Combination of DROOL rules and Protégé knowledge bases in the ONTO-H annotation tool

Corcho O. [1,5], Blázquez, M. [1], Niño M. [1], Benjamins V.R. [1], Contreras J. [1], García A. [2], Navas E. [2], Rodríguez J. [2], Wert C [2], Millán R. [3], Dodero J.M. [4]
[1] Intelligent Software Components, S.A. Spain. Contact author: ocorcho@isoco.com
[2] Residencia de Estudiantes. Spain
[3] Texto Digital S.L. Spain
[4] Universidad Carlos III. Spain
[5] Currently at University of Manchester. United Kingdom

Abstract. ONTO-H is a semi-automatic collaborative tool for the semantic annotation of documents, built as a Protégé 3.0 tab plug-in. Among its multiple functionalities aimed at easing the document annotation process, ONTO-H uses a rule-based system to create cascading annotations out from a single drag and drop operation from a part of a document into an already existing concept or instance of the domain ontology being used for annotation. It also gives support to the detection of name conflicts and instance duplications in the creation of the annotations. The rule system runs on top of the open source rule engine DROOLS and is connected to the domain ontology used for annotation by means of an ad-hoc programmed Java proxy.

1. Introduction

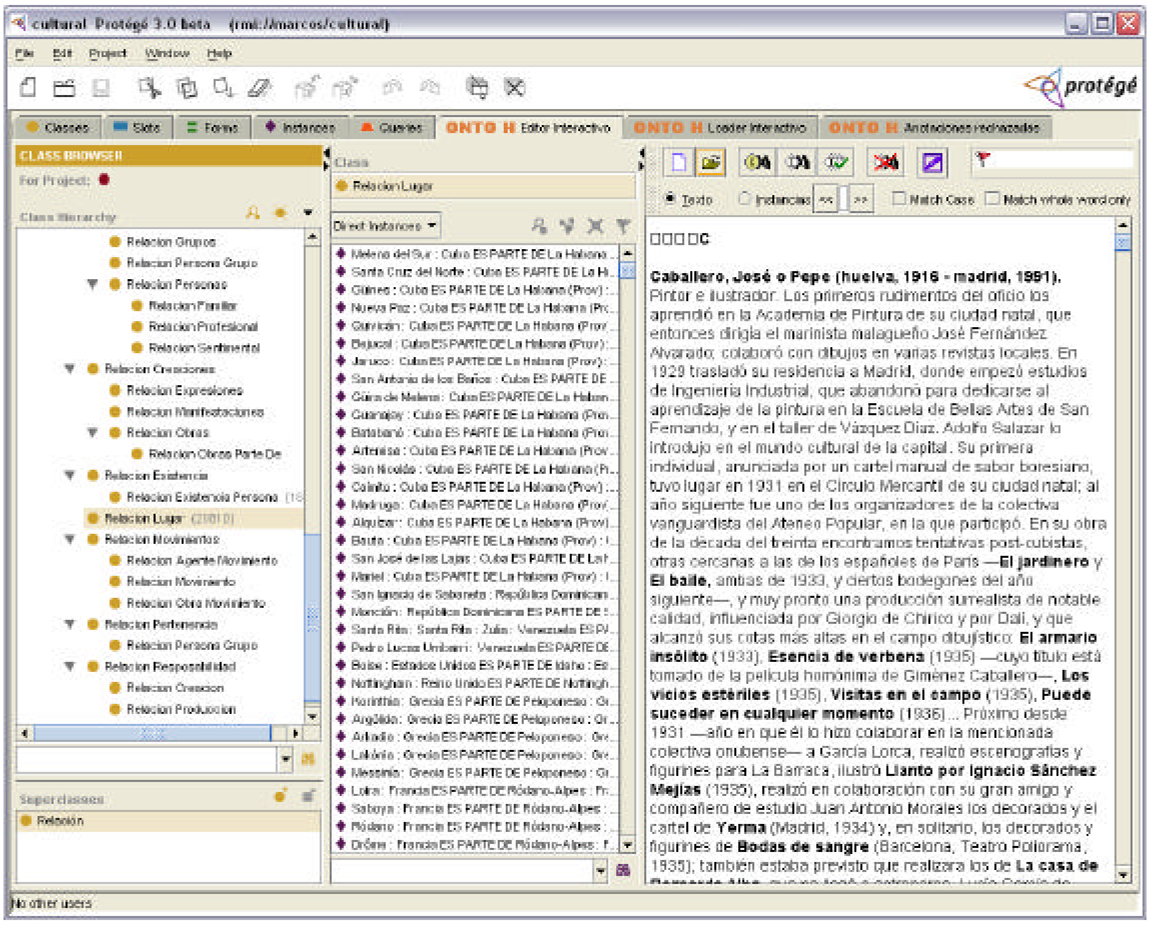
In the context of the Semantic Web, annotation is defined as a process that takes as input existing content (be it structured or unstructured documents, databases, XML files, etc.) and provides as output the semantic annotation of that content. Annotations are instances of existing domain ontologies, and normally have associated a set of pointers to the original content, so as to show where the instances were extracted from.

