Spécification des interfaces 3D pour la visualisation des bases de connaissances

Falquet, Gilles; El Atifi, El Mustapha

Spécification des interfaces 3D pour la visualisation des bases de connaissances

EL Mustapha El Atifi et Gilles Falquet

CUI - Université de Genève.
24, rue Général Dufour
CH-1211 Genève 4. Suisse.
{elmustapha.elatifi, gilles.falquet}@cui.unige.ch

RÉSUMÉ. *Dans cet article nous présentons des modèles et langages pour la spécification d'interfaces 3D pour accéder aux bases de connaissances. L'approche que nous proposons est inspirée des notions d'hypertexte spatial et de document virtuel. La spécification se fait à un niveau abstrait et à un niveau concret. Le langage de spécification d'interface abstraite décrit les contenus de nœuds abstraits obtenus par sélections sur la base de connaissances, et différentes catégories de liens entre ces noeuds, il permet de générer une interface abstraite qui est un hypertexte spatial 3D abstrait. Le langage de spécification d'interface concrète associe des styles et des positionneurs aux composants abstraits de manière à produire une interface concrète où les nœuds possèdent une présentation et une position. Cette interface concrète est ensuite directement traduite dans un langage de représentation de scènes 3D (p.ex. VRML ou X3D) pour être affichée.*

ABSTRACT. *In this paper we present models and languages to specify 3D interfaces for accessing knowledge bases. Our approach is inspired by the notions of spatial hypertext and virtual document. In this approach, a specification has an abstract and a concrete level. The abstract specification language describes the contents of nodes, obtained by selecting knowledge base objects, and different categories of links on these nodes. It serves to generate an abstract interface which is a 3D spatial hypertext. The concrete specification language associates styles and layout managers to the abstract interface components, so as to produce a concrete interface in which the nodes have a presentation and a position. This concrete interface is then easily translated in a 3D scene representation language such as VRML or X3D to be displayed.*

MOTS-CLÉS : *base de connaissances, représentation des connaissances, visualisation, spécification d'interface, interface 3D, hypertexte spatial.*

KEYWORDS: *knowledge bases, knowledge representation, visualization, interface specification, 3D interfaces, spatial hypertext.*

1. Introduction

De nos jours, la connaissance constitue un élément important tant sur le plan opérationnel que sur le plan organisationnel d'un système d'information. Le partage de la connaissance dans le cadre d'un intérêt commun, sa valorisation et sa diffusion se traduisent souvent sous forme de la valeur ajoutée au sein des organisations, que ce soit dans l'accélération de l'analyse des besoins (Grosz et al., 2001), l'amélioration des services rendus aux clients, l'aide à la prise de décision ou encore dans l'amélioration de l'organisation du système d'information (Charlet et al., 2001). Si le développement des bases de connaissance résulte d'une prise de conscience dans les organisations, en l'absence d'outils informatiques efficaces qui aident à les explorer, les visualiser ou supporter leur évolution, il n'est pas facile de tirer le meilleur profit de des connaissances représentées.

Du point de vue informatique, plusieurs techniques et outils de visualisation de données et de l'information sont disponibles, notamment des techniques 3D, mais elles n'ont pas été largement utilisées pour visualiser des connaissances et elles ont été rarement combinées entre elles pour concevoir des interfaces complexes qui répondent à des besoins variés. L'un des obstacles majeurs à la création d'interfaces basées sur ces techniques est l'effort considérable de programmation requis. En outre, on ne peut en général pas associer directement une transcription visuelle aux connaissances abstraites. Il est donc indispensable de passer par un cycle de conception de l'interface (en particulier, trouver de bonnes métaphores), prototypage et test auprès des utilisateurs.

Nous nous intéressons donc à proposer des outils aux concepteurs pour les assister dans la création de ces interfaces, notre but étant de remplacer la programmation par une spécification suivie d'une génération automatique de l'interface. Notre approche consiste à décomposer la spécification d'une interface en (1) la spécification d'une interface abstraite, qui un hypertexte spatial abstrait, et (2) la spécification de la représentation concrète de chaque élément de l'interface abstraite pour produire un hypertexte spatial concret. A partir de ces deux spécifications on peut générer automatiquement l'interface tridimensionnelle dans un langage cible de modélisation 3D tel que X3D ou VRML.

Dans cet article, nous commencerons par présenter un état de l'art sur les techniques 3D de visualisation de l'information et des connaissances, sur la représentation des connaissances à l'aide des modèles hypertextes et sur les systèmes de spécification et génération d'interfaces, nous présentons ensuite le modèle de connaissance utilisé et les principes de représentation, puis les modèles et langages pour les spécifications abstraites et concrètes des interfaces 3D.

2. Etat de l'art

2.1. *Visualisation de l'information et des connaissances.*

Dans les systèmes de représentation graphique des données ou des informations, le recours à la 3D est évident pour visualiser un plus grand nombre d'informations. Une représentation tridimensionnelle offre de nombreux intérêts du point de vue de la visualisation: densité, focus (vue sur les détails) et contexte (vue d'ensemble), navigation, etc. En plus, elle offre un moyen très convivial pour l'utilisateur dont les mécanismes de perception visuelle sont parfaitement adaptés à la troisième dimension (Bruley, 1999). Dans la littérature, beaucoup de techniques de visualisation notamment en 3D ont été largement utilisées pour l'exploration rapide d'ensembles d'informations, pour la classification interactive ou pour la mise en évidence de relations et de structures dans ces informations. On trouve notamment les arbres coniques (Robertson et al., 1991), utilisés à des fins de visualisation d'information à structure arborescente, la géométrie hyperbolique (Lamping et al., 1995), dédiée à la déformation de l'espace disponible pour visualiser un grand nombre de nœuds, des scènes de réalité virtuelle, etc. Mais ces techniques ont été toujours utilisées dans un cadre précis ou un problème particulier.

