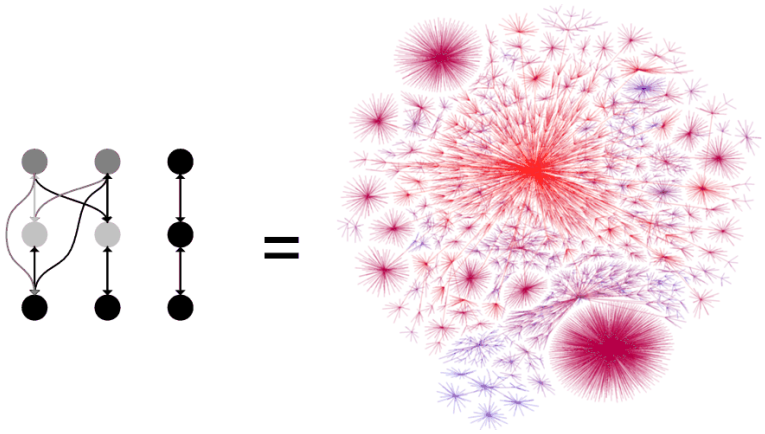
Modèle de représentation visuelle personnalisée de grands corpus de documents multimédia



Romain Vuillemot

Encadrante : Béatrice Rumpler

Rapport de stage de Master Recherche Informatique – spécialité Systèmes d'Information
Février – Juin 2006

LIRIS – INSA de Lyon – Université Lyon 1

Résumé

Ce rapport propose une modélisation de la construction et du parcours d'environnements de navigation dans de grands ensembles de documents, ce qui répond au problème de la représentation visuelle pertinente de données adaptée au contexte de l'utilisateur. Notre approche est dans un premier temps de décrire les ressources et données disponibles grâce à un langage créé spécifiquement. Ensuite donner un sens à ces descriptions afin de recomposer et d'adapter une nouvelle combinaison de représentations en fonction de règles issues du profil utilisateur. Des réalisations pratiques valideront une partie du modèle.

Abstract

This report aims to design the construction and exploration process of large data collections, in a context-oriented navigation environnement. Our approach is to describe available data and ressources with a specificly created language. Then giving these descriptions sens in order to recompose an adapted environnement to rules issued from user's profile. Practical implementations will partially validate this propostion.

Table des matières

Introduction

L'expérience de navigation se doit d'être agréable. Explorer de grandes quantités d'informations, issues aussi bien de résultats d'une recherche que d'une simple exploration, doit offrir un attrait, une liberté particulière, autorisant l'utilisateur à s'approprier l'outil afin d'en faire un usage efficace. D'autant plus qu'aujourd'hui, les évolutions conjointes du stockage et du transfert de données permettent de faire fi de la distance par rapport aux documents et à leur temps d'accès : tout est disponible, partout et n'importe quand. En parallèle, les méthodes d'acquisition et de restitution de documents multimédias les rendent de plus en plus et de mieux en mieux exploitables (augmentation de la résolution). L'accès simple à l'information n'est plus le point critique, l'accès *pertinent* le devient.

Mais dès qu'il s'agit d'effectuer une tâche précise, comme explorer les très nombreux résultats d'une recherche, cela devient très complexe et frustrant pour l'utilisateur, au regard du temps d'apprentissage ou des moyens (matériels, logiciels) nécessaires à sa mise en oeuvre. Et pour le concepteur d'environnements visuels, on se rapproche de l'artisanat, tellement les recommandations manquent de formalisme.

La première raison est que même si un ordinateur possède plusieurs couches logicielles successives qui le rendent facile à utiliser, l'interface visuelle est au sommet de cette pyramide : l'utilisateur y est directement confronté. De surcroît, cette même visualisation est un mélange de domaines aussi différents (psychologie, arts visuels, design) que nombreux et qu'il faut judicieusement intégrer dans la phase de conception. La maîtrise nécessaire est si complexe et subtile qu'elle relève quasiment du savoir-faire. D'autant plus qu'il y a autant d'applications à concevoir qu'il existe de types et combinaisons de données : une généralisation du processus de construction d'environnement est-elle possible ?

L'avantage du processus de visualisation est d'être atemporel et spatial. Contrairement au langage verbal, qui est lié à un système temporel et linéaire, la vision d'une image est immédiate dans sa totalité : c'est un atout essentiel, mais qu'il faut valoriser et pourquoi pas combiner avec d'autres modalités (la voix, le toucher, ..), exigeantes plus ou moins en attention.

L'approche actuelle du problème est hélas soit très technique, quand il s'agit de mettre en avant un apport scientifique en lui-même (réponse à un défi technique ou une représentation d'un jeu de données précis). Soit

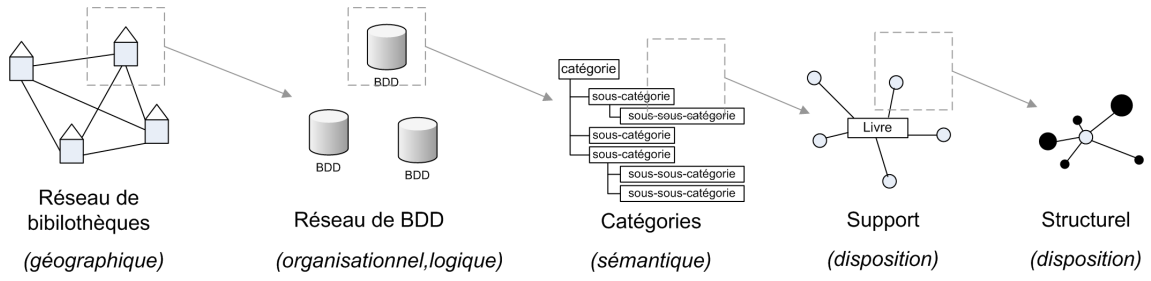


FIG. 1 – **Exemple de visualisation de livres de plusieurs bibliothèques :** *il y a une explosion de la taille des données, de part l'amélioration de la qualité de numérisation mais aussi de part l'interconnexion des réseaux. A chaque niveau d'abstraction, la quantité de données est raisonnable mais une représentation visuelle combinée de toutes les données est tout simplement illisible. Certains niveaux d'abstractions (physiques) peuvent être une forme de connaissance : dans cet exemple la localisation physique des bibliothèques peut être exploitée pour indiquer la localisation géographique d'un livre empruntable.*

comportementaliste, dès que l'on aborde l'étude de l'usage ou des connaissances, côté utilisateur, impliquées sans contribution technique précise au final. Tout cela est au détriment d'intégrations et d'innovations concrètes pour l'utilisateur. Certes le constat est cruel, par manque de liant entre des communautés, mais il semble qu'il manque juste un rien pour rendre ces environnements généralisables à plusieurs situations, personnes, usages ou données.

Ces problématiques sont en parties déjà bien explorées et certaines s'inscrivent dans des programmes cherchant à dynamiser des recherches assez variées. Notre mission est de permettre l'exploitation personnalisée de grandes masses de documents multimédias et s'inscrit dans les travaux du projet national APMD (Accès Personnalisé à des Masses de Données, projet MD-33 2004-2007) [1] de l'ACI (Action Concertée Incitative) Masses de Données[2]. Le sujet initial de ce stage est d'étudier les méthodologies de réduction de l'espace documentaires de ces dernières et un modèle de représentation visuelle adapté. Il s'agira d'effectuer un état de l'art sur la modélisation et le traitement des grandes masses de données documentaire et sur les méthodes et outils de représentation visuelle (3D, etc.), d'identifier les connaissances impliquées dans le processus de sélection personnalisée de l'information et de proposer une méthodologie et un modèle de représentation synthétique (graphique ou autre) des documents multimédias.

L'objectif peut être reformulé comme suit : organiser une succession de représentations visuelles dont l'utilisateur aura besoin dans une situation spécifique afin de résoudre un problème. L'environnement de navigation est un tout, un fil qu'il faut judicieusement préparer et soumettre à l'utilisateur afin que celui-ci effectue son raisonnement dans les meilleures conditions : il faut donc également anticiper les besoins potentiels. Par exemple, répondre à la question « où suis-je ? » quand on est noyé dans de grandes masses de documents, « où vais-je ? » dès que l'on explore non-séquentiellement des données. Notre idée est d'analyser des relations intra et inter-documents et d'intégrer la notion du temps, en se positionnant à un niveau intermédiaire, entre la description *bas-niveau* des données et des représentations visuelles, et des concepts *haut-niveau* de raisonnement. En espérant au final réussir à capter ce savoir-faire de bonne conception, qui pour l'instant n'est pas formulé de manière synthétique et qui ne rime pas nécessairement avec complexité et exhaustivité, mais peut-être avec réactivité d'adaptation ?

Notre approche du problème sera semblable à la structure du rapport qui suit. Il est d'abord indispensable de bien poser le cadre dans lequel nous nous situons, savoir sur quelles communautés scientifiques nous nous appuyons et ce qu'elles nous apportent. Un esprit critique nous permettra d'analyser et définir une stratégie d'approche débouchant sur un modèle théorique proposé dans la deuxième partie. Ce sera le coeur du rapport à partir duquel nous déclinerons des règles d'agencement des interfaces qui serons évaluées dans la troisième partie selon une politique d'implantation et de validation. Une conclusion terminera ce rapport en proposant un programme de recherche permettant d'aller au delà de cette première démarche et de suivre d'autres pistes intéressantes.

Remerciements

Cette recherche a été partiellement soutenue par le Ministère délégué à la Recherche et aux Nouvelles Technologie, dans le programme ACI Masses de Données, projet MD-33.

[1] http\penalty\@M://apmd.prism.uvsq.fr/
[2] http\penalty\@M://acimd.labri.fr/

Chapitre 1

Contexte et problématique

En premier lieu, nous allons cerner puis défricher plusieurs domaines scientifiques sur lesquels il paraît *a priori* judicieux de se positionner, car étant directement liés au sujet initial. Dans un second temps, d'autres domaines seront explorés afin de compléter les manques éventuels et/ou d'en tirer une approche différente. Enfin la formulation d'une problématique générale fera guise de conclusion et nous expliciterons les approches que nous souhaitons entreprendre.

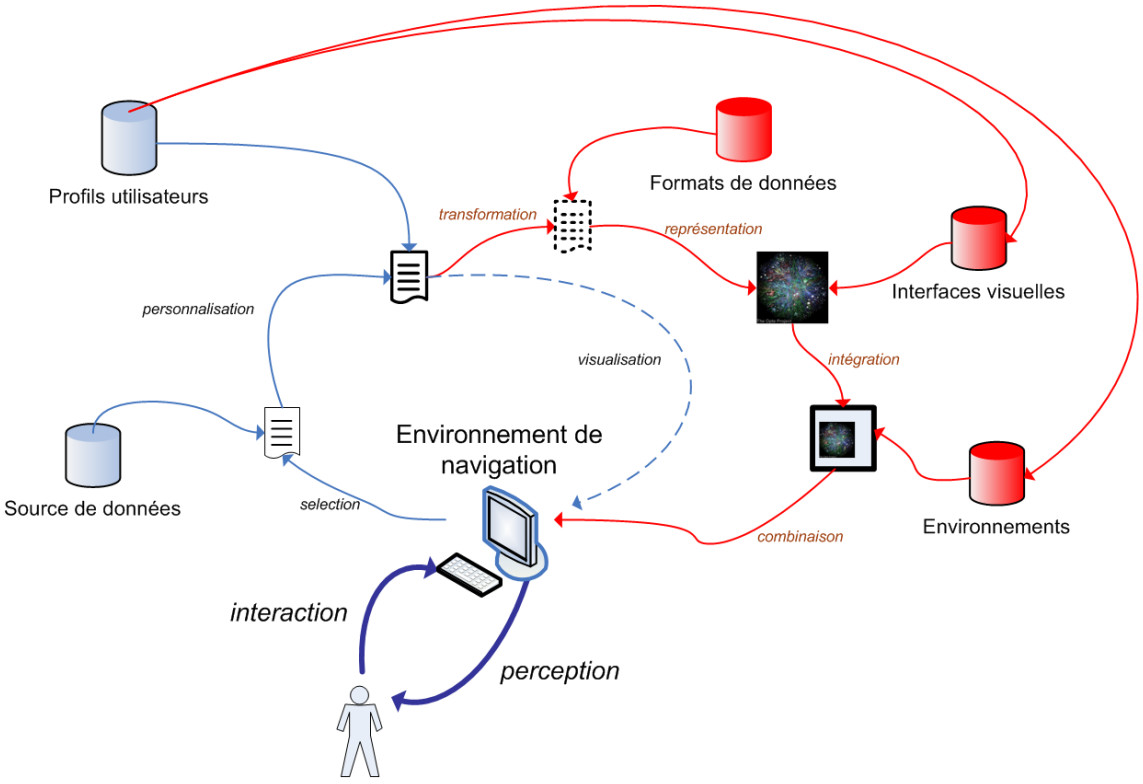


FIG. 1.1 – *Le schéma actuel (simplifié) d'un accès personnalisé à l'information fait défaut d'une représentation visuelle dépendante du contexte. Nous souhaitons combler ce manque en personnalisant les représentations et les environnements visuels afin d'améliorer la perception et l'interaction de l'utilisateur. A noter qu'une première étape technique sera la traduction en structures de données intermédiaires.*

1.1 Cadre précis

Nous nous restreignons, dans ce travail, à un contexte d'usage et de matériel très précis. Comme il n'existe pas de réponse simple à une question très complexe, nous souhaitons valider notre méthode sur un environnement, certes restreint, mais en tout cas bien connu et maîtrisé.

