Conteneurs de connaissances : une approche fondée sur les usages pour le Web Sémantique

P.-A. Champin Y. Prié A. Mille

Équipe Cognition et Expérience*- LISI
Bât. Nautibus - Université Claude Bernard de Lyon 1
43, Boulevard du 11 novembre 1918
69622 Villeurbanne Cedex
{champin,yprie,amille}@lisi.univ-lyon1.fr

Résumé

Dans la vision du Web Sémantique proposée par Tim Berners-Lee, les agents logiciels doivent être capables de retrouver des connaissances pertinentes pour la tâche d'un utilisateur. Nous proposons dans cet article un cadre général pour prendre en compte les usages, centré sur la notion de conteneur de connaissances, et sur des modèles d'utilisation de ces conteneurs.
Nous présentons ensuite deux projets dans lesquels les auteurs sont impliqués, où ce cadre est implanté : ARDECO, qui vise à assister la réutilisation en CAO, et RECIS, dont le sujet est l'indexation du contenu de documents audiovisuels. Nous concluons que toute utilisation d'une ressource devient annotation de cette ressource, et constitue ainsi un nouveau conteneur de connaissances.

Mots Clef

Conteneurs de connaissances, expérience, tâches, usages, Web Sémantique

Abstract

In Tim Berners-Lee's vision of the Semantic Web, software agents must be able to retrieve relevant pieces of knowledge for a user's task. In this paper we propose a general framework intended to take uses into account, based on the notion of knowledge container, and modeling the uses of these containers.
We then present two projects involving the authors, where this framework is being implemented : ARDECO, aiming at assisting reuse in CAD, and RECIS, aiming at indexing the content of audiovisual documents. We conclude that every use of a resource becomes an annotation of that resource, and hence a new knowledge container.

Keywords

Knowledge container, experience, tasks, uses, Semantic Web

*⟨URL :http ://experience.univ-lyon1.fr/⟩

1 Introduction

En une dizaine d'années, le *World Wide Web* est devenu un moyen de partage d'information tellement populaire que la quantité de données disponibles dépasse de loin ce qu'un utilisateur peut gérer seul. Pourtant, la majorité des informations disponibles est destinée aux utilisateurs plutôt qu'à des systèmes informatiques, limitant ainsi l'aide que ces derniers peuvent apporter. Tim Berners-Lee suggère dans [4] que la prochaine étape du Web, qu'il appelle Web Sémantique, devra résoudre ces problèmes.

Le Web Sémantique doit permettre l'interopérabilité entre des agents hétérogènes. Cela suppose qu'outre des données, il doit mettre à disposition des métadonnées, permettant aux agents informatiques d'« apprendre » comment interpréter les données ; ces métadonnées peuvent être disponibles sous la forme de Définitions de Type de Document (DTD), Schémas XML ou annotations RDF, etc. D'après [2], le Web Sémantique devrait même être capable de fournir une réponse argumentée à des questions en langue naturelle, alors que les moteurs de recherche actuels ne fournissent qu'une liste de ressources (censées contenir les éléments de réponse) en retour d'une requête, généralement exprimée comme une liste de mots-clés (censés être représentatifs de la question). Plus généralement, le Web Sémantique doit être plus orienté vers l'utilisateur. Notamment, il doit permettre d'accéder à des *connaissances* plutôt qu'à de simples *données*, *i.e.* des données pertinentes pour la tâche de l'utilisateur. Par ailleurs, des paramètres sociaux doivent être pris en compte, tels que la confidentialité et la fiabilité des informations (*Web of Trust*).

Dans cet article, nous nous intéressons à l'accès aux connaissances. Nous pensons que les ressources du Web peuvent être utilisées de plusieurs manières par des personnes ayant des tâches différentes, et que tous ces usages ne peuvent pas être énumérés *a priori*. Par ailleurs, l'explicitation des connaissances portées par

une ressource relève nécessairement d'une tâche particulière. Ainsi, un agent adaptatif ne devrait pas s'appuyer *uniquement* sur les connaissances explicitées, mais également sur les usages effectifs de la ressource, afin d'être en mesure de répondre pertinemment à une tâche non encore identifiée. Cet article présente un cadre permettant de prendre en compte les usages.
Dans la section 2, nous présentons les notions qui sont à la base de notre approche. Ensuite nous présentons deux projets exploitant ce cadre : ARDECO (section 3), dont le domaine est la conception mécanique, impliquant des documents fortement structurés issus d'un logiciel de CAO, et RECIS (section 4), dont le domaine est la recherche de documents faiblement structurés, impliquant l'audiovisuel. Enfin, nous discutons les avantages de notre approche dans la section 5, et la comparons aux autres travaux dans ce domaine (section 6).

2 Usages et conteneurs de connaissances

Dans cette section, nous présentons les notions qui sous-tendent notre approche, et qui reprennent et étendent les concepts présentés dans [18].

> *Ces notions seront illustrées par un exemple simple : un moteur en ligne de recherche à base de mots-clés.*
>
> *Notons tout d'abord que c'est bien l'utilisation du moteur de recherche lui même que nous décrivons, et non celle du navigateur qui sert d'intermédiaire entre lui et l'utilisateur humain — le navigateur étant beaucoup plus versatile, son modèle d'utilisation, plus vaste, ne conviendrait pas à un tel exemple...*

2.1 Modèle d'utilisation, modèle de tâche

Tout système informatique, y compris un agent logiciel accédant au Web ou au futur Web Sémantique, est appelé à être en interaction, directe ou indirecte, avec un agent humain : l'« utilisateur ». Ce dernier peut, grâce au système, manipuler conceptuellement un certain nombre d'« objets » du système. Nous regroupons cela sous la notion de modèle d'utilisation.

Modèle d'utilisation : Le modèle d'utilisation d'un système informatique est l'ensemble des *objets* de l'application manipulés par l'utilisateur, et de toutes les *opérations* que ce dernier peut effectuer sur ces objets.