2.1.1. *Quelques exemples de systèmes de visualisation 3D*

Data Mountain (Robertson et al, 1998) utilise une métaphore de montagne pour la gestion de documents, ces documents sont arrangés sur une surface inclinée et peuvent être réarrangés par l'utilisateur pour isoler des groupes de documents suivant un intérêt particulier. Parmi les autres techniques de visualisation des documents on cite Web Forager (Card et al., 1996), on trouve aussi celles développées avec une métaphore du paysage, où des documents sont placés dans un paysage virtuel en trois dimensions. Dans ce paysage, les documents similaires - au sens de la mesure de similarité - sont placés de telle sorte qu'ils soient voisins.

Le système SemNet (Fairchild et al. 1988) a été l'un des rares à proposer des vues tridimensionnelles des bases de connaissance. SemNet transforme une grande base de connaissance représentée sous forme d'un graphe orienté en une vue tridimensionnelle à base de rectangles étiquetés, connectés par des arcs colorés suivant la sémantique de la relation représentée. Dans ce système, l'exploration de la base de connaissance s'effectue par une navigation dans l'espace de représentation 3D. Pour le positionnement des éléments (nœuds), SemNet se base sur les propriétés de nœuds, leur connectivité avec les autres nœuds et les préférences de l'utilisateur qui peut choisir un placement pour un ou plusieurs noeuds.
D'autres systèmes de représentation tridimensionnelle comme MUE (Travers, 1989) se sont intéressé à la visualisation des structures dans les connaissances sous formes de réseaux de nœuds habituellement représentés par un graphe. MUT utilise une métaphore de musée virtuel pour visualiser la structure de la base de connaissances

CYC (CYC, 2005) où les noeuds sont représentés par des boxes imbriqués. Le système combine imbrication, couleurs et positionnement de ces boxes pour représenter les relations qui associent les nœuds du réseau.

2.1.2. *Interfaces des outils de conception et de visualisation des Ontologies*

Plusieurs publications dans l'ingénierie des connaissances, l'analyse du langage naturel, les systèmes d'information collaboratifs et de gestion de la connaissance ont souligné l'utilisation des ontologies dans le développement et l'utilisation de leurs systèmes (Klein et al., 2000). Depuis les années 90, plusieurs systèmes de conception des ontologies ont vu le jour. Des systèmes comme Ontosaurus (Swartout et al., 1996) sont basés sur la logique de description, quand à d'autres (Ontolingua (Farquhar et al., 1996), Protégé (Fridman et al., 2001), OntoEdit (OntoEdit, 2005), …) ils sont basés sur les frames. Plusieurs types de présentation sont utilisés à travers ces outils pour visualiser le contenu des ontologies, on trouve principalement les listes indentées, les frames HTML, les graphes, les arbres hyperboliques. Or ces techniques ont montré leurs limites quand il s'agit de grandes bases de concepts. La navigation dans les ontologies passe par des requêtes qui sont exprimées dans les langages de représentation sous jacents à ces systèmes ou en utilisant des formulaires. Pour les systèmes à interface hypertexte, naviguer consiste à suivre des liens explicites exprimés dans le langage de représentation des connaissances.

2.2. *Systèmes de spécification et de génération des interfaces*

Beaucoup de systèmes de conception d'interface proposent d'aider les designers dans leur tâche de conception. On repère dans la littérature les environnements de développement ou de management des interfaces utilisateur (UIDEs/UIMS). Ces environnements dits "Model-Based" se basent sur les descriptions fonctionnelles de l'application cible et adoptent une méthodologie pour supporter la génération automatique de l'interface utilisateur de cette application (Tanriverdi et al., 2001), (Pinheiro, 2000). Parmi les autres caractéristiques à souligner, des systèmes tels que MASTERMIND (Szekely et al, 1996), TADEUS (Elwert et al, 1995), ADEPT, TRIDENT (Bodard, 1994) s'appuient sur des spécifications déclaratives de leurs modèles (cf. paragraphe suivant) pour la génération de l'interface, ce qui signifie une facilité de conception en épargnant l'effort nécessaire à la programmation. Quelques uns de ces systèmes, tels que TADEUS ou TRIDENT, suivent une spécification à deux niveaux d'abstraction dans leur processus de génération d'interface.

En général, ces systèmes incluent un ensemble de modèles : un modèle de domaine, un modèle des tâches, un modèle de dialogue, un modèle de présentation et un modèle utilisateur. Les deux premiers modèles contiennent essentiellement les informations nécessaires à la génération d'interface pour une application

particulière. Parfois, ils intègrent des modules d'aide et d'assistance au design. Ex : détection des erreurs de design dans le système TRIDENT, ou l'utilisation d'une base de connaissance de règles de design contenant un ensemble d'éléments et de règles de représentation de l'interface. Le Modèle de Dialogue reflète la dynamicité de l'interface utilisateur, en d'autre termes les commandes qu'on peut effectuer à travers les éléments de l'interface. Quand au modèle utilisateur, il décrit la spécification de l'interface cible suivant chaque groupe d'utilisateurs et est surtout utile pour la construction des interfaces personnalisables.