Annotation can be performed in several manners, ranging from completely manual to tool-assisted to fully automatic. The type of annotation approach to be chosen usually depends on the rate of structure that the content exhibits. More structure allows for more automation, while maintaining the quality of the annotations. Consequently, and especially in the case of unstructured or semi-structured sources, the annotation effort still remains as a serious barrier to the widespread of the Semantic Web [2].

In the context of the projects Esperonto, ONTO-H and SEGEPAC we have created ONTO-H [1], a Protégé 3.0 tab plug-in that gives support to the manual annotation task of documents written in the RTF format. This tool allows loading the source text to be annotated and performing standard ontology edition operations as provided by Protégé 3.0. Besides, it gives support to the addition of new ontology instances using selection and drag-and-drop functions. The user interface of this plug-in is shown in figure 1.

For instance, we can say that Picasso is an artist by selecting “Picasso” in the source text, dragging it to the ontology panel and dropping it over the concept “Person”. This operation creates an instance of the concept “Person” and pops up the usual Protégé form for creating instances with some information already filled in. The annotation process does not change the source text itself, but creates a link from the instance to the original string in the source text. We can also select a string like “inspired-by” and drag-and-drop it onto the corresponding relation in the ontology. Then the editor creates an instance of this relationship and pops up the corresponding Protégé 3.0 form where the user is prompted to complete the relation domain and range.

Other functions provided by the annotation tool are [1]: annotation suggestion, instance duplication detection and annotation search. ONTO-H also gives support to collaboration life cycles in the annotation process, defining the roles of annotators, knowledge engineers and reviewers.


cultural Protégé 3.0 beta (rmi://marcos/cultural)
File Edit Project Window Help
protégé
Classes Slots Forms Instances Queries
ONTO H Editor Interactivo
ONTO H Loader Interactivo
ONTO H Anotaciones rechazadas
CLASS BROWSER
For Project:
Class Hierarchy
Class
Direct Instances
Texto
Instancias
Match Case
Match whole word only
Superclasses
No other users

Relacion Grupos
Relacion Persona Grupo
Relacion Personas
Relacion Familiar
Relacion Profesional
Relacion Sentimental
Relacion Creaciones
Relacion Expresiones
Relacion Manifestaciones
Relacion Obras
Relacion Obras Parte De
Relacion Existencia
Relacion Existencia Persona (18
Relacion Lugar (21810)
Relacion Movimientos
Relacion Agente Movimiento
Relacion Movimiento
Relacion Obra Movimiento
Relacion Pertenencia
Relacion Persona Grupo
Relacion Responsabilidad
Relacion Creacion
Relacion Produccion

