Published in final edited form as:
Semant Web. 2013 January 1; 4(1): 89–99. doi:10.3233/SW-2012-0057.

WebProtégé: A Collaborative Ontology Editor and Knowledge Acquisition Tool for the Web

Tania Tudorache, **Csongor Nyulas**, **Natalya F. Noy**[*], and **Mark A. Musen**
Stanford Center for Biomedical Informatics Research, Stanford University, 1265 Welch Road, Stanford, CA 94305, USA

Tania Tudorache: tudorache@stanford.edu; Csongor Nyulas: nyulas@stanford.edu; Natalya F. Noy: noy@stanford.edu; Mark A. Musen: musen@stanford.edu

Abstract

In this paper, we present WebProtégé—a lightweight ontology editor and knowledge acquisition tool for the Web. With the wide adoption of Web 2.0 platforms and the gradual adoption of ontologies and Semantic Web technologies in the real world, we need ontology-development tools that are better suited for the novel ways of interacting, constructing and consuming knowledge. Users today take Web-based content creation and online collaboration for granted. WebProtégé integrates these features as part of the ontology development process itself. We tried to lower the entry barrier to ontology development by providing a tool that is accessible from any Web browser, has extensive support for collaboration, and a highly customizable and pluggable user interface that can be adapted to any level of user expertise. The declarative user interface enabled us to create custom knowledge-acquisition forms tailored for domain experts. We built WebProtégé using the existing Protégé infrastructure, which supports collaboration on the back end side, and the Google Web Toolkit for the front end. The generic and extensible infrastructure allowed us to easily deploy WebProtégé in production settings for several projects. We present the main features of WebProtégé and its architecture and describe briefly some of its uses for real-world projects. WebProtégé is free and open source. An online demo is available at http://webprotege.stanford.edu.

Keywords

Web-based ontology editing; knowledge acquisition; collaboration; Protégé; Semantic Web

1. Introduction

Researchers and application developers in many domains have successfully used ontologies to solve a wide range of problems, including data integration, configuration, data analysis, annotation, and search [3, 14]. More and more frequently, many of these ontologies are products of collaborative development, with active involvement of domain experts, and not just knowledge engineers. In order to develop tool support for collaborative ontology development in this setting, we have analyzed the ontology-development process in a large number of projects developing biomedical ontologies [22]. We believe that our findings are common to other domains, as well. In several projects that we have analyzed, a small group of ontology experts develops the core of the ontology. After they identify the main structure and modeling patterns, domain experts contribute content to the ontology. There are two

[*]Corresponding author.noy@stanford.edu.

main challenges at the step of involving domain experts in the ontology development: First, the domain experts must get familiar with knowledge modeling, or even formal underpinnings of the representation language, and ontology-editing tools. These tools are often complex [4] and create a high entry barrier that is not easy for domain experts to surpass. Second, while many users are familiar with collaboration technologies in the style of Web 2.0 applications, we must adapt these approaches to ontology development and make them an integral part of the development process, improving the user experience and productivity. In this paper, we present WebProtégé—a collaborative Web-based platform that supports ontology editing and knowledge acquisition, and that can be easily tailored for domain-expert use. While our previous papers [18,17,19] have focused on particular aspects of WebProtégé in the context of deploying the tool in a number of real-world use cases, this paper gives an overview of the WebProtégé generic features, both functional and architectural, and describes how the tool can be extended and customized to support other use cases.

The paper is organized as follows. Section 2 gives an overview of the existing Web-based ontology editors. We then describe the main features of WebProtégé, including its declarative user interface, collaboration and support for ontology reuse, and its extensibility in Section 3. Section 4 gives a high level overview of the tool architecture. We describe the real-world deployments of WebProtégé in Section 5, and then discuss current limitations and future plans in Section 6.

2. Related Tools

There are a small number of collaborative ontology editors that exist today for the Web. Semantic wikis [8] add semantic capabilities to the traditional wikis. Most of the semantic wikis focus on *enhancing the content* with semantic links that allows a more meaningful navigation and supports richer queries. The semantic wikis usually associate a page to a particular instance in the ontology, and the semantic annotations are converted into properties of that instance. For WebProtégé, our focus is on editing the ontology structure itself (schema, or class-level, TBox), including more complex representations (e.g., editing OWL restrictions), and on knowledge acquisition where the immediate validation of the values based on the definitions in the ontology is possible. OntoWiki [1] is one example of a semantic wiki that supports distributed knowledge engineering, but it also focuses more on acquisition of instance data and not the ontology or schema itself. MoKi[1] [6] is a collaborative tool for enterprise modeling, implemented on top of a Wiki, that has been successfully deployed in a number of real world use cases. MoKi supports the editing of OWL domain models using the Wiki forms. Its focus remains, however, on supporting the easy modeling of business processes and enterprise models, rather than being a fully-fledged OWL editor. Neologism[2] [2] is a web-based vocabulary editor and publishing tool that focuses on building RDF and lightweight OWL vocabularies. As the authors of the tool stress on their web-site, Neologism is not an ontology editor. The tool offers primitive collaboration features, as it is still work in progress. Knoodl[3] is a commercial ontology editor, built on top of a wiki platform that provides basic ontology editing features. Knoodl combines the structured ontology information with a free-text wiki page and focuses more on searching capabilities and linking to SPARQL endpoints. Soboleo[4] [23] and Pool-Party[5] [15] are Web-based tools for creating collaboratively SKOS and RDF vocabularies. They support the lightweight editing of taxonomies, and their focus is on providing services that

1 https://moki.fbk.eu
2 http://neologism.deri.ie/
3 http://knoodl.com
4 http://www.soboleo.com/
5 http://poolparty.biz/

take advantage of these vocabularies, such as annotation or tagging of resources, faceted browsing, and semantic search.

