HAL
open science

Évolution de l'ontologie utilisée comme référentiel sémantique dans un système de téléapprentissage

Delia Rogozan, Gilbert Paquette, Ioan Rosca

To cite this version:

Delia Rogozan, Gilbert Paquette, Ioan Rosca. Évolution de l'ontologie utilisée comme référentiel sémantique dans un système de téléapprentissage. Technologies de l'Information et de la Connaissance dans l'Enseignement Supérieur et l'Industrie, Oct 2004, Compiègne, France. pp.243-249. edutice-00000723

HAL Id: edutice-00000723

https://edutice.hal.science/edutice-00000723v1

Submitted on 16 Nov 2004

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Évolution de l'ontologie utilisée comme référentiel sémantique dans un système de téléapprentissage

Delia ROGOZAN*, **, Gilbert PAQUETTE*, Ioan ROSCA*

* Centre de recherche LICEF (Laboratoire d'Informatique Cognitive et d'Environnements de Formation)
4750 Avenue Henri-Julien, Montréal (Qc.) H2T 3E4, Canada
http://www.licef.teluq.uquebec.ca/eng/index.htm
** Laboratoire PSI (Perception Système Information) - INSA de Rouen
Place Emile Blondel - B.P. 08, 76131 Mont-Saint-Aignan CEDEX, France
http://psiserver.insa-rouen.fr/psi/
{drogozan, gpaquett, irosca @licef.teluq.uquebec.ca}

Résumé

L'utilisation d'une ontologie comme référentiel sémantique dans un système de téléapprentissage demande l'évolution de cette ontologie. Dans cet article nous proposons premièrement une méthodologie pour gérer cette évolution d'une manière consistante. L'évolution de l'ontologie peut causer des incompatibilités qui affectent l'accès aux éléments référencés par les entités ontologiques. Afin de résoudre ce problème, nous proposons une analyse des effets des changements sur la compatibilité entre les versions ontologiques et nous utilisons les résultats de cette analyse pour définir un procédé permettant la préservation du référencement sémantique des éléments après l'évolution de l'ontologie.

Mots-clés : Représentation et gestion de connaissances, Méthodologie, Évolution de l'ontologie, Analyse de la compatibilité, Référencement sémantique évolutif

Abstract

The use of ontology as referencing system for an e-learning environment requires the evolution of this ontology. In this paper, we propose a generic methodology for managing the ontology evolution in an appropriate manner. The ontology evolution may cause incompatibilities, which in turn may affect the pedagogical elements that use ontology entities as references. In order to solve this problem, we propose a conceptual investigation of what are the dimensions of the compatibility analysis between ontology versions together with their results. According to these results, we propose a method to keep all dependant pedagogical elements in a consistent state after the ontology evolution.

Keywords: Formalizing and managing knowledge in organizations, Methodology, Ontology Evolution, Compatibility Analysis, Semantic and evolving classification.

Introduction

Nos travaux de recherche s'intègrent dans le projet LORNET[1] qui vise à élaborer une application permettant la spécification des *systèmes d'apprentissage et de gestion des connaissances (SAGC)* sur le Web sémantique. Ces systèmes doivent supporter les apprenants distants à atteindre un certain niveau de compétence. Pour faire cela, les apprenants accomplissent des activités pédagogiques et utilisent des ressources d'apprentissage. Les compétences des acteurs, les activités pédagogiques ainsi que les ressources d'apprentissage constituent les *éléments pédagogiques* qui composent un *SAGC*. Ces éléments pédagogiques sont référencés sémantiquement par des entités d'une même ontologie, ce qui permet la création d'un *SAGC* où la combinaison physique des éléments composants est accompagnée d'une agrégation de leur sens [1]. Les ontologies utilisées comme référentiels sémantiques dans les *SAGC* doivent évoluer. À mesure que l'apprentissage se concrétise, les connaissances des acteurs évoluent, ce qui peut demander la modification de l'ontologie si les acteurs considèrent qu'elle ne satisfait plus ses fonctions de référentiel sémantique. De plus, l'ontologie devrait permettre un feed-back évolutif au sens que le référencement des nouvelles ressources et activités pourrait demander la modification de leur référentiel sémantique afin de prendre en compte des nouvelles caractéristiques. D'un point de vue plus général, les ontologies du Web sémantique doivent évoluer aussi [2 ; 3 ; 4 ; 5] et des méthodes doivent être conçues pour supporter la gestion de leur évolution [6]. Dans cet article nous proposons des solutions pour supporter l'évolution de l'ontologie dans un environnement dynamique, multi-acteurs et distribué, comme tout *SAGC* du Web sémantique. Dans le deuxième chapitre, nous développons une méthodologie pour gérer l'évolution de l'ontologie d'une manière consistante. Cette évolution peut causer des incompatibilités qui doivent être analysées avant de mettre en opération l'ontologie évoluée. Ainsi, dans le troisième chapitre nous proposons une méthode d'analyse de la compatibilité

