event frequency by event category we find that interaction events account for 65% of events, followed by authoring events (30%) and environment events (5%). 83% of the interaction events are on asserted entities, whilst 17% are on inferred entities. If we consider interaction events individually, the ratio of inferred events to the total number of events (inferred+asserted) shows that 13% of entity selections, 9% of description selections, 34% of hierarchy expansions and 36% of hierarchy collapses occur on the inferred entities.

We compute the correlations between completion times and the frequency of events in each task using Spearman's correlation test. For task 2 we find there is a strong positive relationship between completion time and the number

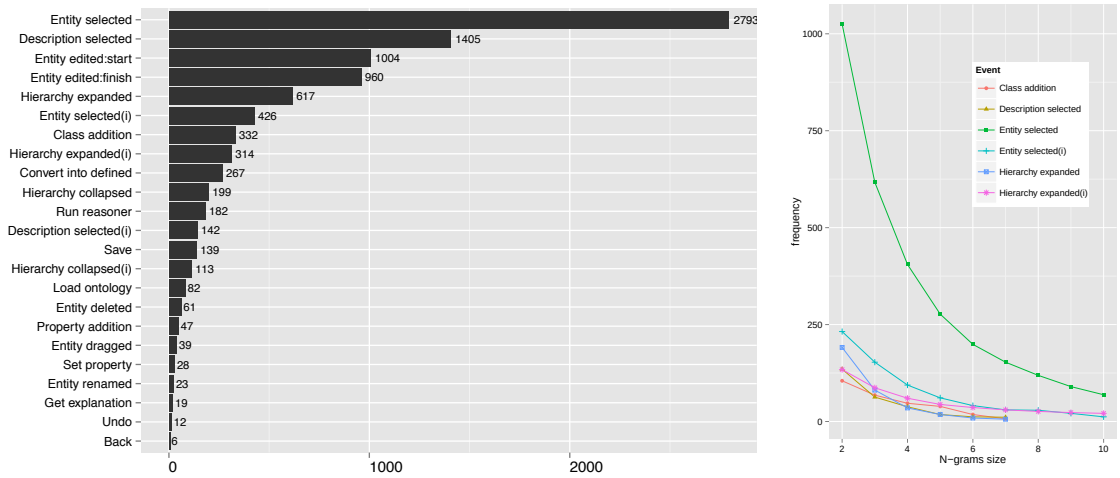

Figure 3. Log data of all participants: frequency of events (left) and frequency of N-grams of the same consecutive event (right) — (i) indicates inferred.

of selected entities $\rho = 0.76, p < 0.002$, and expansion $\rho = 0.67, p < 0.007$ and collapse $\rho = 0.63, p < 0.02$ of the asserted hierarchies. Interestingly, the more times explanations are requested ($\rho = 0.61, p < 0.03$) and the reasoner is invoked ($\rho = 0.48, p < 0.06$) the longer it takes to complete a task. Similar relationships are found for task 3 where there is a strong positive correlation between completion time and reasoner invocation ($\rho = 0.72, p < 0.003$) as well as with the number of times an entity has been renamed ($\rho = 0.63, p < 0.02$). Even if the correlations of the same event do not hold between tasks (except for the invocation of the reasoner), the strong correlations between completion time and number of selected entities, and the times the hierarchy is expanded and collapsed suggest that it took longer for participants to complete tasks because navigating in the asserted hierarchy kept them busy. Also, since the reasoner had to be run to classify entities, there are at least two explanations of why it takes longer to complete tasks for those who run the reasoner more often: (i) it might be a symptom of a trial and error strategy for not achieving the classifications enunciated by the tasks; (ii) it may be indicative of a particular authoring style in which participants run the reasoner at each modification in order to limit the spread of errors [24].

