Aide à la création d'objets dans une base RDF(S) avec des règles de relaxation

Alice Hermann, Sébastien Ferré, Mireille Ducassé

Aide à la création d'objets dans une base RDF(S) avec des règles de relaxation

Alice Hermann[1], Sébastien Ferré[2] et Mireille Ducassé[1]

[1] IRISA – INSA Rennes, Campus de Beaulieu, 35708 Rennes cedex 7, FRANCE
alice.hermann@irisa.fr, mireille.ducasse@irisa.fr

[2] IRISA – Université de Rennes 1, Campus de Beaulieu, 35042 Rennes cedex, FRANCE
sebastien.ferre@irisa.fr

Résumé : Quand un utilisateur crée un nouvel objet dans le Web sémantique, les outils existants n'exploitent ni les objets existants et leurs propriétés, ni les propriétés déjà connues du nouvel objet. Nous proposons UTILIS, une méthode d'aide à la création de nouveaux objets. UTILIS cherche des objets similaires au nouvel objet en appliquant des règles de relaxation à sa description. Les propriétés des objets similaires servent de suggestions pour compléter la description du nouvel objet. Une étude utilisateur menée avec des étudiants en master montre que les suggestions d'UTILIS ont été utilisées. Les utilisateurs ont trouvé les suggestions pertinentes : dans la plupart des cas, ils pouvaient trouver l'élément recherché dans les trois premiers ensembles de suggestions. De plus, ils les ont appréciées, car la majorité souhaitent les avoir dans un éditeur de données du Web sémantique.

Mots-clés : Web sémantique, RDFS, création, acquisition interactive de connaissances, règles de relaxation, interaction utilisateur.

1 Introduction

La mise à jour des données du Web sémantique (SW) est cruciale pour prendre en compte les informations régulièrement découvertes. Cela est, cependant, fastidieux et, en pratique, les données du SW sont rarement mises à jour par les utilisateurs. Or, dans le Web 2.0, les utilisateurs contribuent à la production des données, ce qui prouve que la motivation n'est pas un problème. Des modèles existent pour rapprocher le SW et le Web 2.0, en reliant par exemple les étiquettes créées par les utilisateurs avec le vocabulaire du SW (Passant & Laublet, 2008; Limpens *et al.*, 2009). Cependant, il existe encore des difficultés à intégrer les données du Web

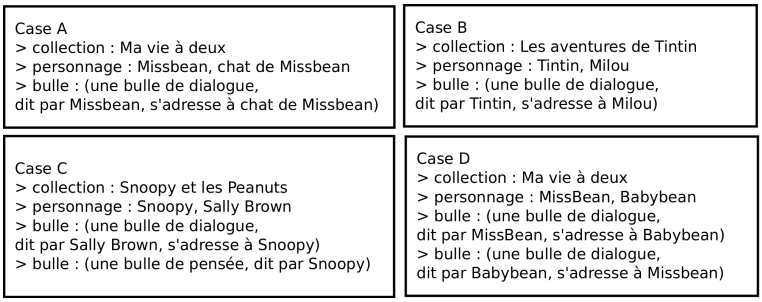

FIGURE 1 – Extrait de la base d'indexation de bande dessinée

2.0 dans le SW, comme des problèmes d'alignements automatiques entre les étiquettes des utilisateurs et les ressources du SW. De plus les données du SW sont plus riches que les étiquettes des utilisateurs. En effet, le SW permet une représentation plus complexe des données et des requêtes plus élaborées. Il est donc important que les utilisateurs puissent directement créer des données sous un format du SW.

Cet article présente UTILIS (Updating Through Interaction in Logical Information Systems), une méthode qui utilise les objets existants et la description partielle d'un nouvel objet, pour aider l'utilisateur à créer le nouvel objet. UTILIS cherche les objets similaires au nouvel objet, c'est-à-dire ayant des propriétés et valeurs en commun. Ces objets et leurs propriétés servent de suggestions pour compléter la description du nouvel objet.

Dans la suite, les exemples et l'expérience sont liés à la mise à jour d'une base d'indexation de cases de bandes dessinées. Un extrait est montré en figure 1. La case A est une case de la collection *Ma vie à deux*, avec Missbean et son chat ainsi qu'une bulle de dialogue dite par Missbean, à son chat.

La principale contribution de ce papier est un processus d'interaction qui aide l'utilisateur à créer de nouveaux objets en suggérant des propriétés et des valeurs adaptées finement à chaque objet en cours de création. Les propriétés déjà connues d'un nouvel objet sont utilisées pour en suggérer d'autres. Supposons qu'un utilisateur ajoute une case et indique la collection de bande-dessinée à laquelle elle appartient, cette dernière peut aider à suggérer des personnages, il est probable que cette case et celles de la même collection aient des personnages en commun. Ce processus utilise un ensemble de règles de relaxation, inspirées par les travaux d'Hurtado *et al.* (2008), et un algorithme efficace pour calculer des suggestions. Un avantage de notre approche est que la définition d'une ontologie n'est pas

nécessaire pour calculer les suggestions, même si UTILIS peut en utiliser une pour améliorer ses suggestions lorsqu'elle est disponible.