Toute tâche impliquant le système peut évidemment être décrite à l'aide des éléments du modèle d'utilisation. Bien que de tels modèles aient longtemps été laissés implicites, encapsulés dans le code du logiciel, cet état de fait est en train de changer sous l'influence de la conception objet, et des langages intermédiaires permettant une description « externe » des objets disponibles à l'utilisateur (notamment pour les interfaces graphiques — cf. par exemple GladeXML[1]).

> *Le modèle d'utilisation de notre moteur de recherche est relativement simple : il implique des requêtes qui sont des ensembles de mots-clés et retournent une liste de résultats. Chaque résultat est un lien vers une ressource du Web.*
>
> *Les requêtes sont soumises, les résultats d'une liste de résultat sont traversés pour atteindre la ressource correspondante, et peuvent être traversés en sens inverse pour revenir à la liste de résultats (généralement en utilisant la fonction* back *du navigateur).*

Bien que la notion de *modèle de tâche* ait été largement étudiée par ailleurs [7, 13], nous pouvons la considérer selon l'éclairage apporté par celle de modèle d'utilisation : lors de la réalisation d'une tâche particulière, il existe un certain nombre de relations et de contraintes supplémentaires entre les éléments du modèle d'utilisation. Nous pouvons donc donner la définition qui suit pour les modèles de tâche.

Modèles de tâche : Tout modèle de tâche est une restriction du modèle d'utilisation décrivant les propriétés de ses objets qui sont toujours vérifiées lors de la réalisation de la tâche en question[2].

Nous attirons l'attention du lecteur sur le fait que les modèles que nous définissons ici rendent compte de la tâche plutôt qu'ils ne la prescrivent. En cela, notre définition est plus large quel celles habituellement adoptées pour la notion de modèle de tâche, tout en les englobant (nos modèles de tâche peuvent éventuellement être prescriptifs).

> *Nous décrivons ici deux tâches, typiques dans l'utilisation d'un moteur de recherche, comme des restrictions du modèle d'utilisation ci-dessus. La tâche de parcours des résultats consiste, étant donnée une liste de résultats, à en traverser un ou plusieurs liens*

[1] ⟨URL:http://glade.gnome.org/⟩, langage XML de description d'interfaces graphiques pouvant être utilisé à l'exécution du programme.

[2] Notons que c'est une vue centrée sur le système. Du point de vue de l'utilisateur, une tâche est motivée par un objectif, souvent non exprimé dans le système ou renvoyant à des éléments extérieurs.
Notons également que le modèle d'utilisation d'un système peut être perçu comme un modèle de tâche dans un système plus large (par exemple, le modèle d'utilisation d'un logiciel de traitement de texte peut être perçu comme un modèle de tâche dans une suite bureautique).

(vers la ressource puis de retour vers la liste). Le dernier lien peut n'être traversé que dans un sens (vers la ressource), auquel cas la tâche est considérée comme un succès, ou dans les deux sens (retour vers la liste) auquel cas la tâche est un échec.

Un autre type de tâche est le filtrage d'une requête *: étant donnée une liste de résultats retournée par une requête R_1, une nouvelle requête R_2 est soumise, qui contient R_1 ainsi que d'autres mots-clés. L'objectif d'une telle requête est de réduire le nombre de liens dans la liste de résultats.*

2.2 Cas d'utilisation

Le système constitué des objets « mobilisés » dans la tâche change d'*état* à chaque opération décidée par l'utilisateur. On peut donc considérer toute trace de toute utilisation du logiciel comme une séquence d'états et d'opérations. Ces traces sont difficilement exploitables en tant que telles (c'est sur ce type de matériau que sont généralement utilisées les techniques de fouille de données afin d'en extraire des connaissances), d'où l'intérêt de la notion plus précise de cas d'utilisation[3].

Cas d'utilisation : On appelle ainsi tout partie de la trace d'utilisation qui instancie un modèle de tâche. On dit que le modèle de tâche permet d'expliquer le cas d'utilisation.

La figure 1 représente une trace d'utilisation de notre moteur de recherche. La séquence 1–2 est un cas d'utilisation, instance du modèle de tâche filtrage de requête*, et correspond donc à cette tâche. Les séquences 2–5 et 6–9 correspondent à des tâches de* parcours des résultats*, aboutissant à un échec pour la première (car elle se termine sur un retour à la liste des résultats), et à un succès pour la seconde (puisqu'elle se termine après la traversée d'un lien).*

Notons que la séquence 5–6 n'est expliquée par aucun de nos modèles de tâche : ce n'est pas un filtrage de requête tel que nous l'avons défini, car la nouvelle requête ne reprend pas tous les mots de la précédente[4]. C'est donc un exemple de tâche non encore identifiée, qui ne peut pas être expliquée par les modèles de tâche disponibles.

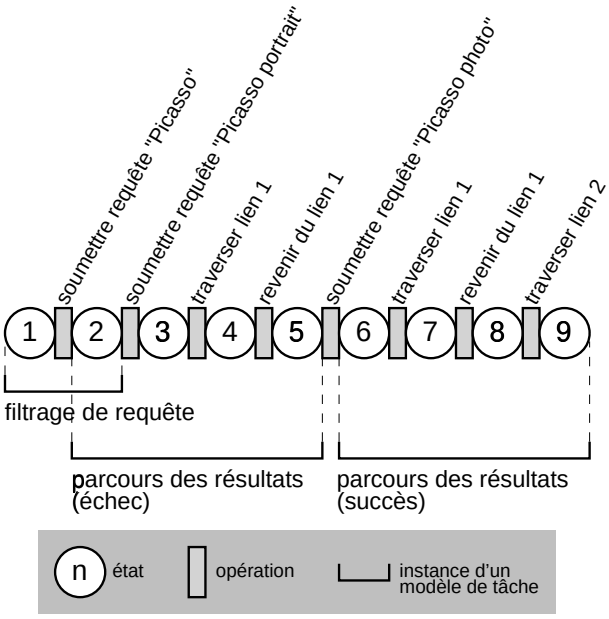

FIG. 1 – Exemple de trace d'utilisation instanciant des modèles de tâche.