We define authoring patterns as common sequences of events. In order to identify such sequences we carry out an N-gram analysis of the events in the log files using the *tau* package [4] for the R statistical computing environment. A preliminary analysis shows there are many repeated N-grams of the same event. This is particularly illustrative of the events shown on the right hand side of Figure 3. In this case, an N-gram of size 3 means there are 3 consecutive occurrences of the same event: e.g. *class addition, class addition, class addition*. Figure 3 depicts that higher N-grams are less frequent. However, surprisingly, there are around 100 N-grams of size 10 for the entity selection event and a high number of size 3–6 N-grams for the rest of the depicted events. It is worth highlighting that all of these events (except *class addition*, which requires text entry) are performed using a mouse click. This facilitates triggering the same consecutive event without much effort. The high number of N-grams containing consecutive entity selections and class expansion events (either asserted or inferred) indicates that a great deal of activity occurs in the class and property hierarchy areas. Results also suggest that adding entities and navigating the class hierarchy is a highly repetitive and monotonous task. Another observation reveals that for N-grams > 3, more consecutive expansions occur in the inferred hierarchy than in the asserted.

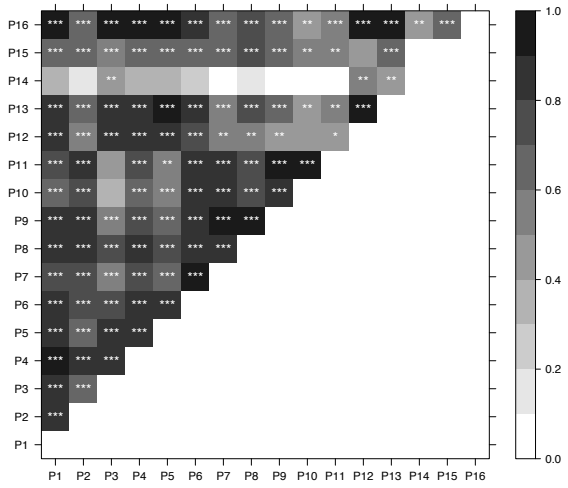

Figure 4. Mantel tests on transition matrices: * stands for $p < 0.001$, ** $p < 0.01$ and * $p < 0.05$, where correlations $r > 0.45$.**

Informed by the work of Thimbleby *et al.* [22] we build an adjacency matrix for each participant, where each cell contains the number of transitions from state S_i to state S_j, and where the states are those events described in §*Instrumenting Protégé*. These transition matrices can be considered the fingerprint of each participant and enable us to identify the similarity of the participants' interaction by comparing the matrices. One way of discerning the similarity between fingerprints is by computing correlations between transition matrices using Mantel's test[3]. An analysis of fingerprints on an

[3]Mantel's test is used in ecology to compare the genetic distance between organisms.

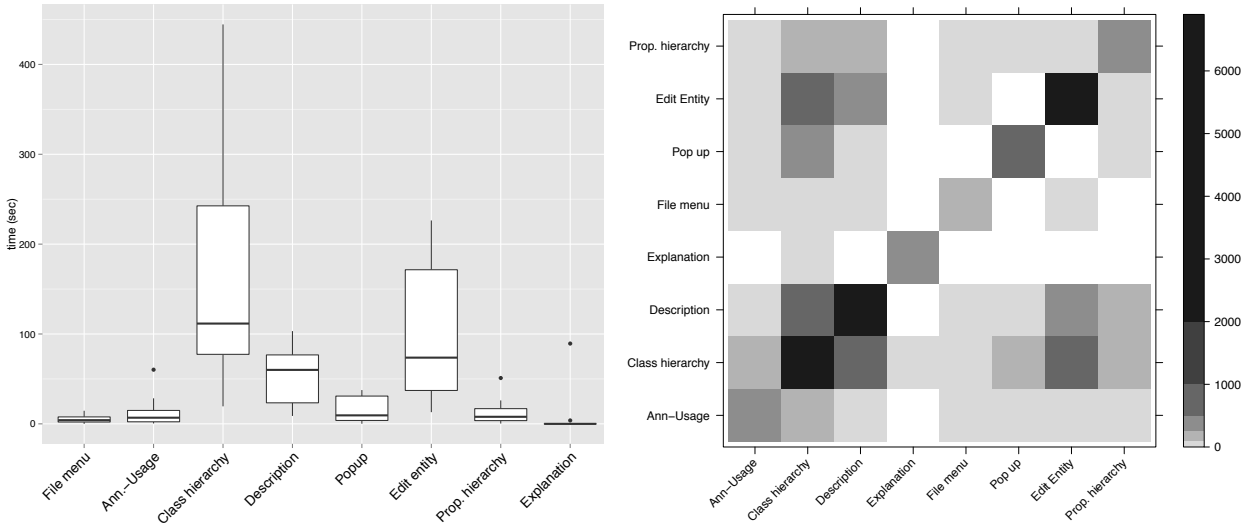

Figure 5. Eye-tracking data: distribution of aggregated dwell times on AOIs (left); matrix of consecutive transitions between AOIs (right).

