CONSERVATOIRE NATIONAL DES ARTS ET MÉTIERS
CENTRE RÉGIONAL ASSOCIÉ DE GRENOBLE

MÉMOIRE
présenté en vue d'obtenir
LE DIPLÔME D'INGÉNIEUR C.N.A.M.
en
Informatique
par
Christophe ALEMANY

Étude et réalisation d'une interface d'édition de bases de connaissances au travers du World Wide Web

Soutenu le 25 Mars 1998

Président :	Mme	Véronique DONZEAU-GOUGE
Membres :	M.	Christian CARREZ
	M.	Jacques COURTIN
	M.	Jérôme EUZENAT
	M.	Alain LANDELLE
	M.	François RECHENMANN

CONSERVATOIRE NATIONAL DES ARTS ET MÉTIERS
CENTRE RÉGIONAL ASSOCIÉ DE GRENOBLE

MÉMOIRE
présenté en vue d'obtenir
LE DIPLÔME D'INGÉNIEUR C.N.A.M.
en
Informatique
par
Christophe ALEMANY

Étude et réalisation d'une interface d'édition de bases de connaissances au travers du World Wide Web

Les travaux relatifs au présent mémoire ont été effectués au sein du projet SHERPA de l'I.N.R.I.A. Rhône-Alpes du 1 Novembre 1996 au 31 Octobre 1997, sous la direction de Jérôme Euzenat.

Remerciements

Je remercie...

Tout d'abord les membres du jury,

Véronique Donzeau-Gouge, Professeur au Conservatoire National des Arts et Métiers de Paris, qui me fait l'honneur de présider ce jury,

Christian Carrez, Professeur au Conservatoire National des Arts et Métiers de Paris, pour avoir accepté d'être membre du jury,

Jacques Courtin, Professeur à l'Université Pierre Mendès-France de Grenoble et responsable du cycle d'ingénieur CNAM en Informatique pour le centre agréé de Grenoble, pour m'avoir suivi durant ma formation,

Alain Landelle, Maître de conférence à l'Institut National Polytechnique de Grenoble, pour m'avoir également suivi durant ma formation,

Jérôme Euzenat, Chargé de recherche à l'I.N.R.I.A qui a dirigé ce mémoire et qui, par ses conseils, ses remarques, ses relectures, a permis l'aboutissement de ce travail,

François Rechenmann, Directeur de recherche à l'I.N.R.I.A. et responsable du projet SHERPA, pour m'avoir accueilli dans son équipe,

Les membres du projet SHERPA,

Isabelle Crampé, Loïc Tricand de la Goutte, Petcko Valtchev, qui par leurs travaux en relation avec mon travail, ont été mis à contribution tout au long de ce stage,

Jérôme Gensel, Thibaut Parmentier, Christophe Arlot pour m'avoir soutenu et attendu pendant les longues séances de footing hebdomadaires,

Les membres du projet SHERPA, pour leur aide, leurs conseils: Danielle Zibelin, Michel Page, Gilles Bisson et Sylvie, Cécile, Suelli, Jean-Marc, Patrice,

Les locataires du bureau B212, José Kahan Oblatt et Laurent Tardif pour leurs conseils et leur soutien quotidien.

Ma famille,

Particulièrement ma grand-mère et Nathalie pour leur soutient moral, leurs encouragements durant toutes ces années,

Océane, plus récemment, pour avoir été sage la plupart des nuits et m'avoir donné une motivation supplémentaire pour la réussite de ce mémoire.

Tables des matières

Président : 1
Membres :..... 1
Remerciements 4
I. Introduction 1
II. Présentation 3
A. Système à base de connaissances.....3
B. Tropes.....6
C. Le World Wide Web 21
III. L'existant..... 27
A. Hytropes.....27
B. Ontolingua.....37
IV. Études..... 45
A. L'interface 45
B. Le contrôle d'accès 52
C. La concurrence d'accès aux données.....58
D. La gestion des erreurs.....61
E. Conclusion.....64
V. Réalisation..... 65
A. Environnement informatique 65
B. Définition des URLs 65
C. Définition de l'interface 68
D. Le contrôle d'accès 81
E. La gestion des erreurs 83
F. Conclusion.....86
VI. Bilan et perspectives..... 89
A. Bilan.....89
B. Perspectives 90
C. Apports personnels.....91
Glossaire 92
Bibliographie..... 94
Annexe 1: Les URLs d'édition..... 96
Annexe 2: Le serveur «Apache» 98
Annexe 3: Le lexique 102
A. La structure du lexique.....102
B. L'API du lexique 102
C. L'interface de consultation/modification 103
Annexe 4: Les écrans de Hytropes 107

I. Introduction

Accéder à des bases de connaissances à l'aide de moyens hypermédias, conviviaux et accessibles par tous est une idée émergeante dans le domaine de la recherche en informatique. Plusieurs raisons permettent en effet de justifier cette idée:

- Tout d'abord, il n'est pas pensable de considérer l'information contenue dans une base de connaissances isolément des sources de connaissances à partir desquelles elle a été tirée (textes, références bibliographiques, expériences...). Un lien entre les deux est donc nécessaire pour peu que l'on veuille détailler, annoter ou simplement documenter une connaissance. Les concepts hypermédias permettent parfaitement de répondre à cette attente.

- D'autre part, le service Word Wide Web[1] d'Internet est un bon moyen de publier une base de connaissances. Il est en effet, accessible de part le monde à l'aide d'un simple ordinateur (même peu puissant) et d'une connexion au réseau.

- Pour terminer en faveur des moyens hypermédia comme accès aux bases de connaissances, il faut souligner que la possibilité de consulter une base de connaissances de la même manière qu'une encyclopédie est considérée comme un intérêt majeur pour de nombreux chercheurs. La prolifération actuelle des encyclopédies multimédias sur cd-rom montre bien l'intérêt de tels supports tant du point de vue pédagogique que pratique. L'utilisation de systèmes à bases de connaissances à l'aide du Web permettrait aux chercheurs d'accéder à la connaissance avec la même convivialité que pour ces encyclopédies. Cela représente une aide précieuse pour la résolution de certains de leurs problèmes.

Le succès d'Internet grandissant au sein de la population des chercheurs, il est intéressant d'appliquer les concepts de consultation hypermédias aux bases de connaissances Tropes[2]. De cette constatation est né le projet de rendre ces bases accessibles par le biais du Web. La base «Knife[3]» concernant les interactions entre gènes chez la drosophile (sorte de petite mouche) définie conjointement par le LGPD[4] de Marseille et le projet Sherpa de l'INRIA[5] Rhône-Alpes confirme l'intérêt d'un tel système. Cependant, la simple consultation de bases de connaissances n'est qu'une étape du projet. Il est en effet frustrant de ne pas pouvoir disposer d'outils permettant d'enrichir la connaissance avec la même ergonomie que pour la consultation. Le sujet de ce mémoire est donc la conception et l'implémentation d'une interface de modification de bases de connaissances Tropes au travers du Web en utilisant au mieux l'outil de consultation déjà existant.

---

[1] ou Web ou WWW ou W3: Service Internet permettant l'édition de documents sur le réseau
[2] Système de gestion de bases de connaissances développé à l'INRIA Rhônes-Alpes
[3]Knowledge on the Networks of Interactions in the Fly Embryo
[4]Laboratoire de Génétique et Physiologie du Développement
[5]Institut National de Recherche en Informatique et en Automatique

Cinq parties permettent d'appréhender ce sujet:

- La première partie définit les différentes notions nécessaires à la compréhension du sujet. Je présenterai notamment ce qu'il faut comprendre par le terme de système à base de connaissances et plus particulièrement le système de représentation de connaissance Tropes qui est le cœur du système sur lequel l'implémentation est effectuée. J'introduirai également la définition de ce qu'est «le Web», quelle est sa structure et quels sont les avantages apportés par un tel environnement pour notre application.

- Suivant cette présentation, je décrirai dans une seconde partie, Hytropes, l'outil actuel de visualisation de bases Tropes sur le Web. Plus précisément, il sera question de détailler son architecture, ses fonctionnalités et son interface. Je présenterai également ici le principal système concurrent existant qu'est «Ontolingua», ce dernier inclut des possibilités d'édition ce qui le rend particulièrement intéressant à étudier.

- Au sein d'une troisième partie, j'aborderai les études nécessaires avant de se lancer dans la réalisation d'un tel projet. Je discuterai notamment des différentes possibilités d'implémentation de l'interface d'édition, des différentes contraintes liées à l'utilisation du Web comme support d'application. Enfin, je détaillerai l'ensemble des points critiques à vérifier dans une application permettant l'édition de bases de connaissances par réseau (contrôle d'accès, concurrence d'accès, gestion des erreurs), ainsi que les différents moyens de les résoudre.

- La partie suivante s'attache à présenter l'interface de modification de bases de connaissances réalisée. J'évoquerai également le contexte matériel et logiciel utilisé pour effectuer ce travail.

- Pour terminer ce mémoire, je ferai un bilan du travail réalisé pendant cette année. Il sera également question de présenter les différentes perspectives qu'un tel projet peut susciter, ainsi que les évolutions prévisibles au regard des technologies futures.

II. Présentation

Cette première partie présente les différentes notions nécessaires à la compréhension du sujet. Dans un premier temps il est question de définir la notion de base de connaissances et en particulier de mettre en valeur les différences entre «base de connaissances» et «base de données» (ces deux termes étant souvent assimilés à tort). Le second point présente plus particulièrement Tropes qui est le cœur du système sur lequel l'implémentation est effectuée. Pour terminer, j'introduirai une présentation de ce qu'est «le Web», quelle est sa structure et quels avantages peuvent être apportés par l'utilisation d'un tel environnement dans le cadre de notre application.

A. Système à base de connaissances

Avant de détailler la structure de Tropes, il s'agit de définir ce que l'on entend par système à base de connaissances. Une brève définition du terme, suivie d'une comparaison avec celui de «système à base de données» permettra de le faire.

1. Définition

Un système à base de connaissances est un outil permettant d'organiser et stocker la connaissance. Il fournit également des services permettant d'utiliser au mieux cette connaissance et notamment il a la propriété de pouvoir faire des déductions à partir des connaissances déjà présentes dans la base. En résumé, un système à base de connaissances comprend en fait une structure de données (la base de connaissance) à laquelle s'ajoute des mécanismes d'inférence et de classification permettant de faire certaines déductions. Bien entendu, plus les connaissances seront précises, plus elles auront de signification pour l'utilisateur.

2. Bases de données et bases de connaissances

Plusieurs points séparent les bases de connaissances des bases de données, il s'agit ici de les mettre en évidence.

a. Mécanismes d'inférence

Les mécanismes d'inférence inhérents aux systèmes à base de connaissances sont les premiers éléments permettant la différenciation des deux systèmes. La structure d'une base de données ne permet pas, sans programme externe, de déduire les valeurs de certains champs à partir des données présentes dans la base, par contre celle des systèmes à base de connaissances l'autorise. C'est d'ailleurs bien l'utilité d'une connaissance.

b. Classification

Alors que dans les bases de données, on ne peut faire que du «filtrage», c'est-à-dire rechercher un ensemble d'objets qui obéissent aux conditions définies dans une requête, les systèmes de représentation de connaissance permettent non seulement le filtrage, mais ils permettent également de déduire dans quelle catégorie (ou classe) un objet peut être rangé (ou plutôt classé) au vue des caractéristiques qu'on lui a attribuées.

c. Sémantique

Un autre point important, lié aux deux remarques précédentes, sépare également les bases de données des bases de connaissances: il s'agit de la sémantique. Alors que les bases de données permettent uniquement de ressortir des données brutes, sans sémantique autre que le nom d'un champ et sa valeur, les systèmes utilisant des bases de connaissances, de part leur structure, donnent un sens à ces données, ces dernières deviennent véritablement des connaissances.

Prenons l'exemple d'une fiche médicale, cas typique des bases de données. Ce qui permet d'affirmer qu'un patient a de la température ou non n'est pas explicitement écrit sur la fiche, c'est au médecin d'interpréter le sens de la valeur du champ «température». Dans une base de connaissances, les champs dont on connaît la signification sont accompagnés des règles de déduction associées, c'est comme si la fiche médicale était accompagnée de notes telles que «température supérieure à 38° signifie que le patient est fiévreux»...

d. Complétion de la connaissance

Pour terminer, on peut noter que dans les bases de données, la structure d'une fiche de données impose de connaître la totalité des données avant d'être mise dans la base. Une connaissance, par contre, peut être incluse partiellement. Elle sera éventuellement, du moins c'est le but, complétée avec le temps. Bien entendu, plus la connaissance sera détaillée, plus il sera aisé d'en déduire quelques propriétés.

Dans le domaine de la connaissance, lorsqu'une information n'est pas connue, elle est dite indéterminée. Dans celui des bases de données, une valeur par défaut compatible avec le type de la donnée est imposée par le système. Il est alors difficile de savoir si une valeur qui vaut celle par défaut a été consciemment saisie par l'utilisateur ou si l'utilisateur ne connaissant pas cette donnée a laissé agir le système. C'est alors au programmeur de gérer tout cela en définissant des règles de gestion propres à chaque information.

3. Systèmes de représentation de connaissances par objet

Dans les systèmes de représentation de connaissances par objets, les éléments de description de la connaissance sont représentés par des objets. La terminologie «par objet» est similaire à celle employée dans les langages de programmation par objet en ce

sens que la connaissance est représentée comme étant un objet du système, une instance que l'on manipule au travers des classes qui définissent cet objet. Deux objets principaux permettent donc de définir la connaissance dans un tel système, la classe et l'instance [Gensel 92] [Marino 90].

a. Les classes

Deux approches différentes et complémentaires permettent de définir ce qu'est une classe:

- Les classes sont des objets générateurs d'instances et permettent le partitionnement du domaine modélisé en diverses catégories, sous-catégories. Elles recueillent l'ensemble des caractéristiques, ou attributs, permettant de définir ces catégories et sont réparties sous forme de hiérarchie de classes représentant des relations de spécialisation. Cela signifie qu'une sous-classe d'une ou plusieurs autres classes est plus spécifique, elle reprend donc toutes les caractéristiques de(s) classe(s) supérieures(s) en y ajoutant des restrictions et peut même y ajouter ses propres attributs.

- Une autre vision définit les classes comme étant des ensembles d'instances. Dans ce cas, ce n'est plus la classe qui est génératrice d'instances, mais le contraire. Le procédé permettant d'obtenir une hiérarchie de classe à partir d'instance est appelé système de classification ou «clustering».

b. Les instances

Les instances représentent les éléments présents dans les catégories définies par les classes. Elles ont la même structure que la classe à laquelle elles appartiennent, c'est-à-dire qu'elles ont les mêmes attributs et elles contiennent en plus les valeurs de ces attributs.

4. Conclusion

Pour être plus concret au sujet des bases de connaissances, je prendrai simplement l'exemple d'un médecin qui analyse la fiche médicale d'un patient. Dans ce cas la base de données est la fiche médicale du malade avec son passé médical, les médicaments prescrits, les contre-indications... Pour interpréter ces données, le médecin a besoin de plus d'informations, il a besoin de ce qu'il a appris en faculté de médecine, le savoir faire qu'il a acquis tout au long de ses années de pratique... Il a besoin des connaissances qu'il a acquis tout au long de sa vie médicale. Les systèmes à base de connaissances sont là pour lui permettre de ne pas oublier, mais aussi et surtout de faire profiter les autres médecins de ces connaissances.

B. Tropes

Les termes de base de connaissances et système de représentation de connaissances par objets ayant été introduits, ce chapitre se consacre plus particulièrement à Tropes, système de représentation de connaissance par objet. Il s'agit en effet du cœur du système sur lequel l'interface d'édition est développée. Je ne m'attacherai pas à présenter la syntaxe permettant de définir les différents éléments composant une base de connaissances Tropes car elle n'apparaît pas dans l'interface. Par contre, il sera question de traiter les grands principes du logiciel. Pour se faire une présentation de l'historique du projet sera tout d'abord effectuée, suivi de la définition de la structure des bases Tropes et des mécanismes d'exploitation intégrés au système. Pour terminer cette présentation de Tropes, un aperçu des possibilité de l'API[6] sera évoqué afin d'appréhender de quelle manière il est possible d'agir sur une base de connaissances Tropes.

1. Historique du projet

Tropes est un système de représentation de connaissances par objets expérimental conçu et développé au sein du projet Sherpa de l'I.N.R.I.A. Rhône-Alpes depuis 1990. Il s'agit du successeur du logiciel Shirka développé également dans l'équipe Sherpa entre 1983 et 1989.

Aucun domaine d'application précis n'est visé par la structure de Tropes. Une base de connaissances concernant les interactions entre gènes chez la drosophile a été mise en place avec la collaboration de Bernard Jacq du laboratoire de génétique et physiologie du développement de Marseille [Euzenat & 97a]. L'apport d'une telle structure sur les quelques centaines de gènes déjà saisis dans la base semble avoir attiré l'attention des biologistes. Il reste à compléter cette base pour pouvoir vraiment juger de ce que peut apporter un tel système dans ce domaine. Un projet de mise en place d'une base de connaissances concernant la structure des réseaux de l'entreprise est actuellement en cours de développement avec la société SGS-Thomson de Crolles. La réussite d'un tel projet confirmerait l'intérêt des bases de connaissances dans deux domaines d'application bien distincts, biologie et informatique, ce qui laisse présager de l'intérêt de ces systèmes dans de nombreuses autres applications.

En tant que système de représentation à base de connaissances par objets, Tropes reprend l'essentiel des caractéristiques qui permettent de définir de tels systèmes, c'est-à-dire mécanismes d'inférences et classification. Il y ajoute cependant quelques particularités remarquables. Deux nouvelles entités descriptives (les concepts et les points de vue) ainsi qu'un mécanisme d'aide à la classification (les passerelles) imposent une nouvelle structuration de la connaissance.

[6] Aplication Programming Interface: fonctions permettant d'agir sur un environement

2. Structure des bases de connaissances Tropes

Quatre niveaux permettent de structurer une base de connaissances, le concept, le point de vue, les classes et les passerelles:

a. Le concept

Le premier niveau permettant d'organiser la connaissance est le concept. Traditionnellement une base de connaissances à objets ne comporte qu'une seule hiérarchie de classe pour spécifier le domaine de connaissance. Dans Tropes, une base de connaissances est constituée de familles d'entités distinctes. Chacune de ces familles correspond à un concept, ils rassemblent l'aspect ontologique des objets et les différentes manières de les observer [Euzenat 93b].

Prenons une modélisation, très simplifiée, de ce que pourrait être une gestion d'appartements suivant la structure des bases de connaissances Tropes. Ce sujet est relativement vaste et il convient dans un premier temps de cerner les différents aspects qui nous intéressent.

Ce qui nous intéresse dans une gestion d'appartement :

- connaître les caractéristiques des appartements gérés.
- connaître les informations concernant la clientèle.

Ces centres d'intérêts vont être représentés dans Tropes par des concepts, nous retiendrons donc en premier lieu les concepts «appartement» et «humains» (clients de l'agence).

Composition du concept

La structure des objets ou attributs de concept (conceptslots)

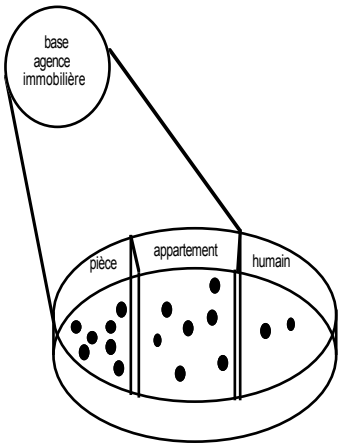
Les attributs de concept permettent de modéliser le domaine de connaissance traité. Ils sont eux-mêmes caractérisés par un nom (chaîne de caractère permettant d'identifier l'attribut dans le concept), un type (élémentaire ou un autre concept), un constructeur (permettant de dire si une seule valeur ou plusieurs sont nécessaires pour caractériser l'attribut) et pour finir la nature de l'attribut (propriété ou composant).

Un appartement a une adresse, il se situe à un certain étage, il a un certain numéro, il occupe une certaine surface, il a un ou plusieurs occupants, il a un certain nombre de pièces... Toutes ces propriétés de l'appartement sont représentées dans Tropes par des attributs qui caractérisent le concept «appartement»:

- Une adresse (nom de l'attribut) est une chaîne de caractère (type de l'attribut), un appartement n'a qu'une adresse (constructeur de l'attribut) et une adresse est une propriété d'un appartement (nature de l'attribut).

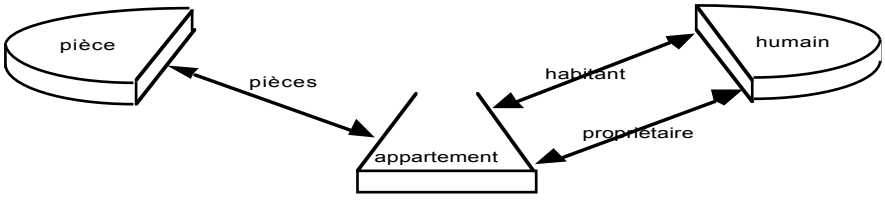
- L'étage est un nombre entier, un appartement n'occupe qu'un étage et l'étage est une propriété d'un appartement.

- La description d'une pièce d'un appartement nécessite plusieurs attributs. Il peut donc être pertinent de créer un concept «pièce» pour décrire cette entité.



Vision multi-concepts de la base «agence immobilière»

La relation entre les concepts «pièce» et «appartement» s'effectue alors par un attribut du concept «appartement» («pièces»). Ce dernier est défini comme étant un ensemble d'instances du concept «pièce», sa nature est de type «composant» car l'ensemble des différentes pièces forme justement l'appartement. De même la relation entre le concept «appartement» et «humain» s'effectue par l'intermédiaire des attributs «habitant» et «propriétaire».



Relation entre les concepts grâce à certains attributs

L'identité de chaque objet ou clé du concept

La clé d'un concept correspond à un sous-ensemble des attributs permettant d'identifier chaque objet de façon unique. Pour qu'un objet puisse exister, il faut obligatoirement renseigner les attributs clés de celui-ci, la clé est en effet le seul moyen d'identifier les objets dans Tropes.

Ce qui permet par exemple d'identifier de façon unique un appartement, c'est son **adresse**, l'**étage** sur lequel il se situe et un **numéro** qui différencie deux appartements se trouvant sur un même palier.

Les instances

Le concept contient également l'ensemble des instances saisies pour celui-ci. Chaque instance contient l'ensemble des attributs qui définissent le concept ainsi que tout ou partie des valeurs de ces attributs (au minimum les valeurs des attributs clés du concept sans lesquels l'objet ne peut exister.

L'appartement «81 rue Jean Jaures ...» étage «1» numéro «21b» est une instance du concept appartement.

Remarque :

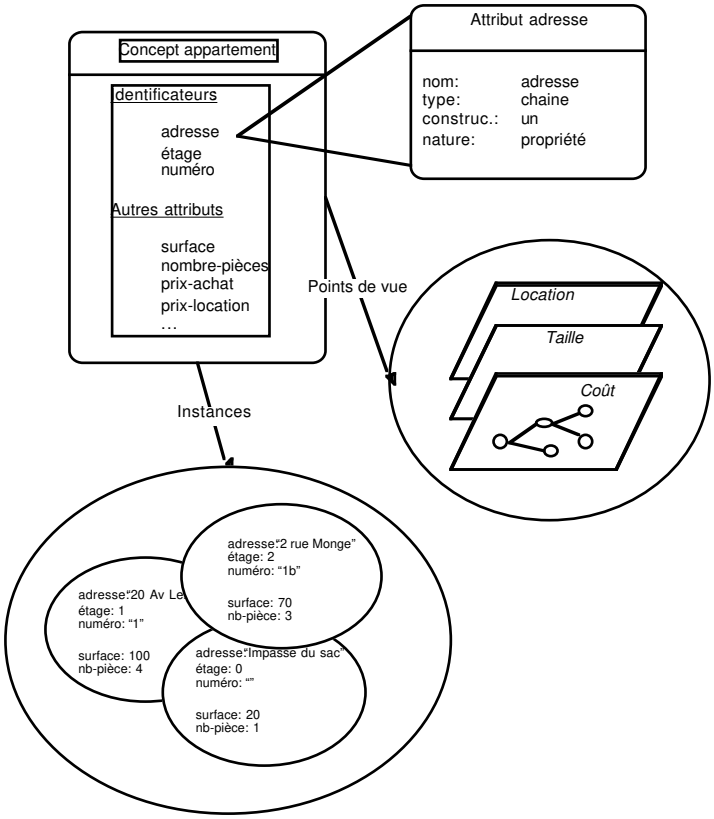
Alors que dans la définition traditionnelle des bases de connaissances à objets les instances sont générées par les classes, dans Tropes elles le sont au niveau du concept.

Les différents points de vue

Le point de vue correspond au deuxième niveau de structure de la représentation de connaissances. C'est à partir du point de vue que l'on commence à organiser la connaissance, sa description est faite dans le point suivant.

Récapitulatif de la structure d'un concept

Pour conclure à propos de la notion de concept, on peut dire que les objets sont ici présentés à l'état brut, comme dans une base de données. On se contente de détailler tout ce qui peut caractériser et identifier l'objet étudié. C'est au niveau des points de vue que l'on commence à définir et organiser la connaissance.



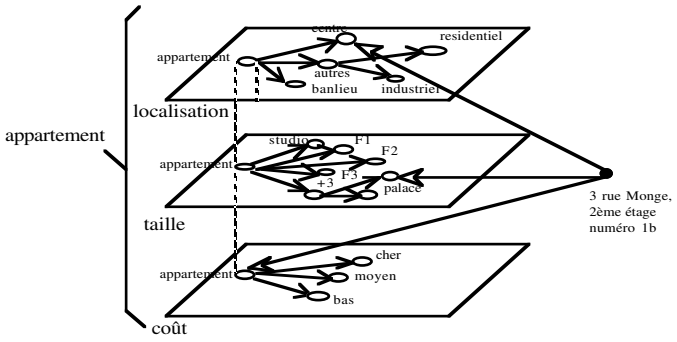
Structure du concept

b. Le point de vue

Le point de vue est la deuxième et principale innovation de Tropes. Traditionnellement la base de connaissances était composée d'une seule hiérarchie de classes, avec la notion de concept ce n'est plus le cas et c'est au niveau du point de vue qu'on retrouve cette hiérarchie. Les points de vue ont été introduits dans Tropes afin que des spécialistes de domaines différents puissent observer une base de connaissances en ne voyant que les attributs qui concernent leur domaine d'application ou bien les attributs communs à ces différents domaines mais auxquels ils donnent un sens différent (exemple de la cylindrée d'une voiture qui a un certain sens pour le trésor public et un autre pour le mécanicien) [Marino 90] . Un même concept peut donc être observé suivant plusieurs points de vue et chaque point de vue possède sa propre taxonomie de classe (donc plusieurs taxonomies de classe par concept).

Exemple:

Différents points de vue peuvent être dégagés du concept d'appartement, on peut par exemple s'intéresser à sa localisation dépendant notamment de l'adresse de l'appartement, sa taille suivant la superficie et le nombre de pièces et pour finir son coût d'après le prix d'achat, la superficie et le nombre de pièces de l'appartement.

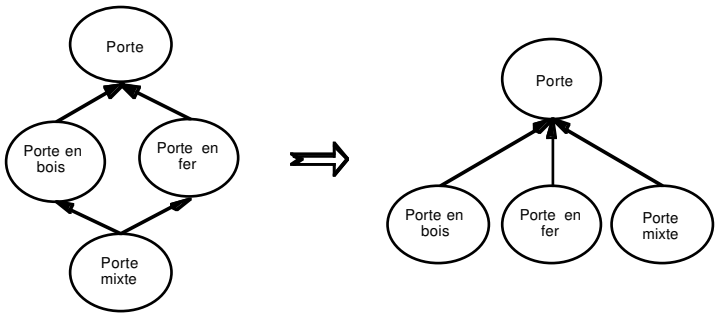


Les différents points de vue du concept «appartement»

c. Les classes

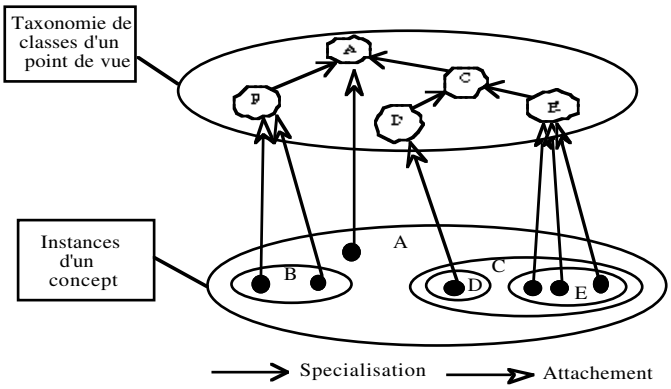
Les classes permettent de regrouper, ou classer, un ensemble d'objets entre eux suivant les caractéristiques communes de certains attributs pertinents. Elles sont organisées de façon arborescente dans le point de vue, de sorte que la classe racine (la plus haute) corresponde à la classe la plus générale et plus une classe s'en éloigne plus elle est spécifique.

- Tropes ne gère pas l'héritage multiple, c'est-à-dire qu'à une classe donnée ne correspond qu'une et une seule classe plus générale à l'exception de la classe racine qui n'a pas de classe supérieure puisque c'est la plus générale de toutes.



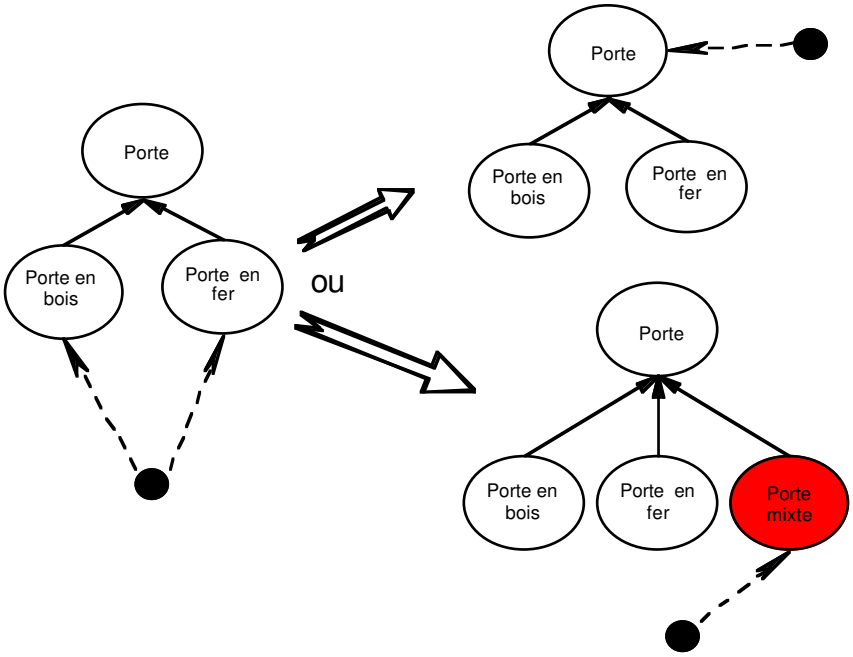
Héritage multiple Héritage simple

- Dans Tropes, les instances sont générées par les concepts et non par les classes. Une conséquence directe de ce fait est qu'un objet pourrait n'appartenir à aucune classe d'un point de vue. Afin d'éviter cette possibilité et faire en sorte que toute instance puisse être visible dans tous les points de vues, les classes racines de chaque point de vue ne comportent pas de contraintes d'exclusion. Elles portent le nom du concept auquel elles appartiennent, ce qui les dégage de toute contrainte sémantique particulière. Une instance d'un concept, nouvellement créée est donc lié à la classe racine de chaque point de vue. Elle peut être attachée à telle ou telle autre classe plus spécifique si les valeurs de ses attributs le permettent.