Acteurs impliqués

Afin d'éviter toute ambiguïté, nous allons donner notre définition des différents acteurs que nous allons évoquer par la suite. Ces profils permettent de cibler les capacités, le domaine d'expertise (ou de non-expertise) de l'acteur. A noter que la réalité est beaucoup plus souple car plusieurs casquettes peuvent être endossées, plus ou moins partiellement, et les limites bien plus floues.

Utilisateur λ : est l'utilisateur final qui doit interagir avec l'environnement de navigation afin de résoudre une tâche précise. Il a une faible capacité d'apprentissage immédiat, et peu flexible dans l'installation et la maîtrise de nouvelles techniques (matérielles ou logiciels). Cependant il a une culture générale de représentations visuelles connues (cartes de métro, réseaux, arbres, ..) et un niveau de connaissance et de raisonnement supérieur à l'ordinateur.

Technicien : a une formation technique pointue, il est ainsi capable d'implanter une nouvelle méthode de disposition ou de visualisation dans un environnement qui permettra de valider une hypothèse de faisabilité ou verrou technique.

Scientifique : cherche une méthode de visualisation ou d'exploration visuelles de grandes quantités de données, à des fins d'exploration de résultats ou un besoin de simulation. Il maîtrise l'outil informatique en esquivant toute optimisation dans ses programmes.

Designer : a une sensibilité artistique et possède la capacité d'évaluer et d'expliquer les subtilités d'une représentation visuelle selon sa signification. Sans qu'il soit pour autant possible de représenter explicitement son raisonnement.

Dispositifs matériels

Cette clarification est également cruciale, car nous verrons par la suite qu'une solution technique simple peut contourner un problème théorique très complexe (exemple : nous verrons plus loin que le défi assez compliqué de la méthode *focus+context* est de mettre sur *le même* écran des données et leurs liens avec la structure globale, alors que si l'on possède *deux* écrans on peut immédiatement résoudre ce problème en mettant une représentation sur chaque écran). Nous prendrons comme référence une station de travail ordinaire (un clavier, une souris, un écran et des haut-parleurs). Avec une carte graphique suffisante pour des représentations 3D avec une résolution assez élevée. Un navigateur web et une connexion internet ordinaire. Pas de droits particuliers sur la machine. Aucun dispositif d'acquisition. Pour résumer, le cadre le plus diffusable et le moins restrictif possible.

A noter que cela ne réduira en rien notre modèle qui sera apte à prendre en compte les évolutions existantes telles la mobilité, l'hétérogénéité des formats ou celles à venir telles que des interactions variées utilisant des dispositifs innovants.

Maintenant que le cadre est posé, intéressons nous aux problématiques de Recherche liées à notre mission.

1.2 Interfaces visuelles

Les interfaces servent d'intermédiaire entre des systèmes hétérogènes, qui par définition fonctionnent de manière différente. On les qualifie de visuelles dès qu'elles impliquent le processus de vision de l'être humain. L'intérêt qu'on y porte est légitime car ce sont des systèmes atemporels et spatiaux. Cela est avantageux par rapport au langage verbal qui est lié à un système temporel et linéaire. Ainsi la vision d'une image est immédiate dans sa totalité. Il reste cependant à la construire suivant des règles la rendant immédiatement compréhensible et véhiculant au mieux les informations que l'on souhaite diffuser.

Les données à représenter sont hétérogènes et au pire des cas en grandes quantités (au moins 10^6 objets ou symboles à visualiser). Ainsi nous allons devoir nous baser sur des travaux assez généraux de représentation visuelle

d'informations, et ensuite étudier des techniques efficaces de représentation de données, à savoir les codes visuels et la disposition de données dans un univers. Si la visualisation d'*information* se penche sur les données abstraites, telles que les collections de documents, nous distinguons et excluons de nos travaux la visualisation *scientifique* où les données on déjà une composante spatiale implicite. Par exemple des courbes, des tableaux de données, des schémas, des images médicales 3D, champs de vecteurs. Les problèmes de ce genre de représentations visuelles sont énumérés dans [Joh04] ou lors de conférences telles que *Visual Research Challenge*[1] et IEEE Visualization Conference [2]. Les principaux problèmes sont la précision de représentation des données (il doit y avoir une grande fiabilité). La convergence avec la visualisation d'*information* est également un défi abordé [RHJ+04].

1.2.1 Visualisation de l'information

Représenter des données a toujours été un défi à relever, quelle que soit l'époque à laquelle nous nous situons. L'apparition de l'informatique, à savoir *le traitement automatique de données*, a amplifié la quantité potentielle de données et les avancées de l'électronique, la résolution d'acquisition et de restitution. En un demi-siècle à peine nous passons de la ligne de commande à des résolutions d'images de qualité quasi-réalistes.

Le but de la visualisation est aussi d'exploiter le système visuel humain afin d'ouvrir un canal de communication. De nombreuses disciplines se sont penchées sur ce processus qui est nécessaire en permanence. Chaque domaine scientifique, aussi poussé qu'il soit comporte une partie visualisation, avec une approche plus ou moins théorique. Afin de maîtriser ces techniques, on peut lire de longs livres, comme *Information Visualization* [Spe00] souvent référencés dans des tutoriaux. En parallèle, des recommandations ont su être suffisamment bien formulées pour émerger et faire foi dans le domaine. A la fois concises et ouvertes, les suivre permet a priori d'offrir un cadre propice à l'utilisateur. Les voici :

Overview first, zoom and filter, then details-on-demand de Ben Shneiderman [Shn96]. C'est à la fois un constat et une proposition de point de départ pour la recherche visuelle d'information, qui se place toutefois à un niveau encore assez abstrait.

focus+context [LRP95] consiste à présenter l'information importante (*focus*) à la taille normale mais disposer autour les données (*context*) pour montrer comment l'information est reliée entre elles dans la structure globale. Les zones les plus éloignées sont diminuées ou supprimées. La vue en *Fisheye* [Fur86] est la plus représentative.

Cependant, des défis plus précis n'ont pas encore de solution générale avec ces méthodes générales : il existe ainsi des groupes d'intérêts qui se sont créés afin de stimuler la recherche en donnant des défis à relever, sur une base commune permettant d'évaluer les contributions. Par exemple le concours annuel d'*INFOVIS* [3] où un jeu de données doit être représenté visuellement afin d'en extraire des caractéristiques. Les méthodes proposées peuvent être une combinaison d'outils et d'algorithmes existants, mais agencés d'une façon adaptée. Un exemple intéressant sont les gagnants du concours de l'année 2004 [ADM+04] où l'échantillon de données était un lot de publications scientifiques.

D'un point de vue logiciel il existe de nombreuses librairies [4] dont les plus connues sont *prefuse* [HCL05] University of California, Berkeley ou *The InfoVis Toolkit* [Fek04] de l'INRIA. Il existe également de nombreux produits commerciaux mais que nous n'avons pu évaluer par manque de temps et de moyens.

De ces années de contributions il en sort périodiquement des documents de synthèse, résumant les apports, traçant les grandes directions à prendre et les verrous actuels du domaine. Afin soit de les résoudre ou bien de les contourner : *Top 10 Unsolved Information Visualization Problems*[Che05].

1.2.2 Codes de représentations

Comme le temps est précieux, il est nécessaire d'accélérer le processus d'apprentissage de représentations globales visuelles. La culture de l'utilisateur est prise en compte dans certains codes qui regroupent des connaissances qui font partie de notre culture générale. Jacques Bertin [Ber83] a pu définir les variables visuelles et structurer

[1] http\penalty\@M://www.sci.utah.edu/vrc2005/

[2] VIS2006 – http\unskip\penalty\@M://vis.computer.org/vis2006/

[3] http\penalty\@M://www.infovis.net/printRec.php\penalty\@M\protect\kern+.1667em\relax?rec=llibre&lang=2

[4] Un classement selon leurs fonctionnalités a été préalablement effectué et est disponible à l'adresse : http\unskip\penalty\@M://liris-7087.insa-lyon.fr/doc/classement.pdf

les premières règles de construction de l'image graphique formulées dans la bible de la visualisation. On parle de métaphores visuelles pour les codes régissant la construction de cartes de métro (entre autres), et la sémiologie graphique (science de la représentation) permet d'établir les conventions de représentation (icônes, symboles) et donc de communication.

Les défis actuels sont dans l'optimisation et l'automatisation de la construction de ces représentations visuelles. Le choix de la bonne représentation visuelle est aussi difficile, car il faut véhiculer la connaissance la mieux adaptée. Dans [BMR+05] les auteurs comparent l'efficacité de la représentation en carte de métro et en diagramme de Gantt de la communication entre les différents acteurs d'un grand projet. Ainsi la carte de métro est une représentation visuelle de la connaissance plus attractive, qui engage plus facilement la conversation et persiste plus longtemps dans la mémoire qu'un diagramme de Gantt.

1.2.3 Disposition des données à représenter

Le monde réel dans lequel nous vivons répond à des règles physiques que tout objet respecte. Newton les a formulées il y a quelques temps, et à vrai dire elles ont peu changées depuis. Cela contraste radicalement avec l'univers virtuel des documents électroniques qui sont liés à moins de contraintes que leurs homologues papiers. Ils ont, entre autres, le dont d'ubiquité, de reproduction infini sans perte de qualité. Les organiser de manière judicieuse est un problème majeur de la visualisation.

Dans certains cas, les données on déjà une composante géographique précise, qu'il est possible de représenter sur des cartes du monde réel [LT92]. Bien que d'autres types de données n'aient pas cette composante, une approche similaire est possible : on associe la sémantique des données avec des métaphores de représentation des données (Exemple : métaphore du bureau 3D) en pensant que c'est efficace. Mais des problèmes liés à la 3D, tels que la manipulation et l'orientation, surviennent. Enfin, la réalité augmentée est un très bon support (plus que la réalité virtuelle ou tout l'univers doit être généré), mélangeant un environnement réel apprivoisé avec des données superposées. Malheureusement elle demande des dispositifs supplémentaires (hors de notre cadre) et occupe trop fortement l'attention de l'utilisateur.

A défaut d'attributs pertinents à exploiter, l'approche actuelle est de donner à nos documents quelconques des propriétés dynamiques ressemblant au monde réel : *spring embedder* (générations en forme d'arbres), charges électriques, lois mécaniques, relaxation de graphes. Il faut considérer les données dans un ensemble virtuel auquel on accède par un angle de vue. Le problème est qu'il n'y a souvent pas assez de place : la solution est de substituer notre univers (euclidien) par un autre plus grand : multidimensionnel, fractal [Man75] [KY93], hyperbolique [WOWR03], .. Ces derniers ont la particularité d'être plus spacieux. Par contre ils n'offrent pas une représentation ordinaire ce qui donne très souvent lieu à des déformations et demandent un effort cognitif non négligeable. Le très connu *Fisheye* [Fur86] garde la propriété du *focus+context*, et a l'air utile mais au prix d'une illisibilité des données censées conserver le contexte. Les cartes auto-organisatrices [LSM91] sont intéressantes : les données voisines se rapproche au sens d'une fonction de similarité à définir.

Nous décidons d'explorer une méthode particulière, les graphes, proche des structures de données et pouvant inclure de la sémantique nous permettant d'exploiter les données d'une manière inédite telle que le *Visual data mining* [FGW02].

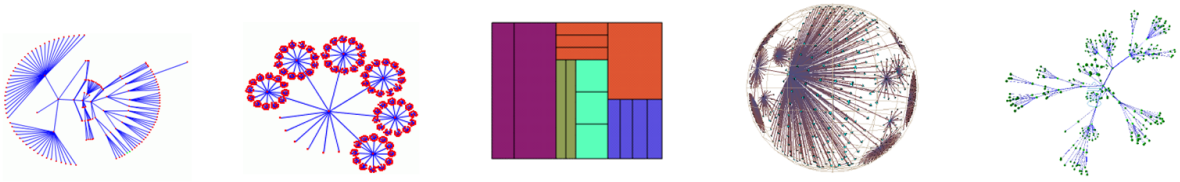


FIG. 1.2 – Exemples de dispositions issues de [HMM00]. Certaines exploitent une plus grande utilisation possible de l'espace. D'autres préfèrent respecter la hiérarchie. A noter que les représentations concentriques combinent bien ces deux propriétés.