individual basis yields an almost linear correlation for the majority of participants across the three tasks ($r = 0.73$ on average across all users), which suggests consistency of behaviour at an individual level. Similarly, results in Figure 4 show that the inter-user behaviour is very similar, which suggests the presence of patterns. However, it can also be seen that not all participants have common patterns, which may be indicative of different authoring styles.

Eye-Tracking Data Analysis

We collected 21 304 fixations in the defined areas of interest (AOI). The Shapiro-Wilk normality test indicates that the number of fixations and the aggregated dwell time on AOIs are not normally distributed, $W = 0.63, p \backsim 0.00$ and $W = 0.69, p \backsim 0.00$ respectively. We therefore apply a Friedman test to see if differences in the number of fixations and aggregated dwell time on the AOIs is statistically significant. Indeed, the Friedman test reveals a significant effect of AOI on dwell time, $\chi^2(7) = 80.932, p \backsim 0.00$. Post-hoc pairwise comparisons using Wilcoxon signed-rank test (with Bonferroni correction) indicate there are significant differences between all AOIs, except for *Property hierarchy* and *Annotations*. There is also a statistically significant difference in the number of fixations on the AOIs, $\chi^2(7) = 82.247, p \backsim 0.00$. Post-hoc pairwise comparisons using the Wilcoxon signed-rank test (with Bonferroni correction) suggests all differences are significant.

Table 1 shows the number of fixations and the overall dwell time on each AOI, while the boxplots on the left side of Figure 5 depict the distribution of dwell time on each AOI across all participants. The data suggest that the class hierarchy gets by far the greatest number of fixations (43%) and receives attention from participants 45% of the time. With the exception of the *Edit entity* dialogue (both 26% of the fixations and dwell time) and the description area (17% of the fixations and 15% of dwell time) the remaining areas obtain only a small number of fixations.

The matrix on the right of Figure 5 conveys the frequency of transitions between different areas. In other words, it shows

	File menu	Ann. -Usage	Class hierarchy	Desc.	Pop-up	Edit entity	Property hierarchy	Expl.
Fixations	314	722	9061	3557	883	5579	791	397
Dwell time	73.75	173.96	2318.29	766.79	227.99	1351.05	172.18	93.14
Mdn	4.11	6.85	111.61	60.05	9.45	73.69	7.85	0
SD	4.45	15.87	130.98	32.92	14.71	79.07	13.67	23.83
Max	14.59	60.26	444.53	103.23	37.60	226.36	51.01	89.36
Min	0	0.43	19.48	8.80	0.15	13.03	0.35	0

Table 1. First two rows: number of fixations and aggregated dwell time on each AOI. Below, descriptive statistics of dwell time across participants. All times are reported in seconds.

where the participants look in t_i based on where they were looking in t_{i-1}. While it is expected that cells at the diagonal of the matrix will get the higher number of fixations —it is likely that if there is a fixation in a given area the next fixation will be in the same area— there are also some unexpected findings. First of all, the matrix is (almost) symmetric, which indicates that frequencies are similar in both directions: i.e. fixating first on the class hierarchy and then on the description area is equally as likely as fixating on the description area and then looking at the class hierarchy. Secondly, there are some symmetric transitions not located on the diagonal, which stand out because of their higher frequency:

- From the class hierarchy to the description area.
- From the class addition pop-up dialogue to the class hierarchy.
- From the *Edit entity* dialogue to the class hierarchy.
- From the *Edit entity* dialogue to the description area.

Again, these findings emphasise the centrality of the class hierarchy and its role as a pivotal element in most activities.