Une étude utilisateur menée avec des étudiants montre que ces derniers ont utilisé les suggestions d'UTILIS. Ils les ont trouvées pertinentes, en effet, dans la plupart des cas, ils pouvaient trouver l'élément recherché dans les trois premiers ensembles de suggestions. De plus, ils les ont appréciées, car 14 sur 18 souhaitent les avoir dans un éditeur de données du SW.

La section 2 donne des définitions relatives à des langages du SW et aux systèmes d'information logiques utilisés pour l'interaction avec l'utilisateur. La section 3 spécifie notre approche, UTILIS. La section 4 présente la création de la description d'une case. La section 5 présente l'étude utilisateur. La section 6 compare notre approche à l'état de l'art.

2 Préliminaires

RDF, RDFS et SPARQL. RDF et RDFS sont des langages du Web sémantique qui permettent l'interopérabilité entre outils. Les éléments de base de ces langages sont les ressources et les triplets. Un triplet *(s, p, o)* peut être lu comme une phrase où *s* est le sujet, *p* est le verbe, appelé prédicat, et *o* est l'objet. Un triplet est composé de 3 ressources. Une ressource peut être soit un URI (nom absolu d'une ressource), un littéral ou une ressource anonyme. RDF permet de représenter des données. Par exemple, dans la base d'indexation, le triplet (<*CaseK*>, :*personnage,* <*Missbean*>) peut être lu comme "La case K a pour personnage Missbean". RDF possède un vocabulaire prédéfini permettant de représenter l'appartenance à une classe de ressources (*rdf*:*type*), la hiérarchie entre les classes (*rdfs*:*subClassOf*) et entre les propriétés (*rdfs*:*subPropertyOf*). Par exemple, le triplet (*:Bulle1, rdf*:*type, :BulleDeDialogue*) dit que "La bulle 1 est de type BulleDeDialogue", ou plus simplement "La bulle 1 est une bulle de dialogue". La ressource :*BulleDeDialogue* est une classe. Le triplet (:*BulleDeDialogue, rdfs*:*subClassOf,* :*Bulle*) dit que "La classe BulleDeDialogue est une sous classe de Bulle", ou " Chaque bulle de dialogue est une bulle". RDFS est un langage de représentation des connaissances avec syntaxe, sémantique et inférence (Hitzler *et al.*, 2009). Par inférence, les deux triplets précédents permettent de déduire le triplet (<*Bulle1*>, *rdf*:*type,* :*Bulle*). Dans la suite, les descriptions des objets sont écrites en notation Turtle (Beckett *et al.*, 2010) par souci de concision. Par exemple, la description d'une nouvelle case sous forme de triplets ((<*CaseK*>, *rdf*:*type, :case*) (<*CaseK*>, :*personnage,* <*Missbean*>)) s'écrit en

Français	Quelles sont les cases avec au moins une bulle dite par Missbean ?
SPARQL	SELECT ?x WHERE { ?x a :case . ?x :bulle ?y . ?y :ditPar <MissBean>}

FIGURE 2 – Une question et sa traduction en requête SPARQL.

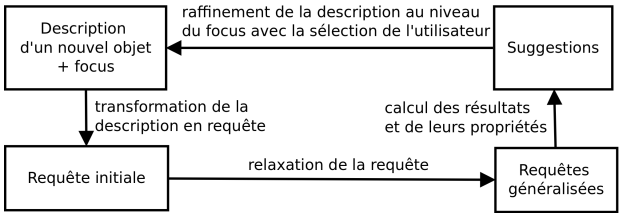

FIGURE 3 – Raffinement interactif de la description d'un nouvel objet.

Turle (<*CaseK*> *a case ;* :*personnage* <*Missbean*>). SPARQL, pour sa part, est un langage de requête pour RDF fondé sur le *graph pattern matching* (Prud'hommeaux & Seaborne, 2008). Une question et sa traduction en SPARQL sont montrées dans la figure 2.

Systèmes d'Information Logiques (LIS). Les LIS (Ferré & Ridoux, 2004) forment un paradigme de recherche d'information et d'exploration, combinant l'interrogation et la navigation. Ils se rapprochent du paradigme de la recherche par facettes (Sacco & Tzitzikas, 2009). Leur langage de requête, LISQL, a une expressivité proche de celle de SPARQL et une syntaxe similaire à Turtle (Ferré & Hermann, 2011). Un prototype, Sewelis [1], a été implémenté. L'utilisateur navigue de requête en requête. Les liens de navigation, calculés automatiquement à partir des données, sont des transformations de requête en s'assurant qu'après transformation, la nouvelle requête ait des résultats et que chaque requête avec des résultats soit atteignable.