None of the Web-based ontology tools that we have investigated provide such extensive editing support for both the class level and instance level information as WebProtégé does. We could also not find Web-based tools that provide customized views of the ontology for different users, or extensive collaboration support, or such an extensible and pluggable architecture. A number of collaborative ontology editors are available as desktop applications, but they do not work in a Web browser and require installation on the user's machine. A good review of these tools is available elsewhere [11].

3. WebProtégé Features

WebProtégé[6] is a Web-based lightweight ontology editor. Our goal in building this tool was not to offer yet another ontology editor, but rather to fill in a significant gap in the landscape of ontology tools. Our aim is to provide an ontology tool that a large spectrum of users, ranging from ontology experts to domain experts, can use. Thus, the ability to customize the user interface for users with different levels of expertise was of utmost importance in our design decisions (see Section 3.1). Furthermore, most projects use community-based approaches to develop ontologies. Section 3.2 describes the main collaboration support in WebProtégé. Ontology reuse is also common in mature ontology domains, such as biomedicine. Section 3.3 gives an overview of how WebProtégé supports the reuse and interlinking of biomedical ontologies. Creating an open, extensible and flexible infrastructure that can be easily adapted to the needs of different projects was also a top priority in our design (Section 3.4).

3.1. User Interface

The WebProtégé user interface (Figure 1) is built as a *portal*, inspired by similar infrastructures, such as, iGoogle[7] and myYahoo,[8] which are already familiar to online users. The WebProtégé portal is composed of *tabs*, either pre-defined or defined by the users. A tab is an empty container made up of smaller components—*portlets*—that provide independent pieces of functionality. WebProtégé comes with a library of portlets, as well as predefined tabs that are familiar to users from desktop ontology editors (see Section 3.1.1). A user can customize the appearance of WebProtégé by rearranging portlets using drag-n-drop, by adding and removing portlets and tabs from the top toolbar. The user may also save a particular layout configuration for a project, thus creating a personalized view of the ontology. The customized layout will be restored the next time the user opens the project in WebProtégé.

3.1.1. The Portlet Library—WebProtégé includes a set of predefined tabs,[9] which contain the most popular functionality in the Protégé desktop editor [12]. For example, the pre-defined *Classes Tab* enables users to browse and edit the class hierarchy and the properties of classes. The *Properties Tab* provides access to the details of the properties in the ontology. The *Individuals Tab* contains forms for acquiring instances of classes.

Besides the pre-defined tabs, users may add their own tabs containing portlets that are useful for a particular task. They can also re-configure the pre-defined tabs, removing the portlets that are not useful in their projects and adding other ones.

[6]http://tinyurl.com/webprotege
[7]http://www.google.com/ig
[8]http://my.yahoo.com/
[9]http://tinyurl.com/webprotege-ug

Users create their own tabs and custom-tailor the pre-defined tabs using a library of WebProtégé portlets, which support common browsing and editing patterns in ontology development. WebProtégé currently supports the full-fledged editing of classes, properties and individuals. For example, the *Class Tree* portlet, the *Property Tree* portlet and the *Individuals List* portlet support the creation, deletion and searching of the respective entities. The *Properties Portlet* displays and edits the data, object and annotation properties of ontology entities (classes, properties or individuals) and can be used in combination with other portlets. The *Properties View* portlet is very popular and displays the properties that have the selected class in its domain. Users find this portlet very intuitive, and interpret its content as seeing the "relationships" of the class. The *Restrictions* portlet allows the editing of the asserted class conditions and includes support for auto-complete.

We implemented portlets for supporting common modeling patterns. For example, the *Instance Table* portlet, which displays instance values in the form of a table (each row is an instance, and columns correspond to properties of the instance), can be used for browsing and editing reified relations. Several projects currently use this portlet.

Other useful portlets include the *HTML* portlet, which allows the embedding of arbitrary HTML snippets as part of a tab (for example, we currently use this portlet to include a Twitter feed frame in one of the tabs), the *Metrics* portlet, which provides essential statistics about the ontology, or the *Ontology List* portlet, which lists the available ontologies for the currently logged in user.

All portlets available in WebProtégé can be configured by means of a property list, which gives great flexibility in customizing a portlet for a particular task. For instance, the *Class Tree* portlet can be configured to display only a certain branch in an ontology, or the *Instance List* portlet to display only the instances of a particular class. Similarly, the buttons, their labels, and toolbars are controlled through a property list. For example, we hide the ontology toolbar for certain projects to discourage users from changing the user interface configuration, at the request of the ontology owners.

3.1.2. Knowledge Acquisition Forms—WebProtégé is not only an editor for classes and their properties, but also provides extensive support for knowledge acquisition of instance data. Three of the real-world projects that we describe in Section 5 show how WebProtégé can be customized into a knowledge-acquisition tool suited for domain experts. The customized tool presents the domain experts with simple forms known from other Web-based applications, without them being really aware that they are, in fact, editing instances of an ontology.

Figure 2 shows how we configured the knowledge-acquisition forms for two projects on which we collaborate with the World Health Organization. The forms are created as part of the *Property Form* portlet,[10] which allows the association of a property in the ontology to a form field (e.g., textfields, checkboxes, drop-down lists, radio buttons, etc.) used for displaying and acquiring the values of that property.