Disposition sous forme de graphes C'est un domaine très actif, car les données, quelles qu'elles soient, possèdent toujours des relations entre-elles plus ou moins explicites et ainsi représentables sous forme de graphe. Par exemple le problème de dessin d'une carte de métro est un problème de disposition (*layout* en anglais) de graphes[HMdN04]. Il peut y avoir du clustering par regroupement de sous-graphes. Par exemple le regroupement en *smallworlds* qui permettent de regrouper des communautés permettant de faciliter la navigation. La Figure 1.2 propose différents exemples caractéristiques de représentation selon la structure que l'on veut mettre en avant. La complexité des problèmes de dessins de graphes est variée mais le temps de calcul est souvent exponentiel en nombres de sommets. C'est à quoi s'attellent des conférences comme dans *Graph drawing* [5] (on remarquera l'importance de la visualisation dans ce domaine). Il existe également des librairies (listées également dans notre classement préalable des librairies) telles que *Java Universal Network/Graph Framework* (JUNG), *GraphViz* [EGK+03] de AT&T's et *TULIP* de David Auber[6] [Aub03]du LABRI. Cette dernière offre une visualisation temps réel de graphes jusqu'au million d'éléments.

1.3 Environnements de navigation

Un environnement de navigation est une succession d'interfaces matérielles et de couches logicielles permettant à un être humain d'interagir avec un système. Il s'agit de différents niveaux qui font progressivement abstraction de la machine et qui permettent à l'utilisateur d'interagir à son niveau de capacité, avec le système. D'un point de vue matériel, on parle d'un ordinateur avec des entrées/sorties (souris/écran), d'un point de vue logiciel d'un système d'exploitation muni de programmes permettant d'effectuer des tâches précises (en phase avec les couches précédentes). C'est sur cette dernière catégorie que nous allons particulièrement nous focaliser, étant la plus flexible (il est plus facile d'installer un logiciel qu'acquérir du matériel) et ciblée.

Il est indispensable d'intégrer l'interaction de l'utilisateur afin d'exploiter et comprendre la nature des représentations visuelles qui sont générées. On parle alors d'Interactions Homme-Machine (IHM) concernent de nombreux aspects relatifs aux applications informatiques interactives. Nous allons cerner quelques points seulement qui nous intéresseront par la suite dans la conception d'interfaces et qui justifierons nos choix.

La conception d'un environnement est très souvent fondée sur des métaphores WIMP : Windows, Icon, Menu, Pointer, reprenant la métaphore du dossier, apparue dans les années 70. Adapté à l'époque (environnement de travails, dispositifs d'entrées réduits), elle est encore de vogue aujourd'hui : l'utilisateur a bien été au centre d'attention, mais a malencontreusement été généralisé avec la démocratisation anarchique des ordinateurs. Les WIMP sont mal adaptés pour des représentations globales : menus réduits, tailles limitées et le parcours des listes est beaucoup trop lent. Cependant cela reste utile dans certains cas et on peut s'appuyer par exemple sur la référence du domaine *«Designing the User Interface : Strategies for Effective Human-Computer Interaction»* [Shn97] de Ben Shneiderman. Emmerge aussi la plasticité qui est la capacité d'un système interactif à s'adapter au contexte en étant toujours utilisable. On parle aussi d'interfaces post-WIMP afin de trouver de nouvelles méthodes d'interaction. On se penche aussi sur les différents matériels et multimodalité comme dans les laboratoires de University of Maryland Human-Computer Interaction Lab[7], de UC Berkeley's User Interface Research Group's [8] et l'équipe *IIHM*[9] du Laboratoire CLIPS-IMAG de Grenoble. De très nombreuses conférences couvrent ces domaines de plus ou moins loin.

La visualisation interactive associe directement la représentation visuelle des données à l'utilisateur sans artefact supplémentaire. Ce dernier peut alors selon ses degrés de libertés virtuels effectuer une translation, zoom avant/arrière, rotation, homothétie, distorsion, filtrage.... Un lien dynamique avec l'information peut être crée, avec les requêtes dynamiques [AWS92] qui sont intéressantes pour réduire l'espace de recherche. Mais aussi sur la disposition. Les liens avec les données peuvent être automatiques, c'est à dire que les données répondent aux liens physiques qui ont été posés. Par exemple le logiciel *Touchgraph* [10] associe des propriétés dynamiques aux éléments à représenter. C'est rendre automatique l'interactivité de la visualisation, cette dernière convergeant vers un état stable. Cela permet de répondre à un besoin visuel global (une répartition homogène dans un plan) avec des règles de constructions primaires et locales (attraction/répulsion).

[5] http\penalty\@M://www.gd2005.org/
[6] http\penalty\@M://www.tulip-software.org/
[7] http\penalty\@M://www.cs.umd.edu/hcil/
[8] http\penalty\@M://guir.berkeley.edu/
[9] http\penalty\@M://iihm.imag.fr/site/
[10] http\penalty\@M://www.touchgraph.com/

Enfin on peut se poser la question sur les méthodes d'évaluation des interfaces, on distingue trois phases *avant* avec le *profiling*, *pendant* avec notamment l'*eyetracking* afin d'étudier qualitativement un échantillon d'utilisateurs, et *après* avec l'analyse de journaux (*logs*) ou traces d'utilisation. Des méthodes statistiques sont également possibles avec un grand nombre d'utilisateurs. On obtient des résultats étonnants [11].

1.4 Personnalisation

La personnalisation d'un environnement consiste à fournir à l'utilisateur un environnement approprié à son profil, c'est à dire le regroupement de ses préférences et de ses centres d'intérêts. De manière opérationnelle, la personnalisation peut être vue de plusieurs perspectives en fonction du domaine de travail. Par exemple, avec la reformulation de requêtes en bases de données. La première étape est la représentation du profil qui peut être ensembliste, sémantique ou multidimensionnel [TB06]. Ensuite il est nécessaire d'associer un profil à un utilisateur. C'est à dire instancier la représentation, par exemple en étudiant son comportant dans un système d'informations [JHK05]. Ou en tirant des informations de l'historique de la navigation peut s'avérer très utiles [Wex99]. Il est enfin nécessaire de faire évoluer le profil, comme l'adaptation au contexte qui est la capacité d'un système à pouvoir prendre en compte différents paramètres tels que les compétences, la langue, les déficiences physiques, la localisation géographique, le profil matériel[BH00]. C'est intéressant aujourd'hui avec l'apparition rapide de nouveaux matériels et la mobilité potentielle de systèmes personnels. L'utilisateur doit avoir une capacité de choix et de modification de son profil. Celle-ci peut être plus ou moins limitée, on parlera alors de *customisation* dès que les modifications seront localisées chez le client.

1.5 Autres domaines

Bien que nous ayons cerné les grandes lignes de Recherche sur lesquels nous allons nous positionner, nous pensons qu'il est possible d'affiner les problématiques en s'intéressant à des domaines dits *connexes*. Il est en effet toujours intéressant de regarder dans d'autres directions, et peut-être trouver de nouvelles approches, ou parfois la même, seulement sous un nom ou un formalisme différent. Cette méthode n'est pas nouvelle, de nombreux résultats scientifiques intéressants ont été obtenus en mélangeant des disciplines. Dans notre cas il s'agit plus d'une influence indirecte dans notre réflexion, qui n'est pas vraiment quantifiable.

La prise en compte de *masses de données* implique des algorithmes à complexité exponentielle qui demande énormément de temps de calcul. Mais la complexité est relative par rapport au niveau de granularité. Ainsi, nous devons effectuer du nettoyage, conversions et réductions de dimension des données : cela est présent dans le domaine de l'*ETL* (Extraction, Transformation et Chargement de données). Quant aux interactions complexes entre les données, nous nous rapprochons des *systèmes complexes* avec une approche souvent multi-variables, ce qui est réalisable par exemple avec des *réseaux de neurones artificiels*.

Ensuite, n'oublions pas qu'un utilisateur se trouve derrière l'écran. On peut exploiter son potentiel mnémonique et distinguer les *types de mémoire* : *sémantique* (acquisitions de mémoires générales sur le monde, par exemple les capitales de pays), procédurale (connaissances reposent par l'apprentissage par l'action, par exemple le musicien conscient de ce qu'ils fait, mais pas de la façon), de formes et structures des objets (visages et mots sans références à leur signification) et de travail (qui permet de maintenir temporairement une certaine quantité d'information). C'est un défi, entre autres, que l'on peut associer aux *sciences cognitives*.

De manière moins formelle, il y a des éléments de *design et d'ergonomie* [Nor02] qui peuvent nous intéresser : quels sont les critères de qualité d'interfaces visuelles ? à quoi correspond la beauté et l'attirance ? qu'entend-t-on par accès simple ou intuitif aux fonctionnalités ? Les *Arts visuels* combinent des approches interdisciplinaires et innovantes. Le Media Lab du MIT[12] propose des bribes d'interfaces souvent intéressantes en mélangeant défi technique et artistique. Enfin d'autres domaines plus imaginatifs comme la *science fiction* peuvent être inspirants et proposer des univers avant-gardistes, source de motivation inépuisable pour de nombreuses personnes. Cela est loin d'être fantaisiste de part souvent la capacité des auteurs à anticiper les dérapages d'utilisation (Exemple : le *Big Brother* de George Orwell), ou à anticiper les prochaines tendances. Notre intérêt étant ce qui est relié au

[11] http\penalty\@M://www.prweb.com/releases/2005/3/prweb213516.htm
[12] http\penalty\@M://www.media.mit.edu/

visuel, citons un film récent : *Minority Report*[13] où l'acteur principal (Tom Cruise) interagit manuellement avec une interface virtuelle pour naviguer dans des documents *visuels*.. à la manière d'un chef d'orchestre de *musique* symphonique.

1.6 Analyse et définition de la problématique

Nous avons désormais des éléments intéressants pour dessiner les directions à prendre afin de résoudre notre mission. Cependant, la couleur générale qui se dégage de cette première phase est un manque de formalisme généralisé. Certes la *visualisation* propose des représentations bien précises en fonction d'un contexte particulier, mais les seules classifications existantes sont effectuées selon le type de données [Chi00], la métaphore ou le thème de l'usage [14]. Il n'y a pas de classification intrinsèque des représentations visuelles qui permettraient de les rendre généralisables et associées à des règles de plus haut niveau. De même, en usage réel, l'*interaction* possède des contraintes beaucoup plus fines et subtiles, telles que la réactivité de l'environnement, qui est indispensable à la bonne navigation de l'utilisateur : celui-ci est exigeant, et surtout délicat à faire changer (pour exemple la persistance des *WIMP* apparues il y a plus de 30 ans) : il faut ainsi le mettre dans un cadre idéal ou à défaut de cela, s'adapter de manière immédiate. Par exemple, l'utilisateur navigue parfois par un aller/retour entre la page de recherche et les résultats : cela impose comme contrainte un temps de réponse très faible, de l'ordre du temps réel (c'est à dire que l'utilisateur ne perçoit pas le temps nécessaire à la réalisation de l'action).

Un exemple *symptomatique*. Il y a une distinction indispensable à faire entre visualisation et navigation. L'affichage des résultats d'une requête sur *Google* [15] est très souvent décrié pour sa représentation unidimensionnelle des résultats (classement par *PageRank*). Mais elle marche car elle est utilisée. En parallèle celle de *Kartoo* [16] semble plus pertinente car il y a une représentation visuelle planaire (2 dimensions) qui proposent des regroupements spatiaux. Mais il faut juste distinguer l'apport de Kartoo en termes d'environnement de Navigation et d'extraction de caractéristiques plutôt que moteur de recherche. Pour résumer il y a l'apport d'une nouvelle idée (2D) mais qui en fait perdre une meilleure qu'elle la pertinence de la recherche Google. Il faudrait donc une approche plus flexible pour quantifier l'apport d'une nouvelle interface. Distinguer ce qui relève de la métaphore ou simplement de la transformation de données.

L'erreur et l'incertitude n'apparaissent jamais ni dans la représentation des comportements, ni dans la visualisation, alors que ce sont des états fréquents dans un raisonnement. L'utilisateur ne doit pas être négligé dans ses défauts, et doit aussi être séduit en permanence : il y a des exemples de combinaisons de beauté qui ne font pas d'ombre à l'efficacité (les fractals par exemple [Man75]). Il faut donc remettre directement ou indirectement l'utilisateur au coeur des décisions, sans avoir à revenir à la longue étape de conception logicielle en amont.