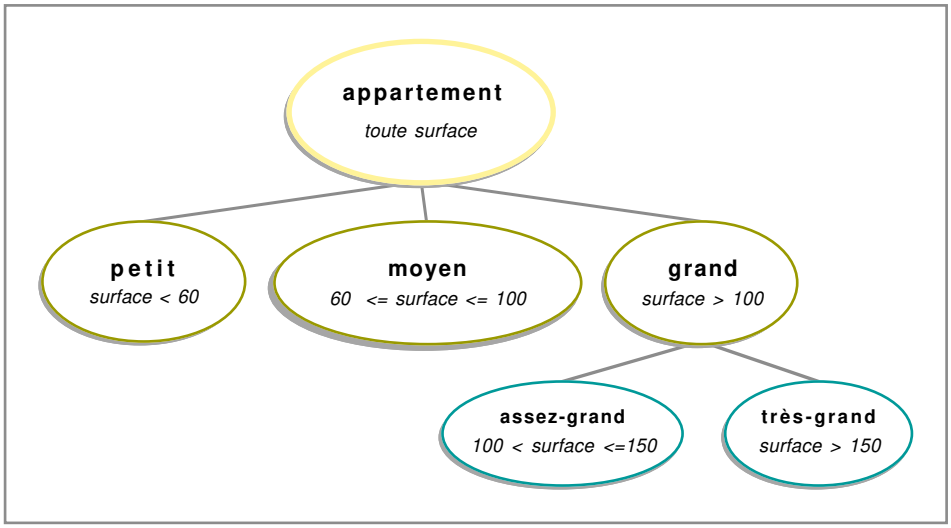


Les classes
dans la structure des bases Tropes

- On pourra remarquer en observant le schéma précédent qu'une instance n'est liée qu'à une et une seule classe (racine ou non). Les classes représentent en effet la décomposition du domaine modélisé en diverses catégories. Le désir de lier une instance à plusieurs classes d'une branche différente de l'arborescence (donc plusieurs catégories différentes) mettrait en évidence un manque dans la modélisation du domaine. Il serait alors judicieux de définir une nouvelle catégorie regroupant les spécialisations de chacune des classes ou de lier l'instance dans une catégorie supérieure (plus générale).



En gardant l'exemple de la gestion d'appartements et en particulier l'attribut surface, supposons que nous disposions d'un appartement de 170 m2 et de l'arborescence de classe suivante:



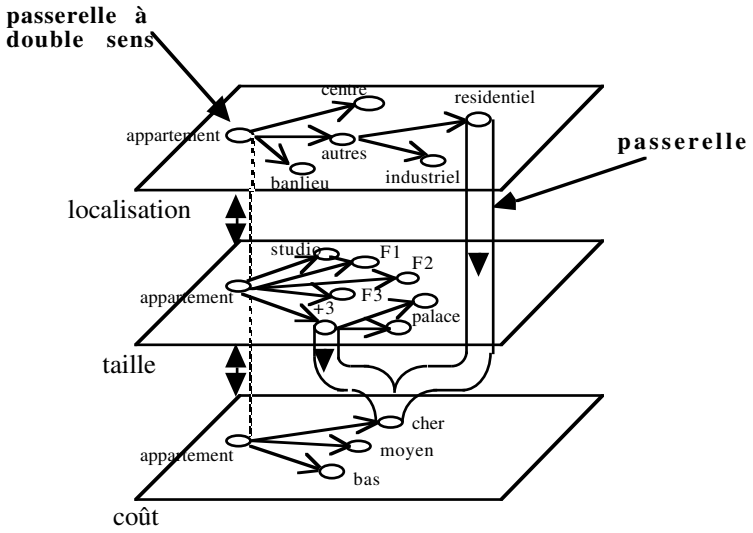
exemple de spécialisation de classes

Nous pouvons décider que l'appartement est grand puisque sa surface dépasse 100 m2, on pourra même par la suite dire qu'il est très grand puisque elle est également supérieur à 150 m2. Il est par contre impossible de lier notre appartement à la classe «petit» puisque sa surface ne correspond pas aux critères requis par cette classe.

d. les passerelles

Parallèlement aux niveaux de structure, une autre notion a été implémentée dans Tropes, toujours dans le but de mieux représenter la connaissance; ce sont les passerelles. Elles permettent de lier plusieurs classes de points de vue différents de sorte que si un objet fait partie de telle(s) classe(s), il fera forcément partie de telle autre. Les passerelles sont définies au niveau du concept puisqu'elles concernent différents points de vue d'un même concept.

Par exemple le «prix» d'un appartement est forcément «cher» si du point de vue de la «taille» il a «plus de trois pièces» et que du point de vue de sa «localisation» il est dans un «quartier résidentiel».



Exemple de passerelles

- la passerelle à double sens signifie qu'une instance est visible dans tous les points de vue au moins par la classe racine de chacun d'entre eux.

- un appartement de plus de 3 pièces situé dans un secteur résidentiel coûte forcément cher (passerelle).

e. Récapitulatif de la structure des bases Tropes

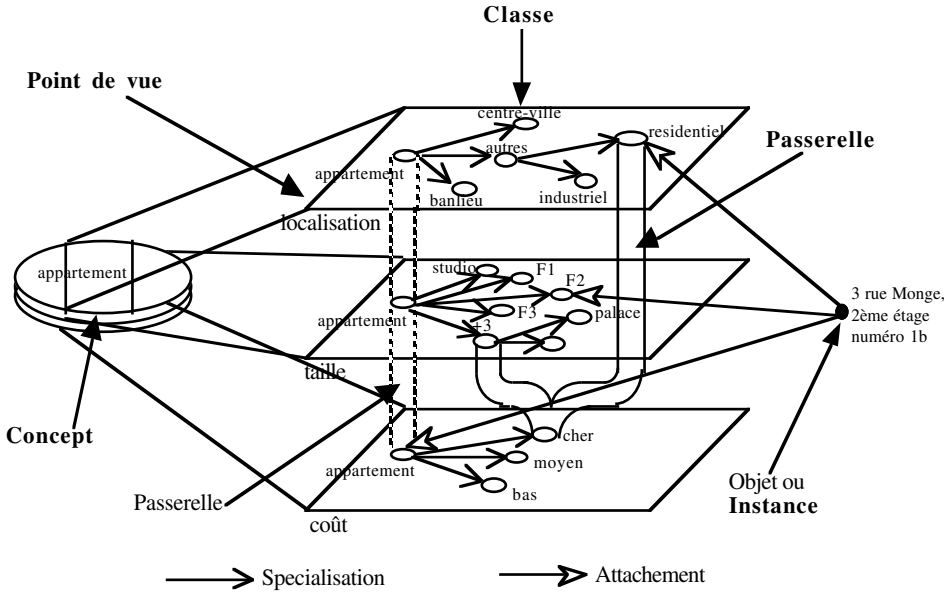


Schéma récapitulatif de la structure des bases de connaissances Tropes

3. Les différents mécanismes d'exploitation de Tropes

a. La classification

La classification est le mécanisme de raisonnement principal de Tropes. Il consiste à déterminer dans chaque point de vue l'ensemble des classes dans lesquelles une instance pourrait être rattachée et ce d'après ses propres valeurs d'attributs et les spécialisations propres à chaque classe.

Exemple:

> *Comment se situe l'appartement «xyz» du point de vue de sa taille, sa localisation, son prix ?*
>
> Les restrictions positionnées dans chacune des classes des différents points de vue évoqués permettent de répondre à cette question. Si l'appartement «xyz» a plus de trois pièces par exemple, il pourrait être classé dans une classe nommée «plus-de-trois» du point de vue de sa taille, si son prix de location est supérieur à six mille Francs on pourrait dire qu'il est cher du point de vue coût...

b. Le filtrage

Le filtrage est le mécanisme le plus proche de celui des requêtes dans les bases de données. Il permet en effet de rechercher dans un concept l'ensemble des instances faisant parties de telles ou telles classes et dont les attributs peuvent avoir telles ou telles valeurs (ou plutôt domaine de valeurs). Dans Tropes, les filtres sont considérés comme des classes. Le filtrage revient donc à créer une classe d'un point de vue imaginaire et à rechercher les instances du concept qui peuvent lui être rattachées (voir définition de la classification ci-dessus). Cette classe regroupe l'ensemble des restrictions présentes dans chacune des classes choisies dans le filtre et celles des domaines de valeurs autorisées saisis pour chaque attribut du concept, toujours dans ce même filtre.

Il s'agit en fait du mécanisme inverse de la classification. Au lieu de rechercher l'ensemble des classes dont les restrictions acceptent les valeurs des attributs d'une instance, on souhaite au contraire obtenir l'ensemble des instances dont les valeurs d'attributs sont compatibles avec les restrictions définies dans le filtre.

Exemple:
Quels sont les appartements de taille moyenne situés au premier ou au second étage d'un immeuble ?

La traduction de cette requête d'après la structure de données de Tropes est la suivante: recherche des instances du concept «appartement» faisant partie de la classe «moyen» du point de vue «taille» et dont l'attribut «étage» vaut un ou deux. Le filtrage revient donc à créer une classe, sous-classe de la classe «moyen», avec en plus une restriction concernant le domaine de valeurs autorisé de l'attribut «étage». Ne seront

finalement retenues que les instances dont la valeur de l'attribut «étage» vaut un ou deux et dont les valeurs de l'ensemble de ses attributs sont bien en phase avec les restrictions de la classe «moyen».

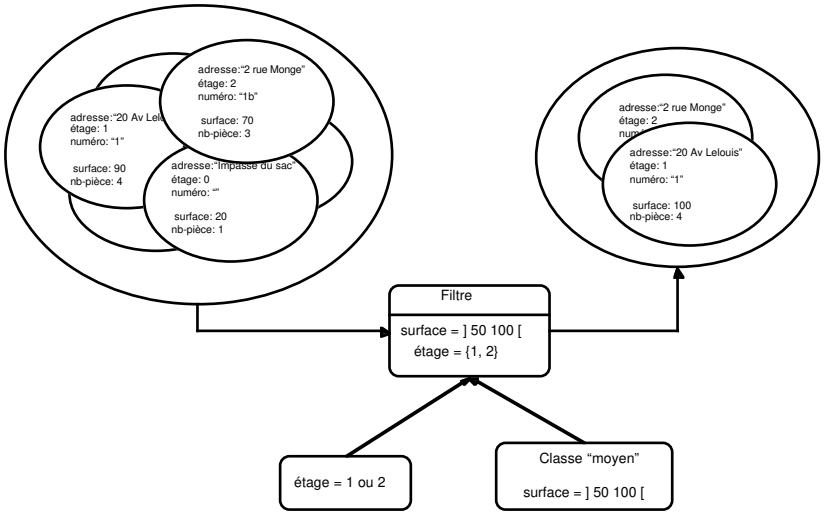


Illustration du filtrage

c. L'inférence de valeurs

L'inférence de valeurs, autre mécanisme propre aux bases de connaissances vient tout juste d'être implémentée à Tropes. Il s'agit de pouvoir proposer à l'utilisateur une ou plusieurs valeurs d'attributs en fonction d'autres valeurs déjà renseignées par ailleurs. Lorsqu'un utilisateur est hésitant sur une valeur d'attribut, il peut alors se résoudre à choisir l'une des valeurs proposées par les mécanismes d'inférence.

- Le premier mécanisme consiste à inférer la valeur d'un attribut «x» à partir de la valeur d'un autre attribut «y». De cette manière, le système sera en mesure de proposer une valeur pour l'attribut «x» si celle de «y» est renseignée.

- Le second mécanisme met en œuvre le filtrage. Lorsqu'un attribut est de type objet, c'est-à-dire que sa valeur est un ensemble d'instances d'un concept, les valeurs acceptables pour la classe concernée sont le résultat du filtre défini sur ce concept.

  Exemple:

  Pour connaître les habitants d'un appartement, le filtrage trouvera les personnes pour lesquelles l'attribut habitation est l'appartement donné et les proposera comme réponse à la question.

- Le troisième mécanisme consiste à définir une fonction utilisant des valeurs d'autres attributs. Après exécution de la fonction, le résultat sera proposé à l'utilisateur comme valeur acceptable.

Exemple:

> La surface d'un appartement correspond à la somme des surfaces de ses différentes pièces.

- Le dernier mécanisme d'inférence de Tropes est tout simplement la valeur par défaut. L'utilisateur définit une valeur par défaut pour un attribut et lorsqu'une instance sera générée sans valeur pour cet attribut, la valeur par défaut sera proposée comme une alternative.

Les mécanismes d'inférence présentés ci-dessus sont définis au niveau de la classe. Cela signifie que les mécanismes d'inférence à mettre en œuvre pour calculer une valeur correspondent à ceux définis pour la classe d'attachement de l'objet ou à ses super-classes (les mécanismes d'inférence sont en effet hérités de la même façon que les contraintes de classes). Autre point à noter pour finir: les mécanismes d'inférence ont été présentés dans leur ordre de priorité, c'est-à-dire que si une valeur d'un attribut «X» et une fonction «Y» ont été définies et que les deux mécanismes trouvent une valeur alternative, c'est celle de l'attribut «X» qui sera proposée.

4. L'API de Tropes

L'API (Application Programming Interface) de Tropes est un ensemble de fonctions permettant à l'utilisateur d'interroger et de manipuler des bases de connaissances Tropes via le langage Ilog Talk ou le langage C. Cette partie présentera quelques fonctions caractéristiques de l'API permettant la modification de bases de connaissances Tropes. Dans une autre section, sera détaillée l'interface de modification et les services de l'API mis en œuvre pour pouvoir manipuler les bases de connaissances Tropes au travers du Web.

L'ensemble des opérations qui agissent sur l'entité base de connaissances Tropes permet une bonne approche de ce qui peut être effectué à l'aide de l'API. Cette partie détaillera donc précisément toutes les opérations agissant sur l'objet base. Un tableau récapitulatif permettra ensuite de voir ce qui peut être fait sur les autres entités de Tropes.

a. L'entité base

Les informations concernant les bases de connaissances Tropes sont stockées dans deux fichiers, le premier contient la description de la structure de la base et le second la description des instances. Afin de pouvoir exploiter les connaissances de ces deux fichiers, il est nécessaire de charger et organiser leur contenu dans la mémoire du système. La fonction «tr-load-base» de l'API sert à cela.

(tr-load-base
#f»nom-du-fichier-de-decription-de-la-structure-de-la-base»)
-> charge dans le système une base à l'aide de son fichier de description

Lorsque l'on souhaite définir une nouvelle base de connaissances on ne dispose d'aucun fichier, il s'agit alors de créer la base dans le système, puis de la sauvegarder sous forme de fichiers. «tr-create-base» permet de créer une base vide (sans concepts) dans le système. Le premier paramètre obligatoire donne un nom à la base. Ce nom est nécessaire puisqu'il permettra par la suite d'identifier la base chaque fois que l'on voudra y faire référence (fonction «tr-find-base» de l'API). Le paramètre suivant, optionnel, indique sous quel nom de fichier on souhaite la sauvegarder. «tr-save-base» permet quant à elle de sauvegarder une base sous la forme de fichiers.

(tr-create-base «nom-base» optional:#f»nom-du-fichier-de-structure»)
-> créé une base de nom «nom-base» si elle n'existe pas déjà, le paramètre optionnel permet d'indiquer le nom du fichier qui servira à la sauvegarde.

(tr-find-base «nom-base»)
-> permet de faire référence à la base de nom «nom-base» si elle existe dans le système

(tr-save-base base)
-> sauvegarde la base «base» sous forme de fichiers

Il est également possible de supprimer une base de la mémoire du système (ne supprime pas les fichiers de description), «tr-delete-base» permet de supprimer une base que l'on vient de créer (sans concept) et «tr-erase-base» permet de supprimer une base contenant déjà une certaine structure (concepts, point de vue, instances...)

(tr-delete-base base)
-> supprime une base vide du système

(tr-erase-base base)
-> supprime une base vide ou non vide du système

Dans la partie concernant la structure des bases de connaissances Tropes on a pu voir qu'une base était constituée de concepts. L'API permet de rechercher l'ensemble des concepts présents dans une base via la fonction «tr-base-concepts», il est bien entendu possible par la suite d'ajouter de nouveaux concepts à la base à l'aide des fonctions «tr-create-concept» et «tr-add-concept» ou bien d'en supprimer «tr-rem-concept».

(tr-base-concepts base)
-> retourne l'ensemble des concepts définis dans la base «base»

(tr-create-concept «nom-concept»)
-> crée le concept de nom «nom-concept» s'il n'existe pas déjà

(tr-add-concept base concept)
-> relie le concept «concept» à la base «base» si cela n'a pas déjà été effectué

(tr-rem-concept base concept)
-> sépare le concept «concept» de la base «base» si cela n'a pas déjà été effectué

b. Tableau récapitulatif des fonctions de l'API

L'ensemble des fonctions de l'API de Tropes ne sera pas détaillé ici, elle se trouve dans le manuel de référence de Tropes [Tropes]. Par contre il convient de préciser que l'ensemble des fonctions de l'API est regroupé par entité suivant la structure des bases de connaissances Tropes.

Le tableau suivant permet de faire une synthèse de ce qui peut être fait à l'aide de l'API pour chaque entité du système.

Entité Tropes	**Action**
Base	-créer, accéder, supprimer -sauvegarder une base -charger un base à partir de ses fichiers descriptifs -obtenir le nom d'une base, les concepts d'une base -ajouter un concept à un base -retirer un concept d'une base
Concept	-créer, accéder, supprimer -obtenir le nom d'un concept -obtenir les attributs clés d'un concept -obtenir l'ensemble des attributs d'un concept -obtenir l'ensemble des points de vue d'un concept -obtenir l'ensemble des passerelles d'un concept -obtenir l'ensemble des instances d'un concept
Attribut de concept	-créer, accéder, supprimer -obtenir le nom d'un attribut -obtenir le concept père d'un l'attribut -connaître son nom, type, nature, constructeur
Point de vue	-créer, accéder, supprimer -obtenir le nom d'un point de vue -obtenir le concept père du point de vue -obtenir la classe racine du point de vue
Classe	-créer une sous-classe d'une autre -accéder, supprimer, obtenir le nom d'une classe -obtenir le point de vue père d'une classe -obtenir sa super-classe, ses sous-classes -obtenir les instances liées à cette classe
Attribut de classe	-créer et ajouter à une classe -accéder, supprimer, obtenir son nom -obtenir la classe d'appartenance d'un attribut -définir des contraintes -définir des mécanismes d'inférence
Passerelle	-créer et ajouter à un concept, accéder, supprimer -obtenir les classes origines et la classe destination
Instances	-créer et ajouter à un concept, accéder, supprimer -obtenir son identité -obtenir le concept père d'une instance -obtenir les classes avec lesquelles il est lié -obtenir les valeur de ses attributs, les modifier -obtenir les valeurs alternatives par inférence -lier une instance à une classe

5. Conclusion

Tropes est un système de gestion de bases de connaissances à objets dont la structure présente quelques originalités. A noter principalement, une représentation multi points de vue du concept, chaque point de vue ayant sa propre hiérarchie de classes. D'autre part, il existe une API standard permettant de visualiser et modifier les différents éléments de structure des bases de connaissances Tropes.

C. Le World Wide Web

Le World Wide Web (ou Web ou WWW ou W3) est aujourd'hui le service d'information électronique le plus représentatif d'Internet. Il permet à toute machine du réseau de créer et présenter sous forme graphique et interactive ses propres informations. Les protocoles de communication qui permettent d'y accéder sont standardisés et permettent également d'y accéder universellement.

Au mois de Novembre 1995, entre 80.000 et 100.000 sites Web,15 millions de pages, étaient enregistrées dans le monde entier. Six mois plus tard c'est 22 millions de pages réparties sur 225.000 sites. Qui de nos jours n'a pas son propre serveur Web ? L'intérêt de ce support est donc largement affirmé. Aujourd'hui, la grande majorité du World Wide Web fonctionne sur le principe de la diffusion: l'information, contenue sur un serveur, est consultée à distance. A cela devrait se substituer un modèle d'interaction où l'utilisateur adressera des requêtes précises au serveur, qui lui adressera en retour une réponse personnalisée. Cette migration d'un système encore statique vers un autre plus dynamique implique aussi d'autres progrès dans la capacité de transport de l'information et dans la sécurisation des transactions.

Cette partie présente le World Wide Web, ses origines, son intérêt, ses caractéristiques techniques et ses principales applications dans l'entreprise. Accéder au WWW de la façon la plus conviviale possible a donné lieu à un nouveau type d'application, les navigateurs Web, il sera donc question de les présenter également.

1. Les origines du projet

La création de World Wide Web a été dirigée depuis le CERN[7] de Genève par Tim Berners Lee et son équipe pour résoudre des problèmes de création, d'accès et de partage d'informations entre membres de la communauté de recherche nucléaire. WWW combine l'utilisation de l'infrastructure de communication Internet et des protocoles de communication communs, définis par le CERN et NCSA[8].

Depuis 1993, l'intérêt des entreprises pour Internet et WWW est net. la connexion à Internet leur permet de consulter les ressources disponibles puis souvent de créer leur propre serveur de données.

- L'apport documentaire de WWW par l'accès aux ressources des réseaux est l'application la plus évidente. Les informations délivrées sont de natures bibliographiques, scientifiques ou techniques, financières, juridiques...

- Des serveurs d'informations commerciales (vente, marketing, catalogues...) ou vecteurs d'images (présence culturelle ou industrielle, mécénat...) sont aussi apparus.

[7]Centre Européen de Recherche Nucléaire
[8]National Center for Superconducting Application

- Un marché étendu de services commerciaux sur Internet existe donc potentiellement, même si la présence commerciale effective est encore faible : En 1995, sur 100000 serveurs, seules 8.500 entreprises américaines ont un site spécifiquement commercial. Aujourd'hui, on s'attend à une croissance rapide, pouvant atteindre 182.000 entreprises présentes sur le World Wide Web en l'an 2000. La présence des entreprises sur le réseau prend des formes diverses, allant de la présence "médiatique" au serveur transactionnel, marginal dans l'attente des développements transactionnels.

  Une évolution remarquable est l'adaptation des réseaux internes aux échanges d'informations calqués sur ceux du réseau Internet: usage de la messagerie, d'hypertextes "corporate"...

On trouve déjà presque tout sur Internet: des serveurs d'informations officielles du gouvernement et du congrès aux serveurs permettant la vente de Pizza ou de matériels informatique, des régies publicitaires aux producteurs d'information financières, des entreprises industrielles ou universités... Il véhicule aussi bien la vente, la communication interne ou externe, ou stimule des formes de travail en collaboration...

2. Source d'information

La pluridisciplinarité d'Internet et son caractère humaniste ont longtemps permis la circulation d'informations inédites et gratuites, sur de multiples sujets. Le volontariat alimentait la base Internet. Aujourd'hui, chaque entreprise, université qui crée son serveur ou une information ajoute au volume des données disponibles (à moins qu'elle n'en restreigne volontairement l'accès public ou en décide la facturation).

Internet/WWW se compose des informations scientifiques issues des universités et des entreprises industrielles, des documents officiels émis par les gouvernements, les ministères, les documents commerciaux et techniques... Les statistiques et les informations financières représentent une part croissante de ce corpus (estimée potentiellement à 30% de l'ensemble), à laquelle s'ajoute des informations commerciales et culturelles.

L'hypertexte et le réseau de connaissance:

La technologie hypertexte permet de s'affranchir de l'aspect linéaire des documents et de la séquentialité de la lecture. La "navigation" permet de désigner et d'agir sur le document, et non plus seulement en parcourir le contenu. Un document comporte souvent des liens à d'autres documents, en plus des informations intrinsèques qu'il véhicule et (des)oriente la recherche par sujet. Aujourd'hui, l'indexation du réseau WWW permet de retrouver l'information pertinente dans cet ensemble en perpétuelle croissance. De nombreux catalogues de ressources, classant par sujet, par origine, et des index en texte intégral permettent l'interrogation globale de WWW, soit la recherche parmi plus de 25000 sites et 3 milliards de pages ou documents.

3. Logiciels d'Accès

a. Mosaic

En 1993, le NCSA a développé un programme permettant la consultation de WWW: "Mosaic", utilisable sur la plupart des machines du marché, gratuit et offrant des fonctionnalités avancées. Ce programme permet à l'utilisateur une manipulation simple des informations tandis qu'il coordonne l'ensemble des aspects techniques permettant l'accès aux informations : connexion au réseau, localisation et transfert des fichiers, affichage multimédia.

- C'est un logiciel hypertexte : On sélectionne simplement des mots ou régions mis en valeur pour faire apparaître les informations qui s'y rapportent. Mosaic recherche et affiche ces documents, quelle que soit la complexité de ces opérations, sans que l'utilisateur ait à se préoccuper de la manière dont cela s'opère.

- Mosaic est multimédia: Il affiche indifféremment du texte, des images fixes ou animées et des sons. Il intègre aussi sous un accès uniforme la communication avec des documents complexes et certaines bases de données.

Mosaic coopère avec le Web grâce à l'architecture client-serveur du World Wide Web. WWW réalise les connexions (l'adressage, les protocoles, et la négociation entre ressources), il s'agit donc du serveur. Mosaic réalise la présentation des données, et est la boîte à outils qui permet de naviguer, sélectionner, consulter les documents et éventuellement les récupérer (le client). Le développement de WWW comme outil de communication ou d'information doit beaucoup à ses capacités avancées et à sa facilité d'emploi.

b. Les évolutions

Netscape Navigator

Une scission du groupe de développement du logiciel Mosaic, en 1994, a donné naissance à une nouvelle entreprise, Netscape Corp, qui développe actuellement le produit commercial "Netscape Navigator". Il a donné à Mosaic une interface plus conviviale et rapide, un affichage plus rapide et progressif des données, l'accès simultané à plusieurs serveurs, une réelle intégration graphique (JPEG), et l'intégration réussie de la messagerie et des forums de discussions. Navigator, bien qu'étant devenu payant, reste le navigateur le plus utilisé, notamment, grâce à son portage sur de nombreux systèmes.

Les alternatives

La niche des navigateurs Web n'est pas restée la seule propriété de Navigator, d'autres logiciels de ce type sont apparues (certains ont déjà disparus). La principale alternative à Navigator est sans doute «Internet explorer» du géant Microsoft. La gratuité de ce

logiciel est sans doute pour quelque chose à la croissance régulière du nombre d'utilisateur. On peut également citer «Hot Java» de Sun.

4. Les aspects techniques du WWW

WWW échange des informations via Internet, il répartit entre les clients et les serveurs les opérations de connexion. Pour cela le Cern a du définir un protocole d'échange de données au travers du réseau, HTTP. En coopération avec le NCSA, il a également conçu et distribué gratuitement l'interface Mosaic, qui intègre cette technologie et utilise le protocole HTTP lors d'un accès aux informations du WWW. Les concepteurs du WWW ont également prévu dans WWW d'intégrer la plupart des protocoles et services qui existaient déjà sur Internet. Ainsi, une application pouvait remplacer toutes les autres et les unifier. La magie du WWW a tenu à cette facilité d'emploi: fini les chaînes de caractères ésotériques et les modes d'emplois fastidieux. Instantanément, WWW savait autoriser des connexions sous FTP, Gopher, WAIS, afficher textes, images et sons sans précautions particulières...

WWW utilise les communications sur le réseau Internet et partage entre "clients" et "serveurs" les étapes qui donnent l'accès à une information, donc la charge d'utilisation du réseau. Le "client" émet une requête vers un "serveur" WWW, en lui demandant la copie à travers le réseau d'un document spécifique. En retour, le serveur accepte (ou non! Le serveur gère et donne l'accès aux documents qu'il juge possible de diffuser) la connexion et commence la copie du fichier. Le client reçoit le document et est responsable de sa présentation.

Ainsi :

1. L'utilisateur souhaite se connecter au serveur http://www.pratique.fr. II entre l'adresse (l'URL) dans son outil de navigation. Ensuite, sur une page déjà affichée, il sélectionne le lien hypertexte correspondant au document qu'il souhaite consulter.

2. Le programme d'accès à WWW utilise l'adresse correspondante à ce lien et se connecte tout d'abord au serveur de nom (DNS), qui lui permet de diriger la requête vers le serveur WWW correspondant. Un protocole spécial, HTTP, gère cette connexion.

3. Le serveur reçoit la demande et retourne en retour le ou les fichiers textes et images au client. Ce dernier utilise son navigateur afin d'effectuer la mise en page du document (grâce au langage de description de page Web, HTML) et son affichage à l'écran. Ainsi, la charge de travail est répartie entre le client (le demandeur de l'information) et le serveur (celui qui la délivre).

Le fonctionnement du Web est donc gouverné par trois entités: un protocole (HTTP), une norme d'adressage (l'URL) et un langage de description des pages Web (HTML). Ce sont ces trois standards qui ont permis la généralisation des mécanismes d'échanges et de présentation de documents, et surtout des mécanismes d'adressage universels.

- **L'URL**: Un «Uniform Resource Locator» correspond à la définition unique d'adresses permettant de localiser un document sur le Web.

- **HTTP**: «Hypertext Transfer Protocol» est un protocole réseau rapide, sûr et aux capacités étendues; il permet le transfert de documents multi-formats. C'est ce protocole qui régit les échanges d'informations entre postes clients et serveurs Web (ou serveurs HTTP), il se sert de la structure de l'URL pour rechercher l'information sur le réseau.

- **HTML**: «HyperText Markup Language» est une syntaxe spécifique permettant de décrire la structure d'un document destiné au Web. Elle permet notamment d'enrichir ces documents de liens vers d'autres informations, de formulaires de saisie, d'images... Le document renvoyé par HTTP est donc au format HTML et le navigateur n'a plus qu'à l'interpréter avant de l'afficher.

III. L'existant

Dans le chapitre précédent une présentation du système à base de connaissances Tropes, ainsi que du Web ont été effectuées. L'objet de ce mémoire est d'ajouter à Tropes une interface permettant la modification des bases de connaissances par l'intermédiaire du Web. Une interface permettant la consultation de bases Tropes au travers du Web (Hytropes) a déjà été développée, aussi cette partie détaillera le fonctionnement et l'ergonomie de cette interface, ce pourrait être un bon support pour l'implémentation des fonctionnalités d'édition. D'autre part un logiciel similaire à Tropes, dans le sens modélisation de connaissances, présente une interface de consultation et également d'édition d'ontologies au travers du Web (Ontolingua), il parait donc particulièrement intéressant d'étudier ce logiciel.

A. Hytropes

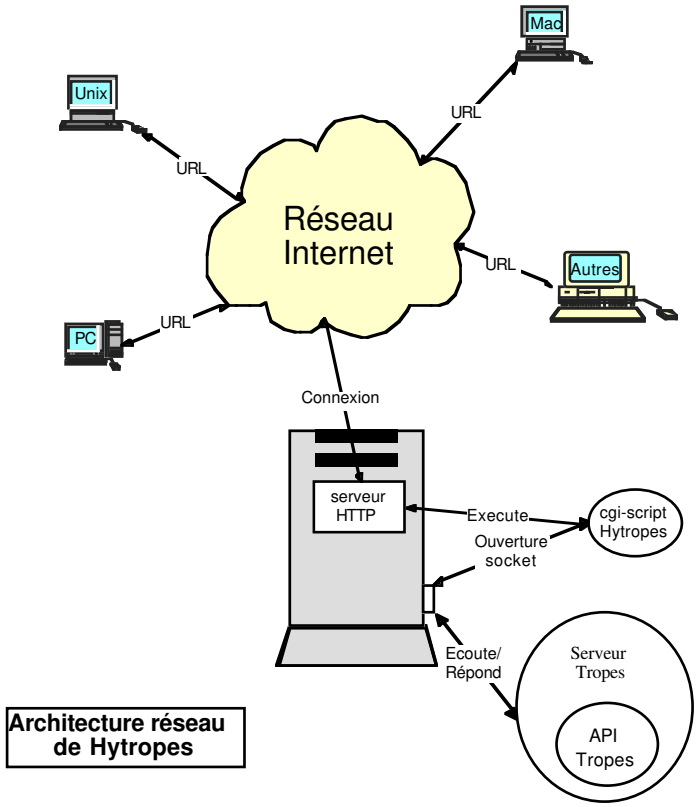
Hytropes est une extension de Tropes. Elle permet, via le Web de visualiser des bases de connaissances créées par Tropes. Le sujet sur lequel je travaille, doit permettre à Hytropes non seulement de visualiser une base Tropes, mais également de pouvoir en modifier le contenu ou d'en créer de nouvelles. Il est donc important de maîtriser le fonctionnement de Hytropes. Pour ce faire, trois parties seront développées:

- La première partie présente l'architecture réseau mise en place afin de pouvoir établir une connexion entre le Web et Tropes.
- La seconde détaille la syntaxe commune de toutes les URLs de Hytropes
- Pour finir, une présentation de la structure et de l'enchaînement des différents écrans permet de se faire une idée plus concrète du fonctionnement de l'application.