Forms, similarly to portlets, are configurable using property lists. All aspects of a form field (label, associated property, size, etc.) are configurable, including the groups of users who are allowed to edit that property (other users will only be able to browse the values). This configuration enables us to define fine grained access control at the level of a property in the ontology.

[10]http://tinyurl.com/webprotege-forms

3.1.3. User Interface Configuration—

One of our main goals in developing WebProtégé was to have a tool that can be easily configured for different settings and types of users. All aspects of the WebProtégé user interface (layout, portlets, forms, etc.) are configured in a XML file[11] stored on the server side. The layout of tabs and portlets can be done in the user interface directly; the users can drag and drop various portlets and then save the configuration. However, the configuration of forms (Section 3.1.2) currently requires changes in the XML file.[12]

The declarative user interface allows us to define custom views for different users. Currently, WebProtégé supports three levels of user interface configurations:

1. A general, default configuration that applies to all projects with no customizations;
2. A configuration for a specific project;
3. A configuration for a specific project and a specific user.

WebProtégé builds the user interface dynamically when the user opens a project by trying to find the most customized configuration (Option 3), and, if does not find it, it will fall back to the previous two options.

This flexible mechanism allowed us to build custom ontology views for different user types. For example, a content expert may see the knowledge acquisition forms for a particular branch in the ontology, while a project manager will be interested in tracking the changes and activity statistics at the level of the entire ontology.

3.2. Collaboration Support

WebProtégé provides extensive collaboration support, including change tracking, contextualized threaded discussions, watches and notifications, an extensible access policy mechanism, and generation of statistics of the ontology-development process [18]. Even though, we reused the functionalities of Collaborative Protégé [22], which are themselves implemented using Semantic Web technologies, we still had to add new functionalities.

All authoring operations in the WebProtégé are tracked as instances of a Changes and Annotation Ontology (ChAO) [9]. Several portlets present this declarative change-tracking information in a user-friendly way. For example, the *Changes* portlet shows the changes performed on an entity: the author of a change, timestamp, and a user-friendly description of a change. A different portlet displays the changes performed in the entire ontology.

Users can also have contextualized threaded discussions and notes attached to different entities in the ontology. The notes are typed and structured and are also stored as instances of ChAO. A user may add the *Notes Tree* portlet to any tab, and the portlet will display the notes attached to the selected entity in a threaded view. For instance, we are using the Notes Tree portlet both in the Classes Tab and in the Properties Tab, and it will display the notes attached to classes or properties in the respective tabs. Figure 1 shows the visual indication of notes attached to classes as a commenting icon. We are also displaying the note counts of subclasses using a smaller icon, so that it is easier for users to quickly assess the most active ontology branches and “drill-down” to the relevant classes.

A *watch* functionality allows users to express their interest in certain entities, and unlike wikis, in branches in the ontology. The *Watched Entities* portlet will display the changes and new notes of the watched entities and branches. A notification mechanism will send email

[11]http://tinyurl.com/webprotege-uiconfig
[12]The configuration of forms in the user interface directly will be part of a future release.

notifications to users containing direct Web links to the watched entities that have changed or have new notes attached to them (This feature is one of the most appreciated ones in WebProtégé that keeps users engaged). Users may configure their profile to receive notifications immediately, hourly, daily, or not at all.

Project managers are able to follow the progress of the development process by using one of the change-analysis plugins available in the Web platform and also as a plugin of the Protégé desktop client.[13] These plugins present statistics of edits, which can be filtered by authors and time frame, as well as author-network dependencies, and tag clouds [5].

WebProtégé supports a flexible *access policy* mechanism. We defined some common access policies that the user interface enforces, such as *Read*, *Write*, *Create new users* or *Display in ontology list*. The latter permission controls if an ontology should show up in the list of available ontologies displayed on the home page of WebProtégé, and it allows us to make certain ontologies private. Administrators may also define other custom access policies, such as the right to enable or disable certain tabs or portlets, or to use certain functionalities in the tool. We use an ontology to store the configuration of the projects available in WebProtégé and their access policies [20].[14] As mentioned in Section 3.1.2, WebProtégé also support a property-level access policy that allows only a certain group of users to edit that property in a knowledge-acquisition form.

3.3. Reuse of Biomedical Ontologies

Reusing terms and interlinking of ontologies are best practices in ontology development. As biomedicine is one of the domains in which most public mature ontologies exist, we provide a generic way of reusing terms from biomedical ontologies stored in the BioPortal repository [10].[15] BioPortal is a repository of over 250 biomedical terminologies and ontologies that serves a large community, and that can be accessed both in a Web browser and through RESTful Web services. Even though BioPortal is a biomedical resource, the technology behind the tool is domain-independent and has already been deployed in several other domains. By accessing the REST services, the *BioPortal Reference* portlet allows WebProtégé users to search for terms in BioPortal ontologies in the context of their ontology editing, to browse their details, and then to create a reference to these terms from their evolving ontology with a single click. The WebProtégé customization used for the authoring of the International Classification of Diseases makes heavy use of the interlinking feature: In the last year, users have created over 40,000 links to terms in BioPortal ontologies.

3.4. Extensibility

We implemented WebProtégé as a pluggable and extensible architecture that can be customized to the needs of any particular project. The plugin infrastructure on the front end side enables developers to easily build their own portlets, forms and tabs by implementing predefined Java interfaces. We also tried to make the plugin interfaces similar to the Protégé desktop client ones, to lower the entry barrier for existing Protégé plugin developers.