3 UTILIS : Guidage interactif pour la création d'objets

Cette section décrit UTILIS, la méthode de guidage interactif pour aider les utilisateurs à créer des objets dans un graphe RDFS. UTILIS cherche les objets similaires à la description du nouvel objet, c'est-à-dire ayant des propriétés et valeurs en commun. La figure 3 illustre le processus de raffinement interactif de la description d'un nouvel objet. L'élément initial est composé de la description courante d'un nouvel objet et d'un focus qui

1. http://www.irisa.fr/LIS/softwares/sewelis

Description	<CaseK> a :case ; :collection <Ma vie à deux> ; :personnage []
Requête initiale	SELECT ?z WHERE { ?x a :case . ?x :collection ?y . ?x :personnage ?z . FILTER (?y = <Ma vie à deux>)}

FIGURE 4 – Une description et sa transformation en requête initiale.

correspond à l'élément de la description que l'on souhaite compléter. Cette description est transformée en requête (section 3.1). Cette requête est utilisée comme point de départ pour obtenir des requêtes généralisées à l'aide de règles de relaxation (section 3.2). Les requêtes initiale et généralisées permettent de calculer des objets similaires. L'intégralité du graphe RDFS est utilisée pour trouver ces objets. Ces objets et leurs propriétés servent de suggestions pour compléter la nouvelle description (section 3.3). Après raffinement, la nouvelle description sera utilisée pour définir une nouvelle requête et ainsi de suite. Un algorithme efficace pour calculer les suggestions à partir du graphe RDFS et de la description du nouvel objet a été implémenté selon les principes de la programmation dynamique (section 3.4).

3.1 Transformation de la description en requête initiale

La description est composée des éléments qui ont été renseignés par l'utilisateur à propos du nouvel objet. La figure 4 montre une description et la requête initiale obtenue après transformation. La description signifie que la case K est une case de la collection *Ma vie à deux*, qui a un ou des personnages, lesquels n'ont pas encore été spécifiés. Le focus est représenté par la partie soulignée. La description est transformée en requête en 3 étapes. D'abord, l'identité du nouvel objet est remplacée par une variable. Par exemple, <CaseK> est remplacé par $?x$. Ensuite, la description est transformée en *graph pattern* SPARQL. Cela permet de n'avoir qu'un élément modifiable par triplet. Chaque triplet n'est composé que d'un seul individu. Les autres éléments du triplet sont des variables ou la propriété pré-définie *rdf :type*. Par exemple, (*:collection <Ma vie à deux>*) est transformé en (*?x :collection ?y FILTER (?y = <Ma vie à deux>)*). Enfin, la description est transformée en requête en définissant la variable au focus comme variable à chercher et le corps de la requête correspond au *graph pattern* de l'étape 2. Cette requête initiale permet d'obtenir les objets qui ont toutes les propriétés et valeurs déjà connues du nouvel objet.

Règle	Triplet ou filtre initial	Triplet relâché	Condition
SuperProperty	$?x\ p_1\ ?y$	$?x\ p_2\ ?y$	p_1 `subp` p_2
SuperClass	$?x$ `a` c_1	$?x$ `a` c_2	c_1 `subc` c_2
Resource	$?x = r_2$	`nil`	
Property	$?x\ p_1\ ?y$	`nil`	$\nexists p \neq p_1.(p_1$ `subp` $p)$
Class	$?x$ `type` c_1	`nil`	$\nexists c \neq c_1.(c_1$ `subc` $c)$

FIGURE 5 – **Règles de relaxation**. Les $?x$ et $?y$ sont des variables, les r_i sont des ressources (URIs ou littéraux), les p_j des propriétés, et les c_k des classes ; `a`, `subc`, `subp` sont des abréviations pour `rdf:type`, `rdfs:subClassOf` et `rdfs:subPropertyOf`.

3.2 Relaxation de la requête initiale

Après avoir renseigné deux ou trois couples propriété-valeur, la description d'un nouvel objet devient unique dans la base, en effet les objets de la base n'ont pas tous les mêmes couples propriété-valeur. La requête initiale, obtenue à partir de la description du nouvel objet, est alors sans résultats. Pour continuer à proposer des suggestions à l'utilisateur, UTILIS cherche les objets similaires au nouvel objet en généralisant cette requête. Pour généraliser la requête initiale, nous avons défini des règles de relaxation, inspirées des règles d'approximation de requêtes d'Hurtado *et al.* (2008). La figure 5 montre l'ensemble des règles de relaxation, applicables aux triplets. La première colonne montre le nom de la règle, la deuxième le triplet avant relâchement, la troisième le triplet relâché et la quatrième les éventuelles conditions pour l'application de la règle. À part les règles *SuperProperty* et *SuperClass*, les règles n'ont pas besoin d'ontologies pour la relaxation. La règle *SuperProperty* s'applique à un triplet ayant une variable comme sujet et objet et une resource p_1 comme prédicat. La condition d'application de cette règle est que p_1 soit une sous-propriété d'une autre propriété p_2. Après l'application de cette règle, le sujet et l'objet restent identiques mais la propriété p_1 est remplacée par la propriété p_2. Par exemple, (*?x,* **:** *sAdresseA, ?y*) peut être relâché en (*?x,* **:** *converseAvec, ?y*) par l'application de cette règle s'il existe le triplet (**:** *sAdresseA,rdfs* **:** *subPropertyOf,* **:** *converseAvec*). Le triplet relâché *nil* correspond à la supression du triplet initial. La distance entre la requête initiale et la requête généralisée est le nombre de règles appliquées pour passer de l'une à l'autre. À distance 0, ce sont les résultats de la re-