1. Architecture réseau

L'architecture réseau du serveur Hytropes est décomposée en trois couches successives permettant de lier une base de connaissances Tropes à un poste utilisateur:

- Le serveur Tropes
- le cgi-script Hytropes
- l'ensemble navigateur Web/serveur HTTP



Architecture réseau de Hytropes

a. Le serveur Tropes:

Le serveur Tropes est ce qu'on appelle (dans le jargon du système d'exploitation Unix) un «démon». Il s'agit d'un programme qui une fois lancé s'exécute en tâche de fond (process). Son rôle est de guetter continuellement un certain port d'une station de travail, ceci afin de voir si une requête (sous forme d'URL) est déposée sur celui-ci. Si tel est le cas, le serveur Tropes traite la demande à l'aide des fonctions de l'API de Tropes, construit la page HTML représentant la réponse à la requête, ou un message d'erreur le cas échéant. Le document ainsi réalisé est retourné sur le port de la machine en question. Techniquement ce programme est réalisé à l'aide du langage Ilog Talk [Talk], ceci pour une connexion plus aisée avec l'API de Tropes qui elle même a été réalisée avec ce langage.

b. Le cgi-script Hytropes

Traditionnellement, lorsque l'on saisit un URL dans un navigateur Web, il nous est retourné un document **statique**, c'est-à-dire un fichier dont les informations ne changent pas tant qu'une personne n'a pas édité ce fichier, modifié à la main son contenu puis décidé de le sauvegarder. Cette méthode n'est pas compatible avec un système d'interrogation de bases de connaissances. Dans ce cas nous avons besoin de créer des pages **dynamiques**. Les cgi-script sont là pour permettre la création de pages dynamiques. Le rôle du cgi-script Hytropes est d'établir la liaison du serveur HTTP avec le serveur Tropes et de soumettre à ce dernier la requête de l'utilisateur.

c. Le couple navigateur Web/serveur HTTP

Aucun service Web sur Internet ne peut fonctionner sans le couple navigateur Web et serveur HTTP. Le navigateur permet à l'utilisateur de visualiser des pages HTML, c'est lui qui établit également la connexion avec le serveur HTTP. Ce dernier permet soit d'aller chercher un fichier statique sur la machine sur lequel il se trouve, soit dans notre cas (pages dynamiques), il permet d'exécuter le cgi-script Hytropes chargé de la connexion avec le serveur Tropes. Ce dernier fabriquera une page HTML, la retournera au serveur HTTP, toujours par le biais du même script et à son tour le serveur HTTP présentera le document aux yeux de l'utilisateur par l'intermédiaire de son navigateur Web.

2. Les URLs et Hytropes

Qui dit Web, dit forcément URLs puisque ce sont eux qui permettent de localiser et d'accéder à une ressource sur Internet. Une syntaxe particulière, mais compatible avec celle du service HTTP d'Internet a été définie afin de pouvoir accéder aux différents composants d'une base Tropes. Elle utilise notamment les scripts qui permettent de soumettre une procédure sur un serveur. Dans ce cas, c'est cette procédure qui est chargée d'interpréter la demande, l'exécuter puis rapatrier le résultat sur l'écran du demandeur.

On peut décomposer un URL Hytropes en quatre parties (dont une est optionnelle):
- Une racine commune
- Un service demandé à Hytropes
- Le composant sur lequel le service est rendu
- Une suite d'arguments (si l'URL est soumis à partir d'un formulaire de saisies)

a. La racine commune

http://nom-serveur-http:num-port/cgi-bin/hytropes/

«http://» signifie que l'on souhaite accéder au service du Web

«nom-serveur-http» correspond au nom du serveur HTTP recherché

«:num-port» est le numéro du port de la machine sur lequel le serveur HTTP est installé. Ce paramètre est optionnel si sa valeur est 80 (port recherché par défaut).

«hytropes» et «/cgi-bin/hytropes/» signifient que pour interpréter l'URL il faudra exécuter le script (cgi-script) de nom «hytropes» avec la suite de l'URL comme paramètres.

b. *Le service demandé à Hytropes*

URL	Fonction
/get	Affichage en mode visualisation d'un composant Tropes
/stats	Affichage des statistiques concernant une base Tropes
/find-similar	Permet de rechercher des objets similaires à d'autres
/get-filter	Affiche un masque de saisie permettant de sélectionner des objets dans un concept correspondant à ce masque
/get-classif	Affiche un masque de saisie permettant d'effectuer la classification d'un objet dans plusieurs points de vue

c. *Le composant Tropes concerné par le service demandé*

URL composant	composant Tropes
/nom-base/nom-base.html	une base
/nom-base/nom-concept/nom-concept.html	un concept
/nom-base/nom-concept/s/nom-slot/nom-slot.html	un attribut de concept
/nom-base/nom-concept/v/nom-vue/nom-vue.html	un point de vue de concept
/nom-base/nom-concept/v/nom-vue/nom-classe/nom-classe.html	une classe
/nom-base/nom-concept/v/nom-vue/nom-classe/nom-classslot/nom-classslot.html	un attribut de classe
/nom-base/nom-concept/b/nom-bridge/nom-bridge.html	une passerelle
/nom-base/nom-concept/o/nom-objet/nom-objet.html	une instance d'un concept

La syntaxe désignant les composants de Tropes que l'on manipule, a été déterminée afin d'être compatible avec la gestion de fichier du système d'exploitation sur lequel est mis Hytropes. De cette façon il est possible de sauvegarder sur le disque dur du système l'ensemble des pages HTML décrivant les composants d'une base de connaissances Tropes, il est alors possible de visualiser cette base sans utiliser Tropes, ni Hytropes. Bien entendu cette consultation est une image figée de la base de connaissances au moment où on l'a sauvegardée.

d. *Une suite d'arguments*

Une suite à l'URL peut être présente lorsque l'on doit traiter des données saisies par un utilisateur par l'intermédiaire d'un formulaire HTML, dans ce cas les données à traiter apparaissent de la façon suivante:

```
?nom-donnée-1=valeur-1&nom-donnée-2=valeur-2&...&nom-donnée-n=valeur-n
```

«?» signifie que la suite de l'URL correspond à une liste d'arguments (issues la plupart du temps d'un formulaire HTML).

«nom-donnée-n» correspond au nom du champ de saisie HTML placé en nième position dans le formulaire et «valeur-n» correspond à ce qu'a choisi l'utilisateur pour ce champ.

3. L'interface de visualisation

La définition de l'interface de visualisation est présentée dans [Euzenat 96]. Je vais cependant en retracer quelques particularités intéressantes pour l'interface de modification. Notamment, j'évoquerai comment est organisée la navigation, comment sont structurés les écrans et quels sont les services fournis par cette application.

a. La navigation dans Hytropes

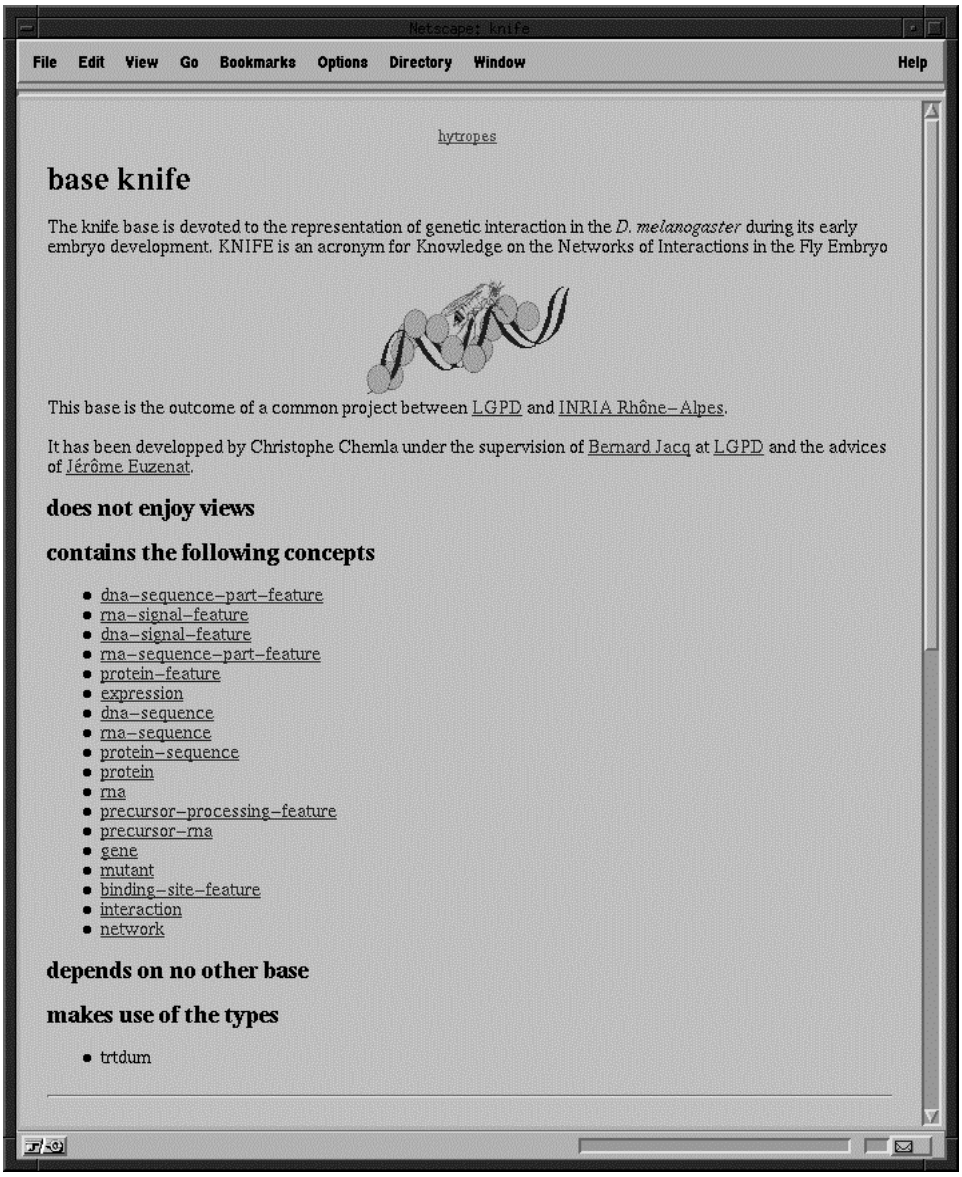
La navigation dans Hytropes suit intégralement la structure des bases de connaissances Tropes. Le point de départ de l'interface présente principalement l'ensemble des concepts définis dans la base. Une différence tout de même, profitant des possibilités graphiques du Web, est la possibilité d'insérer des fichiers HTML comme illustration du composant affiché. Il peut se situer en entête ou pied de page du document présenté.

L'exemple suivant montre la page d'accueil de la base de connaissances KNIFE concernant les interactions entre gènes chez la drosophile.

On peut y découvrir:

- La liste des concepts concernant ce sujet (structure Tropes).

- Une image ainsi que des liens vers d'autres sites concernant le sujet ont été rajoutés en tête de document. Il s'agit d'un document HTML qui a été lié à la base knife à l'aide d'une annotation spécifique.



Exemple d'écran Hytropes

En sélectionnant avec la souris un concept à l'écran, on peut le visualiser et ainsi découvrir les attributs qui le composent, les objets créés et les points de vue depuis lesquels on peut les observer. Ainsi de suite si on sélectionne un point de vue on découvre la hiérarchie de classe qu'il représente...

A noter également que le titre de chaque page fait toujours référence à un composant de niveau inférieur à celui sur lequel on se trouve afin de pouvoir redescendre d'un niveau dans la structure de Tropes.

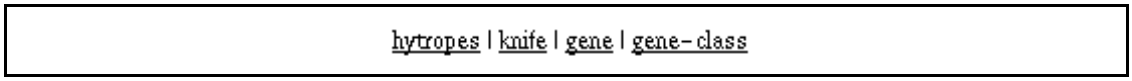
Exemple:

class posterior-system in the viewpoint gene-class

Ce titre indique que nous visualisons la classe «posterior-system» qui se trouve dans le point de vue «gene-classe». Le mot «gene-class» est en fait un lien qui permet de retourner sur le document présentant le point de vue «gene-classe».

Pour finir au sujet de la navigation, il faut noter également qu'en entête de chaque page se trouve ce qu'on appelle un «localizer», qui comme son nom l'indique, permet de se localiser facilement dans la base. Il permet également de pouvoir naviguer rapidement dans celle-ci sans avoir la nécessité de descendre un à un les niveaux comme vu ci-dessus.

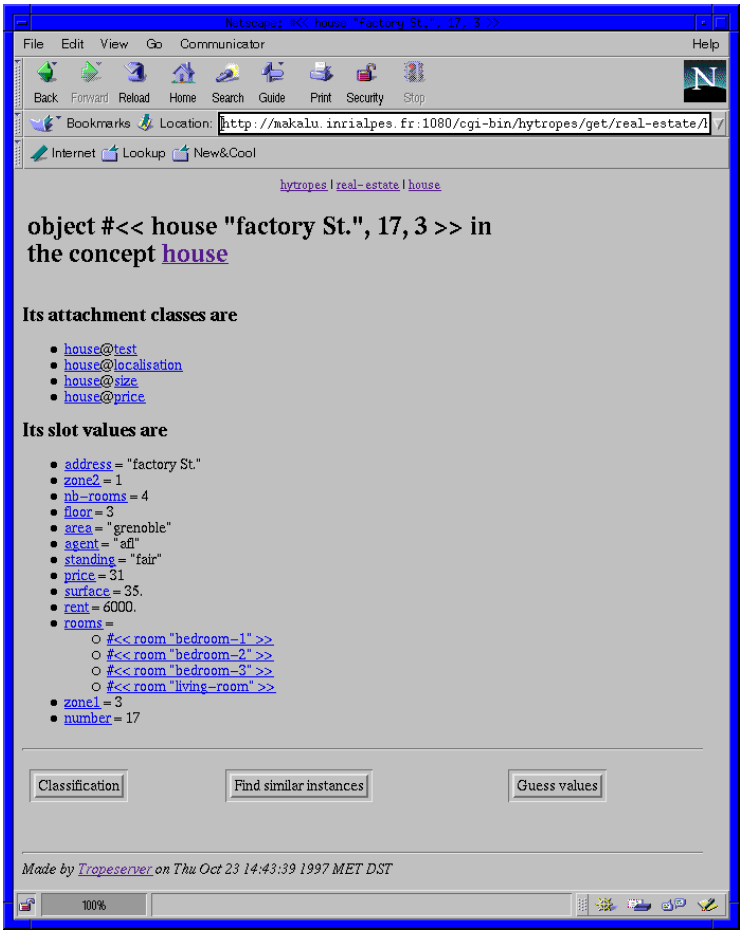
L'exemple suivant montre le «localizer» correspondant à la page HTML représentant une classe du point de vue «gene-class» du concept «gene» de la base «knife»:



En cliquant sur l'un des liens affichés, on accède directement à la page HTML décrivant l'élément de structure choisi, ici soit la page représentant le point de vue «gene-class», soit celle du concept «gene» soit celle de la base «knife» soit sur la page d'accueil de Hytropes qui permet de choisir une base disponible sur le serveur.

b. Les services fournis par Hytropes

J'ai pu évoquer dans la définition de ce qu'est une base de connaissances, quelle est sa finalité. Il s'agit, pour rappel, de stocker, d'organiser la connaissance ainsi que de rendre des services permettant de l'exploiter. L'interface de Hytropes ne se contente donc pas de pouvoir naviguer à travers les différents composants d'une base Tropes. Lorsqu'un service est implémenté pour un composant, un «bouton poussoir» libellé au nom du service permet en cliquant dessus avec la souris d'accéder à ce service.

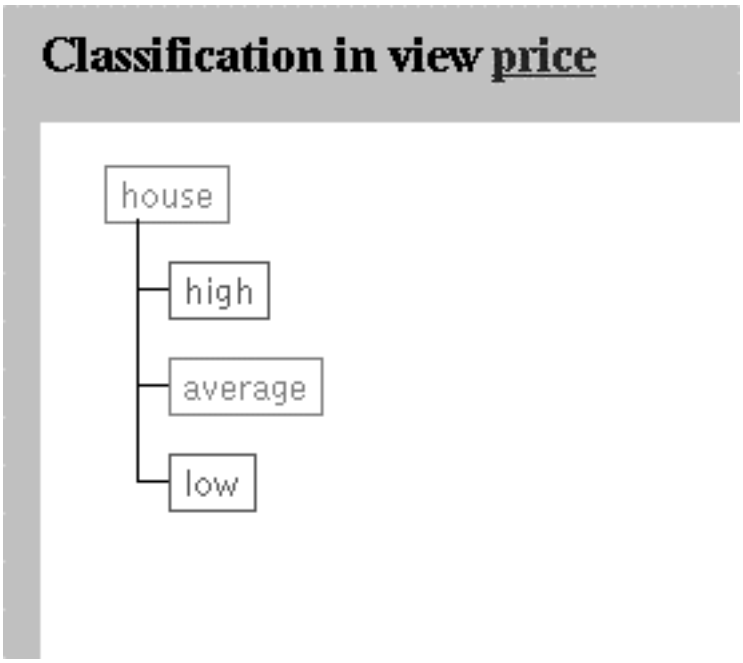


Exemple de la consultation d'une instance:

- On peut visualiser sur cet écran l'ensemble des classes d'attachement (classe «house» du point de vue «test»...) ainsi que la liste et les valeurs des attributs (agent = afl...) de l'appartement «factory St. 17 3»

- On remarquera également les boutons permettant l'accès aux différents services de Hytropes (classification, statistiques sur la base, valeurs inférées).

L'exemple suivant montre le résultat de la classification d'une instance suivant le point de vue du prix (price), on peut découvrir une représentation de la structure arborescente des classes. Les couleurs ne ressortent pas, mais lorsqu'une classe ne peut prétendre l'accueil d'une instance elle est représentée par un rectangle **rouge**, lorsque toutes les informations permettent d'affirmer que l'instance peut être rattachée à une classe, la représentation de cette dernière est un rectangle **vert**. Pour finir lorsque l'ensemble des informations est insuffisant pour effectuer une affirmation, le rectangle représentant la classe est **orange**.



Résultat d'une classification:
classes interdites : «high» et «low» (en rouge)
classes autorisées : «house» et «average» (en vert)
classes incertaines : aucune (en orange)

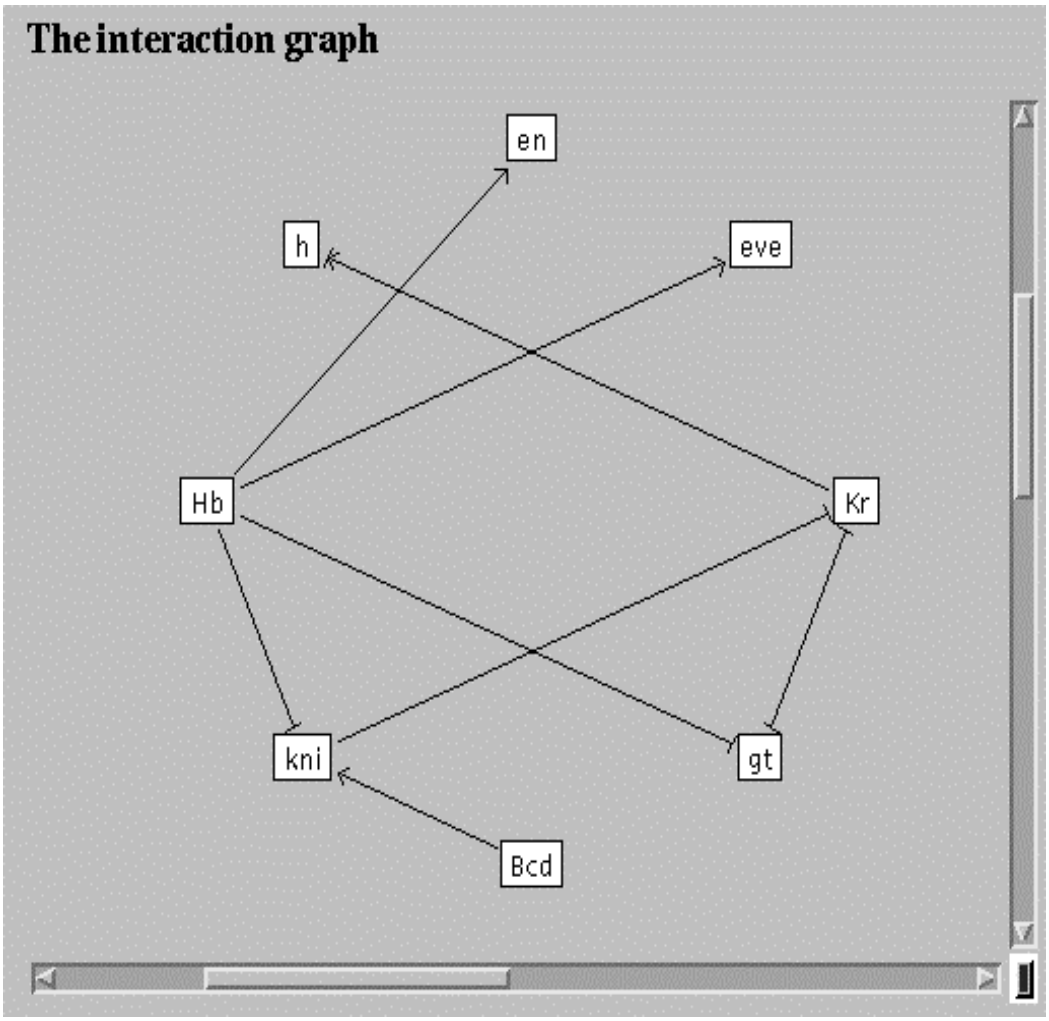
Remarque:

- L'interface a été conçue pour être compatible avec un maximum de «browser Web», elle utilise donc au mieux les possibilités du langage HTML qui est la référence dans ce domaine. Depuis peu, il est possible d'introduire dans les pages HTML des éléments d'interface plus sophistiqués (Applet Java[9], scripts Javascript...). Plus long à charger ils apportent cependant une convivialité accrue au niveau de l'interface utilisateur. Le résultat de la classification ci-dessus est issue d'une Applet Java.

Autre exemple d'Applet :

Le résultat de la demande de recherche d'interactions entre un certain nombre de gènes peut être représenté sous une forme graphique particulièrement parlante pour le biologiste grâce à Java.

[9]Application développée à l'aide du langage de programmation Java et destinée au Web



Réseau d'interactions entre gènes chez la drosophile

Le réseau ci-dessus montre l'interaction entre les gènes en (engrailled), eve (even skipped), Kr (Kruppel), gt (giant), Bcd (Bicoid), kni (knirps), Hb (Hunchback) et h. Ce type de schéma est une aide précieuse pour le biologiste averti, il représente les interactions entre ces gènes d'une façon bien plus conviviale qu'on ne pourrait le faire avec du texte uniquement.

B. Ontolingua

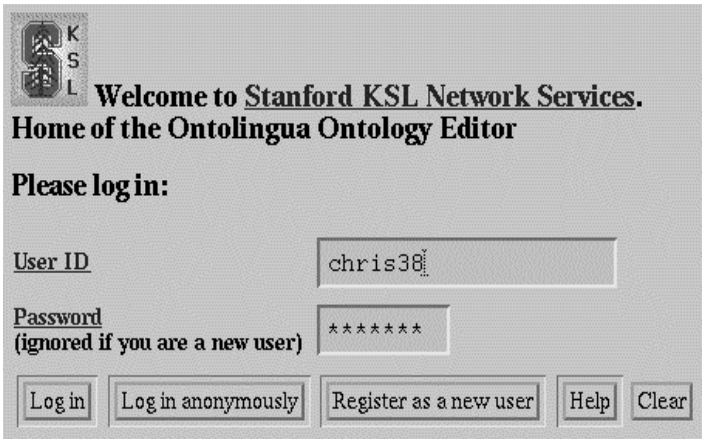
La partie précédente a pu montrer les aspects techniques et le fonctionnement du logiciel de visualisation de bases Tropes qu'est Hytropes, l'objectif est d'ajouter à ce dernier des possibilités d'éditions. Ontolingua est un logiciel permettant la consultation et l'édition d'entités appelées «ontologies» à l'aide du Web. Une ontologie pouvant être assimilée à une base de connaissances, il parait intéressant d'étudier ce logiciel et en particulier de quelle manière sont assurés, le contrôle d'accès, la concurrence d'accès et les fonctionnalités d'édition du serveur.

1. Présentation

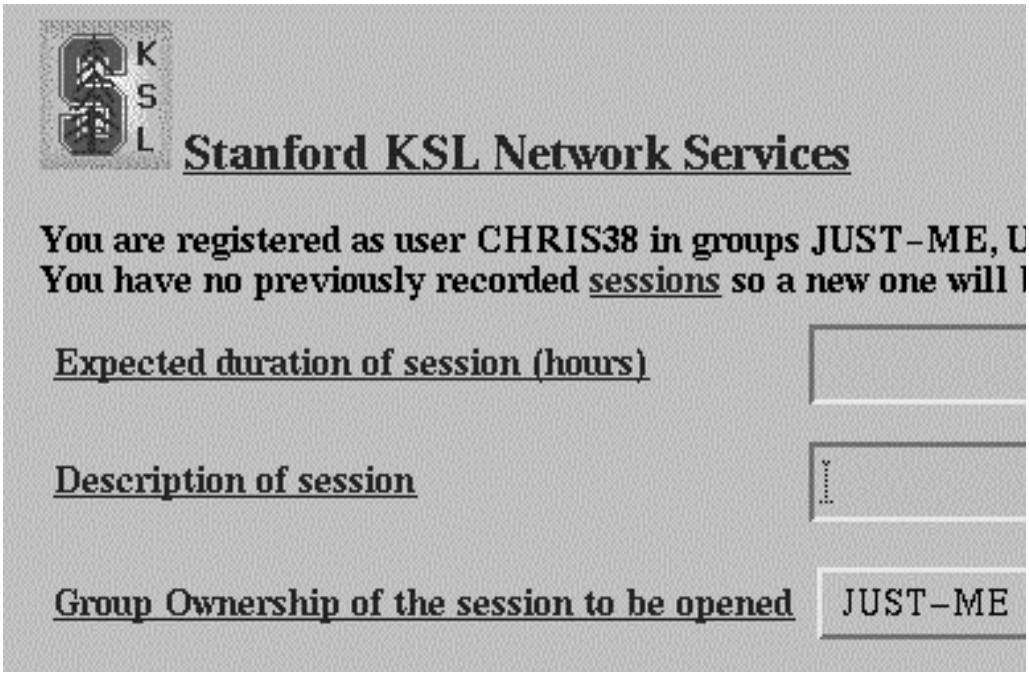
Ontolingua [Farquhar 95] est une application permettant la visualisation, la création et l'édition «d'Ontologies» via le service Web d'Internet. Brièvement, une «Ontologie» est une spécification (pouvant être formelle ou informelle) qui définit le vocabulaire d'un domaine de discours. Cette définition reste très vague, cependant je ne vais pas entrer trop dans le détail de la structure d'une «Ontologie» et ce que l'on peut en faire. L'objet de cette partie est en effet de traiter l'aspect ergonomique et technique du serveur Web mis en place, afin d'en tirer quelques profits dans mon interface; il ne s'agit pas de décortiquer la structure d'Ontolingua en vue d'une réécriture de Tropes. Simplement, pour montrer le lien entre Hytropes et Ontolingua, je dirais que ce dernier représente la connaissance sous forme de groupes d'éléments de type «classes» (au sens objet du terme), avec des attributs, des sous-classes, ... et qu'il offre des services similaires à Hytropes.

2. L'accès au serveur

L'édition de base Tropes via Hytropes n'étant pas encore développée, il n'a pas été jugé prioritaire de gérer de sécurité d'accès au serveur. Ce ne sera plus le cas bientôt, aussi vais-je m'intéresser à ce sujet.
L'écran d'accueil présente un masque de saisie dans lequel l'utilisateur doit s'identifier par un nom d'utilisateur et un mot de passe.



Il se présente alors un second écran dans lequel l'utilisateur prévoit le temps de connexion dont il aura besoin, une brève description de l'objet de la connexion au serveur et pour finir le groupe d'utilisateurs dont il fait partie.



Le nom de l'utilisateur saisi fait partie intégrante de la structure des URLs d'Ontolingua, il permet par la suite de s'assurer qu'un service demandé au serveur soit bien effectué par un utilisateur connecté. Le groupe d'utilisateurs permet au serveur de savoir quelles bases doivent être mises ou non à disposition de l'utilisateur en fonction de certains paramètres.

remarque:

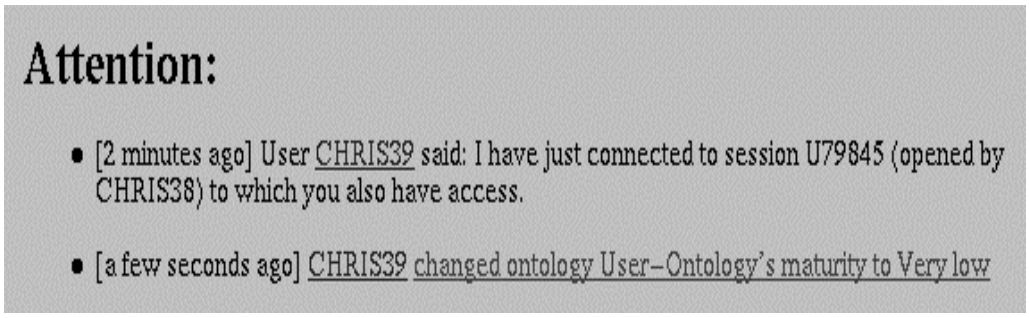
Le nom de l'utilisateur ainsi que son mot de passe apparaissent dans l'URL résultante de la validation du premier écran de saisie, ce qui rend incertain cette méthode de sécurisation.

3. Le travail de groupe

Pour faciliter le travail en groupe, une notion de session a été implémentée dans Ontolingua. Lorsqu'un utilisateur se connecte au serveur au titre d'un certain groupe de travail, il a alors le choix de se joindre à une session déjà ouverte pour ce groupe ou bien d'ouvrir une nouvelle session.

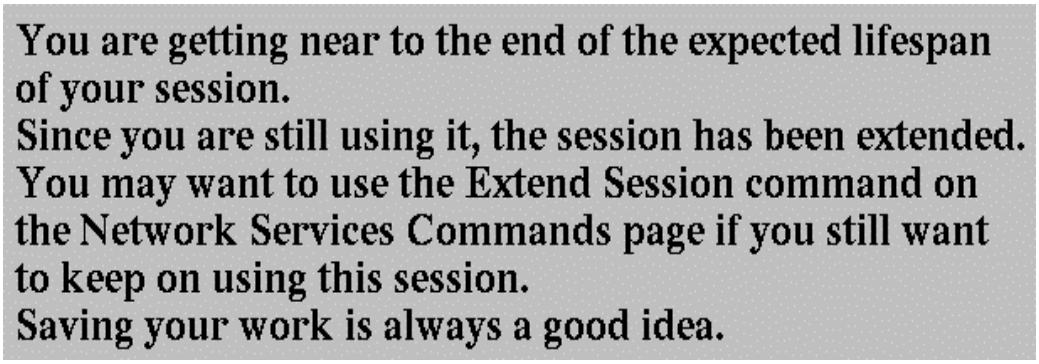
- L'ouverture d'une nouvelle session provoque en fait une copie de l'ontologie choisie dans un espace de travail propre à chaque session. De cette façon, les modifications effectuées dans une session par un utilisateur n'affectent que la copie.