[1] LORNET est un projet du réseau de recherche pan-canadien sur les répertoires d'objets d'apprentissage http://www.lornet.org/.

entre les versions ontologiques et nous présentons, dans le quatrième chapitre, une technique pour assurer la mise en opération d'une ontologie évolutive utilisée comme référentiel sémantique.

Gestion de l'Évolution de l'Ontologie

Nous définissons *l'évolution de l'ontologie* comme le processus de changement de la version précédente de l'ontologie en une version nouvelle, afin de rendre compte des modifications dans son domaine, dans sa conceptualisation ou dans son usage. Ce processus se traduit en réalité par l'exécution d'un ou plusieurs changements ontologiques pour passer d'une version de l'ontologie (V_N) à une version nouvelle (V_{N+1}).

Règles d'Évolution de l'Ontologie

En nous basant sur plusieurs recherches [7 ; 8 ; 9 ; 5], nous avons identifié un ensemble des règles à respecter lors de l'évolution de l'ontologie (tableau 1).

Règle d'évolution	*Description de la règle d'évolution*
Évolution consistante (1)	L'ontologie doit évoluer d'un état consistant vers un autre état consistant, c'est-à-dire l'état où toutes les contraintes de l'ontologie sont satisfaites. Ces contraintes concernent le modèle et les axiomes de l'ontologie.
Préservation des rôles de l'ontologie (2)	Les rôles de l'ontologie doivent être préservés lors de l'évolution. Nous nommons *rôle de l'ontologie* le service assuré par l'ontologie, le but de son utilisation. Par exemple, dans notre contexte de recherche, le rôle de l'ontologie est celui de référentiel sémantique pour les éléments pédagogiques d'un *SAGC*.
Validation (3)	Les changements apportés à l'ontologie doivent être approuvés par les développeurs.

Tableau 1. Règles d'évolution de l'ontologie

Méthodologie d'Évolution de l'Ontologie

Pour supporter la gestion de l'évolution de l'ontologie nous proposons une méthodologie que nous avons conçue en nous basant sur les recherches de [10;8;11; 12 ;13]. Nous avons développé cette méthodologie comme un processus en neuf étapes (figure 1), accomplies par les développeurs à l'aide d'un atelier de support à l'évolution de l'ontologie. Nous définissons *le développeur de l'ontologie* comme tout acteur impliqué dans le processus d'évolution, en précisant qu'il est possible d'avoir un ou plusieurs développeurs qui réalisent certaines tâches lors de ce processus. Dans les sections suivantes, nous présentons brièvement chaque étape du processus d'évolution (figure 1).

Accéder et Télécharger l'Ontologie à Modifier

Les ontologies se trouvent généralement stockées dans des librairies distribuées sur le Web, ces librairies fournissant des mécanismes d'emmagasinage, permettant un accès constant et la possibilité de naviguer dans les ontologies [14]. Dans cette étape, les développeurs accèdent à l'ontologie à modifier et la téléchargent sur leur poste de travail. Nous notons cette ontologie avec $\mathbf{V_N}$ et nous la nommons ***version précédente de l'ontologie***.

Identifier les Changements

Dans cette étape, les développeurs identifient les changements à apporter à l'ontologie selon deux méthodes :

- *identification descendante des changements explicites*. Il s'agit des changements imposés par la modification du domaine de définition ou celui d'utilisation de l'ontologie.
- *identification ascendante des changements implicites*. Il s'agit des changements identifiés à partir de l'analyse de l'ontologie elle-même en utilisant un ensemble des heuristiques, comme, par exemple, une heuristique affirmant que si un concept possède un seul sous-concept, alors ce concept peut être fusionné avec son sous-concept (Maedche et *al*., 2003).