quête initiale, sans généralisation. Par exemple, supposons que la requête initiale, c'est-à-dire celle obtenue directement à partir de la description, est (*SELECT ?x WHERE { ?x a :case . ?x :collection ?y . ?x :personnage ?z . ?x :personnage ?a . FILTER (?y = <Ma vie à deux> && ?z = <Miss-Bean> && a = Fatbean)}*) [2]. Une requête généralisée peut être la même sans le filtre *(?z = <Missbean>)* en appliquant la règle *Resource* sur ce filtre. Cette requête généralisée a une distance de 1. À la même distance, la règle *Resource* peut aussi être utilisée sur d'autres filtres, par exemple sur le filtre *(?a = <Fatbean>)*. L'union de tous les résultats des requêtes généralisées avec une seule étape de relaxation sont les résultats proposés à distance 1. Les règles peuvent être combinées. L'ordre d'application des règles n'a pas d'influence sur les requêtes généralisées. Chaque requête généralisée a une distance unique. Cette propriété de confluence permet d'avoir un algorithme efficace en utilisant la programmation dynamique (section 3.4).

3.3 Raffinement de la description du nouvel objet

Les objets similaires sont les résultats des requêtes généralisées ou non. Ces objets et leur propriétés servent de suggestions pour compléter la nouvelle description. Les suggestions sont des ressources, des classes ou des propriétés. Si le focus est sur une ressource, les suggestions sont des classes et des propriétés. Par exemple, supposons que la description d'un nouvel objet est (*<CaseK> a :case*). Les suggestions faites à l'utilisateur sont des propriétés (ex. *:personnage []*, *:collection []*) et des classes (ex. *a :case muette*). La propriété suggérée *:collection []* peut être sélectionnée pour spécifier que la nouvelle case a une collection, comme toutes les autres cases connues. La nouvelle description est (*<CaseK> a :case ; :collection []*) et le nouveau focus est sur l'objet de cette propriété. Si le focus est sur un *[]*, les suggestions peuvent de plus être des ressources, ici les collections. En sélectionnant ici la collection *Ma vie à deux*, la nouvelle description est (*<CaseK> a :case ; :collection <Ma vie à deux>*) et le nouveau focus est remonté au niveau de l'objet en cours de création.

Pour présenter les suggestions à l'utilisateur et lui permettre de compléter la description du nouvel objet, les mécanismes d'interaction de Sewelis ont été réutilisés pour la création. Les suggestions sont utilisées comme liens de navigation. Par défaut, seules les suggestions de la plus petite distance sont proposées, la liste peut être élargie aux distances supérieures à

2. Quelles sont les cases de *Ma vie à deux* avec les personnages Missbean et Fatbean ?

la demande de l'utilisateur, jusqu'à ce que tous les objets compatibles avec l'élément à chercher soient proposés. À tout moment, l'utilisateur peut saisir manuellement ce qu'il souhaite ajouter. Cette saisie est nécessaire pour les nouvelles valeurs. Des mécanismes d'auto-complétion aident à trouver les valeurs existantes. Une fois la suggestion sélectionnée par l'utilisateur, la nouvelle description sera utilisée pour définir la requête initiale et celle-ci sera généralisée pour permettre de faire de nouvelles suggestions à l'utilisateur.

3.4 Algorithme de relaxation

Un algorithme naïf pour le calcul des résultats à une distance d d'une requête q consiste à générer toutes les requêtes généralisées à une distance d, et à calculer l'union de leurs résultats. Ces requêtes généralisées sont obtenues en appliquant d relaxations sur les n triplets de la requête, ce qui fait au moins C_n^d requêtes généralisées. En raison de la nature des requêtes dont les triplets forment des graphes sans cycle, les résultats de chaque requête généralisée peuvent être calculés en $O(n)$ opérations ensemblistes de type intersection et traversée de relations binaires. Le coût de l'algorithme naïf est trop important, surtout que les requêtes généralisées sont seulement des intermédiaires dans notre approche.

Nous utilisons dans UTILIS un algorithme plus efficace fondé sur la technique de la programmation dynamique qui calcule directement les résultats $E(d, q)$ à une distance d d'une requête q, où la requête est représentée en notation Turtle. La fonction E est définie récursivement en s'appelant elle-même avec de plus petites distances et/ou des sous-requêtes. Les cas de bases sont quand $d = 0$ et q est une requête atomique (c'est-à-dire une ressource, une classe, ou `[]`). L'algorithme est défini par un ensemble d'équations, couvrant toutes les combinaisons d'une distance et d'une requête. Par exemple, l'équation qui définit E pour les conjonctions de requêtes est

$$E(d, q_1; q_2) = \bigcup_{i=0}^{d} (E(i, q_1) \cap E(d - i, q_2)).$$

Cette équation dit qu'un résultat d'une requête $q_1; q_2$ est un objet qui est à la fois un résultat à une distance d_1 de q_1 et un résultat à une distance d_2 de q_2, tel que $d_1 + d_2 = d$, où la distance d est distribuée entre les deux sous-requêtes ($d_1 = i$ et $d_2 = d - i$) de toutes les manières possibles (pour