2.3 Conteneurs de connaissances

Nous ne définirons pas le terme *conteneur de connaissances* (CC) comme nous l'avons fait avec les précédentes notions, simplement parce que ce terme recouvre les mêmes entités que le terme « ressource ». En effet tout CC disponible sur le Web Sémantique est évidemment une ressource (tout ce qui est sur le Web en est une), et réciproquement, toute ressource peut, d'un certain point de vue, apporter des connaissances à l'utilisateur. Cependant, le terme « conteneur de connaissances » nous permet de mettre l'accent sur certains aspects intéressants.

Une ressource peut en effet avoir plusieurs formes, en fonction de nombreux critères : le temps (versions, corrections), le médium (écran, audio), le format (HTML, PDF), etc. D'autre part, certaines ressources ne sont pas accessible sous forme d'un flux d'octets mais ont d'autres types d'utilisation (comme les ressources de type mailto:) ; elles sont plus assimilables en cela à des *services*. Chaque forme, chaque utilisation d'une ressource donnée est appropriée pour une certaine tâche, et peut donc fournir des connaissances pour cette tâche ; cela justifie de considérer la ressource comme un conteneur de « connaissances ».

Par ailleurs, si les ressources « contiennent » des connaissances, elles peuvent le faire de façon transparente (boîte blanche) ou opaque (boîte noire). Dans le premier cas, les CC sont structurés, sous une forme connue et exploitable, les connaissances peuvent en être extraites par des agents logiciels ; dans le second

[3] Ces « cas d'utilisation » ne doivent pas être confondus avec les cas d'utilisation définis par UML [20], qui sont génériques et seraient donc plus proches de nos modèles de tâche.

[4] Du point de vue de l'utilisateur, en revanche, c'est une sorte de filtrage : le lecteur aura compris que l'utilisateur recherche des portraits photographiques du peintre, et que l'avant dernière requête aura retourné des ressources relatives aux portraits peints par Picasso.

cas, les CC sont non structurés et donc opaques du point de vue logiciel, seul les utilisateurs humains ont la capacité d'en extraire les connaissances pertinentes. Comme nous l'avons fait remarquer en introduction, on ne peut prédire toutes les utilisations d'un CC, ni donc toutes les connaissances qui peuvent en être extraites. Les CC boîtes blanches peuvent donc devenir boîtes noires dans le cadre de tâches non encore identifiées.
Ce dernier fait n'empêche évidemment pas de prendre en compte les connaissances directement extractibles dans les CC. D'ailleurs, la première des applications que nous présenterons par la suite s'appuie largement sur la structure interne de ses CC.

> *Les principaux CC intervenant dans notre exemple du moteur de recherche sont les ressources du Web. Ces CC ne sont pas à proprement parler opaques puisque le moteur de recherche peut en extraire des mots-clés ; mais d'un autre coté, ces mots-clés ne représentent qu'une très faible partie des connaissances intéressantes pour l'utilisateur. On voit que les conteneurs de connaissances ne sont pas strictement des boîtes noires ou blanches.*

L'idée principale dans notre approche est que les usages fournissent aux agents des exemples concrets d'utilisation des CC. Dans des contextes similaires (*i.e.* des cas d'utilisation similaires) il est possible que le même CC s'avère pertinent pour l'utilisateur (même si cette pertinence n'est pas « compréhensible » directement par l'agent informatique). Ceci peut être réalisé à l'aide du paradigme du raisonnement à partir de cas (RàPC), dont les mécanismes ont été largement étudiés dans la communauté de l'Intelligence Artificielle [1, 15]. Nous présentons dans les sections suivantes deux applications à des domaines différents, du cadre que nous venons de décrire.

3 Exemple 1 : ARDECO

Les concepteurs de systèmes complexes sont enclins à réutiliser leur expérience de conception. En effet, réutiliser un artefact ou un processus de conception est souvent plus rapide et plus facile que de concevoir à partir de zéro ; cela permet ainsi une réduction des temps et des coûts de conception. Cependant, trouver le matériau réutilisable n'est pas toujours aisé, et les méthodes visant à la capitalisation des connaissances sont souvent considérées trop contraignantes par les concepteurs.
Bien que la plupart des activités de conception est aujourd'hui assistée par ordinateur, les systèmes de CAO ne fournissent pas aux utilisateurs une aide efficace à la réutilisation. Le but du projet ARDECO[5] est de mettre en place un tel système d'assistance : un agent logiciel aidant les concepteurs à retrouver, dans un ensemble de traces d'utilisation du système, des éléments réutilisables pour leur tâche courante. De par cette fonction de recherche d'information dans un ensemble de ressource volumineux et organisé de façon peu structurée *a priori*, ce type d'agent s'apparente aux problématiques du Web Sémantique, malgré son domaine d'application plus restreint.

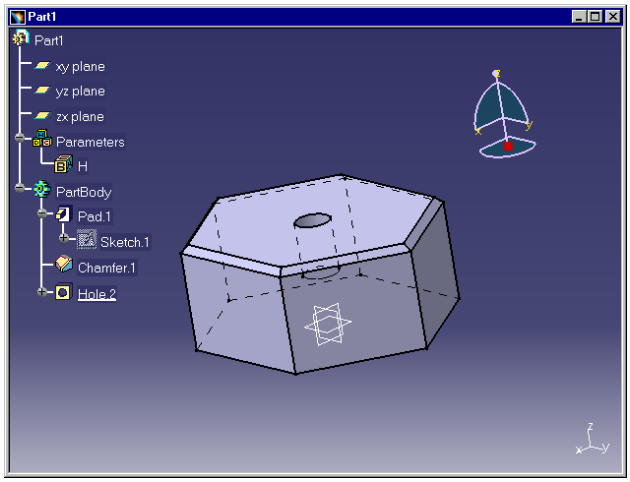

FIG. 2 – Copie d'écran CATIA, ou l'on peut voir les deux représentations de l'artefact (hiérarchique et géométrique). Toutes deux peuvent être manipulées par le concepteur.