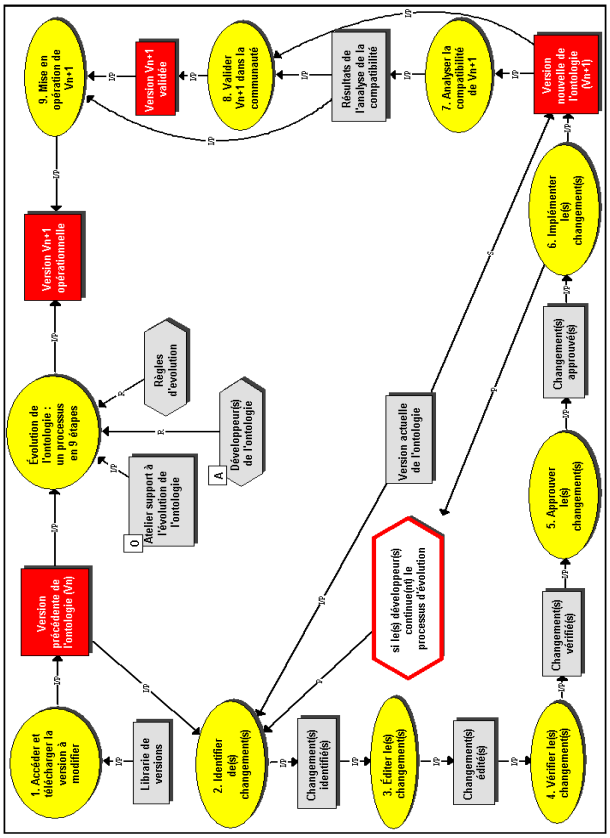

Figure 1. Modèle général de la méthodologie d'évolution de l'ontologie

Légende : Le formalisme de modélisation MOT (Paquette, 2002) s'interprète comme suit : l'ellipse représente une procédure, le rectangle un objet ou concept, l'hexagone une règle, l'hexagone ayant le signe A représente un acteur qui régit la procédure sous-jacente et l'hexagone avec le signe O un outil qui supporte cette procédure. Le lien C est un lien de Composition, I/P de Intrant/Produit, P de Précédence.

Éditer les Changements

Dans cette étape, les développeurs éditent les changements identifiés précédemment. Les changements peuvent être élémentaires ou complexes.

Un ***changement élémentaire*** est un changement primitif et non décomposable, les changements typiques étant l'ajout, l'effacement ou la modification des entités ontologiques.

Un ***changement complexe*** est composé de plusieurs changements élémentaires qui forment ensemble une seule entité logique, par exemple le déplacement, la fusion ou la séparation des entités ontologiques. Klein et Noy (2003) [7] mettent en évidence l'avantage d'utiliser des changements complexes : (1) ils sont plus facilement utilisables et compréhensibles, car leur intention est explicite, contrairement à une suite de changements élémentaires ayant le même résultat, et (2) ils permettent l'évolution de l'ontologie avec moins des pertes des données, par exemple le changement Déplacer_Concept préserve les instances, contrairement à la suite des changements Effacer et Ajouter_Concept.

Pour chaque changement, il est possible de générer différents changements additionnels conduisant à des états finaux différents [5]. Par exemple, si un concept à l'intérieur d'une hiérarchie des concepts est supprimé, ses sous-concepts peuvent être, soit supprimés, soit rattachés au concept-parent, soit rattachés au concept-racine de la hiérarchie.

Le résultat de l'étape d'édition consiste alors dans une séquence des changements à apporter à l'ontologie, chaque changement ayant la forme d'un couple de type (changement édité, scénario de changements additionnels). De plus, chaque changement doit être annoté en utilisant un ensemble des métadonnées décrivant l'auteur, le but du changement, ainsi que le rôle sémantique de l'entité ontologique étant le sujet du changement [3 ; 16].

Vérifier les Changements

Cette étape vise la vérification des effets des changements sur la consistance de l'ontologie. Si des changements invalident les contraintes du modèle ontologique, alors les développeurs doivent être informés et une méthode pour les supporter dans la résolution des inconsistances doit leur être fournie. Par exemple, la méthode proposée par Maedche et *al.*, [8], qui permet la résolution des inconsistances par l'introduction des changements additionnels[2].

En ce qui concerne les changements qui invalident les axiomes de l'ontologie, les développeurs peuvent modifier directement les axiomes invalidés afin de contourner les inconsistances.