Relación

Relacion Lugar

Melena del Sur : Cuba ES PARTE DE La Habana ...
Santa Cruz del Norte : Cuba ES PARTE DE La H...
Güines : Cuba ES PARTE DE La Habana (Prov) : ...
Nueva Paz : Cuba ES PARTE DE La Habana (Pr...
Quivicán : Cuba ES PARTE DE La Habana (Prov ...
Bejucal : Cuba ES PARTE DE La Habana (Prov) : ...
Jaruco : Cuba ES PARTE DE La Habana (Prov) : ...
San Antonio de los Baños : Cuba ES PARTE DE ...
Güira de Melena : Cuba ES PARTE DE La Haban...
Guanajay : Cuba ES PARTE DE La Habana (Prov...
Batabanó : Cuba ES PARTE DE La Habana (Prov...
Artemisa : Cuba ES PARTE DE La Habana (Prov...
San Nicolás : Cuba ES PARTE DE La Habana (Pr...
Caimito : Cuba ES PARTE DE La Habana (Prov) : ...
Madruga : Cuba ES PARTE DE La Habana (Prov...
Alquízar : Cuba ES PARTE DE La Habana (Prov) ...
Bauta : Cuba ES PARTE DE La Habana (Prov) : ...
San José de las Lajas : Cuba ES PARTE DE La H...
Mariel : Cuba ES PARTE DE La Habana (Prov) : ...
San Ignacio de Sabaneta : República Dominicana ...
Monción : República Dominicana ES PARTE DE S...
Santa Rita : Santa Rita : Zulia : Venezuela ES PA...
Pedro Lucas Urribarri : Venezuela ES PARTE DE ...
Boise : Estados Unidos ES PARTE DE Idaho : Es...
Nottingham : Reino Unido ES PARTE DE Nottingh...
Korinthia : Grecia ES PARTE DE Peloponeso : Gr...
Argólida : Grecia ES PARTE DE Peloponeso : Gr...
Arkadia : Grecia ES PARTE DE Peloponeso : Gre...
Lakónia : Grecia ES PARTE DE Peloponeso : Gre...
Messinía : Grecia ES PARTE DE Peloponeso : Gr...
Loira : Francia ES PARTE DE Ródano-Alpes : Fr...
Saboya : Francia ES PARTE DE Ródano-Alpes : ...
Ródano : Francia ES PARTE DE Ródano-Alpes : ...
Drôme : Francia ES PARTE DE Ródano-Alpes : F...

□□□□C

Caballero, José o Pepe (huelva, 1916 - madrid, 1991). Pintor e ilustrador. Los primeros rudimentos del oficio los aprendió en la Academia de Pintura de su ciudad natal, que entonces dirigía el marinista malagueño José Fernández Alvarado; colaboró con dibujos en varias revistas locales. En 1929 trasladó su residencia a Madrid, donde empezó estudios de Ingeniería Industrial, que abandonó para dedicarse al aprendizaje de la pintura en la Escuela de Bellas Artes de San Fernando, y en el taller de Vázquez Díaz. Adolfo Salazar lo introdujo en el mundo cultural de la capital. Su primera individual, anunciada por un cartel manual de sabor boresiano, tuvo lugar en 1931 en el Círculo Mercantil de su ciudad natal; al año siguiente fue uno de los organizadores de la colectiva vanguardista del Ateneo Popular, en la que participó. En su obra de la década del treinta encontramos tentativas post-cubistas, otras cercanas a las de los españoles de París —**El jardinero** y **El baile,** ambas de 1933, y ciertos bodegones del año siguiente—, y muy pronto una producción surrealista de notable calidad, influenciada por Giorgio de Chirico y por Dalí, y que alcanzó sus cotas más altas en el campo dibujístico: **El armario insólito** (1933), **Esencia de verbena** (1935) —cuyo título está tomado de la película homónima de Giménez Caballero—, **Los vicios estériles** (1935), **Visitas en el campo** (1935), **Puede suceder en cualquier momento** (1936)... Próximo desde 1931 —año en que él lo hizo colaborar en la mencionada colectiva onubense— a García Lorca, realizó escenografías y figurines para La Barraca, ilustró **Llanto por Ignacio Sánchez Mejías** (1935), realizó en colaboración con su gran amigo y compañero de estudio Juan Antonio Morales los decorados y el cartel de **Yerma** (Madrid, 1934) y, en solitario, los decorados y figurines de **Bodas de sangre** (Barcelona, Teatro Poliorama, 1935); también estaba previsto que realizara los de **La casa de**

Figure 1: ONTO-H user interface.

In this paper we describe how ONTO-H has been implemented, especially focusing on the internal process used when a drag-and-drop operation is performed or when the user asks for annotation suggestions to the system. As a result of these operations the user interface shows one or several Protégé-standard instance creation forms with some information already filled in. In this process the instance duplication detection functions are fired so as to avoid the creation of duplicate instances in the annotation knowledge base.

The paper is structured as follows. Section 2 describes some annotation examples that show the complexity of the annotation process, giving support to our design and implementation decisions. Section 3 describes the internal architecture and the implementation details of the plug-in, showing how the DROOLS[1] rule engine is connected to the Protégé knowledge base. Section 4 provides some examples of the rule-sets used for each of the situations identified in section 2. Finally, section 5 gives some conclusions and identifies future work to be done.