L'utilisation des spécifications à un haut niveau d'abstraction pour générer l'interface n'est pas limité qu'au UIMs, elle est aussi adoptée dans la conception des environnements de réalité virtuelle où des systèmes tels que VRID (Tanriverdi et al., 2001) ont été développés essentiellement dans le but de fournir des interfaces plus intuitives et plus appropriées à des interactions complexes.

2.3. *Hypertextes, documents virtuels et hypertextes spatiaux*

Le modèle hypertexte est largement utilisé. Une interface hypertexte est traditionnellement composée d'une fenêtre par nœud et des zones réactives pour les liens qui renvoient à des adresses (adresses de documents, partie d'un document, etc.). Bien que ce modèle n'offre pas d'autres possibilités pour exprimer des liens souvent de plusieurs types, il est largement utilisé pour l'exploration de grands nombres de documents dans le web ou localement sur des machines. Des systèmes de représentation utilisent le modèle hypertexte pour décrire et organiser des connaissances. Ce modèle a l'avantage de faire cohabiter facilement l'aspect formel et non formel dans sa description des connaissances. MacWeb (Nanard, 1993) est un bon exemple de ces systèmes qui, en plus de sa capacité à représenter les connaissances formelles ou non formelles, permet d'identifier deux niveaux de ressources, un niveau document (information) et un niveau connaissance (réseau sémantique de concepts). La connexion entre ces deux niveaux se fait à base de liens (liens entre des parties de documents et des concepts).

Parmi les autres techniques de représentation de connaissances issues du modèle hypertexte, on distingue les documents virtuels où des fragments contenant de la connaissance textuelle peuvent être rassemblés pour constituer une interface sous forme de document réel. Selon la définition de Ranwez S. et Crampes M. (Ranwez et al., 1999), un document virtuel est une collection de briques d'informations appelés fragments associés par des techniques permettant de créer un document réel, le document réel dans ce cas n'est autre qu'une séquence de fragments assemblés selon un besoin de l'utilisateur. Ces techniques sont utilisées efficacement pour la publication du contenu des grandes bases de données. L'exemple concret de leur utilisation est le Web où tous les documents HTML ne sont pas préalablement stockés, un grand nombre d'entre eux est généré par des serveurs d'application. Les recherches dans cet axe s'intéressent à la problématique de construction de ces

documents (Crampes, 1997), (Ranwez et al., 1999) et aux techniques de leur assemblage et leur réutilisation dans la perspective d'ingénierie de documents.

Les hypertextes spatiaux sont des systèmes hypertextes dans lesquels les liens de navigation usuels sont complétés par des liens spatiaux implicites (Shipman et al., 1999). L'idée est d'exploiter la proximité spatiale entre des nœuds pour représenter un lien sémantique implicite qui existe entre ces nœuds. L'hypertexte spatial cherche à utiliser les capacités visuelles de l'être humain, en particulier ses capacités à comparer instantanément les distances entre des objets, à repérer des groupes, des alignements, des figures, etc. Cette notion est différente de l'hypertexte traditionnel, car elle fournit d'autres moyens pour exprimer les liens entre nœuds.

Plusieurs systèmes hypertextes, en particulier certains systèmes développés avant le Web, ont proposé l'utilisation d'un espace de travail pour gérer les nœuds d'un hypertexte. Dans MacWeb on peut visualiser une carte de nœuds de l'hypertexte et ses liens. Bien que la position des nœuds n'ait pas de sémantique particulière, le système incite à grouper spatialement les nœuds qui possèdent des liens. D'autres systèmes comme Web Squirrel ou Tinderbox de Eastgate possèdent une vue spatiale des nœuds. Dans le Web on rencontre fréquemment des cartes de sites qui exploitent la proximité spatiale pour indiquer la proximité sémantique des pages référencées. Dans ces systèmes les nœuds sont généralement représentés par des formes arrangés dans une surface 2D, mais des hypertextes spatiaux 3D commencent également à émerger (Gronbaek et al., 2000).

3. Principes et modèle de connaissances

Notre but est de spécifier et générer des hypertextes spatiaux tridimensionnels pour représenter le contenu d'une base de connaissance. Pour ce faire, nous avons adopté une approche de spécification à deux niveaux d'abstraction permettant de séparer la structure de l'interface (définition des objets de l'interface, les liens et relations sémantiques) de sa présentation en terme d'objets 3D et propriétés géométriques (couleurs, formes géométriques, positionnement, etc.). En effet, la distance entre les connaissances dans leur représentation d'origine (modèle de la base de connaissances) et leur représentation en 3D est importante, de ce fait, il est généralement difficile de passer directement d'un concept ou d'une relation abstraite à leur représentation concrète dans l'interface 3D, d'où la justification d'une telle approche. La spécification de l'interface 3D est donc constituée de la spécification d'une interface abstraite et la spécification d'une interface concrète.

Pour spécifier l'interface 3D de visualisation des bases de connaissance à son niveau abstrait, nous nous sommes basé sur les approches de spécifications déclaratives des vues hypertextes dans les bases de données (Falquet et al., 1998), (Mecca et al., 1998). La spécification permet de décrire comment les nœuds abstraits et leurs liens sont conçus à partir de sélections faites sur les objets de la

base de connaissances, cette base de connaissance est elle même exprimée dans un modèle utilisant la famille RDF-RDFS comme expliqué dans la sous section 3.2.

