



HAL
open science

Production automatique d'interfaces graphiques 3D pour le Web

Noémie Esnault

► **To cite this version:**

Noémie Esnault. Production automatique d'interfaces graphiques 3D pour le Web. Autre [cs.OH].
Université Rennes 1, 2011. Français. NNT: . tel-00574772

HAL Id: tel-00574772

<https://theses.hal.science/tel-00574772>

Submitted on 8 Mar 2011

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

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



THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

Ecole doctorale MATISSE

présentée par

Noémie Esnault

préparée à l'unité de recherche 6074 IRISA
Institut de Recherche en Informatique et Systèmes Aléatoires
IFSIC

**Production
automatique
d'interfaces
graphiques 3D
pour le Web**

**Thèse soutenue à Rennes
le 26 janvier 2011**

devant le jury composé de :

Kadi BOUATOUCH

Professeur, Université de Rennes 1 / *président*

Wojciech CELLARY

Professeur, Technical University of Poznan /
rapporteur

Pascal GUITTON

Professeur, Université de Bordeaux 1 /
rapporteur

Rémi COZOT

Maître de conférences, Université de Rennes 1
/ *examineur*

Jérôme ROYAN

Ingénieur de recherche, Orange Labs /
examineur

Christian BOUVILLE

Chercheur associé, INRIA / *directeur de thèse*

"To condense fact from the vapor of nuance."
Neal Stephenson, *Snow Crash*

Remerciements

Avant toute chose, je souhaite remercier Jérôme Royan pour ses conseils et ses encouragements depuis les premières idées et le montage de ce projet, jusqu'à la réalisation de ces travaux de thèse. Son énergie, sa bonne humeur, la profusion de ses idées, et la force avec laquelle il a su défendre mes travaux auprès d'Orange et des autres partenaires ont été un soutien indéniable tout au long de ces trois ans. Un grand merci également à Christian Bouville pour avoir accepté la direction de ces travaux, pour sa réactivité et pour ses connaissances et son intérêt sur des sujets très variés qui m'ont permis à la fois d'approfondir mes connaissances mais aussi de découvrir de nombreux domaines connexes à mon sujet et d'éveiller ma curiosité scientifique. Je remercie également Rémi Cozot pour le co-encadrement universitaire de ma thèse, pour ses discussions passionnées mêlant 3D et sémantique, et pour m'avoir permis de découvrir l'enseignement.

Je remercie également Kadi Bouatouch pour son soutien et pour avoir accepté d'évaluer mes travaux de thèse en tant que président. Merci enfin à Wojciech Cellary et Pascal Guitton pour leur participation à mon jury de thèse et pour avoir accepté la charge de rapporteurs.

Je remercie l'ensemble de l'équipe IAM-IACA à Orange pour les discussions, les pauses café, et la bonne humeur tout au long de la journée. Merci plus particulièrement à Danièle pour avoir soutenu mon projet en toutes circonstances ; à Nicolas, Sébastien, Gaspard et Olivier pour les jeudis du sport ; à Eric pour les discussions enflammées sur la société et l'avenir ; à Philippe pour les vidéos et les rires qui résonnent dans les couloirs. Merci aussi à Rozenn et Pierre qui prennent le relai.

Je remercie également l'ensemble de l'équipe Bunraku à l'IRISA. Si je n'ai pas eu la chance d'être si souvent dans vos locaux qu'à Orange, vous avez su rendre mes séjours sur le campus particulièrement agréables et productifs. Je retiendrai plus particulièrement les séances de hockey et les pauses pâtisseries, sans oublier le séminaire au vert qui m'a permis de m'intégrer à Bunraku en découvrant à la fois les nombreux travaux et thématiques de l'équipe lors des exposés, mais aussi ses membres à travers divers moments de détente et un concert inoubliable. Un grand merci également à Yves pour les échanges techniques qui ont été décisifs pour l'implémentation de mes travaux.

Je remercie Émilie, Khalid, Assya et Noam, Linette et Gérard, Paulette, pour leurs sourires, leur soutien et leurs encouragements.

Je remercie Michaël de tout mon cœur pour sa force dans les moments les plus difficiles, pour son amour, et pour le bonheur et la sérénité qu'il m'offre chaque jour. Merci à Uriel pour ses merveilleux éclats de rire. Merci à vous deux pour la place que je tiens dans votre cœur.

Table des matières

1	Introduction	5
2	Panorama des solutions industrielles et normatives	7
2.1	Du Web au Web 3D	7
2.1.1	Le Web 3D	7
2.1.2	Interfaces graphiques 3D et environnements virtuels	8
2.1.3	Les défis du Web 3D	9
2.1.4	Applications Web	11
2.2	Modélisation 3D, restitution et interactions	12
2.2.1	Modélisation 3D	12
2.2.2	Interfaces d'interaction, de restitution et combinées	13
2.2.2.1	Interfaces de restitution	13
2.2.2.2	Interfaces d'interaction	14
2.2.2.3	Interfaces combinées	14
2.2.3	Conclusion	14
2.3	Formats et outils 3D sur le Web	15
2.3.1	Solutions 3D embarquées	15
2.3.1.1	X3D	15
2.3.1.2	Collada	16
2.3.1.3	Flash	16
2.3.1.4	Unity	16
2.3.1.5	Conclusion	16
2.3.2	Solutions 3D intégrées	16
2.3.2.1	WebGL	17
2.3.2.2	XML3D	17
2.3.2.3	X3DOM	17
2.3.3	Conclusion	18
2.4	Gestion de données	19
2.4.1	Gestion de données sur le Web	19
2.4.1.1	Exécution de scripts côté serveur	19
2.4.1.2	Architectures trois niveaux	20
2.4.1.3	Services Web	20
2.4.1.4	Conclusion	20
2.4.2	Gestion de données et Web sémantique	20
2.4.2.1	Généralités	21
2.4.2.2	Web sémantique	21
2.4.2.3	Langages sémantiques	21
2.4.2.4	Limitations des métadonnées	22
2.4.3	Conclusion	23
2.5	Conclusion	23

3	Production automatique d'interfaces graphiques 3D	25
3.1	Techniques de visualisation des graphes	25
3.1.1	Généralités	25
3.1.2	Systèmes masse-ressort	27
3.1.3	Focales variables et projection Fish-eye	27
3.1.4	Regroupement de données	28
3.1.5	Conclusion	29
3.2	Navigation et structuration	30
3.2.1	Navigation	30
3.2.2	Structuration et urbanisme	31
3.2.3	Conclusion	32
3.3	Métaphores 3D de visualisation de l'information	32
3.3.1	Métaphores génériques pour la visualisation de données	33
3.3.1.1	Métaphores 3D pour la visualisation d'information	33
3.3.1.2	Visualisation de recherches sur le Web	34
3.3.1.3	Environnements 3D pour la visualisation de données multimédia	34
3.3.1.4	Conclusion	34
3.3.2	Langages pour la définition de métaphores de visualisation de l'information	35
3.3.2.1	Feuilles de styles XSLT pour la définition de métaphores 3D	35
3.3.2.2	X-VRML	36
3.3.2.3	Conclusion	38
3.3.3	Conclusion	38
3.4	Conclusion	38
4	Formalisme pour la création d'interfaces graphiques 3D	41
4.1	Vue d'ensemble	41
4.2	Module de structuration de l'information	46
4.2.1	Vue d'ensemble	46
4.2.1.1	Généralités	46
4.2.1.2	Éléments nécessaires à la structuration des données	47
4.2.2	Structuration hiérarchique	48
4.2.3	Langage formel de l'arbre d'exploration : le génotype	51
4.2.4	Langage formel de la structure exploratoire : le génome	52
4.2.5	De la requête à l'arbre d'exploration : le générateur de génotype	54
4.2.6	Conclusion	57
4.3	Module de présentation de l'information	57
4.3.1	Vue d'ensemble	58
4.3.1.1	Généralités	58
4.3.1.2	Sélection de la métaphore de présentation de l'information	62
4.3.1.3	Outil auteur pour la définition de métaphores de présentation de l'information	63
4.3.2	Langage formel de présentation de l'information : le phénomène	63
4.3.2.1	Niveaux d'exploration et hiérarchie de règles	64
4.3.2.2	Les règles	65
4.3.2.3	Opérateurs de modélisation	68
4.3.2.4	Fonctions de production et opérateurs divers	70
4.3.2.5	Insertion de l'information : accès au génotype	71
4.3.2.6	Fonctionnalités d'interaction et d'exploration	72
4.3.2.7	Préambule et compatibilités	74
4.3.2.8	Conclusion	75
4.3.3	De la feuille de style 3D à l'interface graphique 3D : le générateur de phénotype	75
4.3.4	Conclusion	77
4.4	Conclusion	78

5 Résultats et analyses	81
5.1 Visualisation de données multimédia	81
5.1.1 Interface 3D générique : la tour de carrousels	81
5.1.2 Interfaces 3D immersives	83
5.1.2.1 L'univers Orange	84
5.1.2.2 Un univers en papier	89
5.1.2.3 Conclusion	89
5.2 Analyse	89
5.2.1 Métaphores de structuration et de présentation de l'information	89
5.2.2 Modularité	91
5.2.3 Variété et adaptabilité des interfaces graphiques 3D	92
5.2.4 Extensibilité et compatibilité avec les technologies Web	94
5.2.5 Conclusion	95
6 Conclusion et perspectives	97
6.1 Contributions	97
6.2 Perspectives	98
6.2.1 Implémentations	98
6.2.2 Travaux de recherche	98
6.2.2.1 Tests utilisateurs	98
6.2.2.2 Amélioration du formalisme	98

Chapitre 1

Introduction

Ces dernières années, l'engouement pour le contenu 3D sur le Web s'est accru, s'élargissant des joueurs inconditionnels à de nombreuses catégories d'utilisateurs, avec l'apparition de mondes virtuels sociaux (type SecondLife), et de nombreuses applications intégrant ou simulant des contenus 3D (publicités 3D, jeu grand public, ...). Le développement des usages liés aux contenus 3D sur le Web est fortement liés au développement des technologies dans trois domaines :

- les formats de description de scènes 3D pour le Web ;
- le développement d'interfaces d'acquisition et de restitution facilitant les interactions avec le contenu 3D ;
- le développement d'outils auteurs permettant de créer du contenu 3D pour le Web.

Avec l'intégration de la 3D dans la nouvelle norme HTML5 [89] en cours de spécification, les navigateurs Web intègrent progressivement l'accélération matérielle 3D native, c'est-à-dire sans nécessiter l'installation de modules d'extension. Cette évolution entraîne une multiplication et une évolution des formats de description de scène 3D offrant aux concepteurs d'applications 3D pour le Web une large gamme de technologies dont les capacités de calcul, de rendu et d'interaction varient énormément.

Du point de vue matériel, de nouvelles interfaces d'acquisition et de restitution ayant pour objectif de faciliter les interactions avec du contenu 3D apparaissent sur le marché de la consommation de masse. Ainsi, les contrôleurs de jeux vidéos proposent d'interagir avec le contenu à partir de gestes plus intuitifs et plus proches des gestes effectués dans la vie réelle, tandis que les écrans 3D facilitent la compréhension des scènes virtuelles tridimensionnelles et suscitent une augmentation des attentes des utilisateurs finaux en terme d'interfaces graphiques 3D.

Enfin, de nouveaux outils auteur entièrement dédiés à la 3D apparaissent, et certaines technologies de conception d'applications Web 2D intègrent aujourd'hui des fonctionnalités 3D, souvent limitées à des transformation d'objets 2D dans l'espace 3D.

En parallèle de ces évolutions technologiques, les applications Web ont évolué d'applications statiques à des applications dynamiques séparant le contenu de l'application et sa présentation. Or nous pouvons constater que les outils auteur pour la création d'application Web 3D intègrent rarement les technologies Web permettant de créer du contenu dynamique. La prise en compte de ces technologies nécessite l'intégration de systèmes de gestion de données dans les outils de conception d'applications 3D, mais aussi un concept de feuilles de style 3D équivalent au concept des feuilles de style en cascade permettant de décrire les modèles de présentation actuellement utilisés sur le Web.

Dans ce document, nous nous intéressons plus particulièrement à la création automatique d'interfaces graphiques 3D pour le Web, c'est-à-dire à la présentation dans un espace tridimensionnel de résultats de requêtes sur de grands ensembles de données, permettant d'explorer l'information qu'ils contiennent. Un outil auteur de création d'interfaces graphiques 3D doit permettre au créateur d'une interface Web de définir une interface attractive et efficiente, par le biais, par exemple, de l'organisation du contenu à visualiser et de la description d'une feuille de style 3D. Les interfaces définies doivent pouvoir être adaptées au contexte (le domaine applicatif, les utilisateurs finaux, ...) et refléter les choix sémantiques et esthétiques du créateur de l'interface. Un tel outil doit donc permettre de produire des interfaces graphiques 3D variées, allant des interfaces immersives aux interfaces non-immersives, et de définir tout type d'organisation et de présentation du contenu dans l'espace 3D.

Nous considérons ici que la création d'une interface graphique 3D nécessite l'utilisation de modèles 3D, et donc l'existence d'outils auteur pour la création de modèles 3D et de bibliothèques de modèles 3D permettant à tout créateur d'une interface graphique, qu'il soit ou non infographiste 3D, de définir la géométrie de l'interface graphique.

La création d'une interface graphique 3D suppose son accessibilité à des utilisateurs finaux, que ce soit au niveau des interfaces de restitution, des dispositifs d'interaction, ou de l'utilisabilité de l'interface graphique (par exemple, son ergonomie ou encore l'efficacité de l'utilisation de l'interface dans le cadre de la recherche d'information). Dans ces travaux, nous nous intéressons donc brièvement aux interfaces d'acquisition et d'interaction permettant d'améliorer l'utilisation d'une interface graphique 3D. Nous étudions plus en détail un certain nombre de caractéristiques nécessaires aux outils de création d'interfaces graphiques 3D pour permettre la définition d'interfaces efficaces.

Dans le chapitre 2, nous étudions les domaines connexes à la création d'interfaces graphiques 3D pour le Web : l'évolution des technologies 3D et des usages des contenus 3D sur le Web ; les technologies de modélisation d'objets 3D et les interfaces d'acquisition et de restitution ; ainsi que les technologies de gestion de données. D'une part, cette étude nous permet d'assurer l'utilisabilité des interfaces graphiques 3D grâce aux technologies d'interaction et de restitution présentées, ainsi que l'existence d'outils de création d'objets 3D. D'autre part, elle permet de définir les contraintes technologiques que doit respecter un formalisme permettant la création d'interfaces graphiques 3D, que ce soit du point de vue de la compatibilité avec les technologies Web, entre autre pour la description de scènes 3D et la gestion de données, ou du point de vue des usages.

Dans le chapitre 3, nous cherchons à définir plus précisément les contraintes qu'un formalisme de création d'interfaces graphiques 3D doit respecter afin de permettre aux créateurs d'interfaces graphiques de décrire des interfaces à la fois attractives et efficaces. Pour ce faire, nous étudions les contraintes structurelles des interfaces graphiques permettant de faciliter l'accès à l'information contenue dans l'interface, ainsi que les règles esthétiques liées à la création d'interfaces graphiques. Enfin, une étude approfondie des technologies existantes pour la création d'interfaces graphiques 3D nous permet de définir les besoins en terme d'outil auteur pour la création d'interfaces graphiques 3D.

Dans le chapitre 4, nous proposons un formalisme pour la description et la production automatique d'interfaces graphiques 3D pour le Web basé sur la définition de métaphores de visualisation de l'information. Ce formalisme permet la description de métaphores complètes, attractives et efficaces pour la visualisation de données issues de grands ensembles de données. La flexibilité du formalisme présenté assure sa compatibilité avec les formats de gestion de données et de description de scènes 3D utilisés sur le Web et présentés dans le chapitre 2. Cette flexibilité facilite également l'intégration de notre formalisme à des outils auteur accessibles à l'ensemble des créateurs d'applications Web, qu'ils soient ou non spécialistes en programmation ou en infographie 3D. De plus, elle assure une grande variété des interfaces pouvant être décrites à l'aide de notre formalisme. Nous proposons également un certain nombre d'implémentations de ce formalisme adaptées à différents besoins et contraintes que nous explicitons.

Dans le chapitre 5, nous présentons un ensemble non restrictif de cas d'études pour la visualisation de données multimédia, incluant une interface graphique 3D non-immersive générique et une interface graphique 3D immersive sémantiquement corrélée aux données visualisées, ainsi qu'un certain nombre de cas de réutilisation et d'adaptation de ces métaphores. Puis, à partir de ces exemples, nous analysons les caractéristiques du formalisme présenté en regard des contraintes définies à partir de l'état de l'art.

Enfin, dans le chapitre 6, nous concluons ce document en revenant sur l'ensemble des contributions apportées par nos travaux et en proposant des perspectives en termes d'implémentation du formalisme, de validation d'usages, et de sujets de recherche connexes.

Chapitre 2

Panorama des solutions industrielles et normatives

Nous souhaitons proposer un formalisme permettant la création d'interfaces graphiques 3D sur le Web pour la visualisation et l'exploration de grands ensembles de données. Afin d'intégrer au mieux notre projet dans l'écosystème du Web, nous devons donc nous intéresser à la fois aux types de contenu 3D qui sont/seront présents sur le Web, à leurs usages ainsi qu'aux formats et aux standards qui existent ou seront bientôt disponibles. Nous devons aussi déterminer plus précisément le type de données qui seront ou non utilisables par notre formalisme, et à travers quelles technologies. Dans ce chapitre, nous présentons donc ces deux grands domaines connexes à notre problématique : le Web 3D et les technologies de gestion de contenu. En section 2.1, nous revenons sur l'évolution de l'intégration de contenus 3D sur le Web, et nous présentons le contexte actuel du Web 3D ainsi que les défis qu'il lui faut relever afin de se démocratiser. En section 2.2, nous présentons un point rapide sur l'évolution des interfaces pour la création, la visualisation et la manipulation de contenu 3D. En section 2.3, nous présentons les technologies permettant de proposer du contenu 3D interactif sur le Web. Puis nous présentons un rapide état de l'art des technologies de gestion de données en section 2.4. Enfin, en section 2.5, nous présentons nos conclusions sur les technologies 3D et de gestion de données sur le Web, dans le contexte de la visualisation de grands ensembles de données.

2.1 Du Web au Web 3D

2.1.1 Le Web 3D

En 1994, Mark Pesce et Toni Parisi étudient la possibilité de visualiser et de naviguer dans des environnements 3D sur le Web et présentent Labyrinth [68] à la première Conférence World Wide Web à Genève. Un forum est alors mis en place pour la conception des spécifications du VRML (Virtual Reality Modeling Language, [16, 21]), un langage multi plates-formes en liaison avec le World Wide Web, utilisable dans les clients de visualisation, les outils auteurs, et diverses applications. Ce forum évolue et devient le consortium VRML. Pour refléter les évolutions du VRML et l'émergence de nouveaux outils pour la 3D sur le Web (java3D, MPEG4, ...), le VRML Consortium devient le Consortium Web3D [92]. Les évolutions technologiques et l'émergence du langage XML (Extensible Markup Language, [15]), ont entraîné l'apparition d'un nouveau format baptisé X3D (eXtensible 3D, [22]), ayant comme contrainte d'être rétro-compatible avec VRML. Le but du consortium Web3D a toujours été de permettre l'affichage et la navigation à l'intérieur d'environnements virtuels intégrés au Web, proposant ainsi une nouvelle alternative aux sites Web. Cependant, un abus de langage entraîne souvent l'utilisation du terme "Web 3D" pour désigner l'utilisation de la 3D sur le Web, au travers de contenus 3D généralement interactifs intégrés à des pages Web en HTML (Hypertext Markup Language) et affichables par les navigateurs Web. La définition actuelle du Web 3D regroupe ces deux concepts (voir définition 2.1.1).

Définition 2.1.1 *Le Web 3D un système rassemblant un ensemble d'environnements 2D et 3D interactifs et inter-connectés, permettant tout type d'applications (simulation immersive pour l'apprentissage,*

réseau social, jeu en ligne, ...), et à l'intérieur desquels les utilisateurs se déplacent et interagissent avec le contenu de manière naturelle.

Driver et al. [26].

D'après Driver et al., le Web 3D inclura les outils et technologies du Web 1.0, du Web 2.0 et proposera de nouvelles technologies spécifiques aux environnements 3D, favorisant ainsi l'immersion de l'utilisateur et une augmentation significative de la collaboration entre utilisateurs.

D'autres terminologies sont parfois utilisées pour définir l'intégration de contenus 3D interactifs dans l'Internet. Ainsi, le terme *Métavers* (metaverse en anglais, pour méta univers) a été employé pour la première fois dans le roman *Le Samouraï virtuel* (titre original : *Snow Crash*, [83]) de Neal Stephenson paru en 1992. Dans cette fiction, Neal Stephenson décrit un monde virtuel dans lequel chacun se déplace grâce à une représentation de lui-même qu'il nomme déjà un avatar. Le centre de ce monde est une rue immense bordée de bâtiments en tous genres et de néons publicitaires, et possédant des rues secondaires. Toute personne peut construire un bâtiment le long de ces rues, moyennant l'obtention d'un permis de construire. Ce monde, peuplé d'avatars et de bâtiments divers (restaurants, magasins, résidences,...), est appelé "le Métavers". Les humains s'y connectent depuis leur ordinateur en utilisant des lunettes spécifiques permettant une forte sensation d'immersion dans le Métavers. Bien qu'intéressante, cette vision du métavers est restrictive de par son analogie poussée avec le monde réel (modélisation du monde par une boule géante, des rues dans une ville, nécessité d'utiliser des véhicules pour se déplacer plus rapidement d'un point à un autre, ...).

En 2006, Peter Flinn [36] définit le terme *VERSE* comme un environnement de simulation temps réel de terre virtuelle (Virtual Earth Realtime Simulation Environment). Tout comme Internet est un réseau de réseaux, un *Intervers* pourrait être un réseau de VERSEs. Il s'agirait donc d'un réseau mondial de mondes virtuels auxquels l'utilisateur se connecterait à travers une interface unique. Nous pourrions alors parler d'*Intravers* pour un réseau de mondes virtuels privés et d'*Extravers* pour un réseau de mondes virtuels privés accessibles à une ou plusieurs organisations.

Actuellement, le Métavers est plus généralement défini comme un environnement virtuel continu, tandis que le Web 3D fait référence à un ensemble de micro-mondes interconnectés, de type pages Web en 3D. Ces différentes terminologies représentent donc de manière générale le même concept et nous emploieront plus généralement le terme Web 3D dans la suite du document, car il suggère un parallèle intéressant avec le Web actuel. Dans la section suivante, nous étudions les défis à relever pour le déploiement d'un véritable Web 3D.

2.1.2 Interfaces graphiques 3D et environnements virtuels

Définition 2.1.2 *Une scène 3D est un espace tridimensionnel contenant un ensemble d'objets 3D pouvant avoir des propriétés physiques et des comportements.*

Définition 2.1.3 *Une interface graphique représente l'ensemble des composants graphiques, textuels et auditifs permettant à l'utilisateur d'interagir avec une application.*

Définition 2.1.4 *Une interface graphique 3D est une interface graphique dans laquelle les éléments peuvent être représentés et positionnés dans un espace tridimensionnel.*

Définition 2.1.5 *Un environnement virtuel est une scène ou un ensemble de scènes représentée(s) dans un espace à une, deux ou trois dimensions, à l'intérieur de laquelle l'utilisateur peut être représenté par un avatar. Un environnement est qualifié de persistant s'il existe et évolue en permanence.*

Définition 2.1.6 *Un avatar est un objet graphique fixe ou animé (comme par exemple un personnage de synthèse), généralement de type humanoïde, symbolisant l'utilisateur au sein de l'environnement virtuel. L'avatar est personnalisable et peut, entre autre, se déplacer, interagir avec l'environnement et interagir avec d'autres avatars.*

Les robots virtuels permettent de peupler artificiellement un environnement virtuel, ou peuvent avoir une fonction précise comme celle de guide ou d'assistant virtuel.

Dans le cadre de notre étude, nous nous intéressons plus particulièrement aux interfaces graphiques permettant la présentation dans un espace (ou un ensemble d'espaces) tridimensionnel(s)



Figure 2.1 – Interfaces graphiques 3D immersive (a) et non immersives (b,c).

du résultat d'une requête sur un ensemble de données. Une interface graphique sera alors dite *immersive* lorsqu'elle propose un ou plusieurs paradigme(s) de navigation(s) complet(s). Par exemple, l'utilisateur peut explorer l'interface graphique en se déplaçant sur un terrain respectant la gravité (en mode "marche") ou de manière libre (en mode "survol"). Lors de la navigation, les mouvements sont considérés comme effectués par la caméra représentant l'utilisateur dans la scène. Celui-ci est alors souvent représenté par un avatar. Dans une interface *non immersive*, l'utilisateur "examine" les différents objets du monde et n'est généralement pas représenté. Lors de la navigation, nous pouvons alors considérer que les mouvements sont effectués dans la scène face à une caméra fixe. Un environnement virtuel est ainsi considéré comme une interface graphique immersive (voir figure 2.1(a)) tandis que la présentation d'un catalogue d'objets est considéré comme une interface graphique non immersive (voir figures 2.1(b) et 2.1(c)). Dans ce rapport, nous étudions les interfaces graphiques 3D en réseau (Internet, réseau privé, etc.) immersives ou non.

Les environnements virtuels tridimensionnels se subdivisent en diverses catégories en fonction de leur utilisation principale. Les plus courants sont divisés entre jeux en ligne, comme les MMORPG (jeux de rôle en ligne massivement multi-joueurs), et les réseaux sociaux virtuels, plus récents mais dont le nombre d'utilisateurs augmente rapidement depuis quelques années. D'autres types d'environnements virtuels, présents généralement sur des réseaux plus restreints, sont également développés dans le domaine de l'éducation, de la simulation, ou de la collaboration. Par abus de langage, le terme de "monde virtuel persistant" est plus généralement employé dans le cas des réseaux sociaux et des MMORPG.

Ces dernières années, l'engouement pour les environnements virtuels est mesurable grâce à l'augmentation significative de la valeur des échanges monétaires réalisés en monnaie réelle ou virtuelle pour l'achat d'items virtuels : armes et avatars pour les MMORPG tels que World of Warcraft [11], vêtements, biens immobiliers et objets variés dans le cas de mondes virtuels sociaux. SecondLife [57] (voir figure 2.1(a)) a développé un système économique interne basé sur une monnaie appelée LindenDollar et échangeable contre des dollars réels. Cet engouement s'explique en partie par le caractère innovant de la technologie, mais aussi par l'expérience immersive qu'elle propose. L'utilisateur, représenté par un avatar, se déplace dans l'espace 3D, communique avec les autres utilisateurs, collabore avec eux, interagit avec les objets et accède à tout type d'information. Il se retrouve donc virtuellement immergé dans un monde avec lequel il retrouve la sensation d'agir physiquement, et qu'il peut modifier par ses actions.

Les interfaces graphiques 3D non immersives font aussi leur apparition sur le Web : résultats de recherche sous forme de cubes ([84], figure 2.1(b)) ou de sphères, sites Web et contenu multimédia ([93], figure 2.1(c)) ou publicités interactives, l'intégration complète ou partielle d'interfaces graphiques 3D sur les sites Web est de plus en plus courante.

2.1.3 Les défis du Web 3D

Le Web 3D possède une forte analogie avec le Web actuel. Clive [19] propose donc de se baser sur les enseignements du Web actuel pour définir les grandes lignes du Web 3D (qu'il nomme Méta-vers 2.0). La première conclusion que nous pouvons tirer des paragraphes précédents, est que le Web

3D ne sera pas un unique environnement monolithique de type SecondLife, mais un ensemble sans limite d'interfaces graphiques inter-connectées permettant toutes sortes d'activités : jeux, collaboration, pré-visualisation, environnement personnel, art, théâtre, commerce, etc. D'après Clive, certaines interfaces graphiques seront immersives, d'autres non. Cette diversité ne peut être obtenue que si le Web 3D, tout comme l'est le Web, n'est la propriété de personne, individu ou entreprise. De plus, elle nécessite l'existence de multiples outils permettant de créer du contenu 3D, ainsi que des interfaces graphiques 3D dans lesquelles le contenu peut être intégré. Voici une liste non exhaustive de contraintes nécessaires à l'avènement du Web 3D :

- La *scalabilité* : en ce qui concerne les contenus, la scalabilité ou l'échelonnabilité permet de fournir une représentation à différents niveaux de qualité permettant de s'adapter à la fois aux contraintes de transmission et aux capacités des terminaux utilisés pour l'affichage des données. La scalabilité de l'architecture correspond à la capacité d'une architecture à évoluer pour supporter une montée en charge (par exemple, les réseaux pairs à pairs ou encore la virtualisation) ;
- La *standardisation* : aucun standard unique n'est actuellement utilisé par les développeurs d'interfaces graphiques 3D, que ce soit au niveau des serveurs, des clients, des profils d'avatars, du type et de la gestion du contenu, des animations, etc. L'émergence plus nette de standards permettrait entre autre l'utilisation de moteurs de recherches sur le contenu de l'ensemble du Web 3D, ou encore l'agrégation dans une unique interface graphique 3D de contenus provenant de sources différentes ;
- L'*interopérabilité* : est la capacité d'un système à fonctionner avec d'autres systèmes présents ou futurs. Basée sur l'utilisation de standards, l'interopérabilité du Web permet de naviguer entre différentes interfaces graphiques de manière fluide, permettant par exemple d'augmenter la sécurité des transactions. Actuellement, l'interopérabilité est assurée par le format HTML. Déclinée au Web 3D, l'interopérabilité permettrait, par exemple, de conserver son avatar d'un environnement virtuel à un autre ;
- L'aide à la *création de contenu 3D* : une interface graphique 3D composée principalement de cubes texturés (comme Yoowalk [98] par exemple) offre une immersion et des interactions très limitées. De même, la métaphore utilisée pour organiser du contenu à travers une interface graphique a un impact fort sur la convivialité de l'interface, son adéquation à la navigation dans l'information et sur la façon dont l'utilisateur final va interagir avec les différents objets présents dans l'interface. L'engouement pour le Web est entre autre dû à l'apparition d'outils intuitifs et conviviaux facilitant la production de textes, la création et la gestion d'images ou de vidéos d'une part, et d'outils intuitifs et conviviaux d'aide à la création de sites Web classiques et Weblogs d'autre part. Le développement de ces outils a permis à tous les utilisateurs, spécialistes ou non de la programmation, de participer activement à la création de contenu sur le Web. Il est nécessaire de développer divers outils facilitant la création de contenu 3D, ainsi que la gestion et l'organisation de ces contenus dans l'espace 3D, autrement dit des outils de création d'interfaces graphiques 3D ;
- La *liberté* de créer tout type de contenu : une des raisons principales de l'adoption du Web par le plus grand nombre est la liberté qu'il donne à chacun de s'exprimer en ligne avec très peu de contraintes. Que ce soit au niveau de la charte graphique, des interactions, du contenu ou des usages, chaque utilisateur peut créer un site Web personnalisé. Cette liberté est assurée par le haut degré d'abstraction d'HTML et des standards qui se sont développés autour de ce langage (CSS, XML, . . .) ;
- Enfin, la *variété* et l'*uniformité* : le Web 3D a besoin de variété dans les outils et les métaphores utilisées, les contenus produits et les usages. Cependant, comme nous l'avons abordé dans les points traitant des standards et de l'interopérabilité, le Web 3D a également besoin d'uniformité. Si cette uniformité est en partie assurée par la présence de protocoles standardisés, elle s'est également fortement développée, entre autre sur les modes d'interaction, au fur et à mesure de l'évolution des usages sur le Web. Ainsi, par exemple, la mise en avant de liens hypertextes est aujourd'hui communément effectuée par un soulignement ou un changement de couleur. Cependant, seuls les usages imposent cette uniformisation de représentation des liens, les standards utilisés sur le Web n'interdisant pas l'utilisation d'autres moyens de mise en avant des liens (l'italique par exemple), et n'imposent pas non plus de présenter les liens de manière différente du texte. L'uniformité de la représentation des liens hypertextes est donc apparue avec les usages, permettant ainsi aux utilisateurs de naviguer plus rapidement sur le Web. Malgré cette uniformité

relative, la variété de représentation des liens reste importante, par exemple en choisissant les couleurs représentant les liens d'une page Web. Cet exemple démontre l'importance de la dualité variété-uniformité. La définition de nouveaux standards pour la création de contenu 3D pour le Web doit prendre en compte le couple variété-uniformité.

Le développement du Web 3D nécessite donc de relever de nombreux défis, sans perdre de vue les principes de base du Web, tant d'un point de vue technologique que conceptuel. Notre étude a pour but de proposer un formalisme pour la création d'interfaces graphiques 3D, et plus particulièrement la création d'interfaces graphiques 3D pour la visualisation et la navigation dans des ensembles de données quelconques. Ce formalisme doit être suffisamment flexible pour pouvoir répondre aux enjeux de variété du Web, permettant de construire des interfaces graphiques 3D variées du point de vue du contenu et de la mise en page. De plus, si nous n'observons pas de particularité forte dans la présentation de l'information sur le Web en fonction des domaines applicatifs, la présentation d'information dans un espace 3D peut nécessiter une plus grande adaptation de la métaphore de présentation de l'information en fonction de la sémantique des données à visualiser. Ce formalisme doit également assurer le principe d'uniformité, par exemple à travers la définition de standards pour l'exploration des données, ou encore en permettant l'utilisation d'une même interface graphique 3D pour la visualisation de données issues de domaines différents.

2.1.4 Applications Web

Nous cherchons à conserver une harmonie entre les concepts du Web et du Web 2.0 et notre formalisme afin de permettre une transition continue de la création d'interfaces graphiques Web dynamiques vers des interfaces graphiques Web dynamiques tridimensionnelles. Les pages Web dynamiques sont créées à la volée. Le contenu est extrait dynamiquement d'un ensemble de données afin de construire une page Web spécifique qui sera rendue après application d'une feuille de style. Une application Web dynamique est généralement composée de trois modules (voir figure 2.2) :

- un module de *gestion de données* (comme par exemple un Système de Gestion de Bases de Données, ou SGBD) dont la structure doit refléter au mieux les besoins spécifiques de l'application développée et définissant quelles données seront affichées ;
- un module de *structuration de l'application* et des données définissant la structure de navigation de l'application, c'est-à-dire les chemins que l'utilisateur peut emprunter pour explorer le contenu de l'application, et spécifiant ainsi quelles données sont contenues dans chaque page Web ainsi que les liens internes entre les différentes pages ;
- un module de *présentation de l'information* définissant les caractéristiques externes de l'application, c'est-à-dire la mise-en-page de l'application frontale, les modèles de présentation et les feuilles de style.

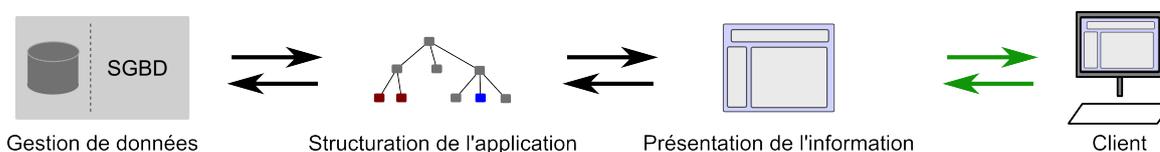


Figure 2.2 – Pipeline d'une application Web dynamique.

L'adaptation de ce formalisme au Web 3D demande un certain nombre d'ajustements pour chaque module. Au niveau du module de gestion de données, aucun changement n'est nécessaire. La présence ou l'ajout de contenu 3D dans la base de données peut être directement intégré à l'interface graphique 3D. L'adaptation du module de structuration de l'application implique la prise en compte de nouvelles contraintes et de nouvelles possibilités offertes par la 3D. L'adaptation du module de présentation de l'information impose de prendre en compte l'augmentation du nombre de variables et d'attributs induite par la géométrie 3D. Ainsi, il est plus complexe d'organiser un ensemble d'objets dans l'espace 3D, prenant en compte les occlusions et l'accès aux données distantes, que d'organiser un ensemble de paragraphes et de sections sous forme de texte, ou un ensemble d'images sous forme de tableau. Cependant, l'ajout d'une troisième dimension permet d'augmenter considérablement l'espace de représentation visuelle et la richesse des métaphores.

2.2 Modélisation 3D, restitution et interactions

La présentation de contenu bidimensionnel, comme du texte ou des images, à travers une interface graphique 3D présente un intérêt limité, et est généralement moins exploitable que la présentation du même contenu à travers une interface graphique 2D. Il est en effet plus complexe de lire du texte à partir d'une interface graphique 3D, du fait de la perte de résolution et des déformations, qu'à partir d'une interface graphique 2D. Cependant, les interfaces graphiques 3D permettent d'augmenter l'espace de représentation visuel et d'offrir une grande richesse des métaphores de présentation de l'information. Pour tirer parti au mieux des avantages d'une interface graphique 3D, il est donc nécessaire de démocratiser la modélisation et l'accès au contenu 3D pour les auteurs d'interfaces graphiques 3D. Au niveau de l'utilisateur final, si la représentation 3D peut dans certains cas être plus naturelle que la représentation 2D, les interfaces de restitution et d'interactions sont aujourd'hui majoritairement 2D (écrans) et limitées à deux degrés de liberté (souris, . . .), ce qui diminue le caractère intuitif initial de la représentation 3D. L'évolution des interfaces d'interaction et de restitution vers des espaces 3D permet de mieux profiter des avantages des interfaces graphiques 3D.

En section 2.2.1, nous présentons un ensemble de systèmes permettant la modélisation et/ou l'acquisition de modèles 3D. Puis, en section 2.2.2, nous nous intéressons à l'utilisateur final en présentant divers technologies permettant d'améliorer la visualisation de contenu 3D à travers les technologies stéréoscopiques d'une part, et d'améliorer l'interaction avec le contenu 3D d'autre part.

2.2.1 Modélisation 3D

La synthèse d'image 3D (le terme 3D réfère aux trois axes x , y et z formant le repère orthonormé de la géométrie dans l'espace) est un ensemble de techniques permettant la restitution de données géométriques tridimensionnelles, ou modèles 3D, sur un écran. Dans le domaine des arts, un modèle est un objet ou une personne servant de référence à l'artiste pour la création de son œuvre. Dans le domaine de la technique, un modèle est au contraire le représentant d'un objet physique. C'est le cas des modèles 3D. Le terme *modélisation* définit alors la création d'une instance d'un objet (et donc d'un modèle, comme la création d'un vase) selon un certain processus ou modèle formel. Dans cette section, nous présentons un ensemble non exhaustif de techniques permettant d'obtenir les modèles 3D peuplant les interfaces graphiques créées à l'aide de notre formalisme.

Scanner 3D Dans le cas de la reproduction d'objets réels, il est possible de procéder à l'acquisition de modèles d'objets à l'aide d'un scanner 3D. Celui-ci collecte l'ensemble des données d'un objet réel correspondant à sa forme et à son apparence (couleurs par exemple). Généralement, l'acquisition d'objets réels nécessite l'utilisation de matériel spécialisé de grande précision et une chaîne de traitement complexe. Certains travaux proposent cependant des solutions peu coûteuses et relativement simples à utiliser (Rusinkiewicz *et al.* [74]).

Modélisation directe par maillages 3D et géométrie de construction solide La majorité des applications 3D utilise des modèles représentés sous forme de maillage polygonal. Tout objet peut ainsi être défini manuellement à l'aide d'un ensemble de sommets, d'arêtes reliant ces sommets et définissant ses faces. Si cette technique permet d'obtenir de très bons résultats et de personnaliser au maximum le modèle créé (forme, texture, . . .), la prise en main des logiciels est souvent difficile et demande un apprentissage long.

La géométrie de construction solide, ou CSG (constructive solid geometry) permet la création de surfaces complexes à l'aide d'opération booléennes sur des objets solides appelés primitives. Plus simple à prendre en main que la modélisation de maillages 3D, la CSG restreint cependant la liberté de modélisation : certains degrés de réalismes et certaines formes (nuages, montagnes, liquides, . . .) ne peuvent être représentés à l'aide de la CSG. La facilité de prise en main de cette technique de modélisation en fait généralement le choix privilégié pour la création de modeleurs intégrés aux environnements virtuels tels que SecondLife.

Modélisation à partir d'esquisses La modélisation de maillage polygonaux peut être rendue plus intuitive grâce à l'utilisation d'esquisses 2D (sketchs en anglais).

SketchUp [44] est un logiciel de modélisation 3D orienté architecture. Son interface est facile à prendre en main, principalement grâce à un système de curseur intelligent permettant à l'utilisateur de manipuler des objets 3D à l'aide de la souris. La manipulation se base sur l'esquisse de plans, l'extrusion et le déplacement de faces. Les quelques primitives proposées limitent cependant les possibilités de modélisation.

Igarashi *et al.* [50] proposent une interface de modélisation nommée Teddy et permettant de "gonfler" des esquisses. L'utilisateur dessine un ensemble de formes qui sont automatiquement interprétées pour former un polygone 3D. Teddy n'est cependant pas facile à prendre en main et il n'est pas possible de maîtriser l'aspect de la surface du modèle.

Cherlin *et al.* [18] présentent un système permettant la modélisation d'objets 3D à partir de quelques lignes tracées sur un plan 2D. L'objet est d'abord esquissé grossièrement, puis raffiné à l'aide d'opérations comme l'ajout de spirales pour extruder un volume le long d'une forme 2D, le griffonnage (scribble) pour l'ajout de détails de type pliage ou bosse, et la courbure pour distordre le modèle.

Modélisation paramétrique Les modélisateurs paramétriques permettent de modéliser une catégorie d'objets grâce à la description des invariants et paramètres de l'objet. FaceGen [78] permet ainsi la modélisation de visages humains à partir de nombreux paramètres. Certains environnements, comme Yoowalk [98], permettent la paramétrisation d'un avatar en proposant un choix minimal pour faciliter l'utilisation, ce qui restreint cependant la personnalisation.

Catalogues d'objets certaines applications proposent l'utilisation de catalogues d'objets pour la création d'interfaces graphiques 3D. Un tel catalogue peut proposer un ensemble d'avatars, d'objets, d'espaces paramétriques ou non. SquareClock [82] propose ainsi un logiciel utilisant divers catalogues d'objets et de meubles paramétriques proposés par des partenaires pour la conception et l'agencement d'intérieur. Un certain nombre de bibliothèques de modèles 3D gratuites ou payantes sont également disponible sur le Web, comme par exemple 3D warehouse [43].

Dans le cadre de notre étude, nous ne cherchons pas à proposer ou à améliorer une méthode de modélisation d'objets 3D, mais un formalisme permettant la présentation de l'information sous la forme d'interfaces graphiques 3D. Les différentes méthodes de création ou d'acquisition de modèles 3D présentées dans cette section offrent de nombreux choix à l'auteur d'une interface graphique 3D pour peupler ses scènes virtuelles, ainsi qu'aux gestionnaires de données pour construire des représentations 3D de l'information contenue dans des systèmes de gestion de données (comme par exemple les modèles 3D des produits disponibles à la vente dans le cadre d'une boutique en ligne).

2.2.2 Interfaces d'interaction, de restitution et combinées

Les interfaces graphiques 3D permettent de proposer aux utilisateurs finaux de nouveaux paradigmes pour la visualisation et la navigation dans de grands ensembles de données. Cependant, l'immersion dans l'interface graphique est limitée par l'utilisation d'écrans 2D et de dispositifs d'interaction 2D, et l'interaction avec les éléments de l'interface est souvent difficile. Dans cette section, nous étudions différentes techniques permettant d'améliorer la visualisation et l'interaction avec du contenu 3D.