Approuver les Changements

Considérons la situation où plusieurs acteurs distants tentent de modifier la même ontologie. Dans ce cas, les changements effectués dans une partie de l'ontologie se propagent vers les autres parties de l'ontologie, ce qui peut produire des situations conflictuelles, par exemple, si un acteur efface un concept utilisé par un autre acteur. Cette situation devient encore plus complexe si les acteurs sont autorisés à pouvoir accepter, refuser ou défier les changements des autres, étant donné que la consistance de l'ontologie doit être maintenue [17 ; 18]. Dans cette étape, les changements doivent alors être validés conjointement par les développeurs de l'ontologie. Si cette validation n'est pas possible dû au caractère distribué de l'environnement d'occurrence du processus d'évolution, alors des mécanismes de gestion des conflits doivent être prévus.

Implémenter les Changements

Cette étape vise l'implémentation des changements tout en assurant l'archivage des versions ontologiques et la préservation de la trace des changements et des métadonnées associées.

À la fin de cette étape, une ***nouvelle version de l'ontologie*** est obtenue et nous la notons avec $\mathbf{V_{N+1}}$. Si besoin, les développeurs peuvent retourner maintenant à l'étape 2 du processus, l'ontologie à modifier étant maintenant la nouvelle version obtenue à la fin de l'étape 6.

Analyser la Compatibilité entre V_N et V_{N+1}

Le processus d'évolution peut causer des incompatibilités pouvant provoquer l'altération des rôles de l'ontologie. Soit, par exemple, un objet pédagogique 'Cahier de Physique – 01' qui utilise le concept 'loi de Newton' comme référence sémantique. Après l'application d'un changement fusionnant les concepts 'loi de Newton' et 'loi de Pascal' dans un seul concept 'lois de la mécanique', cet objet n'est plus accessible au moyen de l'ontologie ainsi modifiée, par exemple, pour une recherche de type «trouver les objets pédagogiques décrivant les lois de la mécanique (concept dans l'ontologie évoluée)». Afin d'identifier ces incompatibilités, le but de cette étape est d'analyser les effets des changements sur la compatibilité de V_{N+1} du point de vue de la préservation des rôles de l'ontologie.

Valider V_{N+1} dans la Communauté

Dans cette étape, les développeurs approuvent ou désapprouvent collectivement la nouvelle version de l'ontologie avant de la rendre opérationnelle.

Opérationnaliser V_{N+1}

En fonction des résultats de l'analyse de la compatibilité, le problème le plus difficile est celui de mettre en opération la nouvelle version tout en préservant les rôles de l'ontologie. La préservation des rôles de l'ontologie peut se faire selon deux types de situation :

- quand la modification des artefacts dépendants (i.e. objets référencés, ontologies et applications dépendantes) en fonction des changements exécutés pour passer de V_N à V_{N+1} peut se réaliser, alors uniquement la nouvelle version de l'ontologie peut être utilisée par la suite pour accéder ou interpréter ces artefacts. Nous empruntons à Stojanovic et *al.*, (2002) [5], le

[2] Sachant que chaque propriété doit avoir minimum un concept comme domaine et comme co-domaine, l'effacement d'un concept étant le seul domaine d'une propriété demande soit d'effacer la propriété elle-même, soit de définir un autre concept comme domaine de cette propriété.

terme « propagation des changements » pour désigner cette situation.

- quand la modification des artefacts dépendants par rapport à la nouvelle version de l'ontologie ne peut pas se réaliser, alors il faudrait définir des mappages entre V_N à V_{N+1} afin de permettre l'accès et l'interprétation des artefacts par les deux versions. Les mappages inter-ontologies [19], dans notre cas entre les versions de la même ontologie, sont des règles spécifiant des liens de correspondance sémantique (p.ex. équivalence, intersection ou incompatibilité) entre des entités ontologiques appartenant aux versions différentes.

Pour conclure ce chapitre, nous mettons en évidence les contributions de cette méthodologie. La première contribution consiste dans l'unification, dans un cadre cohérent, des approches méthodologiques qui existent actuellement pour supporter l'évolution de l'ontologie. Nous obtenons ainsi une méthodologie unifiée décrivant un processus complet d'évolution dans un contexte multi-acteurs, incluant la gestion de versions multiples. La deuxième contribution réside dans la proposition des éléments méthodologiques novateurs, principalement en ce qui concerne l'analyse des effets des changements sur la compatibilité de la nouvelle version de l'ontologie ainsi que la mise en opération adéquate de cette nouvelle version. Dans le chapitre suivant, nous abordons alors plus en détail les deux éléments en question.