- Le choix de joindre une session déjà ouverte par un autre utilisateur permet de travailler à plusieurs sur une même copie. Dans ce cas, lorsqu'un utilisateur effectue une modification, les autres utilisateurs de la même session sont alors notifiés de l'action effectuée et ce dès qu'ils mettent à jour l'affichage de leur navigateur Web (reload) ou dès lors qu'ils tentent eux-mêmes une modification.

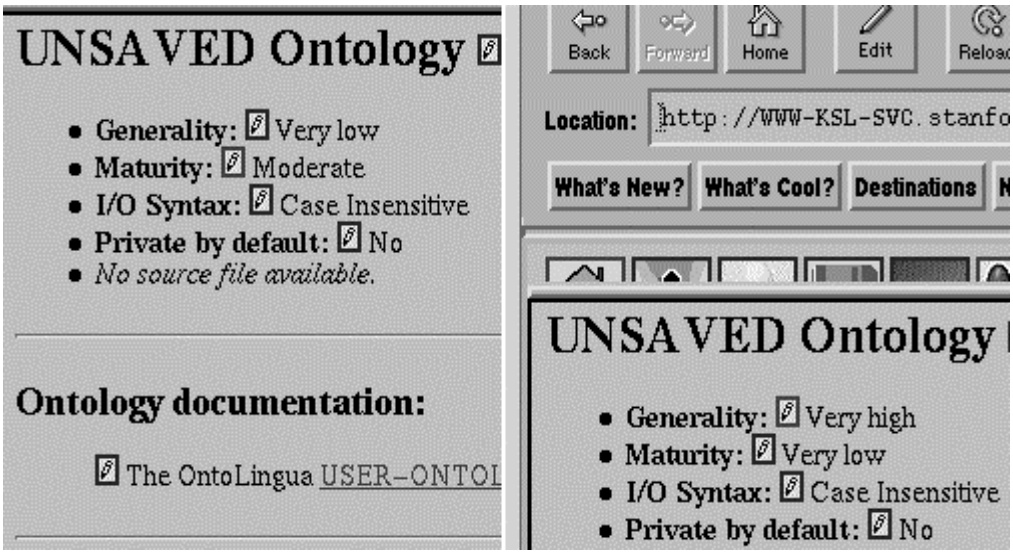


Remarques:

- J'ai pu évoquer dans la première partie concernant Ontolingua qu'un temps de session était requis à la connexion. Lorsque le temps de connexion d'une session est dépassé, la session est détruite (plusieurs utilisateurs peuvent être concernés) et les modifications non sauvegardées sont perdues. Un message signale tout de même lorsqu'on se rapproche du moment fatidique, cependant les conséquences d'un tel procédé peuvent être lourdes.



- Lorsqu'une même Ontologie est manipulée dans deux sessions différentes, chaque utilisateur possède sa propre copie de l'objet manipulé.



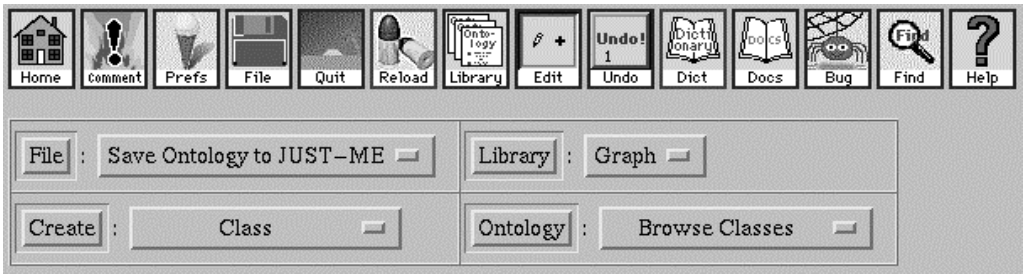
Un mécanisme, visiblement non intégré à l'éditeur, devra permettre d'obtenir au final une base consensuelle regroupant les modifications de chacun, avec tous les soucis que cela peut comporter (incohérences, ...). Cependant je n'ai pas trouvé de documentation à ce sujet. Au sein du projet Sherpa de l'INRIA, des travaux très actifs sont poursuivis afin de résoudre ce problème, mais d'une manière bien différente [Euzenat 95]. Ce travail sera prochainement intégré à Hytropes lorsque les possibilités d'édition seront mises en place et validées par les utilisateurs.

4. L'interface

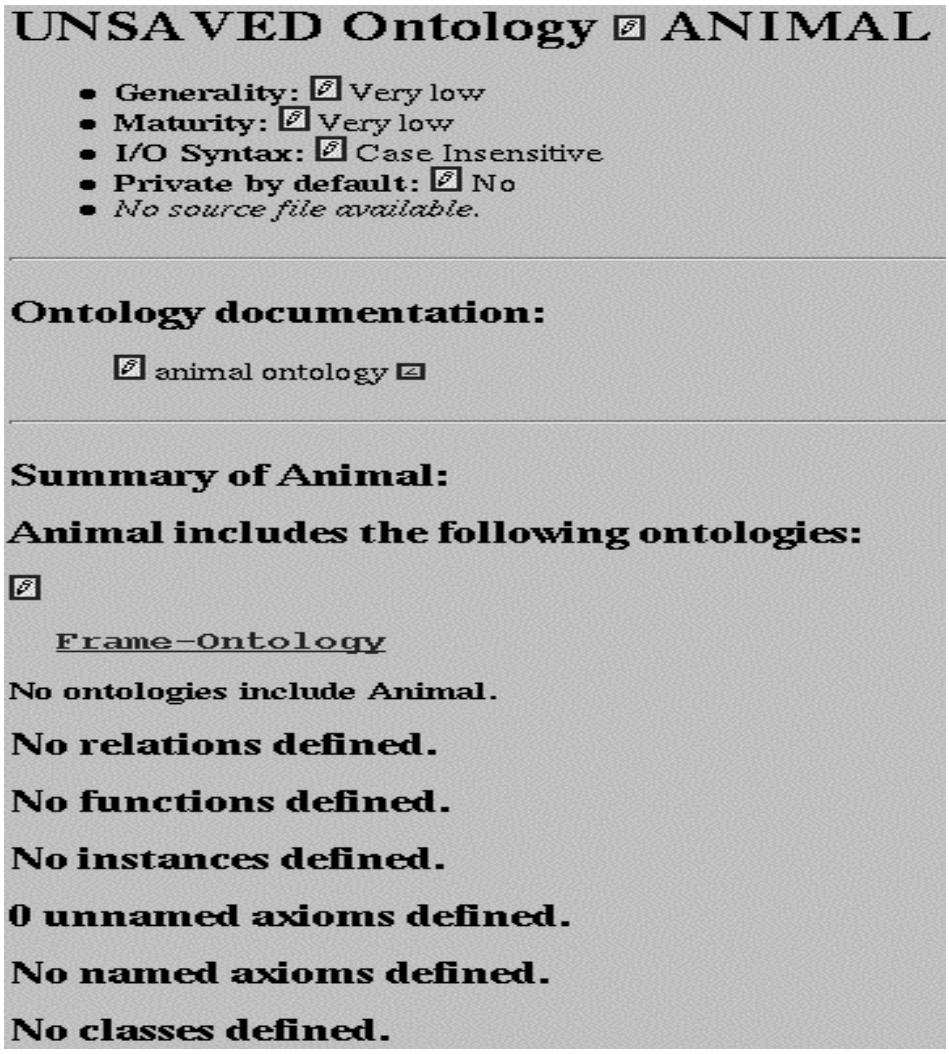
a. Structure des écrans

Chaque écran type de Ontolingua est décomposé en deux sous-écrans (ou «frame»):

- La première partie présente l'ensemble des services communs à tous les écrans (retour accueil, aide, préférences, quitter, ...) . On y trouve également une partie des services de modification spécifiques à l'entité que l'on consulte.



- La seconde partie est le corps principal de chaque écran. On y trouve la présentation de l'entité observée. Lorsque l'on se trouve en mode d'édition, des icônes apparaissent dans le texte et permettent d'accéder à d'autres fonctions de modification que celles présentent dans la partie haute.



b. Bonnes idées

- L'accès aux fonctions d'édition est inclus à l'interface de consultation par l'intermédiaire d'une icône permettent le va-et-vient entre les deux modes (édition/consultation). De cette façon, l'utilisateur n'a pas à apprendre deux interfaces différentes.

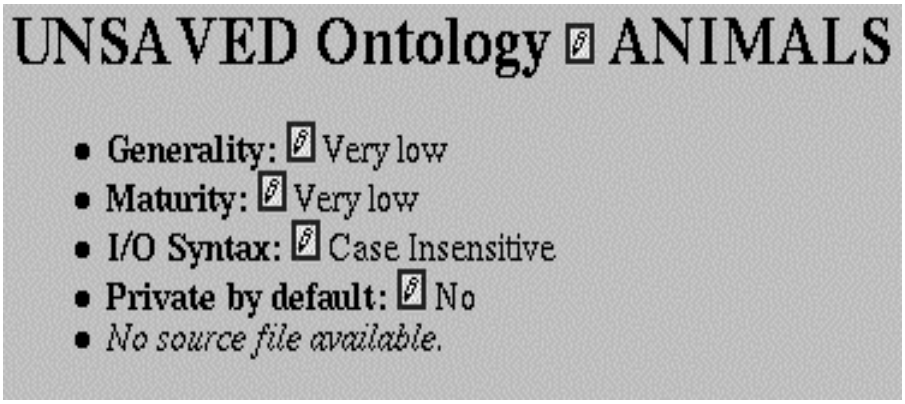
- La possibilité d'annuler plusieurs actions (icône «undo»): lorsqu'un utilisateur s'aperçoit qu'il vient de commettre quelques erreurs, le bouton «UNDO» lui permet de faire machine arrière simplement.

- Accès à une aide en ligne dans tous les écrans: chaque écran présente un bouton permettant l'accès à une aide concernant l'entité affichée.

c. *Améliorations*

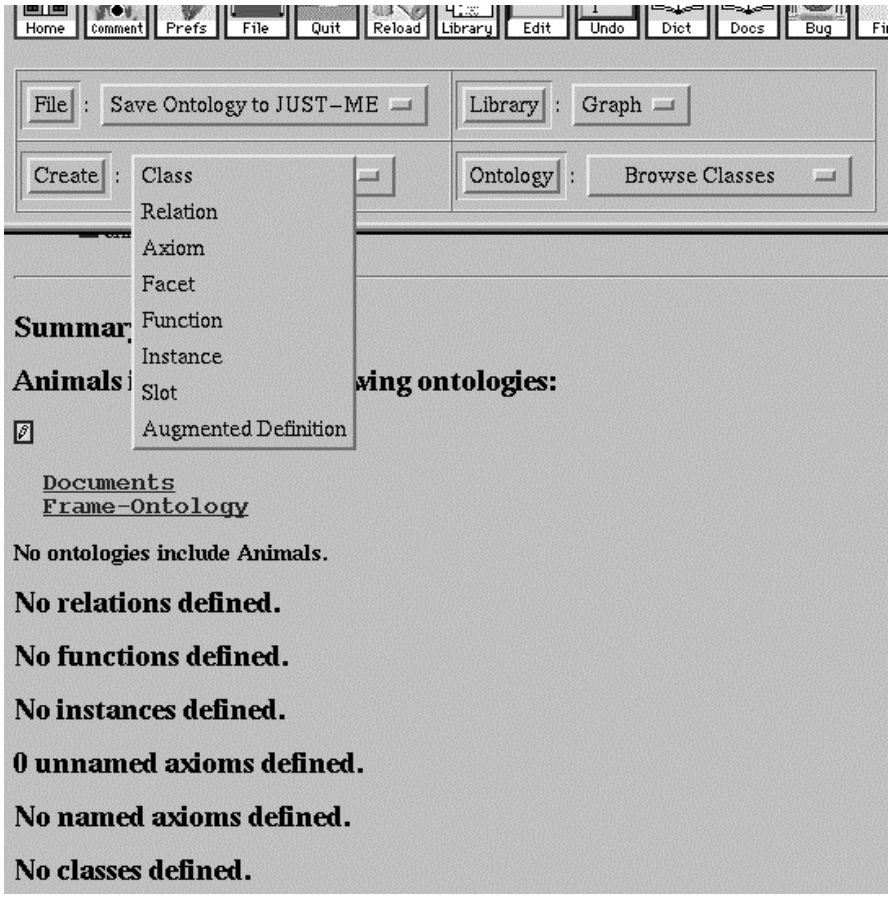
Quelques améliorations seraient souhaitables, principalement au niveau de l'aspect de l'interface, mais également au niveau de l'ergonomie.

- Certaines modifications d'ordre général pourraient être regroupées en un seul masque de saisie plutôt que de proposer une modification pour chaque caractéristique. Chaque modification donne lieu à une soumission d'URL, ce qui est coûteux. Il conviendrait donc de regrouper certains éléments pour améliorer les performances de l'application.

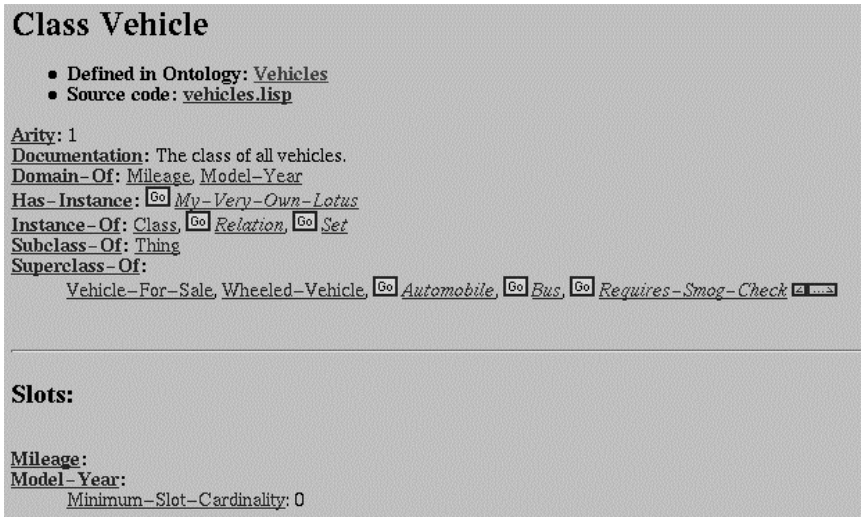


Chaque icône en forme de crayon permet de modifier la valeur (à droite du crayon) d'une variable du système (à gauche du crayon). Une seule icône permettant la modification de toutes ces variables serait souhaitable.

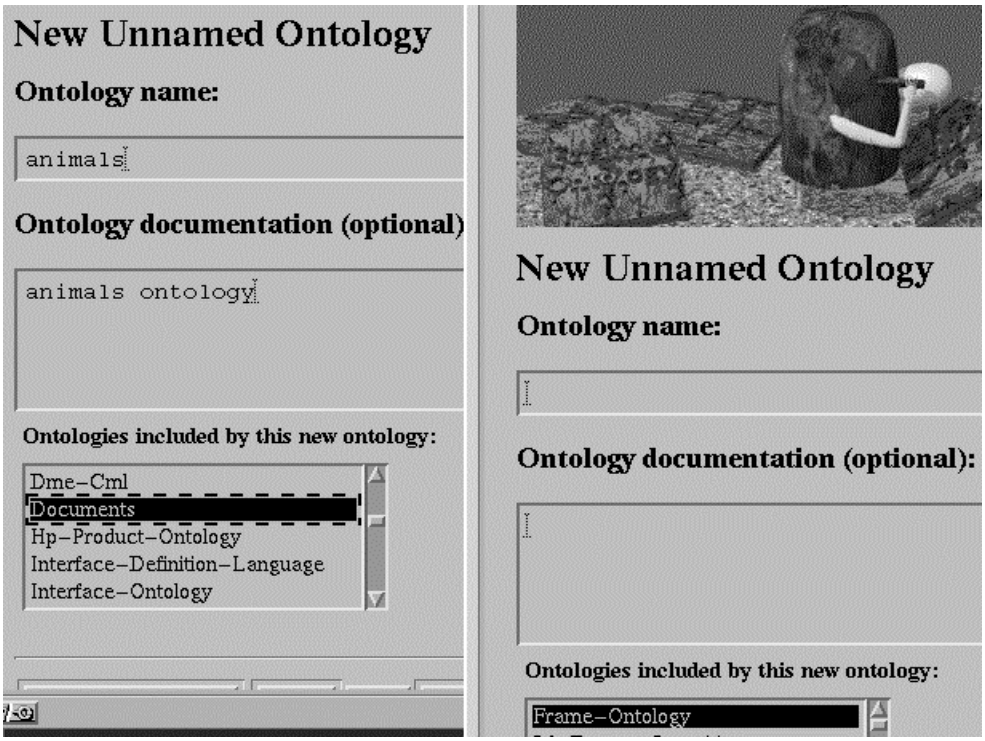
- Certaines fonctions d'édition sont accessibles dans le bandeau du haut et d'autres dans le corps du texte, sans qu'une certaine logique ne semble être respectée, ce qui nuit à la cohérence de l'interface d'édition.



- Certains écrans sont trop chargés de liens HTML, ou mal présentés, ce qui rend très confus la navigation.



- L'utilisation des possibilités de «BACK» (retour à l'écran précèdent) et «FORWARD» (passage à l'écran suivant) des navigateurs Web efface les écrans de saisie. De ce fait lorsque l'on valide un écran de saisie et qu'une erreur se produit, il faut de nouveau tout ressaisir.



5. Conclusion

Ontolingua, comme Hytropes, est donc un outil permettant de visualiser un domaine de connaissances par l'intermédiaire du Word Wide Web. Par rapport à Hytropes, les possibilités d'édition ont déjà été implémentées dans Ontolingua. Après quelques manipulations d'ontologies à l'aide de cette interface il apparaît quelques bonnes idées tel que le passage du mode de consultation au mode d'édition par un simple bouton, l'aide en ligne, la possibilité d'annuler un certain nombre de modifications. En revanche la présentation à l'écran des différents éléments et services est un peu confuse.

IV. Études

Les logiciels Hytropes et Ontolingua ont été présentés dans le chapitre précédent, il en résulte quelques idées utiles à la réalisation de notre interface. Mais avant de passer à ce stade il convient tout d'abord de voir les différentes possibilités d'implémentation des nouvelles fonctionnalités d'édition au logiciel de consultation actuel. Ce chapitre s'intéressera donc à cet aspect ainsi qu'à celui de savoir comment réaliser ces nouvelles fonctionnalités.

D'autre part, la modification de bases de connaissances Tropes à l'aide de Hytropes soulève un certain nombre de problèmes nouveaux par rapport à la simple consultation de bases:

- Les premiers problèmes, traditionnels aux applications de consultation et modification de données par réseau, concernent le contrôle d'accès et la concurrence d'accès aux données.

- Un autre problème concerne la gestion des erreurs. En effet dans le cadre de la seule consultation l'utilisateur est passif, il se contente d'observer la base de connaissances, mais en aucun cas il ne peut la rendre inconsistante. Aucune gestion d'erreur n'a donc été implémentée spécifiquement à Hytropes, ce qui ne convient pas si l'utilisateur à la possibilité d'enrichir une base de connaissances.

Il sera également présenté ici l'étude des différentes solutions envisageables pour résoudre de telles contraintes et lorsque cela le permet une option de réalisation est avancée.

A. L'interface

1. Généralités

a. Introduction

La mise en place d'une interface d'édition de bases de connaissances, lorsqu'une interface de consultation existe déjà, laisse a priori deux possibilités d'implémentation :

- Définir une nouvelle interface, propre à l'édition
- Utiliser l'interface de consultation comme support à l'édition

Plusieurs arguments me permettent de choisir la seconde solution, je vais les évoquer ici.

b. Similarité entre interface de consultation et structure Tropes

L'interface de consultation existante, présentée dans le second chapitre est très proche de la structure physique d'une base de connaissances Tropes. De ce fait, consulter une base de connaissances Tropes à l'aide de cet outil revient à naviguer dans sa structure. La page d'accueil d'une base présente les différents concepts définis et les liens permettant d'y accéder. La sélection de l'un de ces liens permet de visualiser la structure du concept choisi, à savoir l'ensemble des attributs, des points de vue, des passerelles et instances définies. Des liens sont également disponibles pour visualiser toutes ces entités...

Il apparaît particulièrement intéressant d'utiliser une interface permettant la navigation dans une structure afin justement de faire évoluer celle-ci.

c. *Facilités pour l'utilisateur*

Les moyens d'accès à l'édition étant intégrés à l'interface de consultation, l'utilisateur visualisant une base, n'aura pas à sortir de son application pour effectuer des modifications. Si ce n'était pas le cas, il devrait ouvrir une autre application, accéder au service voulu par le biais d'une nouvelle interface qu'il aurait besoin d'apprendre, effectuer ses modifications, puis vérifier le résultat à l'aide de l'outil de consultation.

Ne pas avoir à lancer d'application spécifique et ne pas dérouter l'utilisateur en lui faisant apprendre deux interfaces différentes sont deux facteurs importants.

d. *Réutilisation du code*

D'un point de vue purement informatique, il serait dommage d'avoir à réécrire des programmes similaires à ceux déjà développés pour la consultation. De plus, si le besoin d'écrire une interface spécifique à l'édition était nécessaire, cela mettrait en évidence de grosses lacunes concernant l'interface de consultation. Les biologistes du LGPD[10] de Marseille utilisent Hytropes dans sa version de consultation depuis plus d'un an et en sont satisfaits. La réutilisation du code déjà produit peut être une solution à retenir.

Ne pas refaire ce qui existe déjà.

e. *Conclusion*

La définition d'une nouvelle interface d'édition ou l'ajout de fonctionnalités d'édition à l'interface de consultation était deux orientations possibles pour réaliser mon travail. Après l'étude des spécificités de l'interface de consultation, cette dernière parait être le meilleur support pour accéder à des fonctionnalités d'édition, d'autant plus que l'utilisateur n'aura pas à apprendre deux interfaces différentes et du point de vue informatique elle minimise les développements.

[10]Laboratoire de Génétique et Physiologie du Développement

2. Problèmes posés par l'utilisation du Web

L'utilisation du Web comme support d'application entraîne un certain nombre de nouvelles contraintes par rapport à une application traditionnelle. Il faudra donc en tenir compte pour pouvoir réaliser l'interface d'édition.

a. Contraintes liées à HTML et HTTP

URLs

Pour transmettre une information du navigateur Web au serveur afin qu'il agisse en conséquence, une seule solution: soumettre un URL. Cette action ne pouvant être déclenchée que par l'utilisateur, il est par exemple impossible au système de déclencher automatiquement un message d'information conseillant la sauvegarde du travail ou tout autre action de ce genre. HTML n'offre que deux moyens de soumettre un URL: les liens et les formulaires.

- Un lien permet de soumettre un URL fixe, c'est-à-dire que le contenu de l'URL est déterminé dans la conception du document présenté et aucune action de l'utilisateur ne peut le modifier. Lorsque j'ai détaillé l'interface actuelle de Hytropes, les liens permettent par exemple de faire en sorte de visualiser un concept lorsqu'on le sélectionne avec la souris dans la liste des concepts d'une base.

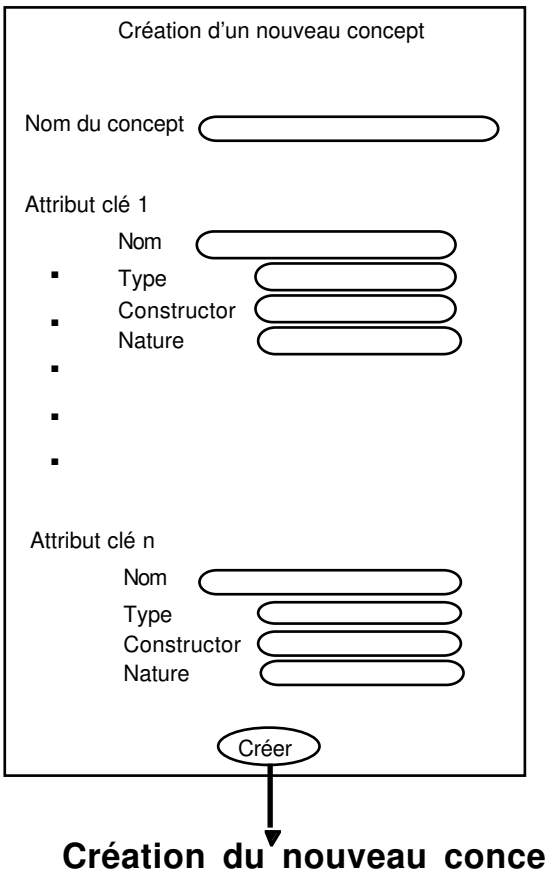
- Un bouton, placé dans un formulaire de saisie permet d'ajouter les différentes valeurs des champs saisis par l'utilisateur à une base déterminée d'un URL. De cette manière, il est possible d'intégrer des données utilisateurs et les soumettre au serveur. Dans Hytropes, les boutons sont notamment utilisés dans le mécanisme de filtrage afin que l'utilisateur puisse saisir les valeurs des différents attributs du concept.

Ces deux moyens ne permettent pas de construire une interface très élaborée, notamment dans le cadre de l'édition de base.

Exemple:

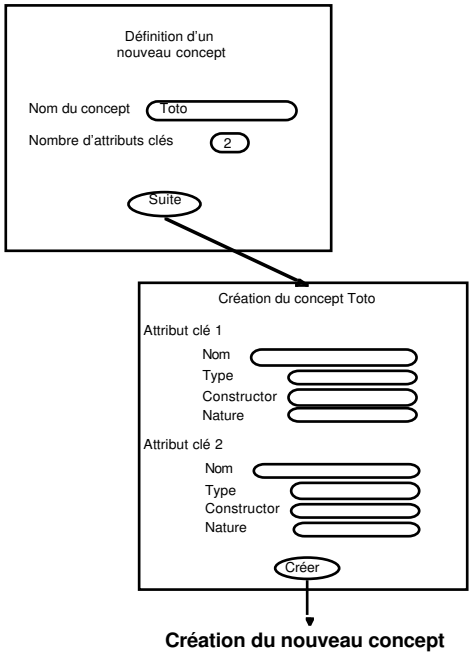
Prenons l'exemple de la création d'un concept, L'API nécessite de connaître le nom du concept à créer et l'ensemble des attributs clé de celui-ci. Le problème de cette fonctionnalité est que le nombre d'attributs clé n'est a priori pas connu, aussi le nombre de champs de saisie des attributs clé est variable.

- Une première interface pourrait présenter un formulaire permettant de saisir le nom du concept et «n» descriptions d'attributs, «n» étant le nombre d'attributs maximal autorisé par le système.



Création du nouveau conce

- Une autre interface possible est de présenter un premier écran contenant un formulaire permettant la saisie du nom du concept et du nombre d'attribut clé «n» à prévoir. Le bouton de ce premier écran renvoie alors vers un second écran présentant un formulaire de saisie des caractéristiques des «n» attributs clé, un bouton permet alors de valider les choix et de créer le concept défini.

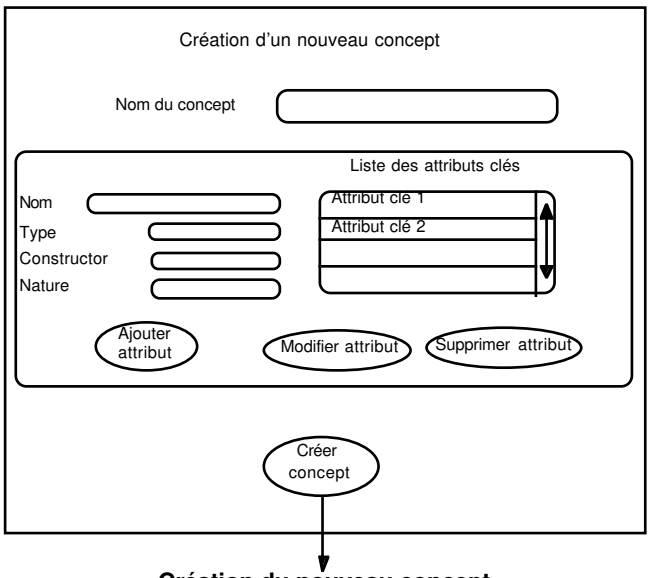


Création du nouveau concept

Évolutions:

L'apparition récente du langage de programmation Java et de Javascript notamment, ont considérablement accru les possibilités d'interaction de l'utilisateur avec son application. Leur reconnaissance aujourd'hui par de nombreux navigateurs Web assure une bonne exécution sur la plupart des plates-formes.

- Java permet d'une part de bénéficier d'une interface graphique pouvant être relativement élaborée. Il permet également, moyennant le pré-chargement des données nécessaires, de simuler l'interaction entre le navigateur Web et le serveur. J'ai déjà eu l'occasion de mettre en valeur l'utilisation de Java dans certains écrans de Hytropes. Pour reprendre l'exemple de la création d'un concept dans le cadre d'une extension de Hytropes aux possibilités d'édition, il est possible de créer un programme (ou plutôt une «Applet») dans lequel l'utilisateur aura la possibilité d'ajouter, modifier ou même supprimer une description d'attribut clé directement à l'écran, sans se soucier du nombre d'attributs dont il aura besoin.



Création du nouveau concept

  - Pour ajouter un attribut clé, saisir sa définition (nom, type, constructor, nature) puis choisir le bouton «Ajouter attribut»

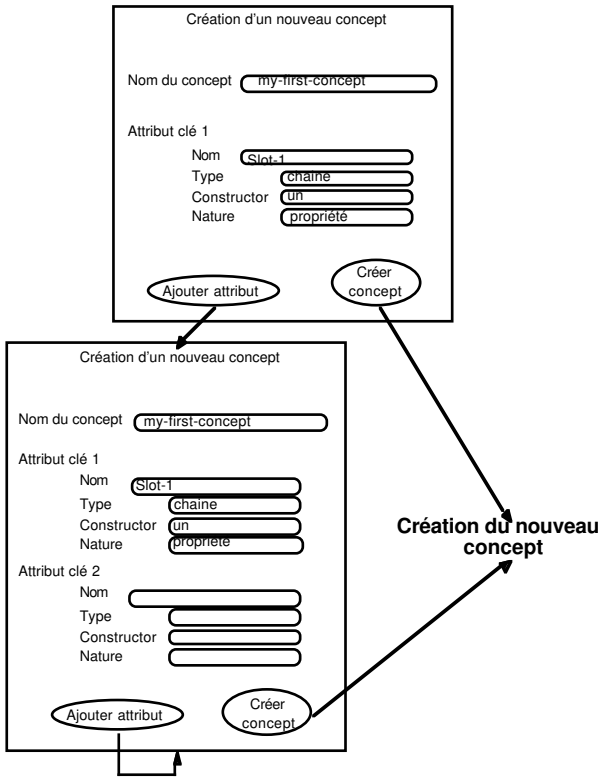
  - Pour modifier un attribut, le sélectionner dans la liste, modifier sa définition puis choisir le bouton «Modifier attribut»

  - Pour supprimer un attribut, le sélectionner dans la liste puis choisir le bouton «Supprimer attribut»

- Javascript quant à lui étend les possibilités de génération d'URL à toutes les actions que peut effectuer l'utilisateur sur son écran (sélection dans une liste, frappe d'un texte dans une zone de saisie, ...), il permet également de positionner

plusieurs boutons par formulaire de saisie, chacun ayant sa propre fonctionnalité. Aucune utilisation de Javascript n'a encore été faite dans Hytropes.