3.1 Modèle d'utilisation et modèles de tâche dans ARDECO

Modèle d'utilisation et conteneurs de connaissances. Puisque ARDECO s'intéresse à un logiciel de CAO spécifique (à savoir CATIA de Dassault Systèmes), le modèle d'utilisation est défini précisément, et les premiers conteneurs de connaissances utilisés (artefacts de conception) ont une structure connue : ils sont représentés sont représentés dans les documents CAO (fichiers enregistrés par l'application) par un arbre de spécifications, où chaque élément est une instance d'une classe, possède un certain nombre de composants, d'attributs, ainsi que des liens éventuels vers d'autres éléments de l'arbre. L'activité du concepteur consiste à ajouter, modifier ou supprimer des éléments dans cet arbre, soit directement, soit à travers la représentation géométrique de l'artefact (cf. figure 2).

Modèles de tâche. Contrairement au modèle d'utilisation, les modèles de tâche ne sont pas précisément définis. En effet, le nombre d'objets manipulables dans le modèle d'utilisation est élevé, ainsi que le nombre de

[5]Aide à la Réutilisation D'Épisodes de COnception. Faisant partie du programme PROSPER du CNRS, ce projet implique des chercheurs dans deux domaines : informatiques et psychologie cognitive, ainsi qu'un partenaire industriel : Dassault Systèmes.

relations possibles entre ces objets. De plus, l'activité de conception fait appel, par essence, à la créativité de l'utilisateur ; elle est caractérisée par le fait que les buts ne sont pas définis à l'avance, mais au fur et à mesure de l'activité [5]. Il en résulte une variété des tâches trop grande pour pouvoir isoler des modèles de tâche — tout au moins des modèles de tâches suffisamment spécifiques pour être effectivement réutilisables (les tâches identifiables dans l'application sont trop génériques, par exemple : dessin de profil 2D).

3.2 Cas d'utilisation : les épisodes

L'absence de modèles de tâche préalables rend nécessaire une utilisation extensive de toutes les connaissances disponibles, notamment grâce à la richesse du modèle d'utilisation (forte structuration hiérarchique des artefacts de conception), mais également grâce à une interaction avec l'utilisateur (autant que possible sans perturber son activité). Nous nous appuyons pour cela sur la notion d'*épisode*, issue de travaux en psychologie et ergonomie cognitive. Dans notre domaine, on peut définir un épisode de conception comme « une partie de l'activité de conception entre le moment où un but est identifié, et le moment où ce but est considéré comme atteint ». Cette définition nous permet de considérer les épisodes comme des cas d'utilisation, instances d'un modèle de tâche (*a priori* inconnu du système).

L'instrumentation de CATIA permet aux concepteurs de déclarer explicitement qu'un cas d'utilisation commence ou se termine. Ils peuvent également annoter leur travail pour indiquer quel objectif ils poursuivent à un moment donné. Enfin, des observations d'utilisateurs de CATIA, menées par les chercheurs en psychologie cognitive [5], ont mis en évidence des indices comportementaux permettant au système de détecter automatiquement les états délimitant un cas d'utilisation.

3.3 Recherche des conteneurs de connaissances

Comme nous l'avons dit, nous considérons les épisodes comme des instances connues de modèles de tâche non complètement connues — ce qui revient à considérer chaque épisode comme l'unique instance d'un modèle de tâche extrêmement spécifique. Ce sont la richesse du modèle d'utilisation et la transparence des CC qui permettent de comparer entre eux les épisodes, et donc indirectement, les modèles de tâche sous-jacents. La comparaison peut utiliser des mesures de similarité statique (comparant les états) ou dynamiques (comparant les transitions entre états).

La trace de l'utilisation de CATIA par le concepteur est enregistrée en permanence ; lorsque il demande l'assistance du système, l'épisode courant est comparé aux autres cas d'utilisation de la base. Les éléments les plus similaires (états ou épisodes) sont alors proposés pour la réutilisation.

De plus, les éléments effectivement réutilisés par le concepteur sont mis en relation avec l'épisode courant, et permettent de définit un cas de *réutilisation* précisant comment la réutilisation a été effectuée. Ce lien entre matériau réutilisé et cas de réutilisation peut être lui aussi mis à profit par le système pour juger de la similarité des épisodes, et pour suggérer des adaptations similaires à l'avenir [6].

On voit que chaque type de structure mise en place devient conteneur de connaissances à son niveau : les états fournissent des connaissances sur les spécifications CAO, les épisodes sur les états, les épisodes de réutilisation sur d'autres épisodes. Toutes ces structures sont représentées dans un graphe à l'aide du langage RDF [16] ; celui-ci permet en effet de mixer différents niveaux de vocabulaire, notamment les données et les méta-données.

4 Exemple 2 : RECIS

Notre second exemple prend place dans le domaine des systèmes d'information audiovisuelle, dans le cadre du projet RECIS[6]. L'objectif de ce projet est de participer à la construction de nouveaux outils permettant d'accéder et d'exploiter des documents audiovisuels : d'une part en facilitant l'indexation sémantique des documents par l'extraction d'information multimédia, et d'autre part en exploitant les sessions de recherche pour aider l'utilisateur à trouver rapidement des requêtes pertinentes. En cela, son lien avec le Web Sémantique est plus évident que pour l'exemple précédent.

4.1 Les documents audiovisuels comme conteneurs de connaissances « opaques »

Les documents audiovisuels (AV) diffèrent des documents textuels en cela qu'ils n'ont pas (encore) de structure documentaire explicite de haut niveau. La seule structure reconnue à l'heure actuelle se limite en effet à celle d'une superposition d'un flux d'images animées et d'un flux audio.