Currently, WebProtégé loads any ontology language and format that is supported in the Protégé 3.x series (OWL 1.0, RDF(S), and Frames). Developers can add support for other ontology formats by implementing an Ontology Service interface[16] that separates the server and the client (see Section 4). In this way, they can easily add different back ends (e.g., OWL-API, Jena, triplestores) without having to make any change to the user interface code.

[13] http://tinyurl.com/change-analysis
[14] http://tinyurl.com/metaproject
[15] http://bioportal.bioontology.org
[16] http://tinyurl.com/wp-ontologyService

As a proof of concept, and to address a common user request, we have already implemented an OWL-API backend for WebProtégé that is available in the Protégé SVN repository[17] and that we will release soon (see Section 6). The service layer adds a level of separation between different components of the system. The services[18], such as the Ontology Service, the Notes Service, Project Configuration service, and so on, dictate the functionality of WebProtégé. Changing the implementation of one service, does not imply changing the other services. For example, if one develops a backend for WebProtégé by implementing the Ontology Service interface, he or she will not have to provide a new implementation of the Notes service. This lego-style architecture allows the combination of components from different software libraries. For example, the OWL-API back-end may work with the Protégé 3 notes and discussions component.

4. Architecture of WebProtégé

Figure 3 shows a high level overview of the WebProtégé architecture. The user interacts with a client application (front end) that runs in a Web browser and is implemented in JavaScript. The server side (back end) runs in a servlet container, such as Tomcat, and is implemented in Java.

The WebProtégé front end and back end interact via a service layer defined as Java interfaces. We created interfaces for accessing and changing the ontology content (e.g., get subclasses of a class), for change tracking (e.g., get all changes of an ontology entity), for project administration and access control, and for project layout and configuration (e.g., get the layout for project NCI for user X). The ontology access services are currently implemented using the Protégé 3 API in the released version (an OWL-API implementation is also available, but not released, yet). As we mentioned in Section 3.4, other back ends can be easily plugged in by providing an implementation of the respective services. The WebProtégé services can be called also by third-party applications.

The WebProtégé back end connects to a Protégé server that provides access to the ontology, storage, collaboration features, change management, and all other back end functionality. The Protégé collaborative framework [22] supports simultaneous editing of an ontology by multiple Web-based or desktop clients, tracks the ontology changes, manages the notes and discussions, and so on. The framework also enforces the access policies, such as read and write. One extremely useful feature in this architecture is the fact that WebProtégé and the Protégé desktop client may access the same ontology for concurrent reading and writing. All changes made by either of the clients are immediately visible in the other clients. This mechanism is important if certain features are not available in the Web client, for example, reasoning. One of the projects we host on the WebProtégé demo site, makes all ontology edits in WebProtégé and it performs the reasoning on the same copy of the ontology in a Protégé desktop client.

The WebProtégé front end is implemented using the Google Web Toolkit (GWT).[19] GWT allows developers to write Java code for the user interface and then compiles it into optimized Javascript to run in a Web browser. One huge advantage of GWT is that it enables Java developers with no Web-based UI experience to write UI code directly in Java and to use the extensive support of the Java development environments. GWT is free and open source and used by a large community around the world.

[17]http://smi-protege.stanford.edu/svn/web-protege/
[18]http://tinyurl.com/wp-services
[19]http://code.google.com/webtoolkit/

5. Use Cases

We deployed WebProtégé in a number of real-world projects. The most prominent one is the development of the 11th revision of the International Classification of Disease (ICD-11) lead by the World Health Organization (WHO) [18,17,19]. Most of the United Nations member countries use ICD to compile health statistics, to monitor health-related spending and to inform policy makers. Hence, ICD is an essential health-care resource around the world.

We created a customized knowledge-acquisition form tailored for the WHO domain experts (left side of Figure 2 shows a part of it) by creating a specific layout configuration for WebProtégé. This project has served as a requirements' driver for the development of several of the portlets and forms currently available in WebProtégé, such as the BioPortal Reference portlet. As the requirements and the core ontology (to which forms were bound) were evolving in parallel at a rather accelerated pace, we had to make all portlets very configurable so that we can adapt the user interface quickly. We also implemented some custom portlets and forms as extensions to the generic ones (e.g., the ICD class tree portlet is an extension of the generic class tree portlet, but uses customized icons for classes based on a *displayStatus* object property).

We branded the customized WebProtégé for ICD-11 as *iCAT*.[20] iCAT has been in production use since October 2009. The platform has 200 registered users from all over the world. Most of the users are international medical experts that are not knowledgeable about OWL or ontologies. They interact with the knowledge-acquisition forms to enter information about diseases and to create the new ICD-11 classification. The ICD ontology has currently over 5 million triples, and contains over 31,000 classes that are actively edited. Users have already made over 105,000 changes in iCAT, and created more than 40,000 cross-links to external ontologies by using the BioPortal Reference portlet. More than 19,000 notes and discussions are recorded in the platform. The Changes and Annotation Ontology (ChAO) used for storing the change tracking and notes instances has over 5 million triples. iCAT is under very active use and it will support the ICD-11 revision until 2015, when WHO will officially release ICD-11. After 2015, iCAT will be used to support minor revisions to ICD-11.

Similarly to iCAT, we deployed two other production platforms used for the development of other WHO classifications: the International Classification of Traditional Medicine (ICTM) and the International Classification of Patient Safety (ICPS). The right-hand side of Figure 2, shows a partial screenshot of the ICTM platform, in which we had to make sure that WebProtégé works properly with international content, as ICTM is concurrently authored in 4 languages: English, Chinese, Japanese and Korean. Both platforms are in current active use in a production setting.