2.2.2.1 Interfaces de restitution

La restitution de modèles 3D se fait généralement sur des écrans 2D après une phase de rendu permettant de projeter la scène 3D dans l'espace 2D en prenant en compte un certain nombre de paramètres tels que l'éclairage. Cependant, les écrans 3D se démocratisent, ajoutant un attrait supplémentaire à la production de contenu 3D. Parmi les types de dispositifs de restitution 3D existants, les plus courants sont les écrans stéréoscopiques avec lunettes et autostéréoscopiques. Les écrans stéréoscopiques fournissent une image différente pour l'œil gauche et pour l'œil droit. L'utilisateur doit alors généralement porter des lunettes spéciales permettant à chaque œil de ne recevoir que l'information qui lui est destinée. Trois technologies stéréoscopiques existent : les anaglyphes (stéréoscopie passive à l'aide d'une lentille rouge et d'une lentille cyan), la polarisation (stéréoscopie passive à l'aide

de lentilles polarisées) et l'alternance d'images (stéréoscopie active avec occultations alternées des lentilles). Les écrans autostéréoscopiques permettent l'affichage d'images stéréoscopiques sans utilisation de lunettes. Les différentes vues créant l'illusion de profondeur sont intégrées dans un cône de vue, celui-ci étant répliqué horizontalement plusieurs fois sur l'écran. Le changement de cône de vue entraîne généralement une perte de l'illusion de profondeur, l'utilisateur doit donc être placé face à un cône de vue et éviter de se déplacer.

Les écrans 3D posent cependant un certain nombre de problèmes. La dynamique (écart de luminosité entre le point le plus sombre et le point le plus lumineux) perçue est souvent atténuée par le port des lunettes. De plus, l'effort oculaire supplémentaire induit par la stéréoscopie peut entraîner une fatigue oculaire gênante, voir l'apparition de nausées. Le développement de la vision chez les enfants de moins de sept ans pourrait aussi être perturbé par la visualisation stéréoscopique.

2.2.2.2 Interfaces d'interaction

L'interaction avec du contenu 3D via des interfaces à deux degrés de liberté telles que la souris est souvent difficile, et implique un apprentissage long. De nombreuses techniques permettent d'améliorer la navigation et/ou l'interaction avec du contenu 3D à partir d'interfaces d'interaction 2D. C'est le cas par exemple des techniques basées sur les points d'intérêts (Mackinlay *et al.* [59], Hachet *et al.* [45]) ou les outils de positionnements interactifs (Bier [10]).

Les interfaces d'interaction 3D offrent une interaction avec le contenu 3D plus intuitive et plus précise en assurant une conversion naturelle des mouvements humains aux contrôles d'interaction (Bowman *et al.* [13]). Si l'utilisation d'interfaces d'interaction 3D a longtemps été principalement réservée aux arcades de jeux vidéos, le succès de la Wii Remote (Wiimote) de Nintendo [66] a entraîné un regain d'intérêt pour les interfaces d'interaction 3D. Les techniques les plus employées sont l'utilisation d'accéléromètres et de gyromètres avec récepteurs à fréquence radio ou infrarouge, ainsi que le traitement vidéo temps-réel. L'association des interfaces d'interaction 3D avec un retour haptique ou pseudo-haptique améliore les interactions de l'utilisateur final avec les interfaces graphiques.

2.2.2.3 Interfaces combinées

Ces dernières années, la combinaison des interfaces de restitution et d'interaction s'est démocratisée avec la généralisation des écrans tactiles. Téléphones, écrans de station multimédia, tables interactives, les écrans tactiles intègrent de plus en plus notre quotidien. L'intérêt principal des écrans tactiles multipoint est la manipulation directe du contenu affiché. Les objets peuvent être tournés, déplacés, redimensionnés de manière à ce qu'un unique point de l'objet reste positionné sous le doigt de l'utilisateur. La prédictibilité et le réalisme des mouvements de l'objet entraînent la sensation de manipuler des objets réels, le principal défaut des écrans tactiles multipoint étant l'occultation très gênante sur les écrans de petite taille. Le passage de la manipulation 2D vers la manipulation et la navigation 3D entraîne une perte de la correspondance directe entre l'action de l'utilisateur final et l'effet produit (déplacement d'un objet par exemple). Afin de diminuer cet inconvénient, la majorité des systèmes permettant de manipuler du contenu 3D à partir d'une interface tactile multipoint propose une interaction avec les objets depuis une vue de dessus (Hancock *et al.* [47], Wilson *et al.* [94], project [70]).

2.2.3 Conclusion

Actuellement, les interactions en trois dimensions (création et manipulation d'objets, déplacement, ...) sont peu intuitives de par l'utilisation d'interfaces d'interaction ne possédant pas suffisamment de degrés de liberté (une souris 2D par exemple). Le développement rapide des interfaces de nouvelle génération, comme la Wiimote [66], laisse présager d'un futur proche où chaque utilisateur sera équipé de façon à interagir de manière beaucoup plus intuitive avec du contenu 3D. Nous considérons donc dans ces travaux que l'interaction en 3D n'est pas un frein à la création ou à la navigation dans un environnement virtuel tridimensionnel. Au niveau des interfaces de restitution, la démocratisation des écrans 3D entraîne une demande croissante d'interfaces graphiques tridimensionnelles.

2.3 Formats et outils 3D sur le Web

Les premières tentatives d'affichage de scènes 3D sur le Web ont commencé en 1994 (voir section 2.1.1). Depuis, de nombreux formats ont été développés, nécessitant souvent l'installation d'un logiciel propriétaire. Malgré les nombreuses tentatives de spécification d'un standard 3D ouvert, générique et unique, la communauté de l'informatique graphique (industriel comme académique) a plutôt tendance à multiplier les formats 3D. En effet, les environnements virtuels se sont principalement développés ces trente dernières années autour de l'industrie du jeu vidéo. Or ses acteurs majeurs, les producteurs de jeux vidéo, ont un fort intérêt à la conservation de la propriété intellectuelle sur les contenus et les technologies. Au contraire, notre technologie, tout comme le Web, est plus adaptée à un standard ouvert permettant à chacun de produire du contenu pour le Web 3D.

Dans les sections suivantes, nous présentons diverses techniques permettant d'afficher du contenu interactif 3D via un navigateur Web. Nous présentons les possibilités et limitations de l'usage de modules d'extension pour la visualisation de contenus 3D, puis les avantages et contraintes liés à l'intégration native de la 3D dans les navigateurs Web, concernant la création d'interfaces graphiques 3D. Tout au long de cette étude, nous nous intéressons plus particulièrement aux capacités des langages présentés au regard des points suivants :

- la *variété* des interfaces graphiques pouvant être représentées. En effet, le standard choisi doit permettre la visualisation et l'interaction avec de nombreux objets, c'est-à-dire le rendu de scènes complexes immersives et interactives ;
- l'*accessibilité*, c'est-à-dire le besoin ou non pour l'utilisateur final d'agir pour pouvoir lire le format en question (par exemple, installation d'un module d'extension) ;
- le *caractère paramétrable* des modèles 3D. En effet, les éléments représentés peuvent appartenir à une même catégorie (ainsi, *Snow Crash*, *Candide*, *Bel-Ami* et *Les Raisins de la Colère* sont tous des romans). Afin de faire ressortir cette appartenance, il peut être intéressant de représenter ces éléments par des objets similaires mais personnalisés (par exemple, un livre). La paramétrisation des modèles 3D permet alors la création d'un objet générique (le livre) possédant des attributs dont un certain nombre (le titre du roman, la photo de l'auteur sur la quatrième de couverture) est initialisé pour chaque élément.

2.3.1 Solutions 3D embarquées

Les modules d'extensions permettent d'ajouter de nouvelles fonctionnalités à un logiciel hôte (ici, le navigateur Web). La 3D n'étant pas originellement intégré dans la norme HTML, la majorité des applications 3D disponibles via un navigateur Web nécessitent l'installation d'un module d'extension spécifique au format 3D choisi. Dans les paragraphes suivants, nous présentons les principaux formats et outils utilisés sur le Web pour présenter du contenu 3D et nécessitant l'installation d'un module d'extension.

2.3.1.1 X3D

Le consortium Web 3D propose un standard ISO libre de droit et extensible pour la création de contenu 3D sur le Web : X3D [22] (voir section 2.1.1). Basé sur le métalangage XML, le format X3D permet la description d'un graphe de scène 3D, et assure une compatibilité descendante avec son prédécesseur VRML97. X3D définit également un environnement d'exécution complet incluant l'exécution de scripts et un système de gestion d'évènements.

Le format X3D permet la description de sous graphes de scènes paramétriques appelés *prototypes*. Ces prototypes peuvent être appelés depuis le graphe de scène les décrivant, ou depuis un graphe de scène externe. Les paramètres du prototype possèdent une valeur par défaut et sont instanciés lors de l'appel au prototype. L'utilisation de prototypes permet par exemple de représenter un ensemble d'éléments de même nature (comme des ouvrages) par un même objet 3D (comme un livre) tout en personnalisant cet objet pour chaque élément (la texture de la couverture du livre est différente pour chaque élément).

Différents modules d'extension existent pour permettre l'affichage de scènes X3D dans un navigateur Web, comme par exemple Octaga, InstantReality ou BSContact. Il est alors possible d'accéder à la scène via l'interface SAI (Scene Access Interface) définie par le standard. Celle-ci est disponible en

EcmaScript/JavaScript et en Java.

Bien que complet, le format X3D peine à s'imposer comme standard pour la description de scènes 3D sur le Web. Parmi les freins à son adoption, nous pouvons constater que ce format est très rarement ou mal supporté par les outils commerciaux de modélisation 3D.

2.3.1.2 Collada

Collada (COLLABorative Design Activity, ou activité de conception collaborative) est un standard définissant un format de fichier d'échange pour les applications 3D. Conçu comme un format intermédiaire pour faciliter le pipeline de création de jeux vidéos, Collada n'inclut ni environnement d'exécution ni module de gestion d'évènements. Il ne s'agit donc pas d'un concurrent direct d'X3D, comme le démontre le cas d'étude présenté par Arnaud et Parisi [2] pour le développement d'applications Web utilisant à la fois Collada comme format d'échange et X3D comme format final.

2.3.1.3 Flash

Adobe Flash [85] est une plateforme multimédia développée par Adobe Systems. Largement répandu sur le Web, Adobe Flash permet principalement de créer des animations et du contenu interactif pour le Web. Depuis la version 10, Flash permet d'effectuer des transformations 3D et d'insérer des objets 3D simples dans les scènes créées. Cette évolution permet aux développeurs peu expérimentés en 3D de créer du contenu 3D en concevant des objets 2D puis en les modifiant dans l'espace 3D. Cependant, la technologie proposée reste limitée et n'exploite pas les accélérations du matériel graphique, limitant la visualisation de scènes 3D complexes interactives en temps réel à l'aide de Flash.

2.3.1.4 Unity

Unity [86] est un outil de création de jeux vidéo 3D. Il permet entre autre de créer des applications Web utilisant le module d'extension Unity Web Player. Unity est un environnement de développement permettant de charger des modèles multimédia (vidéo, 3D, ...) et de les organiser dans un espace 3D. De nombreuses fonctionnalités de rendu et de script sont disponibles permettant de créer des jeux vidéos complexes en séparant nettement la création des modèles multimédia (créés à l'aide de logiciels tiers et importés dans Unity) et la programmation. Orienté création de jeux vidéos, Unity peut cependant être utilisé pour créer des interfaces graphiques 3D.

2.3.1.5 Conclusion

Les modules d'extension pour l'affichage de scènes 3D sur le Web permettent d'obtenir de très bonnes capacités de rendu et d'interaction avec du contenu 3D grâce à la pré-compilation du code. Cependant, de nombreux formats 3D nécessitant l'installation d'un module d'extension coexistent sur le Web. L'utilisateur final se retrouve donc face au problème de l'installation de plusieurs modules d'extensions, de leur compatibilité avec son navigateur ou son système d'exploitation, et d'éventuelles failles de sécurités induites par l'installation des modules en question. Ces inconvénients sont très certainement la raison principale pour laquelle la majorité des applications (3D ou non) nécessitant l'installation d'un module d'extension, mis à part Adobe Flash [85], ne connaissent pas le succès escompté. Or la version actuelle d'Adobe Flash ne permet pas le rendu de scènes complexes et interactives en temps réel. Dans la section suivante, nous nous intéressons à l'intégration native, c'est à dire ne nécessitant pas de module d'extension, de la 3D dans les navigateurs Web.

2.3.2 Solutions 3D intégrées

La version 5 d'HTML (HTML5 [89]), en cours de spécification, propose d'intégrer nativement l'accélération matérielle 3D dans les navigateurs en définissant la balise `canvas`. Cependant, aucun format n'est préconisé pour la description de scènes 3D. Plusieurs propositions sont actuellement en cours de discussions au sein du W3C afin de tirer parti de cette nouvelle fonctionnalité d'HTML.

2.3.2.1 WebGL

En 2009, le groupe Khronos propose une API (Application Programming Interface - interface de programmation) en JavaScript bas niveau pour le développement d'applications 3D sur le Web : WebGL (Web Graphics Library [53]). Grâce à cette API, il est possible d'effectuer des commandes OpenGL-ES (version d'OpenGL pour les plateformes mobiles ou embarquées) en JavaScript. Elle permet d'utiliser la balise `canvas` introduite dans la nouvelle norme HTML5 pour faire du rendu 3D directement dans les navigateurs Web sans utiliser de module d'extension. WebGL est disponible en natif dans la version beta de Firefox 4, et en modifiant une variable dans les versions actuelles de Chrome et WebKit. Dans les prochains mois, WebGL sera inclus en natif dans les versions courantes de Firefox, Chrome, WebKit et Opéra. Aucune annonce n'a été faite quant à une possible intégration de WebGL dans Internet Explorer.

Si WebGL permet une accélération matérielle 3D native dans les navigateurs Web, cette API reste de bas niveau. Un certain nombre de bibliothèques, comme SpiderGL [24], sont basées sur WebGL et proposent des algorithmes plus haut niveau pour la création de graphiques 3D sur le Web. Cependant, elles ne proposent pas de structure du type graphe de scène, l'accès aux données 3D étant uniquement géré en mode immédiat. De plus, les capacités actuelles de JavaScript ne permettent pas d'effectuer des calculs complexes en temps réel. Si la qualité du rendu obtenu est élevée, les mises à jour de scènes complexes sont très lentes. WebGL reste donc un bon candidat pour la présentation de contenu 3D interactif simple mais ne permet pas la création d'interfaces graphiques 3D complexes.

2.3.2.2 XML3D

Sons *et al.* [81] proposent une approche différente pour intégrer du contenu 3D directement dans les navigateurs Web : XML3D. Plutôt que d'adapter les technologies 3D existantes pour les intégrer au Web, ils proposent une évolution minimale des technologies Web pour obtenir un support complet de la 3D. Cette approche est plus à-même de faciliter l'évolution des compétences des concepteurs Web vers des compétences Web 3D que de faciliter l'évolution des compétences des concepteurs 3D vers des compétences Web 3D.

XML3D est un format de description de scènes 3D sous forme d'extension des documents Web actuels. Les scènes 3D peuvent être décrites dans un fichier autonome ou être directement intégrées dans l'arbre DOM (Document Object Model, [88]) d'un document HTML, DOM étant un standard pour l'accessibilité et la manipulation de documents HTML. Contrairement aux usages courants dans le monde de la 3D, le style et le contenu sont en parti séparés. En effet, il est possible d'appliquer des transformations 3D et des shaders à des maillages ou groupes de géométrie de la scène 3D intégrée au DOM via une feuille de style CSS. Les interactions et la gestion d'évènements sont, tout comme pour le reste du DOM, gérées par l'API DOM à l'aide d'évènements DOM et de scripts DOM. Trois implémentations sont proposées par Sons *et al.* : une implémentation native dans Firefox et dans Google Chrome, et une implémentation plus limitée basée sur WebGL.

Si XML3D est plus en harmonie avec les concepts et technologies du Web que WebGL, il reste un format bas niveau de description de scènes 3D et nécessite le développement d'outils auteur pour la création de contenu 3D interactifs.

2.3.2.3 X3DOM

Les premières versions des spécifications de HTML5 [52] proposaient l'utilisation d'X3D, ou de technologies basées sur X3D, pour l'insertion de contenu 3D dans l'arbre DOM d'HTML. Cependant, aucune indication ne précise la façon dont l'intégration d'X3D doit se faire. Behr *et al.* [6, 5] proposent une solution pour connecter du contenu X3D à l'arbre DOM HTML : X3DOM. Le but d'X3DOM est de proposer une plateforme expérimentale d'intégration d'X3D au DOM HTML avant une intégration directe dans HTML5 permettant d'être supporté par l'ensemble des navigateurs. Le pipeline et l'évolution actuelle de l'implémentation d'X3DOM sont présentées en figure 2.3. X3DOM permet de tester l'intégration de scènes X3D dans un document HTML en proposant le meilleur rendu et la meilleure interaction possible en fonction des capacités du navigateur utilisé : si le navigateur intègre nativement le format X3D, celui-ci gère directement la scène 3D ; dans le cas contraire, l'algorithme recherche la

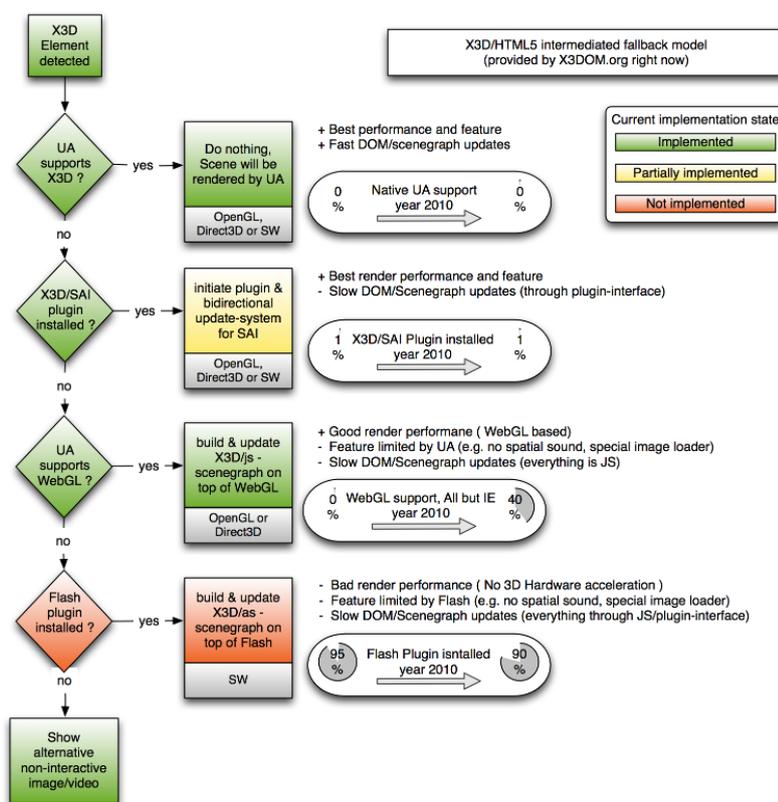


Figure 2.3 – Pipeline de rendu X3DOM [38]

présence d'un module d'extension permettant de conserver de bonnes capacités de rendu et d'interaction ; si aucun module n'est trouvé et que le navigateur est compatible WebGL, la scène sera rendue via OpenGL-ES avec les limitations discutées ci-dessus ; enfin, si aucune gestion de la 3D n'est possible, une simple image 2D ou une vidéo est proposée en remplacement de la scène 3D.

Un profil X3D spécifique au Web (le profil DOM) est proposé par Behr *et al.* [6]. Il conserve la majorité des fonctionnalités du profil Interactif X3D, sans le support des prototypes et des interactions. En effet, ces dernières sont gérées de manière plus proche des interactions disponibles via l'API DOM (utilisations des événements "onclick", "onmouseover", etc.).

L'avancement rapide des fonctionnalités de la plateforme expérimentale X3DOM permet de présager des possibilités offertes par l'intégration directe d'X3D dans l'arbre DOM. Cependant, la version actuelle, principalement basée sur WebGL, ne permet de visualiser que des scènes simples et limite la définition d'interactions nativement disponibles dans X3D telles que l'utilisation de capteurs cylindriques ou planaires.

2.3.3 Conclusion

Si le développement de technologies et d'usages 3D sur le Web connaissent un regain d'intérêt depuis l'annonce d'HTML5, nous pouvons observer un manque d'uniformité concernant les formats 3D pour le Web et les différentes approches permettant d'intégrer du contenu 3D aux pages HTML. De plus, les concepteurs d'applications se trouvent aujourd'hui face à un dilemme complexe : proposer une application contenant une scène 3D complexe et de nombreuses interactions mais nécessitant l'installation d'un module d'extension, ou proposer une application directement affichable par le navigateur Web mais ayant des capacités d'interaction faible. La voie de la cohabitation proposée par X3DOM (adaptation des possibilités offertes par l'interface en fonction des capacités du navigateur Web) est un compromis intéressant, en attendant l'éventuelle apparition d'un format 3D standard sur le Web et directement intégré aux navigateurs.

La diversité des formats disponibles est un élément important à prendre en compte lors de la conception de nouveaux outils d'auteurs pour le Web 3D. L'abstraction du langage de description de scènes permet de proposer un pipeline de création d'interfaces 3D pouvant produire des scènes 3D de différents formats, nécessitant ou non l'installation de modules d'extension, et assurant la pérennité du processus de création quelle que soit l'issue des discussions actuelles concernant l'intégration de la 3D dans les documents HTML.

Dans la section suivante, nous présentons un rapide état de l'art des technologies de gestion d'ensembles de données.

2.4 Gestion de données

La gestion des connaissances est divisée en deux composants principaux. La gestion de l'information d'une part, et l'accès à l'information d'autre part. Le premier permet d'assurer que l'information est à jour, complète et conforme. Le second assure que l'information est disponible pour les personnes concernées, au bon niveau de détail, et d'une façon uniforme, rapide et cohérente.

Le monde actuel, et plus particulièrement le Web, est basé sur l'information. Ainsi, si le manque d'information est devenu rare, sa surabondance pose de nouveaux défis. Celui auquel nous nous attaquons dans le domaine de la gestion de la connaissance est de délivrer rapidement la bonne information à la bonne personne au bon moment. Une meilleure connaissance du sens, de l'usage, de l'accessibilité, de la validité des informations permettrait une automatisation facilitée de la gestion du contenu et des services Web.

Dans le cadre de nos travaux, nous nous intéressons à la présentation d'informations issues d'ensembles de données. En section 2.4.1, nous nous intéressons aux méthodes de gestion de données couramment utilisées sur le Web, puis en section 2.4.2, nous nous intéressons aux méthodes de gestion de données proposée par le Web sémantique. Enfin, en section 2.4.3, nous présentons nos conclusions concernant le type de bases de données à prendre en compte par notre formalisme de création d'environnements graphiques 3D.

2.4.1 Gestion de données sur le Web

Une application Web dynamique permet l'accès à de l'information hautement structurée. Cette structuration est souvent gérée par des systèmes de gestion de bases de données. L'application de techniques issues des systèmes de bases de données permet ainsi de faciliter la création et la maintenance de sites Web.

Une base de donnée permet de stocker de l'information structurée en minimisant la redondance. L'utilisation d'un SGDB permet de gérer la base de données, c'est-à-dire l'accès au données, la gestion des utilisateur et la manipulation des données (insertion, suppression, modification). Le modèle de SGBD le plus courant est le Système de Gestion de Bases de Données Relationnelles (SGBDR), basé sur le modèle relationnel décrit par Codd [20]. Les données et leurs relations y sont stockées sous forme de tableaux. Le langage de requête le plus couramment utilisé pour accéder aux données des SGBDR est SQL (Structured Query Language).

2.4.1.1 Exécution de scripts côté serveur

L'utilisation de scripts côté serveur permet de créer dynamiquement des pages Web. Les instructions sont alors insérées dans un modèle de page HTML, et seront exécutées par le serveur afin de produire la partie dynamique de la page. En sortie, le serveur crée une page HTML qui sera directement visualisable par le client. Cette séparation entre contenu statique (le modèle HTML) et dynamique (le script) permet une meilleure maintenance des sites Web et une adaptation du contenu en fonction du besoin et de l'utilisateur, tout en conservant une homogénéité du site Web. Aujourd'hui, un grand nombre de langages de script côté serveur sont disponibles, parmi lesquels PHP, ASP (Active Server Pages), Perl et JSP (Java Server Pages).

2.4.1.2 Architectures trois niveaux

La création et la maintenance d'applications Web complexes, comme les portails Web ou les applications de e-commerce, nécessitent l'utilisation d'un environnement permettant de répliquer facilement chaque composant de l'application. L'architecture la plus répandue sur le Web est composée de trois modules : un module de gestion de données, un module de structuration de l'application et un module de présentation de l'information (voir section 2.1.4). La gestion des données nécessite la connexion à une base de données au moyen d'une interface appropriée. Un certain nombre d'interfaces de programmation (Application Programming Interface ou API) pour bases de données sont disponibles, comme par exemple ODBC (Open DataBase Connectivity), JDBC (Java DataBase Connectivity), DBI (DataBase Interface) et OLE DB (Object Linking and Embedding DataBase). La structuration de l'application est effectuée par l'intermédiaire de requêtes sur la base de données. Enfin, la présentation de l'information est définie à l'aide de feuilles de style.

2.4.1.3 Services Web

L'interconnexion sur le Web permet théoriquement d'accéder à des données gérées par divers SGDB. Cependant, l'accès à des données distantes nécessite l'utilisation de standards. Les *services Web* ont ainsi été créés pour définir un mécanisme de communication entre applications distantes à travers le réseau. Ce mécanisme est indépendant de tout langage de programmation et de toute plateforme d'exécution afin d'assurer une accessibilité maximale aux différentes données.

Les services Web reposent sur trois couches fondamentales :

- l'*échange de message*, représentant les services de communication et de transport de l'information sous forme de messages XML intégrés à des protocoles de type SOAP (Simple Object Access Protocol) ;
- la *description* des services et de leurs opérations, la sécurité et l'authentification dans l'accès aux services, etc. à travers des interfaces de type WSDL (Web Service Description Language) ;
- les *annuaires* référençant des services Web décrits à l'aide d'UDDI (Universal Description Discovery and Integration).

Si les services Web facilitent les échanges entre applications gérées par une même entreprise, leur atout principal est la possibilité d'ouverture vers d'autres entreprises. L'étanchéité est la base du principe des services Web : les clients et les serveurs ne se connaissent qu'au travers des descriptions publiques des services Web. Les services Web acceptent des requêtes transportées sur le réseau sous une forme neutre, indépendantes des technologies retenues pour l'implémentation et le déploiement de l'application finale. Ils répondent à ces requêtes sous cette même forme neutre, assurant ainsi une interopérabilité entre applications distantes. Ils permettent ainsi l'agrégation et la transformation de l'information par des applications tierces.

2.4.1.4 Conclusion

De nombreux systèmes de gestion de données classiques sont disponibles pour créer des applications Web. L'importance croissante des données sémantiques sur le Web laisse cependant présager l'apparition de nouveaux standards pour la gestion de données sur le Web. Dans la section suivante, nous nous intéressons donc au domaine de la sémantique et de ses applications sur le Web.

2.4.2 Gestion de données et Web sémantique

Définition 2.4.1 Une métadonnée (*MetaData*, du grec "*meta*" signifiant "après" et du latin "*data*" signifiant "information") est une donnée servant à définir ou à décrire une autre donnée.

Nous pouvons observer une augmentation significative de l'utilisation de métadonnées enrichissant le contenu en lui ajoutant différents types d'information : mots clés pour faciliter les recherches, taille du contenu, nombre de mots, résumé, technologie utilisée pour créer la donnée, contexte géométrique et colorimétrique dans lequel l'objet est valide (par exemple, "cette image doit être utilisée sur un fond noir"). L'utilisation et la validité de ces données nécessitent un consensus commun. Un ensemble de professionnels d'un domaine peut par exemple se réunir afin de définir les champs à remplir afin de

décrire une donnée de tel type. Le choix des champs est fortement corrélé au domaine choisi pour la description de la donnée. Dans les sections suivantes, nous présentons les concepts de la sémantique permettant une utilisation généralisée des métadonnées, puis nous proposons une étude de l'apport de ces concepts dans le cadre du Web 3D.

2.4.2.1 Généralités

Définition 2.4.2 *La sémantique est l'étude du langage considéré du point de vue du sens ; théorie visant à prendre en compte des phénomènes signifiants dans le langage.*

Source : Le Petit Robert

La sémantique est l'étude des rapports de sens entre différents mots (relations d'homonymie, de synonymie, etc.) ainsi que des conditions de vérité d'un énoncé. En informatique, la sémantique est le complément du syntaxique : alors que ce dernier réfère plus à la forme qu'au concept, la sémantique s'intéresse à son fond. La sémantique extraite d'un ensemble de données offre une cartographie de l'information en construisant un graphe conceptuel. Les sommets du graphe représentant les informations, et les arcs permettent de situer ces informations les unes par rapport aux autres.

2.4.2.2 Web sémantique

Le Web évolue vers une personnalisation de plus en plus fine prenant en compte des critères tels que le contexte d'une recherche ou le profil de l'utilisateur. Il devient dynamique, c'est-à-dire :

- déterminé au fur et à mesure du besoin et de l'évolution du système ;
- ubiquitaire, c'est-à-dire disponible partout, sur tout type de support ;
- sémantique.

Ainsi, le terme *Web sémantique* [7] est proposé par Tim Berners-Lee dans le cadre du World Wide Web Consortium, ou W3C, comme avenir du Web. L'idée sous-jacente de ce concept est d'évoluer d'une "société de l'information" vers une "société de la connaissance".

Le Web actuel pourrait être qualifié de *syntaxique*. En effet, si la structure des documents qui le composent est bien définie, le contenu de ces documents reste majoritairement inaccessible aux machines. Pour pallier ce manque, le Web sémantique propose une infrastructure pour l'utilisation de connaissances formalisées, rendant le contenu sémantique et donc plus accessible aux machines. Les utilisateurs du Web n'ont alors plus accès à l'information mais à la connaissance. Pour que le Web sémantique soit fiable, il est nécessaire d'assurer une interopérabilité maximale ainsi que des possibilités de transfert entre les différents formalismes et ontologies qui seront utilisés. La section suivante présente une étude des différents formalismes les plus développés dans le cadre du Web sémantique.

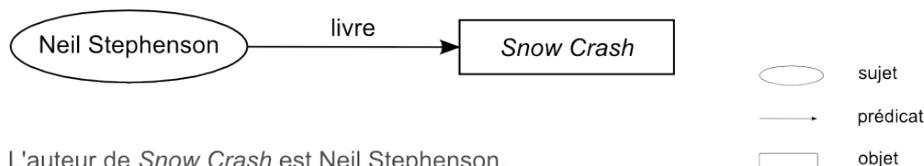
2.4.2.3 Langages sémantiques

Deux grandes familles de standard co-existent dans le monde des données sémantiques : le langage Resource Description Framework (ou RDF [54, 4]) proposé par le W3C et le formalisme des cartes topiques (Topic Maps [27, 67]). Ces standards permettent de décrire des ressources et leurs relations.

RDF a été initialement créé pour supporter la vision du Web sémantique en permettant la gestion de métadonnées sur les ressources. Ce modèle de métadonnées permet de faire des affirmations sur des ressources Web sous la forme d'expressions de type (Sujet, Prédicat, Objet) et appelées *triplets* ou *déclarations* (voir figure 2.4). Une collection de triplets RDF représente un graphe orienté et étiqueté, dont les nœuds correspondent aux Sujets et Objets, et les arcs aux Prédicats.

RDF offre une syntaxe et une sémantique définies strictement, ce qui permet des raisonnements automatisés sur les inférences et conclusions des connaissances, et est particulièrement adapté au Web sémantique.

(a) Neil Stephenson a écrit le livre *Snow Crash*.



(b) L'auteur de *Snow Crash* est Neil Stephenson.



Figure 2.4 – Description RDF de la relation de création entre un auteur et un livre. Chaque relation est exprimé par un triplet (Sujet, Prédicat, Objet).

Les cartes topiques sont issues de recherches permettant d'obtenir un équivalent électronique des index traditionnels du monde de l'édition. Elles permettent de savoir où nous nous trouvons dans l'espace des informations, et de savoir où aller pour trouver l'information recherchée, celle-ci pouvant être présente dans divers ressources.

Tout comme les données RDF, les cartes topiques peuvent être considérées comme un graphe orienté dont les nœuds et les arcs sont étiquetés ([49]). Elles sont basées sur le concept de *thèmes* (*topics* en anglais), d'*associations* et d'*occurrences* (voir figure 2.5). Contrairement aux triplets RDF, les associations sont toujours à double sens et peuvent relier plus de deux thèmes. Les occurrences sont quant à elles les ressources d'information pertinentes pour un thème.

Les cartes topiques permettent de trouver facilement les thèmes essentiels d'un sujet précis. Elles sont ainsi souvent utilisées comme outil de construction de sites Web : la carte topique fournit à la fois la structure et le contenu (composé des thèmes et de leurs occurrences).

Neil Stephenson a écrit le livre *Snow Crash*.

<=>

L'auteur de *Snow Crash* est Neil Stephenson.

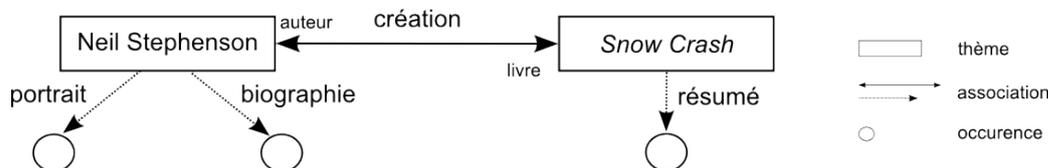


Figure 2.5 – Description dans une carte topique de la relation de création entre un auteur et un livre. Dans cette relation, Neil Stephenson joue le rôle de l'auteur et *Snow Crash* le rôle du livre. Un certain nombre d'occurrences sont associées à chaque thème.

RDF et les cartes topiques sont deux formalismes issus de longs travaux de réflexions ayant une base différente : le Web sémantique dans un cas, et la création d'index de documentation dans l'autre. Un unique formalisme aurait cependant certainement pu voir le jour si chaque communauté avaient eu connaissance du travail effectué par l'autre (Garshol [41]). Actuellement, les deux standards ne sont pas en compétition, chacun étant adapté à des contextes légèrement différents, et il est possible de passer d'un standard à l'autre. Divers travaux ont abondé dans ce sens, comme par exemple Moore [62], Lacher et Decker [56] pour la modélisation de cartes topiques en RDF.

2.4.2.4 Limitations des métadonnées

Résultats de recherche contextuelles, nouveaux services accessibles, la sémantique semble ouvrir de nouveaux horizons au Web. Pourtant, Doctorow [25] présente sept obstacles insurmontables entre le monde dans lequel nous évoluons et un monde de métadonnées exhaustives et fiables. Parmi ces obstacles, nous pouvons noter :

- la *paresse*. Les gens sont paresseux et ne compléteront jamais l'ensemble des champs nécessaires à la description d'une donnée ;
- la *neutralité des schémas*. Il est impossible d'obtenir une représentation unique et universelle d'une donnée, tout schéma n'étant valide que dans un contexte particulier ;
- l'*universalité des descriptions*. Il existe différentes façons de décrire une seule et unique chose.

Malgré ce portrait plutôt sombre des métadonnées, Doctorow insiste sur l'importance de tendre vers ce qu'il nomme la Méta Utopie (Meta Carp en anglais), c'est-à-dire un monde de métadonnées fiables. Certaines métadonnées sont utilisées actuellement de manière intelligente sur le Web : le nombre de liens pointant vers une page Web, la réputation donnée par les utilisateurs, . . . Ces métadonnées implicites sont d'après lui bien plus fortes que les schémas que l'homme pourrait inventer, et permettront de s'approcher de la société de la connaissance recherchée par le Web sémantique.

2.4.3 Conclusion

Dans cette section, nous avons exploré divers formalismes pour la gestion de données sur le Web. Les systèmes de gestion de bases de données sont largement intégrés dans les technologies du Web et permettent l'utilisation d'outils fortement maîtrisés par la communauté Web. Le Web sémantique, s'il reste une utopie, permettrait quant à lui de maximiser la personnalisation du contenu et de rendre la navigation sur le Web et la recherche de connaissances plus efficaces. Dans les deux cas, les formalismes proposés permettent de gérer l'information et d'en extraire le contenu à l'aide de requêtes. L'accès aux données et la structuration du résultat de la requête est donc décorrélié de l'aspect sémantique ou non de l'ensemble de données source, ainsi que du format choisi. Afin d'assurer une compatibilité maximale avec les différentes technologies utilisées ou en cours de développement, notre formalisme doit donc proposer une abstraction de ces requêtes et de leurs résultats.

2.5 Conclusion

Des catalogues d'objets 3D intégrés à des pages Web aux environnements virtuels immersifs en passant par les interfaces graphiques abstraites, les interfaces graphiques 3D sur le Web se multiplient. En parallèle, les usages de la 3D sur le Web se diversifient : jeux vidéo, e-commerce, éducation ou réseaux sociaux, l'apport de la 3D ouvre de nouvelles possibilités dans de nombreux domaines. Cependant, aucun standard 3D ne s'est imposé sur le Web à ce jour. Le développement d'une application 3D interactive composée d'une scène complexe demande donc de faire un choix difficile concernant le format 3D, et nécessite l'installation par chaque utilisateur d'un module d'extension. Les applications ne nécessitant pas l'installation de modules d'extensions sont aujourd'hui encore limitées en terme de complexité de la scène et d'interactivité. Les dernières avancées en matière d'intégration native de la 3D dans les navigateurs Web peuvent laisser présager l'apparition d'un standard permettant la création d'applications 3D interactives complexes visualisables via les navigateurs Web sans module d'extension. Malgré l'appui de nombreux industriels du domaine, la généralisation de cette technologie ne pourra se faire que si elle devient plus mature. Le développement d'un outil auteur pour la création d'interfaces graphiques 3D nécessite donc une abstraction des formats 3D pour assurer l'indépendance face aux formats existants, laissant le choix du format de sortie au créateur de l'interface, et assurant la pérennité de la technologie proposée.

Dans le cadre de notre étude, le problème est équivalent en ce qui concerne l'interopérabilité avec les différents systèmes de gestion d'ensembles de données. Le développement de nombreux outils de gestions de données et de leurs API respectives pour la gestion d'application Web, ainsi que l'émergence d'outils de gestion de données sémantiques, imposent la création d'un outil flexible envers les formats de gestion de données, de requêtes et les résultats de requêtes.

Nous souhaitons donc proposer un formalisme flexible autant en ce qui concerne la gestion des données que le format de la scène 3D finale. Pour ce faire, il doit proposer une abstraction du langage de requête et du langage du résultat de la requête, prenant en compte l'ensemble des paramètres nécessaires à la création d'interfaces graphiques 3D. De même, il doit proposer une abstraction des modèles de graphes de scènes 3D prenant en compte toutes les caractéristiques de la création d'une interface graphique 3D.

Chapitre 3

Production automatique d'interfaces graphiques 3D

De nombreux domaines s'intéressent à la visualisation de données : visualisation de données scientifiques, gestion de ressources, catalogues multimédia, etc. Afin de mieux définir les contraintes et objectifs que doivent respecter notre formalisme pour la production automatique d'interfaces graphiques 3D, nous nous intéressons dans cette section à différentes études concernant les métaphores de visualisation d'information sous trois aspects : la visualisation de la structure interne des données, la structuration des données à visualiser et la présentation de l'information. En section 3.1, nous nous intéressons aux techniques de visualisation des graphes. En effet, quel que soit le formalisme utilisé pour la gestion des données, il est possible d'en extraire un graphe. Nous nous intéressons plus particulièrement aux principes initialement proposés pour la visualisation de graphes et pouvant être étendus à la visualisation de l'information en général. En section 3.2, nous présentons les contraintes principales de la structuration des données afin d'offrir une navigabilité optimale et un accès efficient à l'information pour les utilisateurs finaux. En section 3.3, nous décrivons un ensemble de systèmes permettant la description de métaphores de présentation et d'interaction. Enfin, en section 3.4, nous concluons ce chapitre par un récapitulatif des besoins nécessaires à la définition d'un formalisme pour la création d'interfaces graphiques 3D.

3.1 Techniques de visualisation des graphes

Une application Web dynamique permet l'accès à de l'information structurée (voir section 2.4.1). Il est donc possible de décrire le résultat de chaque requête effectuée sur le système de gestion de données par les nœuds d'un graphe, ses arcs représentant les relations entre les données. Dans cette section, nous étudions donc diverses techniques pour la visualisation de graphes, et plus spécifiquement la visualisation de graphes dans le cadre de la navigation et de la représentation de l'information. Herman *et al.* [48] proposent une étude détaillée des différentes techniques de visualisation existantes et de leur utilisation dans le cadre de la navigation dans une grande quantité d'information. Nous traitons par la suite les points les plus pertinents dans le cadre de notre problématique : la simulation de systèmes masse-ressort, l'affichage de données en utilisant la technologie fish-eye, et le regroupement d'information (clustering en anglais). Enfin, nous présentons les avantages et inconvénients des techniques de visualisation de graphes dans le cadre de la création automatique d'interfaces graphiques 3D.

3.1.1 Généralités

Définition 3.1.1 *Un graphe est un triplet $G = (S, A, \phi)$, où*

- S est l'ensemble des sommets de G ;
- A est l'ensemble des arêtes de G ;
- $\phi \in \Phi, \Phi : A \rightarrow S \times S$ est une fonction d'incidence.

Soit $\phi(a) = (u, v)$, alors a est l'arête reliant les sommets u et v . a et u sont alors incidents, de même pour a et v ; u et v sont adjacents et sont les extrémités de a .

Si le graphe est orienté, A est appelé ensemble de arcs du graphe. u est alors l'extrémité initiale de a et v son extrémité terminale. De plus, deux arcs sont consécutifs s'ils sont adjacents et si leur extrémité commune est initiale pour un arc et terminale pour l'autre.

Définition 3.1.2 Une chaîne est une suite de sommets reliés les uns aux autres par des arêtes.

Définition 3.1.3 Un cycle est une chaîne dont l'extrémité coïncide avec l'origine. Un graphe ne contenant aucun cycle est un graphe acyclique.

Définition 3.1.4 Un graphe connexe est un graphe dans lequel chaque paire de sommets est relié par une chaîne.

Définition 3.1.5 Un arbre est un graphe connexe et acyclique.

Types de graphes Les graphes manipulés dans le cadre de la création automatique d'interfaces graphiques 3D ne sont pas limités aux arbres. Notre but étant la visualisation de l'information, nous nous intéressons aux modes de représentation permettant à l'utilisateur final de naviguer de manière intuitive à l'intérieur du graphe.

Sémantique et plastique D'après Herman *et al.* [48], la visualisation de graphes est principalement contrainte par la sémantique d'une part et la plastique d'autre part. Du point de vue sémantique, il peut être intéressant de mettre en avant certaines étiquettes liant plusieurs données par rapport à d'autres. Par exemple, si nous avons un produit à vendre, nous pouvons mettre en avant la liaison correspondant au prix des articles, celle correspondant au type de produit vendu, ou encore la marque de l'article. Le choix sémantique est déterminant pour le résultat obtenu, la forme elle-même du graphe étant impactée par cette décision.

Les contraintes plastiques quant à elles peuvent préciser par exemple si les arêtes du graphe peuvent se croiser ou non, ou encore déterminer le motif général du graphe, etc. Loin d'être de simples "critères de beauté", les contraintes plastiques influent sur la navigation en indiquant de manière plus ou moins franche le passage d'un sous-ensemble de données à un autre, en mettant en avant certaines informations au détriment d'autres. Par exemple, un motif circulaire n'aura pas le même effet sur un ensemble de données qu'un motif linéaire.