From Events to Activities

The analysis of log and eye-tracking data alone yields a number of interesting findings about how repetitive the authoring process is and how most activities occur around the class hierarchy. In order to deepen our understanding of common authoring activities, we conducted N-gram analysis with log data and eye-tracking data merged as follows:

```
1395152087431,eye,Class hierarchy
1395152088065,eye,Class hierarchy
```

```
1395152090012,log,EntitySelected
1395152090589,log,EntitySelected
1395152091213,eye,Description
1395152091483,log,EntitySelected
1395152092605,log,EntitySelected
1395152093684,log,EntitySelected
1395152094896,log,HExpanded
```

The analysis of log data has already shown that participants exhibit similar behaviours (see Figure 4), as described in in §*Log Data Analysis*. However, the high frequency of the same events occurring consecutively initially obscures some common activities. The eye-tracking data also contains a high number of consecutive fixations in the same AOI, as shown by the diagonal of the matrix in Figure 5. To facilitate the analysis we merge the sequences of consecutive events. We illustrate this merging in the following log file shown, where the 'M' before events stands for 'multiple':

```
1395152088065,eye,M_Class_hierarchy
1395152090589,log,M_entity_selected
1395152091213,eye,Description
1395152093684,log,M_entity_selected
1395152094896,log,HExpanded
```

Using this merged data, the method used to identify the activities was to compute N-grams > 3 increasing the N-gram size one at a time until the output of the computation only yielded concatenation of repeated and smaller N-grams. Then, we selected the atomic N-grams as the representative ones and removed their permutations. The analysis of N-grams reveals that interaction can be classified into three main types of activity: editing, reasoning and exploration. Figure 6 depicts these activities, where an arc between states indicates a transition, and its value gives the probability that the transition is about to happen, while a dashed arc means there may be a gaze event in between two interaction events. These gaze events are typically fixations on the class hierarchy and the description area. For instance, the transition between *Run reasoner* and *Select entity* in Figure 6 (right) indicates that 41% of the time after the reasoner has been invoked, participants select an entity in the class hierarchy right after looking at the class hierarchy and/or at the description area.

Exploration

There are several activities that suggest participants are exploring the class hierarchy. Most notably, Figure 6 (left) shows that participants expand the asserted class hierarchy after loading an ontology, presumably to check the arrangement of the class hierarchy and the features of classes in the description area. The expansion of the asserted class hierarchy is often followed (48% of the time) by the repeated selection of different classes while participants look at their descriptions, and then finally settle on one. Interestingly, exploration of the inferred class hierarchy differs from exploration of the asserted one in several ways. Firstly, the goal of the exploration of the inferred hierarchy is not the selection of a description (for a later modification), but the observation and checking of the inferred class hierarchy and the features of its elements (which may have changed after running the reasoner). Secondly, exploration of the asserted class hierarchy entails expanding the hierarchy until a desired depth level is reached and then selecting and observing a number of items at that level; this suggests that participants may have a clear idea of the location of the classes for which they are looking. By contrast, the exploration of the inferred class hierarchy suggests more uncertainty, as participants expand the hierarchy multiple times, select and check the features of multiple classes and then again, expand the class hierarchy. This finding, together with the higher number of consecutive N-grams on the expansion of the inferred class hierarchy (see Figure 3, right), provides evidence for the exploratory nature of the navigation of the inferred hierarchy. Exploration of the asserted hierarchy is about familiarisation with the ontology and finding a specific location to add or modify an entity, while exploration of the inferred one is to check the state of the ontology.

Editing

The repetitiveness of editing classes is demonstrated by the number of times (362, an average of 22.6 per participant) the editing activity shown in Figure 6 (centre) is found. This four-step activity entails selecting an entity, which is followed by the selection of another one 37% of the time, and the selection of a description 29% of the time. As denoted by the dashed arcs, a fixation on the class hierarchy or on the description area is common in such transitions. If a description is selected it is likely to be modified (63% of the time) and followed by another selection of an entity 59% of the time.

Whilst editing, participants select entities in the class hierarchy and either look at their descriptions, or look for other entities in the class hierarchy. After the desired entity has been selected, it is modified by adding properties to classes or establishing restrictions on the class relationships. The high probability that these events occur consecutively, along with the frequency with which this activity is performed, indicates that entities were modified in batches. This can be explained by the fact that the tasks mostly consisted of adding restrictions to 13 classes that represented potato varieties.