2. Annotation examples and common situations

2.1 Annotation patterns

Massive knowledge acquisition tasks are usually based on typical annotation patterns, which are critical in complex domains. For instance, in the context of the cultural

[1] http://www.drools.org/

domain, and according to the IFLA standard[2], each time a new artistic work is being annotated, it makes sense to also create new instances for its expression and manifestation [3]. In IFLA “work” is defined as the idea of an artwork, like that of the Guernika painting of Picasso (Spanish Civil War and the Guernika bombing). The “expression” is a painting (it could as well be expressed in a poem or movie). The “manifestation” is the actual painting that can be enjoyed in the Reina Sofía museum in Madrid, Spain. These kinds of patterns stem from dependency relations between concepts in the domain ontology.

We can imagine that having to follow the same complex annotation patterns manually once and again when an artistic work is being annotated is a tedious and repetitive task for the person in charge of the annotation. Hence providing support for these patterns can improve the productivity of annotators and annotation reviewers.

2.2 Instance duplication

One of the most complex aspects of the annotation process (and of ontology population in general) deals with avoiding the duplication of instances due to the use of different formal names for them. This task usually requires agreeing beforehand on the naming conventions to be used for creating instances in the knowledge base and following these conventions strictly. Even with such conventions it is usually the case that duplicate instances are created, especially if the annotation process is performed by distributed persons and during a long period of time.

Alternative ways to detect possible duplicates in large knowledge bases have to be found. A domain independent way to detect a large number of duplicates in document-based annotation consists in storing the pieces of text that have been used for the creation of an instance and using them to determine whether a new instance to be created has already that name or a similar one. For example, in the cultural domain each author, place, work, etc. can possess a number of names, variable in the time line or different depending on the relation they participate in (an author can write a book using one pseudonym, then use his legal name when attending an exposition and use an acronym when writing a new book with two colleagues). All these names should point to the same instance of the class “Person”.

2.3 Annotation suggestions

The efficiency and accuracy of the annotation process can be largely improved by providing annotation suggestions to the users. That is, the user can ask for advice for selected parts of the text, and the system provides suggestions about possible annotations that could be performed, based on the information that has been already stored in the annotation knowledge base and also on natural language resources such as lexical and morphological analysers, synonym dictionaries, etc. If the selected word or a part of the selected text is identified as a possible new occurrence of an existing instance the system can suggest it to the annotator, who can decide which action to perform: (i) add a completely new instance, (ii) add a new occurrence to an existing instance (that is, add a new reference from the instance to the document), or (iii) discard any ontology modification. The more instances the ontology contains, the better suggestions the system can usually offer.

[2] Federation of Library Associations and Institutions: http://www.ifla.org/

3. ONTO-H Architecture and Implementation

As described in the introduction, ONTO-H has been created as a tab plug-in of Protégé 3.0.Consequently, the basic ontology management functionalities are provided by the Protégé framework while the plug-in just adds the annotation-specific functionalities: opening and closing documents, creating instances out of those documents by performing drag and drop operations, avoiding duplicate instance creation, searching for annotations inside a document, providing annotation suggestions based on natural language analysers and dictionaries, and giving support to the distributed annotation lifecycle with actors like annotators, knowledge engineers and annotation reviewers (this last functionality is explained in detail in [1]).

We have decided to use a rule-based approach for implementing some of the annotation functionalities described before. The reason for this is twofold:

- On the one hand, some of these functionalities are **domain dependent** (suggestions and automatic support for complex annotation patterns). If these functionalities were hard-coded in the ONTO-H application code a change in the domain used for the annotation would mean that new code would have to be created specifically for the new domain, with the subsequent recompilation and reinstallation of the system.
- On the other hand, the **domain independent** functionalities, such as the detection of possible duplicates and the different behaviours to be followed when a drag and drop operation is performed over a concept or instance, have been constantly refined during the development and evaluation phases of the system, since most of them are based on heuristics.

Rule based systems have proven useful to provide this behaviour of the annotation tool, since they provide a declarative way to express the actions to be performed in different situations and they allow expressing easily the preconditions that have to be satisfied for performing a set of actions on the knowledge base or on the user interface.