Enfin de nouveaux usages émergent, tels que la sérendipité[17], provoqué souvent par angoisse de la page blanche ou trop forte complexité des environnements. Et d'autres notions plus floues encore arrivent, telles qu'intégrer la persistance et l'éphémère pour ne pas encore complexifier ou éveiller l'utilisateur. Ce sont d'ailleurs souvent des phénomènes que l'on retrouve dans des environnements immersifs, artistiques, mais qui hélas ne sortent jamais des musées.

Problématique

Créer un environnement de toute pièce est long, alors qu'il existe une production intéressante de diverses communautés : librairies, logiciels, infrastructures. D'autant que la conception logicielle de représentations visuelles demande des connaissances pointues, et pour l'utilisateur l'apprentissage d'une nouvelle interface est souvent long et rédhibitoire : il semble donc indispensable de réutiliser tout cet investissement de temps, aussi bien côté concepteur qu'utilisateur. Certaines règles sont ainsi à prendre en compte sont à comprendre et à analyser, dans

[13] Film de Steven Spielberg (2002) basé sur une nouvelle de Philip K. Dick.
[14] http\penalty\@M://www.visualcomplexity.com/
[15] http\penalty\@M://www.google.com/
[16] http\penalty\@M://www.kartoo.com/
[17] La sérendipité est le don ou la faculté de trouver quelque chose d'imprévu et d'utile en cherchant autre chose, ou encore, l'art de trouver ce qu'on ne cherche pas (définition Wikipédia).

l'optique finale de pouvoir décrire les connaissances impliquées et les bonnes pratiques – innovantes – disséminées actuellement dans de nombreuses communautés et logiciels expérimentaux.

Notre approche

Les interfaces évoluant vite, un moyen de lutter contre le temps est de créer des outils, des instruments. C'est à dire que nous n'allons pas créer une interface mais le moyen de la créer. Il existe en ingénierie deux approches opposées de conception : celle ascendante (dirigée par les utilisateurs ou l'usage) et celle descendante (dirigée par les concepteurs ou la technologie). Dans notre cas nous proposons plutôt une approche mixte. C'est à dire que l'on ne remonte ou l'on ne redescend qu'à un niveau intermédiaire, comme si respectivement on fabrique des briques de Legos^{TM}, ou bien on les assemble, mais on ne fait pas les deux. Nous nous intéresserons particulièrement à l'utilisateur en codifiant explicitant le savoir qui nous permettra de trouver la combinaison d'interfaces. Cela sera permis en complètent avec une analogie au modèle mental que l'utilisateur se fait de la perception d'un environnement. Située dans sa tête, cette perception est constituée de ses nouvelles expériences, mélangées aux anciennes avec un peu d'intuition ou d'innée. L'objectif final est d'arriver à caractériser ses schémas de navigation et les réinjecter dans le modèle afin de capitaliser, d'améliorer et de personnaliser à des profils d'utilisateurs-types.

Chapitre 2

Propositions

Nous allons d'abord définir un langage de description des interfaces visuelles et des environnements de navigation. Celui-ci comporte un vocabulaire et des règles à la fois assez précis pour décrire finement leurs caractéristiques, mais également assez souples afin d'être accessible et flexible pour l'utilisateur. Dès lors nous pourrons constituer une classification et effectuer des requêtes structurelles (combinaisons) et sémantiques (contenu) sur la collection d'environnements.

Ce langage servira également d'intermédiaire à une structure de données formelle et ses outils associés (définitions, théorèmes, algorithmes) permettant de trouver des éléments de réponse (formelle) au problème posé (Figure 2.1).

La dernière étape sera enfin d'établir le lien entre la réponse formelle à des situations réelles, c'est à dire définir les règles et les contraintes à imposer au modèle formel. Celles-ci permettront au final d'obtenir une interface répondant aux divers critères de construction auxquels nous souhaitons répondre.

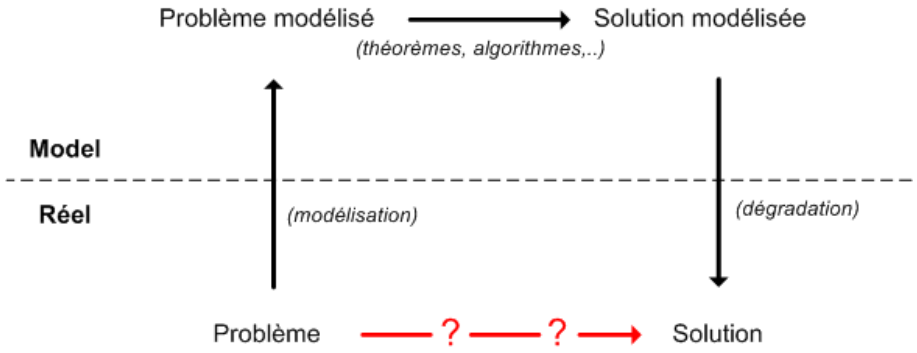


FIG. 2.1 – *De manière générale, la solution à un problème réel peut être trouvée en passant par le modèle de ce même problème. Toute la difficulté réside au début dans la traduction formelle du problème initial, et à la fin dans la dégradation de ce même modèle afin de faire émerger la solution réelle.*

2.1 Description des environnements

Décrire un environnement nous permet de cerner et d'isoler certaines de ses caractéristiques intrinsèques. Le but étant de pouvoir les décomposer et les classer par la suite, deux étapes de décomposition ont été proposées par [Vui06] [1] : d'abord séparer les visualisations reliées entre elles par des actions (système, utilisateur), ensuite décrire le comportement de l'utilisateur en se reposant sur celles-ci.

[1] Article court soumis le 15 avril 2006 à la conférence « *Ingénierie de la Connaissance 2006* » à Nantes. Cet article a été accepté et y sera présenté sous forme de poster le 28 juin 2006. Disponible à l'adresse http\unskip\penalty\@M://liris-7087.insa-lyon.fr/doc/IC06_vuillemot.pdf

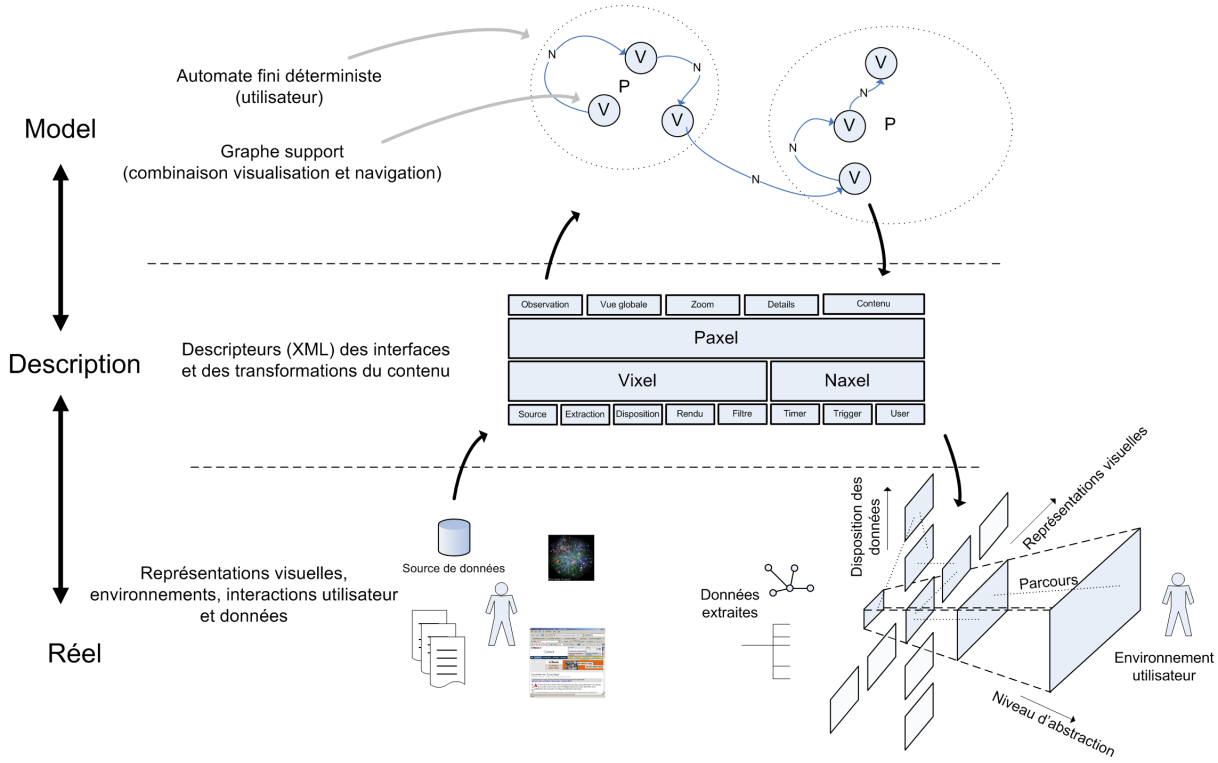


FIG. 2.2 – *Schéma global de notre proposition en se fondant sur la méthodologie présentée sur la Figure 2.1*

Même si nous pouvons facilement quantifier et reproduire la première, tout ce qui implique l'utilisateur – comme dans la seconde – tend vers un système composé d'interactions non-linéaires non-reproductibles, autrement dit différents paramètres sont intégrés dans le temps. Cela nous rapproche de la dualité (complémentaire) proposée par [dR77]. En s'inspirant de cette dernière, nous allons qualifier la première approche d'*analytique*, et la deuxième de *systémique*. [2]

2.1.1 Notre approche analytique : *Vixels* et *Naxels*.

Nous pensons donc qu'un environnement peut être décrit comme une combinaison de représentations visuelles combinées par des actions de navigation possible à décrire comme des processus reproductibles et distincts. Ce choix nous demande de scinder en deux parties les domaines sur lesquels nous avons décidé de nous baser.

Les représentations visuelles : tout ce qui touche le traitement, la disposition des données, le rendu, le traitement d'image est associé au processus de visualisation car il s'agit de mettre en place un univers à partir duquel on va prendre une vue et que l'utilisateur va subir.

Les actions de navigation : tout ce qui est interaction de l'utilisateur ou du système. A savoir capté par l'environnement de navigation. Elles permettent de reformuler, de mettre à jour les données, de mieux les disposer et ainsi effectuer un réarrangement plus adapté.

Comme dans [Vui06], nous parlerons alors respectivement de *Vixels* et *Naxels*. C'est par nécessité que nous introduisons ces deux néologismes inédits (et plus tard *Paxel*). Ils sont la composition de **VI**sualization **EL**ement, **NA**vigation **EL**ement et **PA**th **EL**ement. On connaissait déjà entre autres le *Pixel* (**PI**cture **EL**ement) et *Doxel* (**DO**cument **EL**ement).

[2] Cette dualité est plus profonde qu'elle n'y paraît. Il est fréquent de trouver des articles, comme ceux déjà cités dans ce rapport ([HMM00]) où les auteurs ne considèrent que la solution géométrique (analytique), par exemple, comme intéressante. Autrement dit il y a une totale elusion de l'être humain au profit d'une science pure et belle même si cela rend le problème quasiment irrésoluble.

Définition d'un *Vixel*. Un *Vixel* est un élément de visualisation. Il s'agit d'un descripteur de transformations à effectuer afin de réaliser une représentation visuelle de données fournies en entrées. Il permet de créer le lien entre la source de données (quelle information nous voulons représenter), leur extraction, disposition, rendu et filtrage (traitement d'image, filtres, segmentation). Chaque *Vixel* est associé à un type de surcharge pour l'utilisateur.

Les *Vixels* sont une photo statique de l'univers. Cela permet de fournir une représentation visuelle, de poser une question en attendant une reformulation/changement captée par des actions. Des *Vixels* possibles sont énumérés dans le tableau 2.1.

`Entrée : données structurées`	
Augmentation	Ex : regroupement, colorisation, historique.
Réduction	Ex : agrégation, baisse de résolution.
Disposition	Ex : arbre, réseau, disposition organique.
Rendu	Ex : génération d'image 2D, de modèle 3D.
Filtrage	Ex : flou gaussien, filtres .
`Sortie : image 2D, modèle 3D, metadonnées`	

TAB. 2.1 – *Voici différents types de Vixels en se basant notre état de l'art. L'augmentation permet de rajouter des données par rapport à celles présentes initialement (actions utilisateurs par exemple). La réduction de diminuer les dimensions ou sélectionner des données. La disposition associe une coordonnée dans un plan ou espace aux données. Le rendu produit une matrice de pixels (image) ou un graphe (2D, 3D). Enfin le filtrage permet d'effectuer un traitement d'image. Les Vixels peuvent être combinés entre eux pour obtenir des représentations plus complexes.*

Types de surcharges. Dans [BWK00] les auteurs énumèrent les différents efforts de l'utilisateur : *apprentissage* du système, *charge* sur la mémoire de travail, *comparaison* mentale et *changement de contexte*. De même le système doit répondre à un besoin *puissance de calcul* et d'*espace d'affichage*. Ce sont des surcharges propres aux représentations visuelles.