Reasoning

We uncovered a number of patterns of activities before the reasoner is run, and a number after the reasoning process has finished. As depicted in Figure 6 (right), saving the current ontology occurs before the reasoner is invoked 40% of the time; after converting a class into a defined one, and looking at the class hierarchy, the reasoner is run 17% of the time. There are two key activities that occur after running the reasoner: 41% of the time participants observe the consequences of reasoning on the asserted hierarchy and the description area, then select one or more classes, and check their descriptions. Since the description area shows the inferred features of classes, this behaviour may indicate that participants are checking whether the reasoner had consequences for the features of classes, in addition to classifying classes. To check classification, participants expand the inferred class hierarchy and make selections on inferred entities, which occurs 30% of the time after the reasoner has been run.

How Activities Connect to Each Other

Once the main activities are identified we are able to detect them programmatically. Consequently we can reconstruct the authoring process, which can roughly be described as

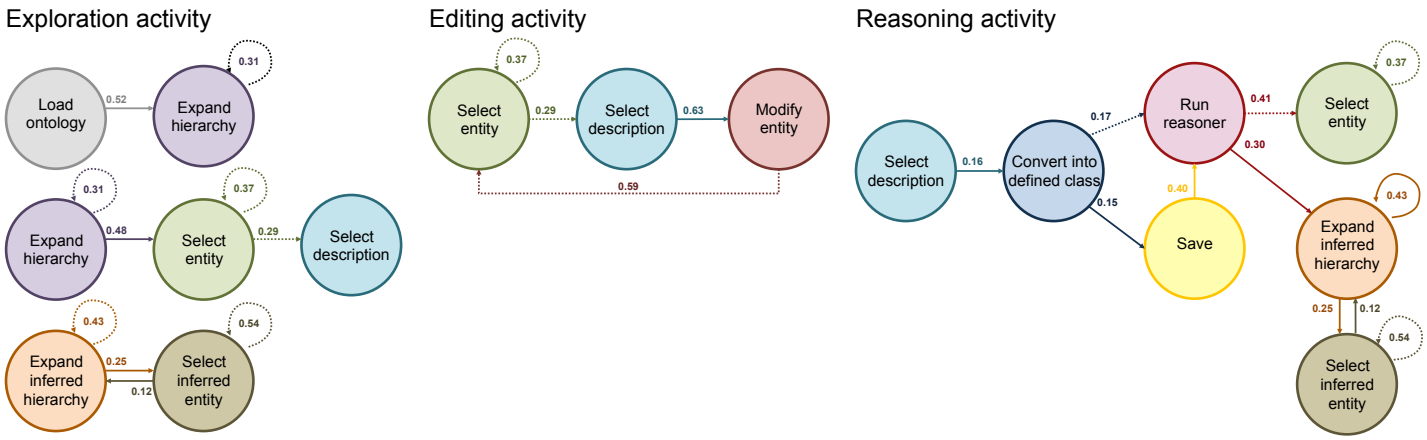

Figure 6. Exploration activities (left); the editing activity (centre); the reasoning activity (right). Dashed arrows indicate fixations on the class or description area between states.

an initial exploration of the ontology, a burst of modifications combined with exploration, and exploration after reasoning. Since activities (as opposed to events) have clear boundaries, activity completion times can be measured. We find that exploring the inferred hierarchy after reasoning takes longer with each task (from $Mdn = 4.56$s in task 1 to $Mdn = 16.27$s in task 3) perhaps because of the increased complexity. On the other hand, the editing activity stays stable (from $Mdn = 9.48$s in task 1 to $Mdn = 8.88$s in task 3). We can also identify different authoring styles: some participants run the reasoner every time they add a defined class; others only run it once all defined classes have been defined; others follow hybrid approaches.

DISCUSSION

Identification of Authoring Activities

The number of consecutive class additions, and the strong patterns found for editing activities, indicate that ontology authoring is highly repetitive and is carried out in batches: first classes and properties are added, then these are edited by adding properties to classes, and establishing restrictions on relationships. While the mouse event driven interface of Protégé certainly *enables* users to perform these monotonous tasks, correlations in §*Log Data Analysis* suggest that exploring the asserted hierarchy to place a new entity is a source of delay.