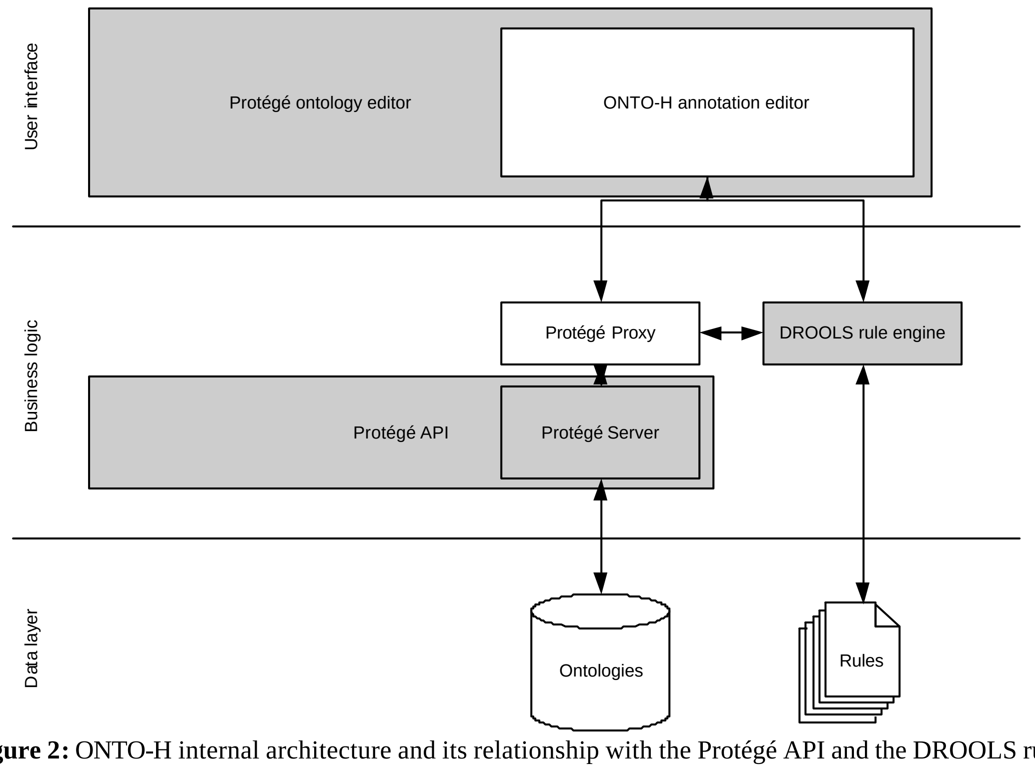


Figure 2: ONTO-H internal architecture and its relationship with the Protégé API and the DROOLS rule engine.

Finally, to better access the Protégé API from the rule based system and to allow a better management of the evolution of that API without affecting too much the internal code of the annotation system, we have created a proxy of the current Protégé API, which is used inside the rules for checking the rule preconditions and for updating the knowledge base.

Figure 2 shows a graphical view of the architecture of ONTO-H, showing the relationships with the Protégé API and with the DROOLS rule engine, where the rules are executed.

As an example of how rules are used from the system, we show one of the rules that can be fired when the annotator drags a text in the document and drops it into an ontology concept. This rule is described in detail in the next section.

```
<rule name="If the instance does not exist, create and fill its attributes">
  <parameter identifier="obj">
    <java:class type="com.isoco.ieditor.DropObj">com.isoco.ieditor.DropObj</java:class>
  </parameter>

  <java:condition>
   (com.isoco.ieditor.Environment.getObject().getBaseCon().
       esSubclaseDe(obj.clase.getNombre(),
    com.isoco.ieditor.Constants.NOMBRE_CLASE_PADRE_DENOMINACION).
       equals(Boolean.FALSE) &amp;&amp;  obj.tipoDuplicidad==0)
  </java:condition>

  <java:consequence>
    import com.isoco.proxyprotege.*;
    Instancia inst = obj.clase.nuevaInstancia(obj.seleccion);
    inst.setAtributoValor("referencia", obj.seleccion);
    inst.setAtributoValor("enlace_fuente", obj.posicion);
    inst.setAtributoValor("fuente", obj.seleccion);
    inst.setAtributoValor("fecha_anotacion",obj.fecha_anotacion);
    obj.instancia = inst;
    obj.resultado = "NEW INSTANCE";
    obj.instancia.mostrar();
  </java:consequence>
</rule>
```

The most important implementation details are that we have used the Protégé 3.0 server, since the tool provides collaborative functionalities for managing annotators, knowledge engineers and annotation reviewers. The Protégé 3.0 server uses a MS Access database for storing and accessing the ontologies. We have preferred to use a database for scalability reasons, since in the cultural domain the number of annotations and the size of the domain ontologies are quite large. With respect to the rule engine, we have used version 2.0 (beta 17) of DROOLS.