3.1. *Principes*

En faisant abstraction de la scène 3D à générer, la première étape dans notre approche consiste à concevoir une interface à partir de nœuds abstraits paramétrés. La spécification de cette interface abstraite permet de traduire le contenu de la base de connaissances vers des objets abstraits d'interface sous formes de nœuds et de liens entre ces nœuds. Le résultat d'une telle spécification permet de générer une interface abstraite qui est un hypertexte spatial abstrait. Une deuxième étape de la spécification est nécessaire pour décrire concrètement chaque nœud et chaque lien de l'hypertexte. Dans cette deuxième étape, le concepteur associe à l'ensemble des composants abstraits (nœuds et liens abstraits) des éléments de style et de placement qui serviront à produire un ensemble de composants concrets (représentation géométrique des noeuds, actions de navigations ou scripts. Comme 3ème étape, les composants concrets formant l'interface concrète vont être transformés en une interface 3D finale exprimée dans un langage de modélisation 3D.

Le schéma ci-dessous montre les trois étapes de la génération de l'interface 3D. Chacune des deux premières étapes s'appuie sur une spécification et un module générateur, on parle donc de deux niveaux de spécification. Le module Générateur IA est un module implémenté qui permet de générer une interface abstraite à partir de la base de connaissances (BdC), cette génération est contrôlée par la spécification abstraite. Le module Générateur IC permet de générer une interface concrète sous le contrôle d'une spécification concrète et à partir de l'interface abstraite. Le générateur de scène est un module permettant de générer la scène à visualiser (l'interface finale) dans l'un des langages de modélisation 3D.

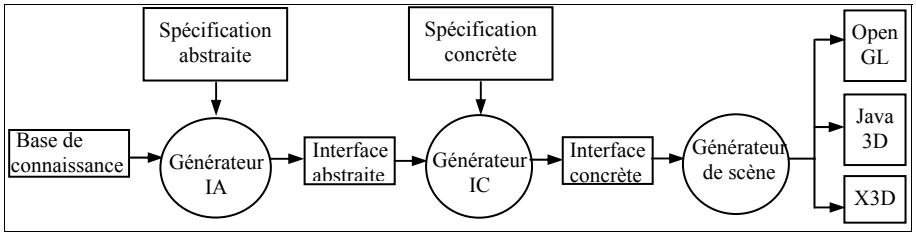

Figure 1. *Système de génération de l'interface 3D*

Dans les sections qui suivent nous allons décrire les différents modèles qui supportent notre approche et les langages utilisés pour la spécification dans ses deux niveaux. Nous détaillerons aussi l'utilité de chacun de ces modèles.

3.2. *Modèle de connaissances*

Les éléments de connaissance auxquels nous nous intéressons et qui sont présents dans la base de connaissance peuvent avoir plusieurs niveaux de formalisation et d'abstraction. Par exemple les ontologies de domaine contiennent de la connaissance formalisée abstraite. Les documents texte contiennent généralement des connaissances factuelles non formalisées, les bases de données contiennent quand à elles des connaissances factuelles formalisées. Pour prendre en charge toutes ces diversités, nous avons choisi le modèle RDF-RDFS comme modèle de représentation de connaissances. RDF est un modèle de donnée semi structuré et simple. Il est basé sur des triplets (sujet, prédicat, objet) formant un graphe sémantique. Dans ces triplets, l'objet et le sujet sont des ressources identifiées par leurs URIs. RDFS ajoute au modèle RDF des possibilités d'exprimer la sémantique des ressources en définissant des "Schéma RDF" de classes de ressources, de classes de propriétés et de leurs instances.

Dans le reste de ce document nous considérons que la base de connaissance est composée d'un ensemble de concepts et de documents (ressources) décrits en RDF. Les documents permettent de stocker des connaissances non formalisées et les pointeurs RDF décrivent les faits et les concepts.

Plusieurs langages de requêtes ont été proposé pour l'interrogation du modèle RDF (TRIPLE, Versa, RQL, SERQL, RDQL, etc.), le plus récent est le "working draft" SPARQL du W3C. Ces langages sont basés sur des triplets contenant des variables. Par exemple le triplet `( ?X author Bob)` correspond à tous les triplets qui ont author comme prédicat et Bob comme objet. La requête `SELECT ?X FROM (?X author Bob)` retourne toutes les ressources X dont l'auteur est Bob. Quelques langages ont des extensions pour interroger les bases RDFS, il contiennent des constructions permettant d'interroger le niveau « schéma » et traitent l'héritage entre classes et sous-classes.

4. Spécification d'interfaces abstraites

L'interface abstraite d'une base de connaissance est un hypertexte spatial abstrait. Cet hypertexte est abstrait dans la mesure où la géométrie n'intervient pas à ce niveau. Par contre, les nœuds et liens abstraits possèdent par des attributs qui, au niveau concret, donneront lieu à différentes représentations spatiales. Par exemple, la valeur d'un attribut de lien pourra déterminer un type de navigation dans l'hypertexte concret (déplacement instantané, « survol », etc.). De même, les liens sémantiques définis au niveau abstrait pourront se traduire au niveau concret sous forme de liens de navigation explicites ou sous forme de liens implicites représentés par le positionnement de nœuds.