L'exemple de la création de concept, développé avec l'utilisation de Javascript pourrait présenter un écran permettant la saisie du nom du concept et du premier attribut clé. Un bouton permettrait d'ajouter un attribut clé et un autre permettrait la création du concept.



Interface de création d'un concept à l'aide de Javascript

HTTP: protocole sans état

Autre point lié à HTTP, ce dernier est dit protocole sans état. Cela signifie que d'une part il n'est pas capable de mémoriser le contexte d'un écran à un autre: tous les arguments nécessaires au traitement doivent figurer dans l'URL. D'autre part après affichage d'un document HTML, la connexion entre le serveur et le navigateur Web est coupée et chaque soumission d'URL donne lieu à une ouverture d'une nouvelle connexion au serveur de l'application. Cette contrainte agit directement sur les performances de l'application. Il est donc préférable d'éviter les écrans intermédiaires (donnant lieu à autant de connexions au serveur) avant de pouvoir exécuter une fonctionnalité de l'API.

b. *Le navigateur Web*

Une des multiples possibilités des navigateurs Web est de garder en mémoire les documents HTML visités par l'utilisateur pendant sa session de travail. Ceci lui permet ainsi de visualiser à nouveau un document sans pour cela établir une nouvelle connexion avec le serveur qui le lui a fourni. Dans un environnement qui évolue tel qu'un environnement d'édition de bases de connaissances, cette possibilité peut provoquer des incohérences. Supposons qu'un utilisateur affiche de nouveau un document dans lequel un lien n'est plus valide (car l'entité Tropes désignée a été supprimé) et imaginons que ce lien soit justement sélectionné...

c. *Conclusion*

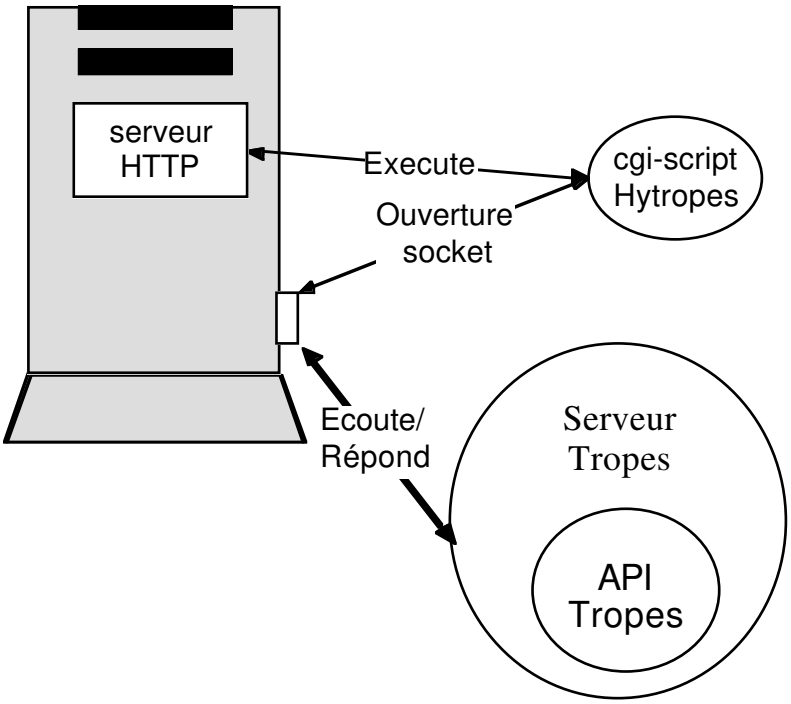
L'étude concernant les problèmes posés par la définition d'une interface d'édition de bases de connaissances nous permet de retenir quelques choix quant à la réalisation de cette interface. Tout d'abord, l'existence d'une interface de visualisation satisfaisante amène à penser qu'il est nécessaire d'intégrer les nouvelles fonctionnalités d'édition à l'interface de consultation et non d'en définir une toute nouvelle. L'utilisateur retrouvera ainsi la même convivialité pour éditer ou consulter ses bases, de plus le code déjà produit pour la consultation peut être récupéré. D'autre part, l'étude des contraintes liées à l'utilisation du Web (et en particulier HTTP et HTML) laisse penser que la réalisation d'une fonctionnalité d'édition devra donner lieu à un minimum d'écrans intermédiaires. L'utilisation de Java notamment, pourra nous y aider.

B. Le contrôle d'accès

Lorsque l'on parle de visualisation et de modification de données dans un environnement réseau, le premier aspect nous venant à l'esprit est celui du contrôle d'accès à ces données. L'objectif de mon projet est de pouvoir accéder et modifier des bases de connaissances depuis n'importe quelle machine ayant une connexion Internet, cependant n'importe qui ne doit pas pouvoir le faire. Ce paragraphe est dédié à l'étude de ce qui peut être mis en place pour protéger l'accès aux données de Hytropes, il présente également l'option que nous avons finalement retenue. Il ne traitera pas, en revanche, de la sécurité sur Internet en général qui est bel et bien un sujet d'actualité, mais n'influant pas sur notre politique de sécurisation, si ce n'est le choix de rendre accessible ou non telle ou telle connaissance via Internet.

L'architecture réseau de Hytropes laisse entrevoir la possibilité d'agir à quatre niveaux différents :

- serveur HTTP
- cgi-script Hytropes
- serveur Tropes
- API Tropes



Rappel de l'architecture

1. Le serveur HTTP

Le premier niveau d'action pour la sécurité et le contrôle d'accès est le serveur HTTP. La plupart des serveurs de ce type prévoient en effet de façon standard la possibilité d'effectuer un contrôle d'accès dans leurs nombreux paramétrages. Il est alors possible de conditionner l'accès à telle ou telle URL ou bien tels ou tels fichiers du serveur. Deux méthodes complémentaires permettent la mise en place d'une telle sécurité :

- Contrôle d'accès par adresse IP[11]

Toutes les machines reliées à un réseau ont une adresse permettant d'acheminer les informations d'une machine vers une autre. Dans le cas du réseau Internet nous concernant, il s'agit de l'adresse IP. Le paramétrage d'un serveur HTTP permet au minimum le filtrage d'accès en fonction de cette adresse. Il est alors possible d'interdire l'accès d'une ou plusieurs machines, ou bien et c'est préférable dans un environnement comme Internet, autoriser l'accès à telles ou telles plages d'adresses seulement. Ce niveau de sécurité peut s'avérer très intéressant lorsque l'ensemble des postes de travail des utilisateurs autorisés est bien identifié. En revanche si ce n'est pas le cas, il ne peut être mis en place. Autre limite à ce mécanisme, l'accès à Internet via un prestataire de service comme AOL[12] ou Compuserve ou autres: chaque utilisateur ne possède dans ce cas pas sa propre adresse IP. C'est en effet à chaque connexion qu'une adresse IP lui est fournie par son prestataire. Un utilisateur peut donc avoir une adresse IP différente à chaque connexion, il n'est alors pas possible de l'identifier par ce biais.

- Contrôle d'accès par mot de passe

Lorsqu'un utilisateur souhaite accéder à une ressource protégée, il est alors prié de s'identifier par un nom d'utilisateur et un mot de passe. L'authentification de l'utilisateur une fois effectuée et validée par le serveur, celui-ci contrôle l'autorisation ou non de l'accès à la requête souhaitée par l'utilisateur.

Avantages:

- Solution simple à mettre en œuvre puisque prévue en standard pour la plupart des serveurs HTTP. Aucun développement ni gestion de contrôle d'accès spécifique à Tropes ou Hytropes n'est nécessaire, il n'y a qu'à paramétrer les fichiers de configuration du serveur.

- Solution sécurisante puisqu'elle agit au premier niveau de l'architecture réseau, c'est-à-dire la connexion au serveur.

- Solution souple dans le sens où la gestion peut être paramétrée finement (URL, groupe d'URLs, fichiers, répertoires).

[11]Internet Protocol
[12]America On Line

Inconvénients:

- La mise en place d'un tel contrôle ralentit les temps de réponse des requêtes des utilisateurs. HTTP est en effet un protocole sans état, c'est-à-dire qu'il ne peut mémoriser d'informations sur une requête lorsque celle-ci est terminée. Il y a donc de nombreux échanges entre le serveur HTTP et le navigateur Web, c'est d'ailleurs à la charge de ce dernier qu'incombe la mémorisation des données concernant l'identification des utilisateurs. Lorsqu'une ressource protégée est accédée, le serveur HTTP renvoie alors une demande d'identification à laquelle le navigateur Web répond s'il connaît l'ensemble des informations requises (nom d'utilisateur et mot de passe pour la ressource requise). Si le navigateur n'en a pas la connaissance, il demande alors à l'utilisateur de s'identifier puis renvoie le résultat de l'identification au serveur qui renvoie ou non l'information demandée en fonction des autorisations mises en place.

- Le second inconvénient est directement tiré de la remarque précédente. C'est au navigateur Web de mémoriser les identifications, il faut donc que celui-ci implémente également cette fonctionnalité de HTTP. J'ai effectué des tests avec succès pour Netscape Navigator version 2 et version 3 ainsi qu'avec Internet Explorer ce qui laisse présager que la sécurité d'accès via HTTP est implémentée dans la plupart des navigateurs, même assez ancien.

- Pour un même serveur Hytropes contenant plusieurs bases de connaissances, si l'on souhaite gérer des autorisations différentes d'une base à l'autre, le paramétrage peut devenir vite fastidieux.

2. Le cgi-script Hytropes

Le cgi-script Hytropes est second maillon de l'architecture réseau de Hytropes. C'est un passage obligatoire pour pouvoir accéder aux ressources de Tropes, il peut donc être intéressant d'intervenir à ce niveau pour intégrer un contrôle d'accès. Cependant, tout comme HTTP, il ne peut retenir d'informations d'une requête à une autre puisqu'il se trouve exécuté à chaque fois au titre d'une nouvelle connexion au serveur HTTP. Il faut donc transporter les informations d'identification par le biais même de l'URL. Le schéma type d'une racine d'URL Hytropes pourrait donc être:

«http://nom.serveur.http/cgi-bin/hytropes/**identification/**»

De cette manière, l'utilisateur aurait à sa disposition un premier écran lui permettant de s'identifier par mot de passe par exemple. Une fois l'identification effectuée et validée par le script, ce dernier assure le relais au générateur de page HTML sans oublier de lui signaler le code d'identification. Ce code sera en effet nécessaire pour la génération des liens HTML valides vers d'autres entités Tropes sans qu'une identification ne doive être effectuée de nouveau.

Scénario type d'un accès au serveur

- L'utilisateur accède pour la première fois au serveur, il s'identifie. Une page HTML présentant l'ensemble des différentes bases de connaissances disponibles sur le serveur est proposée. Les liens HTML permettant l'accès à ces bases incluent l'identificateur saisi dans leurs racines.

- L'utilisateur choisit d'accéder à une certaine base disponible. Le script vérifie alors l'existence de l'identificateur dans sa propre base de données. Il vérifie également la bonne concordance de l'identificateur avec le service demandé (contenu dans le reste de l'URL), ici la visualisation de la page HTML représentant la base de connaissances choisie.

    - Si l'identificateur est valide pour le service demandé, la requête est exécutée

    - Si l'identificateur n'est pas valide, une nouvelle identification est demandée.

- Le contrôle d'accès est effectué de la même façon que dans le second point pour tous les services suivants demandés à Hytropes.

Avantages:

- Solution relativement simple à mettre en œuvre

- Ne dépend pas du navigateur Web ni du serveur HTTP excepté le fait que ce dernier puisse exécuter des scripts CGI, mais dans ce cas ce serait le fonctionnement même de Hytropes qui serait remis en cause.

- affecte moins les performances que la solution précédente

Inconvénients:

- Peu sécurisante puisque l'identification apparaît dans l'URL. Il suffit même simplement de stocker un URL valide pour pouvoir par la suite accéder aux services de Hytropes sans avoir besoin de s'identifier. Une démonstration du serveur Hytropes lors d'une conférence par exemple, l'URL est stockée automatiquement dans les archives du navigateur et de fil en aiguille l'URL valide est diffusée... (méthode similaire à celle employée dans Ontolingua)

3. Le serveur Tropes

Une solution équivalente à la précédente peut être développée à partir du serveur Tropes (trpserver). Elle présente les mêmes caractéristiques, avantages et inconvénients que la méthode précédente. Toutefois, on peut noter que le contrôle s'effectue un peu plus loin dans la chaîne de l'architecture réseau de Hytropes, ce qui n'optimise pas les temps de réponse.

4. L'API de Tropes

Tropes ne contient pas de gestion de contrôle d'accès dans son API. Une dernière solution serait donc d'en intégrer une, ce qui nécessiterait :

- Modifier Tropes pour qu'il intègre la notion de mot de passe et notamment fournir à son API les fonctions nécessaires à leur gestion.
  - création,
  - modification,
  - vérification…

- Modifier également le schéma des bases de connaissances Tropes et notamment d'inclure sous une forme cryptée les différents mots de passe que l'on aura défini.
  - droit de lecture,
  - droit d'écriture…

- Modifier l'ensemble de l'API de Tropes afin que chaque commande soit contrôlée par un mot de passe. Un contrôle de ce dernier est effectué et s'il n'est pas valide une erreur est renvoyée. Exemple pour la création d'une vue de concept:

  sans mot de passe -> (tr-create-conceptview concept conceptview-name)
  avec mot de passe -> (tr-create-concept password concept conceptview-name)

- Modifier la racine des URLs Hytropes afin d'y intégrer le mot de passe, comme pour les deux solutions précédentes:

  «http://nom.serveur.http/cgi-bin/hytropes/**mot de passe/**»

- Puisque la gestion des mots de passe est inclus dans l'API de Tropes, il est nécessaire de développer également les écrans de gestion de mots de passe pour Hytropes.

Scénario type d'un accès au serveur

- L'utilisateur accède pour la première fois au serveur, il s'identifie. Une page HTML présentant l'ensemble des différentes bases de connaissances disponibles sur le serveur est proposée. Les liens HTML permettant l'accès à ces bases incluent l'identificateur saisi dans leurs racines.

- L'utilisateur choisit d'accéder à une base disponible, l'API de Tropes est alors sollicitée. Si l'identificateur n'est pas valide pour cette base, un message d'erreur est mis en forme, puis présenté à l'écran au format HTML.

- Idem pour tous les autres services de Hytropes

Avantages:

- Tropes, sans Hytropes, contient désormais une gestion de contrôle d'accès

- Un mot de passe est lié à une base et non plus au seul service. Il est donc plus facile par la suite de charger plusieurs bases de connaissances sur un seul et même serveur et de gérer leur contrôle d'accès de façon indépendante pour chacune d'entre elle.

Inconvénients:

- De nombreuses modifications sont nécessaires au niveau de l'API de Tropes.

- La présence en clair du mot de passe dans l'URL n'est pas très sécurisante, comme pour les deux solutions précédentes.

- Une page HTML met en œuvre plusieurs fonctions de l'API, une autorisation de prestation Hytropes n'est donc pas équivalente à une autorisation sur une fonction de l'API, mais de toutes celles qui la compose. Contrôler la validité d'une page HTML revient donc à vérifier une à une toutes les fonctions de l'API mis en œuvre avant d'être affichée alors qu'il serait souhaitable de pouvoir dire en un seul contrôle: telle page HTML peut être affichée et telle autre ne le peut pas.

5. Conclusion

Je viens de présenter ci-dessus les quatre possibilités de protection de Hytropes. Parmi celles-ci et à la vue des avantages et inconvénients de chacune des méthodes, l'utilisation des modules de sécurité d'accès du serveur HTTP est la solution la plus sécurisante. Le ralentissement des temps de réponse n'est pas un facteur déterminant pour ce genre d'application et la norme HTTP (voir glossaire) est en passe d'évoluer vers de meilleures performances ce qui laisse présager la suppression très prochaine de cet inconvénient. Autre point en faveur de cette méthode, on peut facilement imaginer la présence d'une seule base de connaissances par serveur Hytropes, l'inconvénient du paramétrage fastidieux est alors éliminé dans ce cas. Pour finir, la protection de serveur par ce biais semble être bien implémentée dans la plupart des navigateurs et serveurs HTTP, il apparaît donc difficile d'être obligé de choisir un serveur HTTP ou un navigateur Web incompatible avec ce mécanisme de contrôle d'accès.

Quant aux autres possibilités, la présence du mot de passe dans l'URL n'est vraiment pas très sécurisante, elles requièrent toutes des développements plus ou moins importants qu'il faudra maintenir. Intégrer un contrôle d'accès à Tropes n'est pas un point critique, les protections des systèmes d'exploitation tel que Unix ou autres sont suffisants pour une utilisation locale sans Hytropes.

C. La concurrence d'accès aux données

Le problème de la concurrence d'accès n'est pas à proprement dit un point nouveau dû à l'apport de la possibilité d'édition de bases de connaissances. Cependant, il s'agit ici de vérifier que toute action de modification d'une base ne sera pas interrompue par tout traitement d'un autre utilisateur. Si cela se produisait, la base ainsi modifiée risquerait de se retrouver incohérente ou plus vraisemblablement le serveur Hytropes finirait par s'écrouler.

1. Exécution séquentielle des tâches

Il s'agit de vérifier ici que l'architecture même d'Hytropes assure un bon déroulement des opérations, de sorte que si deux personnes soumettent en même temps une modification d'un même objet, les demandes seront en fait exécutées complètement l'une derrière l'autre.

- Lorsqu'un URL est soumis au système, l'exécution du cgi-script Hytropes établit une connexion avec un port particulier de la machine sur lequel se trouve le serveur Tropes et y dépose une partie de l'URL (ouverture de socket).

- Lorsque le serveur Tropes a détecté cette connexion, il interprète l'URL déposée, construit le résultat puis renvoie ce dernier vers le script qui clos la connexion (fermeture de socket).

- Le résultat est alors renvoyé vers le serveur HTTP puis finalement vers l'écran de l'utilisateur

Conclusion:

Si une ouverture de socket occupe un port d'une machine, il n'est pas possible d'en ouvrir une seconde. Cette dernière se met donc en attente jusqu'à ce que la première connexion soit close. Il ne peut donc pas y avoir de problème de concurrence d'accès avec une telle architecture, aussi aucune réalisation particulière n'est nécessaire sur ce point.

2. Confusion sur l'interprétation des résultats

Le point précédent montre que le système ne va pas s'écrouler si plusieurs personnes essayent de modifier le même objet en même temps. Cependant certaines confusions, dues à la possibilité de modification, peuvent être induites. Supposons par exemple que deux utilisateurs souhaitent, en même temps, changer le prix de location d'un appartement:

- L'écran de modification de départ est identique pour tous les deux.

- L'un saisit la somme de 3000 Francs puis valide ce choix, il se retrouve avec un écran résultat conforme à sa requête avec notamment un prix de location de 3000 Francs.

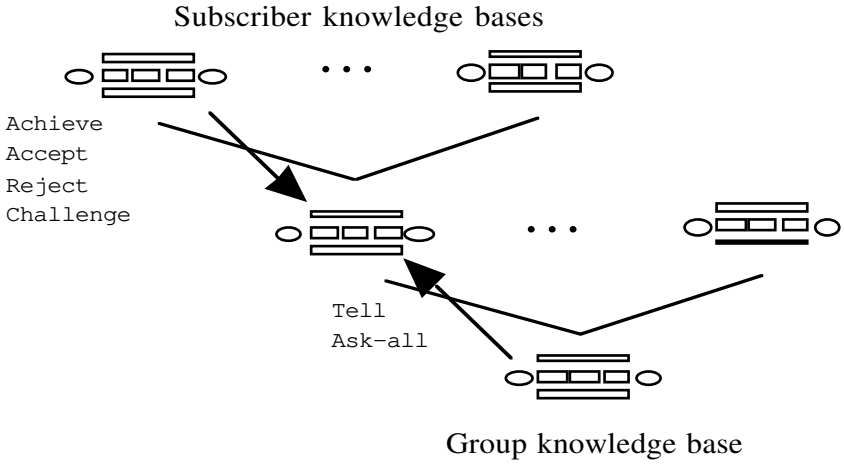
- L'autre utilisateur saisit 3500 Francs puis valide dans le même temps que l'autre, il a également un écran résultat conforme à sa requête, location de 3500 Francs.

=> La question posée est donc la suivante: Quel est le prix de location pour cet appartement ? Seule une nouvelle interrogation de la fiche de cet appartement pourra nous le dire, encore faut-il que quelqu'un ne la modifie pas dans la seconde qui suit.

Conclusion:

Ce problème est connu de l'équipe Sherpa, aussi le protocole CO4 (pour collaborative construction of consensual knowledge) spécifié par Jérôme Euzenat [Euzenat 95] doit permettre de le résoudre.

Très brièvement il s'agit d'un protocole permettant de valider au sein d'un groupe des modifications sur des bases de connaissances. Le but étant d'obtenir une connaissance acceptable par tout le groupe. La connaissance est éclatée en plusieurs bases locales (éditable par une seule personne, voire un petit groupe de trois ou quatre personnes maximum) dans lesquelles les modifications sont libres. L'ensemble des modifications apportées à cette base est ensuite soumis pour acceptation ou refus à une base plus générale concernant un groupe plus important de personnes et ainsi de suite. Je n'en dirais pas plus à ce sujet qui représente un projet de recherche autour de Tropes et de la construction coopérative de bases de connaissances en général.



Le protocole CO4
d'édition coopérative de bases de connaissances

3. Conclusion

En ce qui concerne la concurrence d'accès, l'architecture réseau déjà mise en place dans le serveur de consultation assure parfaitement son rôle. Aucune modification n'est donc nécessaire et je ne reviendrai pas sur ce point dans le chapitre «Réalisation». Les problèmes relatifs au travail en groupes ne sont quant à eux pas encore résolus, cependant le protocole CO4 en projet doit permettre de le faire. Ce point ne sera donc également pas débattu dans le chapitre «Réalisation».

D. La gestion des erreurs

Dans le cadre de la seule consultation de bases de connaissances, aucune gestion d'erreurs n'est nécessaire puisqu'aucune inconsistance ne peut être induite dans la base. Il n'en est pas de même lorsque l'on propose à l'utilisateur la possibilité de la modifier, aussi une gestion des erreurs spécifique à l'édition doit être implémentée à Hytropes. Deux types d'erreurs doivent être considérées.

1. Les erreurs de saisie

Elles comprennent:

- la saisie d'informations comportant une ou plusieurs erreurs syntaxiques
- la saisie de valeurs, correctes d'un point de vue syntaxique, mais incompatible avec la description de la base de connaissances ou valeurs inexistantes dans la structure de Tropes.

La gestion des erreurs intégrée à Tropes [Tropes] n'est ici pas applicable telle qu'elle. D'une part elle ne prévoit pas l'affichage du compte-rendu des erreurs au format HTML et d'autre part, plus grave, elle bloque le serveur indéfiniment si une erreur se produit. Une gestion spécifique des erreurs doit donc être développée pour Hytropes, elle sera détaillée dans le chapitre réalisation.

2. Les erreurs liées à l'utilisation du Web

a. Problème de l'existence des entités Tropes

Dans le cadre de l'interface de visualisation, tout lien vers une entité Tropes est forcément valide et aboutit à la visualisation de cette entité. Ce n'est plus le cas avec l'interface d'édition de bases, supposons par exemple la situation suivante:

- Un utilisateur «A» visualise la page d'accueil de la base «real-estate» contenant notamment un lien vers le «test».
- Un utilisateur «B» supprime le concept «test».
- L'utilisateur «A» sélectionne le lien vers le concept «test».

L'idéal aurait été de mettre à jour en temps réel l'écran de l'utilisateur «A» afin que celui-ci ne présente plus de lien vers le concept supprimé, mais dans un environnement HTTP, une modification d'affichage ne peut intervenir sans que l'utilisateur ne le réclame.

Toujours dans le même ordre d'idée, l'utilisation de la possibilité offerte par les navigateurs Web de revenir en arrière sans avoir à mettre à jour un document présenté auparavant (problème évoqué dans l'étude de l'interface) peut conduire au même résultat.

Deux solutions permettraient de résoudre ce type de problème:

- *Contrôler l'existence de l'entité Tropes à consulter*
  Lorsqu'un utilisateur sélectionne un lien vers une entité Tropes, la première chose à effectuer et de vérifier que cette dernière existe bien toujours. Si tel est le cas, le document HTML la concernant est engendré. Dans le cas contraire un message signale sa disparition et demande à l'utilisateur de réactualiser son écran.

- *Interdire la suppression de l'entité Tropes mise en cause*
  Lorsqu'un utilisateur regarde un document, l'ensemble des entités Tropes visualisées est mis de côté de sorte que la demande de suppression d'une de celles-ci soit interdite.

Conclusion:

La solution d'interdire la suppression de l'entité Tropes n'est pas réalisable car elle est beaucoup trop contraignante : blocage d'une action de modification pour une hypothétique consultation. D'autre part, elle n'est pas possible à mettre en œuvre dans une architecture HTTP. En effet, s'il est possible en générant le code HTML d'un document, de signaler un lien vers une entité Tropes, il n'est pas possible de faire machine arrière, c'est-à-dire libérer les ressources lorsque l'utilisateur visualise un autre document, ou même se déconnecte tout simplement du serveur. J'ai donc choisi de contrôler l'existence des entités Tropes sur lesquelles on souhaite agir avant de poursuivre le traitement.

b. Problème de la validité des URLs

Le fonctionnement théorique du serveur Hytropes suppose que l'utilisateur interagit dans son environnement à l'aide de liens HTML générés par l'application. Cependant, l'utilisateur a toujours la possibilité de saisir manuellement un URL, il risque alors d'introduire des erreurs dans la structure de celle-ci. L'affichage d'un message alertant de ce genre d'erreurs n'est pas implémenté dans Hytropes et il arrive parfois qu'un URL ayant une syntaxe incohérente avec la demande finisse par couper anormalement le serveur. Si dans le cadre d'un serveur de consultation, l'arrêt occasionnel et momentané du serveur n'est pas un problème majeur, dans celui de l'édition de bases il n'en est pas de même (perte de modifications...).

c. Problèmes liés au réseau

Pour conclure sur les différents types d'erreurs liées à l'utilisation du Web, il reste à traiter des erreurs dues à une perturbation de la connexion réseau. La plupart des navigateurs permettent d'interrompre une connexion HTTP à la demande de l'utilisateur, d'autre part les lignes réseaux, bien qu'étant de plus en plus fiables ne garantissent pas forcément une connexion sans histoire. Cette éventualité pose un sérieux problème, non résolu à ce jour.

En coupant la liaison HTTP (entre le serveur HTTP et le navigateur Web) lorsqu'une requête est en cours d'exécution l'utilisateur n'est certes plus bloqué, mais plus aucun accès au serveur Hytropes ne peut aboutir. L'explication est la suivante:

- la connexion établie par le cgi-script Hytropes avec le serveur Tropes reste en effet active après la coupure HTTP, elle bloque donc tous les accès à ce dernier. Il ne reste plus alors qu'à arrêter Hytropes brutalement ce qui provoque la perte des éventuelles modifications non sauvegardées.

3. Conclusion

Quatre types d'erreurs ont été déterminés dans cette étude. Une résolution spécifique a pu être trouvée pour trois d'entre elles, en voici la synthèse:

- Les erreurs de saisies utilisateurs ne peuvent être traitées par la gestion standard des erreurs de Tropes. La réalisation d'une méthode spécifique est donc nécessaire.
- Les erreurs dues à l'appel d'un document concernant une entité inexistante seront résolues par un contrôle systématique de l'existence de l'entité Tropes avant traitement.
- Aucune supposition ne devra être faite quant à la structure d'un URL, ceci afin de prendre en compte la possibilité de saisie manuelle de l'URL par l'utilisateur.

- Les erreurs réseaux restent le point faible de l'application, aucune solution n'a pu être avancée dans cette étude.

E. Conclusion

Le résultat des études concernant l'apport de la modification de bases de connaissances via Hytropes a permis de dégager deux idées majeures:

- Point positif de l'étude: à l'exception d'un problème, l'ensemble des besoins apportés par l'édition de bases (contrôle d'accès, concurrence d'accès aux données et gestion d'erreurs) peut être ou sera prochainement résolu (protocole CO4 pour le travail de groupe).

- Point négatif: La coupure volontaire ou intempestive du réseau provoque irrémédiablement un blocage du serveur sans autre issue que de l'arrêter brutalement. Dans le cadre de la simple consultation de bases, ce n'est pas très grave. Cependant dans celui de l'édition, les pertes de modifications engendrées peuvent être très fâcheuses.

Le bilan de l'étude est positif, d'autant plus que l'unique problème détecté a d'une part, de bonnes chances d'être résolu dans l'avenir et d'autre part que la perte de modifications dans ce cas peut être largement minimisée par une sauvegarde régulière du travail effectué. La réalisation peut donc commencer avec notamment la définition des URLs et de l'interface d'édition en tenant compte des remarques concernant les nouvelles contraintes liées à l'utilisation du Web.

V. Réalisation

L'étude a permis de dégager les idées permettant de résoudre les problèmes liés directement à l'apport de l'édition de bases de connaissances, il reste maintenant à mettre en œuvre celles-ci. Cette partie détaille ce qui a finalement été retenu pour la réalisation du projet. J'y évoquerai l'environnement informatique matériel et logiciel utilisé, la définition des URLs d'édition, la définition de l'interface utilisateur ainsi que la résolution des différents points évoqués dans l'étude.

A. Environnement informatique

Le développement de l'interface d'édition a été réalisé sur une station de travail SparcStation 4 de la société Sun pourvu du système d'exploitation Solaris 2.5 (Unix développé par Sun). Plusieurs outils et langages de programmation ont été mis à contribution, en voici un rapide descriptif.

1. Ilog Talk

Ilog Talk est un langage de programmation orienté objet développé par la société Ilog, anciennement projet de l'INRIA. Couplé à l'éditeur de texte Emacs, Talk présente un environnement programmation convivial contenant la plupart des possibilités attendues par ce type de produit: édition, simulation, compilation, recherche d'erreurs... Cet environnement, méconnu du grand public, est largement utilisé dans le monde de la recherche. Tropes, ainsi que la version de consultation de Hytropes ont d'ailleurs été développés avec celui-ci; c'est pourquoi les fonctions de génération de pages HTML et les traitements de l'interface d'édition l'ont été également.

2. Les navigateurs Web

- Netscape Navigator version 3.1 puis 4.03
- Microsoft Internet Explorer version 3.02
- Amaya (Editeur/Navigateur développé par le consortium W3C à l'INRIA)

3. Le JDK (Java Development Kit)

Le JDK fourni gracieusement par Sun permet d'offrir un environnement de programmation minimum afin de développer des applications ou des «Applets»[13] Java. La version 1.0 du JDK a été utilisée pour développer les «Applets» de l'interface d'édition, ceci aux dépens de sa version 1.1 car la compatibilité de cette dernière version dans les navigateurs existants, s'est effectuée très tardivement (à partir de la version 4.0 de Navigator seulement, 7 mois plus tard et avec des erreurs).

[13]Application présentée sur le Web à l'intérieur d'un document HTML

B. Définition des URLs

1. Syntaxe

J'ai choisi de prendre en compte deux critères principaux pour définir la syntaxe des URLs d'édition de bases de connaissances Tropes :

- Faire en sorte que les URLs soient suffisamment similaires à ceux déjà existants pour le serveur de consultation, ceci afin de profiter de toutes les fonctions de décodage et d'interprétation déjà écrites et opérationnelles dans ce dernier cadre.

  Le résultat obtenu indique une structure d'URL inchangée par rapport aux URL du serveur de consultation de bases Tropes, pour rappel :
  - une racine commune
  - un service Tropes à fournir
  - le composant Tropes sur lequel s'applique le service
  - d'éventuels paramètres dans le cas de traitement de formulaires de saisie

- Faire en sorte qu'ils soient suffisamment distincts des URLs du serveur de consultation, ceci pour deux raisons :

  - L'apport des services d'édition ne doit pas pénaliser les temps de réponse du serveur de consultation par de trop nombreux cas d'analyse.