4 Rule sets

As we have described above, there are different types of situations that can arise when the annotator performs an operation with the ONTO-H user interface. We have identified several groups of rules that are activated according to the events fired by the user interface or fired by the execution of other rules:

- Drag and drop rules, either over an ontology concept or over an ontology instance (that is, over an existing annotation).
- Conflict resolution (instance duplication) rules.
- Cascading instance creation rules.
- Annotation suggestion rules.

The first two groups of rules are completely domain independent, since their functionalities are valid for any domain. The other two sets of rules are usually domain dependent, since the annotation creation patterns and the suggestions proposed by the system will be based on the domain of the annotations performed, as described below.
Let us now see with more detail the rules that are defined in each group.

4.1 Drag and drop rules

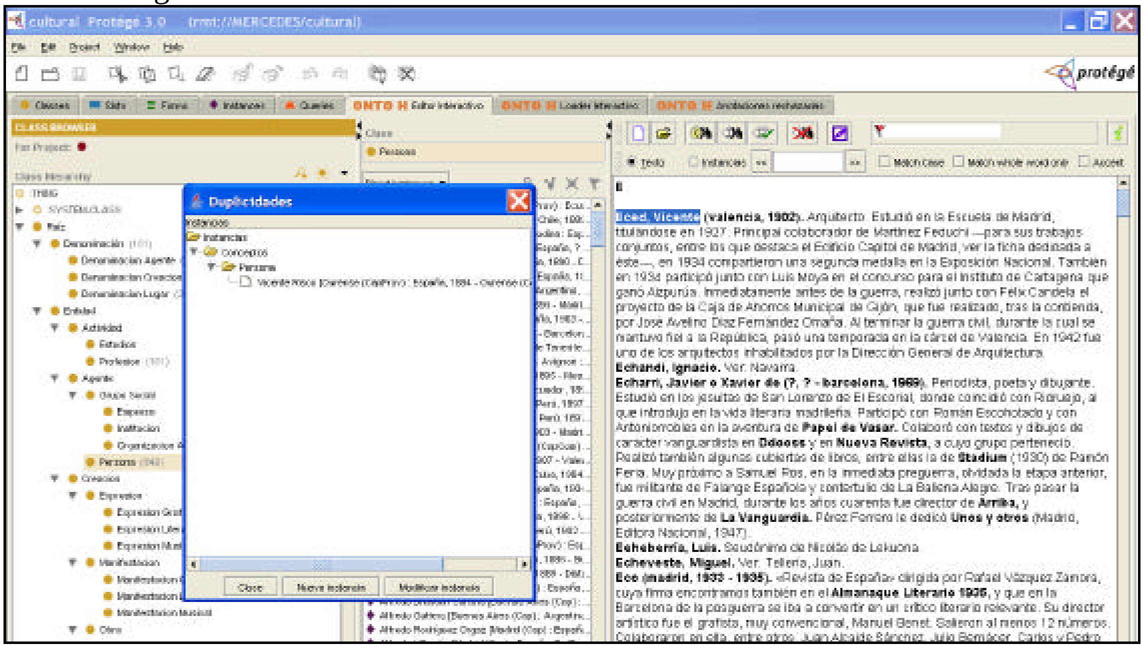
When the user drags a selected piece of text from the document being annotated and drops it onto an ontology concept, an instance of that concept must be created, with a reference to the position of the selection inside the text and with an instance name that contains the selected text.
Several situations may arise here:

- If the instance name did not exist already among the annotations stored in the knowledge base, the annotation system will create it.
- If the instance name already existed among the annotations of any of the subclasses of the concept where the drag and drop operation was performed, then the system prompts the user asking whether he or she was referring to that specific instance.
- If the instance name already existed among the annotations of any other concept in the ontology, then the system prompts the user asking whether they are different instances or not, and hence whether it has to create a new instance or not.