Chaque modèle proposé pour la visualisation de graphes permet de contrôler de manière plus ou moins précise ces deux contraintes primordiales pour la visualisation de l'information.

Visualisation de l'information

Définition 3.1.6 *Problème fondamental du tracé de graphes :*

Étant donné un ensemble de nœuds et un ensemble d'arcs, calculer la position des nœuds et la courbure de chaque arc.

La taille du graphe ne doit pas être une contrainte à la création d'une interface graphique 3D. Or la difficulté principale en visualisation de graphes est la taille du graphe. En effet, il n'est pas toujours possible d'afficher l'ensemble du graphe de façon à assurer la possibilité de discerner chaque nœud et chaque arc du graphe. Lorsque cette contrainte est respectée, l'utilisabilité du graphe (par exemple l'accès aux différents nœuds du graphe ou la lecture de labels associés aux arcs) pose souvent problème. De nombreuses méthodes ont été développées pour permettre une visualisation optimale des graphes, qu'ils soient de forme purement hiérarchique ou non, et respecter le problème fondamental du tracé de graphes (voir définition 3.1.6).

Prédictabilité et carte mentale de l'utilisateur

Définition 3.1.7 *Un algorithme est prédictible si deux calculs à partir du même graphe ou de graphes similaires ne produisent pas des visualisations radicalement différentes.*

Herman *et al.* [48]

Le visiteur d'une interface graphique (qu'elle soit immersive ou non) crée une carte mentale du monde dans lequel il évolue. Cette carte est entre autre basée sur la forme des chemins de navigation et sur la proximité des différentes informations (voir section 3.2 pour plus de détails). Les interfaces graphiques que nous proposons de créer sont persistantes. Un visiteur peut s'y connecter plusieurs fois mais ne retrouvera pas chaque fois l'environnement tel qu'il l'avait laissé ; d'autres visiteurs ont pu le modifier, du contenu a pu être ajouté ou supprimé entre ses visites. Afin de ne pas perdre le visiteur et de le fidéliser, il est nécessaire qu'il soit capable de se repérer rapidement dans une interface graphique qu'il a déjà visitée auparavant. Il doit donc être capable de réutiliser la carte mentale qu'il a formée lors de sa précédente visite, et ne doit lui apporter qu'un nombre limité de modifications. Dans l'étude que nous faisons des différentes méthodes de visualisation de l'information, nous portons donc une attention particulière à la prédictibilité de l'algorithme.

Complexité temporelle La dernière contrainte concernant notre système est la complexité temporelle. En effet, la visualisation du graphe, l'interaction éventuelle entre avatars, et les interactions environnement-avatar doivent être gérées en temps réel. De plus, nous souhaitons nous approcher autant que possible de la production de l'environnement en temps réel pour permettre la visualisation instantanée du nouvel environnement lors de l'ajout ou de la suppression de contenu ou l'utilisation de notre formalisme pour l'affichage de résultats de recherche. Enfin, les environnements créés sont accessibles depuis le Web, ce qui implique de réduire au maximum le temps de chargement et de transfert des données.

3.1.2 Systèmes masse-ressort

Eades [28] propose une visualisation de graphes basée sur la notion physique d'attraction et de répulsion. Il compare alors les arêtes des graphes à des ressorts et les sommets à des masses, en analogie avec la loi physique de Hook. Le système ainsi constitué engendre des forces entre les sommets, entraînant leurs déplacements respectifs. Eades ajoute des forces de répulsion et d'attraction entre les sommets (correspondant à des masses chargées), par l'intermédiaire des arêtes. Après une phase de transition, le système se stabilise et la position finale des sommets est alors considérée comme une configuration satisfaisante du graphe. Si d'un point de vue physique, l'équilibre est atteint pour un minimum énergétique du graphe, Eades choisit un nombre donné d'itérations comme condition d'arrêt. Les avancées techniques et de nombreux développements de l'algorithme proposé ont permis d'améliorer les résultats obtenus par Eades (Frick *et al.* [39], Bertault [8], Brandenburg *et al.* [14]). Ces modèles sont généralement appelés FDP (Force Directed Placement). Dans le cadre de notre étude, cette méthode présente un désavantage majeur : dans certains cas, une petite modification à l'entrée peut produire un résultat très différent à la sortie. En effet, la position d'un sommet du graphe dépend de la position de l'ensemble des autres sommets, ainsi que des relations exactes qu'ils entretiennent. La prédictibilité n'est donc pas assurée et il sera difficile de fidéliser l'utilisateur final.

3.1.3 Focales variables et projection Fish-eye

La visualisation d'un graphe contenant un nombre élevé de sommets peut poser un certain nombre de problèmes, parmi lesquels la distinction entre les différents sommets et arêtes du graphe, les occultations ou encore le croisement des arcs. De plus, il est difficile de traiter un trop grand nombre de données à la fois. L'affichage direct d'un graphe complet peut donc gêner l'utilisateur et ralentir son accès aux données recherchées. L'utilisation d'une focale variable, ou zoom, permet de n'afficher qu'une partie du graphe à la fois. Le zoom peut être de nature géométrique, tel le zoom d'un appareil photo, ou sémantique. Dans le premier cas, la taille des différents éléments est modifiée lors du zoom, tandis que dans le second, le contenu change et plus de détails sont présentés au fur et à mesure que la caméra s'approche.

Le zoom géométrique permet de conserver la topologie globale du graphe. L'affichage des éléments du graphe à un niveau de zoom donné peut, par exemple, être basé sur un ensemble de valeurs (ne sont montrés que les nœuds dont la valeur est supérieure à un palier p). Lorsque la caméra s'éloigne, les données géométriquement proches peuvent être regroupées sous forme d'un super nœud, être

effacées, ou être mises en transparence. Dans le cas d'un zoom sémantique, les données sémantiquement proches sont regroupées et représentées par un nœud unique étiqueté convenablement (par exemple, une étiquette "couleur" pourra être utilisée pour regrouper les nœuds "vert", "violet" et "bleu"). Afin de pouvoir visualiser l'ensemble des données du graphe à différents niveaux de détail, il est nécessaire de coupler la fonction de zoom avec celle de déplacement. Furnas et Bederson [40] introduisent le concept de diagrammes espace-échelle : ils construisent une pyramide inversée comprenant les images du graphe à différents niveaux de détail. Une combinaison de zooms et de déplacements judicieux à travers les cartes obtenues permet de trouver le chemin optimal de déplacement d'un point du graphe à un autre. Leur méthode s'applique aussi bien au cas du zoom géométrique qu'à celui du zoom sémantique.

Si ces méthodes permettent de naviguer de manière plus intuitive dans un graphe de grande dimension, leur inconvénient principal est la perte de contexte lors du zoom. L'utilisateur final risque de se perdre lors de la navigation et d'avoir des difficultés à gérer les différents niveaux de zoom. Pour conserver un changement d'échelle (géométrique ou sémantique) lors de la navigation tout en gardant des informations contextuelles, il est possible d'afficher différentes parties du graphe à différents niveaux d'abstraction. La projection Fish-eye (figure 3.1) s'inspire du concept des objectifs Fish-eye

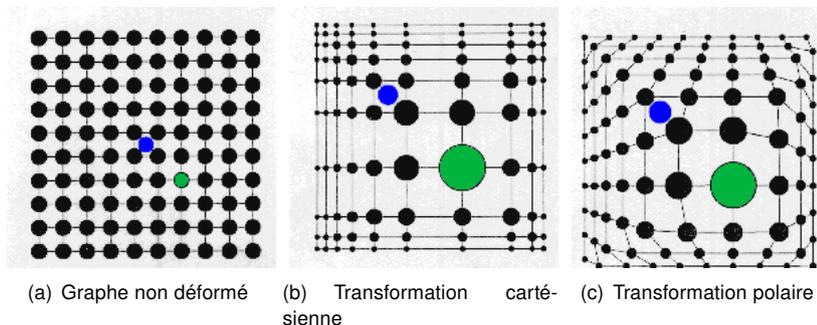


Figure 3.1 – Projections Fish-eye d'une grille régulière. L'élément vert représente le centre d'intérêt du graphe, le point bleu représente la position du centre du graphe avant déformation. Une fois la projection effectuée, l'élément vert peut être affiché à un degré d'abstraction différent des autres éléments du graphe.

utilisés en photographie et offre une solution intéressante. L'utilisateur définit un point de focus, correspondant à la partie du graphe sur laquelle il désire zoomer. L'application d'une fonction de déformation $h(x)$ à chaque sommet du graphe où x est la distance du sommet au point de focus permet de modifier l'affichage du graphe. Plus un sommet sera éloigné du point de focus, plus il sera représenté sous sa forme abstraite (transparence, regroupement, etc.). La fonction $h(x)$ doit être convexe, de l'intervalle $[0, 1]$ vers l'intervalle $[0, 1]$, avec un incrément plus fort autour de 0, puis diminuant en s'approchant de 1. Sarkar et Brown [76] proposent d'utiliser la fonction $h(x) = (d + 1)/(d + 1/x)$, où $d > 0$ est le facteur de distorsion ajusté interactivement par l'utilisateur. Plus d est grand, plus la distorsion est forte. Un avantage de la projection Fish-eye est que l'algorithme s'insère dans une étape finale dans le pipeline graphique, permettant de l'appliquer après une autre méthode d'affichage ou de regroupement d'information par exemple. Si cette technologie n'intervient qu'après la construction de la visualisation du graphe à l'aide d'une autre technique (masse-ressort par exemple), elle permet une meilleure navigation dans l'information en rappelant à l'utilisateur le contexte de la donnée visualisée.

3.1.4 Regroupement de données

Les graphes utilisés pour les techniques de focale variable et Fish-eye permettent le regroupement d'information (clustering en anglais) soit de manière géométrique, soit de manière sémantique. Dans les deux cas, un algorithme de regroupement d'information peut être appliqué à un graphe quelconque avant de pouvoir utiliser ces méthodes. Comme nous l'avons vu, le regroupement d'information permet de réduire le nombre d'éléments visibles en regroupant différents sommets du graphe proches géométriquement ou sémantiquement pour obtenir un sommet unique les représentant. La plupart des algorithmes proposés sont basés sur la découverte de regroupements en fonction de la structure du

graphe. En effet, lorsqu'ils sont basés sur le contenu sémantique du graphe, les algorithmes à utiliser sont souvent spécifiques au domaine et au contexte, ce qui les rend difficilement généralisables. De plus, les algorithmes basés sur la structure du graphe permettent de conserver la forme globale originale du graphe. L'utilisateur conserve donc une bonne orientation lors du passage d'un niveau d'abstraction à un autre. Les algorithmes de regroupement d'information facilitent à la fois la recherche (mise en avant d'un ou plusieurs éléments du groupe) et le filtrage (retrait d'éléments) d'information. Les groupes d'information obtenus peuvent être exclusifs (un élément appartient à un unique groupe) ou peuvent se chevaucher (un élément peut alors appartenir à plusieurs groupes). En général, les algorithmes ont une fonction d'optimisation recherchant l'ensemble des groupes tel qu'il y ait le moins d'arrêtes possibles dans le graphe résultant. Dans le cas des graphes dont les arrêtes comportent un poids, alors l'optimisation est atteinte lorsque les poids des arrêtes entre groupes sont minimaux. Une fois les groupes trouvés, il est possible de n'afficher qu'eux, ce qui permet de réduire le nombre de nœuds à afficher, tout en conservant le contexte. Les groupes peuvent alors être comparés à des super-nœuds.

Le regroupement de données peut permettre de regrouper géométriquement des données en fonction de la sémantique. L'algorithme utilisé est alors similaire à celui effectué pour le regroupement géométrique. Il devra cependant tenir compte d'éventuels éloignements géométriques de données proches sémantiquement. En effet, l'espace de représentation géométrique du graphe possède 1, 2 ou 3 dimensions dans un environnement virtuel, or l'espace sémantique peut être de dimension supérieure. Associé à une technologie Fish-eye, le regroupement de données permet de visualiser le contexte sous forme de super-nœuds, tandis que le groupe de données sur lequel se porte l'intérêt de l'utilisateur final est entièrement visualisé. Cette technique peut cependant poser problème pour la création des arêtes entre les super-nœuds du contexte et les nœuds en cours de visualisation.

3.1.5 Conclusion

Nous cherchons à visualiser des éléments issus d'un ensemble de données. Celui-ci peut être représenté sous forme d'un graphe dont les nœuds représentent les éléments, et les arêtes représentent les relations entre ces éléments. Il semble donc naturel de proposer une visualisation de cet ensemble de données basée sur les techniques de visualisation de graphe. C'est le cas par exemple de Redgraph (Halpin *et al.* [46]), un outil générique permettant de visualiser un ensemble de données sémantiques Web sous forme de graphe à travers une interface de réalité virtuelle. L'utilisateur est installé dans un CAVE (Cave Automatic Virtual Environment) et visualise le graphe sémantique sous forme d'un réseau 2D. Il peut alors extruder les nœuds du graphe sur lesquels il porte son intérêt pour les positionner dans l'espace 3D, mettant ainsi en avant les liaisons de ce nœuds avec le reste du graphe.

Cependant, Schraefel et Karger [77] posent la question de la pertinence de la présentation de données telles qu'elles sont représentées par le système informatique. Par exemple, il ne serait pas productif de présenter les fichiers et dossiers conservés en mémoire sous forme de listes de paires (nomDeFichier, inode) et (inode, infoFichier). De plus, la visualisation de l'ensemble des données avec le même degré d'importance n'a généralement pas de sens lors de la recherche d'information. Certains nœuds du graphes pourront alors être considérés comme des nœuds principaux, tandis que d'autres seront considérés comme des attributs. Dans tous les cas, le problème de la visualisation de données est de pouvoir spécifier la meilleure visualisation, la meilleure représentation et la meilleure interaction possible pour une tâche ou un utilisateur précis. Il serait donc plus approprié de proposer un formalisme pour la définition de métaphores de visualisation de données n'étant pas basé sur la structure sous-jacente de l'ensemble de données mais sur le domaine et les besoins de l'utilisateur.

Cette section présente un certain nombre de technologies, comme la projection Fish-eye ou le regroupement de données, permettant de mieux tirer partie de la visualisation d'un graphe en proposant une observation fine des éléments intéressants tout en conservant le contexte de leur positionnement dans le graphe. De plus, de nombreux concepts, tels que la prédictibilité, ont été approfondis par les différentes recherches sur la visualisation de graphes. La définition d'un formalisme pour la définition de métaphores visuelles pourrait ainsi tirer parti de l'intégration de principes de visualisation de graphes tels que la projection Fish-eye et le regroupement de données.

3.2 Navigation et structuration

La définition d'une interface graphique intuitive permet de réduire de manière significative la charge cognitive de l'utilisateur. Elle facilite l'utilisation d'outils naturels d'observation et de traitement que sont les yeux et le cerveau, entraînant une compréhension et une extraction de la connaissance plus efficace qu'en l'absence d'interface graphique (Gershon et Eick [42]). La perception de la structure des données et la navigation sont alors fortement facilitées.

Comme nous l'avons vu dans la section précédente, la présentation de l'information telle qu'elle est organisée et représentée en mémoire de l'ordinateur n'est généralement pas adaptée à la visualisation et la navigation dans un ensemble de données. Or la visualisation d'information permet à l'utilisateur d'interagir avec le contenu afin de mieux l'appréhender. Ce processus étant généralement de nature exploratoire, nous étudions dans cette section les concepts nécessaires à la structuration des données en fonction du contexte (domaine d'application, besoin, utilisateur) afin de faciliter la navigation et l'accès à la connaissance.

3.2.1 Navigation

L'intuitivité et la convivialité des interfaces graphiques 3D est fortement liée à nos habitudes de navigation et d'interaction dans le monde réel. Afin de mieux comprendre les enjeux de la navigation et de l'interaction dans les interfaces graphiques 3D, nous étudions dans cette section la navigation dans le monde réel. Puis, à partir des observations présentées, nous présentons la nécessité de la structuration des interfaces graphiques 3D pour faciliter l'accès à l'information.

Définition 3.2.1 *La navigation est une activité spatiale guidée par des informations visuelles de l'espace, et un ensemble de mouvements, permettant d'atteindre une destination à partir d'un point de départ.*

Afin de rejoindre une destination précise dans le monde réel, un *navigateur* a besoin d'une connaissance de l'espace qui l'entoure, et de contrôler ses mouvements pour se déplacer dans cet espace. Pour améliorer son orientation, il est donc nécessaire d'améliorer le couple connaissance de l'espace et des mouvements. Berthoz [9] présente deux stratégies de mémorisation de trajets dans l'espace. La première, appelée *stratégie de route*, est basée sur la mémorisation des mouvements effectués par le sujet. Lors de sa navigation, il associe des repères visuels et des événements aux mouvements qu'il effectue. Ses souvenirs étant ainsi définis par son point de vue, cette stratégie est dite subjective. La deuxième stratégie, appelée *stratégie de survol*, est basée sur la mémorisation de cartes géographiques, de plans, entraînant une mémorisation d'images virtuelles et permettant au sujet de définir des distances relatives, des raccourcis, ou encore de changer de point de vue. Cette stratégie étant déconnectée du point de vue du sujet, elle est dite allocentrique. De nombreuses stratégies de mémorisation de trajets situés entre ces extrêmes peuvent être utilisées, les complétant ou les remplaçant. Dans le paragraphe suivant, nous présentons une stratégie en partie subjective et en partie allocentrique faisant appel à la notion de hiérarchie de navigation.

L'amélioration du couple connaissance de l'espace et des mouvements doit s'effectuer à plusieurs échelles. Lynch [58] insiste sur l'importance de la structuration d'un environnement pour aider l'utilisateur à naviguer dans cet environnement. De nombreuses études ont également démontré l'importance de la connaissance de l'espace à différents niveaux d'abstraction lors de la navigation dans le monde réel (obstacles sur le chemin comme une branche, vision plus globale du chemin à effectuer pour atteindre un but, ...). Pour s'élever au niveau d'abstraction supérieur, l'utilisateur regroupe les objets du niveau actuel pour obtenir un composant unique au niveau supérieur. Il pourra ainsi regrouper un ensemble de rues et de bâtiments en un quartier, puis un ensemble de quartiers en une ville. En construisant ainsi sa représentation spatiale de l'environnement externe, l'utilisateur se construit une *carte mentale* hiérarchique qui lui permet non seulement de naviguer dans le présent, mais qu'il pourra réutiliser s'il revient approximativement au même endroit (Tolman [87]). Chase [17] a étudié le comportement de chauffeurs de taxis pour mettre en évidence l'utilisation de la carte mentale lors de la navigation, tandis que Lynch [58] démontre l'importance de la structuration et de l'organisation d'un environnement pour aider à la navigation.

Dans le monde réel, lorsqu'une personne souhaite rejoindre un lieu précis, elle choisit une chaîne de

transports lui permettant de rejoindre son objectif. Ce choix est facilité par la représentation hiérarchique que nous avons de notre environnement. Cette personne pourra par exemple rejoindre le pays de sa destination par avion, puis la ville en prenant le train, le quartier en métro, et enfin l'adresse exacte à pied. Ce processus est transposé dans des interfaces graphiques type Google Earth, où les différents niveaux d'abstractions sont représentés à l'aide de zooms, la navigation restant la même quel que soit le niveau d'abstraction visualisé. Dans le cadre de la présentation de l'information, la carte mentale créée par l'utilisateur n'est pas horizontale, comme dans le cas de la navigation dans le monde réel, mais verticale. La navigation entre deux niveaux d'abstraction correspond à une navigation en profondeur dans l'arbre de représentation de l'information. La structuration de l'information permet alors de faire des raccourcis impossibles dans la vie réelle, et d'accélérer le processus de recherche d'information.

Les travaux de Witmer *et al.* [95] et Ruddle *et al.* [73] indiquent une forte corrélation entre la navigation dans le monde réel et la navigation dans un environnement virtuel. Cependant, les mouvements du corps étant très limités dans le cadre de la navigation dans un environnement virtuel, Elvins *et al.* [29] insistent sur l'importance de l'information visuelle pour améliorer la navigation dans un environnement 3D. L'ajout d'une carte 2D ou d'une vue 3D du monde simplifié permet également d'améliorer la navigation dans un univers virtuel 3D. Si l'interaction est limitée lors de la navigation dans un environnement virtuel, le passage d'un niveau d'abstraction à un autre est facilité puisqu'il n'est pas nécessaire de suivre les lois qui régissent le monde physique. Zhang [99] propose ainsi un modèle progressif multi-échelle pour la navigation en environnement virtuel. L'utilisateur peut alors visualiser le monde, agir et naviguer à différentes échelles. Ce modèle permet donc un meilleur couplage entre connaissance de l'espace et des mouvements à différentes échelles lors de la navigation en environnement virtuel.

Dans la section suivante, nous tentons de comprendre ce qui, dans la structuration d'un environnement, permet à l'utilisateur de mieux appréhender l'espace dans lequel il évolue, mais aussi d'améliorer les interactions éventuelles entre les utilisateurs d'un même environnement.

3.2.2 Structuration et urbanisme

Dans le monde réel, la navigation est fortement influencée par l'urbanisme et la structuration des espaces. Dans cette section, nous étudions les concepts urbanistiques favorisant ou non la navigation et l'interaction dans le monde réel, puis nous transposons ces concepts dans le domaine des interfaces graphiques 3D.

De nombreuses études sur l'urbanisme, qu'il soit virtuel ou réel, permettent de mettre en avant l'importance de la structuration d'un environnement pour favoriser la navigation et l'interaction. Batty et Longley [3], Frankhauser [37] décrivent la ville comme une structure multi-fractale. Différents processus et mécanismes urbains y ont lieu à différentes échelles. Chaque échelle est nécessaire et ne doit pas être éliminée, de plus, chaque échelle nécessite son propre réseau de communication. Par exemple, limiter le réseau piétonnier urbain à un espace longeant le réseau automobile rend le réseau piétonnier redondant et incite les usagers à privilégier l'utilisation d'un véhicule motorisé pour tout type de déplacement, limitant ainsi les interactions entre usagers du réseau et impliquant un besoin croissant en zones de stationnement envahissant l'espace urbain.

Salingaros [75] propose une étude de la configuration des réseaux routiers dans les agglomérations au fil des âges, et de son influence sur la navigation et les interactions entre utilisateurs. Pour lui, la caractéristique principale d'un réseau de communication est la géométrie de ses connexions. Celle-ci influencerait en effet fortement les mouvements des hommes, les incitant ou non à utiliser différents modes de déplacement et d'interaction. Dans son étude, Salingaros [75] distingue deux types principaux d'éléments urbains : les nœuds et les connexions. Les nœuds correspondent à des positions spécifiques dans l'espace ayant une fonction, comme par exemple les bâtiments d'une ville. Les connexions quant à elles permettent de relier les nœuds. Dans la majorité des cas, les connexions correspondent au réseau routier, ferroviaire, aérien ou aux réseaux virtuels.

Cette analogie avec le vocabulaire de la théorie des graphes permet une adaptation de l'analyse de Salingaros au problème de la visualisation de grands ensembles de données. Dans le domaine de l'urbanisme, il est essentiel de faciliter et d'accélérer le transport à différents niveaux de fractalité. Pour se rendre à une adresse spécifique, une personne pourra utiliser les voies aériennes pour rejoindre le pays, puis les voies ferroviaires pour rejoindre la ville, les routes pour rejoindre le quartier, et enfin les trottoirs pour rejoindre l'adresse en question. Imaginons à présent l'utilisation d'une métaphore basée

sur les pays, les villes et les quartiers pour représenter l'information. L'utilisateur choisira d'abord une catégorie à travers la visualisation d'un ensemble de pays, puis une sous-catégorie à travers la visualisation du pays choisi décomposé en villes, puis un quartier et enfin l'information recherchée. Quelle que soit la métaphore choisie pour représenter les données, l'utilisateur aura ainsi besoin de réseaux de navigation à différentes échelles.

Salingaros [75] distingue deux grandes catégories d'agglomérations : les villes pré-modernes (datant d'avant le XXe siècle), et les villes modernes (du XXe siècle à nos jours). Il définit les premières comme étant fractales, c'est-à-dire possédant des composants urbains et des connexions à différentes échelles (nombreux ornements de façades, statues, fontaines décorées ; ruelles, rues, routes reliant les bourgs, . . .). Il considère la perte de fractalité des villes modernes comme pathologique pour la ville. En effet, le réseau de communication permet d'établir des connexions et des échanges d'information entre les différents nœuds de l'agglomération. Épuré le réseau entraîne une manque de communication et d'interactions entre les utilisateurs de la ville.

D'après Salingaros [75], la fractalité joue un rôle essentiel pour la navigation et l'interaction à l'intérieur d'un réseau. Appliquée à une agglomération réelle, sa théorie implique une compétition spatiale entre les réseaux d'échelles différentes (piéton, cycle, voiture, transports en commun). En effet, bien que l'interface entre eux doit être élevée pour un bon fonctionnement d'ensemble, un réseau ne doit pas en rendre un autre redondant, ni créer un risque pour l'utilisateur d'un autre réseau (nécessité récurrente de traverser des voies automobiles pour un piéton). Notre étude transpose cette théorie du monde physique vers un univers virtuel permettant de se libérer des contraintes physiques. Cette capacité nous autorise à créer autant de réseaux de navigation que nécessaire au degré de fractalité de chaque environnement créé, sans avoir à se soucier d'une quelconque compétition pour l'espace. La navigation à travers une interface graphique non immersive nécessite de la même façon de proposer plusieurs réseaux de navigation. La solution choisie pour la création d'interfaces graphiques 3D devra donc permettre de naviguer à travers l'information à différents niveaux d'abstraction, représentant chacun un mode de transport, et possédant chacun son propre réseau de navigation.

3.2.3 Conclusion

La prise en compte de la navigation lors de la construction d'une interface graphique 3D, qu'elle soit immersive ou non, permet un accès aux données plus intuitif. L'utilisateur pourra alors utiliser différentes stratégies pour évoluer dans l'interface graphique et se construire une carte mentale lui permettant d'inférer les propriétés de l'interface et d'accéder plus rapidement aux données lors de sa prochaine visite, mais aussi aux données présentes dans un espace non encore exploré et de structure similaire. Comme nous l'avons vu dans cette section, la création de réseaux de navigation à différentes échelles est primordiale. La structuration des données présentées au travers de l'interface graphique doit donc refléter cette possibilité d'exploration des données à différents niveaux.

Le but de cette étude n'est pas de proposer une interface graphique 3D universelle, ni même spécifique pour un domaine précis, mais un formalisme pour la description d'interfaces graphiques 3D. Celui-ci doit pouvoir être utilisé pour présenter tous types de données, à tous types d'utilisateurs, dans tous types de contexte. Afin de permettre aux auteurs de créer une interface graphique adaptée aux besoins de leurs utilisateurs, notre formalisme doit prendre en compte les problèmes de navigation et de structuration présentés dans cette section. Pour cela, il doit proposer des outils adéquats permettant la définition de la structuration de l'application et des modes d'accès aux données présentées à travers l'interface graphique 3D.

3.3 Métaphores 3D de visualisation de l'information

L'utilisation d'une interface graphique pour la visualisation et la navigation dans un ensemble de données offre de nombreux avantages. Cependant, le potentiel de l'ensemble de données comme source d'information pour l'utilisateur final ne peut être entièrement exploité que s'il est possible de :

- structurer l'information à l'aide de requêtes précises exprimant les besoins de l'utilisateur ;
- présenter les résultats de ces requêtes aux utilisateurs de manière cognitivement efficace, par exemple à travers la prise en compte des connaissances *a priori* qu'a l'utilisateur final sur le domaine des données présentées.

La structuration de l'information permet de définir la topologie de l'interface graphique. La section 3.2 présente les principes de navigation et de structuration nécessaires à la conception de cette topologie. Dans cette section, nous nous intéressons à la présentation des résultats des requêtes, c'est à dire à la métaphore visuelle de l'interface graphique.

Définition 3.3.1 *Une métaphore est une figure par laquelle, se fondant sur une comparaison implicite, on use, pour désigner quelque chose, d'un terme, d'une expression qui, au sens propre, s'applique à une autre réalité.*

Dictionnaire de l'Académie française, neuvième édition

Une métaphore est donc une figure rhétorique permettant de comprendre et d'expérimenter un concept à l'aide d'un autre. Selon Martin [60], une métaphore est ainsi divisée en deux composants regroupant chacun un ensemble de concepts : le composant cible regroupant l'ensemble des concepts que nous cherchons à analyser, et le composant source représentant l'ensemble des concepts qui seront utilisés pour représenter le composant cible. Une mise en correspondance, sous forme d'un ensemble d'associations ou de relations, permet de passer du composant source au composant cible. Une interface graphique permet de représenter les éléments d'un ensemble de données. Ceux-ci peuvent être soit abstraits, soit des abstractions d'éléments du monde réel. La définition d'une métaphore permet alors d'effectuer une mise en correspondance entre l'espace logique des éléments de l'ensemble de données et l'espace de représentation visuelle. D'après Mountford [63] et Erickson [30], la définition de la métaphore est une étape fondamentale du processus de création d'interface car elle permet à l'utilisateur final de travailler dans un environnement à la fois convivial et efficient. Dans cette section, nous présentons différents formalismes permettant de visualiser un ensemble de données sous la forme d'une interface graphique 3D.

3.3.1 Métaphores génériques pour la visualisation de données

De nombreux travaux de recherche proposent des métaphores génériques, souvent abstraites, permettant la visualisation de tous types de données.

Définition 3.3.2 *Une métaphore générique est une métaphore pouvant être appliquée sur tous types de données en entrée du système.*

Une métaphore générique, qu'elle soit abstraite ou non, ne peut donc pas directement prendre en compte le domaine des données à visualiser. Cependant, les systèmes composés de métaphores génériques peuvent prendre en compte un ensemble de paramètres qualitatifs (type des données à visualiser) et quantitatifs (nombre de données à visualiser) pour définir la meilleure métaphore générique pour visualiser un ensemble de données.

Robertson *et al.* [72] proposent la visualisation d'informations hiérarchiques sous forme d'arbres coniques, tandis que Bosca *et al.* [12] proposent la visualisation d'ontologies à l'aide de sphères annotées. L'ensemble de données en entrée du système n'est pas connu a priori. Les relations structurelles des éléments sont présentées par leur positionnement relatif ou à l'aide de connecteurs physiques, comme des tuyaux ou des lignes. Les éléments d'information sont quant à eux généralement présentés sous forme d'objets 3D abstraits. Dans cette section, nous présentons trois systèmes permettant d'appliquer plusieurs types de métaphores visuelles à divers ensembles de données.

3.3.1.1 Métaphores 3D pour la visualisation d'information

Robertson *et al.* [71] présentent Information Visualizer, un système expérimental permettant d'explorer des paradigmes de visualisation 3D pour la manipulation et la compréhension de grands ensembles de données. Des agents sont utilisés pour améliorer l'accès à l'information et la structurer à l'aide d'algorithmes de regroupement non linéaires. Chaque élément d'information est représenté sous la forme d'un *objet interactif* : un texte, un bouton, une porte, etc. Parmi les métaphores proposées, on trouve un arbre conique pour les données hiérarchiques, un mur en perspective pour la visualisation de données linéaires, un bâtiment pour la visualisation de carnets d'adresses ou encore une grille d'information pour la visualisation de données non structurées. Dans ces travaux, le choix de métaphores abstraites est justifié car elles permettent une assimilation plus rapide de l'information et une détection de motifs facilitée.

3.3.1.2 Visualisation de recherches sur le Web

Wiza *et al.* [97] présentent un système appelé Periscope permettant de construire une visualisation 3D de résultats de recherche sur le Web. Le processus de recherche est décomposé en un ensemble de requêtes, chaque requête correspondant à un raffinement de la requête précédente. Chaque requête représente alors un *niveau d'abstraction* dans la recherche d'information. La première requête (niveau d'abstraction le plus élevé) permet d'obtenir une agrégation de catégories. Au fur et à mesure des recherches, les sous-arbres résultats représentent de nouvelles catégories, jusqu'à la requête finale proposant une visualisation fine de l'information recherchée. Une métaphore visuelle, choisie parmi un ensemble de métaphores disponibles, est appliquée à chaque niveau d'abstraction afin de permettre une visualisation optimale des données présentées.

Le choix de la métaphore, basé sur la méthode AVE (Wiza *et al.* [96]), peut être automatique, semi-automatique ou manuel. Trois types de métaphores sont proposées par la méthode AVE : les métaphores *holistiques* permettant de présenter des données regroupées en catégories, les métaphores *analytiques* permettant de représenter les données au niveau de détail le plus fin, et les métaphores *hybrides* permettant de visualiser à la fois les catégories et les données détaillées. Outre sa catégorie, chaque métaphore contient un ensemble de propriétés. La sélection consiste alors en un processus algorithmique choisissant la meilleure mise en correspondance possible entre les attributs qualitatifs (par exemple le type des données disponible, comme du texte ou une image) et quantitatifs (nombre d'éléments à représenter) du résultat de la requête en cours et les propriétés d'une métaphore. Les sélections semi-automatiques et manuelles permettent à l'utilisateur de choisir une métaphore et/ou de définir comment un attribut particulier de l'ensemble de données est associé à une propriété de la métaphore, c'est-à-dire comment cet attribut sera représenté.

3.3.1.3 Environnements 3D pour la visualisation de données multimédia

Massari *et al.* [61], Costabile *et al.* [23] présentent Virgilio, un système permettant l'exploration d'ensembles de données structurés à l'aide de métaphores dites *concrètes*, c'est-à-dire composées d'objets et de scènes du quotidien. Le domaine des données et les requêtes sur l'ensemble de données sont considérés comme extérieurs au système. Le résultat des requêtes permet de créer une représentation visuelle composée d'une collection d'environnements virtuels 3D décrits en VRML. Virgilio est composé de trois modules : un module de structuration de l'information, un module d'édition de modèles d'objets et un module de définition de métaphores.

Le module de structuration de l'information permet d'organiser le résultat de la requête en une structure hiérarchique. L'*arbre de structuration* permettant de représenter la structure de l'information à visualiser contient trois types de nœuds : SET_OF permettant de choisir un type d'éléments, RECORD permettant de représenter un élément, et ATTRIBUTE permettant de représenter les attributs de l'élément.

Les modèles d'objets correspondent à des objets paramétriques utilisés lors de la construction de l'environnement virtuels. Ils sont composés d'un nom, d'une description VRML et d'un ensemble de propriétés et attributs. Trois types d'objets peuvent être définis : les *agrégateurs* (comme une table) permettant de regrouper un ensemble quelconque de données, les *classifieurs* (comme un classeur) permettant de regrouper un ensemble ordonné de données et les *accessoires* (comme un cadre photo) correspondant à des supports pour des types spécifiques de données à afficher (texte, images, ...). Les agrégateurs et les classifieurs correspondent à des conteneur sur/dans lesquels sont disposés des accessoires.

Les métaphores correspondent à une mise en correspondance entre les modèles d'objets et les nœuds de l'arbre de structuration. Les nœuds de type RECORD sont associés à des agrégateurs, les nœuds de type SET_OF sont associés à des classifieurs, et les nœuds de type ATTRIBUTE sont associés à des objets de type accessoire.

3.3.1.4 Conclusion

Les métaphores génériques permettent de représenter tous types de données, quel que soit leur domaine, à travers une interface graphique 3D. Elles peuvent être abstraites, et sont alors plus adaptées à la visualisation d'ensembles de données abstraites. Le choix d'environnements virtuels concrets permet à l'utilisateur final de réutiliser ses connaissances du monde réel pour appréhender les don-

nées à visualiser, leur structuration, et faciliter la recherche d'information. Cependant, l'utilisation des connaissances de l'utilisateur final est diminuée de par la définition de métaphores génériques ne prenant pas en compte le domaine des données. D'après Schraefel et Karger [77], il est nécessaire d'offrir à l'utilisateur final une interface graphique dont la métaphore de présentation de l'information est corrélée à la sémantique des données. Dans la section suivante, nous étudions un ensemble de langages et de cas d'études adaptant la métaphore visuelle au domaine des données à visualiser.

3.3.2 Langages pour la définition de métaphores de visualisation de l'information

L'utilisation de métaphores génériques pour la représentation d'un ensemble de données ne permet pas de tirer parti des connaissances et habitudes de l'utilisateur final envers le domaine de l'ensemble de données. Lorsqu'une interface graphique est spécifiquement créée pour la visualisation de données connues a priori, il est possible de définir une métaphore tentant de recréer la signification de l'information telle qu'elle est perçue par l'utilisateur final dans le monde réel. Le but n'est alors pas d'obtenir un réalisme fort au niveau de la visualisation, mais de donner à l'utilisateur final un certain nombre d'indices visuels, sonores et d'interaction ayant un lien fort dans son imaginaire avec le domaine des données à représenter.

3.3.2.1 Feuilles de styles XSLT pour la définition de métaphores 3D

Le langage XSLT (eXtensible Stylesheet Language Transformations [51]), basé sur XML, permet de transformer des documents XML. Une feuille de style XSLT contient ainsi un ensemble de trames (templates en anglais) définissant les instructions de transformation pour chaque nœud de l'arbre XML source. XSLT est fortement utilisé sur le Web pour la transformation de données XML en documents HTML ou XHTML. Quel que soit le format de l'ensemble de données en entrée du système, il est possible d'obtenir ou de traduire le résultat d'une requête en XML. XSLT pourrait alors être la solution pour la définition des métaphores visuelles d'interfaces graphiques 3D. Dans cette section, nous présentons deux exemples permettant de définir une métaphore visuelle en XSLT pour la visualisation de données structurées.

Chimie et visualisation 3D de molécules Polys [69] présente une méthode pour la définition d'interfaces graphiques 3D. Cette méthode est basée sur l'utilisation de feuilles de style XSLT pour la visualisation de données structurées. Les exemples présentés permettent la visualisation de molécules avec trois niveaux d'interaction et de complexité de l'information visualisée. Le premier niveau permet de visualiser la structure fondamentale de la molécule. Le deuxième niveau propose un niveau de manipulation et d'information correspondant aux besoins de l'enseignement de la chimie au lycée. Enfin, le troisième niveau propose des interactions et des informations plus complexes destinées aux experts.

La structure des molécules est accessible grâce au langage CML (Chemical Markup Language [64]). Le positionnement des atomes les uns par rapport aux autres et leur taille sont donc sauvegardés dans le fichier CML, seule la position relative de l'ensemble de la molécule dans la scène 3D pouvant être définie par l'interface graphique. Un prototype est créé par élément chimique et comporte une couleur, le rayon de l'atome et le nom de l'élément. Un autre prototype permet la description des liaisons atomiques. Ainsi, lors de la définition d'une interface graphique, il est possible de définir la géométrie associée aux atomes (sphères, cubes, . . .) et aux liaisons tout en conservant certains attributs (comme la couleur, la taille et le nom des atomes).

L'interface graphique finale est sauvegardée au format X3D. La transformation de la représentation en CML de la molécule vers une visualisation 3D interactive est effectuée à l'aide d'une feuille de style XSLT. Cette dernière contient donc un certain nombre de fonctions XSLT pour l'accès aux données CML mêlées à un ensemble d'instructions X3D pour la construction de la visualisation (accès aux prototypes d'atomes et de liaisons définis dans un fichier X3D externe, balises de transformations dans l'espace 3D, etc.). Chaque visualisation proposée nécessite la définition d'une feuille de style, aucun mécanisme n'étant proposé pour faciliter la réutilisation d'une feuille de style existante (par exemple,

réutilisation de la feuille de style basique avec ajout d'informations pour obtenir la deuxième visualisation).

La représentation de molécules chimiques est fortement codifiée. Les formats tels que CML permettent de sauvegarder et d'échanger les informations des molécules telles que leur composition (atomes et leurs liaisons) et le positionnement de chaque atome dans l'espace 3D. Il ne s'agit donc pas ici de proposer une métaphore de visualisation permettant une mise en correspondance entre la topologie des données (les atomes) et un ensemble de positions dans l'espace 3D, mais uniquement de personnaliser la représentation des éléments (sphères ou cubes pour les atomes, transparence, ...) tout en conservant les codes généralement employés pour la représentation visuelle de molécules.

La ville comme métaphore pour la représentation de cursus universitaires Sommaruga et Catenazzi [80] présentent une interface graphique 3D définie en XSLT pour la visualisation des cursus d'études proposés par une université. Contrairement au problème de la visualisation de molécules étudié par Polys [69], les éléments de l'ensemble de données ne comportent aucune information intrinsèque de structuration ou de positionnement relatif dans l'espace 3D. Il n'existe pas non plus de métaphore visuelle évidente pour la visualisation de cursus universitaires. L'interface graphique doit donc entièrement définir la structuration et la métaphore visuelle et interactive des données.

La base de donnée en entrée du système proposé est une base de données relationnelle Oracle contenant l'ensemble des données de l'université. Une requête prédéfinie sur la base de données permet d'extraire l'ensemble des départements de l'université, les cursus proposés par chaque département et les modules disponibles pour chaque cursus. Le résultat est sauvegardé dans un fichier XML, un certain nombre d'attributs étant associés à divers éléments (comme par exemple la durée des modules). Les auteurs proposent ici une solution spécifique à la visualisation des cursus d'étude d'une unique université. D'autres universités ou d'autres cas d'études ne sont pas pris en compte lors de la conception du système. Pour chaque nouvel usage, il est donc nécessaire de définir un ensemble de balises XML permettant la traduction d'une requête en un arbre XML pouvant être modifié à l'aide d'une feuille de style XSLT.

Sommaruga et Catenazzi choisissent une simplification de la métaphore de la ville pour visualiser les cursus universitaires. Les départements sont représentés sous forme de districts, les cursus correspondent à des blocs et les modules à des bâtiments. La hauteur des bâtiments correspond au nombre de crédits acquis en cas de réussite au module et leur largeur correspond à la durée du module. Cette feuille de style n'est pas réutilisable telle quelle pour un autre ensemble de données car le format XML choisi pour l'extraction des données n'est pas générique.

Les feuilles de style XSLT permettent la définition de métaphores visuelles pour la création d'interfaces graphiques 3D. Cependant, XSLT n'est pas un format de transformation spécifique à la 3D. Chaque feuille de style ne permet donc d'obtenir qu'un unique format 3D en sortie, l'ensemble des opérateurs de ce format étant intégrés à la feuille de style, ce qui alourdit fortement le code (voir programme 3.1). De plus, les accès au fichier XML d'entrée étant également intégrés à la feuille de style, un formalisme de création d'interfaces graphiques 3D basé sur XSLT serait plus flexible s'il s'appuyait sur un format générique de représentation des résultats de requêtes.

3.3.2.2 X-VRML

X-VRML (Walczak et Cellary [90]) est un langage de haut niveau basé sur XML et permettant de produire dynamiquement des scènes virtuelles à partir de modèles de scènes. Contrairement à XSLT, X-VRML a été pensé entièrement pour créer des interfaces graphiques 3D. C'est un langage orienté objet permettant entre autre la création de classes, prenant en compte les notions d'héritages et de classes abstraites, pour faciliter la création et la réutilisation de modèles de scènes, ou métaphores visuelles et d'interaction. La paramétrisation des métaphores est assurée par un ensemble d'opérateurs de conditions, de structures de contrôles, et la possibilités de définir des variables. Les données de la scène virtuelle peuvent être extraites de plusieurs ensembles de données à partir de requêtes SQL. Originellement prévu pour la création de scènes virtuelles en VRML, X-VRML reste indépendant du langage de description de la scène virtuelle finale. Il est donc possible de produire, à partir d'un fichier

Programme 3.1 Extrait de code XSLT pour la création d'une interface graphique en X3D. Les éléments violets correspondent aux accès au fichier XML, les éléments verts aux opérateurs et éléments X3D.

```
<!-- distribution linéaire en X des enfants ELEM de l'élément courant -->
<xsl:template name="DistributionLineaire">
  <xsl:for-each select="./ELEM">
    <xsl:element name="Transform">
      <xsl:attribute name="translation">
        <xsl:value-of select="count(..ELEM) * 10"/>
        <xsl:text> 0 0</xsl:text>
      </xsl:attribute>
      <... éléments à distribuer .../>
    </xsl:element>
  </xsl:for-each>
</xsl:template>
```

X-VRML, une scène décrite à l'aide de VRML, mais également une scène décrite en X3D ou MPEG4 par exemple.