Bien entendu, cette caractéristique pourrait n'être que transitoire, et des travaux tels que les initiatives MPEG-4 et MPEG-7 ont pour objectif de mettre en place des standards documentaires pour la description structurée des documents AV. Cela dit, cette absence relative de structure a également une raison : les usages innovants des documents audiovisuels (tels la navigation, la recherche, le montage à la volée, etc.) restent à inventer. Aucune application phare n'a encore permis l'émergence et la stabilisation de façons de décrire

[6] Recherche et Exploration de Contenus par l'Image et le Son, projet RNRT impliquant France Télécom Recherche et Développement et l'INRIA (projet Imédia).

les documents qui soit suffisamment partagées : c'est pourquoi les documents audiovisuels restent pour l'instant des conteneurs de connaissances opaques.
Une des conséquences de cet état de fait est que quel que soit le nouveau moyen d'exploitation des documents audiovisuels que l'on souhaite mettre en place (et qui diffère bien entendu de la simple visualisation), il convient d'expliciter la structure documentaire sur laquelle on va prendre appui. Dans le cas le plus général, il est nécessaire de l'établir complètement.[7]
Les remarques que nous venons de faire nous incitent à penser que pour permettre à des agents informatiques de manipuler des documents audiovisuels et ainsi d'enrichir les outils d'accès et de recherche, il est nécessaire de fournir des modèles de description qui puissent s'adapter à des tâches très variées. Ces modèles doivent alors permettre de définir et d'exploiter un nombre non limité de structures de description pour un document ou un ensemble de documents, qui prendra alors la forme générale d'un graphe (au lieu de rester limité à la structure arborescente existant dans le paradigme XML/DTD).

4.2 Modèle d'utilisation et modèles de tâche dans RECIS

Le modèle de description que nous présentons dans cette section a été présenté dans [17] et étendu dans [11], et a pour nom Strates-IA. Nous n'en présenterons que ce qui correspond à notre cadre.

Modèle d'utilisation. Un *élément d'annotation* (EA) est tout d'abord un terme, comme ⟨*Chirac*⟩ ou ⟨*Mouton*⟩, qui annote un fragment de document appelé *unité audiovisuelle* (UAV). Il existe deux façons d'en compléter la structure : tout d'abord en lui fournissant des attributs (par exemple ⟨*Plan, ImageReprésentative=plan27.jpg*⟩) ; ensuite, et surtout, en le mettant en relation avec un autre élément d'annotation. Deux éléments d'annotation quelconques peuvent ainsi être mis en relation (appelée *relation élémentaire*), y compris si ceux-ci annotent deux unités audiovisuelles différentes[8]. L'annotation d'un ensemble de documents (flux sur la figure 3) se ramène alors à la construction d'un graphe d'annotation, contenant autant d'unités audiovisuelles annotées que nécessaire pour exprimer les points de vues, *i.e.* les tâches liées à l'exploitation des documents.
Les éléments d'annotation sont spécifiés à l'aide d'*éléments d'annotation abstraits* (EAA). Un EAA définit le terme de l'EA qui lui est lié, ainsi que ses attributs éventuels (⟨*EAA :Plan, ImageReprésentative*⟩).
Éléments d'annotation, éléments d'annotation abstraits et unités audiovisuelles définissent le modèle d'utilisation des Strates-IA, dont les éléments sont structurés dans un graphe étiqueté, où les informations pourront être retrouvées.

Un élément du graphe est dit « dans le contexte » d'un autre s'il existe un chemin entre eux dans le graphe. La recherche d'un nœud est réalisée en décrivant comme un motif de graphe le contexte qu'on désire lui trouver, que nous appelons *graphe potentiel*. Chaque recherche se fait alors par le calcul d'un isomorphisme de sous-graphe partiel [19]. Nous considérons ainsi que de façon générale toute exploitation du graphe (i.e. la recherche d'un élément en son sein) doit être considérée de façon contextuelle, puisque la signification d'une annotation est déterminée par son contexte dans le graphe (⟨*Chirac*⟩ en relation avec ⟨*Serrer la main*⟩ ne s'interprète pas de la même façon que ⟨*Chirac*⟩ seul).

Modèles de tâche. Les éléments d'annotation abstraits sont regroupés en *dimensions d'analyse* (DA) qui définissent des classes de termes. Ces classes ne sont pas obligatoirement disjointes (par exemple ⟨*DA :Politiciens*⟩ ou ⟨*DA :SalonAgriculture*⟩ peuvent avoir ⟨*Chirac*⟩ en commun). Les dimensions d'analyse regroupent des EAA qui doivent être utilisés de la même façon pour décrire des documents dans le cadre d'une même tâche. Elles représentent donc des modèles de tâche.
Si l'annotation peut être guidée par l'utilisation d'une dimension d'analyse particulière, à un niveau supérieur les *schémas de description* (SD), qui spécifient les relations à mettre en place entre éléments d'annotation (modèle de tâche plus précis), peuvent également être utilisés. Par exemple, un schéma de description ⟨*SD :JournalTélévisé*⟩ pourra spécifier qu'une unité audiovisuelle annotée par un élément d'annotation issu de la dimension d'analyse ⟨*DA :Politiciens*⟩ devrait être mis en relation élémentaire avec un EA extrait de la DA ⟨*DA :Action*⟩, annotant une unité audiovisuelle du même flux (cf. ⟨*Chirac*⟩ et ⟨*Serrer la main*⟩ sur la figure 3). Les SD représentent donc des modèles de tâche, plus élaborés que les DA.