Définition d'un *Naxel*. Un *Naxel* est un élément de navigation. Ils sont associés aux actions de l'utilisateur et donc à un dispositif d'interaction d'entrée (souris, clavier), mais peuvent aussi être automatiques et déclenchés par le système (temporisateurs, règles dynamiques). A chaque *Naxel* est également associé un type de surcharge pour l'utilisateur.

Les *Naxels* sont plutôt une réorganisation de l'univers des données afin de les percevoir autrement. Voir le tableau 2.2.

`Entrée : image 2D, modèle 3D, metadonnées`	
Observation	Ex : déplacement (X, Y ou Z), rotation.
Édition	Ex : saisie et modification de texte.
Automatique	Ex : timer, trigger.
`Sortie : environnement de navigation`	

TAB. 2.2 – *Un classement de Naxels où l'observation ne modifie pas les données et sont des actions utilisateurs. Au contraire de l'édition qui modifie les données. Les Naxels automatiques sont intégrés dans les systèmes.*

Types de surcharges. Les surcharges associées aux actions, sont la complexité et l'effort demandé à l'utilisateur pour l'effectuer ou maîtriser le dispositif d'entrée permettant d'interagir avec le système. On parlera aussi de forte surcharge quand l'action ou la combinaison d'actions demandent une attention très forte de l'utilisateur.

2.1.2 Notre approche systémique : *Paxels*

L'utilisation seule de *Vixels* et *Naxels* pour décrire les représentations et environnements reste très limitée. Certes nous avons déjà capté une forme de connaissance (comme des assemblages potentiels et judicieux de techniques de représentations de données), mais notre problématique initiale nous impose d'aller plus loin. En effet nous devons prendre en compte des contraintes utilisateurs, comme l'historique de navigation, pour définir les règles que doivent respecter les environnements construits.

Définition d'un *Paxel*. Un *Paxel* est un élément de parcours dans un environnement de représentations visuelles. Autrement dit une combinaison de *Vixels* et de *Naxels*, qui sont reliés entre eux. Cela peut être au moyen de l'ajout d'une composante temporelle ou d'une relation d'ordre entre les *Vixels* et les *Naxels*.

Le raisonnement de l'utilisateur est un ainsi caractérisé en suite de *Paxels*, les questions qu'il se pose sont formulées en *Vixels*, et reformulées au moyen de *Naxels*. Voir le tableau 2.3.

Vue globale	Toutes les données sont visibles.
Vue détaillée	Détails d'une unité de sens.
Zoom	Transition entre deux niveaux.
Filtrage	Filtrage à l'aide de requêtes dynamiques.

TAB. 2.3 – *Exemples de Paxels d'après [Shn96]. A noté que le filtrage dynamique est souvent une demande explicite de l'utilisateur : il y a donc deux Naxels impliqués, le premier est une action utilisateur, l'autre une action (ou plutôt réaction) du système.*

Les caractéristiques systémiques des *Paxels* vient de l'intégration des *Vixels* et des *Naxels*. En effet, un *Paxel* intègre ces descripteurs avec leur relation entre eux, par exemple une composante temporelle. Ainsi un *Paxel* contenant les mêmes éléments *Vixels* et *Naxels*, mais à des instants différents, peut avoir un rôle opposé. Nous verrons un exemple dans la partie suivante.

Conclusion

Nous disposons de descripteurs de niveaux intermédiaires. Ils sont volontairement décrits de manière ouverte, car il s'agit d'une première approche totalement inédite dans nos travaux. Désormais, nous ne sommes plus dépendants du support (librairies, environnements techniques), et pouvons nous concentrer sur les agencements qu'il paraît judicieux de faire et les rendre réutilisables.

2.2 Modèle formel

L'objectif est maintenant de définir un formalisme, fondé sur les descripteurs précédents, et dont les propriétés et caractéristiques permettent de $i)$ spécifier la représentation visuelle, $ii)$ combiner les représentations en respectant des contraintes et $iii)$ modéliser le parcours de l'utilisateur. A des fins de simplification, la représentation doit être unique, c'est à dire deux navigations visuelles identiques doivent avoir une même représentation[3].

De manière évidente et intuitive, notre choix s'est porté sur un **Graphe Orienté Pondéré** où les états sont la visualisation, les arcs les navigations et les différentes contraintes d'ordre intégrées dans les poids. Nous pourrons calculer la complexité (théorique) des problèmes et utiliser des algorithmes optimaux (en termes de complexité) nécessaire dans notre cas où les combinaisons et contraintes à gérer sont importantes [CLR89].

En complément, nous modéliserons le parcours de l'utilisateur en **Automate Fini Déterministe** [LR01]. L'ensemble des états de l'automate est un sous-ensemble du graphe G. Il y a un état initial et un ou plusieurs états finaux. Des règles de transitions qui permettent de stocker la connaissance sur l'utilisateur et de modéliser un comportement-type (après avoir effectué différentes observations) que l'automate se doit de vérifier.

[3] Si plusieurs représentations étaient possibles, il faudrait effectuer un travail plus précis dans les descriptions et justifier les caractéristiques permettant de choisir concrètement l'une ou l'autre des représentations dans une situation donnée.

2.2.1 Graphe support de Visualisation et de Navigation

Définitions. Soit D l'ensemble des données que nous souhaitons représenter. V l'ensemble des représentations visuelles disponibles. N l'ensemble des actions possibles. Un graphe dit support de Visualisation et Navigation est un graphe $G = (V, N)$ orienté pondéré muni d'une fonction $w(u, v)$ de pondération qui associe aux arcs $(u, v) \in N$ un poids (positif ou nul). Chaque état $v \in V$ est fonction d'un jeu de données (une même visualisation peut représenter plusieurs données différentes) défini par une fonction de transformation $Trans$ de $D \rightarrow D$ qui à chaque jeu de donnée $d \subseteq D$ en associe un autre $d' \subseteq D$ (on suppose que toutes les caractéristiques des données sont contenues dans D, il faut juste bien les sélectionner). Il existe différentes fonctions de transformation des données : $Abstr$ d'abstraction, Red de réduction, Aug d'augmentation. $Abstr$ est une fonction par niveaux $i \in N$. Soit $Visu$ une fonction de $D \rightarrow V$ telle que $v_d = Visu(Trans(d))$ avec $d \subseteq D, v_d \in V$. Exemple, $v_{d_i} = Visu(Abstr(d, i))$ est une représentation visuelle à laquelle on peut associer une action de transition $n_t \in N$ vers un autre niveau j d'abstraction v_{d_j}.

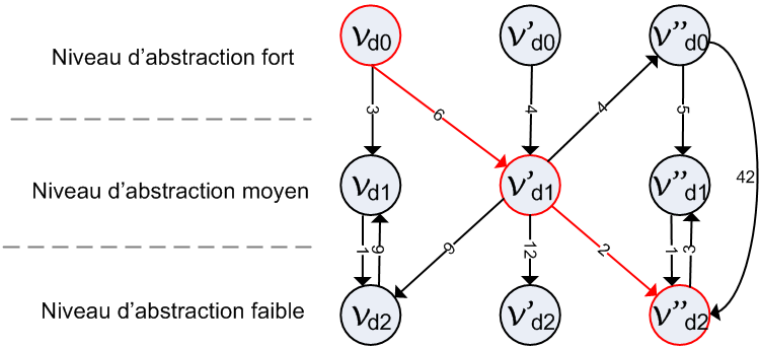


FIG. 2.3 – *Graphe orienté pondéré support. On voit que le parcours $\{v_{d_0}, v'_{d_1}, v''_{d_2}\}$ minimise la somme des poids : cette combinaison d'interfaces visuelles est donc optimal au sens des critères que l'on a défini pour pondérer le graphe. A noter que pour des raisons esthétiques nous n'avons pas représenté tous les arcs du graphe (complet).*

Le problème de visualisation de données. Il s'agit de trouver le bon état v. Autrement dit un sous-ensemble de données $d \subseteq D$, une fonction (ou composition de fonctions) de transformation et une représentation visuelle $v \in V$ adéquats.

Le problème de navigation visuelle dans des données. Se formule ainsi : trouver un chemin judicieux dans G minimisant le coût (somme des poids). Un chemin étant une combinaison interopérable de visualisation.

Construction du graphe. Le graphe support est complet, c'est à dire que tous les sommets sont reliés par une arête. Ainsi toute la difficulté repose dans la définition de $w(v_i, v_j)$ fonction de pondération qui intégrera les paramètres de tâche, d'utilisateur, de taille de jeu de données ou de complexité, pour définir le passage d'une visualisation v_i à v_j).

Quelques autres propriétés intéressantes du graphe.

- La **fermeture transitive** du graphe est l'ensemble des chemins possibles. Si on supprime les arcs dont le poids est supérieur à un certain seuil, la calculer revient à voir toutes les combinaisons de représentations visuelles possibles en dessous de ce seuil.
- La recherche du **plus court chemin** revient à minimiser le nombre de représentations intermédiaires entre deux représentations visuelles disponibles. Il ne suffit pas de sauter des noeuds, car le poids est souvent élevé. Intuitivement cela s'explique qu'une succession d'interfaces est nécessaire et non réductibles pour que l'utilisateur comprenne le cheminement successif qui lui permettent de suivre son raisonnement (induction, déduction) ou de bien garder sa position dans l'univers des données. Exemple : si on part de la terre vue de l'espace on veut zoomer progressivement (jusqu'à la ville que l'on veut visualiser) afin de garder le contexte,

dans ce cas le pays, la région. Dans l'exemple de la Figure 2.4, il est dissuasif de passer directement de v''_{d_0} à v''_{d_2} sans passer par v''_{d_1}.
- Le graphe support est toujours **connexe**. C'est un choix, car nous pensons qu'il est toujours possible de passer d'une représentation visuelle à une autre, même si c'est incohérent, car cette notion est très subjective. Deux représentations visuelles sans aucun lien dans une situation donnée peut en avoir un dans une autre. Et s'il n'est pas toujours possible de le faire automatiquement, l'utilisateur devra alors changer manuellement de représentation (avec la souris, ..) : c'est tout simplement un *Naxel* avec une forte surcharge pour l'utilisateur.

Conclusion.

Utiliser des vecteurs (plusieurs dimensions) au lieu du poids (actuellement une seule variable intègre tous les paramètres). Les complexités des différents algorithmes évoqués sont également à étudier, ainsi que la taille dans des cas réels du graphe support (un problème de combinatoire).

2.2.2 Utilisateur modélisé en automate fini déterministe

Définition. Formellement, un automate fini déterministe (AFD) est un quintuplet : (P, Σ, N, s, F)
- (P) un ensemble fini d'états. Les états sont un chemin *Vixels* et *Naxels*, associé au modèle mental de l'utilisateur, c'est à dire qu'ils représentent l'activité de l'utilisateur dans son raisonnement (observation, approfondissement).
- (Σ) un ensemble fini de symboles ou alphabet que l'automate accepte (règles vérifiées). Par définition, il doit y en avoir une chaque lettre de l'alphabet.
- $(N : P \times \Sigma \rightarrow S)$ une fonction de transition. Les transitions d'un état à un autre sont des *Naxels* (actions) dits de *transition*, c'est à dire une étape notable du raisonnement de l'utilisateur.
- $(p_{init} \in S)$ un état de départ. C'est la première étape du raisonnement de l'utilisateur. Nous pouvons penser qu'il doit contenir des *Vixels* très attractifs visuellement pdarlant, pour séduire l'utilisateur. Par la suite l'attractivité devra laisser place à l'efficacité pour être performant et ne pas distraire l'utilisateur.
- $(F \subseteq S)$ un ensemble d'états acceptant. L'utilisateur trouve l'information qu'il cherchait ou quitte le système.