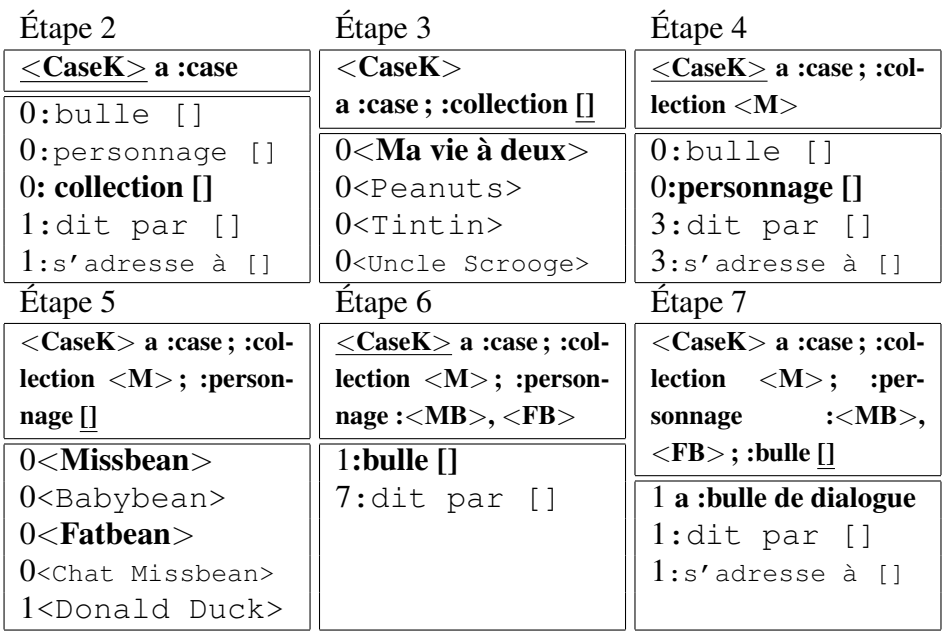

FIGURE 6 – **Étapes de création de la nouvelle case CaseK** : Elle fait partie de la collection *Ma vie à deux*. Elle a pour personnage Missbean et Fatbean et une bulle de dialogue dit par Missbean à Fatbean.

$i = 0..d$). Les équations pour les requêtes $p\ \ q_1$ et `is` p `of` q_1 sont similaires, en utilisant la traversée d'une relation au lieu d'une intersection. Les résultats à distance d pour les classes (propriétés) sont fondés sur les sur-classes (sur-propriétés) à distance d dans la hiérachie des classes (propriétés).

Les résultats intermédiaires de $E(d, q)$ sont stockés dans une table avec une ligne pour chaque distance de 0 à d, et une colonne pour q et chacune des sous-requête de q (soit $O(n)$ colonnes). La complexité de calcul d'une cellule de ce tableau est de $O(d)$ opérations ensemblistes. Par conséquent, la complexité de notre algorithme est de $O(nd^2)$ opérations ensemblistes, c'est-à-dire polynomial au lieu de combinatoire pour l'algorithme naïf.

4 Exemple

Pour illustrer UTILIS, cette section détaille des étapes de la création de la description d'une nouvelle case dans la base d'indexation. La base contient les cases montrées dans la figure 1 et six autres cases. Imaginons

FIGURE 7 – Case de bande dessinée extraite de *Ma vie à deux : Pour le meilleur et pour le pire !*, écrite et dessinée par *Missbean, City Editions*.

Étape 3

Étape 7

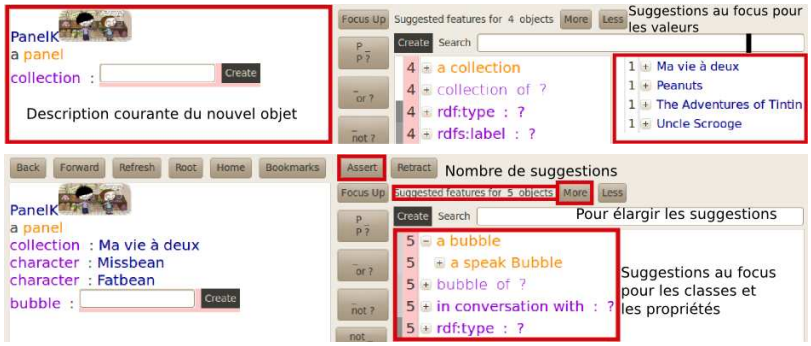

FIGURE 8 – Captures d'écran d'UTILIS des étapes 3 et 7 de l'indexation de la case de la figure 7.

qu'un utilisateur souhaite ajouter la case de la figure 7, nommée CaseK. Elle fait partie de la collection *Ma vie à deux*. Elle a deux personnages, Missbean et Fatbean et une bulle de dialogue dit par Missbean à Fatbean.

La figure 6 présente les étapes 2 à 7 de la création. À chaque étape, la première boîte contient la description courante de l'objet, le focus est souligné, la seconde boîte contient les suggestions avec, devant chacune d'elles, la distance minimale entre la requête initiale et la requête généralisée qui a permis cette suggestion. Pour des raisons de place, nous ne montrons à chaque étape qu'un nombre limité de suggestions. L'élément en gras correspond au choix de l'utilisateur.

