FORMULIS: Dynamic Form-Based Interface For Guided Knowledge Graph Authoring

Pierre Maillot, Sébastien Ferré, Peggy Cellier, Mireille Ducassé, Franck Partouche

FORMULIS: Dynamic Form-Based Interface For Guided Knowledge Graph Authoring

Pierre Maillot[1], Sébastien Ferré[1], Peggy Cellier[2], Mireille Ducassé[2], and Franck Partouche[3]

[1] IRISA/Université de Rennes 1**
[2] IRISA/INSA Rennes
Campus de Beaulieu, 35042 Rennes cedex, France
{firstname}.{lastname}@irisa.fr
[3] IRCGN, 5 Boulevard Hautil, 95000 Cergy, France
franck.partouche@gendarmerie.interieur.gouv.fr

Abstract. Knowledge acquisition is a central issue of the Semantic Web. Knowledge cannot always be automatically extracted from existing data, thus contributors have to make efforts to create new data. In this paper, we propose FORMULIS, a dynamic form-based interface designed to make RDF data authoring easier. FORMULIS guides contributors through the creation of RDF data by suggesting fields and values according to the previously filled fields and the previously created resources.

1 Introduction

Manual acquisition of RDF data requires an effort either from contributors or from Semantic Web experts. In systems based on contributors' efforts, such as WebProtégé [4], the contributors have to learn Semantic Web principles, and how to use the specialized interfaces based on these principles. In systems based on Semantic Web experts' efforts, such as ActiveRaUL [1] or OntoWiki [2], the data input interfaces are easier to use, but have to be set up by the experts. In case of data evolution, contributors depend on experts to adapt the interface.

In this paper, we propose FORMULIS, a system that offers the user-friendliness of forms while guiding users with dynamic suggestions based on existing data. The contribution is threefold. First, the forms can be nested as deeply as necessary. Hence, it is possible to create at once several related entities, linked by property paths, by building nested descriptions. Second, the forms need no set up by experts because their content is based on SEWELIS suggestions [3]. Fields and values are suggested by taking into account all the fields and values already entered in the base and in the current form, hence offering dynamic and accurate suggestions. Last but not least, the forms can be extended with new fields and new sub-forms at any time, according to user needs and data evolution, with a simple click. New forms can also be created for new classes of resources.

** This work is partially supported by ANR project IDFRAud (ANR-14-CE28-0012)

FORMULIS has been used in a real setting. Six forensic experts from the forged ID unit of the document department of IRCGN[4] put into a knowledge base the description of thirty five forged Portuguese ID cards seized during a police operation. That experiment was successful despite experts having no knowledge of the Semantic Web, different level of expertise in forged ID detection, and different specialties. All were able to produce good quality data.

2 FORMULIS: Guided RDF Authoring Through Forms

FORMULIS aims at facilitating RDF authoring without knowledge of RDF by proposing a familiar interface, forms, and by dynamically suggesting values from current and previous inputs. On the one hand, graph-based data representations tend to become quickly illegible with the size of the graph and text-based notations, such as Turtle, need prior knowledge to be understood. On the other hand, forms are a common interface for data input. FORMULIS generates a data-driven form-based interface for the production of RDF data.

Interaction loop. At each iteration, the system computes suggestions to fill the form further, and the user selects one suggestion. The form is initially empty, and it is progressively filled through interaction steps, until it is judged complete by the user. Suggestions are computed with the help of the query relaxation and query evaluation mechanisms of SEWELIS [3]. This requires first to translate the partially-filled form into an initial query that is supposed to retrieve the possible values for the field under focus, given the already filled fields. Query relaxation of that initial query is in general necessary to ensure the existence of results. In order to propose legible suggestions, those results have then to be rendered as form elements, i.e. user interface widgets and controls. Finally, each time a user selects an element or activates a control, the form is modified by filling a field, adding a field, inserting a nested form, etc. From there, a new interaction step can start.

Nested forms and their RDF counterpart. A form F in FORMULIS is composed of an RDF class C, a resource URI R, and a set of RDF properties $p_1, \ldots, p_n$, along with their sets of values $V_1, \ldots, V_n$. Class C (e.g., `dbo:Film`) determines the type of the resource being described, and resource R (e.g., `dbr:Corpse_Bride`) determines its identity. Each property p_i (e.g., `dbo:director`) defines a field of the form. Each field may have one or several values. Indeed, unlike tabular data, the RDF data model allows a resource to have several values for the same property, and it is therefore important to account for this in RDF editing tools. An example about films is the property `dbo:starring` relating films to actors. A field value can be one of: void, i.e. the field has not yet been filled ; a literal, e.g. a string ; an RDF resource already present in the knowledge base (e.g., `dbr:Tim_Burton`) ; or a nested form to create and describe a new RDF resource. Those definitions of forms and field values exhibit the recursive nature of forms in FORMULIS, which allows for cascading descriptions of resources.