Art et outil de création de musées virtuels Walczak *et al.* [91] proposent ARCO (Augmented Representation of Cultural Objects), un système basé sur X-VRML permettant de créer des expositions virtuelles proposant aux visiteurs une interface graphique 3D intuitive et basée sur les métaphores des musées. ARCO permet la création de musées virtuels en ligne ou en local. Il permet de choisir les objets à visualiser, ainsi que la façon de les visualiser (contenu de la collection, mode de présentation, etc.). ARCO est divisé en trois composants principaux :

- un module de production de contenu. Il permet de créer les représentations numériques des œuvres d'art, appelées *objets culturels* ;
- un module de gestion de contenu. Il permet de gérer l'ensemble des objets culturels dans une base de données ;
- un module de visualisation du contenu. Le concepteur définit des expositions virtuelles en créant des *espaces d'expositions* auxquels il assigne les objets culturels à présenter, ainsi que des modèles de présentation.

Les modèles de présentation, ou métaphores, comportent l'ensemble des paramètres permettant de personnaliser l'aspect visuel de l'interface graphique et les interactions disponibles pour le visiteur. Les espaces d'expositions contiennent un modèle de présentation, une instance d'objet culturel et une règle de sélection des objets culturels permettant d'extraire les objets de la base de données correspondant aux critères définis dans la règle.

ARCO permet la création d'interfaces graphiques 3D pour la visualisation d'objets d'arts sous forme de musées virtuels. L'information est arrangée le long d'un ensemble de salles et de scènes 3D, chacune représentant une étape de la navigation dans l'information multimédia.

Le cas d'études des expositions virtuelles démontre les capacités d'X-VRML pour créer des interfaces graphiques 3D dont les métaphores sont adaptées à l'ontologie des données à visualiser. Entre autre, la création de classes et de modèles paramétriques permet de simplifier la création de scènes virtuelles, et d'assurer la gestion des données de façon externe à la scène 3D. Celle-ci est créée dynamiquement au moment de la visualisation. Cependant, l'accès aux ensembles de données et la structuration des données sont directement intégrés dans le code (voir programme 3.2), ce qui restreint les possibilités de réutilisation des métaphores telles qu'elles pour d'autres applications. De même, les opérateurs et éléments du langage de description de scène choisis sont directement intégrés aux modèles de présentation. En XSLT, l'intégration de ses éléments est effectuée à l'aide de balises spécifiques. X-VRML simplifie donc la création du code purement 3D en intégrant directement les opérateurs spécifiques au langage de scène choisis. Cependant, ce format ne propose pas une abstraction de tels opérateurs. Il n'est donc pas possible d'utiliser un unique modèle de présentation en X-VRML pour produire des scènes virtuelles décrites en différents formats (VRML, X3D, ...). Le modèle de départ

doit être modifié pour chaque format de sortie de façon à ce que tous les opérateurs 3D du modèle correspondent aux opérateurs spécifiques au format de sortie.

Programme 3.2 Extrait de code X-VRML pour la création d'une interface graphique en VRML. Les éléments violets correspondent aux accès à la base de données, les éléments verts aux opérateurs et éléments VRML.

```

<!-- distribution linéaire en X des éléments ELEM -->
<set name="'x'" value="0"/>
<db_connect>
  <db_query names="'id'" sql="'select ID from ELEM'">
    Transform {
      translation <insert value="$x"/> 0 0
      <... éléments à distribuer .../>
    }
    <set name="'x'" value="$x+10"/>
  </db_query>
</db_connect>

```

3.3.2.3 Conclusion

Divers langages permettent la définition de métaphores visuelles adaptées au domaine des données. Qu'ils soient ou non spécifiques à la création d'interfaces graphiques 3D, ils possèdent deux inconvénients principaux : les requêtes utilisateur et/ou la récupération des données résultant de ces requêtes ainsi que les opérateurs spécifiques au format 3D choisi pour la description de l'interface graphique sont entièrement intégrés dans la code de description de la métaphore. Aucune abstraction des résultats de la recherche ou de la description de la métaphore n'est proposée, ce qui réduit considérablement la flexibilité et la réutilisabilité des métaphores ainsi décrites.

3.3.3 Conclusion

L'utilisation de métaphores génériques pour la visualisation de données n'est pas adaptée à toutes les situations. Il est donc nécessaire de proposer un formalisme permettant la description de tous types de métaphores visuelles 3D et non pas un simple catalogue de métaphores de visualisation. Un certain nombre de langages permettant la définition de telles métaphores existent, mais les métaphores ainsi créées sont intrinsèquement liées au format des données en entrée du système d'une part, et au format de description de scène 3D choisi en sortie d'autre part.

3.4 Conclusion

Si les résultats de requêtes peuvent être représentés sous forme de graphe, l'utilisation brute de techniques de visualisation de graphe pour la visualisation de grands ensembles de données n'est généralement pas satisfaisante. Les visualisations existantes sont généralement abstraites et ne tirent pas parti des connaissances de l'utilisateur final dans le domaine des données à visualiser. Il est donc nécessaire de proposer un formalisme permettant d'offrir une liberté totale dans la définition de métaphores de visualisations. Cependant, ce formalisme peut s'inspirer de techniques développées dans le cadre de la visualisation de graphe. C'est le cas plus particulièrement du regroupement de données permettant de construire les différents niveaux d'abstractions nécessaires à l'amélioration de la navigation dans l'ensemble de données. La structuration ainsi créée permet à l'utilisateur final de ne traiter que les données nécessaires, diminuant sa charge cognitive, mais aussi de construire une carte mentale de l'interface graphique 3D. Cette carte mentale pourra alors être mise à profit par l'utilisateur final lors d'une prochaine utilisation de l'interface graphique, ou lors de l'exploration de nouvelles données. Afin d'acquérir une plus grande flexibilité des formats de données compatibles

en entrée du système et une plus grande réutilisabilité des métaphores de structuration définies par les auteurs d'interfaces graphiques 3D, le formalisme présenté nécessite l'intégration d'un langage abstrait représentant la structure des interfaces. Ce formalisme doit permettre de définir la façon dont l'utilisateur final accède aux données, ainsi que l'ensemble des contraintes sémantiques imposées par la métaphore de visualisation.

Au niveau de la présentation de l'information, la métaphore visuelle et d'une interface graphique doit pouvoir être entièrement adaptée au contexte et à l'utilisateur final. En effet, la définition d'un catalogue de métaphores ne suffit pas à proposer une interface graphique efficiente. Outre les fonctionnalités de choix de représentants pour chaque données et de positionnement dans l'espace, la description d'une métaphore visuelle doit pouvoir intégrer des fonctionnalités inspirées de la technologies fish-eye. Par exemple, si une catégorie est représentée par un objet 3D, un double-clic sur cet objet peut permettre de le transformer en un ensemble d'objets représentant les éléments de cette catégorie. Les autres catégories sont toujours affichées au niveau d'abstraction supérieur afin de conserver le contexte de la visualisation. De même, différents moyens de navigation peuvent être proposés pour chaque niveau d'abstraction défini à l'aide de la métaphore de structuration de l'information. Afin de conserver une flexibilité maximale concernant le format 3D de description de l'interface graphique, notre formalisme doit intégrer un langage abstrait de description de scène intégrant les fonctionnalités particulières aux interfaces graphiques 3D.

Chapitre 4

Formalisme pour la création d'interfaces graphiques 3D

4.1 Vue d'ensemble

Notre étude se situe dans le contexte de l'évolution du Web vers le Web 3D, c'est-à-dire un ensemble d'environnements 2D et 3D interactifs et inter-connectés. Le contenu 3D y est intégré soit par l'intermédiaire d'objets 3D insérés dans des pages Web, soit au moyen d'interfaces graphiques 3D. Grâce à l'intégration native de la 3D dans HTML prévue par la norme HTML5 en cours de spécification, le développement de technologies 3D pour le Web connaît un fort regain d'intérêt. Afin de contribuer au développement du Web 3D, nous proposons un formalisme intégrable à des outils auteurs pour la création d'interfaces graphiques 3D. Ce formalisme facilite la définition de métaphores complexes, attractives, efficaces et personnalisées pour la visualisation de grands ensembles de données. Afin d'exploiter au mieux le potentiel des ensembles de données, il doit permettre de formuler de manière simple les requêtes représentant les besoins de l'utilisateur final, ainsi que la représentation des résultats de ces requêtes sous une forme cognitivement efficace (voir chapitre 3). L'exemple 4.1.1 permet d'illustrer les principes généraux de notre formalisme et servira de référence dans la suite du document pour faciliter la compréhension de certains points abordés.

Exemple 4.1.1 *Nous souhaitons visualiser une base de données multimédia contenant des films et un certain nombre d'informations associées (durée, date de sortie, réalisateur, affiche, ...) à l'aide d'une interface graphique 3D. Nous souhaitons naviguer dans l'information en choisissant un genre, puis un réalisateur, et enfin le titre du film. Les trois requêtes nécessaires à la structuration de l'information correspondent donc à l'extraction de :*

- l'ensemble des genres contenus dans l'ensemble de données ;
- pour chaque genre, l'ensemble des réalisateurs ;
- pour chaque réalisateur de chaque genre, l'ensemble des films réalisés (voir figure 4.1(a)).

La métaphore visuelle choisie correspond à une tour de carrousels sur trois niveaux (voir figure 4.1(b)), permettant de mettre en correspondance les éléments issus de chaque requête avec une représentation graphique :

- distribution linéaire de cylindres blancs surmontés du nom de l'élément correspondant pour représenter les différents genres ;
- distribution circulaire de sphères colorées pour représenter les différents réalisateurs ;
- distribution circulaire d'affiches pour représenter les différents films.

L'interface graphique obtenue est présentée en figure 4.1(c).

Le formalisme que nous proposons s'inspire des usages actuels pour le développement d'applications Web dynamiques, facilitant ainsi la transition du Web au Web 3D. Nous pouvons distinguer trois métiers principaux intervenant lors la conception d'une application Web : la gestion de données, la conception de l'interface graphique et l'infographisme. Dans le cadre de notre étude, la modélisation des objets 3D intégrés aux interfaces graphiques obtenues, tout comme les autres données multimédia (images, vidéos, ...), est effectuée à partir d'applications tierces : modeleurs 3D, scanners 3D,

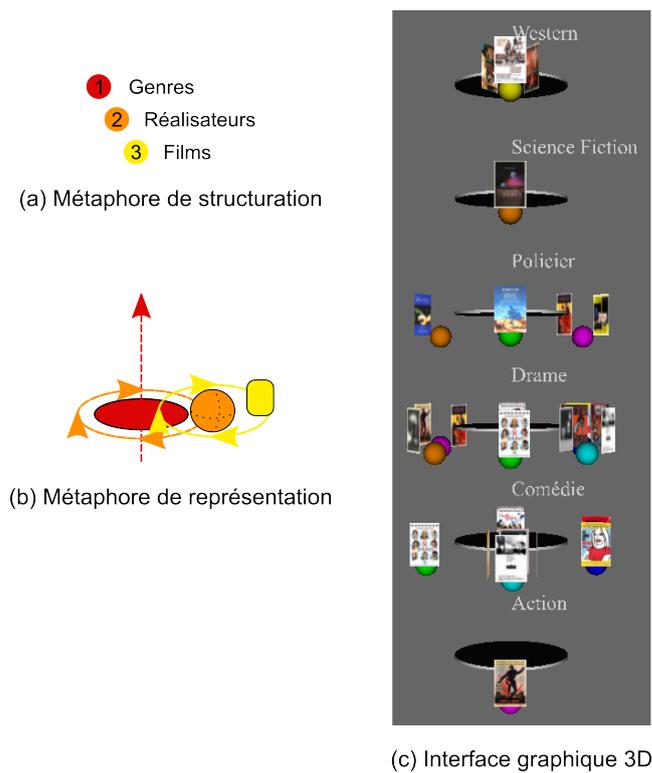


Figure 4.1 – Interface graphique 3D pour la visualisation d'une base de données de films.

etc. Un certain nombre de primitives de base, comme les cubes ou les sphères, peut cependant être proposé en interne de l'application, en complément de l'import d'objets 3D compatibles avec le format choisi pour la description de la scène finale. Notre formalisme ne gère donc pas directement la partie infographie de la création d'interfaces. Cependant, il doit proposer des solutions pour la gestion des données et la conception de métaphores visuelles (voir figure 4.2). Dans la suite de cette section, nous présentons les principales contraintes que notre formalisme doit prendre en compte.

Modularité Nous pouvons observer l'importance de la modularité lors de la conception d'applications Web, comme par exemple la réutilisation partielle de modèles de présentation. Notre formalisme doit donc proposer de décrire la représentation visuelle des résultats sous forme de modules paramétriques (distribution linéaire ou circulaire par exemple) afin de permettre la réutilisation des modules existants pour la création de nouvelles métaphores visuelles. Ainsi, dans l'exemple 4.1.1, la métaphore de la tour de carrousels peut être composée à l'aide d'un module de distribution linéaire combiné à deux modules de distribution circulaire.

Extensibilité Sur le Web, les usages évoluent rapidement. Si l'intégration de contenus 3D est encore faible, elle est en forte progression. Lorsque les interfaces graphiques 3D se seront démocratisées sur le Web, de nouveaux usages risquent d'émerger. Notre formalisme doit donc être extensible pour pouvoir prendre en compte ces nouveaux usages et les nouvelles contraintes qu'ils imposeront aux développeurs d'applications Web 3D.

Généricité des formats d'entrée et de sortie L'information représentée à l'aide de notre formalisme peut être gérée par tous types de systèmes de gestion de données, comme par exemple un système de gestion de bases de données, des cartes topiques, ou encore un ensemble de services Web. De plus, les applications créées ne se limitent pas à un domaine ou à un type d'usage (expositions virtuelles, enseignement, catalogues multimédia, vente en ligne, pages personnelles, ...), mais à l'ensemble des usages exploitant la visualisation de contenus multimédia. Notre formalisme doit donc proposer une

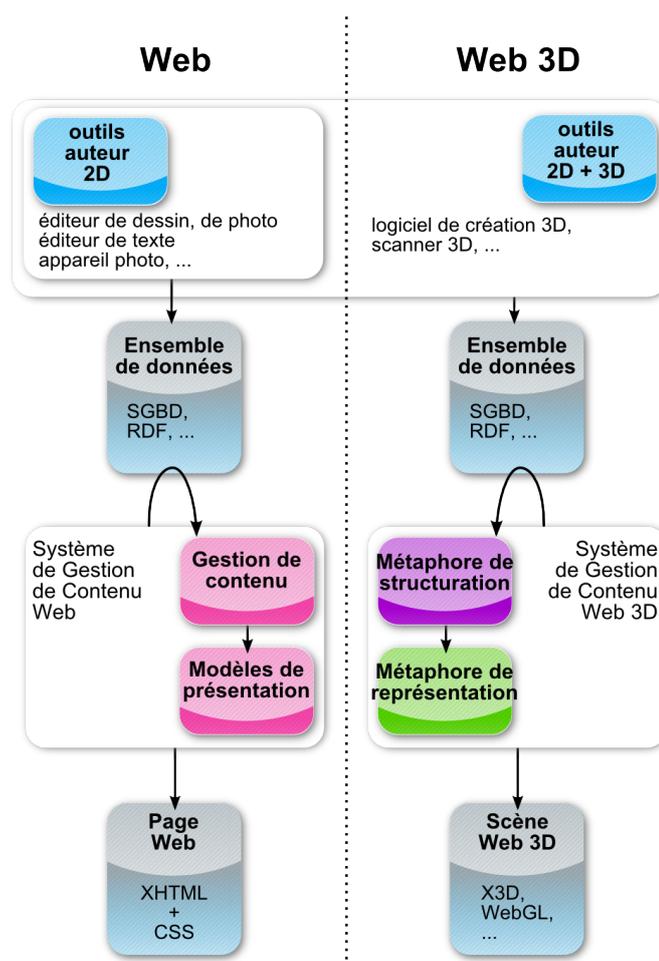


Figure 4.2 – De la modélisation dynamique de sites Web à la modélisation de scènes 3D pour le Web.

gestion générique des données afin d'assurer la flexibilité concernant le type de données en entrée du système, le domaine applicatif concerné, ainsi que le langage utilisé pour formuler les requêtes permettant de représenter les besoins de l'application.

Concernant la présentation des résultats, aucun format 3D ne se distingue clairement pour la création d'interfaces graphiques 3D sur le Web (voir section 2.3). Afin d'assurer la pérennité de notre formalisme, nous devons donc également proposer une solution générique de description de scènes 3D assurant un choix flexible du format de description final de l'interface graphique. Au moment de la production de l'interface graphique, son concepteur pourra alors choisir un format spécifique (comme par exemple X3D ou WebGL) pour la description de la scène produite en fonction des contraintes de son application (complexité de l'interface graphique, possibilité d'utilisation de modules d'extension, ...).

Variété des interfaces graphiques 3D Comme nous l'avons vu en section 3.2, la structuration d'une interface graphique est primordiale pour permettre à l'utilisateur final de naviguer dans l'information de manière efficace. Le formalisme que nous proposons doit donc permettre au gestionnaire de contenu de structurer son application en fonction des usages. La structure finale ne doit pas être limitée aux arbres, mais permettre de représenter tout type de graphe orienté.

Au niveau de la présentation graphique des résultats, nous souhaitons permettre la définition d'interfaces graphiques 3D couvrant toute la gamme d'interfaces imaginables allant des environnements virtuels immersifs à l'examen d'objets élémentaires. De même, les interfaces produites doivent pouvoir être mono ou multi-utilisateurs.

La variété des interfaces graphiques pouvant être définies à l'aide de notre formalisme doit permettre la

prise en compte de la variété des usages du Web. Entre autre, il doit permettre de définir des interfaces graphiques 3D adaptées à différents types de dispositifs d'interaction et de restitution (ordinateur de bureau, téléphone mobile tactile ou non, téléviseur à partir d'un boîtier décodeur, ...).

Le formalisme que nous proposons doit donc être flexible, c'est-à-dire à la fois modulaire, extensible, générique et permettant la définition de métaphores variées. Cette flexibilité doit être présente tout au long du pipeline gérant le flux d'information de l'ensemble de données jusqu'à la présentation graphique. Au niveau de l'architecture générale, la flexibilité est gérée par la division du pipeline de création d'interface graphique 3D en deux modules :

- un module de *structuration de l'information* déterminant la façon dont l'utilisateur découvre et explore l'information ;
- un module de *présentation de l'information* définissant la géométrie de l'interface graphique (entre autre, la manière dont les données à visualiser sont présentées) ainsi que les fonctionnalités d'exploration des données et d'interaction avec le contenu.

L'interface graphique ainsi créée possède alors les caractéristiques suivantes :

- sa *topologie* correspond à la structure extraite de l'ensemble de données en entrée du système ;
- sa *géométrie* et ses *interactions* correspondent à la métaphore de présentation de l'information choisie.

La paramétrabilité de chacun des modules permet de faciliter la production d'interfaces graphiques personnalisables par l'utilisateur final, autant au plan de la structure de l'application (choix d'une structure, modification d'une requête) que de la représentation de l'information (choix d'une couleur par exemple).

Les systèmes de Lindenmayer, ou L-Systèmes, ont fortement inspiré nos travaux. Ils permettent, à partir de peu de données (l'axiome ω), de générer des modèles complexes (à l'aide d'un ensemble P de règles de construction). Les mots produits à l'aide de L-Systèmes peuvent être interprétés géométriquement de différentes manières afin de créer une représentation visuelle du L-Système (voir exemple 4.1.2).

Exemple 4.1.2 Soit le L-Système $L = \{V, \omega, P\}$ composé d'un ensemble fini de symboles V , d'un axiome de départ ω et d'un ensemble P de règles de production, tels que :

- $V = \{A, S\}$
- $\omega = A$
- $P = (A \rightarrow S[A]S[A]A) \wedge (S \rightarrow SS)$

Les symboles $[$ et $]$ représentent respectivement une sauvegarde de la position courante et un retour à la dernière position sauvegardée. La figure 4.3 illustre l'utilisation de différentes métaphores visuelles pour la modélisation géométrique du mot produit par le L-Système L à la troisième itération, c'est-à-dire le mot $M = SS[S[A]S[A]A]SS[S[A]S[A]A]S[A]S[A]A$.

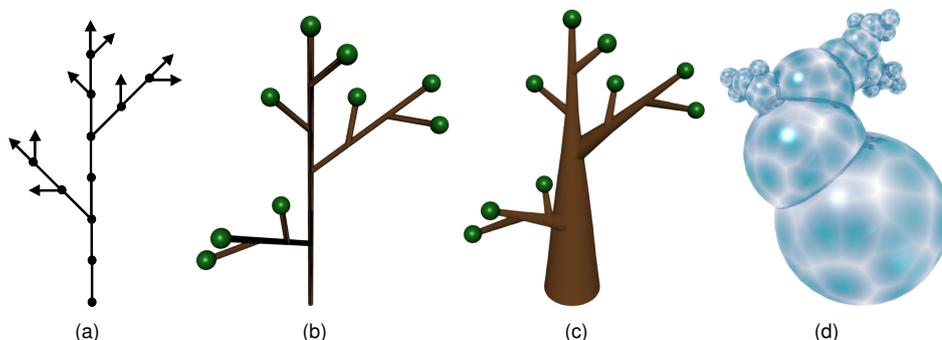


Figure 4.3 – Définition de quatre métaphores visuelles pour modéliser le mot $M = SS[S[A]S[A]A]SS[S[A]S[A]A]S[A]S[A]A$ produit par le L-Système L à la troisième itération.

La première interprétation géométrique (figure 4.3(a)) représente la lettre A par un segment terminé par une flèche, et la lettre S par un segment simple. Les rotations se font autour d'un axe perpendiculaire au plan et la modélisation est en 2D. La deuxième interprétation géométrique (figure 4.3(b)) est similaire, les flèches (lettre A) étant remplacées par un segment brun terminé par une sphère verte et les segments (lettre S) par des segments bruns.

La troisième interprétation géométrique (figure 4.3(c)) représente la lettre A par un cône terminé par une sphère tandis que la lettre S est représentée par un cône tronqué. Les rotations se font autour de l'axe formé par la branche principale, la modélisation étant en 3D. De plus, le diamètre des cônes tronqués est modifié à chaque pas de l'algorithme.

Enfin, la quatrième interprétation géométrique (figure 4.3(d)) représente indifféremment les lettres A et S à l'aide de sphères dont les tailles sont mises à l'échelle à chaque pas de l'algorithme. Les rotations s'effectuent à chaque jonction entre deux éléments, et non uniquement lors des embranchements. La modélisation est en 3D.

Les L-Systèmes étant originellement développés comme outils mathématiques permettant de raisonner sur le développement d'organismes multicellulaires, Smith [79] emprunte les termes *génotype* et *phénotype* au lexique biologique pour représenter respectivement le mot produit par un L-Système et son interprétation géométrique.

La méthode que nous proposons est également procédurale et permet la production d'un modèle plus ou moins complexe à partir d'un grand nombre de données, contrairement aux L-Systèmes qui génèrent des modèles complexes à partir de peu de données. Cependant, nous pouvons observer une similitude entre les L-Systèmes et notre formalisme : tous deux permettent d'appliquer une métaphore visuelle à une structure de données. Dans la suite du document, nous emploieront donc les termes *génom*, *génotype*, *phénom* et *phénotype* définis comme suit pour nommer les différents composants de notre formalisme.

Définition 4.1.1 *Le génome d'une interface graphique 3D est la description de sa métaphore de structuration de l'information. Il correspond à l'ensemble des requêtes permettant d'extraire l'information de l'ensemble de données en entrée du système afin de produire un génotype.*

Définition 4.1.2 *Le génotype d'une interface graphique 3D est la représentation de l'information structurée de cette interface. Il correspond au résultat de l'application d'un génome sur l'ensemble de données en entrée du système.*

Définition 4.1.3 *Le phénomène d'une interface graphique 3D est la description de sa métaphore de présentation de l'information. Il définit la mise en correspondance entre l'information structurée décrite dans le génotype de l'interface et l'espace 3D, ainsi qu'un ensemble de paradigmes d'exploration et d'interaction.*

Définition 4.1.4 *Le phénotype d'une interface graphique 3D est la description de l'ensemble des scènes 3D composant cette interface. Il correspond au résultat de l'application d'un phénomène sur le génotype de l'application.*

Le pipeline correspondant à la création d'une interface graphique 3D pour la visualisation d'un ensemble de données est présenté en figure 4.4. Le module de structuration de l'information permet la définition de requêtes sur l'ensemble de données en entrée du système sous la forme d'un génome. Le générateur de génotype extrait alors les informations de l'ensemble de données conformément au génome et produit la structure de l'application sous la forme d'un génotype. Le module de présentation de l'information permet quant à lui la définition d'une métaphore visuelle et d'interaction sous la forme d'un phénomène. Le générateur de phénotype applique ce phénomène au génotype produit par le module de structuration de l'information afin de produire le phénotype, c'est-à-dire l'ensemble de scènes 3D composant l'interface graphique. Enfin, le phénotype obtenu peut être visualisé par l'utilisateur final à l'aide d'un logiciel tiers, comme par exemple un navigateur Web.

En sections 4.2 et 4.3, nous présentons plus en détails le module de structuration de l'information ainsi que les concepts de génome et de génotype d'une part, et le module de présentation de l'information avec les concepts de phénomène et de phénotype d'autre part.

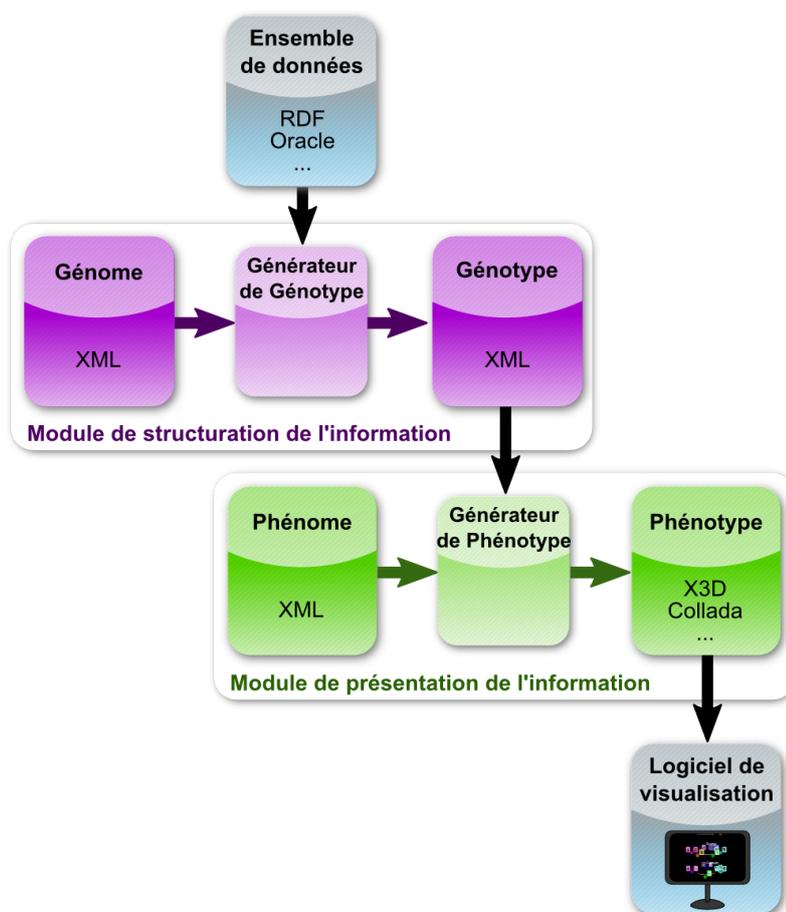


Figure 4.4 – Pipeline de production d'une interface graphique 3D.

4.2 Module de structuration de l'information

Le module de structuration de l'information permet de définir des requêtes sur l'ensemble de données en entrées du système. Ces requêtes permettent d'une part l'extraction de l'information pertinente de l'ensemble de données, et d'autre part la structuration de cette information. En section 4.2.1, nous définissons les concepts généraux du module de structuration de l'information, ainsi que l'ensemble des éléments nécessaires au module pour permettre la création d'une interface graphique 3D. La structuration de l'information n'est pas toujours hiérarchique. Cependant, nous exposons en section 4.2.2 les avantages de la structuration hiérarchique pour la recherche d'information. Puis, en sections 4.2.3 et 4.2.4, nous détaillons les langages formels du génome et du génotype permettant respectivement la description de la structuration de l'interface graphique, c'est-à-dire la façon dont les données sont organisées, et la description des données extraites et structurées. En section 4.2.5, nous présentons deux systèmes permettant la création d'un génotype à partir d'un génome. Enfin, en section 4.2.6, nous concluons sur les caractéristiques du module de structuration de l'information.

4.2.1 Vue d'ensemble

4.2.1.1 Généralités

Comme nous l'avons vu en section 3.2, la structuration d'une interface graphique est primordiale pour faciliter la recherche d'information et fidéliser les utilisateurs finaux. Lors de la navigation dans l'interface, l'utilisateur final crée une carte mentale de cette interface. Lors de sa prochaine visite, cette carte lui permettra de naviguer plus rapidement vers l'information déjà explorée. Une structuration intelligente des données, comme par exemple le regroupement de l'information en niveaux d'abstrac-

tion, permet une meilleure mise à profit de cette carte mentale. En effet, lorsque l'utilisateur parvient à extraire la logique interne de la structure de l'interface graphique, sa carte mentale de l'interface lui permet non seulement de retrouver rapidement une information déjà explorée, mais également d'accéder de manière plus rapide et intuitive aux espaces de l'interface non encore visités. Ainsi, dans l'exemple 4.1.1, l'utilisateur final ayant compris la structure [Genre → Réalisateur → Film] de l'interface lors de sa première visite pourra rapidement trouver l'ensemble des comédies réalisées par Woody Allen. Les regroupements par genre et par réalisateur représentent alors deux niveaux d'abstraction imbriqués.

Dans le cadre de la visualisation d'information issue de grands ensembles de données, la structuration de l'information peut être :

- fixée lors de la création de l'interface graphique ;
- sélectionnée par l'utilisateur final parmi un ensemble de structurations prédéfinies ;
- définie par l'utilisateur final à l'aide d'une requête libre (comme par exemple lors de l'utilisation d'un moteur de recherche) ;
- définie automatiquement par une application (à partir de profils utilisateurs et/ou de recherche de discriminant par exemple).

Par la suite, nous appellerons *gestionnaire de données* la personne ou l'application définissant la structure de l'interface graphique.

La figure 4.5 présente le pipeline du module de structuration de l'information. Les données en entrée de notre système peuvent être gérées par tout type de système de gestion de données, et peuvent, entre autre, provenir de plusieurs sources, chaque source étant gérée par un système de gestion de données différent. Afin d'assurer l'homogénéité de la requête permettant de définir la structure de l'interface graphique, nous proposons la définition d'un langage générique pour la description de la structuration de l'information. Les fichiers décrits à l'aide de ce langage sont appelés génomes. Le générateur de génotype permet de créer un ensemble de requêtes compatibles avec l'ensemble de données en entrée du système afin de structurer l'information qu'il contient conformément au génome. Le fichier correspondant est appelé génotype.

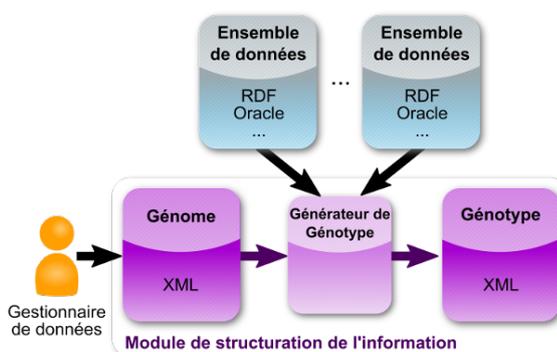


Figure 4.5 – Architecture du module de structuration de l'information.

4.2.1.2 Éléments nécessaires à la structuration des données

Les langages de description de métaphores, tels que XSLT ou X-VRML (voir section 3.3.2), intègrent directement les accès à l'ensemble de données dans la description visuelle de la scène 3D à produire. La séparation entre module de structuration des données et module de présentation des données est donc faible, et le code décrivant la présentation des données peut intégrer des requêtes directes sur l'ensemble de données en entrée du système. Cependant, l'intégration des requêtes dans la définition de la métaphore visuelle permet d'extraire de l'ensemble de données les éléments multi-média nécessaires à la présentation de l'information. Par exemple, selon la métaphore choisie, un film pourra être représenté par son affiche ou par son résumé, chaque métaphore intégrant directement la requête correspondant au type de donnée dont elle a besoin. Dans le cadre de nos travaux, nous

avons opté pour une séparation nette entre structuration de l'information et présentation des données afin d'augmenter la réutilisabilité des différentes métaphores de structuration et de présentation de l'information. Le module de présentation de l'information ne possède donc pas d'accès direct à l'ensemble de données en entrée du système (voir figure 4.4). Le module de structuration de l'information doit donc permettre de transférer au module de présentation de l'information un certain nombre de données nécessaires à la création des scènes 3D de l'interface graphique.

Le génotype produit par l'application d'un génome sur un ensemble de données représente l'information structurée de l'interface graphique. Il définit la topologie des scènes 3D qui seront produites par le module de présentation de l'information. Les éléments visuels et sonores représentant chaque élément du génotype sont définis dans le module de présentation de l'information par l'intermédiaire du phénomène. Ainsi, dans l'exemple 4.1.1, les genres sont représentés par des cylindres blancs. Afin de personnaliser plus précisément la représentation des éléments, le module de présentation de l'information peut exploiter des informations issues de l'ensemble de données de départ. Si nous revenons à l'exemple 4.1.1, nous pouvons ainsi observer que les films sont représentés par leur affiche. Le génotype étant l'interface privilégiée du module de présentation de l'information avec l'ensemble de données, l'affiche de chaque film est représentée (par exemple sous la forme d'une url) dans le génotype. Le module de structuration de l'information doit donc prendre en compte à la fois l'information à présenter, et un certain nombre d'éléments additionnels de présentation associés à cette information. Ces éléments de représentation, que nous nommons *attributs*, peuvent correspondre à des informations textuelles, numériques, ou tout type d'élément multimédia, comme par exemple une couleur, une vidéo ou encore un modèle 3D. Plusieurs attributs peuvent être associés à un unique élément d'information. Ainsi, la figure 4.6 présente le cas d'un élément de type film ayant un attribut image correspondant à l'url de son affiche, et deux attributs vidéo correspondant respectivement à l'url de sa bande annonce et à l'url du film complet. Dans cette exemple, nous pouvons observer que deux attributs de même type (ici, vidéo) peuvent être associés à un même élément. Afin de pouvoir les différencier, un ensemble de labels peut être associé à chaque attribut pour déterminer le domaine de validité de l'attribut.

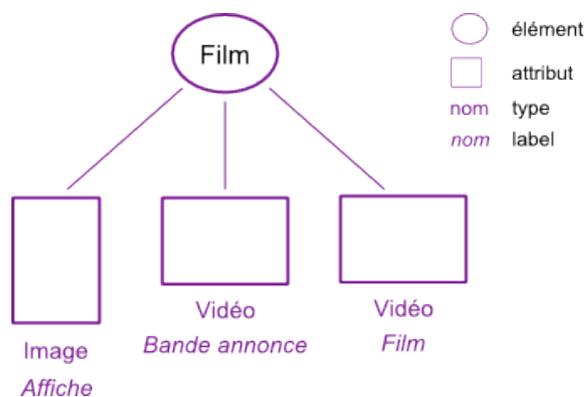


Figure 4.6 – Attributs d'un éléments de type film.

4.2.2 Structuration hiérarchique

Dans le cadre de la présentation de l'information issue de grands ensembles de données, la structuration des données sous forme d'arbre n'est pas suffisante. Notre système doit donc permettre la construction de tout type de graphe orienté. Cependant, lors de l'exploration des données, la représentation de l'information sous une forme hiérarchique offre de nombreux avantages.

Définition 4.2.1 Une hiérarchie est une classification dans laquelle les termes classés sont dans une relation de subordination, chaque terme dépendant du précédent et commandant le suivant.

En mathématique, une hiérarchie est un ensemble de données ordonné, c'est-à-dire un graphe acyclique, ou encore un arbre.

Ahl et Allen [1] proposent un vocabulaire pour la théorie de la hiérarchie. Ils présentent ainsi une hiérarchie comme un ensemble de niveaux imbriqués ou non. Le flux d'information circule librement

à l'intérieur d'un niveau, mais est filtré pour le passage depuis ou vers un autre niveau, les niveaux supérieurs étant considérés comme le contexte des niveaux inférieurs. L'ordre des niveaux peut varier en fonction du domaine applicatif, du contexte ou du besoin. L'observation des données permet alors de créer une hiérarchie spécifiant la façon dont l'utilisateur final accède à la connaissance.

D'après Nicolis [65], malgré la variété des systèmes complexes connus, seuls les systèmes hiérarchiques sont stables. Dans la nature, la finalité de l'évolution de tout système complexe serait ainsi de prendre la forme d'une structure hiérarchique. Kulish [55] insiste sur l'universalité des structures hiérarchiques, que ce soit au niveau biologique, au niveau de l'organisation dans la société, au niveau du savoir ou au niveau des objets et de la technologie. Il propose cinq grands principes de la hiérarchie :

- le *principe général de la hiérarchie* : tout dans l'Univers possède une structure hiérarchique ;
- le *principe de la compression de l'information* : chaque niveau hiérarchique supérieur est toujours plus simple que le niveau précédent ;
- le *principe de la ressemblance hiérarchique* : chaque niveau hiérarchique, dans ses propriétés générales, représente le système dans son ensemble ;
- l'analogie hiérarchique du second principe de la thermodynamique : chaque niveau hiérarchique supérieur a une entropie de l'information inférieure au niveau précédent ;
- l'analogie hiérarchique du troisième principe de la thermodynamique : le niveau le plus haut d'un système hiérarchique fermé est caractérisé par une entropie de l'information volatile.

La fractalité, telle que présentée par Salingaros [75] (voir section 3.2.2), reprend ce principe de hiérarchie et insiste sur son importance dans le cadre de la navigation en environnement urbain. Dans la vie courante, nous organisons en effet naturellement l'information de manière hiérarchique, au moins d'un point de vue local. Ainsi par exemple, nous choisissons nos déplacements en parcourant un arbre mental correspondant aux villes, aux quartiers et aux bâtiments. De même, l'organisation des documents dans la majorité des systèmes d'exploitation est proposée sous la forme d'une hiérarchie de dossiers. Malgré la représentation hiérarchique des données, la structure explorable peut ne pas être hiérarchique. Par exemple, dans un système de gestion de fichiers, il est possible d'insérer dans un dossier des raccourcis vers des dossiers externes. Si l'utilisation de ces raccourcis permet de conserver la sensation d'une navigation hiérarchique dans les dossiers, le chemin réel emprunté pour accéder à l'information ne suit pas la hiérarchie initiale proposée par le système de gestion de fichiers.

D'un point de vue local, nous pouvons donc considérer que l'information peut être structurée de manière hiérarchique. La duplication de l'information dans l'arbre permet de simuler l'existence de cycles et assure la possibilité de proposer une structuration non hiérarchique des données. Cependant, la duplication pose un certain nombre de problèmes, comme par exemple, la taille exponentielle de l'arbre. Afin de pallier à ce problème et de permettre une structuration de l'information sous forme de graphe orienté, mais conservant une hiérarchie au niveau local, deux solutions sont possibles :

- l'insertion de liens dynamiques et l'instanciation d'éléments ;
- la production dynamique d'un sous-ensemble de la structure de l'information en fonction des actions de l'utilisateur.

L'exemple 4.2.1 étend l'exemple 4.1.1 à la présentation de données globalement non hiérarchiques.

Exemple 4.2.1 *Dans l'exemple 4.1.1, l'information est structurée hiérarchiquement en trois niveaux : [Genre->Réalisateur->Film]. Nous souhaitons à présent permettre à l'utilisateur ayant choisi un film d'accéder à l'ensemble des genres auxquels il appartient (voir figure 4.7). Ainsi, sur la figure 4.7(a), le film Manhattan appartient aux genres "Comédie" et "Drame". Nous pouvons donc compléter la structure en ajoutant pour chaque film les genres auxquels il appartient. À partir de ces genres, nous pouvons de nouveau obtenir une liste de réalisateurs, puis une liste de films pour chaque réalisateur. Nous obtenons la structuration suivante : [Genre(1)->Réalisateur(2)->Film(3)->Genre(4)->Réalisateur(5)->Film(6)]. Si nous nous contentons de cette structuration de l'information, l'utilisateur final ne pourra pas accéder aux genres correspondant à un film choisi au niveau 6. Pour assurer l'accès aux genres auquel un film appartient, quel que soit le niveau de ce film, la boucle [Genre->Réalisateur->Film->Genre] doit donc être répétée à l'infini (voir figure 4.7(b)). Pour pallier à ce problème, deux solutions sont offertes :*

- l'instanciation des éléments préalablement définis à l'aide d'opérateurs *DEF* et *USE* ;
- la production dynamique des génotypes.

Dans cet exemple, nous optons pour la production dynamique de génotypes.

La boucle ajoutée permet de relier les éléments de niveau 3 (les films) aux éléments de niveau 1 (les genres). D'un point de vue local, le génotype peut donc uniquement contenir l'information correspon-

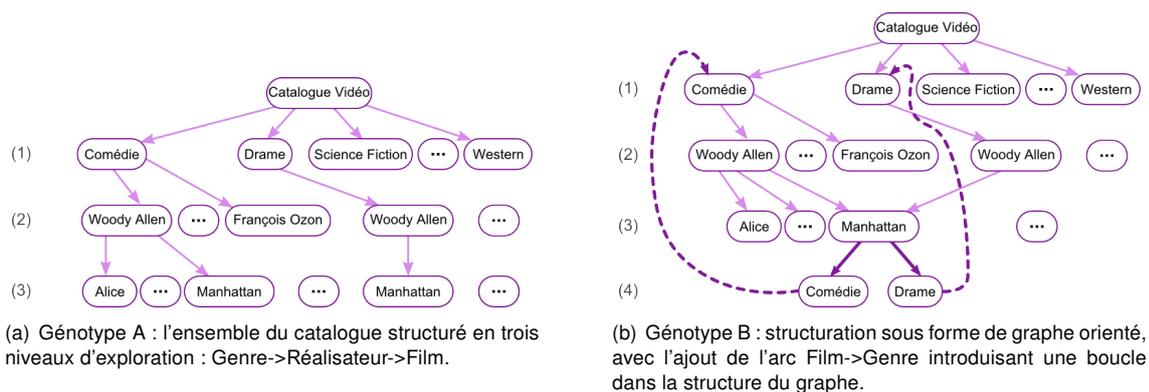


Figure 4.7 – Perte de la caractéristique hiérarchique de la structuration des données par l'ajout de niveaux d'exploration.