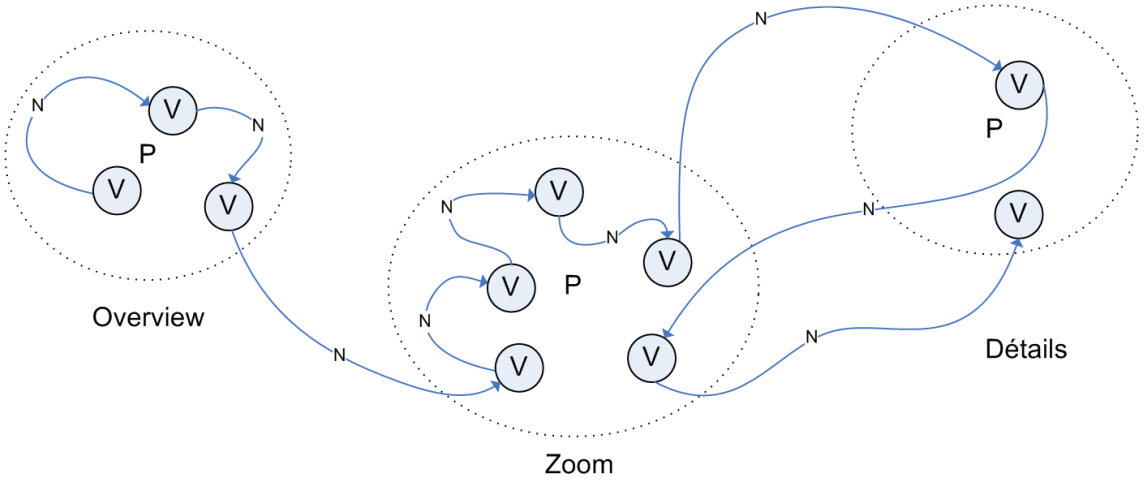


FIG. 2.4 – *L'automate se base sur une classification d'un sous-graphe du graphe support. Cela permet de regrouper de manière souple le parcours de l'utilisateur, dans chaque état de l'automate auquel on va donner une signification (stockée dans le nom de l'état).*

Construction de l'automate. La difficulté réside dans l'intégration des différentes *Vixels* dans chacun de ses états. En effet, parfois l'état de l'automate reste identique (autrement dit la tâche qu'on effectue est la même), même si on change de représentation visuelle. Cela permet de modéliser de manière flexible le comportement

de l'utilisateur qui peut changer de représentation mais pour toujours effectuer la même tâche. La classification des *Naxels* sera elle aussi nécessaire : certains *Naxels* ne changerons pas l'état en cours, tandis que d'autres oui (fonction de transition).

Le problème parcours d'utilisateur. Il s'agit de trouver un parcours adéquat pour un utilisateur. Entant donné des recommandations (préférences pour certains états), c'est la recherche d'une suite de visualisations et d'interactions classées combinées selon ces états. A savoir un chemin dans G annoté du temps ou de l'ordre de parcours.

Exemple. Reprenons de manière simplifiée les deux stratégies de visualisations évoquées dans la première partie sous forme de *Paxels* : *overview, zoom and details* et *focus and context*. Nous souhaitons vérifier automatiquement qu'une interface répond à ces règles, c'est à dire qu'il est possible de valider les *Paxels*, à une combinaison de *Vixels* et *Naxels*. La première étape sera de partir de l'interface et la décrire en *Vixels* et *Naxels*. Ensuite effectuer une classification en *Paxels*. Et enfin si l'automate construit vérifie le «langage» initial, à savoir les combinaisons de *Paxels*.

Conclusion.

Toute la difficulté reste désormais à ajouter de la connaissance à ce modèle, comme reconnaître les classifications ou parcours pertinents. D'autres approches peuvent être utilisées en ajoutant des probabilités et ainsi en simulant le parcours de l'utilisateur : on obtiendra un automate fini *probabiliste*. La représentation en chaînes de Markov semble adaptée ou en réseaux de neurones artificiels qui permettent d'intégrer de manière flexible et de propager plusieurs contraintes. D'autres représentations (réseaux de pétri, model checking, logique temporelle) mais pas assez de temps pour comparer et en déduire la pertinence de chacun des modèles dans notre contexte.

2.3 Personnalisation et adaptation d'un environnement

Il est désormais nécessaire de définir précisément à quelles contraintes doivent se soumettre le modèle spécifié ci-dessus. Nous allons les formuler sous forme de règles définissant un environnement adéquat. Il y a autant de règles que de points de vue sur l'application. Mais heureusement notre séparation initiale et caractérisation précise des environnements nous permet désormais de les réorganiser de manière flexible. Par exemple si l'on se focalise sur les données, on peut réorganiser les interfaces selon des *Vixels* adaptés à ces mêmes données.

2.3.1 Définition des profils

Profils utilisateurs. Il es défini sous forme de *Vixels* : $V_u = \{v_1, .., v_n\}$, de *Naxels* $N_u = \{n_1, .., v_m\}$ et de sélections de données (favoris par exemples) $D_u = \{d_1, .., d_n\}$. Le choix de ces ensembles peut être établi selon l'historique de navigation (chemin dans le graphe) et les besoins (source, transformation et visualisation de données) formulés sous forme de *Paxels* $P_u = \{p_1, .., p_n\}$.

A noter que la définition de profils peut être faite sous forme d'*expressions rationnelles* permettant de d'écrire un ensemble de règles plus ouvertes à respecter : par exemple la règle `{overview, *, details}` permet de construire ou vérifier qu'un automate, à savoir une combinaison d'interfaces et d'actions, commencent toujours par la vue globale et qui finissent toujours par la vue détaillée. Peut importe les transitions intermédiaires.

Le problème personnalisation d'un environnement de navigation C'est la recherche d'un chemin dans G satisfaisant les contraintes du profil utilisateur. Autrement dit, le choix d'une combinaison de *Vixels* v et de *Naxels* n qui répondra à ses besoins potentiels. Les autres *Vixels* et *Naxels* du graphe support sont toujours disponibles mais leur accès n'est pas favorisé (l'utilisateur n'est pas censé en avoir besoin). Les profils peuvent évoluer selon les préférences de la situation, et le graphe support peut voir ses poids évoluer pour intégrer cette forme connaissance.

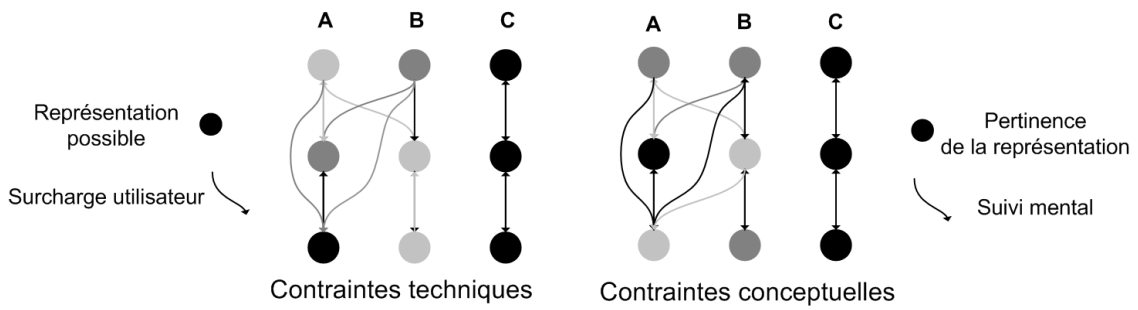


FIG. 2.5 – *En gris clair sont représentés les poids faible (interopérabilité grande) et en plus foncé, les poids les plus forts. Ne sont pas représenté les poids infini (pas d'interopérabilité du tout entre les interfaces). De même chaque Vixel est coloré en fonction sa capacité à représenter un jeu de données. Nous avons ainsi une vue rapide de la description d'interfaces et des capacités à représenter un jeu de données ou à se combiner. Par exemple la A et la B semblent complémentaires, alors que la C est marginalisée.*

2.3.2 Réorganisation par contraintes

Une réorganisation des environnements consiste concrètement à trouver le meilleur parcours selon les objectifs initiaux, notamment en respectant les profils préalablement définis. Les contraintes se basent à deux niveaux : dans le support de base (graphe G support) et sur l'automate. Dans le premier cas on parlera de réorganisation descendante, c'est l'usage qui conditionne la succession de représentations visuelles. Le deuxième cas présente une réorganisation ascendante, ce sont les contraintes techniques qui conditionnent. La difficulté est de mélanger les deux approches indispensables (une interface qui marche doit être intéressante, tout comme une interface intéressante doit marcher). Voyons l'approche *ascendante*.

Au niveau du graphe support, les contraintes peuvent être par exemple de mobilité, de changement de dispositif d'affichage, de résolution d'écran. Il faut ainsi réorganiser mettre à jour les poids du graphe support afin que les parcours utilisateurs restent cohérents. Parfois les changements ne se font qu'au niveau des *Vixels*, s'il s'agit d'un choix des paramètres adaptés (couleurs par exemple) chez le client (il n'y a plus d'interactions avec le serveur). On parlera de customisation de représentations visuelles (pas de changement de structure ni de pondération du graphe support). Voyons maintenant l'approche *descendante*.

2.3.3 Stratégies de réorganisation

On entend par stratégie la sélection judicieuse de paramètres à intégrer (certains étant prioritaires) lors du parcours du graphe support G. Ces derniers correspondent à des règles définies par les concepteurs et experts en interfaces visuelles (ou d'autres domaines connexes) ou par les besoins d'un utilisateur. Voici le cas d'étude d'une stratégie possible, et quelques éléments de conversion en *Vixels*, *Naxels* et *Paxels*.

Stratégie de changement de réactivité du système. L'utilisateur souhaite visualiser des données situées sur un serveur distant, dans son environnement de navigation local au moyen d'une connexion réseau. la règle est que l'environnement soit toujours réactif, c'est à dire que l'utilisateur doit en permanence avoir le contrôle (ou le sentiment comme quand on zoom on voit des pixels car l'image n'est pas mise à jour). En cas de coupure ou baisse du débit, l'utilisateur doit être perturbé au minimum. Dans le cas d'une phase d'observation (faible interactivité) l'utilisateur peut toujours se satisfaire d'une vue globale, non mise à jour. Par contre s'il est dans une phase d'exploration, il faut baisser la résolution des images, ce qui lui permet de conserver sa position dans le modèle avec la mise à jour des données.

Conversion en *Vixels*, *Naxels* et *Paxels*. Tout d'abord, notre hypothèse de séparation de visualisation et de navigation est respectée (l'environnement de navigation est installé chez le client, quand les représentations visuelles sont générées et transférées depuis un serveur distant). Ensuite, si il y a une baisse de débit du réseau nous souhaitons pouvoir passer facilement d'un *Vixel* de haute résolution à un autre de résolution moyenne (autrement dit avec un *Naxel* de faible surcharge voir invisible pour l'utilisateur). Ou alors garder le même *Vixel* (car chargé

sur le client avant le changement de débit) mais le représenter de manière différente, en appliquant par exemple un traitement local au *Vixel* (zoom sur des pixels). A la condition d'être dans un *Paxel* d'observation où par définition la pertinence des données n'est pas prioritaire. Savoir quelle règle employer dans une situation est possible via le profil de l'utilisateur (un choix explicite de celui-ci ou sa réaction précédente dans une situation voisine, par exemple).

Chapitre 3

Réalisations pratiques

« Inventing something once is Genius.
Inventing something twice is stupidity »

Les réalisations ont pour but d'évaluer les propositions théoriques, afin de valider ou non nos hypothèses. Les critères d'évaluation habituels qui permettent de positionner un travail dans notre domaine, sont les validations par l'usage qui confrontent un panel d'utilisateurs, à l'environnement construit. Le panel doit être homogène (capacités techniques) et il faut aussi vérifier qu'ils n'ont pas d'anomalies visuelles, trier les réponses pertinentes et offrir des conditions stables et optimales. Au final, ces validations sont longues à préparer, effectuer et analyser. Une alternative est de remplacer les tests utilisateurs par des experts designers visuels [KHI+]. Ceux-ci permettent de rapidement évaluer des propositions et suggérer des améliorations constructives.

Cependant nos hypothèses à valider ne se situent pas dans ce cadre : nous souhaitons uniquement vérifier des faisabilités techniques. Les trois étapes sont :

1. *La description de représentations et environnements visuels* au moyen d'une implémentation des concepts de *Vixels*, *Naxels* et *Paxels*.
2. *L'interopérabilité des programmes*, à savoir faire fonctionner ensemble automatiquement et sans perte de données (métadonnées surtout) un assemblage de librairies graphiques utilisant des structures de données différentes.
3. *La combinaison de visualisation et d'interactions* qui permet à l'utilisateur de spécifier une représentation visuelle indépendante de son environnement de navigation local.

Notre politique de validations est la confection de *preuves de concepts*, à savoir des outils simples afin de pouvoir valider une hypothèse intermédiaire ou une idée originale. Chaque petit prototype aura ainsi le mérite de montrer très clairement à quoi il sert, son potentiel et ses limites. Les résultats obtenus (journaux, temps issus de bancs d'essais) seront stockés sur les descripteurs respectivement crées (*Vixels*, *Naxels* et *Paxels*). L'idée est de capitaliser dans ces éléments tous les résultats obtenus, afin de leur confier un historique et pouvoir tracer, analyser et disséquer à posteriori les différentes évaluations.