Analyse de la Compatibilité entre Versions Ontologiques

Dans ce chapitre nous présentons une étude théorique qui nous a permis de dégager les dimensions essentielles de l'analyse de la compatibilité entre versions ontologiques ainsi qu'une classification des changements effectués pour passer de V_N à V_{N+1} en fonction de leurs effets sur la préservation des rôles de l'ontologie.

Dimensions de l'Analyse de Compatibilité

Heflin [20] fournit la preuve que les changements qui ajoutent des entités à l'ontologie produisent une version rétrocompatible, tant que les changements qui effacent des entités produisent une version incompatible. Cette conclusion est exacte, mais elle ne prend pas en compte le fait que « *defining the compatibility between different ontology versions becomes a salient issues since there are several dimensions to compatibility* » [4]. Prenons un exemple. Supposons V_N où les concepts *'Professeur'* et *'Tuteur'* sont déclarés disjoints à l'aide d'un axiome de classe. Dans ce cas, un moteur d'inférence peut conclure que toute instance du concept *'Professeur'* est différente de toute instance du concept *'Tuteur'*. Supposons maintenant V_{N+1}, où cet axiome a été effacé. Dans ce cas, le moteur d'inférence utilisant V_{N+1} ne peut plus arriver à la conclusion ci-dessus, ce qui provoque une altération de son comportement. Par contre, si l'ontologie n'est pas utilisée par un moteur d'inférence, mais elle est utilisée uniquement pour servir comme référentiel des connaissances (concepts), alors l'effacement de l'axiome en question ne provoque aucune altération de l'accès aux objets référencés au moyen des concepts de l'ontologie. En conséquence, en fonction des rôles de l'ontologie, nous avons besoin de considérer plusieurs dimensions de l'analyse de la compatibilité. Nous proposons alors une analyse selon trois dimensions : la préservation des instances, la préservation de la structure conceptuelle et la préservation de l'axiomatisation de l'ontologie.

Préservation des Instances

La préservation des instances signifie qu'aucune instance n'est perdue lors du processus d'évolution de l'ontologie. Nous présentons dans le tableau 2 les types de résultat pour la préservation des instances.

Types de résultat/préservation des instances	*Exemple de changements*
Préservation des instances sans ajout $\boldsymbol{P_{IS}}$ - l'ensemble des instances de V_N est le même que celui de V_{N+1}	- Fusionner ou Séparer Concept - Déplacer ou Effacer Propriété
Préservation des instances avec ajout $\boldsymbol{P_{IA}}$ - l'ensemble des instances de V_N est élargi en V_{N+1}	- Ajouter_Instance - Ajouter_Instance de propriété
Non-préservation des instances $\boldsymbol{P_{IN}}$ - l'ensemble des instances de V_N est réduit ou modifié en V_{N+1}	- Effacer_Concept (si le concept a des instances) - Effacer_Instance

Tableau 2. Préservation des instances

Préservation de la Structure Conceptuelle

La préservation de la structure conceptuelle signifie qu'aucun des concepts qui composent le vocabulaire de l'ontologie n'est perdu lors de l'évolution de l'ontologie. Nous présentons dans le tableau 3 les types de résultats possibles pour cette dimension de l'analyse.

Types de résultat/préservation de la structure conceptuelle	*Exemple de changements*
Préservation de la structure conceptuelle sans ajout $\boldsymbol{P_{CS}}$ - l'ensemble des concepts de V_N est le même que celui de V_{N+1}	- Déplacer ou Effacer_Instance - Déplacer ou Effacer_Propriété
Préservation de la structure conceptuelle avec ajout $\boldsymbol{P_{CA}}$ - l'ensemble de concepts de V_N est élargi en V_{N+1}	- Ajouter_Concept - Reclassifier une instance comme un concept
Non-préservation de la structure conceptuelle $\boldsymbol{P_{CN}}$ - l'ensemble des concepts de V_N est réduit ou modifié en V_{N+1}	- Fusionner_Concepts - Séparer_Concept

Tableau 3. Préservation de la structure conceptuelle

Préservation de l'Axiomatisation
La préservation de l'axiomatisation de l'ontologie signifie qu'aucun des axiomes et des propriétés qui forment le système logique de l'ontologie n'est perdu lors du processus d'évolution. Nous présentons dans le tableau 4 les types de résultats possibles pour cette dimension.