Le niveau abstrait sert essentiellement à déterminer la composition des nœuds et leurs liens sémantiques. Un nœud hypertexte abstrait peut être simple ou composite,

dans ce dernier cas, le nœud inclut d'autres nœuds abstraits. Le lien d'inclusion entre le composant et le composite peut avoir des attributs spécifiant des éléments de placement des nœuds composant par rapport au nœud parent. Le lien d'inclusion ne signifie pas forcément qu'au niveau concret les nœuds seront emboîtés les uns dans les autres.

4.1. *Le modèle*

Un hypertexte spatial abstrait est un 4-uplet (N, I, V, S) ou N est un ensemble de nœuds, I est un ensemble de liens d'inclusion, V est un ensemble de liens de navigation et S un ensemble de liens sémantiques (liens implicites). Un nœud abstrait appartenant à N est un couple (i, c), où i est un identifiant et c le contenu du nœud. Les liens de navigation et les liens sémantiques sont des 4-uplets (s, d, t, v) où s est l' identifiant de la source, d l'identifiant de la cible, t le type du lien et v en ensemble de couples (attribut, valeur). Un lien d'inclusion est un 4-uplet (s, a, u, v) où s est l' identifiant du nœud composé, a l'élément dans lequel le nœud composant u va être inclus et v un ensemble de couples (attribut, valeur).

La spécification abstraite permet de décrire comment les nœuds abstraits et leurs liens sont composés à partir de sélections faites sur les objets de la base de connaissances. Dans cet article nous ne présentons pas toute la syntaxe du langage de spécification, mais nous allons exprimer un exemple à base de ce langage dans la section suivante. La spécification abstraite est un ensemble de spécifications de type de nœuds (classes de nœuds). Une classe de nœuds est un 5-uplet (n, p, s, c, l) où n est un nom, p est un ensemble de noms de paramètres, s est une expression de sélection, c est la spécification du contenu du nœud abstrait et l est la spécification de liens. Une spécification abstraite peut être vue comme un ensemble de requêtes paramétrées destinées à une interrogation de la base de connaissances et dont les résultats permettent de construire les nœuds de l'interface abstraite et les liens qui les associent à d'autres nœuds dans la même interface. L'expression de sélection est une expression d'interrogation de la base de connaissances exprimée dans un langage de requêtes. Dans notre cas nous utilisons un sous ensemble de SPARQL. Ses variables (variables de requête) représentent les ressources de la base de connaissances.

La spécification de contenu est un arbre de spécifications élémentaires. Une spécification élémentaire est un triplet (t, l, c) où t est le type de cet élément, l est une spécification de lien (éventuellement vide) et c une spécification de contenu. La spécification du contenu peut être : un littéral, un nom de paramètre, une variable de la BdC ou une liste de spécifications élémentaires. En outre, le contenu c peut contenir un sous arbre c_{iter} appelé contenu itéré. L'intérêt du contenu itéré est de pouvoir définir à l'intérieur d'un nœud une partie du contenu qui va être instancié pour chaque résultat de sélection dans la BdC. De ce fait les variables de la BdC ne peuvent apparaître que dans ce sous-arbre.

La spécification d'un lien est définie par la désignation du type du lien (inclusion, navigation ou sémantique), du nœud cible et d'un ensemble d'attributs du lien. Un exemple de spécification abstraite est montré dans la sous section suivante.

4.2. *Exemple :*

Il s'agit de représenter une base de connaissances d'un cours qui contient des concepts, des relations entre ces concepts et des liens typés entre ces concepts et des URIs (URIs vers des documents). Un document dans cet exemple désigne une description ou une définition de concept, son illustration, etc. L'exemple de spécification détaillé ci-dessous montre la génération d'un hypertexte spatial abstrait, représenté concrètement par une exposition virtuelle. Dans cette exposition, chaque concept est représenté par un stand avec divers affiches d'exemples, définitions et descriptions. Les exemples sont exposés dans la partie gauche du stand et les descriptions dans sa partie droite.

Les classes de nœuds abstraits sont définies à l'aide de la syntaxe concrète du langage de spécification de notre environnement. Une spécification de classe de nœud est de la forme :

```
abstract-node: <nom> [<parameters>]
selection: <expression SPARQL>
contents: <specification du contenu et des liens>
```

La partie du contenu itérée sur chaque résultat de la sélection est placée entre { et }.

```
abstract-node: ConceptPresentation [c]
selection: (c ?r ?c2)(?c2 rdf:type rdf:Class)
content:
{semantic link: to: ConceptPresentation[c2]
                type: "semanticRel"
}
inclusion link: to: ExamplesOf[c]
                attributes: (position: "left")
inclusion link: to: DescriptionsOf[c]
                attributes: (position: "right")

abstract-node: Exhibition
selection: (?c rdf:type rdf:Class)
contents: { include ConceptPresentation[c] }

abstract-node: ExamplesOf [c]
selection:  (c ex:example ?d)
content: { include TextPanel[d] }

abstract-node: TextPanel [d]
content: <theText>(d)
```

5. Spécification d'interfaces concrètes

L'interface concrète est un hypertexte spatial tridimensionnel où les nœuds provenant de l'interface abstraite sont représentés par des nœuds 3D concrets, les liens dans l'interface abstraite sont transformés en des liens spatiaux ou en contraintes géométriques portant sur le positionnement ou le rendu des nœuds qui constituent la portée du lien lui-même. Les nœuds concrets font appel à des positionneurs qui gèrent le placement de leurs composants (nœuds qui y sont inclus) et à des éléments de style (formes géométriques, couleurs, etc.) qui constituent le rendu visuel de ces nœuds.