The class hierarchy receives the attention of users 45% of the time and most of the action occurs within its boundaries. Findings in §*Eye-Tracking Data Analysis* indicate that users' attention centres here when they add new classes and edit existing entities on pop-up dialogues, suggesting that the class hierarchy acts as an external memory of the ontology. This means that the decision making process for adding names to new entities and making modifications is based on the current state of the class hierarchy. Alongside the class hierarchy, the description area also plays an important role across all activities. When users explore the class hierarchy, seek the location of a class to be modified, or run the reasoner, their gaze fixates on the class hierarchy and the description area (see Figure 6). This behaviour suggests that the class hierarchy plays the role of an index containing pointers to extended information; this information is then accessed by looking at the container of the features of each element, i.e. the description area.

Even if reasoning engines have improved over the years, they (or their interaction with Protégé) sometimes show unstable behaviour that causes the authoring environment to freeze. The observed activity of saving the current ontology before running the reasoner might therefore be a strategy to prevent information loss. Those participants who ran the reasoner more often tended to be those who took longer to complete the tasks. Frequent reasoning may also be due to a particular authoring style, or an indicator of a 'trial and error' strategy for debugging [24].

We found that navigation of the inferred class hierarchy tends to be of an exploratory nature, and occurs to check the state of the hierarchy. By contrast, navigation of the asserted hierarchy is more directed —users know what they are looking for— and focuses on finding the specific location of a class to which a modification is planned. The direct relationship between a greater number of class hierarchy expansions and higher completion times might be explained by unsuccessful directed search. Similarly, exploring the hierarchy right after the ontology is loaded and checking the description of its elements suggests a strategy of becoming familiar with the ontology.

Design Recommendations

We enumerate a set of design recommendations that cater for authors' information needs, as expressed by the activities identified above.

- **Support bulk editing**: Authoring tools should provide bulk editing functionalities to add entities, and to establish restrictions on these entities —this is especially useful when building ontologies from scratch. This may be through including spreadsheet functionalities, and/or allowing authors to upload CSV (or similar) files as a mechanism to populate ontologies; such features and tools do exist, but these are not part of Protégé by default.
- **Place editing features close to the class hierarchy**: Because the class hierarchy appears to play the role of an external memory, functionality for editing ontologies should

be placed closer to the class hierarchy, and ideally the authoring environment should include recommendations for establishing restriction on entities, and suggestions for the names of those being added based on lexical patterns [17].
- **Show entity descriptions close to the class hierarchy**: The class hierarchy also plays the role of an index that users navigate when they wish to obtain information about the features of entities, which they must then read in the description area. While Protégé allows the description and class hierarchy to be juxtaposed, this is not the default; these areas should be merged rather than scattered, allowing details of entities to be hidden and shown on demand, and avoiding a division of attention across the screen.
- **Anticipate reasoner invocation**: As saving the ontology is a good predictor of running the reasoner, this could be used to anticipate the invocation of the reasoner in the background to save time (reasoning can be a lengthy process).
- **Automatic detection of authoring problems**: The 'trial and error' authoring strategy, which is operationalised as the concatenation of editing and reasoning activities, is used for debugging and indicates authoring difficulties [24]. Detecting this strategy automatically opens up new avenues to problem pre-emption.
- **Make changes to the inferred hierarchy explicit**: In order to check the (sometimes unexpected) consequences of running the reasoner, users engage in a detailed exploration of the inferred class hierarchy, which is time consuming and may be prohibitively difficult in large ontologies. This occurs because the changes made to the ontology as a result of reasoning are not made explicit by Protégé, and therefore they have to be actively sought. Authoring tools should give explicit feedback about the consequences of reasoning, without obliging users to perform a manual exploration of the class hierarchy. There could, for example, be an area for containing notifications produced by a tool that computes the semantic difference of the ontology before and after reasoning [9].

Methodological Considerations

The uptake figures of Protégé as the preferred environment to build ontologies indicate that it is an authoring environment used by the vast majority of authors (around 70%). However, our study contains some confounding factors that should be taken into account when assessing its external validity. There is the question as to whether the relatively small number of expert users, educated in computer science, are representative of novice authors, or those from other domains. Our findings are naturally most applicable to the population studied (i.e. OWL and Protégé experts), but we used an accessible domain to minimise the effect of domain expertise.