Type de résultat/préservation de l'axiomatisation	*Exemple de changements*
Préservation de l'axiomatisation sans ajout $\boldsymbol{P_{AS}}$ - l'ensemble des faits inférés à partir de V_{N+1} est le même que l'ensemble des faits inférés à partir de V_N	- Modifier ou Déplacer_Concept - Déplacer ou Effacer_Instance
Préservation de l'axiomatisation avec ajout $\boldsymbol{P_{AA}}$ - l'ensemble des faits inférés à partir de V_{N+1} est un super ensemble de l'ensemble des faits inférés à partir de V_N	- Ajouter_Propriété, Axiome - Déplacer la propriété (axiome) d'un concept au concept-parent
Non-préservation de l'axiomatisation $\boldsymbol{P_{CN}}$ - l'ensemble des faits inférés à partir de V_{N+1} est réduit ou modifié par rapport à l'ensemble des faits inférés à partir de V_N	- Effacer_Propriété, Axiome - Déplacer la propriété (axiome) d'un concept à son sub-concept

Tableau 4. Préservation de l'axiomatisation

Classification des Changements Ontologiques

En prenant en compte les dimensions de l'analyse de la compatibilité, nous pouvons classifier les changements exécutés pour passer de V_N à V_{N+1} en changements totalement compatibles, rétrocompatibles ou incompatibles.

Changements totalement compatibles. Sont les changements qui préservent entièrement les rôles de l'ontologie. Généralement, l'ensemble des changements totalement compatibles, que nous notons C_{ch}, exécutés pour passer de V_N à V_{N+1}, peut être calculé de la manière suivante :

$$C_{ch} = \left(P_{IS} \text{ I } P_{CS} \text{ I } P_{AS}\right)$$

Changements rétrocompatibles. L'ensemble des changements rétrocompatibles, que nous notons BC_{ch}, peut être calculé de la manière suivante :

$$BC_{ch} = \left(P_{IA} \text{ Y } P_{CA} \text{ Y } P_{AA}\right) - \left(P_{IN} \text{ Y } P_{CN} \text{ Y } P_{AN}\right).$$

Les changements rétrocompatibles préservent les rôles de l'ontologie, mais le nouveau contenu, introduit en V_{N+1} au moyen de ces changements, n'est pas utilisé pour élargir les rôles de l'ontologie, par exemple pour référencer les éléments pédagogiques d'un *SAGC* en fonction des nouvelles connaissances introduites à l'ontologie.

Changements incompatibles. Sont les changements qui ne préservent pas les rôles de l'ontologie. L'ensemble des changements incompatibles, que nous notons IC_{ch}, peut être calculé de la manière suivante :

$$IC_{ch} = \left(P_{IN} \text{ Y } P_{CN} \text{ Y } P_{AN}\right)$$

Cependant, selon les rôles de l'ontologie, il est possible que les trois dimensions d'analyse ne soient pas toutes nécessaires pour analyser la compatibilité de la nouvelle version de l'ontologie. Il faudrait alors calculer l'ensemble de changements selon les formules ci-dessus en considérant seulement la ou les dimensions nécessaires.

Référencement Sémantique Évolutif

Rendre utilisable la nouvelle version de l'ontologie ne signifie pas cependant qu'elle soit opérationnelle, c'est-à-dire que les rôles de l'ontologie sont préservés [21]. Comme nous l'avons montré ci-dessus, tout changement incompatible peut produire des inconsistances conduisant à l'altération des rôles de l'ontologie. Quant aux changements rétrocompatibles, les rôles de l'ontologie sont préservés, mais le nouveau contenu, ajouté à l'ontologie, n'est pas utilisé pour assurer leur élargissement.

Dans notre contexte de recherche, l'ontologie est utilisée comme référentiel sémantique pour les éléments pédagogiques d'un *SAGC*. Pour préserver ce rôle de l'ontologie, nous proposons une méthode qui propage les changements ontologiques, effectués pour passer de V_N à V_{N+1}, dans les objets référencés par l'ontologie. La propagation des changements dans les objets référencés signifie la modification de leurs liens de référence, un lien de référence étant le chemin qui relie un objet à une entité ontologique utilisée comme référence sémantique.

Définition d'un UKI

Nous définissons un ***UKI*** *(Uniform Knowledge Identifier)*, comme une chaîne compacte des caractères qui explicite le lien de référence reliant un objet à un de ses repères sémantiques dans un espace de connaissances, ce repère étant identifié avec certitude, de manière claire et unique.

Pour l'illustrer, considérons le UKI suivant : *"http://www.w3.org/ActeursTA_3.0#Formateur"*, où *http://www.w3.org/* donne le chemin d'accès à l'ontologie, *ActeursTA_3.0* donne le nom et la version (3.0) de l'ontologie et *#Formateur* identifie, par son nom (ID)[3], le concept utilisé comme référence.