The WebProtégé demo platform[21] hosts several real-world projects. These projects are developed by groups of researchers from different domains and we have configured the access permissions to these ontologies based on the preferences of the ontology authors. Therefore, some of the ontologies can be browsed publicly, while others are available only to a closed group of users. Some of the publicly accessible ontologies are: the Ontology of Parasite Lifecycle (OPL) developed by a group of researchers from University of Pennsylvania and other institutions, an extensions of the Ontology for Biomedical Investigations (OBI) for Web service annotations, the Product-Service Systems ontology,

[20]A demo platform is available at: http://icatdemo.stanford.edu. Demo account: swj/swj.

[21]Available at: http://webprotege.stanford.edu. Guest account: Guest/guest.

and the NIH Health Indicators. These ontologies use different ontology languages (most use OWL, but some are using Frames), and they differ in the type of edits they are performing. For example, OPL developers heavily edit class restrictions, and use knowledge-acquisition forms for authoring the classes metadata, while developers of other ontologies follow a more lightweight approach and edit only the class hierarchy and the class annotation properties.

As anyone can freely download and install WebProtégé on any machine, we do not have exact counts of external installations or users of WebProtégé. We use the mailing lists as a gauge of interest and we do get regular emails about WebProtégé.

6. Discussion and Future Work

We performed several usability studies of WebProtégé and published the results elsewhere [18,17,19]. These results were all encouraging. We have also created a usability questionnaire and published it on the WebProtégé wiki.[22] We have posted a request on the Protégé mailing list to help the Protégé team evaluate our software. We asked respondents to fill out a questionnaire on various aspects of the Protégé tool. In the questionnaire, 13% of respondents (19) indicated that they use WebProtégé. We have contacted these 19 users and asked them to fill out another questionnaire, this new one specifically for WebProtégé. 18 of them responded. We have received further responses to the survey from users who have followed the link directly from the WebProtégé wiki page. Even though, we omitted to ask about the amount of experience users had with Protégé or WebProtégé, we concluded from the responses we received that most respondents have used WebProtégé for a real project, rather than just trying out the online demo. In this survey, we were mainly interested in learning whether users find WebProtégé easy to use, and which features they appreciate the most and which features are missing. We also used the responses of the survey to guide our future development plans for WebProtégé. We received in total 28 responses to the online questionnaire. 88% of respondents found the tool easy to use, and 75% of them found it easy to learn. When we asked what the respondents particularly liked about WebProtégé, they highlighted the fact that the interface is web-based and there is no installation required. When we asked what was missing, many users listed the features that were available in the desktop version of Protégé, and said that they wanted all of those features in the Web version.

One missing feature that came up several times is the support for OWL 2. We already implemented an OWL-API back end [7] for WebProtégé that covers most of the current WebProtégé services, and we plan to release it in the very near future. Using an OWL-API back end will also open up possibilities that were difficult to support in the current architecture.

For example, several of the user studies, and our other interactions with users highlighted the need for users to download a local copy of the ontology and to edit it off-line. There are challenges in implementing this "sandbox" feature because we must be able to fold back the changes into the master version. Thus, we must address cases such as a user creating a subclass of a certain class in the sandbox and someone else deleting this class from the ontology in the master version. We envision developing different strategies for merging and in certain cases asking the users to resolve the conflicts manually (e.g., similar to the work of Stojanovic [16]). We have already made significant steps in addressing this challenge. Our group has implemented an OWL-API server that supports the concurrent access to an ontology similar to source control repositories, such as SVN, using check-in and check-out operations [13]. The server has an extensible conflict-resolution mechanism. We plan on

[22]http://tinyurl.com/wp-survey2

using the OWL-API server and its conflict-resolution mechanism to support ontology snapshots and sandboxes in WebProtégé.

When we started WebProtégé, we envisioned the WebProtégé interface as a lightweight version of the interface in the Protégé desktop client. However, the usability studies and our interactions with users indicate that there is not a feature in the desktop client that some user does not want in the Web client. We have already added features to WebProtégé that cannot be considered light weight (e.g., restriction editing) because users were requesting it. We are gradually adding the more complex features to the Web client as users express a need for them, while at the same time trying to preserve the default lightweight feel of the Web tool.

Another area of active research in our group is studying how we can represent better the declarative user interface of WebProtégé. In the current implementation, we are using an XML file to configure the user interface. This approach is reasonable for projects that do not require complex configurations. However, the WebProtégé configuration file for ICD is over 8000 lines long (a class has over 45 fields with multiple configuration), and this configuration is getting very hard to manage, maintain, and validate. To address this issue, we created an OWL representation of the user interface[23] [21]. This ontology allows us to create templates of configurations, to associate constraints for configuration parameters and constraints or dependencies between portlets (e.g., the restriction portlet should be used only with OWL ontologies). We hope that this approach will make the management and validation of UI configurations easier.

We are currently working on integrating additional collaboration mechanisms in the platform and analyzing the dynamics of collaborative ontology development. We already performed a quantitative and qualitative analysis on the change tracking and notes activity on several collaborative projects that are using WebProtégé, and we identified emerging user roles that are common to all projects [5]. Users with these different roles have different interests and different requirements for the user interface. As we learn more about the way domain experts develop ontologies in a distributed setting, we can adjust the tools to support collaboration even better.