[4] "Institut de Recherche Criminelle de la Gendarmerie Nationale", Forensic Sciences Institute of the French Gendarmerie.

Fig. 1. Screenshot of FORMULIS during the creation of a film directed by Tim Burton using the DBpedia vocabulary, with parts in dotted frames for explanation purposes.

User interface. Figure 1 shows an example of nested form, as it is displayed in the user interface of FORMULIS. In this example, the form is used to create the description of the film "Corpse Bride". The first step done by the user was to choose "Film" among the available classes in order to get the initial form to create a new film. The suggested fields of the initial form are based on the description of existing films in the base. The user interface follows the classical layout of a form with each field labeled by a property of the described resource. In frame (A), the "director" and "release date" are filled with RDF resources, respectively the URI labeled "Tim Burton" and the literal "2005". Note that RDFS labels are used to display URI in forms. Frame (B) shows two nested forms: one for the description of a new person as "music composer" of the current film ("Danny Elfman"), and another for the description of his birth place ("United States"). In frame (C), the resource labeled "Helena Bonham Carter" has been entered as starring in the movie. In another field for the "starring" property, the user is given six suggestions corresponding to actors that have been starring in movies similar to "Corpse Bride", either because they were also directed by "Tim Burton", released in 2005 and/or starring "Helena Bonham Carter". Below those suggestions, the user is also given the possibility to create

a new resource ("New element"), or to ask for more suggestions by not taking into account the previously filled fields ("More general").

Suggestions. In order to suggest values for a field, we need to translate the form to a SPARQL query that retrieves existing values for that field in the knowledge base. Suppose that the user has set the focus on the "starring" field in Figure 1 because she wants to get suggestions for the actors of the film. We could retrieve all values of the "starring" property in the base, using query: `SELECT ?v WHERE {?x dbo:starring ?v}`. However, this would make suggestions unspecific. We make suggestions dynamic and more accurate by taking into account all filled fields. The principle is to generate a SPARQL query whose graph pattern is the current description: `SELECT ?v WHERE {?x a dbo:Film; dbo:director dbr:Tim_Burton; dbo:releaseDate "2005"; dbo:musicComposer [a dbo:Person; dbo:birthPlace [a dbo:Country]]; dbo:starring dbr:Helena_B_Carter, ?v.}`. That query is sent to SEWELIS for relaxation and evaluation.

Cold start. In the case of an empty base or a new class, the creation of the first instance of the new class requires the creation of a new form and its fields, in addition to entering field values. That first instance is enough to generate the forms for the following instances. The user can also extend generated forms with new fields, as it can be seen in Figure 1 at the bottom of each form ("New line").

3 Demo

The demo will show that the accuracy of suggestions improves when more fields are filled, and when more resources are put in the knowledge base. That accuracy will be shown by the suggestions of values reflecting patterns commonly found in data, for example the fact that actor Johnny Depp is often starring with Helena B. Carter in films directed by Tim Burton. Participants will be given the opportunity to enter the description of recent movies, starting from DBpedia films. The demo will also show the cold start case with the creation of new forms. FORMULIS is available in beta version at http://servolis.irisa.fr:8080/formulis/.

References

1. Anila Sahar Butt, Armin Haller, Shepherd Liu, and Lexing Xie. ActiveRaUL: A Web form-based User Interface to create and maintain RDF data. In *Int. Semantic Web Conf., Posters & Demonstrations Track-Volume 1035*, pages 117–120, 2013.
2. Philipp Frischmuth, Michael Martin, Sebastian Tramp, Thomas Riechert, and Sören Auer. OntoWiki – an authoring, publication and visualization interface for the data web. *Semantic Web*, 6(3):215–240, 2015.
3. Alice Hermann, Sébastien Ferré, and Mireille Ducassé. An interactive guidance process supporting consistent updates of RDFS graphs. In *Knowledge Engineering and Knowledge Management*, pages 185–199. Springer, 2012.
4. Tania Tudorache, Csongor Nyulas, Natalya F Noy, and Mark A Musen. WebProtégé: A collaborative ontology editor and knowledge acquisition tool for the web. *Semantic web*, 4(1):89–99, 2013.