Figure 4.8 – production dynamique de génotypes au fur et à mesure de l'exploration des données.

dant au trois premiers niveaux d'abstraction (voir figure 4.8(a)). Lorsqu'au cours de son exploration, un utilisateur final accède à un film, il peut alors effectuer une interaction (à l'aide d'une icône par exemple) lui permettant de produire le génotype correspondant aux niveaux 4, 5 et 6 (voir figure 4.8(b)). L'utilisateur peut alors naviguer librement dans l'information de façon récursive, tout en conservant l'illusion d'une exploration hiérarchique induite par la structure hiérarchique de chaque génotype produit.

Le génotype peut donc être considéré comme un arbre d'exploration, c'est à dire un arbre définissant la façon dont l'utilisateur accède aux données. Chaque niveau de l'arbre correspond à un niveau d'abstraction et est appelé *niveau d'exploration*. Comme nous l'avons vu, le génotype contient, outre les éléments d'information, un certain nombre d'attributs labellisés associés à ces éléments. Le génotype correspond donc à un *arbre d'exploration augmenté* (voir figure 4.9). Une fois produit par le module de structuration de l'information, le génotype est transmis au module de présentation de l'information qui génère l'interface graphique 3D finale (figure 4.4).

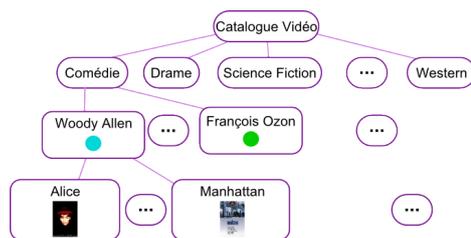


Figure 4.9 – Génotype correspondant à l'exemple 4.1.1.

Dans la section suivante, nous détaillons plus particulièrement le génotype et proposons un langage formel de description de génotype. Puis, en section 4.2.4, nous présentons le langage formel de description de génome permettant de décrire la structuration désirée pour le génotype.

4.2.3 Langage formel de l'arbre d'exploration : le génotype

L'arbre d'exploration, ou génotype, est produit par le module de structuration de l'information puis transféré au module de présentation de l'information. Il correspond à l'interface privilégiée du module de présentation de l'information avec l'ensemble de données en entrée du système. Il doit donc contenir toutes les informations (issues de l'ensemble de données) nécessaires à la construction de l'interface graphique 3D. Pour se faire, le génotype contient quatre types d'informations :

- les *niveaux d'exploration* correspondent aux niveaux d'abstraction issus de la structuration des données et regroupent un ensemble d'éléments d'information de même catégorie. La structure du génotype étant hiérarchique, les niveaux d'exploration sont imbriqués les uns dans les autres. Le premier niveau d'exploration correspondant au niveau d'abstraction le plus élevé ;
- les *éléments* d'information correspondent aux données à représenter. Ils sont issus de la base de données et sont organisés hiérarchiquement à l'intérieur des niveaux d'exploration ;
- les *attributs* correspondent à des informations permettant la représentation de l'élément auquel ils sont associés, soit en positionnant directement l'attribut dans l'espace (si il s'agit d'un modèle 3D par exemple), soit en attribuant sa valeur (couleur, chaîne de caractère, ...) à un paramètre d'un objet de l'interface (comme la texture d'un cube par exemple). Chaque attribut possède un type (par exemple `3Dmodel`, `Color` ou encore `Video`) et un ensemble de données complémentaires dépendant de ce type. Ainsi, un attribut de type `Video` possède également une url pointant vers la vidéo correspondante, tandis qu'un attribut de type `Color` possède trois valeurs correspondant à ces composantes rouge, verte et bleue ;
- les *labels* permettent d'ajouter des métadonnées à l'élément du génotype auquel ils sont associés, et de définir le domaine de validité de l'information représentée par l'élément parent. Lorsqu'ils sont associés à un attribut, ils peuvent permettre au module de présentation de l'information de choisir l'attribut le plus adapté parmi plusieurs attributs de même type associés à un élément d'information (voir figure 4.6).

Souhaitant conserver une flexibilité maximale quant au format de l'ensemble de données d'entrée et au domaine applicatif, nous avons choisi un vocabulaire générique pour la grammaire formelle permettant de construire des génotypes (voir définition 4.2.2).

Définition 4.2.2 *La grammaire formelle permettant de produire un Génotype est composée d'un ensemble fini de symboles V , d'un axiome de départ ω et d'un ensemble P de règles de production, tels que :*

- $V = \{Genotype, Explo, Elem, Att, Tag\}$ où
Genotype est la racine de l'arbre d'exploration
Explo est un niveau d'exploration
Elem est un élément d'information de l'ensemble de donnée à représenter
Att est un attribut typé destiné à être utilisé par le module de présentation de l'information
Tag est un label sémantique
- $\omega = Genotype$
- P est composé des règles de production suivantes :
 $(Genotype \rightarrow Explo + Att^* + Tag^*),$
 $(Explo \rightarrow Elem^+ + Att^* + Tag^*),$
 $(Elem \rightarrow Att^* + Tag^* + Explo^{0|1}),$
 $(Att \rightarrow Tag^*)$

Le génotype contient donc un ensemble de niveaux d'explorations imbriqués les uns dans les autres. Chaque niveau d'exploration comprend un ou plusieurs éléments, chaque élément contenant à son tour un ou plusieurs attributs éventuellement labellisés et l'éventuel niveau d'exploration suivant. Au niveau de l'implémentation, le génotype est sauvegardé sous forme d'un fichier XML conforme au schéma XML correspondant à la grammaire formelle présentée dans la définition 4.2.2. Un extrait du génotype produit pour l'exemple 4.1.1 est présenté dans le programme 4.1.

Programme 4.1 Extrait du génotype correspondant à l'exemple 4.1.1. Les éléments extraits de l'ensemble de données sont indiqués en violet.

```

<Genotype name="CatalogueVideo">
  <Explo name="Genres">
    <Elem name="Comedy">
      <Att type="text" string="Comédie">
        <Tag>nom</Tag>
        <Tag>fr</Tag>
      </Att>
      <Explo name="Realisateurs">
        <Elem name="WAllen">
          <Att type="color" r="0" g="1" b="1"/>
          ...
        </Elem>
        ...
        <Elem name="FOzon">
          ...
        </Elem>
      </Explo>
    </Elem>
    ...
  </Explo>
</Genotype>

```

Le génotype est produit par le module de structuration de l'information par l'application d'un génome sur un ensemble de données. Pour un génome donné, plusieurs génotypes peuvent être produits en changeant l'ensemble de données en entrée du système, ou pour chaque modification de cet ensemble de données. De même, lorsque différents génomes sont appliqués à un unique ensemble de données, un génotype éventuellement différent des autres est produit pour chaque génome. Dans la section suivante, nous présentons le langage formel du génome permettant de décrire la structure d'un génotype.

4.2.4 Langage formel de la structure exploratoire : le génome

Le génotype est une représentation de l'information structurée présentée à travers l'interface graphique 3D. Sa structure est décrite dans un fichier nommé génome. Ce dernier permet la définition des requêtes qui seront envoyées à l'ensemble de données pour la production du génotype. Comme nous l'exposons en section 4.2.2, la structure locale de l'information peut être considérée comme arborescente. Le génome décrit l'imbrication des niveaux d'exploration, ainsi que les attributs et labels éventuels associés aux éléments de chaque niveau d'exploration. Cette description de la structure exploratoire de l'interface 3D correspond aux invariants du génotype. Le génome correspond donc à la représentation du squelette du génotype, et les informations qu'il contient sont de même types que les informations contenues dans le génotype, avec les différences suivantes :

- les *niveaux d'exploration* ne contiennent qu'un seul et unique élément d'information, dit élément générique ;
- les éléments d'information sont des *éléments génériques*, chaque élément représentant l'ensemble des éléments pouvant appartenir au niveau d'exploration auquel il est attaché ;
- les attributs sont des *attributs génériques typés*. Leur type est donc défini dans le génome, mais les données associées (url, composantes rouges vertes et bleues, ...) peuvent être soit fixées par le génome, soit attribuées dynamiquement à partir de l'ensemble de données lors de la création du génotype. Lorsqu'elles ne sont pas fixées, les données associées peuvent avoir une valeur par défaut, assurant ainsi la validité des attributs correspondants produits dans le génotype.

La grammaire formelle correspondante est décrite dans la définition 4.2.3. Tout comme le génotype, le génome est implémenté à l'aide du langage XML, un schéma XML ayant été défini pour assurer la conformité des génomes créés.

Définition 4.2.3 La grammaire formelle permettant de produire un génome est composée d'un ensemble fini de symboles V , d'un axiome de départ ω et d'un ensemble P de règles de production, tels que :

- $V = \{Genome, Explo, Elem, Att, Tag\}$ où
 - $Genome$ est la racine de la structure exploratoire
 - $Explo$ est un niveau d'exploration
 - $Elem$ est un élément d'information générique de l'ensemble de donnée à représenter
 - Att est un attribut générique typé
 - Tag est un label
- $\omega = Genome$
- P est composé des règles de production suivantes :
 - $(Genome \rightarrow Explo + Att(type)^* + Tag^*)$,
 - $(Explo \rightarrow Elem + Att(type)^* + Tag^*)$,
 - $(Elem \rightarrow Att(type)^* + Tag^* + Explo^{0|1})$,
 - $(Att(type) \rightarrow Tag^*)$

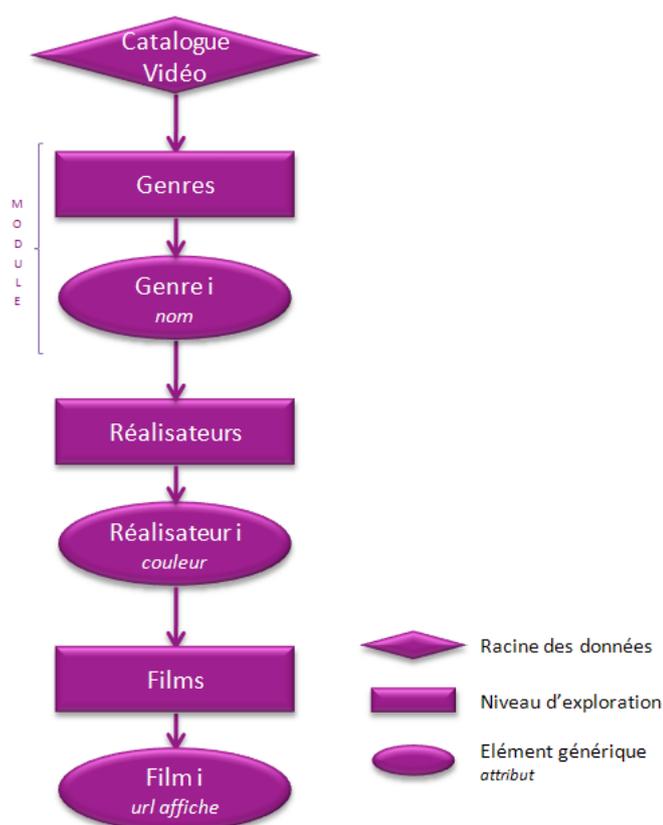


Figure 4.10 – Diagramme du génome correspondant à l'exemple 4.1.1

Un génome correspond donc à une imbrication de niveaux d'explorations. Chaque niveau d'exploration contient un unique élément générique, lui-même contenant un ou plusieurs attributs génériques dont le type est précisé. Les attributs peuvent être labellisés. Ainsi, dans l'exemple 4.1.1, il y a trois niveaux d'explorations : les genres, les réalisateurs, puis les films. Un attribut de type texte et possédant un label "nom" et un label "fr" (précisant que le nom de l'élément est indiqué en français) sont associés à chaque élément de type genre (premier niveau d'exploration), un attribut de type couleur

possédant des valeurs par défaut est associé à chaque élément de type réalisateur (deuxième niveau d'exploration). La décomposition de la requête en niveaux d'exploration assure la modularité du génome. Chaque module ainsi défini peut être réutilisé et combiné avec d'autres modules afin de définir de nouveaux génomes. Le diagramme du génome correspondant à l'exemple 4.1.1 est présenté en figure 4.10 et un extrait du code correspondant est présenté dans le programme 4.2.

Programme 4.2 Extrait du génome correspondant à l'exemple 4.1.1. Les variables sont indiquées en violet.

```
<Genome name="CatalogueVideo">
  <Explo name="nom1">
    <Elem name="nom2">
      <Att type="text" string="texte1">
        <Tag>nom</Tag>
        <Tag>fr</Tag>
      </Att>
      <Explo name="nom3">
        <Elem name="nom4">
          <Att type="color" r="c1" g="c2" b="c3"
            r_default="0.5" g_default="0.5" b_default="0.5" />
          ...
        </Elem>
      </Explo>
    </Elem>
  </Explo>
</Genome>
```

Dans la section suivante, nous décrivons comment produire un arbre d'exploration augmenté, ou génotype, à partir d'un ensemble de données et d'un génome.

4.2.5 De la requête à l'arbre d'exploration : le générateur de génotype

Le composant de génération de génotype permet d'appliquer un génome à un ensemble de données afin de produire un génotype. La transformation permettant d'obtenir un génotype à partir d'un génome est la suivante :

- l'élément racine *Genome* est transformé en un élément *Genotype* ;
- les *niveaux d'explorations* et les *labels* sont conservés tels quels, et dupliqués au besoin ;
- les *éléments génériques* sont dupliqués pour chaque élément de l'ensemble de donnée correspondant à leur niveau d'exploration respectif ;
- les *attributs* sont initialisés à l'aide d'informations présentes dans l'ensemble de données et des valeurs par défauts de manière à respecter leur typage.

Si le génome décrit la structuration des données de l'interface graphique, il est écrit dans un format générique et ne contient pas les requêtes exactes permettant d'extraire l'information correspondante dans l'ensemble de données. Le composant de génération de génotype (voir figure 4.11) permet donc de transformer le génome en un ensemble de requêtes compatibles avec l'ensemble de données en entrée du système et permettant de produire un génotype. Ces requêtes sont composées de critères de recherche hiérarchiques permettant de décrire les niveaux d'exploration et de sélectionner les éléments qu'ils contiennent, ainsi que de critères de représentation permettant de sélectionner les attributs qui seront associés à ces éléments.

Dans la section 2.4, nous présentons un grand nombre de systèmes de gestion d'ensembles de données, chacun possédant son propre langage de requête. Afin d'être compatible avec notre technologie, il est donc nécessaire d'implémenter un composant, nommé *traducteur*, qui produit un génotype conforme à la définition 4.2.2 à partir des requêtes spécifiques au système de gestion de données. Le traducteur permet d'imbriquer un certain nombre d'éléments issus du génome (comme par exemple les opérateurs *Explo*, *Elem*, ...) avec des fonctions issues du langage de requête associé à l'ensemble

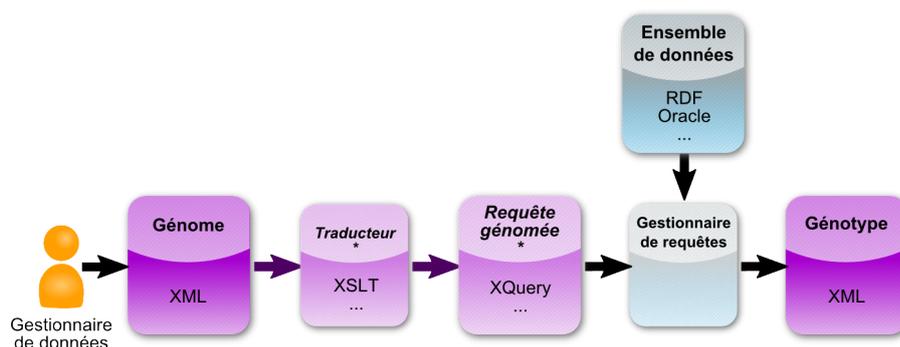


Figure 4.11 – Pipeline de production d'un génotype.

de donnée. Par exemple, si l'ensemble de donnée est sauvegardé en XML, la requête pourra être définie sous forme d'un fichier XQuery mêlant les balises du génome à des fonctions XQuery (voir le programme 4.3 donné en exemple).

Programme 4.3 Extrait de requête XQuery pour la création d'un génotype. Les éléments indiqués en violet sont directement extraits du génome.

```

<Genotype name="CatalogueVideo">
{
  (: base de donnée en entrée du système :)
  let $vod := doc($inputDocument)/Catalogue
  ...
  <Explo name="Realisateurs">
  {
    (: niveau d'exploration 2 : extraction des realisateurs :)
    for $r in $vod/Film/Realisateur/Nom
    order by $r
    return
      <Elem name="{ $r }">
        <Att type="color" r="{ $r/../../Coul/@r } g="{ $r/../../Coul/@g } b="{ $r/../../Coul/@b }"/>
        <Explo name="Films">
          ...
        </Explo>
      </Elem>
    }
  </Explo>
}
</Genotype>
  
```

La requête obtenue est appelée requête génomée et correspond à un génome (représentant la structure exploratoire de l'interface graphique 3D) augmenté de fonctions d'extraction définies dans le langage de requête de l'ensemble de données. Ainsi, tous les éléments invariants du génotype sont décrits dans la partie de la requête extraite du génome, tandis que les éléments dépendant de l'ensemble de données sont décrits à l'aide des fonctions du langage de requête.

Dans certains cas, la structuration de l'information peut être définie directement à partir d'une application de gestion de l'ensemble de données, comme par exemple lors de l'utilisation du protocole Google WebSearch permettant de récupérer le résultat d'une recherche effectuée à l'aide du moteur de recherche de Google sous forme d'un fichier XML. La requête correspondante est alors directement définie dans le langage de requête associé à l'ensemble de données (dans le cas du protocole Google

WebSearch, une requête HTML) et ne possède aucun lien avec un génome (voir figure 4.12). Elle contient toutes les informations permettant de sélectionner et de hiérarchiser les données nécessaires à la création du génotype (éléments à représenter, attributs, ...), mais le résultat obtenu est sauvegardé dans un format spécifique à l'application (par exemple, le format spécifique de WebSearch). Dans ce cas, le résultat obtenu peut être transformé en un génotype par un *traducteur* associé à l'ensemble de données. Par exemple, si le résultat de la requête est exporté en XML, il peut être traduit à l'aide d'une feuille de style XSLT.

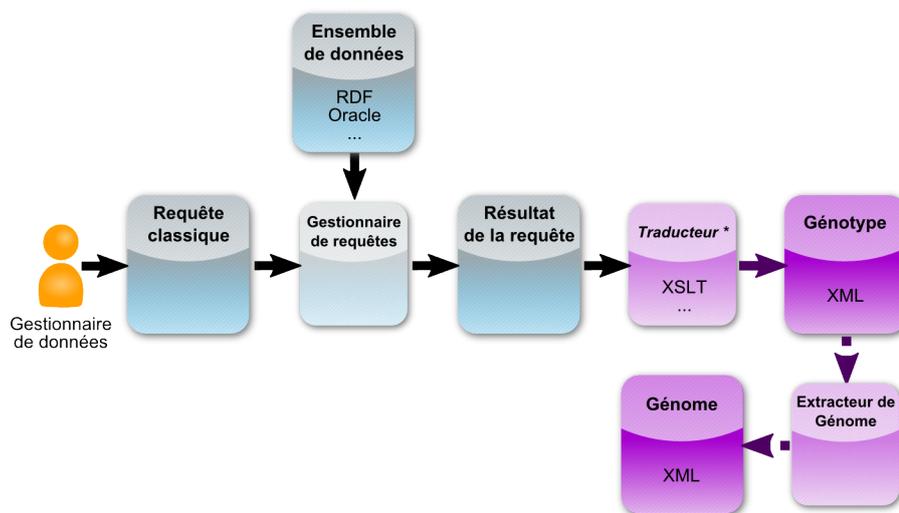


Figure 4.12 – Pipeline de production d'un génotype à partir du résultat d'une requête externe au formalisme.

Lorsque le génotype n'est pas créé à partir d'un génome, la réutilisation de la structuration des données sur un autre ensemble de données n'est pas directement possible. Dans ce cas, le génome peut être extrait du génotype à l'aide d'une feuille de style spécifique. Ce génome pourra alors être appliqué à un ensemble de données différent pour lequel un traducteur de génome en requête génotypée est disponible.

Deux solutions permettent donc la création d'un génotype à partir d'un ensemble de données :

- l'intégration du génome dans la requête initiale permettant de produire automatiquement un génotype ;
- la définition d'un traducteur permettant de traduire le résultat d'une requête classique en un génotype.

Dans le premier cas, l'utilisation d'un génome pour définir la structuration des données assure la généralité du module de structuration. En effet, ce génome peut être réutilisé sur un ensemble de données différent, que ce soit du point de vue du système de gestion choisi ou du domaine applicatif des données. Dans le deuxième cas, l'extraction automatique du génome à partir du génotype permet de conserver les avantages de la généralité du génome malgré l'utilisation d'un système de gestion de requêtes non directement basé sur notre architecture.

Nous avons implémenté la première solution pour produire directement un génotype à partir du catalogue de vidéos à la demande d'Orange. La requête génotypée est créée en XQuery et intègre directement les balises du génome correspondantes. Cette requête XQuery est ensuite appliquée au fichier XML représentant la base de données de vidéos à la demande et permet la production automatique de génotypes à partir de génomes.

Nous avons également implémenté la deuxième solution pour des bases de données créées à partir d'OpenOffice Calc. Une fois les données triées à l'aide de l'interface graphique d'OpenOffice, nous avons effectué un export XML de la feuille de calcul auquel une feuille de style XSLT est appliquée pour produire le génotype correspondant. Dans ce cas, aucun génome n'est préalablement défini.

Le système que nous proposons est compatible avec tout type de système de gestion de données. Cependant, l'intégration de métadonnées sémantiques offre de nombreux avantages. Ainsi, par exemple, la sélection des attributs peut être améliorée par l'ajout de contraintes sémantiques. Par exemple, la bande annonce choisie pourra être disponible en plusieurs langues. De même, selon l'utilisateur final, le représentant d'un élément d'information peut différer (complexité d'un article sur un sujet en fonction de l'âge et des connaissances de l'utilisateur final, couleurs différentes selon les cultures, ...). La sélection automatique des niveaux d'exploration à partir de l'extraction des éléments discriminants d'un ensemble de données est également améliorée par l'utilisation de règles sémantiques.

4.2.6 Conclusion

La structuration de l'information permet d'améliorer l'exploitation de l'information. Si cette structuration peut être quelconque, nous pouvons observer que la définition d'une structure localement hiérarchique augmente l'amélioration de l'exploration de l'information. Nous proposons donc un système permettant de définir cette structuration de l'information sous la forme d'arbres d'exploration. Afin de conserver les avantages de la structuration non hiérarchique des données, ces arbres peuvent être produits dynamiquement lors de l'exploration des données. Ceci implique l'extraction de nouvelles données pour la construction de chaque génotype produit au fur et à mesure de l'exploration des données. Selon les applications, le génome peut être fixe ou défini par des requêtes utilisateur. Or le temps de production d'un génotype peut varier fortement en fonction du système de gestion de données, du nombre de données accessibles et de l'implémentation du traducteur de génome. Pour certaines applications temps réel, il peut donc être nécessaire de pré-calculer un ensemble de génotypes qui seront ensuite sauvegardés sur le serveur, ce qui limite les requêtes pouvant être effectuées par l'utilisateur final à l'ensemble de requêtes prédéfinies.

Le module de structuration de l'information permet de définir la structure exploratoire de l'interface graphique à travers un fichier appelé génome. Celui-ci définit les niveaux d'exploration et les attributs associés à chaque élément d'information extrait de l'ensemble de données. Le générateur de génotype traduit ce phénomène en une requête compatible avec l'ensemble de données en entrée du système. Le résultat obtenu, appelé génotype ou arbre d'exploration augmenté, est ensuite transféré au module de présentation de l'information.

Si la séparation nette entre la structuration de l'information et la requête obtenue assure la généralité des génomes créés, elle nécessite la création d'un traducteur pour chaque système de gestion de données pris en compte par notre formalisme. La création d'un tel traducteur pouvant s'avérer complexe, et l'accès direct au système de gestion de données n'étant pas assuré dans tous les cas, nous proposons également la traduction de résultats de requêtes issus d'applications tierces vers le génotype.

Dans la section suivante, nous détaillons le module de présentation de l'information prenant en entrée un génotype et permettant la production de l'interface graphique 3D finale.

4.3 Module de présentation de l'information

Le module de structuration de l'information permet de produire un génotype représentant la structure exploratoire de l'interface graphique 3D. Une fois l'information structurée, il est nécessaire de lui appliquer une métaphore visuelle, d'interaction et d'exploration permettant de passer d'un mode de représentation logique à une présentation dans l'espace 3D. Le module de présentation de l'information permet au concepteur de l'interface 3D de définir cette métaphore, puis de produire l'interface graphique 3D présentée à l'utilisateur final. En section 4.3.1, nous définissons les concepts généraux du module de présentation de l'information. Puis, en section 4.3.2, nous présentons plus en détail le langage formel de description de métaphores de présentation de l'information, ses concepts et ses classes d'opérateurs, ainsi qu'un ensemble de fonctionnalités permettant entre autre de valider l'application d'une métaphore de présentation à un génotype et de valider la compatibilité de la métaphore pour l'utilisation de l'interface via certains dispositifs d'interaction et de restitution. En section 4.3.3, nous présentons deux implémentations du générateur de phénotype permettant d'appliquer un phénomène à un génotype pour produire l'ensemble de scènes 3D composant l'interface graphique. Enfin, en section 4.3.4, nous concluons sur les caractéristiques du module de présentation de l'information.

4.3.1 Vue d'ensemble

4.3.1.1 Généralités

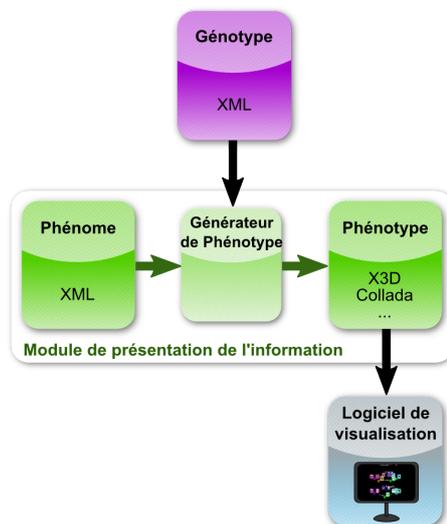


Figure 4.13 – Architecture du module de présentation de l'information.

Le module de présentation de l'information (voir figure 4.13) permet la définition de la métaphore visuelle, d'interaction et d'exploration de l'interface graphique 3D. Les langages de description de métaphores, tels que XSLT ou X-VRML (voir section 3.3.2), intègrent directement les opérateurs de modélisation du format 3D choisi comme format final de description des scènes produites. Dans notre étude, nous souhaitons proposer un formalisme indépendant du format 3D en sortie. Une même interface graphique 3D peut donc être produite dans plusieurs formats 3D, et ce sans modifier le fichier de description de la métaphore de présentation de l'information. Le format de description des scènes 3D produites n'est pas prédéfini mais déterminé par le composant de génération de phénotype. Nous proposons donc un langage générique pour la description de la métaphore de présentation de l'information d'une interface graphique. Les métaphores décrites à l'aide de ce langage sont appelées phénomènes. Le générateur de phénotype permet d'appliquer la métaphore de présentation de l'information au génotype issu du module de structuration de l'information afin de produire un ensemble de scènes 3D possédant les caractéristiques suivantes :

- sa géométrie est conforme au phénomène ;
- sa topologie est conforme au génotype.

Cet ensemble de scènes est appelé phénotype. Une application tierce (comme par exemple un module d'extension) compatible avec le format 3D choisi effectue le rendu du phénotype. Elle assure l'affichage de l'interface graphique 3D et gère les interactions entre l'utilisateur final et l'interface graphique.

Comme nous l'avons vu en section 3.3, la métaphore choisie pour la présentation de données doit pouvoir être entièrement adaptée au domaine applicatif des données ainsi qu'aux utilisateurs finaux visés par l'application. En effet, l'utilisation d'une interface générique ne permet pas de prendre en compte les connaissances a priori de l'utilisateur final concernant les données visualisées. Ces connaissances ne peuvent donc pas être exploitées pour améliorer l'accès à l'information recherchée. Par exemple, l'utilisation d'une métaphore urbaine basée sur des immeubles, des étages et des appartements peut être adaptée pour la recherche de personnes, mais elle risque de perdre l'utilisateur si elle est utilisée pour la visualisation de la structure d'un programme informatique. Les interfaces abstraites sont quant à elles plus facilement adaptables à tous types de données. Cependant, elles ne permettent pas la prise en compte des connaissances de l'utilisateur final et peuvent entraîner un temps d'adaptation à l'interface supérieur à l'utilisation d'une métaphore exploitant la sémantique des données à visualiser. Dans le cadre de nos travaux, nous souhaitons permettre aux concepteurs d'interfaces graphiques de définir tous types de métaphores, qu'elles soient abstraites ou non, génériques ou spécialisées, etc. De plus, nous souhaitons proposer un formalisme permettant la définition de

métaphores diverses, allant des environnements virtuels immersifs multi-utilisateurs aux dispositions d'objets type catalogue virtuel, et tout type de métaphore intermédiaire.

Les métaphores de présentation créées à partir de notre formalisme ont pour but de présenter l'information structurée issue de grands ensembles de données. Elles peuvent être considérées comme un génome dans lequel un certain nombre d'instructions de modélisation et de présentation de l'information sont insérées. Le diagramme ainsi obtenu pour la description du phénomène de l'exemple 4.1.1 est présenté en figure 4.14. Ce diagramme correspond au diagramme du génome présenté en figure 4.10 (éléments en violet) modifié par l'ajout d'éléments de présentation de l'information et d'instructions de positionnement dans l'espace 3D (éléments en vert). Le contenu de l'interface graphique représentant des éléments d'information issus du génotype est appelé contenu générique. Il peut être entièrement ou partiellement dépendant de données non connues a priori. Ces données seront extraites du génotype au moment de la création du phénotype.

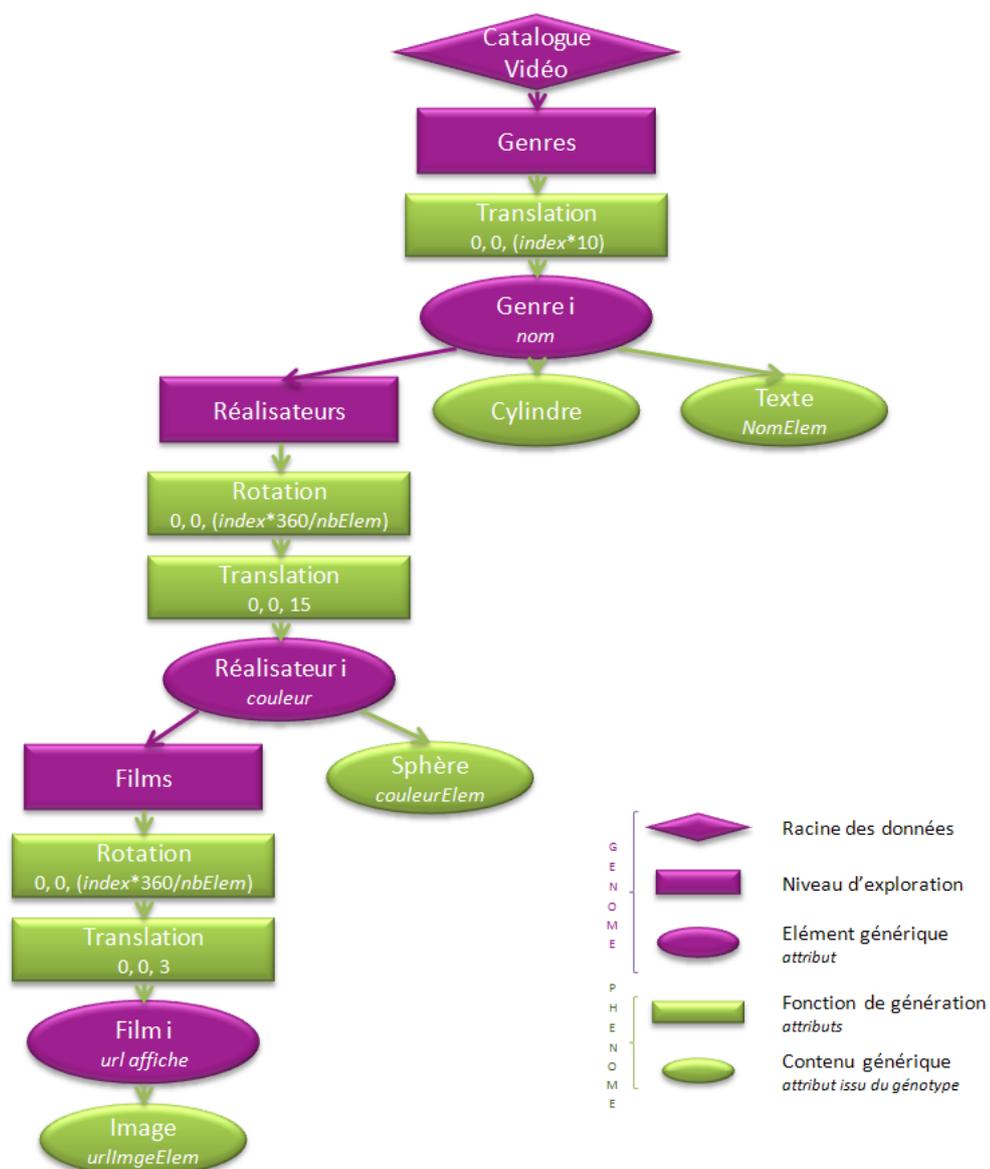


Figure 4.14 – Diagramme simplifié du phénomène correspondant à l'exemple 4.1.1. Les éléments en vert, correspondant à des opérateurs spécifiques au phénomène, sont insérés dans le diagramme du génome (éléments en violet, voir figure 4.10) afin de définir le positionnement et la représentation 3D des éléments du génotype.

Les métaphores doivent permettre de présenter à la fois les données présentes dans le génotype, mais aussi les relations entre ces données, que ce soit en interne d'un niveau d'exploration (relations de fratrie) ou entre données appartenant à des niveaux d'exploration différents (relations de filiation). Ces relations structurelles peuvent être représentées via la position relative des objets représentant les données, via des connecteurs physiques (lignes, tuyaux, . . .), ou via la présence d'objets ou d'attributs spécifiques. Dans l'exemple 4.1.1, les relations de fratrie et de filiation sont représentées par la position relative des éléments. Si un cadre coloré en fonction de la couleur associée au réalisateur du film était ajouté autour de chaque affiche, la relation de filiation [Réalisateur->Film] serait représentée par l'attribut *couleur* du cadre (voir figure 4.15). Afin de caractériser la relation de filiation au moyen d'objets ou d'attributs, le phénomène doit pouvoir accéder aux ancêtres et aux descendants dans le génotype de l'élément courant pour lequel une représentation est définie.

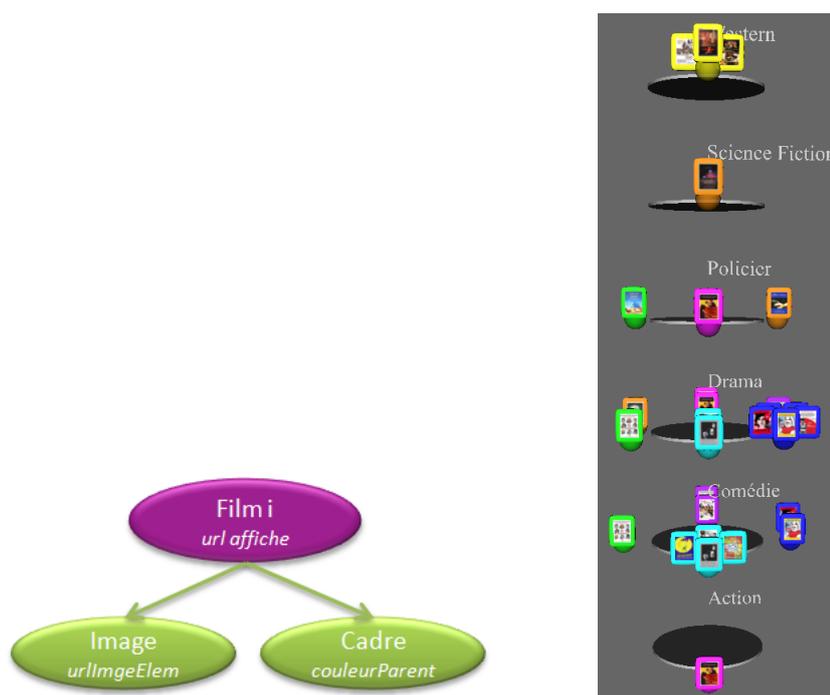


Figure 4.15 – Modification du diagramme 4.14 : ajout d'un cadre autour des affiches de film dont la couleur correspond à la couleur du parent de l'élément courant (ici, le réalisateur).

Dans l'exemple 4.1.1, l'ensemble de l'information contenue dans le génotype est représentée par une seule scène 3D. La métaphore proposée est dite multi-niveaux. Nous souhaitons également permettre la définition de métaphores de présentation produisant une scène 3D pour chaque niveau d'exploration du génotype. Chaque scène représente alors un niveau d'abstraction (par exemple, la scène représentant les comédies réalisées par Woody Allen contient les films *Alice* et *Manhattan*). Au fur et à mesure de la navigation, l'utilisateur explore donc de nouvelles scènes, chaque scène représentant une information de plus en plus précise. Le formalisme doit également permettre de mêler des métaphores multi-niveaux à des métaphores ne modélisant qu'un seul niveau d'exploration par scène 3D. De plus, lorsqu'une métaphore de présentation permet la modélisation de plusieurs niveaux d'exploration dans une même scène 3D, la présentation de l'ensemble des données peut nécessiter l'utilisation de technologies de type regroupement de données, fish-eye ou encore de focus sémantique contextuel (voir section 3.1). Le regroupement des données et le contexte sont directement accessibles grâce à la structuration de l'information sous forme d'arbre d'exploration (voir figure 4.16). Dans ce cas, le regroupement de rang i d'éléments de niveau d'exploration n correspond à l'élément parent de niveau d'exploration $n - i$. De même, le contexte de rang i d'éléments de niveau d'exploration n correspond aux frères de l'élément parent de niveau d'exploration $n - i$. Le module de présentation de l'information doit donc permettre de tirer parti des informations contextuelles et de structuration disponibles dans le génotype pour définir les objets visibles à un instant t , en fonction des paramètres de focus et

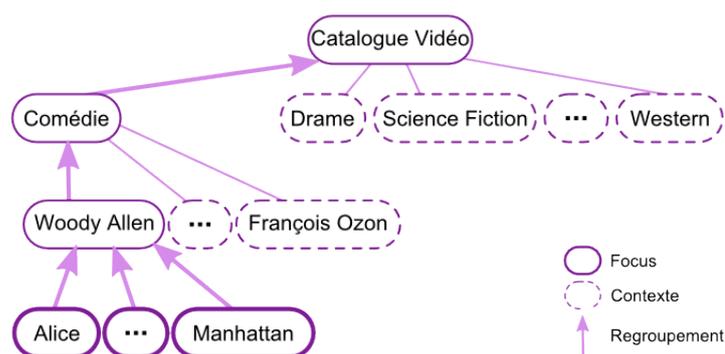


Figure 4.16 – Récupération des informations de regroupement et de contexte en fonction des éléments de focus. Ici, Woody Allen et Comédie sont respectivement les regroupements de rang 1 et 2 des films *Alice* et *Manhattan*. Les frères des éléments Woody Allen et Comédie définissent le contexte des films sur lesquels le focus est effectué.

de contexte définis. L'exemple 4.3.1 ci-dessous propose une extension de l'exemple 4.1.1 dans lequel un sous-ensemble de l'information est présenté à chaque instant. Une fonctionnalité supplémentaire permet de remplacer un objet par un autre, ou par un groupe d'objets.

Exemple 4.3.1 *Nous souhaitons ajouter à la métaphore de la tour de carrousel un certain nombre d'actions permettant à l'utilisateur final de découvrir l'information contenue dans le catalogue vidéo. Lors de l'accès à l'interface graphique, seuls les genres sont affichés. Un double clic sur le cylindre permet de faire apparaître les sphères représentant ses enfants (les réalisateurs). Lorsqu'un cylindre est développé (ses sphères sont apparentes), il est refermé (ses sphères disparaissent) au moment du développement d'un nouveau cylindre. Un double clic sur une sphère permet de faire apparaître ses enfants (les affiches de films). Cependant, plusieurs sphères peuvent être développées à un instant t . Un double clic sur une sphère développée la referme. La figure 4.17 présente l'interface graphique 3D obtenue par un utilisateur final après sélection des genres Drame, puis Comédie et des réalisateurs Pedro Almodovar puis Woody Allen.*

Comme exposé en section 2.3, aucun format 3D ne prédomine actuellement pour la description de scènes 3D sur le Web. Afin de préserver une certaine généralité, nous souhaitons donc pouvoir produire des phénotypes modélisés dans différents formats 3D (X3D, WebGL, ...). Le concepteur de l'interface graphique pourra sélectionner le format du phénotype en fonction des besoins de son application, comme par exemple la possibilité de proposer certains types d'interactions à l'utilisateur final, ou encore selon des critères d'accessibilité de l'interface graphique (nécessité ou non d'installer un module d'extension pour visualiser l'interface graphique, ...). Pour assurer cette généralité, le langage choisi pour le phénomène ne sera donc pas directement basé sur un unique format 3D. De plus, nous souhaitons proposer un formalisme modulaire permettant la réutilisation partielle ou complète des métaphores visuelles. Afin de garantir cette modularité, une métaphore visuelle est composée d'un ensemble de règles. Ces règles sont imbriquables, réutilisables et peuvent être paramétriques. Ainsi, si un phénomène $P1$ contient une règle $R(O)$ possédant un paramètre O et permettant la distribution linéaire d'objets O , un phénomène $P2$ peut définir la distribution bi-linéaire $R(R(O))$.

Le langage formel du phénomène doit donc permettre, grâce à des règles prédéfinies, de positionner des éléments multimédia (vidéos, images, modèles 3D, ...) dans un ensemble de scènes 3D, d'accéder à l'information stockée dans le génotype et de présenter cette information dans l'espace 3D. Elle doit également permettre de définir un ensemble de paradigmes d'interaction et d'exploration permettant respectivement l'interaction avec la scène et l'accès à l'information extraite.

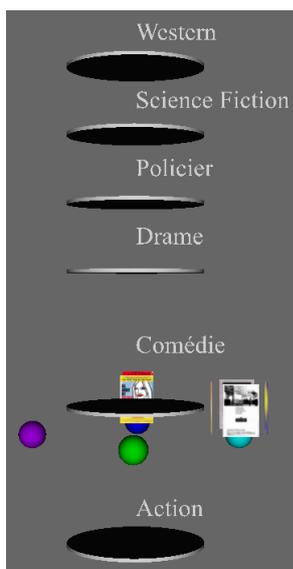


Figure 4.17 – Interface graphique 3D comportant des éléments pouvant être affichés/cachés en fonction des choix de l'utilisateur. Cette visualisation est obtenue par l'utilisateur final après sélection des genres Drame, puis Comédie et des réalisateurs Pedro Almodovar puis Woody Allen (voir exemple 4.3.1).

4.3.1.2 Sélection de la métaphore de présentation de l'information

Sur le Web, certains services proposent des interfaces définies par le concepteur d'un site, et des interfaces pouvant être personnalisées par l'utilisateur final. Afin de conserver ces caractéristiques, nous proposons quatre modes de définition de métaphores 3D :

- définition de la métaphore par le concepteur de l'interface graphique 3D lors de sa création ;
- sélection par l'utilisateur final d'une métaphore parmi un ensemble de métaphores prédéfinies par le concepteur de l'interface graphique 3D ;
- définition de la métaphore par l'utilisateur final ;
- sélection automatique de la métaphore par une application tierce parmi un catalogue de métaphores.