As the Web access gets more wide spread, and the modern Web browsers providing rapid improvements of their JavaScript engines, we can envision that some users might prefer to use only the Web version of Protégé, rather than the “classic” desktop client. Nevertheless, we believe that both the Web and the desktop client have their own role, and complement each other, in a similar way that Web and desktop email clients coexist and fulfill different requirements. We will continue to develop in parallel both WebProtégé and the Protégé desktop client, and we will strive to make them interoperate seamlessly.

7. Conclusions

We presented WebProtégé—a lightweight ontology editor and knowledge-acquisition tool for the Web. WebProtégé provides extensive collaboration features that are integrated as part of the ontology development itself. The tool also comes with highly customizable and declarative user interface that can be adapted to any level of user expertise, as we have proven in the customizations for several real-world projects. Our experience in using WebProtégé in the projects run by our collaborators as well as the feed-back that we received from our larger user community, show that WebProtégé is actively used, fills in a niche that no other ontology-development and knowledge-acquisition tool does, and its user interface is simple enough to be used by domain experts.

[23]The ontology is available for browsing on the WebProtégé demo server, called the UI Ontology.

Acknowledgments

We are grateful to Jennifer Vendetti and Timothy Redmond for contributing code to the WebProtégé user interface and its back end. We are also very thankful to the users of WebProtégé and of the Protégé mailing lists who have provided us with feedback and requirements for the tool. Our sincere thanks to Alex Skrenchuk for providing system administration support for the WebProtégé servers. The work presented in this paper is partially supported by the NLM Grant 1R01GM086587-01. Protégé is a national resource supported by grant LM007885 from the U.S. National Library of Medicine.

References

1. Auer, S.; Dietzold, S.; Riechert, T. OntoWiki–a tool for social, semantic collaboration. Fifth International Semantic Web Conference, ISWC, volume LNCS; Athens, GA. Springer; 2006. p. 736-749.
2. Basca, C.; Corlosquet, S.; Cyganiak, R.; Fernández, S.; Schandl, T. Neologism: Easy vocabulary publishing. The 4th Workshop on Scripting for the Semantic Web; Tenerife, Spain. CEUR; 2008.
3. Bodenreider O, Stevens R. Bio-ontologies: current trends and future directions. Briefings in bioinformatics. 2006; 7(3):256. [PubMed: 16899495]
4. Dzbor, M.; Motta, E.; Buil, C.; Gomez, JM.; Görlitz, O.; Lewen, H. OWL: Experiences and Directions. OWLED; 2006. Developing ontologies in OWL: an observational study.
5. Falconer, S.; Tudorache, T.; Noy, N. An analysis of collaborative patterns in large-scale ontology development projects. The 6th international Conference on Knowledge Capture (K-CAP); ACM. 2011. p. 25-32.
6. Ghidini C, Kump B, Lindstaedt S, Mahbub N, Pammer V, Rospocher M, Serafini L. Moki: The enterprise modelling wiki. The Semantic Web: Research and Applications. 2009:831–835.
7. Horridge M, Bechhofer S. The OWL API: A Java API for OWL ontologies. Semantic Web Journal. 2011; 2(1):11–21.
8. Krotzsch, M.; Vrandecic, D.; Volkel, M. Semantic MediaWiki. The 5th International Semantic Web Conference (ISWC06), volume 4273 of Lecture Notes in Computer Science; Athens, GA, USA. November 2006; Springer; p. 935-942.
9. Noy, N.; Chugh, A.; Liu, W.; Musen, M. A Framework for Ontology Evolution in Collaborative Environments. The 5th International Semantic Web Conference (ISWC-06); 2006. p. 544-558.
10. Noy N, Shah N, Whetzel P, Dai B, Dorf M, Griffith N, Jonquet C, Rubin D, Storey M, Chute C, et al. BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic acids research. 2009; 37(suppl 2):W170. [PubMed: 19483092]
11. Noy, NF.; Alani, H.; Stumme, G.; Mika, P.; Sure, Y.; Vrandecic, D., editors. Workshop on Social and Collaborative Construction of Structured Knowledge (CKC 2007) at WWW 2007; Banff, Canada. 2007.
12. Noy NF, Sintek M, Decker S, Crubézy M, Fergerson RW, Musen MA. Creating Semantic Web contents with Protégé-2000. IEEE Intelligent Systems special issue on Semantic Web Teconology. 2001; 16(2):60–71.
13. Redmond, T.; Smith, M.; Drummond, N.; Tudorache, T. Managing change: An ontology version control system. OWL: Experiences and Directions, 5th Intl. Workshop, OWLED; 2008; Karlsruhe, Germany. 2008.
14. Rubin D, Shah N, Noy N. Biomedical ontologies: a functional perspective. Briefings in bioinformatics. 2008; 9(1):75. [PubMed: 18077472]
15. Schandl T, Blumauer A. Poolparty: SKOS thesaurus management utilizing linked data. The Semantic Web: Research and Applications. 2010:421–425.
16. Stojanovic, L. PhD thesis. University of Karlsruhe; Germany: 2004. Methods and Tools for Ontology Evolution.
17. Tudorache, T.; Falconer, S.; Noy, NF.; Nyulas, C.; Üstün, BT.; Storey, M-A.; Musen, MA. Ontology Development for the Masses: Creating ICD-11 in WebProtégé. 17th Int. Conf. on Knowledge Engineering, volume LNAI; Lisbon, Portugal. Springer; 2010. p. 74-89.

18. Tudorache, T.; Falconer, S.; Nyulas, C.; Noy, N.; Musen, M. Will Semantic Web Technologies Work for the Development of ICD-11?. The 9th International Semantic Web Conference (ISWC 2010); Springer. 2010. p. 257-272.