  - Les services d'édition doivent pouvoir être sécurisés d'une façon distincte des services de consultation.

Pour différencier aisément «édition» et «consultation», la partie de l'URL représentant le service Tropes à fournir est simplement décomposée en deux parties:

- l'une est le mot-clé «edit» qui indique que c'est un URL d'édition

- l'autre représente réellement le service Tropes mis en œuvre

Quant au choix du nom du service Tropes à fournir, l'objectif est de rester le plus proche possible de l'API de Tropes (comme pour la consultation) afin que toute nouvelle possibilité de modification de l'API puisse aisément être implémentée dans Hytropes. Le nom du service à fournir contient donc, à de rares exceptions près, le nom de la fonction de l'API mis en œuvre.

résultat:

racine-commune/**edit**/nom-du-service/entité-tropes[?paramètres]

2. Organisation

Chaque service Tropes implémenté nécessite, en général 2 URLs :

- Le premier URL appelle l'écran de saisie des données à traiter (ou écran de confirmation pour les possibilités de suppression d'entités)

- La seconde permet d'effectuer le traitement demandé

Prenons l'exemple de la fonction de l'API Tropes «tr-create-conceptslot» permettant la création d'un attribut de concept, elle nécessite les URLs suivants :

- «.../edit/get-create-conceptslot/nom-base/nom-concept/nom-concept.html»

  provoque l'affichage d'un formulaire de saisie des données nécessaires à la création d'un attribut de concept, à savoir un nom, un type, une nature et une propriété. Ici, l'attribut de concept est à créer pour le concept de nom «nom-concept».

- «.../edit/create-conceptslot/base/concept/nom-concept.html?nom=«titi»&...»

  traite les données du formulaire de saisie, à savoir ici, crée un attribut de concept qui a pour nom «titi», ..., pour le concept de nom «nom-concept».

Remarque :

L'ensemble des URLs valides de l'interface d'édition est fourni en annexe 1.

C. Définition de l'interface

Comme nous l'avons vu dans la partie étude de l'interface, deux possibilités étaient offertes concernant le choix de l'implémentation des services d'édition à Hytropes. J'ai choisi de conserver l'interface de consultation comme moyen d'accès à l'édition, je présente ici le détail de la réalisation finalement retenue.

1. Accès aux fonctions d'édition

L'accès aux possibilités d'édition vient donc se greffer à l'interface de consultation, mais de quelle manière, dans quelles conditions et pourquoi de tels choix ?

a. Problématique

Trois types d'utilisations du serveur sont à considérer :

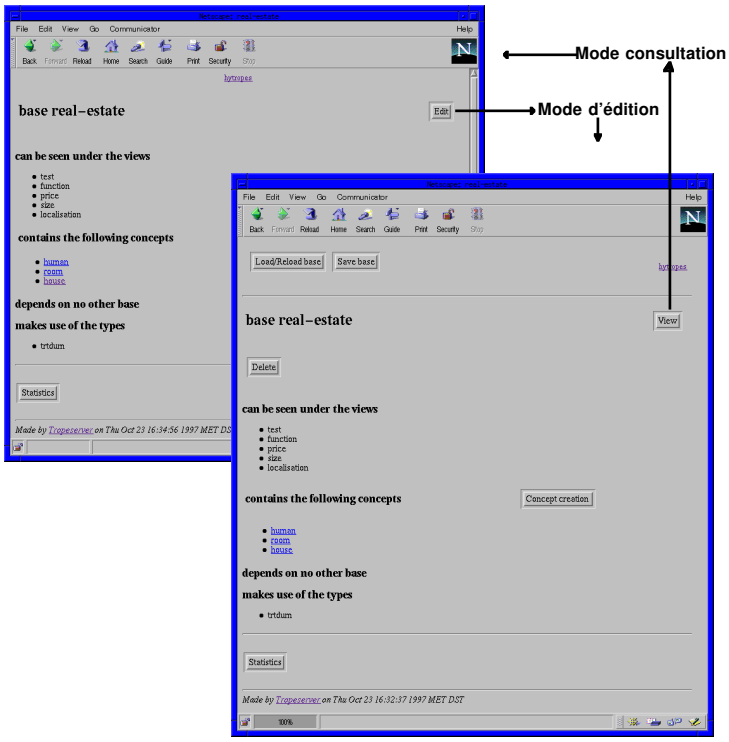
- Aucune modification des bases n'est permise, le serveur doit donc permettre uniquement la consultation de celles-ci.
    - L'apparence doit être identique au serveur Hytropes déjà implémenté, sans accès aux fonctions d'édition.
- Le serveur doit pouvoir permettre l'édition et la consultation.
    - L'utilisateur souhaite consulter la base. Les options d'édition ne doivent pas gêner, encombrer sa navigation, mais la possibilité d'effectuer une modification doit lui être fournie à tout moment.
    - L'utilisateur souhaite effectuer des modifications, les options d'édition propres à l'entité consultée doivent apparaître.

b. Résolution

L'implémentation mise en place tient compte de ces deux possibilités:

- Pour traiter le premier cas, une variable globale d'environnement (au sens Ilog Talk) est positionnée de manière irréversible au lancement du serveur. Elle permet d'autoriser, ou non, le mode d'édition sur ce serveur. Suivant la valeur de cette variable, les pages HTML engendrées en consultation contiendront ou non la possibilité de réaliser des modifications. Le même code est donc utilisé dans ces deux cas.
    - Consultation uniquement si la variable n'est pas positionnée
    - Consultation ou édition si la variable est positionnée

- Lorsque le serveur autorise le mode d'édition (variable globale décrite ci-dessus correctement positionnée), le problème du passage du mode d'édition à celui de consultation est résolu par l'ajout d'un bouton de type «va et vient» à l'interface de navigation. Il est situé en fin de la ligne de titre de chaque écran et permet, en cliquant dessus avec la souris, de passer instantanément du mode dit de consultation (sans les outils d'édition) au mode d'édition (avec les outils d'édition).



Mode d'édition vs. consultation d'une base Tropes

Ce choix de réalisation permet à l'utilisateur qui souhaite faire uniquement de la consultation, de ne pas être encombré par les différentes possibilités d'édition de base. D'autre part, d'un point de vue sécurité, il ne risque pas d'effectuer de modifications accidentelles lors de sa navigation. Son positionnement en haut à droite permet de ne pas le mêler au corps principal du document, la sélection accidentelle s'en trouve fortement réduite et sa localisation grandement facilitée. Pour terminer, l'accès à l'aide d'un bouton poussoir libellé, plutôt que d'une icône comme dans Ontolingua, présente l'avantage d'être plus explicite, plus rapide à afficher et à transmettre que ce dernier. Ces constatations compensent l'esthétique moindre du bouton poussoir face à l'icône.

c. *Le mode d'édition*

L'interface de modification, désormais intégré à l'outil de consultation, se devait de rester homogène par rapport à celui-ci. J'ai donc fait en sorte que son intégration se fasse de la façon la plus transparente possible aux yeux de l'utilisateur. Les écrans propres à l'édition conservent donc la même philosophie, la même convivialité, que ceux conçus pour la visualisation. On y retrouve notamment le «localizer», permettant

à tout moment de savoir où l'on se situe dans la base et les références arrières dans les titres autorisant un retour immédiat vers la structure mère de celle sur laquelle on souhaite agir.

Dans le mode d'édition, les services de modification sont présentés comme ceux de la consultation, par des boutons poussoirs libellés au nom du service apporté. Ils peuvent être visibles à trois emplacements distincts suivant la nature de la modification apportée:

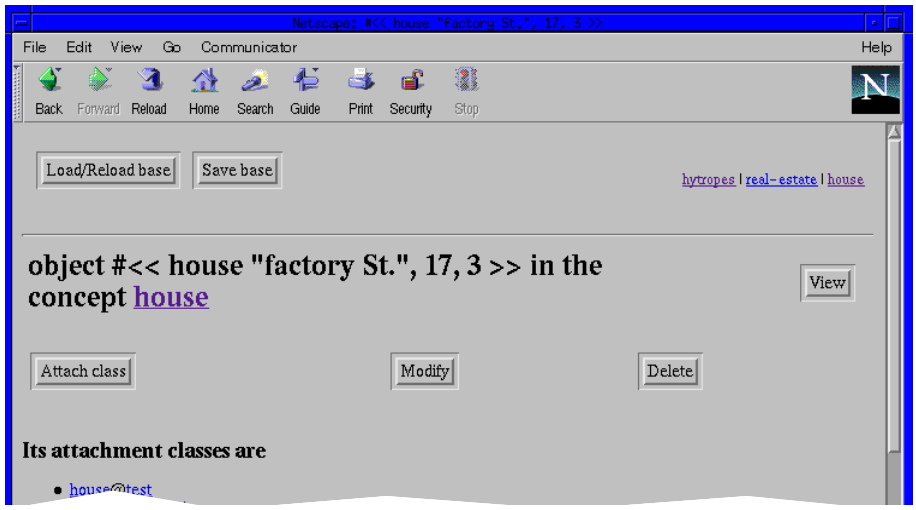
1. En en-tête de page

   En en-tête de document (à gauche du «localizer» comme on peut le voir sur la copie d'écran ci-dessus) se trouvent les fonctions d'édition qu'il est souhaitable de trouver à tout moment sans avoir à perdre de temps en sélectionnant un lien spécifique vers l'entité de Tropes sur laquelle elle agit. On y trouve donc la possibilité de sauvegarde de la base pour valider une modification ou au contraire le rechargement de celle-ci après s'être aperçu d'une erreur (et avant d'avoir sauvegardé la base).

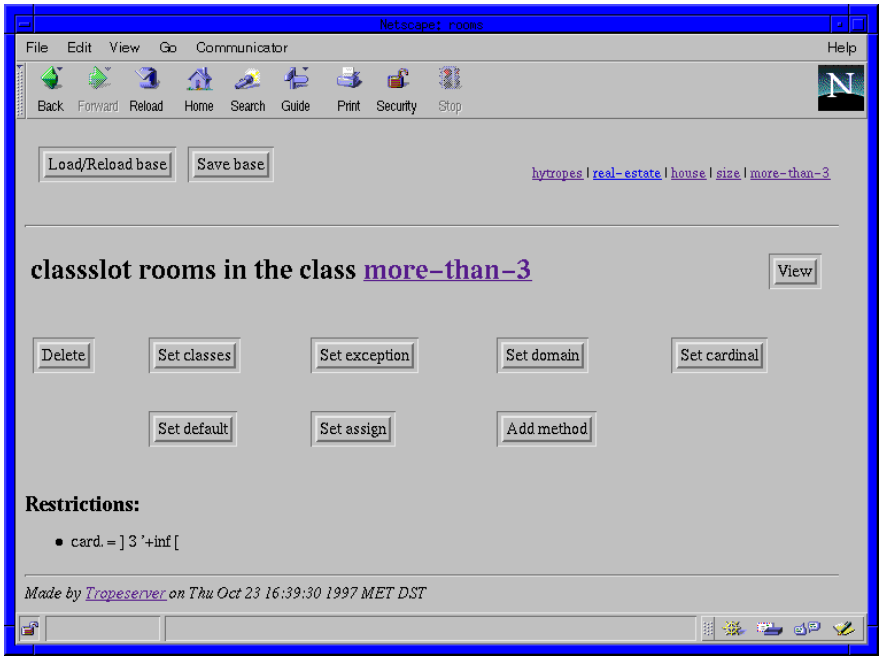
2. Sous le titre du document

   On peut les trouver d'une part, juste sous le titre, sous la forme d'une barre de boutons poussoirs. Ces boutons représentent les services d'édition qui ne modifient pas la structure de l'entité éditée, mais permettent par contre de compléter une description partielle de celle-ci. On y trouve également la possibilité de supprimer purement et simplement cette entité. Autrement dit, les éléments dits terminaux dans la structure des bases Tropes (attributs de concept, passerelles, instances, attributs de classe) voient l'ensemble des services de modification regroupés à cet emplacement. En revanche, les éléments composés d'entités inférieures (bases, concepts, points de vue, classes) ne présentent ici que les services d'édition ne modifiant pas la hiérarchie de structure.

   L'exemple suivant montre l'écran d'édition d'une instance du concept «house» (appartement), on y trouve les possibilités de supprimer l'instance (Delete), modifier les valeurs de ses attributs (Modify) ou lier l'instance à une classe dans chacun des points de vue définis (Attach).



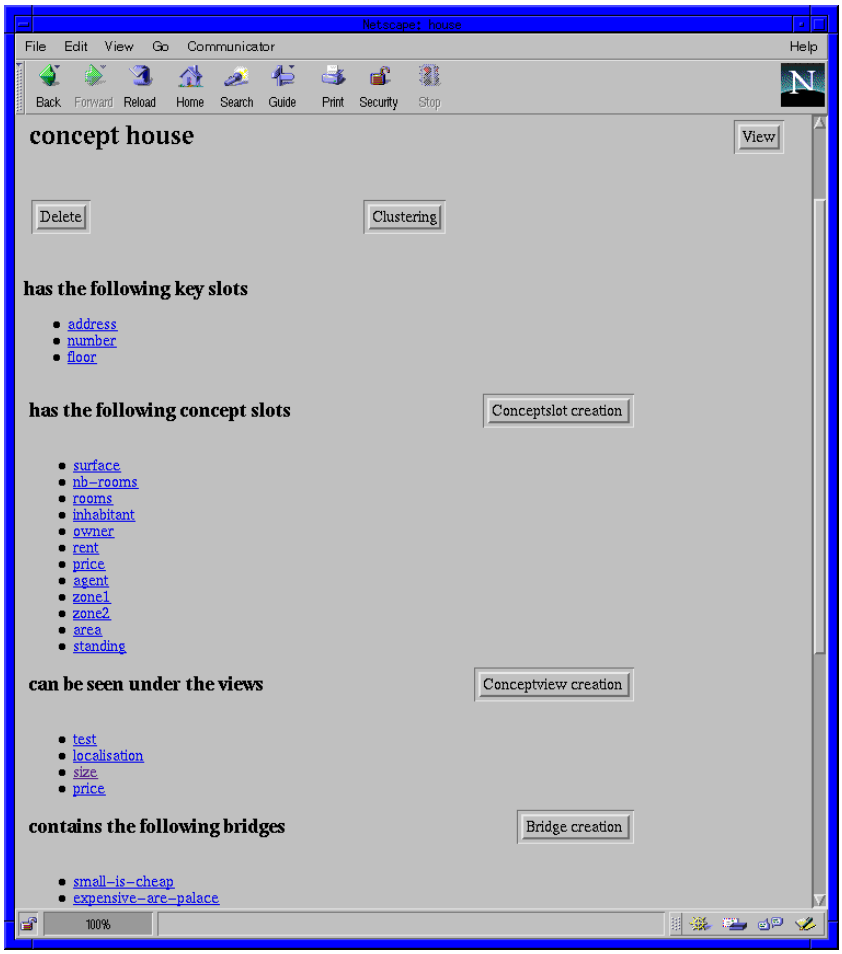
Ci-dessous l'édition d'un attribut de classe avec les différents services permettant de spécifier les contraintes de spécialisation ainsi que les mécanismes d'inférence de l'attribut de classe «more-than-3».



3. A la suite des sous-titres présentant les sous-structures d'un élément

   L'autre emplacement dans lequel on peut trouver des services d'édition concerne uniquement les entités Tropes composées de sous-structures, il se situe en fin de chaque ligne présentant justement ces dernières. Les services présents à cet endroit, permettent en fait de définir et d'ajouter un élément de structure inférieur à l'entité éditée. J'ai choisi cet emplacement car c'est ici que l'on peut visualiser la liste des éléments qui composent l'entité éditée. Si l'on s'aperçoit qu'un élément est manquant dans cette liste, il n'est aucunement besoin d'avoir à chercher longtemps l'accès à la fonction d'ajout de structures, elle se trouve toujours à proximité.

L'exemple suivant montre l'édition du concept «house». Le concept est une entité non terminale de la structure des bases Tropes, il est composé d'attributs, de points de vue, de passerelles et d'instances. On peut noter l'emplacement des boutons permettant l'ajout de ces structures, non loin de la liste déjà définie et juste sous le titre du document la barre de boutons comportant les possibilités de supprimer le concept «house» ou d'effectuer un clustering[14] sur ce dernier.

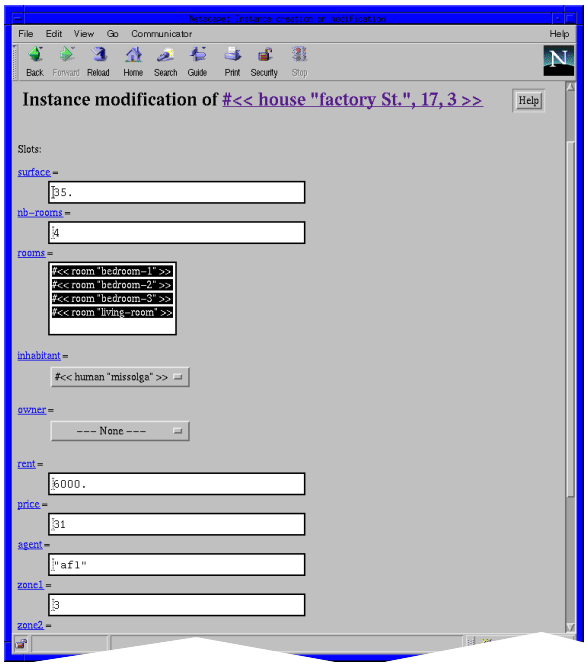


2. Les services de l'interface de modification

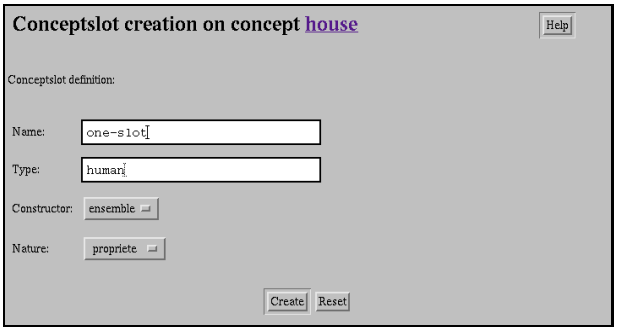
L'interface de modification couvre l'ensemble des fonctions d'édition de Tropes. Je ne vais pas en énumérer ici toutes les fonctions d'édition possibles de l'API et comment elles se traduisent au travers de mon interface, mais seulement les plus représentatives afin de mieux cerner ce que peut apporter une telle interface pour les manipulations de bases de connaissances.

- Modification des valeurs d'attribut d'une instance

[14]Construction d'une taxonomie de classes à partir des instances du concept



- Création d'un attribut de concept

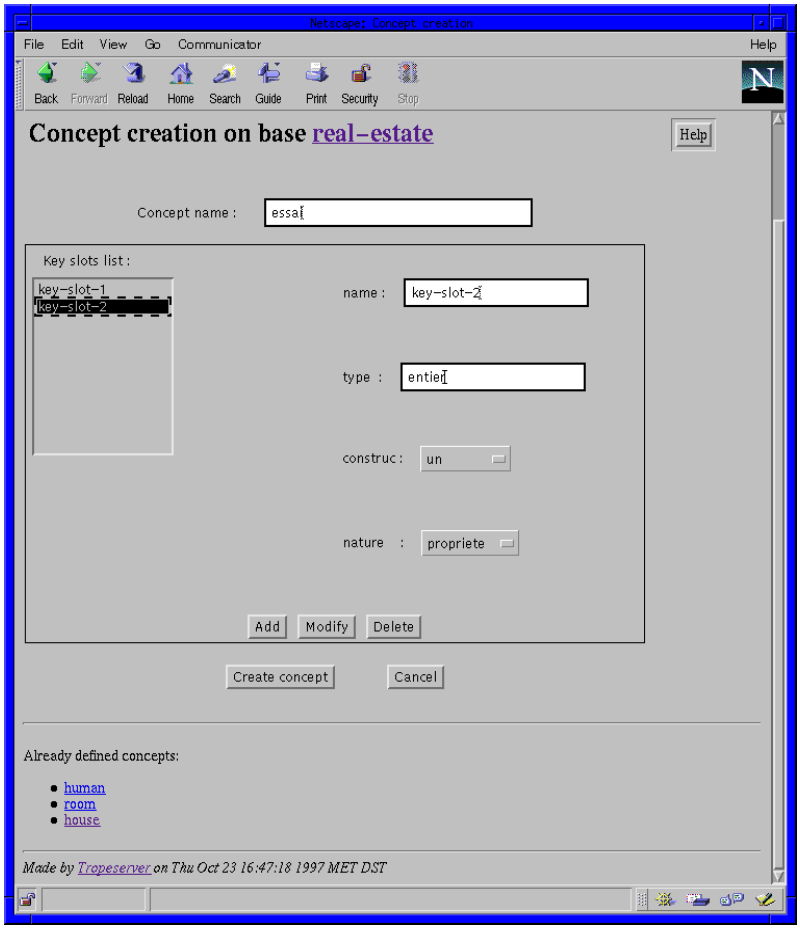


3. L'apport de Java

C'est dans le domaine de la modification que l'on voit les limites du langage HTML au profit de Java. Certaines fonctions de l'API nécessiteraient en effet de présenter plusieurs écrans à l'utilisateur avant de pouvoir être exécutées. A l'aide de Java et de ses possibilités graphiques l'interface est plus conviviale, la puissance d'un langage de programmation permet plus de fonctionnalité. La création de concept et la définition d'une cardinalité pour un attribut de classe ont été développés à l'aide d'une «Applet» Java afin d'apporter plus de souplesse dans la définition de ces entités. L'exemple suivant de la création d'une passerelle illustre bien ce que peut apporter une interface graphique, elle utilise notamment l'affichage d'arborescence de classes déjà utilisé pour la visualisation d'un point de vue.

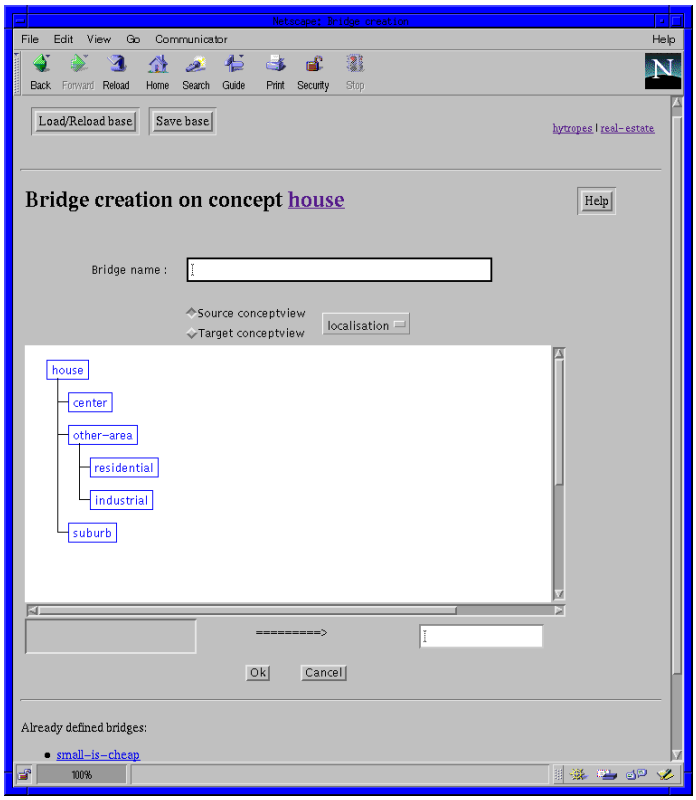
- Création d'un concept

  Comme vue dans l'étude, la création d'un concept nécessite la définition de l'ensemble des attributs clés. Le nombre de ces attributs n'étant pas connu à l'avance et l'objectif étant de minimiser les écrans intermédiaires, la solution finalement adoptée est celle qui met en œuvre une applet Java.



- Création d'une passerelle

  La création d'une passerelle pour un concept nécessite de lui donner un nom, de choisir une ou plusieurs classes sources et une classe destination (une seule par point de vue). l'Applet suivante montre ce qu'apporte Java pour implémenter cette fonctionnalité:



- L'affichage principal, au centre, montre l'arborescence de classes du point de vue sélectionné au-dessus.

- Lorsque l'utilisateur sélectionne une classe de cette arborescence avec la souris, elle vient alors s'ajouter aux classes sources si le choix «source conceptview» est choisi, ou bien s'inscrire comme classe destination si c'est «target conceptview» qui est choisi.

- Pour annuler la sélection d'une classe d'un point de vue, il suffit de sélectionner le point de vue en question puis de choisir avec la souris un point en dehors de l'arborescence.

- Une fois le nom de la passerelle saisie, les classes sources et la destination choisies, il ne reste plus qu'à créer la passerelle en choisissant le bouton «create» de l'écran.

- Les contraintes de Tropes sont respectées par l'interface. A savoir que si un utilisateur sélectionne deux classes différentes dans un même point de vue, la

première sélection est écrasée par la seconde car deux classes d'un même point de vue ne peuvent être sélectionnées.

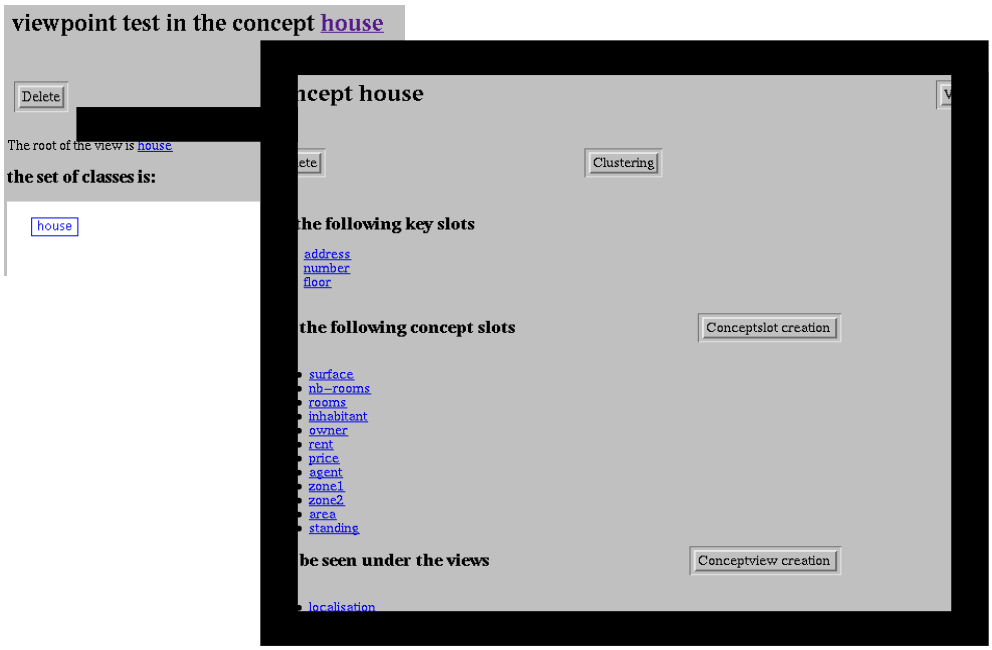
4. Les enchaînements d'écrans

L'enchaînement des écrans est sans aucun doute la chose la plus simple à réaliser, par contre elle fait partie des points les plus critiques qui font qu'une interface est plaisante ou non à utiliser. Ce paramètre est d'autant plus important du fait de l'utilisation du Web comme support d'application: une requête est transmise par un URL qui donne lieu à une connexion; il faut donc minimiser les écrans intermédiaires.

En ce qui concerne la navigation dans la structure d'une base Tropes en mode d'édition, l'enchaînement des écrans est dicté par l'interface déjà mise en place pour la consultation. La partie étude de l'interface a déjà permis d'expliquer pourquoi l'interface d'édition est greffée à l'interface existante de Hytropes. Simplement, dans le mode d'édition, un lien vers une entité aboutira à l'édition de cette dernière alors qu'en mode visualisation on obtiendrait la consultation de celle-ci. Il reste donc à définir l'enchaînement des écrans propres à l'édition. Trois cas sont à considérer:

a. Suppression d'une entité

Ce cas ne laisse pas vraiment d'alternative, aussi la suppression d'une entité Tropes entraîne l'affichage de l'écran d'édition de l'entité mère de celle que l'on souhaite détruire. Une exception tout de même: la suppression d'une base, qui n'a pas de structure mère, amène à l'écran d'accueil de Hytropes.



b. Ajout d'un élément de structure

Les choses se compliquent ici puisque trois possibilités sont offertes. L'ajout d'une entité peut en effet déboucher sur l'affichage à nouveau de l'écran permettant l'ajout

d'une entité (le nouvel élément est alors présent dans la liste des entités déjà définies) , il est également possible d'aboutir sur l'écran d'édition de l'entité nouvellement ajoutée ou bien pour finir sur l'écran d'édition de l'entité mère.

Chacune de ces possibilités présente des avantages et des inconvénients suivant le contexte:

- Présenter de nouveau l'écran d'ajout d'élément

  L'avantage de cet enchaînement est certain si l'utilisateur souhaite justement ajouter une autre entité de ce type. Un concept possède en général de nombreux attributs, ce choix paraît donc intéressant pour la fonction d'ajout d'attributs dans un concept. Dans le cas où l'utilisateur ne voudrait plus ajouter d'attributs, cela lui coûtera soit un accès à l'entité dernièrement ajoutée soit un accès à l'entité mère. Pour d'autres éléments de structure cela ne paraît pas forcément évident.

- Afficher l'écran d'édition de l'entité ajoutée

  Ici aussi tout dépend de la nature de l'entité ajoutée. Un élément de type attribut de concept ne propose pas d'autre possibilité d'édition que la suppression. Il n'apparaît donc pas pertinent de visualiser un attribut de concept lorsque l'on vient d'en définir un nouveau. Par contre une entité comme l'attribut de classe (voir copie d'écran dans la partie «Le mode d'édition» plus haut) est généralement ajoutée dans le but de définir des

contraintes ou des mécanismes d'inférences de valeurs sur cet attribut. Pour cet élément il est par contre très intéressant d'afficher son écran d'édition après ajout.

- Afficher l'écran d'édition de l'entité mère

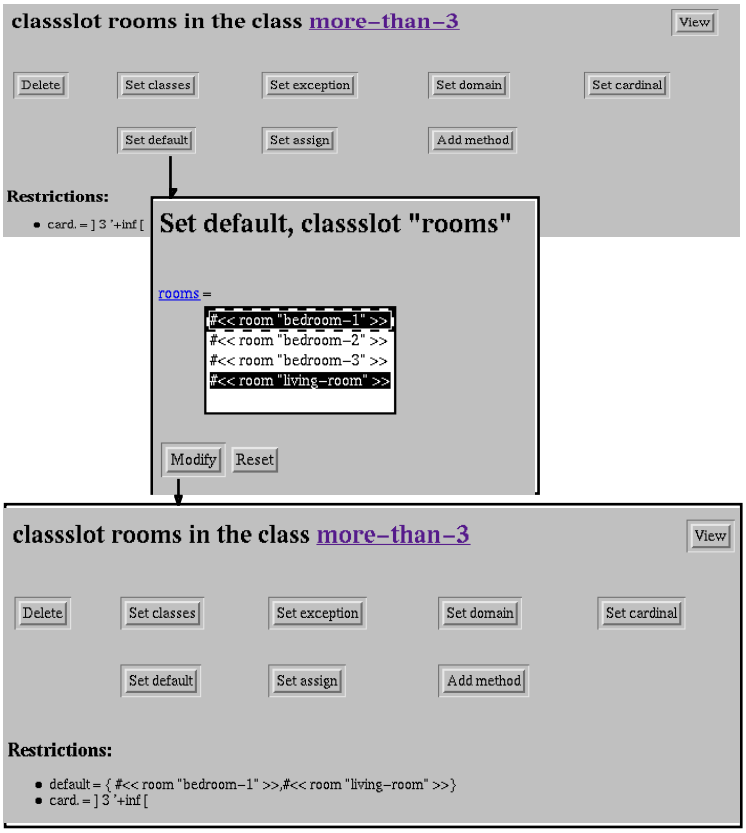
  Un utilisateur venant d'ajouter un élément au système a sûrement envie de compléter les connaissances sur celui-ci. Dans ce cas, l'affichage de l'entité mère coûte forcément à l'utilisateur un accès pour accéder à l'écran d'édition de l'entité ajoutée. Dans le but de minimiser les écrans intermédiaires, cette possibilité n'est pas souhaitable. Par contre, supposons qu'un utilisateur souhaite créer un attribut de concept dans le simple but de donner une valeur à cet attribut dans une instance déjà définie. Si une des deux possibilités ci-dessus avaient été choisie, il faudrait d'abord sélectionner le concept pour accéder à l'instance alors qu'ici nous sommes déjà positionnés sur celle-ci.