It is difficult to say to what extent the activities we found are generalisable to the ontology authoring process, and to what extent they are particular to the tool used, and the tasks carried out. We suggest that the identified activities will occur in other tasks, although their periodicity and the way in which activities are connected may vary. That is, sometimes there will be less reasoning and more editing, but the activities will still be present. We establish a baseline for future work that will include running studies in the wild, where the activity of users will be monitored on real projects, and running studies with different user groups, including novices.

Implications Beyond Knowledge Artefacts

Studying usability in highly-specialised domains requires the investment of significant resources, in terms of familiarisation with the domain and acquiring a minimum level of knowledge [6]. The identification of typical tasks is fundamental to informing usability walkthroughs. Our method, which is domain agnostic, facilitates the identification of activities by allowing them to emerge from the data. This methodology will thus have applications in highly-specialised complex software domains such as bioinformatics, e-health or design.

Previous work investigating the construction and curation of complex structured data has primarily explored the workarounds individuals employ when carrying out information management tasks in the fragmented landscapes of databases and spreadsheets [26]. End-users are able to publish complex data such as online spreadsheets, JSON, RDF and Bibtex files, provided that they have the appropriate development environment [1]. These studies, including those in which the authoring patterns of software programmers are discussed [16], conclude that the demands of users often exceed the capabilities of tools and, as a result, users develop sophisticated authoring strategies to mitigate the limitations.

Often, identifying such strategies relies on qualitative enquiry methods, which are able to provide rich information, but do not scale, and may be subject to experimenter or recall bias. Here we describe a bottom-up method for identifying user activities, unknown *a priori*, when they deal with complex information structures. Whilst this does not necessarily provide the same depth of information as a qualitative study, it provides potentially greater breadth, and supports a more ecologically valid design, as participants are free to engage in the tasks without interruption. We propose that this approach is useful not only for identifying the authoring strategies carried out by those who construct complex conceptual knowledge artefacts, but also those used for the construction of other complex artefacts of the kinds mentioned above.

CONCLUDING REMARKS

Understanding the activities performed by ontology authors is important, not only to make sense of the authoring process, but also to build tools that provide better support for engineering complex knowledge artefacts. Following an emergent, data-driven approach, we identify common authoring activities, discover how the user interface of the most popular current tool introduces inefficiencies into the authoring process, and propose a set of design guidelines that address these issues. Implications of this work go beyond the realm of ontology authoring; we suggest that the method we employ is generalisable to other scenarios in which users must deal with domain specific software and complex structured data.

ACKNOWLEDGMENTS

The instrumented version of Protégé and datasets are openly available at `http://owl.cs.manchester.ac.uk/whatif/`. The research was funded by the EPSRC (EP/J014176/1).