If the user drags a selected piece of text from the document being annotated and drops it onto an ontology instance, a new reference to the piece of text will be created for that instance, and possibly a new alternate name for it, in the case that the information in the piece of text is different from any of the current instance names.

4.2 Conflict resolution (instance duplication) rules

As aforementioned once the annotator performs a drag and drop operation from the document to the concept or instance panes, the system analyses whether the corresponding instance may already exist in the knowledge base using the same name. This is quite easy to check and obviously avoids easily the duplication of an instance in the knowledge base.



Eced, Vicente (valencia, 1902). Arquitecto. Estudió en la Escuela de Madrid, titulándose en 1927. Principal colaborador de Martínez Feduchi —para sus trabajos conjuntos, entre los que destaca el Edificio Capitol de Madrid, ver la ficha dedicada a éste—, en 1934 compartieron una segunda medalla en la Exposición Nacional. También en 1934 participó junto con Luis Moya en el concurso para el Instituto de Cartagena que ganó Aizpurúa. Inmediatamente antes de la guerra, realizó junto con Félix Candela el proyecto de la Caja de Ahorros Municipal de Gijón, que fue realizado, tras la contienda, por José Avelino Díaz Fernández Omaña. Al terminar la guerra civil, durante la cual se mantuvo fiel a la República, pasó una temporada en la cárcel de Valencia. En 1942 fue uno de los arquitectos inhabilitados por la Dirección General de Arquitectura.
Echandi, Ignacio. Ver: Navarra.
Echarri, Javier o Xavier de (?, ? - barcelona, 1969). Periodista, poeta y dibujante. Estudió en los jesuitas de San Lorenzo de El Escorial, donde coincidió con Ridruejo, al que introdujo en la vida literaria madrileña. Participó con Román Escohotado y con Antoniorrobles en la aventura de **Papel de Vasar.** Colaboró con textos y dibujos de carácter vanguardista en **Ddooss** y en **Nueva Revista**, a cuyo grupo perteneció. Realizó también algunas cubiertas de libros, entre ellas la de **Stadium** (1930) de Ramón Feria. Muy próximo a Samuel Ros, en la inmediata preguerra, olvidada la etapa anterior, fue militante de Falange Española y contertulio de La Ballena Alegre. Tras pasar la guerra civil en Madrid, durante los años cuarenta fue director de **Arriba,** y posteriormente de **La Vanguardia.** Pérez Ferrero le dedicó **Unos y otros** (Madrid, Editora Nacional, 1947).
Echeberría, Luis. Seudónimo de Nicolás de Lekuona.
Echeveste, Miguel. Ver: Tellería, Juan.
Eco (madrid, 1933 - 1935). «Revista de España» dirigida por Rafael Vázquez Zamora, cuya firma encontramos también en el **Almanaque Literario 1935**, y que en la Barcelona de la posguerra se iba a convertir en un crítico literario relevante. Su director artístico fue el grafista, muy convencional, Manuel Bonet. Salieron al menos 12 números. Colaboraron en ella, entre otros, Juan Alcaide Sánchez, Julio Bombácer, Carlos y Pedro

Figure 3: Instance duplication warning message.

However, there are situations where the instance that the annotator wants to create is already in the knowledge base with a different name. A set of specialised rules is used to detect these situations warning the annotator about possible conflicts or duplications in the knowledge base. These rules check whether the knowledge base contains already instances that refer to the same piece of text, to exactly the same content available inside the piece of text or to some of the terms contained in the piece of text that the user selected in the drag and drop operation.

In the example shown in figure 2 the system alerts the user about the possible duplication of an instance in the knowledge base when the annotator has selected the "Eced, Vicente" piece of text and has dropped it into the concept "Person". The knowledge base already contains somebody called "Vicente Risco", which might refer to the same person or not.

4.3 Cascading instance creation rules

An example of the need of cascading instance creation patterns was outlined in section 2 with an example of how annotations of a work in the cultural domain have to be created by the instantiation of several concepts, according to IFLA.