3.1 Langages de description `.vxl`, `.nxl` et `.pxl`

Il s'agit d'implémenter les *Vixels*, *Naxels* et *Paxels* dont nous avons donné la définition préalablement. Rappelons qu'il s'agit de descripteurs de ressources et leur assemblage, ils doivent alterner entre précision de description et généricité. Nous souhaitons les décrire dans un format de fichier standard, pour faire en sorte qu'ils soient accessibles facilement (contrairement à une base de données où il faut effectuer des requêtes pour extraire les données). Nous utiliserons un langage du type XML, idéal pour écrire et échanger des informations. Le format proposé répond à des contraintes de lisibilités (notamment dans le choix éléments/attributs).

`.vxl`	*Vixel* général.
`.source.vxl`	Pointeur vers la source de données.
`.extraction.vxl`	Opérations d'extractions et transformation de données.
`.layout.vxl`	Disposition des données.
`.render.vxl`	Rendu des données sur un plan 2D ou dans espace 3D.
`.filter.vxl`	Opérations de traitement d'image (flous gaussien, ..).

TAB. 3.1 – *Sous-fichiers composant un Vixel. Leur syntaxe est la même que pour un Vixel.*

La représentation des descripteurs peut ainsi se faire de manière visuelle (par application d'une feuille de style), sous forme de design symbolique (voir image de gauche sur la couverture) qui permet une compréhension facile et intuitive du potentiel d'un environnement. Un *Vixel* est spécifié sous le format de fichier `.vxl` décomposé en sous-fichiers présentés dans le Tableau 3.1. Un `.pxl` contient un *Paxels* ou une combinaison de *Paxels*. Les fichiers *.nxl* doivent être encore approfondis. Les spécifications des DTD sont à effectuer.

Listing 3.1 – Exemple de `.pxl` (*Paxel*)

```
<paxel>
  <name>Exploration de données</name>
  <description>De la vue globale aux détails</description>
  <transition from="overview" to="zoom">
    <naxel type="trigger" mode="auto">
      <exec>
        <var name="prev" value="$vixel.overview.layout.altitude">
        <var name="next" value="$vixel.zoom.layout.altitude">
        <if>"$prev = $next"<if>
      </exec>
    </naxel>
      </transition>
  <layer name="zoom">
    <vixel>zoom.vxl</vixel>
    <naxel>tmr.timer.auto.nxl</naxel>
    <naxel>tgr.trigger.auto.nxl</naxel>
  </layer>
</paxel>
```

Listing 3.2 – Exemple de `.vxl` (*Vixel*)

```
<vixel type="vixel" id="Dd9eb59aecd2c">
        <name>Un Vixel</name>
        <description>Vixel parent de sous vixels</description>
        <vixel>
                <type>source</type>
                <file>src.source.vxl</file>
        </vixel>
        <vixel>
                <type>extraction</type>
                <file>ext.extraction.vxl</file>
        </vixel>
        <vixel>
                <type>layout</type>
                <file>lyt.layout.vxl</file>
        </vixel>
        <vixel>
                <type>render</type>
                <file>rnd.render.vxl</file>
        </vixel>
        <vixel>
                <type>filter</type>
                <file>flt.filter.vxl</file>
        </vixel>
</vixel>
```

3.2 Notre serveur de visualisation à la demande *VizOD*

Afin de répondre à la fois à la fois à une contrainte de l'environnement de l'utilisateur (lui imposer le moins de contraintes possibles) et aux limites logicielles soulevées dans l'état de l'art, nous allons centraliser nos efforts sur un serveur. Nous pensons que par la suite il sera judicieux d'effectuer une phase de décentralisation, entre autre pour des questions de confidentialité et d'optimisation afin d'exploiter la puissance de calcul actuel des stations de travail.

La description des représentations visuelles des données se fait chez le client ou sur le serveur (interface web) alors que le rendu est quant à lui réalisé exclusivement sur le serveur. Toutefois on peut estimer que les temps de calculs ou la confidentialité de l'information nous incite à réaliser certaines tâches chez le client. Notre serveur nous permet un cadre précis[1] d'évaluation technique (installation) mais aussi de benchmarking (calcul de vitesses d'exécution).

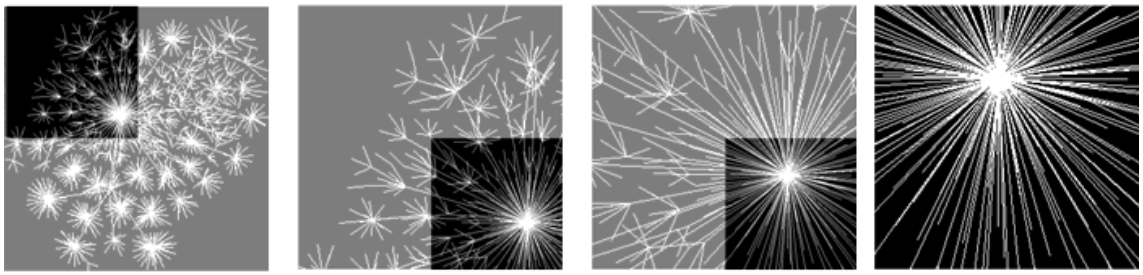


FIG. 3.1 – *Nous avons généré la représentation visuelle du répertoire `/etc/` du serveur. Ce répertoire contient 2 353 fichiers et 67 répertoires et sous-répertoires. La disposition 2D prend 153s avec la librairie LGL. Il y $n = 4$ niveaux d'abstractions, soit $N = 2^{2n} = 64$ image à générer. Chaque génération d'image prend en moyenne 0.8s, la disposition initiale restant la même le coût final est de 63,8s (il y a aussi eu un traitement de flou gaussien pour améliorer le rendu final des images de moyenne 0.1s). La génération peut donc quasiment se faire en temps réel (il faudrait aussi prendre en compte le temps de latence du réseau) en optimisant les zones à générer.*

Librairies installées

- LGL (Java) version 1.1
- Graphviz (C++) version 2.8-1
- Tulip (C++) version 2.0.4-1

Le choix des librairies dépend des fonctionnalités, mais aussi du dynamisme de la communauté autour, des mises à jour. Elles sont OpenSource également pour pouvoir comprendre le code, le modifier. Elles ont été rendues interopérables par la conversion de leur langages propres (`.dot`, `.lgl`) en un langage commun le plus exhaustif possible (dans le nombre d'attributs stockés), GraphML. Nous souhaitons finaliser ce prototype afin de fournir un serveur centralisé de génération de représentations visuelles (*Vixels*), doté d'une interface distante (page web) sous la forme générale d'un flot de données, mais plus particulièrement adapté aux états intermédiaires de la représentation visuelle, comme spécifiés dans le *Data State Reference Model* [Chi00]. Et aussi permettre un traitement interactif (l'utilisateur peut voir les données selon l'angle qu'il veut) de données à des fins de data mining [Tho].

[1] C'est une machine sous Linux (distribution Fedora/Red Hat 4.0.2-8) avec un noyau 2.6.16, 512 Mo de Ram, un processeur AMD Athlon™XP 2500+ (1.8 GHz) doté de 512 KB de mémoire cache.

Tableau de bord Cela permet d'avoir une vision instantanée sur les librairies installées, leur interopérabilité, statuts, et diverses statistiques : fréquences utilisation, jeux de données récurrents,.. . Ainsi pourquoi ne pas en déduire des règles selon les usages les plus fréquents.

3.3 Intégration de *VizOD* dans des environnements existants

Dans l'optique de valider l'hypothèse de séparation puis de recomposition de la visualisation et de la navigation, nous allons assembler des environnements avec des représentations visuelles, de manière inédite. Notre choix s'est porté sur deux types d'environnements, 2D et 3D, dans lesquels nous allons pouvoir interagir.

3.3.1 Google Earth/Map

Google Earth (GE) [2]est un outil Gratuit (pour un usage personnel) accessible à des non-experts. Il s'agit d'un globe virtuel sur lequel il est possible de superposer géographiquement plusieurs couches (images et données). GE fournit un langage spécifique, KML[3] (Keyhole Markup Language) pour modéliser une ou plusieurs propriétés, dont celle qui nous intéresse en particulier et qui est la superposition d'une image ou d'une étiquette à un lieu géographique précis (défini par sa longitude, altitude et latitude). Ce n'est pas une représentation géographique du monde, mais une représentation des données du monde de manière géographique.

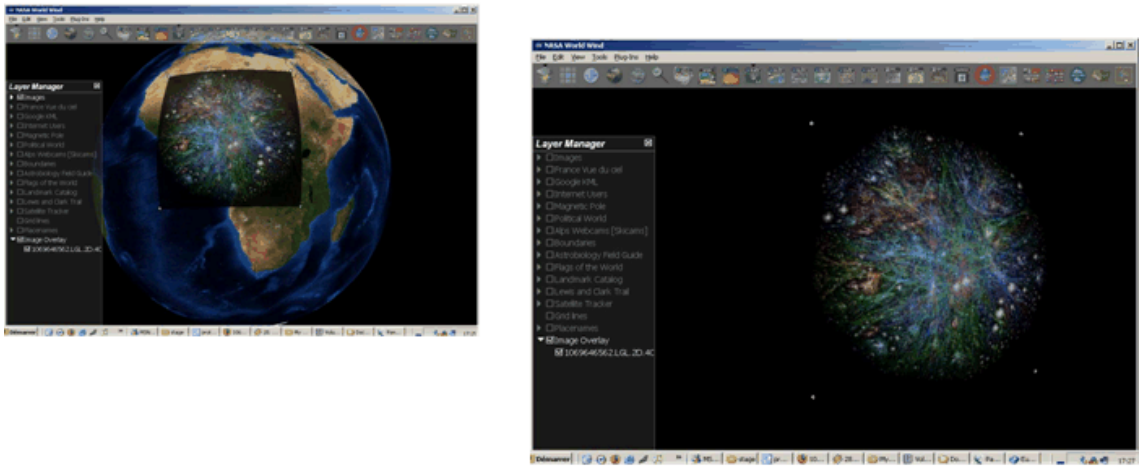


FIG. 3.2 – *A gauche la superposition d'une représentation visuelle générée à partir de* VizOD *et rapatriée automatique par GE afin de la disposer à un endroit précis sur le globe terrestre. Les couches représentant les données géographiques y sont encore visibles, mais supprimées sur l'image de droite. Un zoom permet de se rapprocher des données représentées et GE s'occupe automatiquement de télécharger l'image adaptée au niveau d'abstraction (ou altitude) que l'on a par rapport aux données. Des métadonnées (textuelles) sont également superposables à ces images disposées sur le globe.*

Google Maps[4] est un programme très simple écrit en javascript et accessible depuis n'importe quel navigateur. Il ne demande aucune installation de plugin et répond donc à notre cadre utilisateur initial. Une démonstration est disponible[5].

3.3.2 LaTeX

Combiner LaTeX avec des représentations visuelles peut paraître surprenant car un document texte de quelques dizaines de pages n'est pas un grand ensemble de données qui pose beaucoup de contraintes de navigation. De

[2] http\penalty\@M://earth.google.com/
[3] http\penalty\@M://earth.google.com/kml/kml_toc.html
[4] http\penalty\@M://maps.google.com/
[5] http\penalty\@M://liris-7087.insa-lyon.fr/

plus, la *métaphore* du livre est bien maîtrisée et le rend facilement exploitable avec un accès hiérarchique au contenu (couverture, sommaire, numéro de page), comme illustré sur la Figure 3.3 (l'image originale est issue du projet *Opte*[6]).

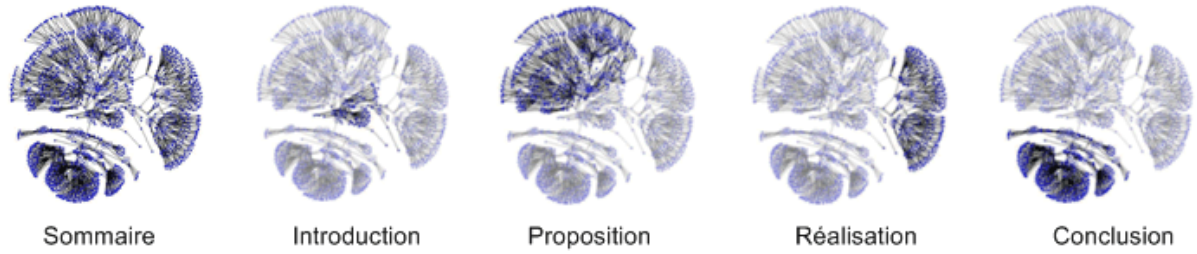


FIG. 3.3 – *L'idée est de générer une image permettant de représenter visuellement la vue globale des chapitres d'un rapport. Certes nous perdons une information sur la sémantique, à savoir le contenu des chapitres comme les titres ou les numérotations, dans les images. Mais nous y gagnons ailleurs, comme dans la rapidité de voir les proportions du rapport, quelle partie est plus consistante qu'une autre. Cette transformation est facilement traduisible en* Vixels*, combinaison de programmes existants (source : .tex, extraction : LaTeXML, disposition et rendu : LGL et enfin des règles de traitements d'images pour rendre des zones plus ou moins visibles). Mais cette séquence d'images a été générée manuellement (par manque de temps).*

Par contre, si l'on baisse le niveau de granularité, en passant du document au mot, un livre peut être vu comme un grand ensemble de données dans lequel il est complexe d'avoir une vue *globale immédiate*. De plus avec BibTeX on peut effectuer une représentation pertinente des citations, aussi bien selon leur fréquence dans le rapport que leur hiérarchie dans une communauté. Il est nécessaire d'établir la DTD et le schéma XML pour le modèle contenu dans BibTeX. Hélas, la réalisation technique complète n'a pas aboutit, par manque de temps.