Le choix de l'un ou l'autre des enchaînements n'est donc pas évident et dépend fortement du type de modification que souhaite l'utilisateur (édition après création d'un tout nouveau concept ou édition d'un concept existant auquel on ajoute de nouvelles informations). Le choix dépend également de la nature de l'entité manipulée. Le tableau suivant montre les enchaînements d'écran que j'ai retenu d'après ces quelques remarques. La livraison très prochaine de Hytropes avec son interface d'édition à SGS-THOMSON permettra de valider ou non ces choix.

Écran source	Écran suivant
Création d'un concept	Édition du concept créé
Création d'un attribut de concept	Édition du concept père
Création d'un point de vue	Édition du point de vue créé
Création d'une passerelle	Édition du concept père
Création d'une instance	Édition de l'instance créée
Création d'une classe	Édition du point de vue père
Ajout d'un attribut de classe	Édition de l'attribut de classe créé

c. Modification des spécifications d'une entité

Comme pour la suppression, les alternatives optimales ne sont pas nombreuses. Aussi la modification d'une spécification d'une entité entraîne l'affichage de l'édition de l'entité stipulée. Aucun élément de la structure n'est en effet ajouté ou supprimé par ce genre de possibilité d'édition, il n'est donc pas pensable d'afficher une autre entité que celle sur laquelle on vient d'agir. D'autre part l'écran d'édition (image de l'écran de consultation avec ajout des fonctions d'édition), affiche non seulement la description, mais également le résultat de ce qui a été réalisé.



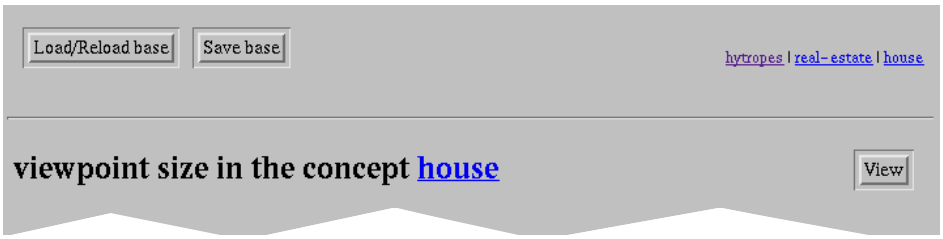
5. Autres possibilités de l'interface

Certaines possibilités, bien que non présentes explicitement dans l'API de Tropes, ont été ajoutées afin de donner plus de souplesse et de convivialité à l'application. En voici les descriptifs.

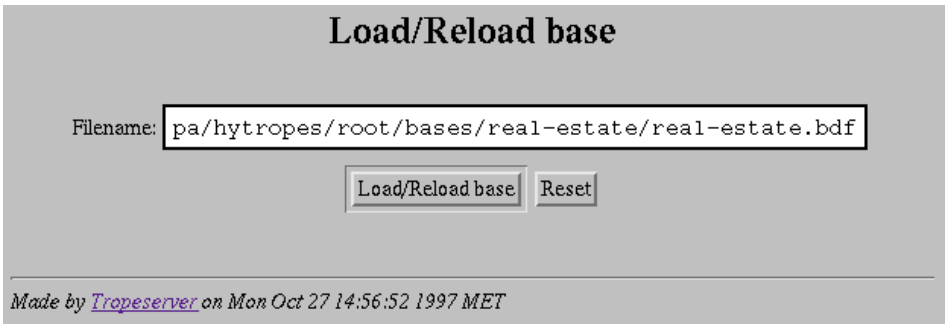
a. Annulation des dernières modifications

Il n'est actuellement pas possible à Tropes d'annuler une modification dernièrement effectuée. Cette possibilité est pourtant particulièrement intéressante lorsqu'un utilisateur a effectué une modification et qu'il désire finalement l'annuler (cf. Ontolinga). Pour palier ce manque, j'ai pu mettre en œuvre une solution spécifique. Lorsque le travail n'est encore pas sauvegardé, il est toujours possible de recharger le contenu d'une base à partir de ses fichiers descriptifs. De cette manière, l'état de la dernière sauvegarde avant modification, est alors restitué.

L'accès à la possibilité de recharger les derniers fichiers descriptifs, est présenté dans tous les écrans d'édition, au même titre que la sauvegarde. Il est ainsi possible, à tout moment et rapidement, d'annuler les dernières modifications.



Exemple:

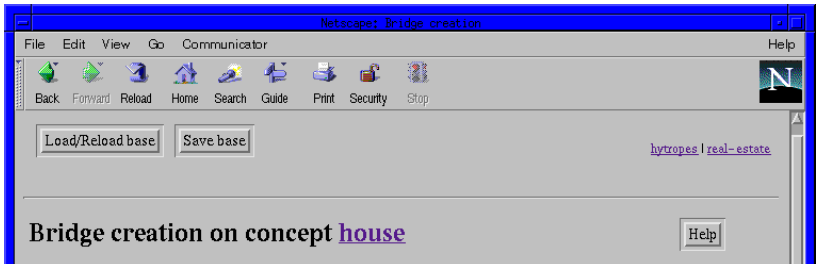


Perspective:

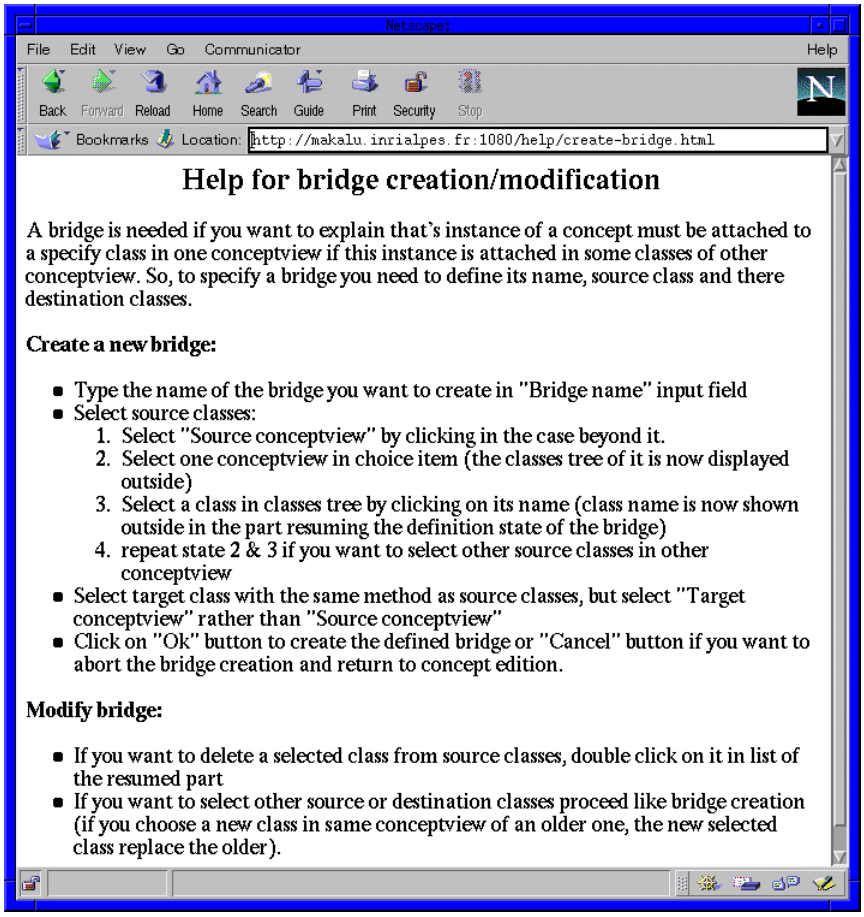
L'écran ci-dessus propose la saisie du nom et de l'emplacement du fichier à recharger. Une Applet Java avec navigation dans l'arborescence des répertoires du serveur remplacerait avantageusement la saisie d'un champ texte. Ceci n'a pas été développé puisque Tropes ne gère (à ce jour) pas les sauvegardes multiples («sauvegarder sous», «save as»). Le fichier proposé par défaut correspond donc (excepté les cas de tests) à celui dont on a besoin et aucune saisie n'est alors nécessaire.

b. *Aide en ligne*

Certaines opérations d'édition peuvent s'avérer délicates à mettre en œuvre, aussi une aide en ligne a été réalisée. L'appel de celle-ci s'effectue grâce à un bouton libellé «help» situé dans la page d'édition pour laquelle l'aide est requise.



exemple:



Perspective:

Ceci n'est qu'une ébauche (avec ses fautes d'Anglais). L'étude et la réalisation d'une aide en ligne peuvent en effet représenter à elles seules un sujet de mémoire complet en ajoutant notamment des liens hypertextes entre les pages, des accès indexés par mots clés, une recherche...

D. Le contrôle d'accès

La conclusion de l'étude à ce sujet me pousse à choisir une mise en place du contrôle d'accès à partir du paramétrage des serveurs HTTP. D'une part, c'est une méthode standardisée et facile à mettre en œuvre sans développement spécifique. D'autre part, l'étude a pu montrer qu'elle était très sécurisante. Pour rappel, deux méthodes permettent en général le paramétrage de la sécurité d'accès à un serveur HTTP: filtrage sur adresse IP ou par mot de passe.

Le filtrage par adresse IP ne peut être retenu dans le cadre de notre étude car tous les utilisateurs ne peuvent être identifiés par ce biais (prestataires de services et possibilité d'utilisation depuis n'importe où dans le monde). Je me suis donc tourné vers la seconde possibilité qui comprend la sécurisation des accès aux répertoires et fichiers présents sur le serveur et également à l'appel de certaines URLs. Trois niveaux de sécurité, présentés dans

leur ordre croissant du point de vue des droits, peuvent être dégagés. La structure des URLs doit permettre de les différencier:

- La sécurité en lecture pour le droit à la consultation
  => URL de type «http://nom.serveur.http/cgi-bin/hytropes/...»
- La sécurité en écriture pour le droit à l'édition
  => URL de type «http://nom.serveur.http/cgi-bin/hytropes/edit/...»
- Une sécurité complémentaire permettant l'administration du serveur
  => URL de type «http://nom.serveur.http/cgi-bin/hytropes/admin»
  + URL de type «http://nom.serveur.http/cgi-bin/relance-serveur»

Remarques:

- Je n'ai pas évoqué jusqu'à maintenant la possibilité d'administrer le serveur depuis Hytropes puisque le besoin n'a pas été évoqué et qu'il ne correspond pas au sujet de ce mémoire. Cependant dans la plupart des systèmes de gestion de bases de données, un niveau d'utilisation de type «administrateur» est présent. Prenons l'exemple d'un serveur Hytropes originellement prévu pour la consultation. On souhaite maintenant lui appliquer le droit d'édition. Trois étapes sont nécessaires pour réaliser cette action:

  1. arrêter le serveur,
  2. modifier son fichier de configuration afin de positionner correctement la variable globale correspondante (vu précédemment),
  3. enfin, relancer le serveur.

Le serveur Hytropes est donc indisponible quelques temps, d'autre part il est nécessaire d'être présent sur le site même où il se trouve, afin d'effectuer la manœuvre. Il aurait été intéressant d'avoir une commande, accessible par le Web, permettant d'effectuer ceci de façon transparente sans coupure du serveur. Tous les utilisateurs ne doivent bien entendu pas avoir accès à ces URLs, ils doivent donc être protégés spécifiquement.

- L'annexe 2 de ce mémoire présente l'ensemble des fichiers de configuration du serveur HTTP apache version 1.2 utilisé.

E. La gestion des erreurs

1. Les erreurs de saisies

En Ilog Talk, l'instruction permettant de guetter et signaler une erreur est «trap-error» [Ilog Talk Reference Manual]. On lui fournit comme argument la fonction que l'on souhaite surveiller, si une erreur intervient une variable est positionnée et permet de savoir si la fonction s'est bien déroulée ou non. Dans le premier cas, on peut alors poursuivre la suite normale des traitements alors que dans le second, on peut récupérer les données concernant l'erreur et les afficher à l'écran pour permettre à l'utilisateur de la corriger.

Macro «.TRAP-HTRP-ERROR» :

```
(defmacro .trap-htrp-error (act res)
  `(trap-error (() cond ())
       ,act
       (if cond
         (.print-htrp-error cond)
         ,res)))
```

Deux paramètres sont nécessaires:

«act» correspond au traitement que l'on souhaite surveiller
«res» correspond à la suite des traitements dans le cas du succès de «act»

Si la variable «cond» est positionnée c'est qu'il y a eu une erreur. L'affichage d'un écran d'information au format HTML est provoqué (.print-htrp-error), il contient notamment la description de l'erreur trouvée. Dans le cas où «cond» n'est pas positionné, la suite normale du traitement est alors exécutée.

Le principe de ce traitement est donc de solliciter le traitement «.trap-htrp-error» chaque fois que l'on doit traiter des données saisies par l'utilisateur, susceptibles de contenir des erreurs.

Tropes permet de distinguer trois types d'erreurs différentes, les erreurs dites récupérables, les erreurs standards et les «super» erreurs. Pour ces trois types d'erreurs les informations disponibles et récupérables sont légèrement différentes, aussi je tiendrai compte de l'implémentation dite par objets pour offrir un affichage spécifique (ou méthode spécifique) à chaque type d'erreur.

- *Les erreurs Tropes standards* contiennent:
  - le nom de la fonction qui a provoqué l'erreur (condition-caller)
  - un texte d'explication de l'erreur (condition-message)

- les paramètres venant s'insérer dans l'explication (condition-message-arguments)

Méthode «.print-errors» spécifique aux erreurs Tropes standards :

```
(defmethod .print-errors ((c <tropes-error>))
  (.print-line-error c))
```

- *Les erreurs dites récupérables*

Elles retournent une valeur supplémentaire correspondant à une valeur par défaut qui permettrait de poursuivre la demande sans intervention. Dans notre cadre, seul l'utilisateur doit décider de la pertinence ou non d'une modification, aussi je n'utiliserai pas ce paramètre pour corriger le problème. Ce type d'erreur sera traité comme une erreur standard, cependant l'information supplémentaire sera affichée dans le message afin d'éclairer l'utilisateur.

Méthode «.print-errors» spécifique aux erreurs Tropes récupérables :

```
(defmethod .print-errors ((c <recoverable-error>))
  (.print-line-error c)
  (let ((cont (tropes.error.continuation-value c)))
    (when cont
      (let ((string-stream (make-string-stream)))
        (sprintf string-stream "%A" cont)
        (flush string-stream)
        (printf "<DL><DT>replacement value could be :<DD>%s</DL>\n"
                (string-stream-contents string-stream))))))
```

- *Les «super» erreurs*

Ce sont en fait des erreurs d'ordre général et une erreur plus spécifique (ou erreur sous-jacente) leur est toujours rattachée. Cette dernière permet de mieux localiser l'erreur. L'affichage du message pour ce type d'erreurs fera donc apparaître la sous-erreur comme étant la cause de la super-erreur.

Remarque: La sous-erreur peut elle même être une «super» erreur, ...

Méthode «.print-errors» spécifique aux super-erreurs Tropes :

```
(defmethod .print-errors ((c <super-error>))
  (.print-line-error c)
  (let ((sub (tropes.error.sub-error c)))
    (when (and sub (subclassp (class-of sub) (class <condition>)))
      (printf "<P> because : </P>\n")
      (.print-errors sub))))
```

- *Autres erreurs*

Les erreurs de type programmation, principalement lorsque l'on fait des suppositions sur le contenu d'un URL non valide (saisie à la main) sont également captées par ce mécanisme. Ceci permet de ne jamais écrouler le

système dans de tels cas. Les informations disponibles sont identiques à celle des erreurs Tropes standard, aussi l'algorithme est le même.

```
(defmethod .print-errors ((c <condition>))
  (.print-line-error c))
```

Remarque:

Un module de révision de bases de connaissances [Crampé 96] est sur le point d'être achevé autour de Tropes. Le principe très général d'une révision de base de connaissances est de proposer intelligemment des corrections lorsqu'un utilisateur risque d'introduire une incohérence dans une base [Crampe 96b]. Lorsque ce module sera validé et intégré à Tropes, il sera aisé de l'intégrer à la gestion d'erreurs de Hytropes. Il s'agira en effet d'un type d'erreurs particulier qui donnera lieu à un traitement spécifique à la place du simple affichage du message d'erreur. Dans un premier temps le module de révision récupère les informations fournies dans l'erreur, il les analyse ensuite, afin de déterminer l'ensemble des solutions envisageables. Puis dans un deuxième temps, il affiche un formulaire présentant les solutions trouvées, l'utilisateur n'a plus qu'à choisir celle qui lui convient.

2. Contrôle de l'existence des objets Tropes

La conclusion de l'étude ne laisse pas de choix, il s'agit ici de vérifier l'existence dans le système, de l'entité Tropes sur laquelle une requête (aussi bien d'édition que de consultation) est soumise. Aussi, à chaque fois qu'un URL est soumis, le décodage de la partie de celui-ci contenant l'entité Tropes sur laquelle on souhaite agir a été modifié afin de traiter trois cas:

- L'entité existe toujours
- L'entité n'a pas été trouvée
- L'URL a une syntaxe incorrecte pour désigner une entité Tropes

Dans le cas où l'entité existe bien toujours, l'architecture réseau de Hytropes assure qu'elle ne va pas être supprimée avant la fin du traitement commencé (contrôle de la concurrence d'accès), on peut donc poursuivre celui-ci en toute sécurité. Dans le cas contraire un message indique à l'utilisateur que sa demande ne peut aboutir et l'invite à réactualiser le document à partir duquel il a tenté d'effectuer sa requête. Le dernier cas est équivalent à une erreur de saisie, il est donc traité de la même manière.

3. Généralisation du contrôle d'existence

Le contrôle d'existence d'une entité Tropes avant affichage du document la concernant doit être généralisé à l'ensemble des fonctions du système, c'est-à-dire non seulement dans les nouvelles fonctionnalités d'édition, mais également dans celles de consultation déjà existantes.

- Lorsqu'une personne décide de rechercher certaines instances d'un concept à partir d'un filtre de sélection. La génération de l'écran de saisie du filtre de sélection

nécessite la présence du concept sur lequel l'utilisateur souhaite interagir. D'autre part après avoir renseigné le filtre, l'utilisateur soumet sa requête. Il faut à nouveau contrôler l'existence du concept avant de pouvoir fournir un résultat.

- Dans le cadre de l'édition, prenons l'exemple d'une personne désirant créer un nouvel attribut de concept:

  contrôle 1 - le concept doit toujours exister pour autoriser la génération du masque de saisie des éléments nécessaires à la création d'un attribut. Ce contrôle a déjà été évoqué au point précédent.

  contrôle 2 - Lorsque l'utilisateur valide ses saisies, il faut de nouveau vérifier l'existence du concept avant de poursuivre le traitement. Le concept peut en effet avoir été supprimé après l'affichage du masque de saisies et avant que l'utilisateur ne finisse de le renseigner.

4. Erreurs d'URLs

Dans une utilisation courante du serveur Hytropes, les liens vers les différentes entités Tropes et les services associés à celles-ci sont générés par le système. Cependant, pour des raisons pratiques, l'utilisateur (ou même un programme) peut être amené à saisir ses propres URLs et y introduire des erreurs imprévisibles. La gestion de telles erreurs n'est pas gérée dans le cadre du serveur de consultation, cependant il n'est pas tolérable que le serveur d'édition s'écroule si de tels cas se produisent (modification perdues). Plusieurs actions sont donc à mener en ce sens, à la fois pour le serveur d'édition, mais également pour le serveur de consultation:

- Aucune supposition, quant au contenu d'un URL, ne doit être effectuée lorsqu'on décompose celui-ci. Si ce n'était pas le cas, s'assurer que cette supposition ne peut en aucun cas générer d'erreurs ou mieux, inclure celle-ci dans un mécanisme de contrôle d'erreurs (.trap-htrp-error), comme présenté dans le cas des erreurs de saisie.

- On a vu précédemment qu'il fallait vérifier l'existence d'une entité Tropes avant d'entreprendre une action sur celle-ci. Ceci ne suffit pas toujours, il faut également contrôler la nature de l'entité Tropes trouvée. L'utilisateur peut, en effet, vouloir sélectionner les instances d'un concept et saisir l'URL d'une sélection d'instances pour un attribut de concept ce qui n'est pas possible dans Tropes.

F. Conclusion

Les différents points évoqués dans l'étude, ainsi que le matériel informatique disponible, ont finalement permis l'obtention d'une interface d'édition de bases de connaissances Tropes, cohérente vis-à-vis des spécifications demandées:

- L'ensemble de l'API d'édition de Tropes est en effet couvert par l'interface.

- Les aspects sécurisation du serveur, aussi bien du point de vue de la gestion des accès, que de celui de la concurrence d'accès ou bien de la gestion des erreurs sont également assurés:

  - contrôle d'accès à l'aide du paramétrage des serveurs HTTP;
  - concurrence d'accès assurée par l'architecture réseau de Hytropes;
  - gestion des erreurs assurée par des traitements spécifiques.

- Une généralisation de la structure des URLs d'édition, en accord avec celle déjà existante pour la consultation a été définie, elle permet un décodage et une interprétation rapide du service demandé à Hytropes.

Lorsque cela est permis dans le paramétrage, au lancement du serveur, l'accès aux fonctions d'édition est assuré par un bouton de type va et vient. Ce dernier permet de passer simplement du mode de consultation à celui d'édition. Dans ce dernier cas, les possibilités d'édition sont réparties sur trois emplacements d'après la nature de la modification demandée. Pour conclure sur ce chapitre, il est important de noter également, l'utilisation du langage Java, nécessaire à l'implémentation de certaines fonctions de l'API grâce à ces possibilités graphiques notamment.

remarque:

La réalisation, elle même, a été survolée dans ce chapitre. Cependant l'annexe 4 issue de la documentation Tropes, présente l'ensemble des écrans de l'interface et leurs enchaînements.

VI. Bilan et perspectives

A. Bilan

Ce travail avait pour objectif l'étude et la réalisation d'une interface d'édition de bases de connaissances Tropes au travers du Web. Plus précisément il s'agissait d'étendre le logiciel de navigation dans une base Tropes (Hytropes), afin qu'il soit possible de modifier, compléter les connaissances contenues, directement à partir du réseau.

Pour atteindre cet objectif, la démarche suivie a nécessité trois étapes préalables:

La première étape a permis d'exprimer clairement les bases du problème au travers d'une présentation des éléments clés de notre application. Les termes de bases de connaissances et système à base de connaissances par objets ont ainsi été définis, afin de s'imprégner des besoins de ce type d'application et de leur objectif. Par la suite, une analyse du système Tropes et particulièrement de son API, nous présente la structure d'une base de connaissances Tropes et l'ensemble des possibilités d'édition qu'il a fallu développer dans notre système. Enfin, une présentation du World Wide Web nous confirme, s'il en était besoin, l'intérêt de cet outil comme support de conception de base de connaissances. L'édition de bases au travers du réseau permettra, en effet, à des spécialistes d'un même domaine de travailler ensemble sur une même base, depuis pratiquement n'importe quel endroit de la planète et ce avec un coût minimal.

Les besoins de l'application et ses justifications étant posées, une seconde étape du projet consista à analyser le logiciel de navigation déjà développé (Hytropes) et à étudier une application similaire à celle que nous devions réaliser (Ontolinga). De cette manière, nous avons pu déterminer les éléments essentiels qui caractérisent les écrans de Hytropes tel que l'accès aux entités Tropes par des liens hypertextes, l'accès aux outils d'exploitation par boutons poussoirs ou l'utilisation d'interfaces graphiques pour représenter les arborescences de classes… D'autre part, toujours du point de vue de l'ergonomie, nous avons pu souligner l'importance de la présence sur chaque écran, d'une localisation rapide dans la structure de la base et d'un retour arrière facilité. L'étude d'une application similaire a quant à elle permis de réfléchir aux particularités nécessaires à l'implémentation d'une interface d'édition réseau telles que le contrôle d'accès ou la gestion des saisies de l'utilisateur. Nous avons pu également juger de l'aspect ergonomique de ce logiciel afin de dégager les fonctionnalités particulièrement intéressantes et d'autres à éviter.

Les deux premières étapes, qui représentent l'état de l'art du projet, ont permis de dégager de nombreux points de réflexion techniques et fonctionnels. Parmi ceux-ci, nous pouvons noter:

- le problème de la gestion des erreurs
- le problème du contrôle et de la concurrence d'accès

- la définition des écrans de l'interface d'édition

A chacun des problèmes évoqués, nous avons pu détailler l'ensemble des solutions afin d'y remédier et dans la mesure du possible, nous avons pu avancer des choix de réalisation d'après les avantages et inconvénients présentés dans chacune des solutions.

Le logiciel finalement réalisé répond à l'ensemble des besoins initiaux et tient compte des études évoquées précédemment. Il contient les caractéristiques suivantes:

- Édition de l'ensemble des éléments de structure de Tropes;
- Utilisation de Java pour gérer les interactions utilisateurs complexes;
- Contrôle d'accès par mot de passe;
- Sauvegarde ou annulation des dernières modifications;
- Rattrapage des erreurs;
- Aide en ligne.

Nous pouvons donc affirmer que l'objectif a été atteint. Une interface d'édition de base de connaissances Tropes a en effet pu être intégrée à Hytropes de façon transparente. Elle couvre l'intégralité des possibilités de l'API en terme de modification et tient compte des particularités d'une application d'édition de bases dans un environnement réseau. Elle est distribuée sur le réseau et installée chez certains partenaires avec la version 1.1 de Tropes.

B. Perspectives

Les perspectives de ce type d'application sont nombreuses. Disposer de bases de connaissances mondiales exploitables dans des domaines d'application tels que la médecine, la biologie ou bien d'autres encore rendraient en effet bien des services. Cependant, la réalisation du prototype d'interface d'édition n'était qu'une étape obligatoire pour obtenir de telles bases, d'autres évolutions sont encore nécessaires pour rendre Hytropes complètement opérationnel.

Il n'est tout d'abord pas pensable d'obtenir une base de connaissances consensuelle, à l'aide d'une seule base commune à plusieurs milliers de personnes. Une notion de travail en groupe devra donc être intégrée à Hytropes.
De cette manière un petit groupe de travail agira uniquement sur une base personnelle et devra soumettre à d'autres groupes ses modifications avant de pouvoir les intégrer dans une base plus générale. Une perspective immédiate est donc l'intégration dans Hytropes du protocole CO4 permettant la construction coopérative de bases de connaissances.

Un autre point d'évolution concerne la rigidité de l'API. Lorsqu'un utilisateur saisit des données incohérentes, le système les rejette et informe de l'anomalie par un message d'erreur. Suivant la complexité de la structure de la base et la connaissance qu'a l'utilisateur du domaine modélisé, la correction peut s'avérer fastidieuse sans

aucune assistance. Une autre évolution prévisible est donc l'intégration d'un module de révision interactive de base de connaissances.

Par ailleurs un premier bilan, technique celui-ci, permet d'observer certains manques de l'API quant aux possibilités d'édition de celle-ci.

Il est par exemple impossible de renommer une entité, insérer une classe dans une hierarchie ou bien obtenir l'ensemble des types de données utilisables pour définir un attribut de concept. Une étude pourra donc être menée afin de compléter au mieux l'API de Tropes. Le travail réalisé dans le cadre d'édition de bases par le biais du réseau permettra alors d'ajouter aisément ces nouvelles fonctionalités à Hytropes.

Pour conclure, une perspective immédiate de l'interface d'édition va être la définition prochaine d'une base de connaissances concernant les réseaux informatiques de l'entreprise SGS-THOMSON à Crolles (Isère). Cette opération permettra d'une part la validation de l'interface en environnement réel, mais également de la faire évoluer. Un premier retour positif a d'ores et déjà été effectué par les intervenants de SGS-THOMSON, ce qui laisse présager un avenir florissant à ce projet...

C. Apports personnels

Effectuer ce stage dans un laboratoire de recherche a été une expérience très positive, tant sur le plan humain, par les contacts avec les chercheurs (et enseignants) que sur le plan technique.

Le changement d'environnement informatique a en effet été radical pour moi, puisqu'il a fallu me familiariser avec le système Unix, les langages Ilog Talk, Java et HTML et également à l'utilisation du MacIntosh pour rédiger ce mémoire.

Glossaire

API	l'API pour «Aplication Programing Interface» est un ensemble de fonctions standards qu'il faut utiliser si l'on veut agir sur un environnement. Dans notre étude, l'API de Tropes représente l'ensemble des fonctions qui permettent de consulter et manipuler les différentes structures d'une base de connaissances Tropes.
CGI-script	La Common Gateway Interface (CGI) est une norme définissant l'interfaçage d'applications externes avec des serveurs d'information (dans le cas qui nous intéresse: des serveurs HTTP). La motivation conduisant à utiliser CGI pour faire appel à des programmes externes est la suivante: lorsqu'un document HTML est envoyé sur le Web, il s'agit d'un document statique, un fichier texte dont l'information ne change pas tant que vous n'avez pas édité ledit fichier. Il existe néanmoins un grand nombre de cas où l'on aimerait envoyer de l'information dynamique, voire changeante d'un instant à l'autre. Un programme externe appelé par CGI ("programme CGI") permet d'arriver à ce résultat dans la mesure où ce programme est exécuté en temps réel, au moment où le client soumet un URL au serveur HTTP.
HTML	HTML est une norme permettant de structurer des document afin qu'il puissent être visible à l'aide du Web.
HTTP	HTTP pour «HyperText Transfert Protocol» est le nom du protocole d'échange de données qui régit actuellement l'accès aux ressources du Web. Basé sur le protocole réseau TCP/IP, on dit qu'il s'agit d'un protocole léger, sans état. Cela signifie que chaque requête (ou URL) donne lieu à une ouverture de connexion entre le poste client et le serveur; une fois l'URL interprété, le résultat est retourné et une déconnexion est immédiatement effectuée. Cette connexion et déconnexion systématique à chaque envoi d'URL n'est pas optimisée et si dans les débuts d'Internet peu de données graphiques transitaient à travers le réseau ce n'est plus le cas aujourd'hui. D'autant plus que le succès du Web fait de lui un excellent support de communication, de publicité, de commerce, ... Aussi un nouveau protocole est en cours d'étude pour lui succéder: HTTP 1.1.
HyTropes	Logiciel permettant l'accès aux bases de connaissances Tropes via le Web.