4.3 Recherche de conteneurs de connaissances

Les dimensions d'analyse et les schémas de description peuvent être plus ou moins partagés entre communautés d'utilisateurs. Par exemple des institutions comme l'INA ont des manières de décrire des documents audiovisuels qui sont plus standardisés que celles des particuliers. Le modèle des Strates-IA nous semble alors adapté, car il permet de décrire n'importe quel flux

[7] Une autre conséquence — particulièrement intéressante pour la communauté du document est la suivante : chaque structure documentaire mise en place dans cette optique est forcément *sémantique*, ce qui signifie qu'il n'existe pas de structure documentaire « canonique » pour les documents audiovisuels. Ceci contraste avec les documents textuels, dont la tradition typographique séculaire a eu le temps de figer usages et descriptions.

[8] D'où le nom du modèle : Strates Inter-connectées par les Annotations.

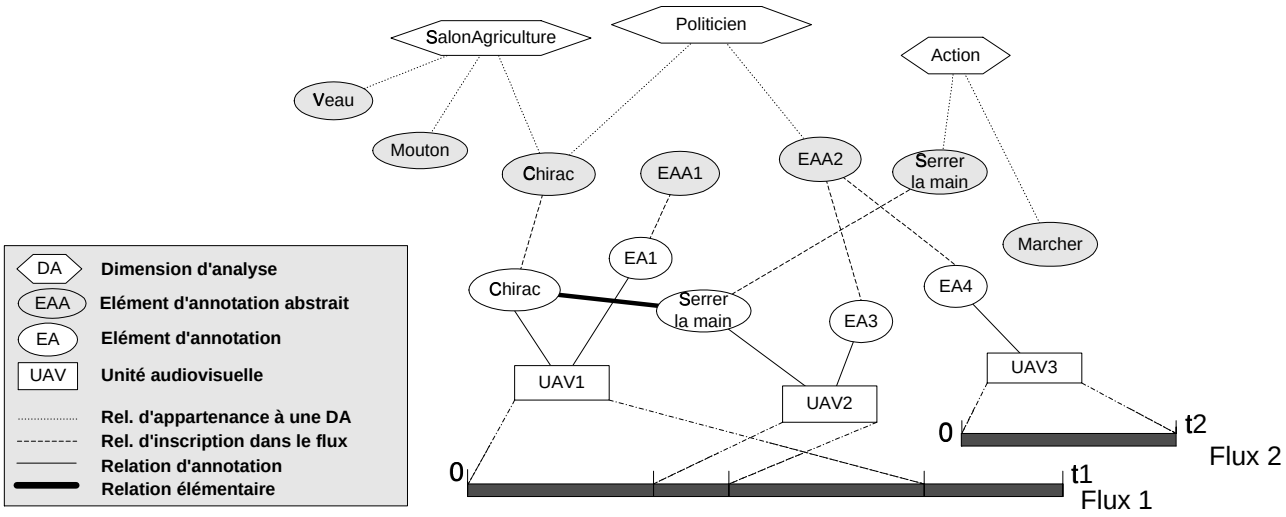

FIG. 3 – Un exemple de documents audiovisuels annotés. Les flux AV sont découpés en unités audiovisuelles annotés par des éléments d'annotation issus de dimensions d'analyse.

audiovisuel de multiples façons : depuis la description d'un document en entier (notice documentaire classique) jusqu'à sa fragmentation en un ensemble structuré d'unités audiovisuelles, et d'une description par simples mots-clés à une représentation de connaissances plus classique (si les dimensions d'analyse et les schémas de description constituent une ontologie formelle, avec des termes ayant une sémantique strictement fixée). Toutes ces mises en place d'un graphe d'annotation conformément à un modèle de tâche (DA ou SD) sont donc les *cas* d'utilisation dans RECIS [11].

Un point important est que, quelle que soit la rigidité du modèle, les utilisateurs peuvent toujours mettre en place des relations supplémentaires entre éléments d'annotation, lesquelles ne sont pas (et pour cause !) définies dans un schéma de description donné. De la même manière, il est possible d'utiliser en même temps des schémas de description différents. Il devient alors possible de mettre en place dans le graphe d'annotation, ainsi que dans les graphes potentiels de recherche, plus de connaissances que celles qui avaient été initialement captées dans les modèles de tâche initiaux — cette pratique se rapproche alors de celle décrite avec le projet ARDECO dans la section précédente.

Bien que le but initial du modèle des Strates-IA ait été de retrouver des documents audiovisuels (UAV) à l'aide de graphes d'annotation, les graphes potentiels permettent également aux utilisateurs de retrouver des nœuds quelconques, tels que EA ou EAA, dans le cadre de leur tâche. Chaque nœud du graphe d'annotation peut en effet porter autant d'information que le document audiovisuel annoté lui-même (par exemple des annotations factuelles sur le nombre de pattes d'un mouton au cours d'un reportage télévisé). C'est pourquoi les documents audiovisuels ne sont pas les seuls conteneurs de connaissances du modèle : n'importe quel sous-graphe du graphe d'annotation peut en fait être considéré comme tel.

5 Discussion

L'approche fondée sur les usages que nous avons proposée à été utilisée dans les deux projets décrits dans cet article, mais également dans d'autre projets[9] appliqués à des domaines différents, tels que l'enseignement à distance (projet PIXED) ou la gestion de connaissances en conception collaborative (projet OSCAR). Le point commun dans ces projets est la grande variété des tâches possibles pour l'utilisateur, d'où la nécessité d'une approche comme la notre permettant la prise en compte de nouveaux usages. Le tableau 1 est un résumé de la façon dont ARDECO et RECIS implantent cette approche.

Les deux projets sont assez différents du point de vue des connaissances disponibles à chaque niveau. Alors que ARDECO dispose de nombreuses connaissances relatives au modèle d'utilisation, puisqu'il est lié à une application précise, RECIS offre un modèle d'utilisation minimal. En revanche RECIS permet une description explicite des modèles de tâche grâce aux dimensions d'analyse et aux schémas de description, alors que seuls les cas d'utilisation, qui sont des instances de ces modèles, peuvent être identifiés dans ARDECO, sans que les modèles de tâche eux même ne soient rendus explicites. Il en résulte une utilisation très différent des ressources impliquées par chaque application, comme boîte noire ou boîte blanche selon le cas.