À l'étape 2, la description courante est *<CaseK> a :case*. Les suggestions s'adaptent à cette description : ce sont les propriétés d'au moins une case. L'utilisateur choisit *:collection []*. La partie supérieure de la figure 8 montre l'interface utilisateur pour l'étape 3, les annotations en rouge ayant

été faites à la main, la description courante est dans la partie gauche. Dans la partie droite, les ressources suggérées pour le focus sont listées, ici les collections. Au-dessus de cette partie, le nombre de suggestions est indiqué et un bouton *More* permet de les élargir aux distances supérieures contenant des résultats. À l'étape 3, l'utilisateur choisit *<Ma vie à deux>* parmi toutes les collections. Retournons à l'étape 4 de la figure 6, la description est *<CaseK> a :case ; :collection <Ma vie à deux>*. Les suggestions sont les propriétés d'au moins une case de *Ma vie à deux*. La propriété choisie par l'utilisateur est *:personnage []* pour spécifier les personnages de la nouvelle case. À l'étape 5, les suggestions sont tous les personnages des cases déjà indexées de *Ma vie à deux*. L'utilisateur choisit **<*Missbean*>** et **<*Fatbean*>**. À l'étape 6, la requête initiale, correspondant à la description, n'a pas de résultats. En effet, ces 2 personnages n'apparaissent ensemble dans aucune case de la base. Par relaxation, UTILIS fait des suggestions à l'utilisateur de distances 1 et 7. La partie inférieure de la figure 8 montre l'étape 7, les classes et propriétés suggérées sont dans la partie du milieu. À partir de là, l'utilisateur peut continuer la description de la case et quand il décidera qu'elle est complète, il l'ajoutera à la base avec le bouton *Assert*.

5 Expérience utilisateur

Afin de savoir si les suggestions proposées par UTILIS lors de la création de nouveaux objets peuvent être utiles aux utilisateurs, une expérimentation a été réalisée. Des utilisateurs ont testé et évalué l'utilisabilité d'UTILIS et de Protégé [3]. L'interface et le guidage de Protégé sont représentatifs des éditeurs actuels. En effet, deux tiers des utilisateurs du Web sémantique l'utilisent comme éditeur (Cardoso, 2007). Nous avons rentré le domaine et le co-domaine de chaque propriété. Quand un utilisateur choisit la classe d'un individu, un formulaire avec les propriétés, ayant cette classe comme domaine, est créé. Pour les valeurs, les suggestions sont les individus du co-domaine de chaque propriété.

5.1 Méthodologie

Les sujets étaient composés de 18 étudiants en master d'informatique. Ils avaient une connaissance préalable en bases de données relationnelles, mais ne connaissaient ni UTILIS, ni Protégé, ni le Web sémantique. Pour chaque éditeur, ils devaient effectuer les mêmes tâches. La procédure de

3. `http://protege.stanford.edu`

l'expérience pour chaque sujet a été la suivante : 1) lire une note d'introduction sur l'expérience globale ; 2) apprendre à utiliser un éditeur à l'aide d'un tutoriel de 30 minutes avec un exemple de création d'indexation ; 3) formuler et créer des indexations avec cet éditeur pendant 30 minutes ; 4) remplir un questionnaire sur cet éditeur ; 5) procéder en répétant les étapes 2 à 4 avec l'autre éditeur, et 6) remplir un questionnaire comparatif.

Les sujets devaient mettre à jour une base déjà existante, décrivant des cases de bandes dessinées. La base était identique à chaque début de session de test. Elle était composée de 362 individus, divisés en 16 classes et reliés par 20 propriétés, incluant 89 cases. Les sujets devaient mettre à jour la base avec deux jeux de cases. Ils ont été divisés en 4 groupes, chaque groupe a effectué 2 sessions de 1h30. Chaque groupe a testé un premier éditeur avec un des deux jeux puis l'autre éditeur avec le second jeu. Chaque jeu était composé de 11 cases, la case utilisée pour le tutoriel et 10 cases à créer, avec au moins 2 cases de la même collection. Les cases étaient présentées sur papier dans le même ordre pour tous les sujets. Elles existaient déjà dans la base, mais n'avaient aucune classe ni propriété. Les sujets avaient pour instruction d'entrer un maximum d'informations sur chaque case, prenant la description des cases existantes comme modèle. Ils avaient pour consigne de réutiliser autant que possible l'information existante. Ils pouvaient néanmoins créer des nouveaux éléments s'ils en avaient besoin.

Pendant les expériences, un fichier de log des actions utilisateur a été créé, enregistrant en particulier le nombre de descriptions créées sous UTILIS et Protégé, le mode de sélection des éléments des descriptions et le nombre d'élargissements des suggestions. En effet, comme déjà mentionné, les utilisateurs peuvent élargir les suggestions aux distances supérieures si l'élément recherché n'est pas présent dans les premières.

5.2 Résultats

Cette section discute des résultats de l'expérience. Il n'existe pas de différence significative entre le nombre de descriptions créées par les deux groupes initiaux sous Protégé et UTILIS, ni dans le nombre moyen d'erreur. L'ordre d'utilisation des outils n'a pas eu d'influence. Les erreurs sous Protégé pouvaient être un oubli ou une erreur dans un élément de la case et sous UTILIS également un mauvais placement du focus. Les sujets ont trouvé les suggestions pertinentes et souhaitent les avoir dans un éditeur de données du Web sémantique.