For this purpose a set of domain specific rules have been created in the context of the cultural domain, where ONTO-H has been evaluated. These rules are evaluated whenever a new instance is created in the knowledge base and they are fired depending on whether their corresponding preconditions are correctly evaluated. In this domain the following rules have been created:

- If an instance of the concept "Manifestación" is created, then an instance of the concept "Expresión" must be also created, with a relationship "tieneManifestación" instantiated between the latter and the former. Similarly with its subconcepts:
    - If an instance of "Manifestación Gráfica" is created, then an instance of "Expresión Gráfica" must be also created.
    - If an instance of "Manifestación Literaria" is created, then an instance of "Expresión Literaria" must be also created.
    - If an instance of "Manifestación Musical" is created, then an instance of "Expresión Musical" must be also created.
- If an instance of the concept "Expresión" is created, then an instance of the concept "Work" is created, with a relationship "tiene Expresión" instantiated between the latter and the former. Similarly with its subconcepts:
    - If an instance of "Expresión Gráfica" is created, then an instance of "Obra Gráfica" must be also created.
    - If an instance of "Expresión Literaria" is created, then an instance of "Obra Literaria" must be also created.
    - If an instance of "Expresión Musical" is created, then an instance of "Obra Musical" must be also created.

The activation of all these rules results in the appearance of several instance edition forms in the user interface, with the relationships between all these instances already filled in and with most of the information inside the instances pre-filled.

4.4 Annotation suggestion rules

As described in section 2, the annotation process can be improved by providing suggestions to the annotator. These suggestions are usually domain dependent and are based on the use of natural language resources. As an example, for the cultural domain we have the following rules, among others:

- If a word starts with a capital letter without being the first word of the sentence then the system suggests that it could be referring to an instance of the concept "Person", "Work" or "Place".
- If a set of consecutive words start with capital letters then the system suggests that they could be referring to instances of the concept "Person", "Work" or "Place".
- If a word or set of words appears in a synonym dictionary as a synonym of another word that has been already annotated, then the system suggests that it could be another name for the existing instance and also suggests the addition of a reference to that part of the text.

5. Conclusions and future work

In this paper we have shown how we can combine a rule-based system with a Protégé knowledge base to build a tab plug-in that can be used for the semantic annotation of documents.

The annotation process is usually based on a set of heuristics that are followed by the person in charge of creating the annotations. These heuristics are used for detecting pieces of the text source that can be candidates for annotations, detecting possible duplicates when creating instances of the domain ontologies used for the annotation process, and improving the efficiency of annotation by giving support to complex annotation patterns.

Due to the heuristic behaviour of the annotation process and to the fact that in some cases the heuristics depend on the domain of the source documents, the annotation tool relies on a rule-based system (implemented on top of the DROOLS engine) that is driven by the events fired by the user interface. The rules can be easily modified when the annotation tool is used in a different domain, when new annotation patterns are identified or in the case that refinements of the current behaviours are to be provided.

Though the use of the DROOLS engine has proven to be good for the purposes of our system and has a good behaviour even with large knowledge bases, we plan to explore the use of other rule-based systems, especially those that will support the emerging SWRL, so as to provide a Semantic Web compliant solution for our tool and to allow a better reuse of our rules in other applications. In this direction, we also plan to explore the use of the Protégé-OWL API instead of the core Protégé API by modifying our Protégé API proxy, described in section 3.

Besides, since the tool uses documents specified in the RTF format another future piece of work will be the exploitation of the document layout during the annotation process. The current architecture allows this kind of exploitation, since we can create sets of annotation rules specifically devoted to exploiting such characteristics of the source documents.

Acknowledgements

This work has been supported by the EU IST project Esperonto (IST-2001-34373) and the Spanish PROFIT projects ONTO-H and SEGEPAC.

References

1. Benjamins V.R., Contreras J., Blázquez, M., Niño M., García A., Navas E., Rodríguez J., Hernández F., Wert C, Dodero J.M. ONTO-H: a Collaborative Semiautomatic Annotation Tool. Available at: http://protege.stanford.edu/conference/2005/submissions/abstracts/accepted-abstract-benjamins.pdf
2. Benjamins VR, Contreras J, Corcho O, Gómez-Pérez A (2002) Six challenges for the Semantic Web. *KR2002 Workshop on Semantic Web*. Toulouse, France.
3. Dodero JM, Contreras J, Benjamins VR (2004) D9.2: Test Case Ontology Specification Cultural Tour. Esperonto Project, www.esperonto.net