5.1. *Le modèle*

Nous proposons de noter l'hypertexte spatial concret par un 6-uplet (N, I, V, S, M, B) ou N est un ensemble de nœuds concrets, I est un ensemble de liens d'inclusion, V est un ensemble de liens de navigation, S un ensemble de liens sémantiques (liens implicites), M un ensemble de positionneurs et B un ensemble de liens entre nœuds concrets et positionneurs. Un nœud concret est un triplet (i, c, p) avec i comme identificateur, c comme contenu et p un ensemble de couples (attribut, valeur) de description du style de présentation concrète tel que la couleur, la visibilité, etc. Une liaison nœud – positionneur est un ensemble de couples (p, v) où p est un paramètre du positionneur et v sa valeur associée. La valeur d'un paramètre est soit de type simple (nombre, chaîne de caractères, etc.) soit le nom d'un type de lien sémantique (dans les cas où le positionneur a besoin d'une relation entre les nœuds à positionner pour calculer leurs positions relatives).

Même si nous l'avons appelé concret, ce modèle est plus abstrait que les modèles 3D habituels comme VRML, Java3D ou OpenGl, cela est justifié par le fait que dans notre modèle, les coordonnées de positionnement ou les dimensions d'objets 3D ne sont pas déclarés explicitement, ils sont gérés directement par les positionneurs.

La spécification concrète décrit le passage d'une interface abstraite formée de nœuds et de liens abstraits vers une interface concrète formée d'objets 3D et d'actions dans un hypertexte spatial. Elle est similaire au principe des feuilles de style dans la présentation de documents mais elle est plus sophistiquée en termes de placement d'objets dans l'espace hypertexte, car elle dédie ce placement à des positionneurs. Plus précisément, chaque nœud concret a son propre positionneur qui positionne ses sous nouds dans un espace réservé à ce même nœud. Un positionneur peut avoir deux types de paramètres, (1) des paramètres globaux : par exemple la distance minimale entre deux objets, le nombre de lignes et de colonnes dans un positionneur de type grille, (2) des contraintes géométriques de placement : associées à des liens implicites ou à des relations sémantiques dans un nœud. Une fois associée à un nœud, la spécification du positionneur décrit son

comportement par l'association de valeurs à ses paramètres. Par exemple, les éléments "`<top> include A </top> <bottom> include B, include C </bottom>`" provenant d'une interface abstraite, doivent être utilisés par le positionneur selon la spécification concrète suivante:

"`Layout:CubeWidhFaces(topface:./top,bottomface: ./bottom)`".

Pour les liens de navigation, les éléments de l'interface abstraite peuvent déterminer des attributs de positionnement de leurs ancres (éléments actifs du nœud: généralement un sous nœud mais peut être le nœud en sa totalité).

Une spécification concrète est un triplet (n, a, l) où n est le type de nœuds abstraits auxquels est associée cette spécification, a est une liste de couples (attribut, valeur) et l est la spécification du positionneur. Les attributs dont il est question ici sont des attributs de style, par exemple : `(shape : Box, position : (5, 3, 2.5))`. La valeur d'un attribut est soit une constante ou le résultat de calcul d'une expression qui fait intervenir des valeurs trouvées dans l'interface abstraite et qui proviennent à leur tour de la BdC comme par exemple pour l'attribut size : `(size : ./size)`.
La spécification du positionneur est composée de son nom (un identificateur unique) et d'une liste de correspondances entre ses paramètres et leurs valeurs en plus des contraintes géométriques et des relations sémantiques qui leurs correspondent.

5.2. *Exemple :*

Si on se situe dans l'exemple cité dans la section précédente, la spécification concrète de la scène 3D qu'on veut générer est constituée des spécifications de nœuds concrets suivants :

```
concrete-node: Exhibition
shape: Box;
layout-manager: 2DSpringDistances(spring => "semanticRel")

concrete-node: ConceptPresentation [c]
shape: Box
layout-manager: BoxBorder(location => position("left"-
>"north", "rigth"->"south")

concrete-node: ExamplesOf[c]
shape: Wall; layout-manager: Sequence

concretenode: TextPanel [d]
shape: Text: content: ./theText; layout-manager: HTMLViewer
```

La scène générée (cf. figure 2), permet de naviguer entre les différents stands représentant les concepts et d'examiner les fragments liés à ces concepts.

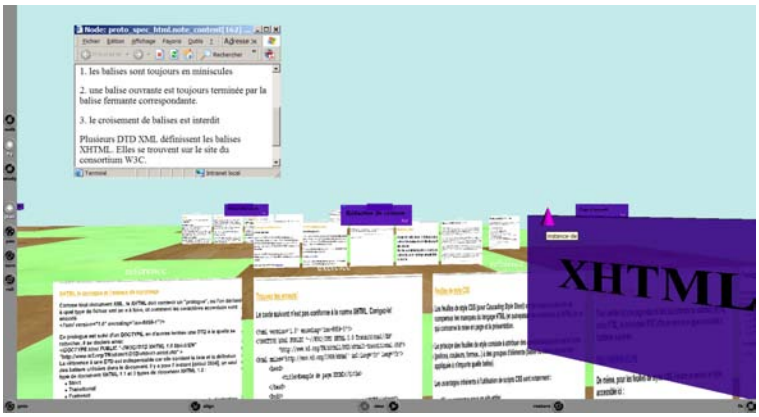

Figure 2. *Vue tridimensionnelle d'une base de connaissance sous forme d'exposition virtuelle.*

La figure ci-dessous montre le résultat d'une autre spécification d'interface sur la même base de connaissance. Ici on a simplement représenté les concepts et leurs liens sémantiques sous forme d'un graphe en 3D.