Des suggestions pertinentes. Les suggestions ont été utilisées par les

	Propriétés et classes	Collections	Personnages	Émetteurs	Destinataires	Lieux	Autres éléments
Sélection dans les suggestions (%)	91	91	90	89	82	58	21
Saisie avec auto-complétion (%)	9	9	8	11	15	25	21
Création (%)	0	0	2	0	3	17	58
Sélection : 3 premiers ensembles (%)	91	91	92	89	84	68	44
Plage d'élargissements	0-4	0-1	0-2	0-1	0-3	0-4	0-6

TABLE 1 – Pourcentages d'éléments sélectionnés dans les suggestions, par saisie et par création, la plage d'élargissements des suggestions et le pourcentage sélectionnés dans les 3 premiers ensembles de suggestions.

utilisateurs pendant la création des nouvelles cases. Le tableau 1 montre la proportion d'éléments de la description choisis par sélection d'une suggestion, par saisie et par création, ainsi que la plage d'élargissements des suggestions et le pourcentage sélectionné dans les trois premiers ensembles de suggestions. On peut voir, par exemple, que l'ajout des personnages a été fait, dans 90% des cas par la sélection d'une suggestion, 8% par saisie et 2% par création. Les éléments relatif à la collection, aux personnages et aux interlocuteurs des bulles, ont été, s'ils existaient déjà, sélectionnés à plus de 80% dans les 3 premiers ensembles de suggestions. Pour les lieux et les autres éléments, les suggestions ont été moins utiles. Ces éléments de la description sont plus subjectifs que les précédents et ont beaucoup été créés par les utilisateurs. Les questionnaires remplis après les expériences donnent une image cohérente. En effet, 16 sujets ont trouvé les suggestions pertinentes pour les personnages, 15 pour la collection, 12 pour les interlocuteurs, mais seulement 6 pour les lieux et 3 pour les autres éléments.

Des suggestions appréciées. Les sujets ont été interrogés sur les éléments de chaque éditeur qu'ils aimeraient avoir dans un éditeur. Dix sujets souhaitent avoir les propriétés dans un formulaire comme dans Protégé. Cependant, 13 ont déclaré avoir été gênés par le parcours des listes entières d'objets sous Protégé. Dans UTILIS, 14 souhaitent conserver les suggestions adaptées à l'objet en cours de création, et 11 le mode de recherche par auto-complétion. Neuf (resp. sept) sujets souhaitent avoir les suggestions d'UTILIS pour certaines (resp. toutes les) propriétés.

6 Comparaison avec l'état de l'art

La connaissance utilisée pour assister les utilisateurs à la création de nouveaux objets peut être de différents types et change selon l'éditeur.

Protégé utilise le vocabulaire et les axiomes de l'ontologie. À l'étape 2 de notre scénario, avec la base utilisée pour l'expérience, si le domaine et co-domaine des propriétés sont renseignés, 6 propriétés seraient proposées, 20 sinon. À l'étape 5, les 51 personnages de la base seraient proposés et 362 objets sinon. OKM (Davies *et al.*, 2010) utilise les objets existants pour créer automatiquement des formulaires. À l'étape 2, les propriétés des objets de la même classe que le nouvel objet seraient proposées. À l'étape 5, l'utilisateur devrait commencer sa saisie pour que des objets lui soient proposés, et les premiers seraient ceux qui sont déjà objet de la propriété *:personnage*. Semantic MediaWiki (Völkel *et al.*, 2006) est une extension de MediaWiki, utilisée par Wikipédia. Son extension Semantic Forms [4] permet de créer des formulaires, pour créer des pages de wikis avec des liens sémantiques. À l'étape 2, les propriétés sélectionnées par le créateur du formulaire seraient proposées. À l'étape 5, le système proposerait des objets ayant déjà cette propriété, selon la saisie de l'utilisateur. Le wiki sémantique KiWI (Schaffert *et al.*, 2009), anciennement IkeWiki, nécessite une ontologie pré-existante. À l'étape 2, toutes les propriétés seraient proposées. À l'étape 5, il faudrait commencer à saisir pour avoir une suggestion. Avec UTILIS, aucun travail préliminaire, ni aucune ontologie pré-existante ne sont nécessaires. UTILIS n'a pas besoin non plus de connaître les domaine et co-domaine des propriétés et la liste des objets s'adapte au nouvel objet.