L'engouement pour le Web est en parti dû à la participation active des utilisateurs dans la production du contenu, et aux possibilités offertes quant à la personnalisation des interfaces. Ainsi, les interfaces graphiques de visualisation de courriers électroniques (webmails en anglais) ou encore les portails Web personnalisables offrent aux utilisateurs finaux de nombreux modèles de présentation. Certains proposent également aux utilisateurs de définir leur propre modèle de présentation, augmentant ainsi la personnalisation et l'appropriation du service par l'utilisateur (Weblogs, portails Web personnalisables, ...).

La sélection automatique d'une métaphore visuelle par une application peut être nécessaire dans différents contextes, comme par exemple dans le cadre des interfaces plastiques. Le choix de la métaphore visuelle peut alors reposer sur des critères directement liés au profil de l'utilisateur final. Par exemple, le style graphique de la métaphore peut être différent lorsque l'utilisateur final est un enfant ou un adulte, la taille du texte ou les couleurs choisies peuvent différer en fonction des handicaps de l'utilisateur afin d'augmenter l'accessibilité de l'interface graphique, etc. Le choix de la métaphore visuelle peut également être lié à la sémantique des données, comme par exemple lors de l'affichage d'une requête par un moteur de recherche Web. Ainsi par exemple, la visualisation de résultats de recherche sur des romans pourrait être affichée sous la forme d'une bibliothèque tandis que la visualisation d'une recherche sur des animaux pourrait être affichée sous la forme d'un zoo.

La sélection d'une métaphore visuelle peut aussi dépendre du dispositif de restitution ciblée. En effet, nous pouvons imaginer un service accessible sur plusieurs types de terminaux, comme par exemple un téléviseur, un téléphone portable ou un ordinateur. La taille des écrans, leur résolution, ou encore les capacités de calcul de ces différents terminaux sont très variables. Il est donc nécessaire de définir

des métaphores visuelles adaptées à chacun de ces terminaux. De même, les dispositifs d'interaction disponibles peuvent contribuer à la sélection de la métaphore visuelle. Notre formalisme facilite la définition de métaphores de présentation de l'information diverses adaptées aux dispositifs d'interaction et de restitution ciblées en proposant un unique langage de description de métaphores de présentation de l'information.

La sélection automatique d'une métaphore visuelle pour la visualisation de données peut combiner plusieurs des critères énumérés ci-dessus. Au-delà de ces critères, la sélection d'une métaphore doit également prendre en compte la structuration des données et leur compatibilité avec la métaphore choisie. Ainsi par exemple, il n'est pas possible de représenter l'ensemble des informations contenues dans un génotype contenant 4 niveaux d'exploration à l'aide de la métaphore de la tour de carrousels définie dans l'exemple 4.1.1. De même, le nombre maximum/minimum d'éléments d'un niveau d'exploration et le type d'attributs associées (image, couleur, ...) sont des paramètres déterminants pour la sélection d'une métaphore de présentation de l'information (Wiza *et al.* [96]). En section 4.3.2.7 nous présentons un certain nombre d'outils offerts par notre formalisme pour assurer la compatibilité d'un phénomène avec un génome ou un génotype, et ainsi faciliter la sélection automatique d'une métaphore visuelle.

4.3.1.3 Outil auteur pour la définition de métaphores de présentation de l'information

L'intégration du formalisme que nous proposons à un outil auteur facilite la création de métaphores de présentation de l'information. Afin d'être accessible au plus grand nombre tout en conservant la flexibilité et la généralité de notre formalisme, cet outil peut proposer trois interfaces correspondant à trois niveaux d'expertise des concepteurs d'interfaces graphiques 3D :

- *expert* : permet la création de règles et une modification directe du phénomène à travers le langage formel que nous proposons ;
- *intermédiaire* : permet l'imbrication de règles à partir par exemple de programmation graphique ;
- *occasionnel* : permet la sélection de règles existantes et la modification de leurs paramètres.

Ainsi un concepteur d'interface graphique 3D ayant de bonnes notions de programmation pourra créer de nouvelles règles comme par exemple une distribution linéaire suivant une fonction cosinus. Un concepteur intermédiaire pourra sélectionner et imbriquer des règles existantes, et éditer leurs paramètres, afin de créer de nouvelles métaphores de présentation. Il pourra par exemple créer la métaphore de la tour de carrousels présentée dans l'exemple 4.1.1 en imbriquant une règle de distribution linéaire avec deux règles de distribution circulaire. L'utilisateur occasionnel pourra quant à lui personnaliser la métaphore en sélectionnant une métaphore préexistante et en éditant les paramètres des règles et des objets. Par exemple, à partir de la métaphore de la tour de carrousels, il pourra remplacer les cylindres blancs par des cubes oranges et changer la police de caractères utilisée pour les noms des genres.

La modularité de notre formalisme, assurée par la création de règles, facilite l'implémentation d'un tel outil auteur.

4.3.2 Langage formel de présentation de l'information : le phénomène

Le phénomène permet de décrire la métaphore de présentation de l'information qui sera appliquée au génotype afin de produire le phénotype, c'est-à-dire l'ensemble de scènes 3D composant l'interface graphique. Il permet la mise en correspondance entre chaque niveau d'exploration du génotype et un ensemble de règles de présentation de l'information. Ces règles contiennent la description des différentes scènes 3D qui composent l'interface graphique, c'est à dire l'environnement dans lequel l'information est présentée, l'agencement des objets symbolisant l'information, les objets représentant les différentes données ainsi qu'un ensemble d'éléments liés à l'interaction avec le contenu et à l'exploration des données. Dans l'exemple 4.1.1, seules les données extraites du génotype sont représentées : les genres sont représentés par des cylindres blancs distribués linéairement à la verticale, les réalisateurs sont représentés par des sphères colorées distribuées circulairement autour du genre correspondant, et les films sont représentés par leur affiche et distribués circulairement autour du réalisateur correspondant (voir figure 4.1(b)). Dans d'autres cas, un certains nombres d'objets de décor ou facilitant l'interaction, par exemple, peuvent être intégrés à la scène.

Le langage formel du phénomène comporte cinq grandes catégories d'opérateurs :

- les opérateurs de *gestion de règles* ;
- les opérateurs de *modélisation*, permettant la gestion de scènes ainsi que la description de la géométrie et des transformations dans l'espace 3D ;
- les opérateurs d'*accès au génotype* ;
- les opérateurs d'*interaction* et d'*exploration* ;
- les opérateurs divers, comme par exemple les opérateurs mathématiques ou encore les opérateurs de structure de contrôle.

Le phénomène est une représentation d'un modèle procédural de présentation de l'information. Il comporte, entre autre, un ensemble de graphes de scènes génériques contenant à la fois les invariants du modèle, et les paramètres et fonctions de production permettant la présentation de l'information structurée du génotype. Lors de l'exécution, le phénomène est appliqué à un génotype afin de produire un ensemble de graphes de scènes décrits dans un format 3D spécifique (le phénotype). Celui-ci est ensuite exécuté par une application tierce afin d'afficher l'interface graphique finale.

Au niveau de l'implémentation, le phénomène est décrit sous la forme d'un fichier XML conforme au schéma XML que nous avons défini. Il est décomposable en trois parties : le préambule contenant diverses informations, la mise en correspondance entre les règles et les niveaux d'exploration, et enfin la description des règles. Les sections suivantes détaillent les différentes catégories d'opérateurs disponibles.

4.3.2.1 Niveaux d'exploration et hiérarchie de règles

Le phénomène décrit la métaphore visuelle, d'interaction et d'exploration permettant de visualiser interactivement l'information structurée du génotype. Cette métaphore permet l'application de règles spécifiques pour chaque niveau d'exploration du génotype. La mise en correspondance entre chaque niveau d'exploration et les règles correspondantes est organisée en niveaux segmentés et est décrite à l'aide de l'ordonnancement de règles `RuleOrder`. Cet opérateur hiérarchique contient un ensemble d'enfants `Level` représentant chaque niveau d'exploration visualisable à l'aide du phénomène. Le premier niveau correspond au niveau 0 du génotype, c'est-à-dire sa racine. Sur la figure 4.8 (page 50), le niveau 0 du génotype C correspond au catalogue vidéo, tandis que le niveau 0 du génotype D correspond au film *Manhattan*. Un extrait du phénomène de la tour de carrousels est présenté dans le programme 4.4.

Programme 4.4 Extrait du phénomène correspondant à l'exemple 4.1.1 : ordonnancement des règles.

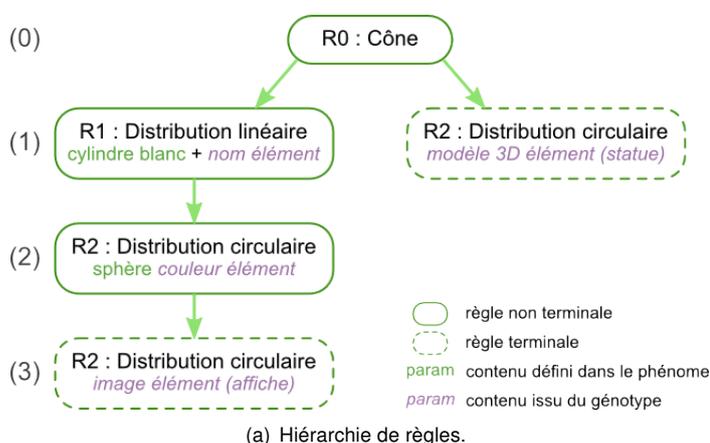
```
<Phenome>
  <RuleOrder>
    <Level><!--level 0 : catalogue vidéo -->
      <Rule>...</Rule>
    </Level>
    <Level><!--level 1 : genres -->
      <Rule>...</Rule>
    </Level>
    <Level><!--level 2 : réalisateurs -->
      <Rule>...</Rule>
    </Level>
    <Level><!--level 3 : films -->
      <Rule>...</Rule>
    </Level>
  </RuleOrder>
  ...
</Phenome>
```

Chaque opérateur `Level` permet d'appeler un ensemble de règles décrivant la présentation du niveau d'exploration correspondant. Généralement, une seule règle suffit à l'affichage de l'ensemble des éléments d'un niveau d'exploration donné. Cependant, d'autres règles peuvent être ajoutées, par exemple

pour combiner plusieurs représentations des éléments, pour ajouter un décor global à la scène, ou encore pour ajouter des éléments d'exploration. Par exemple, en supplément d'une règle de présentation 3D des éléments d'un niveau d'exploration, l'utilisation d'une règle décrivant une présentation 2D du contexte de l'exploration permet l'affichage d'une carte simplifiée de l'interface graphique pour faciliter l'exploration de l'utilisateur et l'utilisation de raccourcis lors de ses déplacements.

L'appel à une règle peut intégrer l'opérateur `NextRule` définissant l'insertion des règles correspondant au prochain niveau d'exploration. Dans le cas contraire, la règle est dite règle terminale. Ainsi par exemple, si un niveau utilise une règle R1 définissant une représentation 3D complète des éléments actuels et une règle R2 définissant une carte 2D simplifiée du contexte de l'exploration, la règle R1 nécessitera l'imbrication des règles correspondant au niveau d'exploration suivant, tandis que R2 sera considérée comme une règle terminale. La définition des règles associées à chaque niveau d'exploration permet la définition d'une hiérarchie de règles dont les feuilles sont les règles terminales. L'exemple 4.3.2 étend l'exemple 4.1.1 en ajoutant une règle terminale pour la présentation du premier niveau d'exploration.

Exemple 4.3.2 Nous ajoutons à la métaphore de carrousels de l'exemple 4.1.1 une règle de présentation des éléments du premier niveau d'exploration (les genres) : chaque élément est représenté par une statue 3D distribuée circulairement sous la tour de carrousels, et autour d'un cône. L'interface obtenue est présentée en figure 4.18(b). Nous ajoutons à l'ordonnanceur de règle (pour les éléments du premier niveau d'exploration) un appel supplémentaire à la règle de distribution circulaire avec pour paramètre la statue représentant l'élément courant et extraite du génotype. Une nouvelle règle est insérée à la métaphore de la tour de carrousels afin de disposer le cône dans la scène (associée au niveau d'exploration zéro). La hiérarchie de règles correspondante est présentée en figure 4.18(a).



(b) Interface graphique obtenue.

Figure 4.18 – Statues représentant les éléments du premier niveau d'exploration (les genres) ajoutées à la métaphore de la tour de carrousels.

Les règles appelées par l'ordonnanceur de règles peuvent être soit définies dans le phénomène courant, soit dans un phénomène différent. Dans ce dernier cas, la localisation du phénomène implémentant la règle est indiquée lors de l'appel. Les règles externes peuvent être accessibles via des catalogues en ligne ou des bibliothèques de règles pré-définies.

4.3.2.2 Les règles

Le générateur de phénotype permet de créer un ensemble de graphes de scènes, appelé phénotype, à partir du phénomène. Chaque règle du phénomène représente un ensemble de graphes de scènes

et/ou sous-graphes de scènes génériques. Elle contient les invariants des graphes de scène et sous-graphes de scène produits, ainsi qu'un certain nombre de fonctions de production. Les nœuds du graphe de scène générique sont également considérés comme génériques, c'est-à-dire pouvant être instanciés et initialisés par le générateur de phénotype afin de modéliser l'interface graphique finale.

Graphes de description de métaphores Une règle peut être représentée sous la forme d'une arborescence contenant, outre un ensemble d'objets et leurs relations, un ensemble de fonctions et paramètres qui seront respectivement appelées et instanciés par le générateur de génotype. Les nœuds d'une règle peuvent donc contenir un certain nombre de champs spécifiques correspondant aux nœuds enfants, définissant ainsi la hiérarchie de la règle. Ces nœuds enfants peuvent à leur tour contenir des nœuds. Les descendants d'un nœud correspondent alors aux nœuds qu'il possède dans ses champs, et aux descendants de ces nœuds. Au contraire, les ancêtres d'un nœud N correspondent aux nœuds ayant N pour descendant. Les nœuds d'une règle sont appelés *éléments* et sont classés en trois catégories selon le graphe auxquels ils appartiennent :

- la *hiérarchie de graphes de scène* décrit l'imbrication des différents graphes de scènes. Les nœuds *NewScene* correspondent à la création d'un nouveau graphe de scène générique et définissent la hiérarchie de graphes de scènes. Tous les descendants d'un nœud *NewScene* N, hormis ses enfants de type *NewScene* et leurs descendants, appartiennent au graphe de scène générique N. Les descendants directs d'un nœud *NewScene* sont appelés *nœuds racine* du graphe de scène générique correspondant ;
- la *hiérarchie de transformations* décrit les relations spatiales entre les différents objets. Une hiérarchie de transformation est associée à chaque graphe de scène décrit par le phénomène. Elle contient l'ensemble des nœuds ayant une localisation dans l'espace géométrique défini par le graphe de scène générique auquel ils appartiennent. Le phénomène suppose une notion de *système de coordonnées locales*, défini en termes de transformations relatives au système de coordonnées de l'ancêtre. Le système de coordonnées des nœuds racine est appelé *système de coordonnées global* du graphe de scène générique correspondant. La hiérarchie de transformation contient l'ensemble des invariants et fonctions de production permettant de décrire les parties directement perceptibles des scènes 3D produites ;
- le *graphe de comportements* décrit le flux d'information circulant dans l'ensemble du phénomène et les connexions entre les différents champs. Un graphe de comportement est associé à chaque graphe de scène décrit par le phénomène. Le modèle d'évènements du phénomène permet la déclaration de connexions entre différents champs, et un modèle de propagation d'évènements le long de ces connexions. Le graphe de comportements contient l'ensemble de ces connexions de champs. Il permet, entre autre, l'utilisation de sources de données provenant de dispositifs variées (par l'intermédiaire de capteurs) dans le but de modifier le graphe de scène produit en réponse aux instructions contenues dans le graphe de comportements.

Chaque nœud élément peut posséder comme attribut un nom. Dans la majorité des cas, tous les paramètres (hormis son nom) d'un élément sont définis en tant qu'enfant de cet élément afin de faciliter l'insertion de fonctions de production et d'appels au génotype à tous les niveaux de la définition d'un élément.

L'exemple 4.3.3 ci-dessous illustre la décomposition du phénomène en hiérarchie de règles, hiérarchie de graphes de scènes, hiérarchies de transformations et graphes de comportement, ainsi que la transformation induite lors de la création du phénotype.

Exemple 4.3.3 Dans cet exemple, nous reprenons la métaphore de la tour de carrousels en subdivisant l'interface graphique de manière à créer une nouvelle scène pour chaque niveau d'exploration. La première scène présente l'ensemble des genres sous la forme d'une ligne de statues. Pour chaque genre, une nouvelle scène est créée et présente les réalisateurs sous la forme d'un carrousel de sphères colorées. Enfin, pour chaque réalisateur de chaque genre, une nouvelle scène est créée et présente les films sous la forme d'un carrousel d'affiches. La hiérarchie de règle est présentée en figure 4.19 (a). L'ordonnement des règles permet de définir la hiérarchie de graphes de scènes correspondant, ainsi que les hiérarchies de transformations et les graphes de comportement associés à chaque graphe de scène correspondant (voir figures 4.19 (b), (c) et (d)). Ces graphes de métaphore du phénomène sont ensuite exécutés par le générateur de génotype pour produire l'ensemble de fichiers correspondant. L'interface graphique obtenue est présentée en figure 4.20.

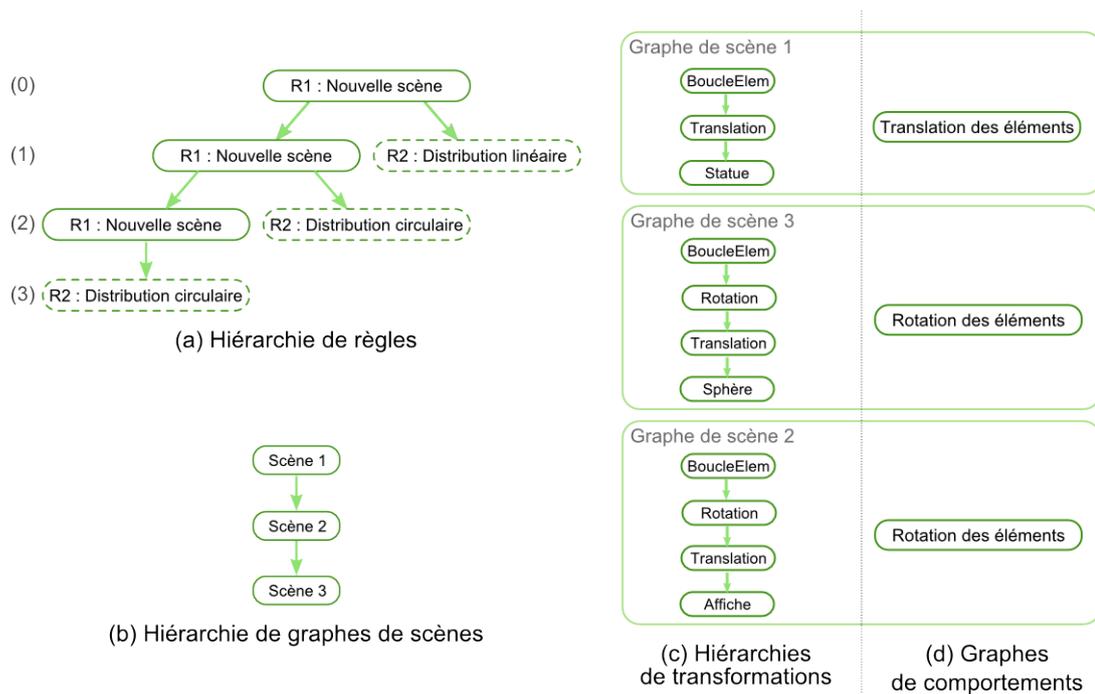


Figure 4.19 – Graphes de description des métaphores (exemple 4.3.3).

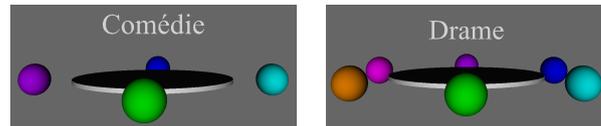
Au moment de la création du phénotype, le générateur de phénotypes produit un ensemble de fichiers correspondant aux différentes scènes de la hiérarchie de graphes de scènes. Pour chaque scène de cette hiérarchie, le générateur produit autant de scènes que d'éléments appartenant au niveau d'exploration correspondant. Soit g le nombre de genres et $r_i, i \in (1, g)$ le nombre de réalisateurs enfants du genre de rang i . Le générateur de phénotypes produit un fichier pour la scène 1, g fichiers pour la scène 2 et $\sum_{i=1}^g r_i$ fichiers pour la scène 3. Ainsi, lors de l'application du phénomène présenté dans l'exemple 4.3.3 au génotype présenté en figure 4.9 (page 50), le générateur de phénotype produit $1 + g + \sum_{i=1}^g r_i$ fichiers. Chaque fichier contient le modèle 3D de la scène correspondante décrit dans le format 3D choisi par le concepteur de l'interface graphique. Une application tierce (comme par exemple un navigateur Web) permet alors la visualisation de ces fichiers, et gère les interactions avec le contenu ainsi que le chargement des différents fichiers (voir figure 4.20).

Imbrication de règles et règles paramétriques L'imbrication des règles peut se faire de deux façons différentes : soit par l'insertion de la règle suivante, soit par l'instanciation d'une autre règle. Dans le premier cas, la règle insérée correspond à la règle suivante, si elle existe, dans la hiérarchie de règles définie par l'ordonnanceur de règles. L'opérateur correspondant est `NextRule()`. Dans le deuxième cas, l'appel à la règle correspondante est effectuée à l'aide de l'opérateur `CallRule(ruleName)`. Lorsque la règle imbriquée est implémentée dans un autre phénomène que le phénomène courant, la localisation de ce phénomène est indiquée en paramètre de l'appel. L'imbrication des règles implique la possibilité de la répartition d'un graphe de scène générique en sous-graphes de scènes génériques appartenant à des règles différentes. Ainsi, dans l'exemple 4.1.1, l'unique graphe de scène générique représentant la métaphore de la tour de carrousels est construit à partir des trois règles correspondant aux trois niveaux d'exploration.

Une règle peut posséder un certain nombre de paramètres. Elle possède alors une interface définissant les types, noms et valeurs par défaut de chacun de ses paramètres. Ainsi, dans l'exemple 4.1.1, les deux règles permettant de former les carrousels peuvent être définies comme une règle unique de distribution circulaire autour de l'axe vertical de la scène (voir programme 4.5). Cette règle contient



(a) Graphe de scène de niveau 1 : les genres.



(b) Graphes de scène de niveau 2 : les réalisateurs.



(c) Graphe de scène 3 : les films.

Figure 4.20 – Interface graphique composée de plusieurs scènes 3D (exemple 4.3.3). Chaque image présente l'ensemble des éléments géométriques de la scène 3D correspondante.

alors deux paramètres : le diamètre du carrousel et le nœud à distribuer. Ce dernier correspond à un nœud représentant une sphère colorée pour le carrousel intermédiaire (les réalisateurs), et à une affiche pour le carrousel terminal (les films). Par défaut, un paramètre correspond à un nœud de type élément. Il peut cependant être restreint à un type plus spécifique (comme par exemple une expression mathématique). L'appel à un paramètre se fait à l'aide de l'opérateur `Param` possédant pour argument le nom du paramètre correspondant. Par la suite, tous les nœuds de type "élément" peuvent être remplacés par un nœud `Param` de type compatible.

4.3.2.3 Opérateurs de modélisation

La métaphore de présentation décrite par le phénomène permet de représenter différents niveaux d'exploration dans différentes scènes 3D. La déclaration d'une nouvelle scène est effectuée par l'opérateur `NewScene`. Tous les descendants de cet opérateur, hormis ses enfants `NewScene` et leurs descendants, appartiennent à la nouvelle scène 3D ainsi définie. Chacune de ces scènes est définie par un graphe de scène générique incluant une hiérarchie de transformation et un graphe de comportement. La navigation entre les scènes est activée par un capteur de type ancre (voir section 4.3.2.6).

L'ensemble des transformations et objets possédant une localisation dans l'espace 3D de la scène définissent la hiérarchie de transformation. Nous pouvons distinguer trois sous-catégories d'opérateurs.

Les opérateurs de transformation et de regroupement permettent de regrouper un ensemble de nœuds et de leur appliquer la transformation correspondante. Ils sont de deux types : les opérateurs de translation (`Translation`), de rotation (`Rotation`) et de mise à l'échelle (`Scale`) d'une part, et deux opérateurs de regroupement de nœuds (`Group` et `Viewable`) d'autre part. Outre les opérateurs de regroupement, ces opérateurs possèdent trois à quatre enfants correspondant aux attributs de la transformation (transformation selon l'axe X , l'axe Y et l'axe Z , plus un angle α dans le cadre de la rotation) ainsi qu'un ou plusieurs nœuds de type élément sur lesquels la transformation est appliquée. L'opérateur `Group` correspond à un regroupement neutre de nœuds. L'opérateur `Viewable` permet de décrire des métaphores basées sur les concepts de zoom, de contexte et de regroupement de données (voir section 3.1). Il définit un ensemble de contenus potentiellement visibles, dont l'état par défaut peut être visible (`Show`) ou caché (`Hide`). Ainsi, l'exemple 4.3.1 (voir figure 4.17 page 62) utilise l'opérateur `Viewable` pour déterminer les éléments visibles au fur et à mesure de l'exploration des données.

Programme 4.5 Extrait du phénomène correspondant à l'exemple 4.1.1. Une règle paramétrique de distribution circulaire est utilisée pour représenter les niveaux d'exploration 2 et 3. Au niveau d'exploration 3, le paramètre correspondant au rayon de la distribution circulaire n'étant pas indiqué, la valeur par défaut sera utilisée. Le niveau d'exploration 3 étant terminal, il ne possède pas d'opérateur `NextLevel`.

```

<Phenome>
  <RuleOrder>
    ...
    <Level><!--level 2 : réalisateurs -->
      <Rule>
        <RuleRef>carrousel</RuleRef>
        <RuleParam><Name>Object</Name><Value>sphère colorée</Value></RuleParam>
        <RuleParam><Name>Ray</Name><Value>15</Value></RuleParam>
        <NextLevel/>
      </Rule>
    </Level>
    <Level><!--level 3 : films -->
      <Rule>
        <RuleRef>carrousel</RuleRef>
        <RuleParam><Name>Object</Name><Value>affiche du film</Value></RuleParam>
      </Rule>
    </Level>
  </RuleOrder>
  <Rule name="carrousel">
    <RuleParam type="elem">
      <Name>Object</Name>
      <Value>cube</Value>
    </RuleParam>
    <RuleParam type="func">
      <Name>Ray</Name>
      <Value>5</Value>
    </RuleParam>
    ...
    <Param name="Ray"/>
    ...
  </Rule>
</Phenome>

```

Les opérateurs de contenu représentent l'ensemble des éléments géométriques et sonores de l'interface. Ils sont divisés entre les opérateurs *géométriques* et les opérateurs *multimédia*. Les opérateurs géométriques correspondent soit à la description de primitives géométriques (par exemple *Cube*, *Sphere* ou *Circle*), soit à l'appel à un modèle 3D défini dans un fichier séparé (*External3DModel*). Les opérateurs de description de primitives géométriques possèdent un certain nombre d'enfants correspondant aux attributs de la forme géométrique (côtés du parallépipède rectangle, rayon de la sphère, ...) et peuvent posséder un attribut de type *apparence* définissant les attributs de texture et de matériaux de l'objet. L'appel à un modèle 3D possède pour paramètre le format du modèle. Celui-ci définit alors, à travers ses paramètres, un ensemble de données permettant l'utilisation du modèle. Par exemple, pour un modèle de type X3D, deux sous-types de modèles peuvent être proposés (voir figure 4.21) : les modèles complets X3D (accessibles via l'url du fichier) ou les prototypes X3D (accessibles via l'url du fichier et l'interface du prototype). Les modèles 3D peuvent appartenir à la catégorie des objets intelligents, c'est-à-dire décrivant leurs propres comportements. Ainsi, par exemple, un modèle de porte peut contenir les transformations nécessaires à l'ouverture et la fermeture de la porte ainsi que les capteurs permettant de déclencher l'ouverture et la fermeture de la porte (par exemple, un clic sur la porte, un capteur de proximité, ou encore un capteur de visibilité).

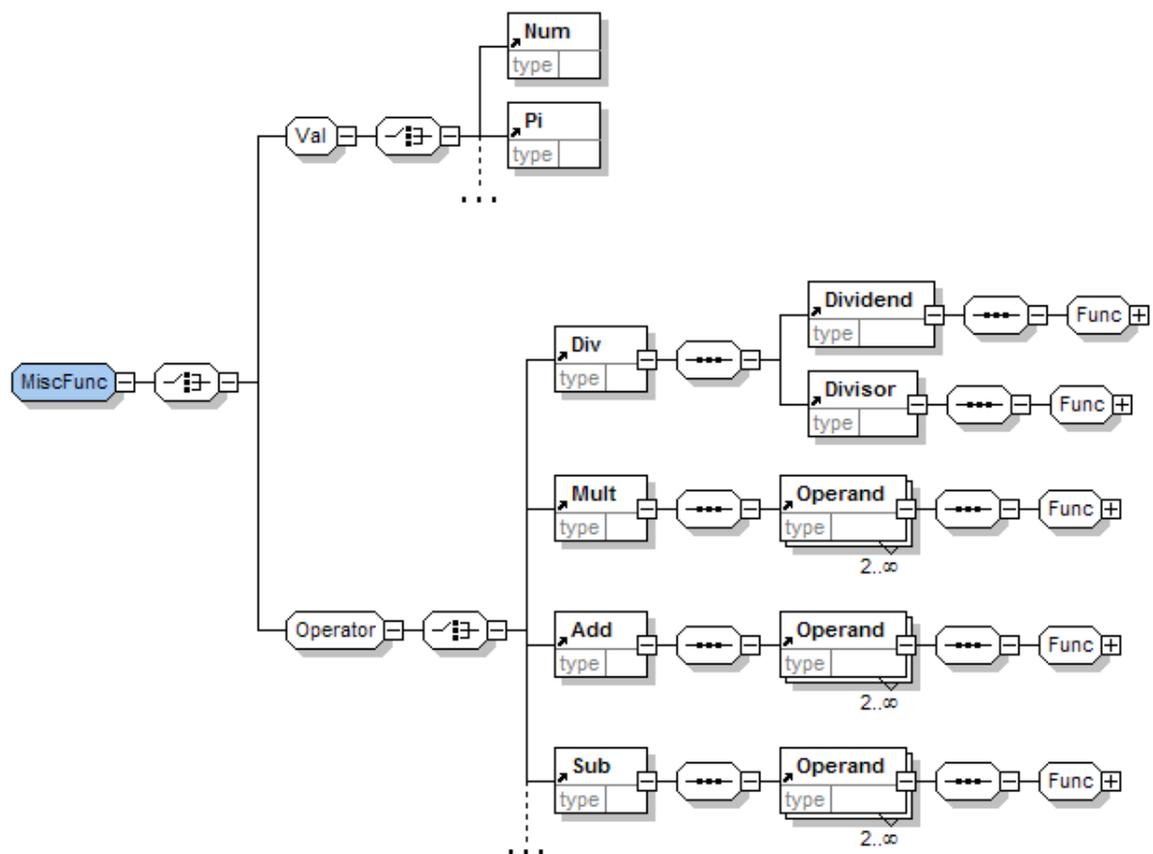


Figure 4.22 – Extrait du schéma XML correspondant aux fonctions mathématiques.

4.3.2.5 Insertion de l'information : accès au génotype

Notre formalisme permet de définir des métaphores de visualisation interactive de l'information. Le phénomène permet de définir la présentation de l'information de cette métaphore. Il doit donc accéder aux données structurées sauvegardées dans le génotype et les intégrer aux fonctions de production définies à l'intérieur de ses règles. Deux types principaux d'information sont nécessaires à la définition d'une métaphore de présentation : les attributs typés (couleur, texte, modèle 3D, ...) associés à chaque élément du génotype et un certain nombre de fonctions permettant d'extraire l'information concernant la structure du génotype (nombre d'éléments d'un niveau d'exploration par exemple). De nombreux opérateurs possèdent donc un équivalent permettant d'accéder aux attributs et informations structurelles du génotype :

- les opérateurs de contenu géométrique et multimédia (par exemple `GenoShape` correspondant aux primitives géométriques, `Geno3DModel` correspondant à l'opérateur `External3DModel`, `GenoVideo`, ou encore `GenoColor`);
- les opérateurs de chaînes de caractères (par exemple `GenoString`, `GenoURL`, ou encore `GenoName` permettant d'extraire le nom d'un élément du génotype);
- les valeurs numériques (par exemple `GenoNum`, `NbElem` indiquant le nombre d'éléments d'un niveau d'exploration, `Index` indiquant la position d'un élément dans le niveau d'exploration auquel il appartient);
- les opérateurs de structure de données (par exemple `ForEachElem` permettant d'effectuer une boucle sur les éléments d'un niveau d'exploration).

Chaque règle du phénomène est associée à un niveau d'exploration du génotype (voir section 4.3.2.1). Le niveau d'exploration courant d'une règle correspond donc au niveau d'exploration auquel elle est associée, le changement de niveau d'exploration étant effectué au moment de l'application de la règle

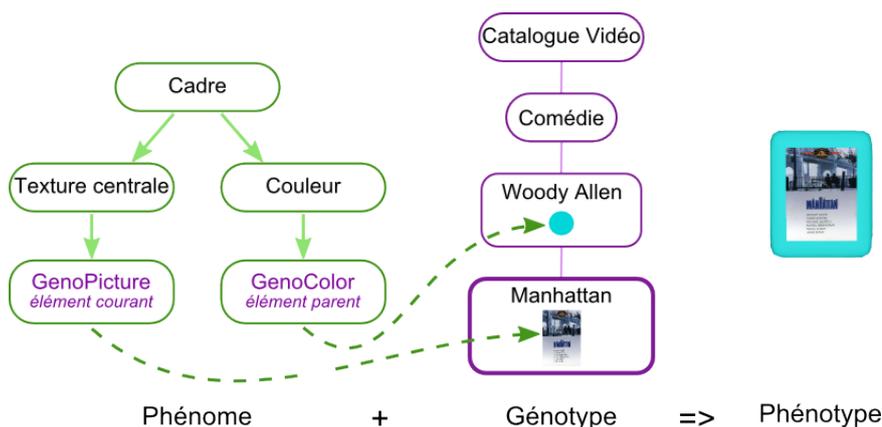


Figure 4.23 – Insertion d'information issue du génotype : accès au génotype depuis le phénomène, information correspondante dans le génotype et phénotype obtenu. L'élément courant dans le génotype est le film *Manhattan*. L'accès au génotype permet au phénomène de définir une métaphore visuelle intégrant des informations de filiation (la couleur de l'élément parent).

suivante dans la hiérarchie de règles. Dans la majorité des cas, l'accès au génotype est utilisé pour représenter les éléments du niveau d'exploration courant, et plus particulièrement l'élément courant dans le génotype. Par défaut, un opérateur d'accès au génotype est donc appliqué au niveau d'exploration courant (par exemple `ForEachElem`) ou à l'élément courant (par exemple, `GenoPicture`). Cependant, il peut être nécessaire d'accéder à des attributs ou informations structurelles concernant les ancêtres ou les descendants d'un élément afin de représenter le contexte de l'information. Par exemple, le cadre entourant l'affiche d'un film peut conserver la couleur de son parent (le réalisateur, voir figure 4.23), tandis que le rayon du cylindre représentant un genre peut changer en fonction du nombre de ses descendants. Les opérateurs d'accès au génotype possèdent donc un attribut permettant de définir le chemin relatif entre l'élément actuel et la position, dans le génotype, de l'information recherchée. De plus, les attributs du génotype sont typés et peuvent éventuellement posséder un ou plusieurs labels (voir section 4.2.1.2). Les opérateurs d'accès aux attributs d'un élément du génotype peuvent donc posséder des enfants permettant de sélectionner les attributs de l'élément du génotype en fonction de leur type et ou de leurs labels.

4.3.2.6 Fonctionnalités d'interaction et d'exploration

Une interface graphique permet de présenter, de façon interactive, l'information à l'utilisateur final. Dans la majorité des cas, elle permet également à cet utilisateur d'interagir en modifiant, par ses actions, les paramètres de l'interface, comme par exemple en déplaçant le point de vue, en manipulant les objets présentés, ou en activant des comportements. Afin d'être complet, notre formalisme doit donc permettre la description de capteurs d'actions effectuées par l'utilisateur final, et la description de flux d'événements entraînant une modification de l'interface graphique en fonction de ces capteurs. Ainsi par exemple, un capteur sur un objet doit permettre à l'utilisateur de l'observer en modifiant son orientation selon un, deux ou trois degrés de libertés, tandis qu'un capteur sur une porte permettra à l'utilisateur de l'ouvrir d'un simple clic, déclenchant ainsi une modification de la scène (la rotation de la porte) sur un laps de temps déterminé. En supplément des capteurs d'actions, notre formalisme doit également gérer des capteurs d'événements temporels pour produire les animations et des capteurs liés aux objets de la scène (par exemple, la collision).

L'information présentée selon notre formalisme est structurée dans le génotype sous la forme d'un arbre d'exploration. Nous devons donc également proposer un ensemble d'opérateurs permettant de naviguer dans l'information contenue dans cet arbre, que ce soit de manière verticale (dans la filiation) ou transversale (dans la fratrie). Ces opérateurs sont appelés opérateurs d'exploration.

Les opérateurs de captation permettent de prendre en compte un changement d'état et de transférer cette information aux opérateurs d'interactions auxquels ils sont liés. Nous pouvons distinguer les

capteurs de temps (TimeSensor, ...), les capteurs physiques (CollisionSensor, ...), les capteurs environnementaux (ProximitySensor, VisibilitySensor, ...) et les capteurs utilisateur (KeySensor, TouchSensor, DragSensor, ...) ainsi que les opérateurs de réseau et de gestion de fichier (LoadSensor, Anchor, ...).

Les opérateurs accessibles aux interactions sont des opérateurs sur lesquels une interaction peut être appliquée. De nombreux opérateurs sont accessibles aux interactions, comme par exemple l'opérateur Color dont les valeurs des enfants R G et B peuvent être modifiées par une interaction, les opérateurs de transformation, les opérateurs d'environnement, etc.

Les opérateurs d'exploration permettent de définir sur quel voisinage de l'élément courant une interaction est effectuée. L'élément courant dans le phénomène correspond alors à un élément courant dans le génotype. Les opérateurs présentés permettant l'exploration des données structurées, le voisinage est défini dans le génotype. Parmi ses opérateurs, nous trouvons l'opérateur Parent permettant d'accéder au parent de l'élément courant, l'opérateur Children permettant d'accéder à l'ensemble des éléments enfants de l'élément courant, l'opérateur Child permettant d'accéder à un enfant particulier dans la fratrie et possédant pour attribut la position de l'élément enfant auquel l'interaction est associée, les opérateurs PreviousSibling et NextSibling permettant d'accéder aux frères et sœurs de l'élément courant, mais aussi des opérateurs tels que CurrentGeneration, NextGeneration, PreviousGeneration permettant d'accéder à l'ensemble des éléments de même profondeur dans l'arbre d'exploration, ou encore AllSiblings et OtherSiblings permettant respectivement d'accéder à l'ensemble des enfants du parent de l'élément courant et à l'ensemble des enfants du parent de l'élément courant hormis l'élément courant. Ces opérateurs sont imbriquables, ce qui permet à tout moment d'accéder à tout élément de l'arbre d'exploration.

Les connecteurs d'interaction permettent d'établir un flux d'évènements allant d'un capteur à un opérateur interactif, avec une éventuelle prise en compte d'opérateurs d'exploration. L'exemple 4.3.4 étend l'exemple 4.1.1 en ajoutant des fonctions d'exploration du contenu présenté.

Exemple 4.3.4 *Nous souhaitons ajouter un ensemble de fonctions d'exploration à la métaphore de la tour de carrousels (voir exemple 4.1.1) :*

- *premier niveau d'exploration (les genres), un clic sur le nom d'un élément déplace le point de vue de manière à centrer cet élément à l'écran tandis qu'un glissement sur le cylindre représentant un élément entraîne la rotation de tous ses enfants autour de l'axe z ;*
- *deuxième niveau d'exploration (les réalisateurs), un glissement sur la sphère représentant un élément entraîne la rotation de tous ses enfants autour de l'axe z ;*
- *troisième niveau d'exploration (les films), un clic sur l'affiche représentant un film effectue un zoom sur cet objet et déplace le point de vue de manière à centrer l'affiche sur l'écran.*

Ses interactions sont décrites dans le phénomène de la manière suivante :

- *pour chaque élément du premier niveau d'exploration, un TouchSensor TS1 est associé au nom représentant le genre, un point-de-vue VP1 centré sur les représentants du genre courant est ajouté à la scène, et un CylinderSensor CS1 est associé au cylindre représentant le genre courant. Une interaction (TS1->VP1) permet d'activer le point de vue VP1 lorsque TS1 est actif, et une interaction (CS1->Rotation(Children)) permet de transférer la rotation appliquée à CS1 par l'utilisateur à un nœud Rotation contenant l'ensemble des enfants de l'élément courant ;*
- *pour chaque élément du second niveau d'exploration, un CylinderSensor CS2 est associé à la sphère représentant le réalisateur courant. Une interaction (CS2->Rotation(Children)) permet de transférer la rotation appliquée à CS2 par l'utilisateur à un nœud Rotation contenant l'ensemble des enfants de l'élément courant ;*
- *pour chaque élément du troisième niveau d'exploration, un TouchSensor TS3 est associé à l'affiche représentant le film, et un point-de-vue VP3 restituant l'affiche en gros plan est ajouté à la scène. Une interaction (TS3->VP3) permet d'activer le point de vue VP3 lorsque TS3 est actif.*

4.3.2.7 Préambule et compatibilités

Le préambule d'un phénomène permet de définir le domaine de validité du phénomène, que ce soit sur le plan des formats de sortie pris en compte, des dispositifs d'interaction et de restitution, ou encore des géotypes compatibles avec le phénomène. Ces domaines de validités peuvent être exclusifs (unique format de sortie géré par exemple) ou non (par exemple, phénomène optimisé pour la visualisation via un ordinateur et des dispositifs d'interaction de type clavier-souris).

Compatibilité avec les formats de sortie Le langage du phénomène est générique et propose de nombreuses fonctionnalités. Or chaque format 3D possède ses propres fonctionnalités. Dans certains cas, il peut être impossible de représenter certaines fonctionnalités de modélisation du phénomène dans un format 3D donné. Lorsque de telles fonctionnalités sont utilisées par le phénomène, ce dernier peut interdire la création d'une interface graphique 3D décrite dans les formats 3D non compatibles, ou avertir des fonctionnalités de la métaphore qui ne seront pas gérées par tel ou tel format 3D. De même, pour assurer l'utilisation optimale de la métaphore, le concepteur peut restreindre les formats 3D compatibles avec un phénomène à une liste prédéfinie.

Le phénomène peut également contenir un ensemble de modèles 3D représentés dans des formats 3D externes (comme par exemple un modèle en Collada ou en X3D). Dans ce cas, le concepteur peut restreindre le format de sortie au format 3D correspondant aux modèles insérés, ou à un ensemble de formats 3D pouvant importer des modèles 3D décrits dans ce format, ou encore préciser les formats utilisés pour les modèles 3D externes.