REFERENCES

1. Benson, E., and Karger, D. R. End-users publishing structured information on the web: An observational study of what, why, and how. In *Proc. of CHI '14* (2014), 1265–1274.
2. Bizer, C., Heath, T., and Berners-Lee, T. Linked data - the story so far. *International Journal on Semantic Web and Information Systems 5*, 3 (2009), 1–22.
3. British Broadcasting Corporation. BBC ontologies. Retrieved from `http://www.bbc.co.uk/ontologies`.
4. Buchta, C., Hornik, K., Feinerer, I., and Meyer, D. tau: Text analysis utilities. Available at `http://cran.r-project.org/web/packages/tau/`.
5. Cardoso, J. The semantic web vision: Where are we? *IEEE Intelligent Systems 22*, 5 (2007), 84–88.
6. Chilana, P. K., Wobbrock, J. O., and Ko, A. J. Understanding usability practices in complex domains. In *Proc. of CHI '10* (2010), 2337–2346.
7. Duineveld, A., Stoter, R., Weiden, M., Kenepa, B., and Benjamins, V. WonderTools? a comparative study of ontological engineering tools. *International Journal of Human-Computer Studies 52*, 6 (2000), 1111–1133.
8. Dzbor, M., Motta, E., Buil, C., Gomez, J. M., Goerlitz, O., and Lewen, H. Developing ontologies in OWL: An observational study. In *OWL: Experiences and Directions Workshop* (2006).
9. Gonçalves, R. S., Parsia, B., and Sattler, U. Categorising logical differences between OWL ontologies. In *Proc. of CIKM '11* (2011), 1541–1546.
10. Horridge, M., Bail, S., Parsia, B., and Sattler, U. The cognitive complexity of OWL justifications. In *Proc. of ISWC '11*, vol. 7031 of *LNCS*. 2011, 241–256.
11. Horrocks, I. Ontologies and the semantic web. *Communications of the ACM 51* (2008), 58–67.
12. Khalili, A., and Auer, S. User interfaces for semantic authoring of textual content: A systematic literature review. *Web Semantics: Science, Services and Agents on the World Wide Web 22* (2013), 1 – 18.
13. Lambrix, P., Habbouche, M., and Pérez, M. Evaluation of ontology development tools for bioinformatics. *Bioinformatics 19*, 12 (2003), 1564–1571.
14. Martin, D., Brun, C., Remy, E., Mouren, P., Thieffry, D., and Jacq, B. Gotoolbox: functional analysis of gene datasets based on Gene Ontology. *Genome Biology 5*, 12 (2004), R101.
15. McGuinness, D. L., and Patel-Schneider, P. F. Usability issues in knowledge representation systems. In *Proc. of the AAAI'98* (1998), 608–614.
16. Piorkowski, D. J., Fleming, S. D., Kwan, I., Burnett, M. M., Scaffidi, C., Bellamy, R. K., and Jordahl, J. The whats and hows of programmers' foraging diets. In *Proc. of CHI '13* (2013), 3063–3072.
17. Quesada-Martínez, M., Fernández-Breis, J., and Stevens, R. Lexical characterization and analysis of the BioPortal ontologies. In *Artificial Intelligence in Medicine*, vol. 7885 of *LNCS*. 2013, 206–215.
18. 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.
19. Rector, A., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R., Wang, H., and Wroe, C. OWL pizzas: Practical experience of teaching OWL-DL: Common errors & common patterns. In *Proc. of EKAW '04*, vol. 3257 of *LNCS*. 2004, 63–81.
20. Spackman, K. A., and Campbell, K. E. Compositional concept representation using SNOMED: towards further convergence of clinical terminologies. In *Proc. of the AMIA Symposium* (1998), 740.
21. The Gene Ontology Consortium. Gene Ontology: Tool for the Unification of Biology. *Nature Genetics 25* (2000), 25–29.
22. Thimbleby, H., Cairns, P., and Jones, M. Usability analysis with Markov models. *ACM Transactions on Computer-Human Interaction 8*, 2 (2001), 99–132.
23. Vigo, M., Bail, S., Jay, C., and Stevens, R. Overcoming the pitfalls of ontology authoring: Strategies and implications for tool design. *International Journal of Human-Computer Studies 72*, 12 (2014), 835–845.
24. Vigo, M., Jay, C., and Stevens, R. Design insights for the next wave ontology authoring tools. In *Proc. of CHI '14* (2014), 1555–1558.
25. Vigo, M., Jay, C., and Stevens, R. Protégé4US: Harvesting ontology authoring data with Protégé. In *The Semantic Web: ESWC 2014 Satellite Events*, vol. 8798 of *LNCS*. 2014, 86–99.
26. Voida, A., Harmon, E., and Al-Ani, B. Homebrew databases: Complexities of everyday information management in nonprofit organizations. In *Proc. of CHI '11* (2011), 915–924.
27. W3C Consortium. The Wine Ontology. Retrieved from `http://www.w3.org/TR/owl-guide/wine.rdf`.
28. W3C OWL Working Group. OWL 2 Web Ontology Language Document Overview, 2012. `http://www.w3.org/TR/owl2-overview/`.
29. Wang, H., Tudorache, T., Dou, D., Noy, N. F., and Musen, M. A. Analysis of user editing patterns in ontology development projects. In *On the Move to Meaningful Internet Systems: OTM 2013 Conferences*, vol. 8185 of *LNCS*. 2013, 470–487.
30. Warren, P. Ontology users' survey – summary of results. Tech. Rep. KMI-13-1, Knowledge Media Institute, 2013.
31. Warren, P., Mulholland, P., Collins, T., and Motta, E. The usability of description logics. In *Proc. of ESWC '14*, vol. 8465 of *LNCS*. 2014, 550–564.