Les éditeurs Gino (Bernstein & Kaufmann, 2006) et QuiKey (Haller, 2010) et le wiki sémantique ACEWiki (Kuhn, 2009) utilisent un guidage s'appuyant sur la syntaxe et le vocabulaire, pour créer des triplets RDF(S). Lorsqu'un utilisateur commence sa saisie, Gino suggère, par autocomplétion, les ressources, les classes, les propriétés et les mots grammaticaux syntaxiquement correct. À l'étape 2, l'utilisateur devrait saisir des caractères pour que Gino lui propose des propriétés. À l'étape 5, Gino proposerait des objets. Avec Quikey, les suggestions proposées diminuent en fonction de la saisie de l'utilisateur. À l'étape 2, toutes les propriétés seraient proposées. À l'étape 5, le même mécanisme s'appliquerait avec les objets. ACEWiki propose des listes contenant les mots-clés et les ressources du graphe, autorisés pour compléter le triplet. À l'étape 2, la liste

4. `http://www.mediawiki.org/wiki/Extension:Semantic_Forms`

des propriétés serait proposée. À l'étape 5, la liste des objets remplacerait celle des propriétés. Avec UTILIS, le filtrage est effectué sans saisie.

7 Conclusion

Nous proposons un système, UTILIS, qui guide les utilisateurs lors de la création d'objets dans un graphe RDFS. Le guidage s'appuie sur les objets existants, et sur la description courante du nouvel objet. À chaque étape, la description courante est utilisée pour trouver des objets similaires, dont les propriétés sont ensuite utilisées comme suggestions pour compléter cette description. Les objets similaires sont les résultats de requêtes qui sont généralisées à partir de la description courante. Ces requêtes sont obtenues par application des règles de relaxation. Un algorithme efficace pour le calcul des suggestions a été conçu et implémenté dans Sewelis.

Comparé à d'autres éditeurs RDFS, les suggestions d'UTILIS sont fondées sur les objets existants, plutôt qu'uniquement sur le schéma RDFS. Elles sont ainsi adaptées à chaque objet en cours de création. Un avantage est qu'UTILIS n'a pas besoin qu'une ontologie soit définie, même s'il peut en utiliser une pour améliorer ses suggestions lorsqu'elle est disponible. L'expérience a montré que les sujets ont trouvé les suggestions utiles et les ont réellement utilisées. En effet, dans la plupart des cas, ils pouvaient trouver l'élément recherché dans les trois premiers ensembles de suggestions. De plus 14 sur 18 souhaitent les conserver dans un éditeur.

Remerciements : Nous tenons à remercier Marie Levesque (alias Missbean) pour l'utilisation de la case extraite de *Ma vie à deux : Pour le meilleur et pour le pire !* et son éditeur *City Editions*. Nous remercions également les 18 étudiants de l'école d'ingénieurs INSA qui ont participé à l'expérimentation.

Références

BECKETT D., BERNERS-LEE T. & PRUD'HOMMEAUX E. (2010). Turtle - Terse RDF Triple Language. W3C Recommendation.

BERNSTEIN A. & KAUFMANN E. (2006). Gino - a guided input natural language ontology editor. In *Proceedings of the International Semantic Web Conference (ISWC)* : Springer.

CARDOSO J. (2007). The semantic web vision : Where are we ? *IEEE Intelligent Systems*, p. 84–88.

DAVIES S., DONAHER C. & HATFIELD J. (2010). Making the Semantic Web usable : interface principles to empower the layperson. *Journal of Digital Information*, **12**(1).

FERRÉ S. & HERMANN A. (2011). Semantic search : Reconciling expressive querying and exploratory search. In *Proceedings of the International Semantic Web Conference (ISWC)* : Springer.

FERRÉ S. & RIDOUX O. (2004). An introduction to logical information systems. *Information Processing & Management*, **40**(3), 383–419.

HALLER H. (2010). QuiKey – an efficient semantic command line. In *Knowledge Engineering and Management by the Masses (EKAW)*, p. 473–482 : Springer.

HITZLER P., KRÖTZSCH M. & RUDOLPH S. (2009). *Foundations of Semantic Web Technologies*. CRC Press.

HURTADO C. A., POULOVASSILIS A. & WOOD P. T. (2008). Query relaxation in RDF. *J. Data Semantics*, **10**, 31–61.

KUHN T. (2009). How controlled english can improve semantic wikis. In *Semantic Wiki Workshop (SemWiki)*, volume 464 : CEUR-WS.org.

LIMPENS F., GANDON F. & BUFFA M. (2009). Sémantique des folksonomies : structuration collaborative et assistée. In *Ingénierie des Connaissances (IC)*, p. 37–48 : Presses Universitaires de Grenoble.

PASSANT A. & LAUBLET P. (2008). Meaning of a tag : A collaborative approach to bridge the gap between tagging and linked data. In *Workshop Linked Data on the Web (LDOW)* : CEUR-WS.

PRUD'HOMMEAUX E. & SEABORNE A. (2008). SPARQL query language for RDF. W3C Recommendation.

G. SACCO & Y. TZITZIKAS, Eds. (2009). *Dynamic Taxonomies and Faceted Search : Theory, Practice, and Experience*, volume 25 of *The Information Retrieval Series*. Berlin : Springer.

SCHAFFERT S., EDER J., GRÜNWALD S., KURZ T., RADULESCU M., SINT R. & STROKA S. (2009). Kiwi - a platform for semantic social software. In *Semantic Wiki Workshop (SemWiki)* : CEUR-WS.org.

VÖLKEL M., KRÖTZSCH M., VRANDECIC D., HALLER H. & STUDER R. (2006). Semantic Wikipedia. In *International conference on World Wide Web (WWW)*, p. 585–594 : ACM Press.