Dans les deux cas, le concepteur peut également indiquer les formats optimaux pour la visualisation de la métaphore, et proposer des alternatives à l'intérieur du phénomène à l'aide des opérateurs de condition. Par exemple, un phénomène pourra proposer un modèle 3D complexe représentant un bâtiment pour les formats de sortie compatibles avec Collada, et un cube pour les autres formats 3D.

Compatibilité avec les dispositifs d'interaction et de restitution Un phénomène peut posséder un certain nombre de caractéristiques liées aux dispositifs d'interaction et/ou de restitution utilisés lors de la visualisation de l'interface graphique 3D produite. Par exemple, la taille et la résolution d'un écran de téléphone mobile peuvent nécessiter l'utilisation de métaphore de présentation épurées, tandis que la restitution sur une table tactile peut impliquer des choix d'interaction spécifiques. Le préambule peut donc restreindre l'utilisation du phénomène à certains dispositifs d'interaction et/ou de restitution, ou encore préciser les dispositifs optimaux pour l'utilisation de l'interface graphique 3D produite.

Compatibilité structurelle avec le géotype Le phénomène décrit la métaphore visuelle d'interaction et d'exploration qui sera appliquée à un géotype afin de produire l'interface graphique finale. Ce géotype décrit la structure de l'information à visualiser sous forme de niveaux d'exploration. Le phénomène décrit alors la métaphore de présentation de l'information en associant à chaque niveau d'exploration un ensemble de règles. Cette mise en correspondance nécessite une compatibilité structurelle entre le géotype et le phénomène. En effet, si un géotype contient un nombre de niveaux d'exploration supérieur au phénomène qui lui est appliqué, seule une partie de l'information du géotype sera accessible à travers l'interface graphique produite. Dans ce cas, le concepteur de l'interface graphique pourra soit choisir un phénomène différent, soit combiner plusieurs phénomènes jusqu'à ce que le nombre de niveaux d'exploration pris en compte soit supérieur ou égal au nombre de niveaux d'exploration du géotype. Le phénomène doit donc indiquer le nombre maximal de niveaux d'exploration qu'il prend en compte afin d'assurer sa compatibilité avec un géotype. Il peut également préciser le nombre de niveaux d'exploration exact ou optimal que doit comporter le géotype auquel il est appliqué afin de garantir la création d'une interface graphique cohérente.

De plus, le géotype contient un ensemble d'attributs associé à chaque élément d'information. Ces attributs sont utilisés par le phénomène pour représenter l'élément d'information correspondant. Dans son préambule, le phénomène peut donc également préciser les attributs typés nécessaires ou optimaux pour la production de l'interface graphique, ainsi que les éventuels labels associés à ces attributs. Lorsqu'un géotype est compatible avec un phénomène en ce qui concerne le nombre de niveaux d'exploration et les attributs typés présents, mais que les labels ne correspondent pas, le générateur de géotype peut permettre de mettre en correspondance les labels du géotype avec les labels du phénomène.

Pour définir la compatibilité entre un génotype et un phénomène, il est donc nécessaire de connaître leur structure respective (nombre de niveaux d'exploration, attributs disponibles/nécessaires, ...). Or le génome représente la structure exploratoire générique de l'interface 3D, c'est-à-dire les invariants du génotype. Il permet donc de valider la compatibilité entre un génotype et un phénomène. Le préambule du phénomène peut donc contenir un génome augmenté d'opérateurs de restriction (précisant par exemples les attributs obligatoires ou optionnels). Afin de faciliter la vérification de compatibilité, le génome utilisé pour la production d'un génotype peut être intégré en préambule de ce génotype afin de préciser sa structure exploratoire et les attributs disponibles pour chaque élément. Le génome peut être inséré dans le préambule du phénomène et du génotype, ou associé aux phénomènes et génotypes compatibles à l'aide de son url. Plusieurs génomes peuvent être associés à un unique phénomène. Tout génome respectant la structure minimale (niveaux d'exploration, nombre d'éléments, typage des attributs, ...) définie par les génomes associés sont compatibles avec le phénomène. Lorsqu'un génotype ou un phénomène ne possède pas de génome associé, celui-ci peut être produit automatiquement, à l'aide d'une transformation XSLT par exemple.

4.3.2.8 Conclusion

Le phénomène permet donc la définition d'une métaphore de présentation de l'information par la mise en correspondance entre les niveaux d'explorations du génotype et un ensemble de règles. Ces règles permettent la description des invariants de l'interface graphique, et d'un ensemble de fonctions de modélisations permettant de présenter l'information structurée du génotype. L'imbrication des règles définit la hiérarchie de règles du phénomène, tandis que leurs opérateurs définissent la hiérarchie de graphes de scènes de l'interface graphique 3D, ainsi qu'une hiérarchie de transformation et un graphe de comportement pour chaque scène 3D composant l'interface graphique. Les restrictions et compatibilités du phénomène envers les formats 3D de description de scène, les interfaces d'acquisition et de restitution, ainsi que les génotypes, sont définies dans le préambule du phénomène.

Un sous-ensemble des concepts du phénomène est inspiré de concepts issus de formats de description de scènes 3D, et plus particulièrement X3D. Cependant, la sémantique des opérateurs présentés est différente, car il ne s'agit pas de proposer un format de description de scènes 3D, mais un format de modélisation permettant la production de scènes 3D. Par exemple, si la gestion d'évènements en X3D nécessite un certain nombre d'opérateurs (comme les interpolateurs), notre formalisme n'a pas besoin d'une gestion si fine des flux d'évènements. En effet, chaque format de description de scènes 3D compatible avec notre formalisme assure en interne la gestion des évènements. L'abstraction induite par la définition d'un formalisme pour la modélisation d'interfaces graphiques, et non d'un format de description de scènes 3D, simplifie la modélisation et la gestion des interaction.

4.3.3 De la feuille de style 3D à l'interface graphique 3D : le générateur de phénotype

Le générateur de phénotype permet d'appliquer un phénomène à un génotype afin de produire un phénotype, c'est à dire les représentations des scènes 3D de l'interface graphique. L'association du phénomène à un génotype permet la création d'un phénotype selon la transformation suivante :

- les invariants du phénomène sont dupliqués ;
- les opérateurs d'accès au génotype sont exécutés de manière à récupérer l'information correspondante dans le génotype ;
- les fonctions de productions sont exécutées.

Les scènes 3D visualisées par l'utilisateur final sont décrites dans un format 3D choisi par le concepteur de l'interface graphique, l'utilisateur, ou un programme tiers. La transformation du phénomène décrit dans un langage propre à notre formalisme vers un phénotype décrit dans un format 3D tiers peut être effectuée de deux manière :

- par la traduction vers le format désiré d'un phénotype générique résultant de l'application du phénomène au génotype ;
- par la traduction du phénomène en un générateur de phénotype mêlant les règles du phénomène aux opérateurs définis par le format 3D de sortie choisi.

Dans les deux cas, la généralité du module de présentation de l'information est assurée par la définition d'un langage générique de description de métaphores de présentation de l'information. Un même phé-

nome peut ainsi être utilisé pour produire des interfaces graphiques décrites dans différents formats 3D.

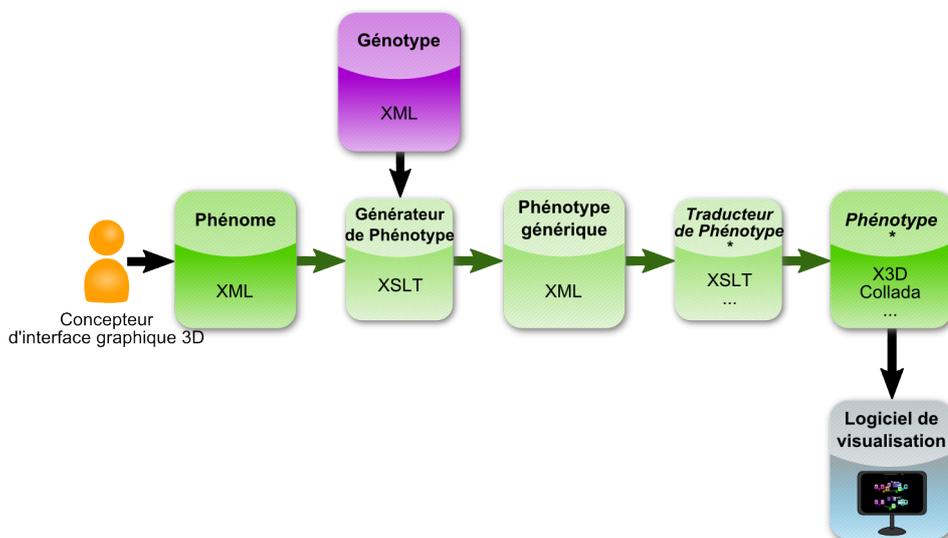


Figure 4.24 – Pipeline de production d'un phénotype avec traduction du phénotype.

Traduction du phénotype Dans le premier cas (voir figure 4.24), le générateur de phénotype applique directement le phénotype au génotype et génère un phénotype générique. Le langage de description de phénotypes génériques correspond à une restriction du langage de description de phénotype : le phénotype ne peut pas contenir de fonction de production, tous les éléments étant initialisés après l'application du phénotype au génotype. Le passage du phénotype au phénotype est alors équivalent à la transformation du génotype en un génotype, et est effectué de la manière suivante :

- l'élément racine `Phenome` est transformé en l'élément `Phenotype` ;
- le préambule est dupliqué tel quel ;
- les règles sont exécutées conformément à l'ordonnement spécifié pour les règles du phénotype :
 - les invariants des règles sont dupliqués ;
 - les accès au génotype et fonctions de production sont exécutés.

Un traducteur de phénotype est implémenté pour chaque format 3D de sortie disponible. Ce traducteur transforme l'ensemble des éléments du phénotype obtenu (hors préambule) vers le format 3D correspondant.

Le programme 4.6 présente un extrait du phénotype de la tour de carrousels (voir exemple 4.1.1) correspondant à la distribution linéaire des éléments du premier niveau d'exploration. Le programme 4.7 présente un extrait du phénotype générique obtenu suite à l'application de ce phénotype sur un génotype possédant cinq éléments au premier niveau d'exploration. Le programme 4.8 présente un extrait du traducteur de phénotype générique vers le format X3D, et le programme 4.9 présente un extrait du code obtenu après traduction du phénotype générique en un phénotype X3D.

Au moment de l'exécution, l'application du phénotype au génotype entraîne donc deux transformations successives pour permettre l'affichage de l'interface graphique : la création du phénotype générique, puis la traduction de ce phénotype en un phénotype décrit dans un format 3D choisi. La traduction du phénotype, que nous décrivons dans le paragraphe suivant, permet de diviser cette double transformation avec création d'un fichier intermédiaire (le phénotype générique) de manière à pré-calculer une partie de l'information produite.

Traduction du phénotype Dans le deuxième cas (voir figure 4.25), le phénotype est traduit en un générateur de phénotype spécifique au format 3D de sortie et incluant les opérateurs du phénotype.

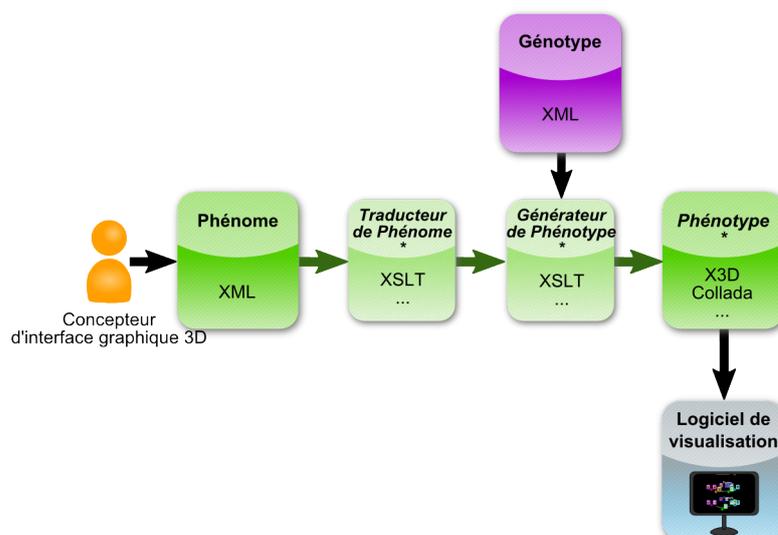


Figure 4.25 – Pipeline de production d'un phénotype avec traduction du phénotype générique.

Une fois appliqué au génotype, il génère un phénotype décrit dans le format 3D choisi. Le passage du phénotype au générateur de phénotype spécifique est effectué de la manière suivante :

- l'ensemble des fonctions d'accès au génotype et des fonctions de production du phénotype sont transformées dans le langage choisi pour le générateur de génotype (XSLT par exemple) ;
- l'ensemble des invariants du phénotype sont traduit conformément au format 3D de sortie.

Le programme 4.10 présente un extrait du générateur de phénotype permettant de produire des phénotypes au format X3D. L'application de ce générateur au génotype permet de produire directement le phénotype de l'exemple 4.1.1 (voir programme 4.9).

Le générateur de phénotype spécifique peut être pré-calculé, seule la transformation du génotype par ce générateur est effectuée au moment de l'exécution. Cependant, l'implémentation du traducteur de phénotype associé à chaque format 3D de sortie est plus complexe que l'implémentation du traducteur de phénotype car elle nécessite de prendre en compte l'ensemble des fonctions de production et d'accès au génotype proposés par le langage de description de phénotype.

Nous avons implémenté la deuxième solution pour la production de phénotypes X3D, c'est-à-dire un traducteur de phénotype décrit en XSLT permettant de traduire un phénotype (au format XML) en un générateur de phénotype X3D décrit en XSLT. Les fonctions de production et d'accès au génotype du phénotype d'origine sont transformés en un ensemble d'opérateurs XSLT, tandis que l'ensemble des invariants du phénotype sont traduits en opérateurs X3D.

4.3.4 Conclusion

L'adaptation de la métaphore de présentation de l'information au domaine applicatif des données à visualiser, au contexte et à l'utilisateur permet d'augmenter l'efficacité et l'attrait des interfaces graphiques 3D. Au moyen de la définition du module de présentation de l'information, nous proposons un formalisme permettant au concepteur d'une interface graphique 3D de définir des métaphores visuelles incluant des interactions et des modes d'exploration variés pour la visualisation interactive de grands ensembles de données. L'ensemble de scènes 3D, ou phénotype, résultant de l'application d'un phénotype à un génotype possède une géométrie conforme à la métaphore de présentation définie dans le phénotype et une topologie conforme à la structuration des données décrite dans le génotype. Étant basé sur XML, le module de présentation de l'information est extensible. Il permet la définition de métaphores de présentation de l'information variées et modulaires, ainsi que la production d'interfaces graphiques décrites dans différents formats de description de scènes 3D.

Programme 4.6 Extrait du phénomène correspondant à l'exemple 4.1.1 : définition de la distribution linéaire des éléments du premier niveau d'exploration. Les informations extraites de l'ensemble de données sont indiquées en violet et les invariants de présentation en vert.

```

<ForEachElem>
  <Translation>
    <X><Num val="0"></X>
    <Y><Num val="0"></Y>
    <Z>
      <Mult>
        <Operand>
          <Index>
        </Operand>
        <Operand>
          <Num val="10">
        </Operand>
      </Mult>
    </Z>
    <!-- éléments à distribuer -->
  </Translation>
</ForEachElem>

```

Programme 4.7 Extrait du phénotype générique produit à partir de l'extrait du phénomène correspondant à l'exemple 4.1.1 et présenté dans le programme 4.6. Les informations issues des fonctions de production sont indiqués en violet et les invariants de présentation en vert.

```

<Translation><!-- index = 1 -->
  <X><Num val="0"></X>
  <Y><Num val="0"></Y>
  <Z><Num val="10"></Z>
  <!-- éléments à distribuer -->
</Translation>
<Translation><!-- index = 2 -->
  <X><Num val="0"></X>
  <Y><Num val="0"></Y>
  <Z><Num val="20"></Z>
  <!-- éléments à distribuer -->
</Translation>
<!-- ... -->
<Translation><!-- index = 5 -->
  <X><Num val="0"></X>
  <Y><Num val="0"></Y>
  <Z><Num val="50"></Z>
  <!-- éléments à distribuer -->
</Translation>

```

4.4 Conclusion

De l'intégration native de l'accélération matérielle 3D aux navigateurs Web à l'ajout de fonctionnalités 3D dans les formats de conception d'applications Web (comme Flash par exemple), les technologies Web sont en train d'évoluer vers la généralisation des contenus 3D sur le Web. Cependant, l'absence de l'intégration, dans les processus de création de sites Web, des nouvelles fonctionnalités et possibilités offertes par les représentations 3D limite le développement du Web 3D. Or de nombreux sites Web permettent la visualisation d'information issues de systèmes de gestion de données,

Programme 4.8 Extrait du fichier XSLT permettant de transformer un phénotype générique en un phénotype X3D.

```
<xsl:template match="Translation">
  <xsl:element name="Transform">
    <xsl:attribute name="translation">
      <xsl:value-of select="./X/Num/@val"/ >
      <xsl:text> </xsl:text>
      <xsl:value-of select="./Y/Num/@val"/ >
      <xsl:text> </xsl:text>
      <xsl:value-of select="./Z/Num/@val"/ >
    </xsl:attribute>
  </xsl:element>
</xsl:template>
```

Programme 4.9 Extrait du phénotype X3D produit à partir de l'extrait du phénomène correspondant à l'exemple 4.1.1 et présenté dans le programme 4.6. Les informations issues des fonctions de production sont indiqués en violet et les invariants de présentation en vert.

```
<Transform translation="0 0 10"><!-- index = 1 -->
  <!-- éléments à distribuer -->
</Transform>
<Transform translation="0 0 20"><!-- index = 2 -->
  <!-- éléments à distribuer -->
</Transform>
<!-- ... -->
<Transform translation="0 0 50"><!-- index = 5 -->
  <!-- éléments à distribuer -->
</Transform>
```

Programme 4.10 Extrait du fichier XSLT correspondant au phénomène X3D obtenu après transformation de l'extrait du phénomène correspondant à l'exemple 4.1.1 et présenté dans le programme 4.6. Les informations extraites de l'ensemble de données sont indiquées en violet et les invariants de présentation en vert.

```
<xsl:for-each select="./Elem">
  <xsl:element name="Transform">
    <xsl:attribute name="translation">
      <xsl:text>0 0 </xsl:text>
      <xsl:value-of select="position() * 10"/>
    </xsl:attribute>
    <!-- éléments à distribuer -->
  </xsl:element>
</xsl:for-each>
```

comme par exemple les services de e-commerce, les albums photos en ligne ou encore les portails Web. Nous proposons donc un formalisme permettant de décrire des interfaces graphiques 3D pour le Web au moyen de la définition de métaphores efficaces et attractives pour la visualisation de grands ensembles de données.

Le formalisme présenté s'inspire des usages actuels dans la conception de sites Web, d'une part par la séparation nette entre la gestion du contenu et la présentation des données, et d'autre part en se basant sur les technologies Web par la définition de langages formels implémentés en XML. La définition de langages formels génériques permet la présentation d'informations issues de tous types de

systèmes de gestion de données, et permet d'être indépendant des formats de description de scènes 3D existants, laissant le concepteur de l'application choisir le format de description de scènes 3D le plus adapté à ses besoins et à ses contraintes. De plus, cette généralité assure la pérennité de la technologie proposée. Notre formalisme est basé sur deux modules : un module de structuration de l'information permettant de définir, sous la forme d'un arbre d'exploration, les contraintes sémantiques de l'interface graphique 3D ainsi que la façon dont l'utilisateur explore les données, et un module de présentation de l'information permettant de définir les contraintes plastiques de l'interface graphique 3D à travers la définition d'une métaphore visuelle permettant d'interagir avec le contenu et d'explorer les données. Chacun de ces modules est à son tour modulaire, de par la définition de niveaux d'exploration dans le cas du génome, et de par la définition de règles de présentation dans le cas du phénomène. La modularité de ce formalisme permet une grande réutilisabilité, que ce soit sur le plan des métaphores structurelles et visuelles qui peuvent être combinées de différentes manières (un unique génome pour plusieurs phénomènes, et vice-versa), ou sur le plan des niveaux d'exploration et des règles existants pouvant être réutilisés pour définir respectivement de nouvelles métaphores structurelles ou visuelles.

Enfin, le pipeline de création permet de définir des interfaces graphiques variées, que ce soit du point de vue du contenu ou de la mise en page. Le concepteur d'une interface graphique 3D peut ainsi définir des métaphores allant des scènes 3D immersives multi-utilisateurs aux scènes 3D non immersives, et définir des métaphores reflétant ou non la sémantique des données à visualiser. De plus, nous proposons un ensemble de fonctionnalités permettant de définir les contraintes de compatibilité : compatibilité entre métaphores de structuration et métaphores de présentation, et compatibilité avec des dispositifs d'interaction et de restitution.

Si le formalisme que nous proposons est flexible, son implémentation en XML rend la définition de génomes et de phénomènes difficile. Afin de faciliter la création de métaphores de visualisation de l'information, il est donc nécessaire d'intégrer notre formalisme dans des outils auteurs pour la description de métaphores de structuration et de présentation de l'information. Ces outils peuvent proposer différents niveaux de définition en fonction de l'expertise du créateur de l'interface graphique (voir section 4.3.1.3).

Chapitre 5

Résultats et analyses

5.1 Visualisation de données multimédia

Dans le chapitre précédent, nous proposons une solution complète permettant la définition de métaphores de visualisation et produisant un ensemble de scènes 3D à partir d'informations issues de grands ensembles de données. Cette définition se fait en deux étapes :

- la description de la structure exploratoire des données effectuée à l'aide du module de structuration de l'information.
- la description de la métaphore visuelle interactive à l'aide du module de présentation de l'information.

Afin d'illustrer les avantages de notre concept, nous présentons dans cette section divers cas d'utilisation de notre formalisme. Parmi l'ensemble des domaines d'application visés par notre technologie, nous avons choisi la visualisation de bases de données multimédia, et plus précisément deux bases de données de films et une base de données d'albums musicaux. Deux bases de données ont été créées pour les besoins du démonstrateur et comprennent respectivement 21 films et 22 albums. La seconde base de données de films correspond à la base de vidéos à la demande (VoD) d'Orange et contient plus de 3000 vidéos. Les exemples présentés ont été choisis afin de démontrer d'une part la possibilité d'appliquer une métaphore de présentation de l'information à différents types de contenu et inversement d'appliquer plusieurs métaphores de présentation de l'information à une unique structuration de données, et d'autre part de démontrer les capacités de personnalisation et de réutilisation de modules existants.

Le format de description de scènes 3D choisi dans ces exemples comme format de sortie est X3D. Les métaphores de présentation proposées sont donc contraintes par les fonctionnalités offertes par ce format.

En section 5.1.1, nous présentons le cas d'un utilisateur possédant deux bases de données multimédia représentant respectivement les albums musicaux et les vidéos de sa médiathèque. Cet utilisateur souhaitant partager ces données en ligne à l'aide d'une interface graphique 3D, il utilise notre solution pour créer une métaphore générique appelée *tour de carrousels*. En section 5.1.2, nous présentons une métaphore immersive dédiée au monde du cinéma respectant l'identité visuelle d'Orange, ainsi que divers cas de réutilisation de la structure exploratoire des données et de la métaphore visuelle proposée.

5.1.1 Interface 3D générique : la tour de carrousels

Dans cette section, nous revenons sur l'exemple 4.1.1 afin de démontrer les capacités de notre formalisme en terme de réutilisation d'un génome et d'un phénomène pour la visualisation de données de domaines différents (des films dans un cas et des albums musicaux dans l'autre).

Le concepteur A d'un site Web souhaite proposer à ses visiteurs une interface graphique 3D pour présenter les films qu'il possède dans sa médiathèque. La base de données de ses vidéos a déjà

été créée à partir d'un logiciel tiers. A l'aide du module de structuration de l'information que nous proposons, il définit la structuration hiérarchique de son espace cinématographique : les visiteurs auront accès aux genres de ses vidéos, puis pour chaque genre aux réalisateurs et enfin aux noms des films de sa médiathèque. Il ajoute à sa requête les critères de présentation suivants : les champs *nom*, *couleur* et *url de l'affiche* respectivement associés aux éléments de type genre, réalisateur et film. Le génome A correspondant est présenté dans la table 5.1. Le module de gestion de données produit

Niveau d'exploration	Critères hiérarchiques	Critères de représentation
1	Genre	nom
2	Réalisateur	couleur
3	Nom du film	url de l'affiche du film

Table 5.1 – Génome A : structuration d'une base de données de films

automatiquement un arbre d'exploration, que nous nommerons génotype A, à partir de la requête représentée par le génome A appliquée à la base de données représentant la médiathèque. L'auteur A ayant défini la structure exploratoire de son environnement, il doit à présent définir la métaphore visuelle interactive lui permettant de présenter les données obtenues.

L'auteur A ne souhaite pas faire appel aux compétences d'un infographiste 3D et choisi donc d'utiliser uniquement les primitives géométriques disponibles pour la définition de la métaphore visuelle. Il choisit de représenter l'ensemble des trois niveaux d'exploration à l'intérieur d'une unique scène 3D. Les éléments du premier niveau d'exploration (les genres) seront représentés par des cylindres distribués linéairement selon l'axe Y pour former une tour. Les éléments du deuxième niveau d'exploration (les réalisateurs) seront représentés par des sphères colorées distribuées circulairement autour du cylindre parent. Enfin, les éléments du troisième niveau d'exploration (les films) seront représentés par leur affiche et distribués circulairement autour de la sphère parente. En ce qui concerne les interactions, l'utilisateur pourra faire tourner les enfants d'un éléments de niveau 1 ou 2 en effectuant un glissement sur l'élément parent (cylindre ou sphère). Le phénomène A correspondant est présenté dans la table 5.2. En combinant le phénomène A avec l'arbre d'exploration obtenu à l'aide du module de structuration de

Niv. d'expl.	Distribution des éléments	Éléments	Interaction/Exploration
1 : tour	Linéaire le long de l'axe Y	cylindre blanc nom de l'élément courant	capteur cylindrique : rotation sur les enfants clic : rejoindre le point-de-vue associé à l'élément courant
2 : carrousel 1	Circulaire autour de l'élément parent	sphère colorée à l'aide de l'attribut <i>couleur</i> de l'élément courant	capteur cylindrique : rotation sur les enfants
3 : carrousel 2	Circulaire autour de l'élément parent	rectangle texturé à l'aide de l'attribut <i>url</i> de l'élément courant	clic : rejoindre le point-de-vue associé à l'élément courant

Table 5.2 – Phénomène A : la tour de carrousels

l'information, le module de présentation de l'information produit une scène virtuelle 3D interactive dont un visuel est présenté en figure 5.1(a).

L'auteur A souhaite à présent proposer une interface 3D pour présenter les CDs qu'il possède dans sa médiathèque. Il complète donc sa base de données multimédia à l'aide du logiciel tiers utilisé pour ses vidéos. Souhaitant conserver la métaphore visuelle et d'interaction qu'il a définie pour ses vidéos, l'auteur A modifie le génome A pour l'adapter au sous-ensemble musical de sa base de données en définissant les critères hiérarchiques comme suit : les genres puis les artistes puis les albums. Le génome A' correspondant est présenté dans la table 5.3. La combinaison du génotype A', généré en appliquant le génome A' à la base de données multimédia, avec le phénomène A génère une nouvelle scène virtuelle 3D interactive dont un visuel est présenté en figure 5.1(b).

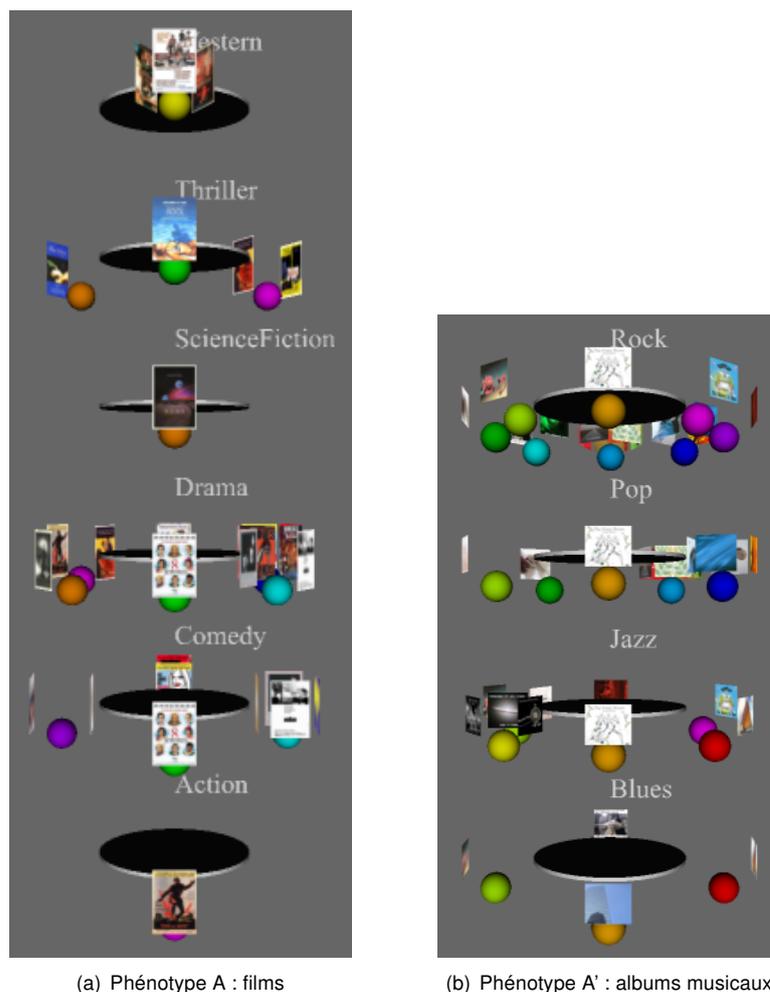


Figure 5.1 – Une interface 3D générique : la tour de carrousel

Dans cet exemple, l'auteur A a défini deux génomes A et A' pour naviguer respectivement dans un ensemble de film et d'albums musicaux. Il a appliqué le phénomène A à chacun des des génotypes obtenus. Il a ainsi obtenu deux scènes virtuelles 3D interactives (phénotypes A et A') représentant chacune une tour de carrousel.

5.1.2 Interfaces 3D immersives

Dans cette section, nous présentons un ensemble d'exemples permettant de naviguer dans l'information à partir d'une interface 3D immersive dont la sémantique est corrélée à celle des données visualisées. Cette section étant entièrement dédiée à la visualisation de bases de données de films, nous avons choisi l'univers des salles de cinéma. Le but de cette section est multiple. Il s'agit de démontrer les possibilités offertes par notre technologie :

- en terme de gestion de scènes multiples et de découverte de l'information ;
- pour la création d'interfaces immersives ;
- concernant l'adaptation d'une métaphore visuelle :
 - à un domaine précis (ici l'univers cinématographique) ;
 - à des situations s'inspirant de la vie réelle mais conservant les avantages de la réalité virtuelle.

Nous présentons tout d'abord une hypothétique interfaces 3D proposée par Orange pour permettre à ses clients de naviguer dans sa base de données de VoD. Puis nous présentons un cas de réutilisation de la métaphore visuelle interactive proposée pour une page personnelle. Enfin, nous présentons un cas de modification de la métaphore visuelle pour la personnalisation d'une page personnelle.

Niveau d'exploration	Critères hiérarchiques	Critères de représentation
1	Genre	nom
2	Groupe / Artiste	couleur
3	Titre d'album	url de la pochette de l'album

Table 5.3 – Génome A' : structuration d'une base de données d'albums musicaux

5.1.2.1 L'univers Orange

Dans cette section, nous proposons un cas d'étude hypothétique dans lequel Orange proposerait une interface 3D immersive pour la navigation dans sa base de données de VoD. Pour la réalisation de l'interface, Orange fait appel à trois professionnels : un concepteur de site Web, un infographiste 3D et un gestionnaire de contenu.



(a) Niveau d'exploration 1 : les genres.



(b) Niveau d'exploration 2 : les réalisateurs.



(c) Niveau d'exploration 3 : les films.

Figure 5.2 – Feuille de style Orange appliquée à une base de données de films.

La conception d'un univers cinématographique Le concepteur du site Web souhaite définir une interface permettant à l'utilisateur final de réutiliser des connaissances et des modes d'interactions inspirés du monde réel. Le domaine ciblé par Orange pour cette interface étant celui du cinéma, il propose une métaphore visuelle s'inspirant des complexes cinématographiques : l'utilisateur final commencera par choisir un cinéma. Une fois dans le hall de celui-ci, il choisira une porte menant à une salle de projection. Enfin, dans la salle de projection, il aura accès à un ensemble de films disponibles. Chaque scène sera composée d'un ensemble d'éléments *statiques*, c'est-à-dire ne dépendant pas de la base

de données, et d'un ensemble d'éléments *dynamiques*, c'est-à-dire dont le nombre, la position, la couleur, ... pourront dépendre de données de présentation issues du génome. La métaphore proposée est donc divisée en trois niveaux de scènes virtuelles interactives dont les caractéristiques sont détaillées ci-dessous.

La première scène visitée par l'utilisateur final représente une place (voir figure 5.2(a)) proposant un certain nombre de services : point info permettant d'obtenir de l'aide (navigation, interactions, ...), espaces de discussion, visionnage de bandes-annonces, publicités. Autour de cette place, un certain nombre de cinémas sont distribués circulairement, chaque cinéma représentant un élément du premier niveau d'exploration.

Une fois le cinéma choisi, l'utilisateur final est téléporté dans le hall d'entrée correspondant (voir figure 5.2(b)). Celui-ci propose également un point info, ainsi qu'un ensemble de publicités 2D et 3D, un espace restauration permettant de commander en ligne et un espace jeux. Sur le mur du fond du hall, un certain nombre de portes sont distribuées linéairement, chaque porte représentant un élément du deuxième niveau d'exploration.

Enfin, une fois la porte choisie, l'utilisateur final est téléporté dans la salle de projection correspondante (voir figure 5.2(c)). Celle-ci propose également un point info. Les affiches correspondant aux films du troisième niveau d'exploration sont linéairement distribuées dans un espace situé sous l'écran de projection. Un clic permet alors de lancer sur l'écran de projection la bande-annonce du film choisi.

L'exploration des éléments d'un même niveau s'effectue à l'aide d'un glissement sur les éléments ayant pour effet soit de faire tourner les éléments autour de l'axe vertical de la scène pour le premier niveau d'exploration (les cinémas), soit de faire glisser les éléments le long de l'axe horizontal de la scène pour les éléments de niveau d'exploration deux et trois (portes et affiches). Le retour au niveau d'exploration précédant (hall ou place extérieure) s'effectue à l'aide d'un clic sur la porte de la salle en cours de visualisation (la salle de projection ou le hall de cinéma).

La création d'objets n-D La partie graphique de la métaphore est assurée par l'infographiste 3D, à l'aide de logiciels tiers. Il doit respecter l'identité visuelle d'Orange tout en suivant le cahier des charges défini par le concepteur du site Web. Pour chaque niveau, l'infographiste devra fournir un certain nombre de scènes comprenant un ou plusieurs éléments appartenant aux catégories suivantes :

- les éléments statiques comme par exemple les points info, le point restauration, les espaces publicitaires ;
- les éléments dynamiques ou prototypes dont un certain nombre de paramètres (couleur, texture, ...) seront déterminées par les attributs ou informations contextuelles (nombre d'éléments d'un niveau d'exploration, ...) issus du génotype ;
- les éléments intelligents dont une partie de l'interaction et/ou de l'animation (ouverture d'une porte, ...) est directement contenue dans le fichier ;
- les éléments d'aide à la navigation et à l'exploration des données (points-de-vue, niveau d'exploration suivant/précédent, ...) (voir figure 5.3).

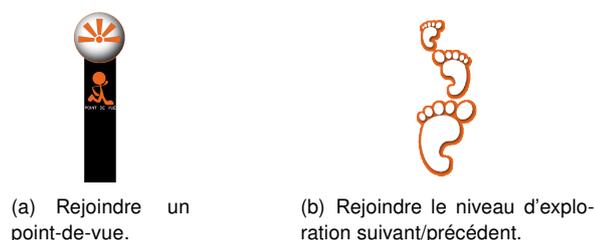


Figure 5.3 – Univers Orange - Aide à la navigation et à l'exploration de données.

Certains éléments peuvent appartenir à plusieurs catégories à la fois. Chaque niveau d'exploration est décomposé en une scène principale représentant les éléments statiques de la métaphore (le décor) et un ensemble de scènes secondaires permettant de représenter les éléments de la base de données. Ces scènes sont décrites ci-dessous.

La scène principale du premier niveau d'exploration représente la place (figure 5.4(a)). Sa forme générale est circulaire et elle possède un élément d'aide à la navigation par point d'intérêt (point dis-

cussion, ...). Un prototype de bâtiment de cinéma (figure 5.4(b)) est disponible dans un fichier séparé. Il possède un champ *nom* permettant d'insérer le nom de l'élément qu'il représente, ainsi que deux champs de *texture* permettant de lui attribuer les textures d'affiches de films présents dans la base de données (par exemple, prochaines sorties de la catégorie correspondant à l'élément représenté). Enfin, il possède un champ *couleur* permettant de distinguer les cinémas pour chaque élément de la base de données. Le bâtiment est également un objet intelligent, ses portes s'ouvrant lorsqu'elles entrent dans le champ de la caméra représentant l'utilisateur final. Un exemple de scène ainsi créée est présenté en figure 5.4(c).



(a) Scène principale de l'univers Orange : la place centrale.



(b) Prototype de cinéma.



(c) Scène obtenue pour la visualisation du catalogue de vidéo à la demande.

Figure 5.4 – Univers Orange - Premier niveau d'exploration.

La scène principale du deuxième niveau d'exploration représente le hall du cinéma choisi (voir figure 5.5(a)). Afin de contextualiser l'expérience de l'utilisateur, il s'agit d'un prototype comprenant un champ *nom* et un champ *couleur* reprenant les mêmes attributs que ceux du bâtiment extérieur (ou élément parent). Le prototype de porte (voir figure 5.5(b)) représentant les éléments du deuxième niveau d'exploration est contenu dans un fichier séparé contenant un champ *nom* représentant cette fois le nom de l'élément courant dans le génotype (et non celui de son parent). La porte s'ouvre lorsque l'utilisateur se trouve à proximité. Un exemple de résultat obtenu lors de la création de la scène finale est présenté figure 5.5(c).

Enfin, la scène principale du troisième niveau d'exploration représente la salle de projection (voir figure 5.6(a)). Elle comprend également un champ *couleur* faisant référence à la couleur du parent de niveau 1. Une scène séparée représente le prototype d'un cadre (voir figure 5.6(b)) dont la texture est extraite de la base de données. Une autre scène représente un bouton *lecture* et sera positionnée légèrement en avant du cadre pour permettre de lancer la bande annonce correspondant au film si celle-ci est disponible. Un exemple de résultat obtenu lors de la création de la scène finale est présenté figure 5.6(c).

Présentation des données L'association de la métaphore visuelle définie par le concepteur de l'interface graphique et des modèles créés par l'infographiste permet au concepteur de l'interface 3D de définir le phénomène de l'univers cinématographique d'Orange (que nous nommerons phénomène B). Afin de spécifier les contraintes de ce phénomène, le concepteur de l'interface graphique intègre un génome



(a) Scène principale de l'univers Orange : le hall de cinéma.



(b) Prototype de porte.



(c) Scène obtenue pour la visualisation du catalogue de vidéo à la demande.

Figure 5.5 – Univers Orange - Deuxième niveau d'exploration.



(a) Scène principale de l'univers Orange : la salle de projection.



(b) Prototype de panneau.



(c) Scène obtenue pour la visualisation du catalogue de vidéo à la demande.

Figure 5.6 – Univers Orange - Troisième niveau d'exploration.

type en préambule du phénomène (voir table 5.4), et indique les caractéristiques suivantes concernant les modèles 3D intégrés au phénomène B :

- le modèle paramétrique permettant la représentation des éléments du premier niveau d'exploration (le cinéma) contient un champ "nom", un champ "couleur" ;
- le modèle paramétrique permettant la représentation du deuxième niveau d'exploration (le hall de cinéma) contient un champ "nom" et un champ "couleur" ;
- le modèle paramétrique permettant la représentation des éléments du deuxième niveau d'exploration (la porte) contient un champ "nom" ;
- le modèle paramétrique permettant la représentation du troisième niveau d'exploration (la salle de projection) contient un champ "nom" et un champ "couleur" ;
- le modèle paramétrique permettant la représentation des éléments du troisième niveau d'exploration (le panneau) contient un champ "url d'image" ;
- l'ensemble des modèles utilisés sont décrits dans le format X3D.

Niveau d'exploration	Critères hiérarchiques	Critères de représentation
1	Niv1	couleur, nom
2	Niv2	nom
3	Niv3	url d'image url de vidéo

Table 5.4 – Extrait du préambule du phénomène B : compatibilité avec les génotypes.

Structuration des données la métaphore visuelle proposée par le concepteur du site Web comporte trois niveaux d'exploration. Le gestionnaire de contenu propose donc une structuration hiérarchique sur trois niveaux : les *genres*, puis les *années* et enfin les *films*. Le génome B ainsi défini, et conforme au préambule du phénomène B, est présenté dans la table 5.5.

Niveau d'exploration	Critères hiérarchiques	Critères de représentation
1	Genre	couleur, nom
2	Année de sortie	nom
3	Nom du film	url de l'affiche du film url de la bande-annonce du film

Table 5.5 – Génome B : structure exploratoire de la base de données de vidéos à la demande d'Orange.

Génération de l'interface graphique 3D L'application du phénomène B, incluant l'ensemble d'objets 3D présentés, sur le génotype B obtenu à partir du génome B produit un ensemble de scènes 3D interactives :

- une scène 3D pour le premier niveau d'exploration (la place centrale entourée de cinémas) ;
- une scène 3D par élément du premier niveau d'exploration (les halls de cinéma de chaque genre) ;
- une scène 3D par élément du deuxième niveau d'exploration (les salles de projection pour chaque année correspondant à un genre).

Nous obtenons bien une interface graphique 3D immersive, respectant l'identité visuelle d'Orange et à l'intérieur de laquelle les utilisateurs naviguent de façon similaire au monde réel.

Métaphore Orange et pages personnelles Souhaitant proposer à ses clients de créer leur propre site Web 3D, Orange met le phénomène B à disposition des clients souhaitant présenter une base de données de films (films de leur médiathèque, films vus, ...). L'auteur A étant client Orange, il peut à présent proposer deux visualisations de sa base de données de films : la tour de carrousels et l'interface immersive d'Orange. Il modifie légèrement le génome A pour ajouter une couleur à chaque genre et un nom à chaque réalisateur. Un extrait des scènes générées par la combinaison du génome A modifié avec le phénomène B fourni par d'Orange est présentée en figure 5.2. Le phénomène B n'utilisant

pas de couleur issue du génotype et associée aux éléments de niveau deux (les réalisateurs), seul un sous-ensemble des informations fournies par le génotype A sont utilisées pour la génération de l'interface graphique 3D. Afin de répondre aux attentes de ses visiteurs, l'auteur A propose également une deuxième structuration de ses vidéos en intervertissant les modules 1 et 2 du génotype A. L'exploration des données se fait alors de la manière suivante : les *réalisateurs*, puis les *genres* et enfin les *films*. Le génotype ainsi obtenu est compatible avec les phénomènes A et B car il possède tous les attributs nécessaires pour la représentation de chaque niveau d'exploration.