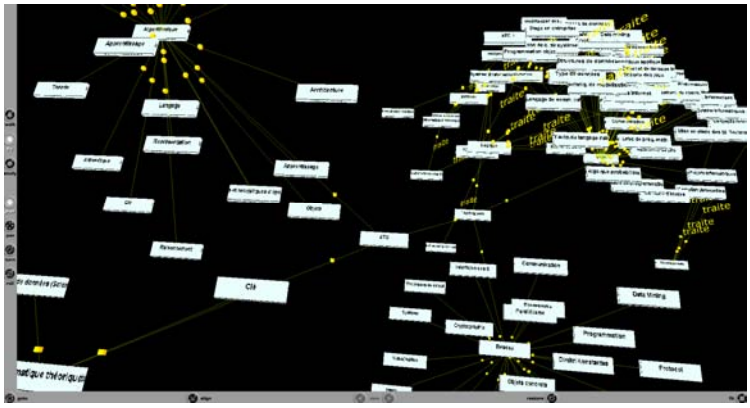

Figure 3. *Vue tridimensionnelle des concepts d'une base de connaissance sous forme de graphe en 3D.*

6. Implémentation

Pour tester notre modèle, nous avons développé des générateurs pour les deux interfaces: abstraite et concrète. Le générateur de l'interface abstraite appelé

"Générateur IA" a comme entrée la spécification abstraite et la base de connaissances, il produit une interface abstraite comme montrée dans la figure 1. La génération commence par le nœud racine et descend récursivement vers les nœuds associés. Dans le système actuel, afin de réutiliser les outils que nous avons déjà développés, la base de connaissance est représentée de manière directe dans une base de données relationnelle. Les spécifications des nœuds abstraits sont traduites en des nœuds Lazy (Falquet et al., 1998). Le système Lazy est ensuite invoqué pour générer l'interface abstraite sous forme de documents XML.

Le générateur de l'interface concrète "Générateur IC" charge l'interface abstraite à l'aide d'un parseur XML, transforme les nœuds abstraits en un arbre X3D partiel et invoque les positionneurs pour affecter un placement aux différents groupes et objets de l'arbre X3D. Les positionneurs sont des classes java qui implémentent une interface Layout. Le module appelé "Générateur de Scène" permet de générer les objets 3D directement en X3D, nous travaillons actuellement sur un format d'échange qui permettra d'implémenter la génération dans un langage de 3D cible choisi préalablement par le concepteur comme montré dans la figure1. Un module de transformation X3D vers VRML a été intégré au système pour générer la scène directement en VRML. Actuellement, les moteurs de rendu tels que Cortona ou Cosmo-player ne supportent pas directement le format X3D.

7. Conclusion et perspectives

Dans cet article, nous avons montré comment on peut générer des scènes 3D complexes de visualisation du contenu d'une base de connaissance à l'aide des spécifications écrites dans un langage simple. Le processus de spécification passe par deux étapes : une étape de spécification abstraite qui produit une interface hypertextuelle abstraite et une étape de spécification concrète permettant de générer l'interface 3D constituée d'objets et d'actions. Le langage de spécification est un langage simple et déclaratif et permet des spécifications sans aucune connaissance des langages de modélisation 3D.

Notre approche est actuellement statique: la scène générée est figée à cause des positions fixes des objets 3D présents dans la scène. Actuellement aucun module ne permet de recalculer les positions suite à une action de l'utilisateur sur la scène, c'est justement l'une des perspectives de nos recherches. Nous allons travailler pour générer des représentations 3D dynamiques où les nœuds sont réactifs et capables de s'auto réorganiser dans l'espace selon l'interaction de l'utilisateur. A titre d'exemple, un positionneur de type arbre hyperbolique en 3D, doit recalculer les positions des nœuds qui gère à chaque fois que l'utilisateur tire un nœud au centre. Nous devons aussi offrir une possibilité de manipulation interactive des paramètres de la visualisation des connaissances.

8. Bibliographie

Bodard F., Hennebert A-M., Leheureux J-M., Provot I., Vanderdonckt J., A model-based Approach to Presentation: A Continuum from Task Analysis to Prototype, *Proc. of the Eurographics Workshop on Design, Specification and Verification of Interactive Systems*, p. 25–39., 1994.

Bruley Christophe, Analyse des représentations graphiques de l'information : extension aux représentations tridimensionnelles, Thèse de doctorat, Université Joseph Fourier, Grenoble -FR, 3 juin 1999.

Card S., Robertson G., York W., The WebBook and the Web Forager: An Information Workspace for the World-Wide Web, *Proc. of the CHI Ô96, ACM Conference on Human Factors in Software*, 1996.

Charlet J, Reynaud C. et Teulier R., *Ingénierie des connaissances pour les systèmes d'information, Ingénierie des systèmes d'information*, éditions Hermès, p. 283, 2001.