Java (& applets Java)	Java est un langage de programmation orienté objet très récent. La possibilité d'exécuter une application Java sur la plupart des systèmes existants rend particulièrement intéressant l'utilisation de Java dans le cadre d'applications Web (on parle alors d'»applets Java» pour différencier les applications traditionnelles et les applications Web).
Serveur HTTP	Logiciel permettant d'établir la connexion entre un client Web (le navigateur) et les fichiers demandés par l'utilisateur (serveur).
Talk	Langage de programmation orienté objet développé par la société Ilog.
Tropes	Système de gestion de bases de connaissances à objet développé au sein du projet Sherpa de l'I.N.R.I.A. Rhône-Alpes.
URL	l'URL ou «Uniform Ressource Locator» est une extension du nom de fichier dans un environnement de type réseau Internet. Il permet en effet, de part sa syntaxe et sa sémantique, de localiser et d'accéder à n'importe quelles ressources sur Internet. Classiquement il contient les informations suivantes : • le nom de service Internet demandé • le nom du serveur qui devra répondre à l'URL • le chemin à parcourir pour localiser l'information demandée La «ressource» recherchée peut être de nature différente suivant le service Internet demandé. Il peut s'agir d'un fichier que l'on veut rapatrier (service ftp) ou un fichier HTML que l'on veut interpréter et afficher à l'écran (service http ou Web), ...
Web	ou World Wide Web ou WWW ou W3 est le service le plus connu fourni par Internet. Il permet la consultation de documents interactifs par n'importe quel utilisateur relié au réseau Internet et disposant d'un navigateur Web (logiciel permettant la consultation de pages Web).

Bibliographie

[Crampé & 96] Isabelle Crampé, Jérôme Euzenat
«Fondements de la revision dans un langage d'objets simple»
Actes LMO'96, Leysin (Suisse), pp134-149, 1996

[Euzenat & 95] Jérôme Euzenat, François Rechenmann
«SHIRKA, 10 ans, c'est TROPES ?»
Actes 2nd journée «langages et modèles à objets», Nancy (FR), pp 13-34, 1-3 Octobre 1995

[Euzenat 95] Jerôme Euzenat
«Building consensual knowledge bases: context and architecture»
Actes 2nd International conference on building and sharing very large-scale knowledge bases, Enschede (NL), 10-13 avr 1995

[Euzenat 96] Jérôme Euzenat
«Hytropes: a WWW front-end to an object knowledge management system»
Actes 10th knowledge acquisition workshop demonstration track, Banff (CA), pp(62)1-12, 9-14 novembre 1996

[Euzenat 97a] Jérôme Euzenat, Christophe Chemla, Bernard Jacq
«A knowledge base for D. melanogaster gene interactions involved in pattern formation»
Actes 5th International conference on inteligent Systems Molecular Biology (ISMB), Halkidiki (HK), pp108-109, 21-25 Juin 1997

[Euzenat 97b] Jérôme Euzenat
«Représentation de connaissances par objets»

[Farquhar 95] Adam Farquhar, Richard Fikes, Wanda Pratt, James Rice
«Collaborative Ontology Construction for Information Integration»
Knowledge Systems Laboratory (KSL) of Stanford University
url: ftp://ftp.ksl.stanford.edu/pub/KSL_Reports/ksl-95-63.ps

[Flanagan 96] David Flanagan
«Java in a Nutshell»
aux éditions O'Reilly Internnational Thomson

[Gennari] John H. Gennari, Diane E. Oliver, Wanda Pratt, James Rice, Mark A. Musen
«A Web-based architecture for a medical vocabulary server»
Knowledge Systems Laboratory (KSL) of Stanford University
url: http://camis.stanford.edu/projects/intermed-web/vocab/

[Gensel & 94] Jérôme Gensel, Pierre Girard, Olivier Schmeltzer
«Intégration de contraintes, d'objets composites et de tâches dans un modèle de représentation de connaissances par objets»
Actes 9ième RFIA, Paris, pp281-292 (11-14 Jan 1994)

[Gensel 92] Jérôme Gensel
«Contraintes et représentation de connaissances par objets: Application au modèle Tropes»
Thèse de l'université Joseph Fourier, pp 9-45, 26 Oct 1992

[Ilog 96] Ilog Talk reference manual vol I, II, III

[Lemaire 96] Florence Lemaire
«Une structure terminologique pour améliorer l'intelligibilité d'une base de connaissances scientifique»
Thèse des sciences cognitives de l'INP de Grenoble, 7 Nov 1996

[Lemay 96] Laura Lemay, Charles L. Perkins
«Le programmeur Java»
Éditions Macmillan

[Marino 90] Olga Marino, François Rechenmann, Patrice Uvietta
«Multiple perspectives and classification mechanism in object-oriented representation»
Actes 9th ECAI, Stockholm (SE), pp 425-430, 1990

[Rice & 95] James Rice, Adam Farquhar, Philippe Piernot, Thomas Gruber
«Lessons learned using the Web as an application interface»
Knowledge Systems Laboratory (KSL) of Stanford University
url: ftp://ftp.ksl.stanford.edu/pub/KSL_Reports/ksl-95-69.ps

[Tropes] Tropes reference manual version 1.0a, Juillet 1995
pojet Sherpa, Tropes reference manual version 1.1.a, Novembre 1997
url: ftp://ftp.inrialpes.fr/pub/sherpa/rapports/troeps-manual.ps.gz

Annexe 1: Les URLs d'édition

- Rappel de la structure d'un URL d'édition:

«http://nom-serveur:numéro-port/cgi-bin/hytropes/edit/mot-clé/objet-tropes?arguments»

- Le tableau suivant montre l'ensemble des services d'édition réalisés:

Mot clé	Objet Tropes	Arguments	Fonctionnalité
get	Tous		Mode d'édition de l'objet
delete-confirm	Tous		Confirmation de suppression d'un objet
delete	Tous		Suppression d'un objet
erase	Tous		suppression forcée d'un objet
get-create-base			Formulaire de création d'une base
create-base		nom de base & répertoire fichiers	Création d'une base
save-base-confirm	Base		Confirmation de la sauvegarde d'une base
save-base	Base		Sauvegarde d'une base
get-load-reload-base	Base		Formulaire de chargement ou rechargement d'une base à partir de son fichier descriptif
load-reload-base	Base		chargement ou rechargement d'une base
get-create-concept	Base		Formulaire de création de concept
create-concept	Base	(nature & constructeur & type & nom d'attribut) * n & nom du concept	Création d'un concept défini par les «n» attributs clés
get-create-conceptslot	Concept		Formulaire de création d'un attribut de concept
create-conceptslot	Concept	nature & constructeur & type & nom d'attribut	Création d'un attribut de concept
get-create-conceptview	Concept		Formulaire de création d'un point de vue de concept
create-conceptslot	Concept	nom de point de vue	Création d'un point de vue
get-create-bridge	Concept		Création d'une passerelle
create-bridge	Concept	(point-de-vue-source = classe-source) * n & point-de-vue-cible = classe-cible & nom de passerelle	Création d'un passerelle définie par ses «n» classes sources et son unique classe destination
get-create-or-modify-object	concept ou instance		Formulaire de création ou modification d'une instance
create-modify-object	concept ou instance	(nom-attribut-clé = valeur-attribut-clé) * m & (nom-attribut = valeur-attribut) * n	Création ou modification d'une instance définie par les valeurs de ses attributs clés (création) et non clés (création et modification)
get-modify-bridge	Passerelle		Formulaire de modification des caractéristiques d'une passerelle

modify-bridge	Passerelle	(point-de-vue-source = classe-source) * n & point-de-vue-cible = classe-cible & nom-pass.	Modification des caractéristiques d'une passerelle
get-modify-value	Instance	nom-attribut=««	Formulaire de saisie pour la modification de la valeur d'un seul attribut d'une instance
modify-value	Instance	nom d'attribut = valeur	Modification de la valeur d'un seul attribut d'une instance
get-attach-kv	Instance		Formulaire de choix d'un point de vue pour lier une instance à une classe
get-attach-cl	Instance	nom du point de vue	Formulaire de choix d'une classe dans un point de vue pour lier une instance à une classe
attach	Instance	nom-pt-vue=nom-classe	lier une instance à une classe
get-create-classe	Classe		Formulaire de création d'une sous-classe
create-classe	Classe	nom de classe	Création d'une sous-classe
get-create-classslot	Classe		Formulaire de création d'un attribut de classe
create-classslot	Classe	nom d'attribut	Création d'un attribut de classe
get-set-classslot-type	att. de classe		Formulaire de choix de types pour un attribut de classe
set-classslot-type	att. de classe	(pt-vue=classe) * n	Choix de types pour un attribut de classe
get-set-classslot-domain	att. de classe		Formulaire de choix d'un domaine pour un att. de classe
set-classslot-domain	att. de classe	domaine	Choix d'un domaine pour un attribut de classe
get-set-classslot-except	att. de classe		Formulaire de choix d'exceptions pour un attribut de classe
set-classslot-except	att. de classe	exception	Choix d'exceptions pour un attribut de classe
get-set-classslot-intervals	att. de classe		Formulaire de saisie d'intervalles pour un attribut de classe
set-classslot-intervals	att. de classe	(intervalle) * n	Positionner une série d'intervalles pour un attribut de classe
get-set-classslot-card	att. de classe		Formulaire de saisie d'une cardinalité pour un att. de classe
set-classslot-card	att. de classe	cardinalité	Positionner une cardinalité pour un attribut de classe
get-set-classslot-default	att. de classe		Formulaire de saisie d'une valeur par défaut pour un att. de classe
set-classslot-default	att. de classe	valeur par defaut	Positionner une valeur par défaut pour un attribut de classe
get-set-classslot-assigns	att. de classe		Formulaire de saisie des valeurs assignées à un att. de classe
set-classslot-assigns	att. de classe	liste de références	Positionnement des valeurs assignées à un attribut de classe

Annexe 2: Le serveur «Apache»

Apache est le serveur HTTP utilisé pour ce travail, il s'agit d'un projet développé au sein du consortium W3C. Trois fichiers sont nécessaires au paramétrage d'un serveur Apache, en voici les descriptifs.

Le fichier httpd.conf

Il s'agit des paramétrages du système et du réseau.

```
serverType standalone

# Port permettant l'accès au serveur
Port 1080

# Affiche dans le log le nom des client au lieu des numéro IP
HostnameLookups on

# Limitation d'accès à certains utilisateurs, groupes (au sens Unix)
# -1 => pas de limitation
User #-1
Group #-1

# Adresse Email de l'administrateur du serveur
ServerAdmin Christophe.Alemany@inrialpes.fr

# Repertoire racine à partir duquel les fichiers de config, log... sont trouvés
ServerRoot /home/makalu/alemany/apache_1.2.0/

# Nom des fichiers logs
ErrorLog logs/error_log
TransferLog logs/access_log
PidFile logs/httpd.pid
ScoreBoardFile logs/apache_status

# Nom du serveur
ServerName makalu.inrialpes.fr

# paramètres systèmes divers (nb maxi clients connectés en même tps...)
Timeout 400
KeepAlive 0
KeepAliveTimeout 15
MinSpareServers 1
MaxSpareServers 1
StartServers 1
MaxClients 20
MaxRequestsPerChild 50
```

Le fichier srm.conf

Il s'agit du fichier permettant principalement de déterminer l'aspect du serveur. Il permet également d'inclure des modules particuliers tel que la gestion des cgi-scripts.

```
# Répertoire racine du serveur (à partir duquel les fichiers HTML sont trouvés)
DocumentRoot /home/makalu/alemany/html/

# UserDir: The name of the directory which is appended onto a user's home
# directory if a ~user request is recieved.
UserDir  public_html

# Fichier(s) recherché(s) par défaut dans un répertoire du serveur
DirectoryIndex  menu.html

# FancyIndexing is whether you want fancy directory indexing or standard
FancyIndexing on

# Liste des icônes représentant certains types de fichiers
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*
AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
…
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^

# Icône par défaut
DefaultIcon /icons/unknown.gif

# DefaultType is the default MIME type for documents which the server
# cannot find the type of from filename extensions.
DefaultType text/plain

# Liste des langages supportés et leurs priorités
AddLanguage en .en
AddLanguage fr .fr
LanguagePriority en fr

# Nécessaire pour utiliser des CGI scripts:
AddHandler cgi-script .cgi
```

Le fichier acces.conf (sécurité d'accès)

Il s'agit du fichier permettant de gérer les autorisations d'accès au serveur.

```
# Accès libre à tous les fichiers statiques du serveur (sous répertoires compris)

<Directory /home/makalu/alemany/html>
  Options Indexes FollowSymLinks
  AllowOverride None
  order allow,deny
  allow from all
</Directory>


# *** Sécurisation du serveur Hytropes ***


# protection d'accès au repertoire des scripts cgi-bin (tout les accès à Hytropes notamment)
# (sont autorisés les utilisateur déclarés dans «users» et faisant partie du group tropes dans «groups»)

<Directory /home/makalu/alemany/apache_1.2.0/cgi-bin>
 Options Indexes FollowSymLinks
 AllowOverride AuthConfig
 AuthName "tropes"
 AuthType Basic
 AuthUserFile /home/makalu/alemany/apache_1.2.0/conf/users
 AuthGroupFile /home/makalu/alemany/apache_1.2.0/conf/groups
 require group tropes
</Directory>


# protection de l'execution des URLs commençant par... (URLs d'édition)
# (sont autorisés les utilisateur déclarés dans «users» et faisant partie du group edit dans «groups»)

<Location ~ "/cgi-bin/hytropes/(clustering|lex-edit|edit)/*">
 AllowOverride AuthConfig
 AuthName "tropes"
 AuthType Basic
 AuthUserFile /home/makalu/alemany/apache_1.2.0/conf/users
 AuthGroupFile /home/makalu/alemany/apache_1.2.0/conf/groups
 require group edit
</Location>


#  protection de l'execution des URLs commençant par... (URLs d'administration)
# (sont autorisés les utilisateur déclarés dans «users» et faisant partie du group admin dans «groups»)

<Location ~ "/cgi-bin/hytropes/(get-talk-cde|talk-cde)/*">
 AllowOverride AuthConfig
 AuthName "tropes"
 AuthType Basic
 AuthUserFile /home/makalu/alemany/apache_1.2.0/conf/users
 AuthGroupFile /home/makalu/alemany/apache_1.2.0/conf/groups
 require group admin
</Location>
```

Les fichiers «users» et «groups»

Les droits d'accès à ces deux fichiers doivent être particulièrement bien étudiés, il permettent en effet de connaître et paramétrer l'ensemble des utilisateurs autorisés à se connecter sur le serveur (lorsqu'un contrôle d'accès est nécessaire).

«users»

Le fichier «users» permet de déclarer des utilisateurs. Chaque utilisateur est identifié par un nom et un mot de passe (présenté sous forme crypté dans le fichier). Pour obtenir ce fichier un utilitaire (htpasswd) est livré avec Apache. Il faut l'utiliser de la façon suivante:

- Création du fichier et ajout d'un utilisateur

  1) taper «htpasswd -c nom-fichier-users nom-utilisateur» puis valider
  2) saisir le mot de passe de l'utilisateur puis valider (2 fois)

- Ajout d'un utilisateur

  1) taper «htpasswd nom-fichier-users nom-nouvel-utilisateur» puis valider
  2) saisir le mot de passe du l'utilisateur puis valider (2 fois)

- Suppression d'un utilisateur

  Supprimer manuellement l'utilisateur dans le fichier «users»

- Changement de mot de passe

  1) taper «htpasswd nom-fichier-users nom-utilisateur» puis valider
  2) saisir le nouveau mot de passe et valider (2 fois)

«groups»

Le fichier «groups» permet de regrouper plusieurs utilisateurs dans un seul et même groupe d'utilisateurs. Il s'agit d'un simple fichier texte, saisi manuellement suivant le formalisme décrit ci-dessous:

```
nom-groupe_1: nom-utilisateur_1 nom-utilisateur_2 ... nom-utilisateur_n
nom-groupe_2: ...
.
.
.
nom-groupe_m: ...
```

Annexe 3: Le lexique

L'apport de l'utilisation d'un lexique, pour enrichir une base de connaissances est indéniable [Lemaire 96]. Il permet, en effet, de répertorier et définir l'ensemble des termes utilisés dans les différentes bases, c'est-à-dire les termes dénommant les objets décrits dans la base de connaissances. Dans un environnement de partage de connaissances, l'ajout d'un tel outil permet d'une part, la définition d'un vocabulaire commun à tous les utilisateurs de la connaissance et d'autre part, il doit aider l'utilisateur dans sa consultation et sa compréhension du contenu de la base de connaissances.

Le développement d'un lexique, suivant les spécifications de «SiLex» décrites par Florence Lemaire dans sa Thèse [Lemaire 96], a donc été réalisé. Lié à Tropes, il permet de définir la terminologie de l'ensemble des entités de description des bases de connaissances Tropes. Ces dernières sont désormais accessibles et définissables par l'intermédiaire du Web, il convient donc également de définir une interface permettant la consultation et la construction du lexique à l'aide de cet outil. L'objet de cette annexe est donc de présenter la structure et l'API du lexique, ainsi que l'interface de consultation/modification des éléments lexicaux à l'aide du Web.

A. La structure du lexique

Le lexique regroupe un ensemble d'éléments lexicaux pouvant éventuellement être reliés entre eux par certaines spécifications. L'élément de base est donc l'élément lexical, il s'agit donc ici de décrire la structure de celui-ci.

L'élément lexical (le terme) est composé des informations suivantes:

- Son nom (unique dans le lexique), correspond à l'élément lexical lui-même.
- Sa définition permet de définir par du texte le sens du terme
- L'hyperonyme est un élément lexical représentant un terme plus général que le terme.
- Les hyponymes sont des éléments lexicaux représentant des termes plus spécifiques que le terme.
- Les synonymes sont des éléments lexicaux qui représentent des termes ayant le même sens que le terme.
- Les antonymes sont des éléments lexicaux qui représentent des termes ayant un sens contraire que le terme.
- Les références bibliographiques regroupent l'ensemble des ouvrages concernant un terme.
- Les entités Tropes liées à un terme.

B. L'API du lexique

Tout comme Tropes, le lexique contient une API permettant de le manipuler. Elle ressemble grandement à celle de Tropes. On y retrouve donc la même syntaxe (développée en Ilog Talk également) et le même type de possibilités.

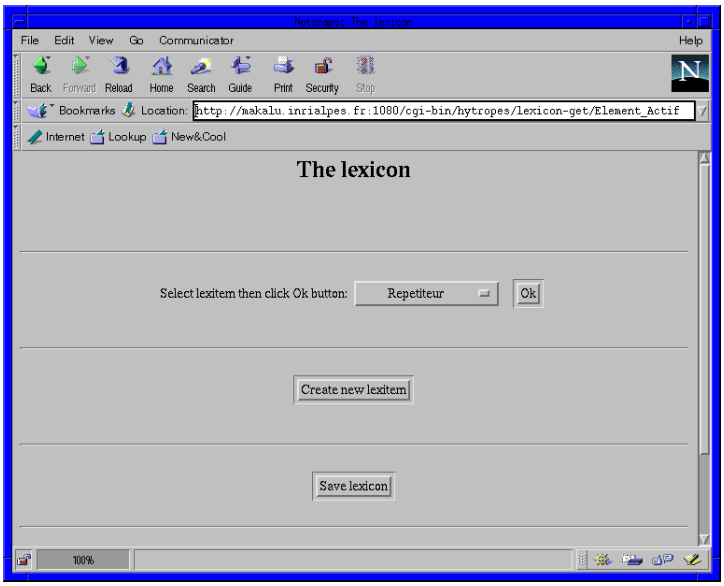
- Création, suppression, accès à un terme;
- Définition, suppression d'un lien vers un hyperonyme;
- Ajout, suppression d'un lien vers un hyponyme, antonyme, synonyme;
- Ajout, suppression d'une référence bibliographique;
- Création, suppression d'un lien vers une entité Tropes.

C. L'interface de consultation/modification

Aucune étude spécifique n'a été menée pour mettre en place cette nouvelle interface. Cette dernière est en effet incluse intégralement dans Hytropes, les études concernant Hytropes ont donc largement été prises en compte. De la même manière, l'ergonomie de l'interface reprend grandement celle de Hytropes, ceci afin de ne pas perturber l'utilisateur lorsqu'il passe du lexique à la base de connaissances et inversement.

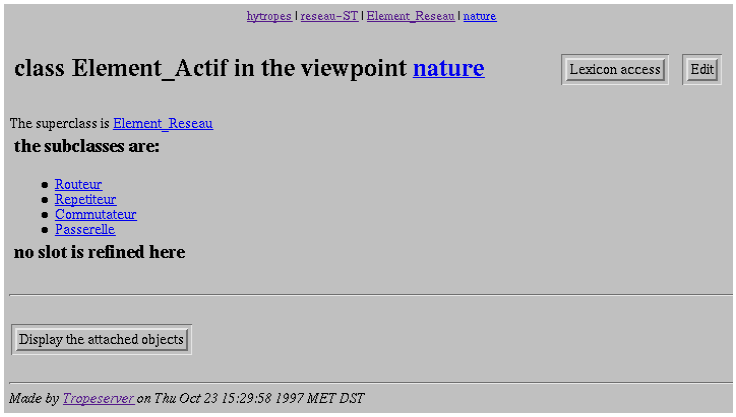
1. L'écran d'accueil du lexique

Cet écran permet d'accéder à un élément lexical du lexique grâce à une liste de tous les éléments lexicaux définis. Il offre également la possibilité de sauvegarder le lexique ou d'ajouter un nouvel élément lorsque le mode d'édition est activé.



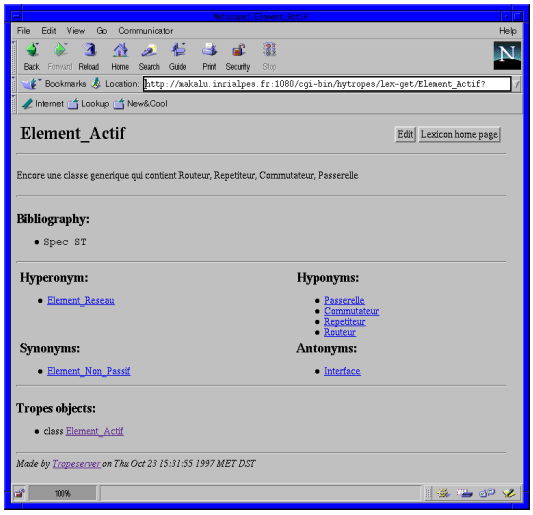
2. L'accès au lexique depuis Hytropes

Lorsqu'un élément de structure d'une base de connaissances est défini par un élément du lexique, un bouton permet l'accès direct à la consultation de ce dernier.



3. La consultation

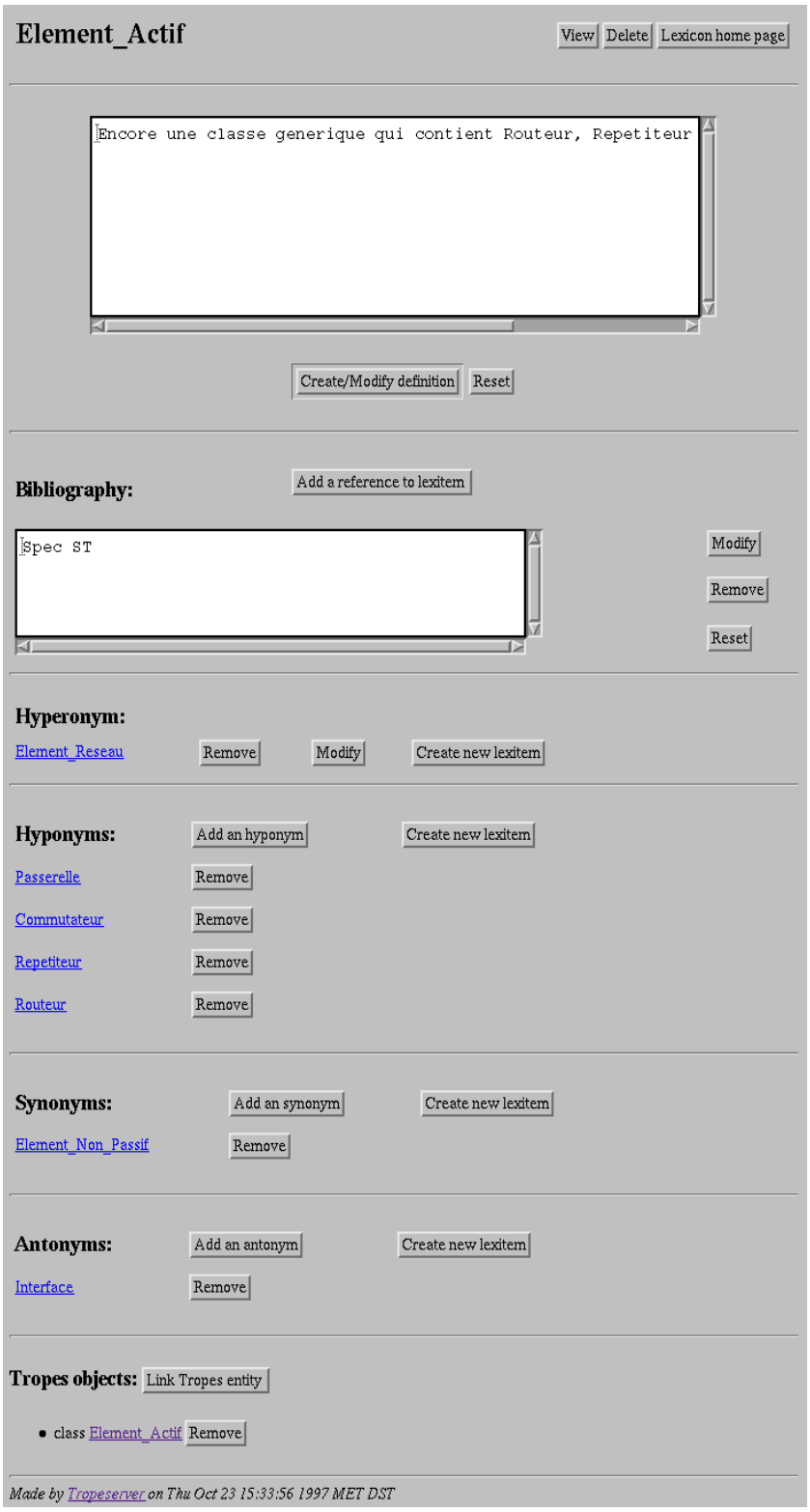
L'écran de consultation d'un élément du lexique est du même type que ceux présentant les entités Tropes. On y découvre donc l'ensemble des éléments qui permettent de définir un terme (nom, définition, hyperonyme...).



La sélection d'un lien vers un autre terme renvoie logiquement à la consultation (ou l'édition suivant le mode) du terme sélectioné.

4. L'édition

L'écran d'édition d'un terme permet de synthétiser et réaliser l'ensemble des possibilités de l'API du lexique dans ce domaine.


Element_Actif
View
Delete
Lexicon home page
Encore une classe generique qui contient Routeur, Repetiteur
Create/Modify definition
Reset
Bibliography:
Add a reference to lexitem
Spec ST
Modify
Remove
Reset
Hyperonym:
Element_Reseau
Remove
Modify
Create new lexitem
Hyponyms:
Add an hyponym
Create new lexitem
Passerelle
Remove
Commutateur
Remove
Repetiteur
Remove
Routeur
Remove
Synonyms:
Add an synonym
Create new lexitem
Element_Non_Passif
Remove
Antonyms:
Add an antonym
Create new lexitem
Interface
Remove
Tropes objects:
Link Tropes entity
class Element_Actif
Remove
Made by Tropeserver on Thu Oct 23 15:33:56 1997 MET DST

La simplicité de la structure du lexique (un seul objet: le terme) a permis de regrouper en un unique écran toutes les possibilités d'édition du lexique:

- Le bouton «Delete» permet de supprimer le terme édité.
- La modification ou la création de la définition d'un terme s'effectue en saisissant le nouveau texte dans le champs de saisie correspondant et de le valider avec le bouton «Create/Modify definition».
- L'ajout d'une référence bibliographique s'effectue à l'aide du bouton «Add a reference to lexitem» qui requiert ensuite la saisie de la référence puis sa validation. Modifier une référence existante de la même manière que pour la modification/création d'une définition de terme.
- La sélection d'un bouton de type «Create new lexitem» dans les sections hyperonymes, hyponymes, synonymes et antonymes permettent la création d'un nouveau terme désigné respectivement comme hyperonyme, hyponyme, synonyme ou antonyme du terme édité.
- La sélection d'un bouton «Modify» (Hyperonym) ou «Add an hyponym», «synonym», «antonym» permet de choisir un terme dans une liste. Ce dernier sera désigné respectivement comme hyperonyme, hyponyme, synonyme, ou antonyme du terme édité.
- Les boutons «Remove» permettent de supprimer des propriétés de l'entité concernée.
- La partie Tropes permet d'accéder directement à une entité Tropes (le retour s'effectue comme vu précédemment). Lier un terme à une entité Tropes affiche la liste des éléments Tropes chargés sur le serveur et permet de lier le terme édité avec l'un d'entre eux.

Remarques:

- Aucune applet Java n'a été développé pour la réalisation de cette interface, ceci pour des questions de temps de réalisation. Cependant la navigation et la recherche dans un lexique (à l'aide des hyperonymes et hyponymes notamment) se prêtent particulièrement bien à une représentation à l'aide d'arborescences des termes. De même la possibilité de lier une entité Tropes à un terme serait facilitée par le remplacement de la liste hiérarchique des entités Tropes disponibles par une représentation graphique des bases de connaissances.

- La taille d'un lexique peut être particulièrement grande, aussi la recherche manuelle d'un terme dans une liste pour y accéder peut se retrouver fastidieuse. Il serait donc intéressant de définir dans l'API du lexique une recherche par similarité, puis d'intégrer cette possibilité à l'écran d'accueil. De même la liste des entités Tropes pouvant être liées à un terme serait améliorée grâce à une applet Java.

Annexe 4: Les écrans de Hytropes

Cette annexe est un extrait de la documentation de Hytropes, elle présente de façon exhaustive les écrans de l'application et les liens entre ceux-ci suivant les interactions de l'utilisateur.

L'annexe a été supprimée de cette version pour une raison de taille du fichier PostScript. Elle peut être retrouvée dans le manuel de Troeps.

Résumé

Accéder à des bases de connaissances à l'aide de moyens hypermédias, conviviaux et accessibles par tous est une idée émergeante dans le domaine de la recherche en informatique. Le succès d'Internet grandissant au sein de la population des chercheurs, les nouvelles technologies, définies pour ce support, ont permis l'élaboration d'un outil de consultation de bases de connaissances au travers du World Wide Web. Cependant, la simple consultation n'est qu'une étape du projet. Il est en effet frustrant de ne pas pouvoir disposer d'outils permettant d'enrichir la connaissance avec une ergonomie comparable à celle utilisée pour la consultation. Ce mémoire, présente donc les différentes phases nécessaires à la conception et l'implémentation d'une interface de modification de bases de connaissances, au travers du WWW. Dans un premier temps, nous développons une présentation des éléments clés de l'application. Les termes de bases de connaissances et systèmes à base de connaissances par objets sont ainsi définis et plus concrètement, une analyse du système Tropes et de son interface fonctionnelle (API), présente la structure d'une base de connaissances et l'ensemble des possibilités d'édition qu'il a fallu développer. Enfin, une présentation du World Wide Web nous confirme l'intérêt de cet outil comme support de conception de base de connaissances. A la suite de ces présentations, nous analysons le logiciel de navigation déjà développé (Hytropes), ainsi qu'une application similaire, déjà pourvue de possibilités d'édition (Ontolinga). Ces deux premiers volets, qui représentent l'état de l'art du projet, ont permis de dégager des problèmes techniques et fonctionnels tels que celui de la gestion des erreurs, ou celui du contrôle et de la concurrence d'accès. A chacun des problèmes évoqués, nous observons l'ensemble des solutions envisageables et dans la mesure du possible, nous nous avançons sur des choix de réalisation. Pour terminer ce mémoire, nous présentons les caractéristiques du logiciel finalement développé, ainsi que les perspectives qu'un tel projet peut susciter.

Mots clés	Bases de connaissances, systèmes à bases de connaissances par objets, Tropes, Hytropes, World Wide Web, Internet

Keywords	Knowledge bases, Object knowledge base systems, Tropes, Hytropes, World Wide Web, Internet