Il convient de noter, cependant, que les agents construits selon notre approche ne s'appuient pas principalement sur la transparence des ressources en tant

[9] ⟨URL:http://experience.univ-lyon1.fr/projets.html⟩

	ARDECO	RECIS
Modèle d'utilisation	Modèle d'utilisation de CATIA	Unités audiovisuelles, éléments d'annotation, éléments d'annotation abstraits
Modèles de tâche	Inconnu, instancié par les épisodes	Dimensions d'analyse, schémas de description
Cas d'utilisation	Épisodes	Graphe d'annotations
Conteneurs de connaissances	États, épisodes et cas de réutilisation	Documents audiovisuels, n'importe quelle partie du graphe d'annotations

TAB. 1 – Implantation de notre cadre par ARDECO et RECIS

que conteneurs de connaissances, mais plutôt sur la description de leurs usages, qui est représentée à l'extérieur de ces ressources. Ils reposent en effet sur le principe qu'un CC utilisé dans une situation sera réutilisable dans une situation similaire, donc si les descriptions des usages sont similaires. La *pertinence* des résultats d'un tel agent n'est donc plus conditionnée par sa capacité à « comprendre » le contenu du CC ou la tâche de l'utilisateur, mais par sa capacité à décrire et à comparer les usages.
Le second intérêt de l'approche est que les cas d'utilisations peuvent également servir à l'utilisateur lui même (à condition de les lui présenter sous une forme appropriée, bien sûr). Ainsi, toutes les structures mises en place pour décrire les usages des CC deviennent eux mêmes des CC de plein droit. C'est pourquoi les applications que nous avons présentées ne se limitent pas à la remémoration de documents CAO ou audiovisuels, mais de tout élément du modèle.

6 Comparaison avec d'autres approches

La plupart des travaux relatifs au Web Sémantique visent à permettre l'interopérabilité entre agents informatiques, tout en tenant compte des particularités du Web : il est volumineux, ouvert (par opposition à l'hypothèse de « monde fermé »), et dynamique (en changement constant) [14]. Il est communément admis qu'une façon de répondre à ces problème est l'utilisation d'ontologies : des vocabulaires formels dont les relations entre les termes sont précisément définies, partagés par des communautés d'utilisateurs telles que les *Ontogroups* du système ONTOBROKER [9].
Les descriptions fondées sur les ontologies sont inévitablement orientées vers une tâche (ou une famille de tâches) particulière. Bien sûr, dans de nombreux domaines l'utilisateur n'est pas censé faire preuve de créativité, et encore moins d'agir en dehors d'un modèle de tâche précis — un exemple parlant étant le commerce électronique. Dans le système SHOE [14], les ontologies peuvent être définies et étendues au sein de tout document HTML, mais chaque assertion est de plus attaché à son « prétendant » (en anglais : *claimant*), l'agent humain ou informatique responsable de l'assertion. Ainsi le système est capable de gérer les contradictions et les problèmes de fiabilité.
De plus, les ontologies peuvent fournir aux agent informatiques de puissants moyens d'inférence pour la tâche correspondante. Par exemple, le langage OIL [12] a été conçu de façon à limiter la complexité des mécanismes d'inférence, afin de répondre au caractère volumineux du Web. Il est fondé sur les logiques de description, dont les algorithmes d'inférence ont été largement étudiés. OIL a récemment été intégré au projet DAML[8], qui vise à décrire des ontologies à l'aide de RDF.
Cependant il existe également des travaux s'appuyant sur les usages, notamment un modèle d'utilisation minimal du Web, n'impliquant que des ressource et des liens entre elles. C'est en effet tout ce dont le Web est fait (avec le langage HTML qui est à son origine), et donc ce dont le Web Sémantique émergera probablement.
Les anciens moteurs de recherche ne s'appuyait que sur le contenus des documents HTML pour évaluer leur pertinence par rapport à une requête constituée de mots-clés (comme celui de notre exemple, section 2) ; le taux de bruit de ces moteurs était énorme. à l'inverse, le moteur de recherche GOOGLE[10] s'appuie sur le texte portant des hyperliens vers les pages. Cette approche part du principe qu'une page est pertinente par rapport à un mot-clé si d'autres pages (et donc leurs auteurs) pointent vers cette page à l'aide de ce mot-clé. Cette approche s'est montrée fructueuse, et nous pensons que c'est parce qu'elle se base effectivement sur les usages des pages (le fait de mettre en place un lien vers elles) plutôt que sur le contenu. Une extension de cette approche consisterait à mettre en place des structures de liens plus riches, comme celles proposées par RDF, XLink [10] ou le projet Annotea [3].
Nous pensons que les approches centrées sur les usages, comme celle que nous proposons, complètent les approches fondées sur les ontologies : ces dernières fournissent des description de ressources orientées vers une tâche donnée, décrite *a priori*, associées à des mécanismes d'inférences puissants. Au contraire, notre approche ne fournit pas de mécanismes d'inférence

[10] ⟨URL:http://www.google.com/⟩

stricte, mais permet de s'adapter *a posteriori* à des tâches non complètement identifiées.

Conclusion

Dans cet article, nous avons proposé un cadre visant à prendre en compte les usages, dans la mise en place d'agents informatiques pour le Web Sémantique. Notre approche est fondée sur la notion de conteneur de connaissance. Considérer les ressources comme des conteneurs de connaissance permet en effet de mettre l'accent sur le fait que certaines des connaissances ne sont accessibles qu'à l'utilisateur, et non à l'agent logiciel. Il est alors nécessaire pour ce dernier de s'appuyer sur un modèle d'utilisation, éventuellement expliqué par des modèles de tâche. De plus, les structures d'explication ainsi mises en place prennent alors elles aussi le statut de conteneurs de connaissances — car les méta-connaissances sont d'abord des connaissances. Deux implantations de cette approche ont été présentées, dans les domaines de la conception mécanique et de la recherche de documents audiovisuels.
Ces deux exemples, ainsi que la discussion qui les suit, montrent l'applicabilité du cadre proposé à des domaines très variés. Comme il l'a été dit en section 5, nous appliquons actuellement cette même approche à d'autres projets et étudions un cadre méthodologique général pour sa mise en œuvre.
Par ailleurs, il serait intéressant de travailler sur l'explicitation de modèles de tâches émergeant des cas d'utilisation récurrents. Ces derniers représentent en effet les usages effectifs d'un système. Une telle explicitation pourrait même permettre de concevoir *a posteriori* une ontologie de ces nouvelles tâches.