Crampes M., Auto-Adaptative Illustration through Conceptual Evocation, *Proc. of the ACM Digital Library*, Philadelphia. PA., USA, July 1997, ACM Press, p. 247-253, 1997.

CYC Ontology, site web : http://www.cyc.com/, visité le 10 février 2005.

Elwert T., Schlungbaum E., Modelling and Generation of Graphical User Interfaces in the TADEUS Approach. *Designing, Specification and Verification of Interactive Systems*, Vienna, Springer Editions, p. 193-208, 1995.

Fairchild Kim M., Poltrock S.E., Furnas G.W., SemNet: Three-Dimensional Graphic Representations of Large Knowledge Bases, R. Guindon (editor), *Cognitive Science and Its Applications for Human-Computer Interaction*, Hillsdale, NJ: Lawrence Erlbaum, 1988.

Falquet G., Guyot J., Nerima L., Languages and Tools to Specify Hypertext Views on Databases, *Proc. of The World Wide Web and Databases, International Workshop (webDB'98)*, Valencia, Espagne, Mars 1998, LNCS 1590, 1998.

Farquhar A., Fikes R., Rice J., The Ontolingua Server: a Tool for Collaborative Ontology Construction, Knowledge Systems Laboratory, Stanford University, 1996.

Fridman Noy N., Grosso W., Musen M. A., Knowledge-Acquisition Interfaces for Domain Experts: An Empirical Evaluation of Protégé-2000, SMI Technical report SMI-2000-0825, Retrieved See "http://smi-web.stanford.edu/pubs/SMI_Abstracts/SMI-2000-0825.html" last visit April 5 - 2005.

Gronbaek K., Mogensen P., Hypermedia in the virtual project room - toward open 3D spatial hypermedia. *Proc. of the 11th ACM conf. on Hypertext and hypermedia*, San Antonio, Texas, USA, 2000.

Grosz G. et Rolland C., *De la modélisation conceptuelle à l'ingénierie des besoins, Ingénierie des systèmes d'information*, Editions Hermès, collection informatique et systèmes d'information, p. 101, 2001.

Kayser D., *La représentation des connaissances*, Ed. Hermès, collection informatique, 1997.

Klein M., Fensel D., van Harmelen F., Horrocks I., The relation between ontologies and schema-languages: Translating OIL-specifications in XML-schema, Benjamins, V. R., Gomez-Perez, A., and Guarino, N., editors, *Proc. of the Workshop on Applications of Ontologies and Problem-solving Methods*, *14th European Conference on Artificial Intelligence (ECAI 2000)*, Berlin, Germany, 2000.

Lamping J., Rao R., Pirolli P., The Hyperbolic Browser: A Focus+Context Technique for Visualizing Large Hierarchies, *Proc. of the ACM CHI'95*, New York, 1995.

Lauridsen Ole, Abstract Specification of User Interfaces, *Proc. of the ACM CHI'95 – short papers*, New York, 1995.

Mecca G., Atzeni P., Masci A., Merialdo P., Sindoni G., From Databases to Web-Bases: The ARANEUS Experience, Report Technique n. 34-1998, Dipartimento di Informatica e Automazione, Università di Roma, 1998.

Nanard J., Nanard M., Should Anchors be Typed Too? An Experiment with MacWeb, *Proc. of the HT'93*, p. 51-62, 1993.

OntoEdit, See " http://www.ontoprise.de/products/ontoedit_en". Last visit February 10 2005.

Pinheiro Paulo da Silva, User Interface Declarative Models and Development Environments: A Survey, *In Lecture Notes in Computer Science*, volume 1946, p. 207-226, 2000.

Ranwez S., Crampes M., Conceptual Documents and Hypertext Documents are two Different Forms of Virtual Document, *Proc. of the Workshop on Virtual Documents,*1999.

Robertson George G., Mackinlay Jock D., Card Stuart K., Cone Trees: animated 3D visualizations of hierarchical information, *Proc. of the SIGCHI : Human factors in computing systems: Reaching through technology*, New Orleans, Louisiana, USA, p. 189 - 194, 1991.

Robertson George, Czerwinski M., Larson K., Robbins D. C., Thiel D., Maarten van Dantzich, Data Mountain: Using Spatial Memory for Document Management. *Proc. of the UIST'98*, San Francisco, CA, 1998.

Shipman F., Marshall C., Spatial hypertext: an alternative to navigational and semantic links, *in the ACM Computing Surveys*, Vol. 31, No. 4, December 1999.

Swartout B., Patil R., Knight K., Russ T., Toward Distributed Use of Large-Scale Ontologies, *Proc. of the 10th Workshop on Knowledge Acquisition*, Banff, Canada, 1996.

Szekely P., Sukaviriya P., Castells P., Muthukumarasamy J., Salcher E., Declarative Interface Models for User Interface Construction Tools: The Mastermind Approach., Bass L. and Unger C. (editors), *Engineering for Human-Computer Interaction*, p. 120-150. New York, Editions Chapman et Hall, 1996.

Tanriverdi Vildan, Robert J.K Jacob, VRID: A Design Model and Methodology for Developing Virtual Reality Interfaces, *Proc. of the ACM-VRST'01*, 2001.

Travers M., A visual representation for knowledge structures, *Proc. of the 2nd ACM conf. on Hypertext*, p. 147-158, November 1989.

Winfried H. Graf, Constraint-Based Graphical Layout of Multimodal Presentations, *Reading in Intelligent User Interfaces,* p. 263-285, 1998.