Modification des UKIs

Afin de préserver le rôle d'une ontologie évolutive utilisée comme référentiel sémantique, nous proposons une méthode de modification des UKIs des objets

[3] Dans une ontologie formalisée en OWL (*Ontology Web Langage*), une entité ontologique peut avoir seulement un ID unique, c'est-à-dire qu'aucune entité de l'ontologie ne peut être identifiée par l'ID utilisé par une autre entité.

référencés, en fonction des changements exécutés pour passer de V_N à V_{N+1}.

Pour illustrer brièvement cette méthode, considérons l'exemple d'une banque d'éléments pédagogiques référencés à l'aide des concepts d'une ontologie, où la seule dimension nécessaire pour l'analyse de la compatibilité est la préservation de la structure conceptuelle de la nouvelle version de l'ontologie. En considérant les changements effectués pour passer de V_N à V_{N+1}, nous pouvons rencontrer trois situations :

- Pour les changements incompatibles qui n'affectent pas les UKIs, car ceux-ci référent à des entités ontologiques non touchées par ces changements, il faudrait modifier seulement le nom de la version. Par exemple, pour le UKI *"http://www.w3.org/ActeursTA_3.0#Formateur"*, le nom *ActeursTA_3.0* de la version V_N est remplacé par le nom *ActeursTA_4.0* de la version V_{N+1}.
- Pour les changements incompatibles qui affectent les UKIs, il faudrait modifier le nom de la version, mais aussi le repère sémantique à l'intérieur de V_{N+1}. Prenons l'exemple d'un changement qui fusionne les concepts *'Formater'* et *'Professeur'* afin d'obtenir le concept *'Facilitateur'* dans la version *ActeursTA_4.0*. Dans ce cas, le concept *'Facilitateur'* de la version *ActeursTA_4.0* inclut la signification des concepts *'Professeur'* et *'Formateur'* de la version *ActeursTA_3.0*. En conséquence, la modification des UKIs peut se faire comme nous le montrons ci-dessous.

UKIs référant à V_N
http://www.w3.org/ActeursTA_3.0#Formateur
http://www.w3.org/ActeursTA_3.0#Professeur
UKIs modifiés, référant à V_{N+1} (les deux UKIs ont la même forme)
http://www.w3.org/ActeursTA_4.0#Facilitateur

- Pour les changements compatibles et ceux rétrocompatibles, les UKIs ne sont pas affectés. Dans ce cas, il faudrait changer seulement le nom de la version dans les UKIs. Toutefois, pour prendre en compte les nouvelles connaissances introduites dans la version V_{N+1} par les changements rétrocompatibles il faudrait définir, au besoin, de nouveaux UKIs.

Conclusion

Si les travaux sur la construction et l'utilisation des ontologies ont déjà un bon nombre d'années, ceux sur l'évolution sont bien plus récents. Ces derniers constituent alors le pas suivant à faire dans les recherches actuelles, principalement en ce qui concerne la définition des méthodes pour supporter l'évolution de l'ontologie [5].

Dans ce contexte, la contribution principale de notre article est d'ordre méthodologique : nous avons développé une méthodologie pour la gestion de l'évolution en unifiant dans un cadre cohérent des approches existantes dans la littérature et en proposant de nouvelles étapes portant sur l'analyse de la compatibilité de la nouvelle version de l'ontologie et son opérationnalisation.

Les méthodes pour supporter ces deux dernières étapes sont actuellement presque inexistantes [22]. Pour répondre à ces besoins, nous avons proposé l'ébauche de deux méthodes :

- une méthode pour l'analyse des effets des changements sur la relation de compatibilité entre les versions d'une ontologie selon trois dimensions : préservation des instances, préservation de la structure conceptuelle et préservation de l'axiomatisation de l'ontologie.
- une méthode pour assurer un référencement sémantique évolutif des objets référencés par des entités ontologiques. Ce référencement évolutif est réalisé par la modification des UKIs (*Uniform Knowledge Identifier*) en fonction de l'analyse des changements exécutés pour passer d'une version de l'ontologie à une autre. Nous précisons que cette méthode est essentielle pour assurer la préservation des rôles de l'ontologie utilisée comme référentiel sémantique dans un *SAGC* distribué sur le Web.