[6] http\penalty\@M://www.opte.com/

Conclusions et perspectives

« I didn't have to take the normal classes,
I decided to take a calligraphy class to learn how to do this.
I learned about serif and sans serif typefaces,
about varying the amount of space between different letter combinations,
about what makes great typography great.
It was beautiful, historical, artistically subtle
in a way that science can't capture, and I found it fascinating.
(..) And we designed it all into the Mac. »
Steve Jobs [7]

Nous avons, dans ce rapport, proposé des éléments de solution à un problème auquel nous sommes directement ou indirectement confrontés tous les jours : explorer de grands ensembles de documents. Et cela au moyen de deux approches :

Une première approche descriptive des caractéristiques et des contraintes visuelles et interactives au moyen de descripteur que sont les *Vixels* et les *Naxels*. Leur flexibilité est telle, que l'on peut effectuer des combinaisons efficaces pour résoudre des contraintes assez subtiles.

Une seconde approche intégrant les contraintes se greffe dessus la première, en permettant une réorganisation des représentations visuelles en fonction de connaissances captées chez l'utilisateur (son comportement, ses préférences) et qui sont bien plus implicites que nos méthodes formelles. Mais justement, c'est leur puissance, en étant peut-être plus proche de la réalité, c'est à dire d'interactions compliquées, systémiques.

Nous avons testé les propositions grâces à notre serveur de visualisation *VizOD* et intégré dans des environnements quasi-habituels pour l'utilisateur. Quant à certains points qui n'ont pas été explicitement abordés dans ce rapport, notamment les problématiques propres aux données multimédias (hétérogénéité des documents, temporalité), il s'agit finalement de contraintes supplémentaires à décrire et à rajouter dans les différentes stratégies que le modèle peut respecter.

Quelques perspectives techniques sont intéressantes. Par exemple proposer notre propre *Service web de visualisation* : rendre le serveur de *VizOD* de «visualisation à la demande» interopérable afin qu'il puisse échanger des messages et être découverts et couplé avec d'autres applications.

De même des communautés peuvent se créer sur le partage de combinaisons successives d'interfaces afin de mettre en avant une ou plusieurs caractéristiques des données, échangées et capitalisées sur des *Vixels*, *Naxels* et *Paxels*. De même, un guide de bonnes pratiques pour mettre au point un moyen concis de représenter la connaissance contenue dans la plupart des environnements de navigation est à établir. Il s'agit d'établir un référentiel consensuel à la fois des problèmes, des différentes approches pour les résoudre et des solutions lors de la navigation dans des contextes très précis. Mais ce qui fait un peut l'intérêt de notre proposition, c'est sa rapide prise en main et donc elle ne doit pas devenir une architecture trop complexe qui sera inutilisable et pire encore, inutilisée [Sta].

D'autres approches mélangent des domaines complémentaires : les fractales [Man75] répondent idéalement aux critères de conceptions d'un bon environnement. Elles ont ce côté inspirant, qui capte notre attention dès le premier coup d'oeil. Elles possèdent un univers perpétuellement complexe et donc un grand espace qui peut contenir l'information par multi-résolution [KY93]. La similarité de structures à différentes échelles nous permet

[7] CEO of Apple, 'You've got to find what you love,' Stanford Commencement speech. http\unskip\penalty\@M://news-service.stanford.edu/news/2005/june15/jobs-061505.html

de ne pas être perdus localement, en retrouvant des formes que l'on a su apprivoiser globalement. Ainsi il faut s'intéresser à un langage simple de description des représentations visuelles complexes des données à l'aide de grammaires de formes très simples[8]. Un premier pas vers la navigation fractale ?

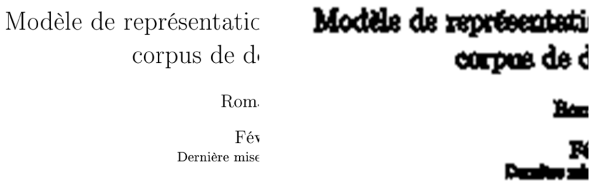


FIG. 3.4 – *A gauche une image vectorielle, à droite bitmap (la faible résolution fait que les pixels sont visibles). Notre modèle ne tient pas compte directement de cette différence dans nos descripteurs, mais elle peut émerger indirectement au moyen de l'apprentissage.*

Pour conclure, nous n'avons pas la prétention d'instrumentaliser et d'harmoniser la connaissance, juste de lever au maximum les verrous techniques qui sont une forme de barrière à l'expression de la créativité et donc de l'innovation. Notre approche possède des limites dans sa capacité de décrire, à la fois de ce que l'on ressent, appelé *sensibilité* chez l'être humain, si subtile et propre à chaque individu, mais également des limites dans ce que l'on perçoit et ce que *« [what] science can't capture »*...

[8] http\penalty\@M://www.contextfreeart.org/

Bibliographie

[ADM+04] Adel Ahmed, Tim Dwyer, Colin Murray, Le Song, and Ying Xin Wu. Wilmascope graph visualisation. In *INFOVIS*, 2004.

[Aub03] D. Auber. Tulip : A huge graph visualisation framework. In P. Mutzel and M. Jünger, editors, *Graph Drawing Softwares*, Mathematics and Visualization, pages 105–126. Springer-Verlag, 2003.

[AWS92] Christopher Ahlberg, Christopher Williamson, and Ben Shneiderman. Dynamic queries for information exploration : An implementation and evaluation. In *Proc. ACM Conf. Human Factors in Computer Systems, CHI*, pages 619–626, 3–7 1992.

[Ber83] Jacques Bertin. *Semiology of graphics : diagrams, networks, maps*. University of Wisconsin Press, Madison, WI, 1983.

[BH00] Jay Budzik and Kristian J. Hammond. User interactions with everyday applications as context for just-in-time information access. In *IUI '00 : Proceedings of the 5th international conference on Intelligent user interfaces*, pages 44–51, New York, NY, USA, 2000. ACM Press.

[BMR+05] R.A. Burkhard, M. Meier, P. Rodgers, M.T.J. Smis, and J. Stott. Knowledge visualization : A comparative study between project tube maps and gantt charts. In K. Tochtermann and H. Maurer, editors, *Proceedings of the 5th International Conference on Knowledge Management*, pages 388–395. Know-Center, Austria, June 2005.

[BWK00] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. Guidelines for using multiple views in information visualization. In *Advanced Visual Interfaces*, pages 110–119, 2000.

[Che05] Chaomei Chen. Top 10 unsolved information visualization problems. *IEEE Comput. Graph. Appl.*, 25(4) :12–16, 2005.

[Chi00] Ed H. Chi. A taxonomy of visualization techniques using the data state reference model. In *INFOVIS*, pages 69–76, 2000.

[CLR89] Thomas Cormen, Charles Leiserson, and Ronald Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, MA, 1989.

[dR77] Joël de Rosnay. *Le macroscope*, pages 119–121. Le Point, 1977.

[EGK+03] J. Ellson, E.R. Gansner, E. Koutsofios, S.C. North, and G. Woodhull. Graphviz and dynagraph – static and dynamic graph drawing tools. In M. Junger and P. Mutzel, editors, *Graph Drawing Software*, pages 127–148. Springer-Verlag, 2003.

[Fek04] Jean-Daniel Fekete. The infovis toolkit. In *INFOVIS*, pages 167–174, 2004.

[FGW02] Usama Fayyad, Georges G. Grinstein, and Andreas Wierse, editors. *Information visualization in data mining and knowledge discovery*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[Fur86] G. W. Furnas. Generalized fisheye views. In *CHI '86 : Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 16–23, New York, NY, USA, 1986. ACM Press.

[HCL05] Jeffrey Heer, Stuart K. Card, and James A. Landay. prefuse : a toolkit for interactive information visualization. In *CHI*, pages 421–430, 2005.

[HMdN04] Seok-Hee Hong, Damian Merrick, and Hugo A. D. do Nascimento. The metro map layout problem. In *CRPIT '35 : Proceedings of the 2004 Australasian symposium on Information Visualisation*, pages 91–100, Darlinghurst, Australia, Australia, 2004. Australian Computer Society, Inc.

[HMM00] Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization : A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1) :24–43, /2000.

[JHK05] Sung Young Jung, Jeong-Hee Hong, and Taek-Soo Kim. A statistical model for user preference. *IEEE Transactions on Knowledge and Data Engineering*, 17(6) :834–843, 2005.

[Joh04] Chris Johnson. Top scientific visualization research problems. *IEEE Computer Graphics and Applications*, 24(4) :13–17, 2004.

[KHI+] Robert Kosara, Christopher G. Healey, Victoria Interrante, David H. Laidlaw, and Colin Ware. Thoughts on user studies : Why, how, and when.

[KY93] Hideki Koike and Hirotaka Yoshihara. Fractal approaches for visualizing huge hierarchies. In Ephraim P. Glinert and Kai A. Olsen, editors, *Proc. IEEE Symp. Visual Languages, VL*, pages 55–60. IEEE Computer Society, 24–27 1993.

[LR01] K. Lodaya and R. Ramanujam. An automaton model of user-controlled navigation on the Web. *Lecture Notes in Computer Science*, 2088 :208–??, 2001.

[LRP95] John Lamping, Ramana Rao, and Peter Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proc. ACM Conf. Human Factors in Computing Systems, CHI*, pages 401–408. ACM, 1995.

[LSM91] Xia Lin, Dagobert Soergel, and Gary Marchionini. A self-organizing semantic map for information retrieval. In *SIGIR '91 : Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 262–269, New York, NY, USA, 1991. ACM Press.

[LT92] R. Laurini and D. Thompson. *Fundamentals of Spatial Information Systems*. Academic Press, London, UK, 1992.

[Man75] B.B. Mandelbrot. *Les objets fractals : forme, hasard et dimension*. 1975.

[Nor02] Donald A. Norman. *The Design of Everyday Things*. Basic Books, September 2002.

[RHJ+04] Theresa-Marie Rhyne, William L. Hibbard, Chris Johnson, Chaomei Chen, and Steve Eick. Panel 1 : Can we determine the top unresolved problems of visualization ? In *IEEE Visualization*, pages 563–566. IEEE Computer Society, 2004.

[Shn96] Ben Shneiderman. The eyes have it : A task by data type taxonomy for information visualizations. In *VL '96 : Proceedings of the 1996 IEEE Symposium on Visual Languages*, page 336, Washington, DC, USA, 1996. IEEE Computer Society.

[Shn97] Ben Shneiderman. *Designing the User Interface : Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

[Spe00] Robert Spence. *Information Visualization*. Addison Wesley, 2000.

[Sta] Steffen Staab. Ontologies' kisses in standardization. *IEEE Intelligent Systems*, 17(2) :70–79.

[TB06] Lynda Tamine and Wahiba Bahsoun. Définition d'un profil multidimensionnel de l'utilisateur : vers une technique basée sur l'interaction entre dimensions. . In *COnférence en Rechercherche d'Information et Applications (CORIA), Lyon*, pages 225–236. ARIA, 15-17 mars 2006.

[Tho] Hans-Jorg Schulz Thomas. A framework for visual data mining of structures.

[Vui06] Romain Vuillemot. Modèles de navigation dans de grands corpus de documents : décomposition, classification et personnalisation. *IC2006 : 17e journées francophones d'Ingénierie des connaissances, Nantes, France. Session poster*, 28-30 juin 2006.

[Wex99] Alan Wexelblat. History-based tools for navigation. In *HICSS*, 1999.

[WOWR03] Walter Walter, Ontrup Ontrup, Daniel Wessling, and Helge Ritter. Interactive visualization and navigation in large data collections using the hyperbolic space. *icdm*, 00 :355, 2003.