Références

[1] Agnar Aamodt and Enric Plaza. Case-based reasoning : Foundational issues, methodological variations, and system approaches. *AICom*, 7(1) :39–59, 1994.
⟨URL:http://www.iiia.csic.es/People/enric/AICom.html⟩.

[2] Hans Akkermans. Future of KM Technology in the Knowledge Economy. Invited talk at WM'2001, Baden-Baden, mar 2001.
⟨URL:http://wm2001.aifb.uni-karlsruhe.de/InvitedTalks/⟩.

[3] The Annotea project.
⟨URL:http://www.w3.org/2001/Annotea/⟩.

[4] Tim Berners-Lee and Mark Fischetti. *Weaving the Web*. Harper San Fransisco, 1999.

[5] Patrick Bougé, Françoise Détienne, and Laurent Di Cesare. Épisodes de conception : une étude ergonomique pour le recueil de « cas ». In *Ingénierie des Connaissances 2001*, Grenoble, jun 2001. .

[6] Pierre-Antoine Champin. A model to represent design episodes for reuse assistance with interactive case-based reasoning. In Ivo Vollrath, Sascha Schmitt, and Ulrich Reimer, editors, *GWC-BR'2001*, pages 189–197, Baden-Baden, Germany, mar 2001.

[7] Balakrishnan Chandrasekaran, John R. Josephson, and V. Richard Benjamins. Ontology of tasks and methods. *IEEE Intelligent Systems*, may/jun 1999.
⟨URL:http://www.cis.ohio-state.edu/~chandra/Ontology-of-Tasks-Methods.PDF⟩.

[8] The Darpa Agent Markup Language (DAML) project.
⟨URL:http://www.daml.org/⟩.

[9] Stefan Decker, Michael Erdmann, Dieter Fensel, and Rudi Studer. Ontobroker : Ontology based access to distributed and semi-structured information. In R. Meersman et al., editor, *Semantic Issues in Multimedia Systems. Proceedings of DS-8*, pages 351–369. Kluwer Academic Publisher, Boston, 1999.

[10] Steve DeRose, Eve Maler, David Orchard, and Ben Trafford. XML Linking Language (XLink) Version 1.0. W3C candidaTE recommendation, jul 2000.
⟨URL:http://www.w3.org/TR/2000/CR-xlink-20000703⟩.

[11] Elöd Egyed-Zsigmond. Manipulation des documents multimédias, idées d'application du RàPC pour une aide à l'utilisateur. In *Atelier Raisonnement à partir de cas*, pages 73–78, Plate-forme AFIA / Grenoble, jun 2001.

[12] D. Fensel, I. Horrocks, F. Van Harmelen, S. Decker, M. Erdmann, and M. Klein. OIL in a Nutshell. In R. Dieng et al., editor, *Knowledge Acquisition, Modeling, and Management, Proceedings of the European Knowledge Acquisition Conference (EKAW-2000)*. Lecture Notes in Artificial Intelligence, LNAI, Springer-Verlag, oct 2000.

[13] Dieter Fensel, Enrico Motta, V. Richard Benjamins, Stefan Decker, Mauro Gaspari, Rix Groenboom, William Grosso, Mark Musen, Enric Plaza, Guus Schreiber, Rudi Studer, and Bob Wielinga. The Unified Problem-solving Method Development Language UPML. Delivrable, University of Karlsruhe, Institute AIFB, 1999.

[14] Jeff Heflin, James Hendler, and Sean Luke. SHOE : A Knowledge Representation Language for Internet Applications. Technical report, Dept. of Computer Science, University of Maryland at College Park, 1999.
⟨URL:http://www.cs.umd.edu/projects/plus/SHOE/pubs/techrpt99.pdf⟩.

[15] Janet Kolodner. *Case Based Reasoning.* Morgan Kaufmann Publishers, 1993.

[16] Ora Lassila and Ralph R. Swick. Resource Description Framework (RDF) Model and Syntax Specification. W3C recommendation, feb 1999. ⟨URL:http://www.w3.org/TR/1999/REC-rdf-syntax-19990222⟩.

[17] Yannick Prié, Alain Mille, and Jean-Marie Pinon. Connaissances et documents audiovisuels : un modèle pour l'exploitation contextuelle des annotations. *Document numérique, numéro spécial "Gestion des documents et gestion des connaissances"*, 3(3–4) :241–262, dec 1999.

[18] Yannick Prié, Alain Mille, and Jean-Marie Pinon. Modèle d'utilisation et modèles de tâches pour l'assistance à l'utilisateur basée sur l'expérience : le cas d'un système d'information audiovisuelle. In *Ingénierie des Connaissances*, pages 21–30, Palaiseau, Jun 1999.

[19] Yannick Prié, Tahar Limane, and Alain Mille. Isomorphisme de sous-graphe pour la recherche d'information audiovisuelle contextuelle. In *12ème congrès Reconnaissance de Formes et Intelligence Artificielle, RFIA2000,*, volume 1, pages 277–286, Paris, feb 2000. ⟨URL:http://lisi.insa-lyon.fr/~yprie/download/rfia2000.pdf⟩.

[20] Unified Modeling Language (UML). ⟨URL:http://www.omg.org/technology/uml/⟩.