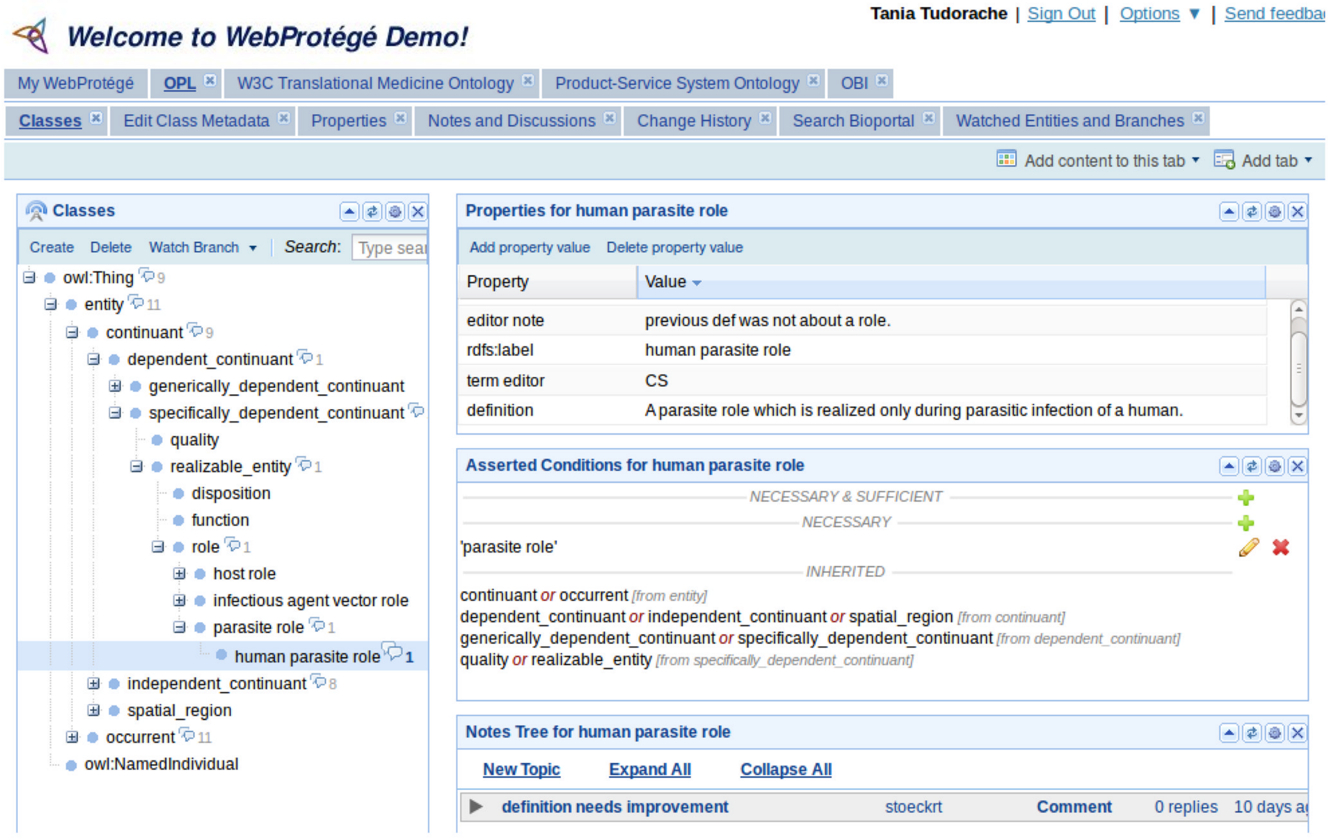
19. Tudorache, T.; Falconer, S.; Nyulas, C.; Storey, M.; Ustun, T.; Musen, M. Supporting the collaborative authoring of ICD-11 with WebProtégé. AMIA Annual Symposium Proceedings; American Medical Informatics Association; 2010. p. 802

20. Tudorache, T.; Musen, M. Collaborative Computational Technologies for Biomedical Research. Collaborative development of large-scale biomedical ontologies; p. 179-200.

21. Tudorache T, Noy N, Falconer S, Musen M. A knowledge base driven user interface for collaborative ontology development. 2011:411–414.

22. Tudorache, T.; Noy, NF.; Musen, MA. Supporting Collaborative Ontology Development in Protégé. Seventh International Semantic Web Conference, ISWC; 2008; Karlsruhe, Germany. 2008.

23. Zacharias, V.; Braun, S. SOBOLEO-social bookmarking and lighweight engineering of ontologies. Workshop on Social and Collaborative Construction of Structured Knowledge at WWW 2007; Banff, Canada. May 2007; 2007.



owl:Thing 9
entity 11
continuant 9
dependent_continuant 1
generically_dependent_continuant
specifically_dependent_continuant
quality
realizable_entity 1
disposition
function
role 1
host role
infectious agent vector role
parasite role 1
human parasite role 1
independent_continuant 8
spatial_region
occurrent 11
owl:NamedIndividual

Properties for human parasite role

Property	Value
editor note	previous def was not about a role.
rdfs:label	human parasite role
term editor	CS
definition	A parasite role which is realized only during parasitic infection of a human.