La suite de nos travaux consiste à raffiner les deux méthodes et à les implémenter dans un outil capable de supporter des étapes essentielles du processus d'évolution de l'ontologie : l'analyse de compatibilité et la mise en opération de la nouvelle version d'une ontologie utilisée comme référentiel sémantique.

Références

[1] Paquette, G., et Rosca, I. (2002). Organic Aggregation of Knowledge Objects in Educational Systems. Canadian Journal of Learning and Technology, vol. 28(No. 3), 11-26.

[2] Charlet, J., Bachimont, B., et Troncy, R. (2003). Ontologies pour le Web sémantique. In J. Charlet & P. Laublet & C. Reynaud (Eds.), Web sémantique: Action_spécifique_32_CNRS/STIC.

[3] Klein, M. (2002a). Supporting evolving ontologies on the internet. Paper presented at the Proceedings of the EDBT 2002 PhD Workshop, Prague, Czech Republic.

[4] Noy, N., et Klein, M. (2003). Ontology evolution: Not the same as schema evolution. Knowledge and Information Systems, 5.

[5] Stojanovic, L., Maedche, A., Motik, B., et Stojanovic, N. (2002). User-driven Ontology Evolution Management. Paper presented at the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02), Sigüenza, Spain.

[6] OntoWeb. (2002). A survey on methodologies for developing, maintaining, evaluating and reengineering ontologies (IST-2000-29243 - Deliverable 1.4). [7] Klein, M., et Noy, N. (2003). A component-based framework for the ontology evolution. Paper presented at the Workshop on Ontologies and Distributed Systems, IJCAI-2003, Acapulco, Mexico.

[8] Maedche, A., Motik, B., Stojanovic, L., Studer, R., et Volz, R. (2003). Ontologies for Enterprise

Knowledge Management. Intelligent Infromation Processing, March/April 2003.

[9] Oberle, D., Volz, R., Motik, B., et Staab, S. (2004). An extensible ontology software environment. In S. Staab & R. Studer (Eds.), Handbook on Ontologies (pp. 299-320): Springer Verlag.

[10] Klein, M. (2002b). Versioning of distributed ontologies (Deliverable D20 V1.1, EU/IST Project WonderWeb).

[11] Noy, N., et Musen, M. (2003). Ontology Versioning as an Element of an Ontology-Management Framework. IEEE Intelligent Systems, in press.

[12] Stojanovic, L., et Motik, B. (2002). Ontology Evolution within Ontology Editors. Paper presented at the Knowledge Acquisition, Modeling and Management (EKAW), Siguenza, Spain.

[13] Stojanovic, L., Stojanovic, N., et Handschuh, S. (2002). Evolution of the Metadata in the Ontology-based Knowledge Management Systems. Paper presented at the Experience Management 2002, Berlin, Germany.

[14] Ding, Y., et Fensel, D. (2001). Ontology Library Systems: The key for successful Ontology Reuse. Paper presented at the First Semantic Web Working Symposium (SWWS'1), Stanford, USA.

[15] Paquette, G. (2002). Modélisation des connaissances et des compétences. Québec: Presses de l'Université du Québec.

[16] Oliver, D. E., Shahar, Y., Musen, M., et Shortliffe, E. H. (1999). Representation of change in controlled medical terminologies. AI in Medicine, 15(1), 53–76.

[17] Euzenat, J. (1995). Building consensual knowledge bases: context and architecture. In N. Mars (Ed.), Towards very large knowledge bases (pp. 143-155). Amsterdam: IOS press.

[18] Sunagawa, E., Kozaki, K., Kitamura, Y., et Mizoguchi, R. (2003). An Environment for Distributed Ontology Development Based on Dependency Management. Paper presented at the International Semantic Web Conference (ISWC), Florida, USA.

[19] Wache, H., Vogele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., et Hfibner, S. (2001). Ontology-based integration of information - a survey of existing approaches. Paper presented at the IJCAI Workshop on Ontologies and Information Sharing.

[20] Heflin, J. (2001). Towards the Semantic Web: Knowledge and representation in a dynamic, distributed environment. Faculty of the Graduate School of the University of Maryland.

[21] Ding, Y., Fensel, D., Klein, M., Omelayenko, B., et Schulten, E. (2004). The role of Ontologies in eCommerce. In S. Staab & R. Studer (Eds.), Handbook on Ontologies (pp. 593-616): Springer Verlag.

[22] Klein, M., & Fensel, D. (2001). Ontology versioning for the Semantic Web. Paper presented at the International Semantic Web Working Symposium (SWWS), USA.