Asserted Conditions for human parasite role

NECESSARY & SUFFICIENT

NECESSARY

'parasite role'

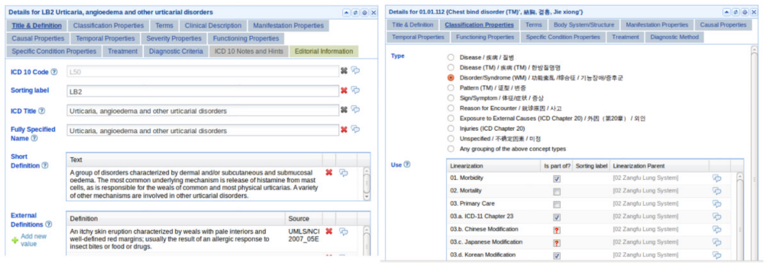
INHERITED

continuant *or* occurrent *[from entity]*
dependent_continuant *or* independent_continuant *or* spatial_region *[from continuant]*
generically_dependent_continuant *or* specifically_dependent_continuant *[from dependent_continuant]*
quality *or* realizable_entity *[from specifically_dependent_continuant]*

Notes Tree for human parasite role

definition needs improvement stoeckrt Comment 0 replies 10 days a

Fig. 1.
The user interface of WebProtégé showing the Ontology for Parasitic Lifecycle (OPL). The user interface consists of several tabs (pages), such as Classes, Properties, Change History, etc. Each page contains several portlets (e.g., Class tree, Properties, Restrictions, Notes, etc.). A user can add new tabs and portlets from the toolbar and save the configuration.



Details for LB2 Urticaria, angioedema and other urticarial disorders

ICD 10 Code	L50
Sorting label	LB2
ICD Title	Urticaria, angioedema and other urticarial disorders
Fully Specified Name	Urticaria, angioedema and other urticarial disorders

Short Definition

Text
A group of disorders characterized by dermal and/or subcutaneous and submucosal oedema. The most common underlying mechanism is release of histamine from mast cells, as is responsible for the weals of common and most physical urticarias. A variety of other mechanisms are involved in other urticarial disorders.

External Definitions

Definition	Source
An itchy skin eruption characterized by weals with pale interiors and well-defined red margins; usually the result of an allergic response to insect bites or food or drugs.	UMLS/NCI 2007_05E

Details for 01.01.112 {Chest bind disorder (TM)', 結胸, 결흉, Jie xiong'}

Type
- Disease / 疾病 / 질병
- Disease (TM) / 疾病 (TM) / 한방질명명
- [✓ Disorder/Syndrome (WM) / 功能紊乱 /综合征 / 기능장애/증후군]
- Pattern (TM) / 证型 / 변증
- Sign/Symptom / 体征/症状 / 증상
- Reason for Encounter / 就诊原因 / 사고
- Exposure to External Causes (ICD Chapter 20) / 外因（第20章） / 외인
- Injuries (ICD Chapter 20)
- Unspecified / 不确定因素 / 미정
- Any grouping of the above concept types

Use

Linearization	Is part of?	Sorting label	Linearization Parent
01. Morbidity	✓		[02 Zangfu Lung System]
02. Mortality	☐		[02 Zangfu Lung System]
03. Primary Care	☐		[02 Zangfu Lung System]
03.a. ICD-11 Chapter 23	✓		[02 Zangfu Lung System]
03.b. Chinese Modification	?		[02 Zangfu Lung System]
03.c. Japanese Modification	?		[02 Zangfu Lung System]
03.d. Korean Modification	✓		[02 Zangfu Lung System]

Fig. 2.

Knowledge-acquisition forms in WebProtégé tailored for domain experts showing the forms for the International Classification of Diseases (left) and the International Classification of Traditional Medicine (right). The WebProtégé forms allow the creation of customized user interfaces for specific projects by simply changing the layout configuration.

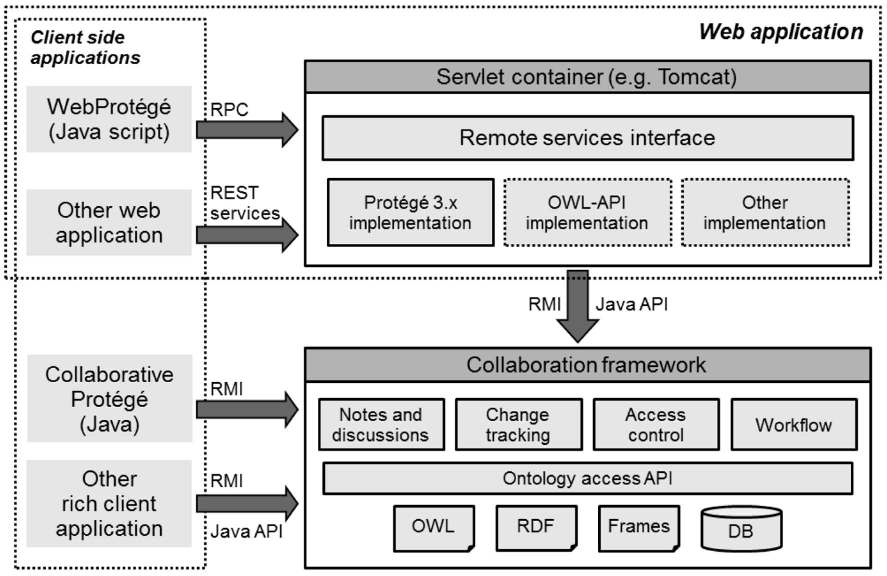


Fig. 3.
The architecture of WebProtégé. The user interacts with the client applications, such as WebProtégé in a Web browser, or Collaborative Protégé on the desktop. WebProtégé and other Web clients connect to a servlet engine that contains the application server logic. All clients will eventually connect to a common collaborative framework that provides services such as change tracking, storing of notes and discussion, access control, and so on.