5.1.2.2 Un univers en papier

L'auteur A souhaite personnaliser le phénomène proposée par Orange en changeant uniquement le style graphique, c'est-à-dire les objets utilisés, et en conservant les mode de navigation et de disposition des éléments. Sur le Web, il trouve une base de modèles 3D inspirés de maquettes en cartons comprenant entre autre un cinéma complet (bâtiment extérieur, hall, salle de projection), une porte et un panneau (figure 5.7). Il associe les modèles trouvés à la feuille de style proposée par Orange et obtient la visualisation présentée en figure 5.8.



Figure 5.7 – Modèles 3D papier.

5.1.2.3 Conclusion

Dans cet exemple, nous avons démontré les capacités de notre formalisme pour la description d'interfaces graphiques 3D immersives dont la sémantique est corrélée à la sémantique des données à visualiser. De plus, nous avons démontré les capacités de réutilisation d'un phénomène avec des données gérées par des systèmes de gestion de données différents, ainsi que la possibilité de modifier les objets 3D statiques d'un phénomène pour obtenir un univers visuel très différent.

5.2 Analyse

5.2.1 Métaphores de structuration et de présentation de l'information

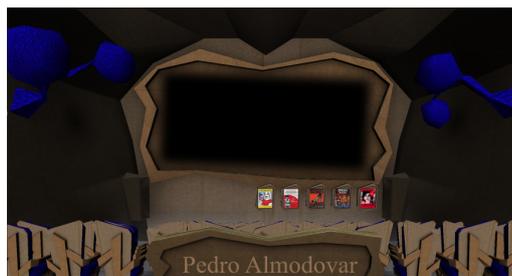
D'après [75], la fractalité d'un environnement, c'est-à-dire la présence d'information à différents niveaux d'abstraction, facilite la navigation (voir section 3.2). Le créateur d'une interface graphique doit donc pouvoir structurer l'information présentée en différents niveaux d'abstraction définissant la façon dont l'utilisateur accède aux données. Le module de structuration de l'information permet au gestionnaire de données de définir ces niveaux d'abstraction, que nous nommons niveaux d'exploration, à travers la description du génotype. Une fois appliqué à l'ensemble de données à visualiser, le génotype permet la génération d'un génotype, ou arbre d'exploration augmenté, représentant la structure exploratoire des données. Le gestionnaire de données peut donc, à travers la description du génotype, définir les paradigmes de navigation dans l'espace sémantique des données, et ce de manière adaptée au contexte, c'est-à-dire au domaine des données à visualiser, aux utilisateurs finaux cibles, etc. Afin d'améliorer l'accès à l'information ainsi structurée, le créateur de l'interface graphique doit également pouvoir définir un ensemble de paradigmes de navigation dans l'espace 3D permettant d'explorer



(a) Niveau d'exploration 1 : les genres



(b) Niveau d'exploration 2 : les réalisateurs



(c) Niveau d'exploration 3 : les films

Figure 5.8 – Métaphore de présentation Orange avec un style graphique papier appliqué à la base de données de films de l'auteur A.

l'information à chaque niveau d'exploration, et entre les niveaux d'exploration. Le module de présentation de l'information permet au concepteur de l'interface graphique 3D de définir ces paradigmes d'exploration à travers la description du phénomène. Ainsi, la métaphore de la tour de carrousels définie par l'auteur A (voir section, 5.1.1) permet à l'utilisateur final de naviguer de manière linéaire et verticale entre les éléments du premier niveau d'exploration, et de manière circulaire et horizontale pour les éléments des niveaux d'exploration 2 et 3. L'ensemble des éléments d'information étant présentés dans la même scène 3D, le passage d'un niveau d'exploration à l'autre est directement corrélé au déplacements internes aux niveaux d'exploration. Au contraire, le paradigme de navigation proposé par la métaphore de l'univers Orange (voir section 5.1.2.1) définit un changement de scène 3D à chaque changement de niveau d'exploration.

Notre formalisme permet donc, à travers la définition du génome et du phénomène de l'interface graphique 3D, de décrire les paradigmes d'exploration de l'information, que ce soit dans l'espace sémantique ou dans l'espace 3D. La description de ces paradigmes permet au créateur de l'interface d'améliorer l'efficacité de son interface graphique 3D et de faciliter la navigation et l'accès à l'information.

Si la structuration de l'information joue un rôle primordial lors de la définition d'une interface graphique 3D, la définition de la métaphore de présentation de l'information est tout aussi cruciale (voir section 3.3). Le créateur de l'interface graphique doit donc pouvoir définir une métaphore de présentation de l'information adaptée au contexte, c'est-à-dire au domaine applicatif des données, aux utilisateurs finaux cibles, aux dispositifs d'interaction et de restitution cibles, etc. Les outils nécessaires à la

définition de cette métaphore sont fournis par le langage de description de phénomènes et les attributs de représentations associés aux éléments d'information décrits dans le génotype. En effet, à travers la définition d'un phénomène, le concepteur de l'interface graphique peut spécifier l'aspect visuel et les modes d'interaction, comme par exemple le type et la taille du contenu permettant de représenter les éléments d'information (modèles 3D, couleur, ...), la façon dont ces objets sont distribués dans l'espace 3D et dans différentes scènes 3D, ou encore un certain nombre d'éléments de décor ou liés à l'interface graphique.

Dans la section suivante, nous analysons le caractère modulaire de notre formalisme, permettant une meilleure personnalisation et une meilleure adaptabilité des métaphores de visualisation aux contraintes définies par le créateur d'une interface graphique 3D.

5.2.2 Modularité

Dans ce document, nous proposons un formalisme pour la définition de métaphores attractives et efficaces pour la visualisation d'information. Parmi les contraintes que nous avons définies, notre formalisme doit être accessible à tout concepteur d'interface graphique, spécialiste ou non de la programmation, il doit permettre une forte réutilisabilité des métaphores définies, et doit être compatible avec un maximum de formats de données en entrée et en sortie du système. Pour ce faire, nous avons défini un formalisme modulaire, que ce soit sur le plan du pipeline de création de métaphores de visualisation, sur le plan des langages formels de description des métaphores de structuration et de présentation de l'information, ou encore de l'implémentation proposée de notre formalisme.

Pipeline de création La séparation forte entre la structuration et la présentation des données, définies respectivement par le génome et le phénomène de l'interface graphique 3D, assure la modularité du pipeline de définition des métaphores de visualisation interactive. Comme nous l'avons vu en section 5.1, différents génotypes, représentant des données de domaines différents, peuvent être associés à un même phénomène (la tour de carrousels) pour générer des interfaces graphiques 3D similaires. De même, un unique génotype peut être associé à différents phénomènes pour la génération d'interfaces graphiques 3D différentes. C'est le cas par exemple du génotype A modifié présenté en section 5.1.2.2 permettant de générer trois types d'interfaces graphiques : une tour de carrousels, un univers cinématographique respectant l'identité visuelle d'Orange et un univers cinématographique en papier.

Langages formels Les langages de description de génomes et de phénomènes sont modulaires. Le premier permet la définition de requêtes sous forme de modules, chaque module correspondant à un niveau d'exploration et décrivant les attributs labellisés associés à chaque élément de ce niveau d'exploration. Ainsi, l'auteur A intervertit les modules représentant les niveaux d'exploration 1 et 2 du génome A pour créer un nouveau génome (voir section 5.1.2.1).

Le langage de description de phénomène permet quant à lui de définir un certain nombre de modules paramétriques imbriquables, appelés règles, modélisant les métaphores de présentation de l'information. L'auteur A définit ainsi un module paramétrique de distribution circulaire qu'il utilise pour la présentation des niveaux d'exploration 2 et 3 de la métaphore de la tour de carrousels (voir section 5.1.1).

Implémentation Afin de s'adapter au mieux aux contraintes liées à l'intégration de nouveaux formats de données en entrée et en sortie de notre pipeline de création, nous proposons deux implémentations pour les composants de génération de génotypes (voir section 4.2.5) et de génération de phénotypes (voir section 4.3.3). Au niveau de la structuration de l'information, nous proposons deux implémentations que nous avons mises en œuvre respectivement dans le cadre de la structuration du catalogue VoD d'Orange et de la base de données multimédia de l'auteur A (voir section 4.2.5).

En ce qui concerne la présentation de l'information, deux implémentations sont proposées pour la production de phénotypes. L'intégration d'un nouveau format 3D de description de scène correspond alors à l'implémentation d'un composant de traduction. Dans le cadre du développement des démonstrateurs présentés en section 5.1, nous avons mis en œuvre la deuxième implémentation du module de présentation de l'information, avec un composant de traduction de phénomène en un générateur de phénotypes décrits en X3D.

Dans les deux cas, les composants de génération de génotypes, tout comme les composants de génération de phénotypes, sont déclinés pour chaque format de données pris en compte. Associés à la généralité des langages de description de génomes et de phénomènes, cette modularité assure la généralité de notre formalisme concernant les formats de gestion de données en entrée du système et les formats de description de scènes 3D en sortie du système. Le créateur d'une interface graphique choisira donc, en fonction de ses besoins et de ses contraintes, le composant le plus adapté à son application.

La modularité du pipeline de création associée à la modularité des langages formels facilite l'utilisation de notre formalisme par les créateurs d'interfaces graphiques 3D, quel que soit leur niveau d'expertise en programmation. Ils peuvent ainsi personnaliser la métaphore de visualisation choisie par la modification de paramètres, l'imbrication des modules ou la définition de nouveaux modules.

5.2.3 Variété et adaptabilité des interfaces graphiques 3D

Grâce à la généralité du module de structuration de l'information, tout type de contenu issu d'ensembles de données peut être visualisé à travers des interfaces graphiques 3D. Si les exemples proposés dans la section 5.1 se limitent à l'exploration de bases de données multimédia, notre formalisme permet également de créer des interfaces graphiques 3D pour la visualisation de données pour des applications variées, comme le e-commerce, la culture, ou la visualisation de données scientifiques par exemple. La variété des opérateurs du langage de description de phénomènes permet également de définir des métaphores de présentation de l'information très variées, allant des interfaces graphiques immersives (comme par exemple la métaphore de l'univers Orange) aux interfaces graphiques non immersives (comme par exemple la métaphore de la tour de carrousels). Ces métaphores de présentation peuvent être génériques, c'est à dire adaptées à tout type d'information, ou sémantiquement corrélées au domaine applicatif des données à visualiser.

Les métaphores de présentation de l'information permettent d'adapter l'interface graphique 3D à différents dispositifs d'interaction et de restitution. Par exemple, lorsque l'interface graphique est destinée à être utilisée via un téléphone mobile, le concepteur peut choisir des modèles 3D simples et une interface limitant le nombre de données visibles à un instant t , soit en privilégiant le regroupement de données, soit en subdivisant l'interface graphique en plusieurs scènes, chaque scène étant chargée à la demande sur le terminal en fonction des actions de l'utilisateur final. De même, les interactions proposées peuvent différer si le dispositif d'interaction cible est limité à un clavier de téléphone ou à une surface tactile.

Architecture Que ce soit au niveau des contenus ou des dispositifs de visualisation et d'interaction, notre formalisme permet donc de décrire des interfaces graphiques 3D adaptées à de multiples usages. La modularité de notre formalisme permet également d'adapter l'architecture en fonction de l'application. Ainsi, les différents fichiers (génomes, phénomènes et génotypes, phénotypes) peuvent être sauvegardés sur le serveur ou le client. De même, les transformations permettant de produire les génotypes et phénotypes peuvent être exécutées sur le serveur ou le client. La création de fichiers intermédiaires (génotype, générateur de phénotype spécifique à un format 3D et à une métaphore de présentation de l'information, ...) facilite la création d'architectures mixtes où une partie de l'information et des transformations sont gérées par le serveur, et l'autre partie est gérée par le client. De telles architectures permettent par exemple à un utilisateur d'appliquer une métaphore de présentation de l'information sauvegardée en local au génotype généré par l'auteur A pour la visualisation de sa base de données multimédia. Pour information, la taille des fichiers, après compression zip, correspondant aux métaphores de la tour de carrousels et de l'univers Orange sont indiqués dans la table 5.6. Le génotype auquel le phénomène X3D est appliqué correspond au génotype de la base de VoD d'Orange. L'Univers Orange total correspond à l'ensemble des fichiers produits à partir du génotype extrait de la base VoD d'Orange, tandis que l'Univers Orange minimal correspond à la taille moyenne des trois fichiers nécessaires à l'exploration d'un film (c'est-à-dire la scène principale, un hall de cinéma et une salle de projection). Les prototypes et textures utilisés ne sont pas pris en compte. Pour information, l'ensemble des prototypes de la métaphore de l'Univers Orange pèse 3,02Mo et l'ensemble des textures associées pèse 4,6Mo, après compression zip.

Métaphore	Phénomène	Phénomène X3D	Phénotype
Tour de carrousels	1,56	1,85	3,08 (1 fichier)
Univers Orange (total)	3,3	3,46	548 (366 fichiers)
Univers Orange (minimal)	3,3	3,46	4,3 (3 fichiers)

Table 5.6 – Poids des fichiers après compression zip (en kilo octets).

Notre formalisme peut donc être adapté à plusieurs architectures. Cependant, le système de gestion de données choisi, l'implémentation du générateur de génotype et le nombre des données à traiter peut entraîner un temps de production du génotype incompatible avec la structuration de l'information en temps réel. La création dynamique de sous-génotypes (voir section 4.2.2) permet alors de diviser l'organisation des données en sous-problèmes et de ne calculer que le sous-ensemble de l'organisation nécessaire pour la visualisation des données à un instant t . Ainsi, pour la métaphore de l'univers Orange, il n'est pas nécessaire de calculer le génotype en entier. Pour la première scène, seuls les genres sont extraits de la base de données. Lors du choix d'un cinéma, seuls les réalisateurs ayant produit des films correspondant au genre choisi sont extraits, et ainsi de suite. Un système de prédiction de l'exploration de l'utilisateur peut améliorer la création dynamique de sous-génotypes en prenant en compte les déplacements de l'utilisateur dans la scène 3D. Ainsi, le calcul du sous-génotype correspondant à un genre est initialisé/stoppé selon que l'utilisateur s'approche ou s'éloigne du cinéma correspondant. L'intégration de fonctionnalités de regroupement de données au composant de création dynamique de sous-génotypes permettrait également d'adapter le nombre de niveaux d'exploration en fonction du nombre d'éléments résultant d'une requête.

Lorsque la création dynamique de génotype ne convient pas, le créateur de l'application peut pré-calculer un ensemble fini de génotypes correspondant à un ensemble fini de structurations de l'information. Les génotypes correspondant sont sauvegardés côté serveur, et l'utilisateur de l'interface graphique pourra alors choisir la structuration qui lui convient. Ces problèmes et solutions s'appliquent également à la création de phénotypes. Ainsi, comme nous l'avons vu en section 4.3.2.2, un phénomène permettant la visualisation de trois niveaux d'exploration à l'aide de métaphores uni-niveau produit

$1 + g + \sum_{i=1}^g r_i$ fichiers, où g est le nombre d'éléments du premier niveau d'exploration, et r_i le nombre

d'enfants de l'élément de rang i du premier niveau d'exploration. La production dynamique des fichiers en fonction de l'exploration de l'utilisateur permet de répartir de manière efficiente le temps de calcul nécessaire à la visualisation de l'interface graphique interactive. Un ensemble de phénotypes peut également être pré-calculés et sauvegardé côté serveur. La table 5.7 présente les temps de calcul nécessaires à la création non dynamique des phénotypes correspondant aux métaphores de la tour de carrousel et de l'univers Orange. La transformation Phénomène vers Phénomène X3D correspond à la création du générateur de phénotype X3D (voir section 4.3.3) et peut être pré-calculé. La transformation Phénomène X3D vers Phénotype correspond à l'application du phénomène sur le génotype issu de la base de VoD d'Orange et produit un ensemble de scènes X3D.

Métaphore	Phénomène -> Phénomène X3D	Phénomène X3D -> Phénotype	Total
Tour de carrousels	703	562	1265
Univers Orange	703	610	1313

Table 5.7 – Temps d'exécution des transformations permettant la génération d'un phénotype (en millisecondes).

Le choix de l'architecture doit donc être adapté aux contraintes de l'application, comme par exemple le volume des données, les performances visées, ou encore les différentes plateformes cibles.

Interfaces graphiques immersives et multi-utilisateurs Notre formalisme doit permettre la définition de métaphores de présentation de l'information variées, dont des interfaces graphiques 3D immersives. Si l'immersion peut apporter un certain nombre d'avantages, caractère ludique, exploitation maximale des trois dimensions, etc., elle peut entraîner un certain nombre de problèmes. Par exemple, dans le cadre d'une navigation non contrainte, il est difficile de gérer les points-de-vue associés aux

éléments d'information de manière relative à l'élément en cours de visualisation. Ainsi, si un utilisateur choisi un point-de-vue associé à un élément E d'information, puis se déplace librement vers un élément F, la gestion de l'opérateur `NextSibling` pose des difficultés : l'élément suivant peut référer au frère de l'élément E ou au frère de l'élément F.

Les interfaces graphiques 3D peuvent également être multi-utilisateurs, ajoutant de nouvelles contraintes aux métaphores de présentation de l'information. Par exemple, la métaphore de l'univers Orange n'est pas utilisable telle quelle pour une utilisation multi-utilisateurs. En effet, l'utilisateur final peut effectuer une rotation sur les cinémas représentant les éléments de niveau 1, et une translation sur les portes et panneaux représentant les éléments de niveau 2 et 3 respectivement. Si plusieurs utilisateurs finaux sont présents dans la même scène 3D, ces actions entraînent l'inutilisabilité de l'interface graphique. Notre formalisme permet de définir plusieurs protocoles d'usages gérés par le moteur 3D :

- seules un certain nombre d'interactions sont autorisées dans les univers multi-utilisateurs. Dans ce cas, la métaphore de présentation de l'information des niveaux d'exploration 2 et 3 pose problème car une partie de l'information peut devenir inaccessible (par exemple, lorsqu'il y a trop de portes pour toutes les positionner dans l'espace réservé) ;
- la scène est dupliquée pour chaque groupe d'utilisateurs finaux. Dans la majorité des cas, l'interface graphique sera donc en mode mono-utilisateur. Cependant, un utilisateur peut inviter d'autres utilisateurs à le rejoindre dans l'interface graphique. Ainsi par exemple, un groupe d'amis souhaitant visualiser un film pourra se retrouver dans l'interface graphique 3D proposée par Orange et naviguer ensemble pour choisir un film ;
- une partie de la scène 3D est commune à tous les utilisateurs (les invariants de la métaphore de présentation de l'information), tandis qu'une partie (les éléments d'information) est dupliquée pour chaque utilisateur. Dans ce cas, deux utilisateurs positionnés côte à côte dans une scène 3D (par exemple face à un cinéma) ne visualiseront pas la même information (par exemple, le genre Comédie et le genre Policier). Cependant, une structuration de l'information intelligente (par exemple le classement par ordre alphabétique) permet à ces utilisateurs de discuter pour obtenir la même visualisation, ou pour suivre les recommandations de l'autre utilisateur.

Dans le dernier cas proposé, le fait que deux utilisateurs visitant la même scène aient une visualisation différente (position des éléments d'information par exemple) est acceptable et est considérée comme ne posant pas de problème pour l'exploration de l'information et la recherche de données, tout en ne gênant pas l'interaction entre les utilisateurs. Dans ce cas, il est possible d'imaginer deux utilisateurs finaux présents dans une même scène 3D, mais chacun d'eux ayant choisi un style graphique différent (l'univers Orange et l'univers en papier par exemple). Ces utilisateurs pourront alors interagir et dialoguer sur le contenu de l'interface malgré les différences graphiques de la scène visualisée, et ce grâce aux invariants de représentation communs aux deux métaphores (par exemple, le nom et la couleur associée aux cinémas, l'univers cinématographique conservé avec la place, les bâtiments, les halls, les maillages de collision, ...).

Dans tous les cas, ces implémentations sont directement liées à la plateforme finale de visualisation et non pas à notre formalisme.

Qu'il s'agisse de contraintes liées à l'immersivité d'une interface graphique, de la gestion des utilisateurs (duplication ou non des scènes 3D, mutex, ...), ou tout autre contrainte, les opérateurs proposés pour la définition de métaphores de présentation de l'information ont pour objectif d'assurer une forte variabilité des interfaces graphiques 3D pouvant être définies. Le concepteur d'une interface graphique 3D doit donc choisir les opérateurs adaptés à ses contraintes et s'assurer de la cohérence de la métaphore de présentation de l'information choisie. De même, le gestionnaire de données (créateur de l'interface graphique, utilisateur final ou application tierce) est le seul garant de la cohérence de la structuration de l'information choisie.

5.2.4 Extensibilité et compatibilité avec les technologies Web

Dans ce document, nous proposons un formalisme pour la description et la génération d'interfaces graphiques 3D pour le Web. Afin de faciliter l'adoption de notre technologie par les développeurs Web, notre pipeline de création séparant structuration et présentation de l'information est directement inspirée des pipelines de création de sites Web dynamiques. De plus, l'indépendance de notre formalisme vis à vis des systèmes de gestion de données facilite l'intégration de bases de données existantes à notre pipeline de création. Au niveau des technologies choisies, nous avons opté pour les langages

XML et XSLT pour l'implémentation de notre formalisme afin d'assurer sa compatibilité avec les technologies Web existantes et son extensibilité, lui permettant ainsi de s'adapter à de nouvelles contraintes et de nouveaux usages. La définition de langages génériques pour la description des métaphores structuration et de présentation de l'information assurent la compatibilité de notre formalisme avec les technologies de gestion de données et de description de scènes 3D existants sur le Web, et assure la pérennité de notre formalisme.

5.2.5 Conclusion

Notre formalisme de description et de génération d'interfaces graphiques 3D pour le Web est donc flexible :

- généralité des formats de gestion de données et des formats de description de scènes ;
- modularité du pipeline de création et des langages de description présentés ;
- extensibilité ;
- possibilité de décrire des interfaces graphiques variées, que ce soit au niveau des contenus, des usages, ou des métaphores de structuration et de présentation de l'information.

Ce formalisme permet également la diminution des coûts de production en automatisant la production de contenus dynamiques exploitant de grands ensembles de données. Ainsi, la description de la métaphore de l'univers Orange a demandé trois jours de développement au créateur de l'interface graphique, et un mois de création des éléments 3D par un infographiste 3D. La métaphore de visualisation ainsi définie est directement adaptable à de nombreux ensembles de données. Un démonstrateur a ainsi été développé en partenariat avec une société spécialisée dans la gestion de données (en particulier l'agrégation de données sémantiques) afin de visualiser via l'interface graphique 3D reprenant la métaphore de l'univers Orange les résultats de requêtes effectuées à l'aide des technologies développées par cette société. À partir de la diffusion du schéma XML du génome et du génotype, cette société a produit les génotypes correspondant aux résultats de recherches exploitables par nos phénomènes présentés en section 5.1. L'intégration a duré quelques jours, et quelques jours supplémentaires ont permis la modification de la métaphore de l'univers Orange pour intégrer un certain nombre d'éléments liés à cette collaboration (logo de la société partenaire, images de présentation de la technologie, ...).

Chapitre 6

Conclusion et perspectives

6.1 Contributions

Les travaux présentés dans ce documents ont pour objet la création automatique d'interfaces graphiques pour le Web 3D, et plus particulièrement à la visualisation dans un espace tridimensionnel de résultats de requêtes effectuées sur de grands ensembles de données. Afin de mieux définir le contexte de nos travaux et les contraintes qui en découlent, nous avons donc effectué un bref état de l'art des solutions industrielles et normatives concernant quatre domaines : les outils de modélisation 3D, les interfaces d'acquisition et de restitution, les technologies et usages relatifs aux contenus 3D sur le Web, et enfin les technologies de gestion de données. Nous avons effectué une étude plus approfondie des techniques existantes de génération automatique d'interfaces graphiques 3D, ainsi que des caractéristiques nécessaires à un outil de création d'interfaces graphiques pour permettre à ses utilisateurs de définir des interfaces efficaces, que ce soit du point de vue de la gestion des données ou de leur présentation.

À partir de cet état de l'art, nous avons établi un cahier des charges définissant les caractéristiques et contraintes nécessaires à la définition d'un formalisme pour la production automatique d'interfaces graphiques 3D pour le Web, puis nous avons proposé un formalisme complet répondant à ces critères ([32, 31, 33, 34, 35]). Ce formalisme permet d'une part la définition de métaphores variées pour la visualisation de données, et d'autre part de générer les interfaces graphiques correspondantes sous forme d'un ensemble de scènes 3D. Le pipeline de création est divisé en deux modules : un module de structuration de l'information et un module de présentation de l'information. Chacun de ces modules intègre un langage générique de description de métaphore et compatible avec les systèmes de gestion de données et les formats de description de scènes 3D utilisés sur le Web. Nous proposons également deux implémentations pour chacun des modules de la chaîne de création. Ces implémentations sont fortement basées sur les technologies Web et permettent une intégration rapide de notre technologie aux technologies Web existantes.

Nous avons mis en œuvre trois des quatre implémentations proposées et réalisé trois démonstrateurs prenant en compte la chaîne complète de création d'interfaces graphiques, de la gestion des données à la visualisation interactive et l'exploration de l'information. Le premier démonstrateur présente le cas de la définition d'une métaphore de visualisation de l'information générique et non-immersive. Les deux autres démonstrateurs correspondent à la définition de métaphores de visualisation de l'information immersives et corrélées au domaine applicatif des données visualisées. Les deux derniers démonstrateurs ont été développés en collaboration avec un infographiste 3D et ont permis de valider la séparation des métiers entre la conception de l'interface graphique et la création des modèles 3D. La validation des usages de notre formalisme avec un système de gestion de données sémantiques permettant l'agrégation de données issues de sources variées est en cours avec le développement d'un démonstrateur en collaboration avec une société partenaire.

Enfin, à partir de ces démonstrateurs, nous avons pu valider la concordance de notre formalisme avec le cahier des charges défini, notamment en ce qui concerne la flexibilité du formalisme proposé.

6.2 Perspectives

6.2.1 Implémentations

Dans sa version actuelle, nous avons intégré deux systèmes de gestion de données et un format de description de scènes 3D à notre formalisme de description et de création d'interfaces graphiques pour le Web 3D. Nous souhaitons continuer la mise en œuvre des modules de génération de génotypes et de génération de phénotypes afin d'élargir la base de formats de données compatibles avec notre pipeline de création. Nous souhaitons également implémenter la création dynamique de génotypes au fur et à mesure de l'exploration des données par l'utilisateur à partir du génome, et permettre l'association d'un unique phénomène avec l'ensemble des génotypes ainsi générés.

Afin de rendre notre formalisme accessible aux concepteurs d'interfaces graphiques pour le Web, nous souhaitons développer un outil auteur permettant la définition du génome et du phénomène de manière graphique, et intégrant une pré-visualisation interactive du résultat. L'interactivité de la pré-visualisation permettrait entre autre de modifier directement les paramètres du phénomène depuis la scène 3D affichée (comme par exemple le rayon d'une distribution circulaire ou la couleur d'un élément). Cet outil auteur intégrera un certain nombre de règles et d'objets et widgets couramment utilisés dans le cadre de la présentation d'information, et adaptés à la 3D, comme les listes, les arbres, les boutons, etc.

6.2.2 Travaux de recherche

6.2.2.1 Tests utilisateurs

Une fois l'outil auteur développé, nous souhaitons tester notre formalisme auprès de concepteurs d'applications Web. De plus, grâce à la flexibilité de notre formalisme, nous souhaitons utiliser cet outil pour générer des interfaces graphiques 3D testées par des utilisateurs finaux, dans le cadre par exemple de l'estimation de l'intérêt d'une représentation 3D par rapport à une représentation 2D, de recherches sur l'ergonomie des interfaces 3D, ou en phase de développement d'interfaces graphiques 3D commerciales afin de tester rapidement et à faible coût plusieurs métaphores de visualisation de l'information. Notre formalisme peut également permettre le développement d'interfaces graphiques 3D multi-utilisateurs pour tester différentes configurations de présentation et d'exploration de l'information. En particulier, nous aimerions utiliser notre formalisme pour tester la duplication de l'information, ainsi que la personnalisation du style graphique de l'interface, pour chaque utilisateur présent dans une même scène 3D (voir section 5.2.3).

6.2.2.2 Amélioration du formalisme

Enfin, nous souhaiterions continuer les recherches sur la conception des interfaces graphiques 3D et ajouter un certain nombre de fonctionnalités à notre formalisme. Tout d'abord, nous souhaitons continuer la collaboration autour de la visualisation de données sémantiques, et développer un certain nombre d'outils permettant de sélectionner automatiquement des niveaux d'exploration à partir de discriminants quantitatifs et qualitatifs.

Nous souhaitons également développer l'intégration de la sémantique dans les langages de description de génomes et de phénomènes. Parmi les usages visés, nous souhaiterions permettre la sélection et la modification automatique d'une métaphore de présentation de l'information en fonction de la sémantique des données visualisées (par exemple, choix d'une métaphore de bibliothèque pour une recherche littéraire, et d'une métaphore cinématographique pour une recherche liée au septième art) et de contraintes plastiques (par exemple, les univers Orange ou papier définis en section 5.1.2 correspondant respectivement à l'identité visuelle du fournisseur de données ou aux préférences de l'utilisateur final).

Afin d'améliorer l'accès à l'information, nous souhaitons intégrer à notre formalisme des opérateurs de navigation permettant de définir des modes de navigation plus ou moins contraints. De plus, nous souhaitons étudier le positionnement automatique de caméras afin d'améliorer l'accès à l'information présentée.

Enfin, nous souhaitons ajouter un certain nombre de fonctionnalités à notre formalisme pour permettre, outre la création d'interface graphiques 3D autonomes, des interfaces graphiques 3D intégrables à

des univers existants. Ainsi, la tour de carrousels pourrait être intégrée dans un univers type Second-Life pour représenter dynamiquement des informations issues d'une base de données. Cette nouvelle fonctionnalité nécessite la prise en compte de nouvelles contraintes, telle que la boîte englobante de l'interface, ou encore l'apparition et la disparition de modèles 3D. Prenons l'exemple de la tour de carrousels. En l'intégrant à une scène 3D externe, le concepteur peut contraindre la hauteur de la tour tout en conservant une largeur fixe des cylindres. Dans ce cas, si le premier niveau d'exploration contient un nombre trop important d'éléments, seule une partie d'entre eux seront affichés (cinq par exemple). Pour afficher les autres éléments, l'utilisateur final devra effectuer un glissement vertical sur la tour. Dans ce cas, les cylindres représentant les éléments du premier niveau d'exploration apparaissent et disparaissent en fonction de leur coordonnée verticale. Il faut donc intégrer au langage de description de phénomènes un certain nombre d'opérateurs permettant de définir les comportements dynamiques de l'interface graphique en fonction de contraintes spatiales.

Bibliographie

- [1] V. Ahl et T. Allen : *Hierarchy theory : a vision, vocabulary, and epistemology*. Columbia Univ Pr, 1996.
- [2] R. Arnaud et T. Parisi : Developing web applications with collada and x3d. *white paper*, 15, 2007.
- [3] M. Batty et P. Longley : *Fractal cities : a geometry of form and function*. Academic Press Professional, Inc. San Diego, CA, USA, 1994.
- [4] D. Beckett et B. McBride : Rdf/xml syntax specification (revised), 2004. URL <http://www.w3.org/TR/rdf-syntax-grammar/>.
- [5] J. Behr, Y. Jung, J. Keil, T. Drevensek, M. Zoellner, P. Eschler et D. Fellner : A scalable architecture for the html5/x3d integration model x3dom. In *Web3D '10 : Proceedings of the 15th International Conference on Web 3D Technology*, p. 185–194, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0209-8.
- [6] J. Behr, P. Eschler, Y. Jung et M. Zöllner : X3dom : a dom-based html5/x3d integration model. In *Web3D '09 : Proceedings of the 14th International Conference on 3D Web Technology*, p. 127–135, New York, NY, USA, 2009. ACM.
- [7] T. Berners-Lee, J. Hendler et O. Lassila : The semantic web. *Scientific American*, 284(5):28–37, 2001.
- [8] F. Bertault : A force-directed algorithm that preserves edge-crossing properties. *Information Processing Letters*, 74(1-2):7–13, 2000.
- [9] A. Berthoz : Bases neurales et référentiels de la mémoire des trajets. etudes par imagerie cérébrale. In *Neurophysiologie des mémoires*, p. 91–97. Elsevier Masson, 2005.
- [10] E. Bier : Skitters and jacks : interactive 3d positioning tools. In *Proceedings of the 1986 workshop on Interactive 3D graphics*, p. 183–196. ACM, 1987.
- [11] BlizzardEntertainment : World of warcraft, 2005. URL <http://www.worldofwarcraft.com/>.
- [12] A. Bosca, D. Bonino, M. Comerio, S. Grega et F. Corno : A reusable 3d visualization component for the semantic web. In *Web3D '07 : Proceedings of the twelfth international conference on 3D web technology*, p. 89–96, New York, NY, USA, 2007. ACM.
- [13] D. Bowman, E. Kruijff, J. LaViola et I. Poupyrev : 3d user interfaces : Theory and practice. 2004.
- [14] F. Brandenburg, M. Himsolt et C. Rohrer : An experimental comparison of force-directed and randomized graph drawing algorithms. *Lecture notes in computer science*, 1027:76–87, 1996.
- [15] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler et F. Yergeau : Extensible markup language (xml) 1.0 (fourth edition), 1998. URL <http://www.w3.org/TR/xml/>.
- [16] R. Carey et G. Bell : *The annotated VRML 2.0 reference manual*. Addison-Wesley Longman Ltd. Essex, UK, UK, 1997.
- [17] W. Chase : Spatial representations of taxi drivers. *Language*, p. 391–405, 1982.

- [18] J. J. Cherlin, F. Samavati, M. C. Sousa et J. A. Jorge : *Sketch-based modeling with few strokes*. Proceedings of the 21st spring conference on Computer graphics. ACM, Budmerice, Slovakia, 2005.
- [19] J. Clive : *The metaverse 2.0*. Rap. tech., Pelican crossing, 2007.
- [20] E. Codd : A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):387, 1970.
- [21] W. W. W. Consortium : Vrml 2.0 specification, 1994. URL <http://www.web3d.org/x3d/specifications/vrml/>.
- [22] W. W. W. Consortium : X3d specifications, 2004. URL <http://www.web3d.org/x3d/specifications/>.
- [23] M. Costabile, D. Malerba, M. Hemmje et A. Paradiso : Building metaphors for supporting user interaction with multimedia databases. *In Proceedings of 4th IFIP*, vol. 2, p. 27–29. Citeseer, 1998.
- [24] M. Di Benedetto, F. Ponchio, F. Ganovelli et R. Scopigno : Spidergl : a javascript 3d graphics library for next-generation www. *In Web3D '10 : Proceedings of the 15th International Conference on Web 3D Technology*, p. 165–174, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0209-8.
- [25] C. Doctorow : Metacarp : putting the torch to seven straw-men of the meta-utopia. *E-learning guru newsletter*, 2002.
- [26] E. Driver, C. Moore, P. Jackson, T. Keitt, C. Schooley et J. Barnett : *Web3d : the next major internet wave*. Forrester Research, Inc, 2008.
- [27] P. Durusau, S. Newcomb et R. Barta : Iso/iec 13250 topic maps, 1999. URL <http://www.isotopicmaps.org/tmrm/>.
- [28] P. Eades : A heuristic for graph drawing. *Congressus Numerantium*, 42(149160):194–202, 1984.
- [29] T. Elvins, D. Nadeau et D. Kirsh : Worldlets-3d thumbnails for wayfinding in virtual environments. *Proceedings of the 10th annual ACM symposium on User interface software and technology*, p. 21–30, 1997.
- [30] T. Erickson : Working with interface metaphors. *The art of human-computer interface design*, p. 65–73, 1990.
- [31] N. Esnault, J. Royan, R. Cozot et C. Bouville : Métaphores de présentation de l'information basées sur des méthodes procédurales. *In AFIG 2009*, Arles France, 11 2009.
- [32] N. Esnault, J. Royan, R. Cozot et C. Bouville : Procédé pour la visualisation dans un espace à n dimensions d'un ensemble de données structurées, 2009.
- [33] N. Esnault, J. Royan, R. Cozot et C. Bouville : A flexible framework to personalize 3d web users experience. *In Web3D '10 : Proceedings of the 15th International Conference on Web 3D Technology*, p. 35–44, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0209-8.
- [34] N. Esnault, J. Royan, R. Cozot et C. Bouville : A style-sheet technique for creating 3d web contents. *In Web3Dw '10 : Proceedings of the International Conference Web Virtual Reality and Three-Dimensional Worlds*. IADIS, 2010.
- [35] N. Esnault, J. Royan, R. Cozot et C. Bouville : A style-sheet technique for creating interactive 3d web contents. *IADIS International Journal on WWW/Internet*, 9(1), 2011.
- [36] P. Flinn : *Virtual universe community*, 2006.
- [37] P. Frankhauser : *La fractalité des structures urbaines*. Anthropos : Diffusion, Economica, Paris, 1994.

- [38] FraunhoferIGD : X3dom. URL <http://www.x3dom.org/>.
- [39] A. Frick, A. Ludwig et H. Mehldau : A fast adaptive layout algorithm for undirected graphs. *Proceedings of the DIMACS International Workshop on Graph Drawing*, p. 388–403, 1994.
- [40] G. Furnas et B. Bederson : Space-scale diagrams : understanding multiscale interfaces. *Proceedings of the SIGCHI conference on Human factors in computing systems*, p. 234–241, 1995.
- [41] L. M. Garshol : Living with topic maps and rdf. 2003.
- [42] N. Gershon et S. Eick : Visualization's new tack : Making sense of information. *IEEE spectrum*, 32(11):38–40, 1995.
- [43] Google : 3d warehouse, . URL <http://sketchup.google.com/3dwarehouse/>.
- [44] Google : Google sketchup, . <http://sketchup.google.com/>.
- [45] M. Hachet, F. Decle, S. Knodel et P. Guitton : Navidget for easy 3d camera positioning from 2d inputs. *In Proceedings of the 2008 IEEE Symposium on 3D User Interfaces-Volume 00*, p. 83–89. IEEE Computer Society, 2008.
- [46] H. Halpin, D. Zielinski, R. Brady et G. Kelly : Exploring semantic social networks using virtual reality. *In Proceedings of the 7th International Conference on The Semantic Web*, p. 614. Springer, 2008.
- [47] M. Hancock, T. ten Cate et S. Carpendale : Sticky tools : full 6DOF force-based interaction for multi-touch tables. *In Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, p. 133–140. ACM, 2009.
- [48] I. Herman, G. Melancon et M. Marshall : Graph visualization and navigation in information visualization : A survey. *Visualization and Computer Graphics, IEEE Transactions on*, 6(1):24–43, 2000.
- [49] H. Holger Rath : The topic maps handbook, 2003.
- [50] T. Igarashi, S. Matsuoka et H. Tanaka : Teddy : a sketching interface for 3d freeform design. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, p. 409–416, 1999.
- [51] M. Kay : Xsl transformations (xslt) version 2.0, January 2007. URL <http://www.w3.org/TR/xslt20/>.
- [52] Khronos : Html5 : A vocabulary and associated apis for html and xhtml, 08 2009. URL <http://www.w3.org/TR/2009/WD-html5-20090825/no.html#declarative-3d-scenes>.
- [53] Khronos : WebGL, 2009. URL <http://www.khronos.org/webgl/>.
- [54] G. Klyne, J. J. Carroll et B. McBride : Resource description framework (rdf) : concepts and abstract syntax, 2004. URL <http://www.w3.org/TR/rdf-concepts/>.
- [55] V. Kulish : *Hierarchical methods : Hierarchy and hierarchical asymptotic methods in electrodynamics*, vol. 1 de *Fundamental Theories of Physics*. Springer, 2002.
- [56] M. Lacher et S. Decker : On the integration of topic maps and rdf data. *Proceedings of SWWS*, 1:331–344, 2001.
- [57] LindenLab : Second life, 2002. <http://secondlife.com/>.
- [58] K. Lynch : *The image of the city*. MIT Press, 1960.
- [59] J. D. Mackinlay, S. K. Card et G. G. Robertson : Rapid controlled movement through a virtual 3d workspace. *In SIGGRAPH '90 : Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, p. 171–176, New York, NY, USA, 1990. ACM. ISBN 0-89791-344-2.

- [60] J. H. Martin : *A computational model of metaphor interpretation*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.
- [61] A. Massari, L. Saladini, M. Hemmje et F. Sisinni : Virgilio : A non-immersive vr system to browse multimedia databases. *In Proceedings of the IEEE International Conference on Multimedia Computing and Systems, IEEE Computer Society Press, Los Alamitos, CA*. Citeseer, 1997.
- [62] G. Moore : Rdf and topic maps : An exercise in convergence. *Proceedings of the XML Europe*, 2001.
- [63] S. J. Mountford : Tools and techniques for creative design. p. 128–141, 1995.
- [64] P. Murray-Rust : Chemical markup language. *World Wide Web Journal*, 2(4):147, 1997.
- [65] J. Nicolis : *Dynamics of hierarchical systems : an evolutionary approach*. Springer-Verlag London, UK, 1986.
- [66] Nintendo : Wii remote, 2005.
- [67] S. Pepper et G. Moore : Xml topic maps (xtn) 1.0, 2001. URL <http://www.topicmaps.org/xtn/1.0/index.html>.
- [68] M. Pesce, P. Kennard et A. Parisi : Cyberspace. *World Wide Web conference*, 1994.
- [69] N. F. Polys : Stylesheet transformations for interactive visualization : towards a web3d chemistry curricula. *In Web3D '03 : Proceedings of the eighth international conference on 3D Web technology*, p. 85–ff, New York, NY, USA, 2003. ACM.
- [70] V.-C. project : V-city. URL <http://vcity.c-s.fr/EN/index.html>.
- [71] G. Robertson, S. Card et J. Mackinlay : Information visualization using 3d interactive animation. *Communications of the ACM*, 36(4):57–71, 1993.
- [72] G. Robertson, J. Mackinlay et S. Card : Cone trees : animated 3d visualizations of hierarchical information. *In Proceedings of the SIGCHI conference on Human factors in computing systems : Reaching through technology*, p. 194. ACM, 1991.
- [73] R. Ruddle, S. Payne et D. Jones : Navigating buildings in "desk-top" virtual environments : Experimental investigations using extended navigational experience. *Journal of Experimental Psychology : Applied*, 3(2):143–159, 1997.
- [74] S. Rusinkiewicz, O. Hall-Holt et M. Levoy : Real-time 3d model acquisition. *ACM Trans. Graph.*, 21(3):438–446, 2002.
- [75] N. A. Salingaros : Connecting the fractal city. *In 5th Biennial of towns and town planners in Europe*, Barcelona, 2003.
- [76] M. Sarkar et M. Brown : Graphical fisheye views. *Communications of the ACM*, 37(12):73–83, 1994.
- [77] M. M. Schraefel et D. Karger : The pathetic fallacy of rdf. *In International Workshop on the Semantic Web and User Interaction (SWUI) 2006*, 2006. URL <http://eprints.ecs.soton.ac.uk/12911/>.
- [78] SingularInversions : Facegen. URL <http://www.facegen.com/>.
- [79] A. R. Smith : *Plants, fractals, and formal languages*. Proceedings of the 11th annual conference on Computer graphics and interactive techniques. ACM, 1984.
- [80] L. Sommaruga et N. Catenazzi : Curriculum visualization in 3d. *In Web3D '07 : Proceedings of the twelfth international conference on 3D web technology*, p. 177–180, New York, NY, USA, 2007. ACM.

- [81] K. Sons, F. Klein, D. Rubinstein, S. Byelozyorov et P. Slusallek : Xml3d : interactive 3d graphics for the web. *In Web3D '10 : Proceedings of the 15th International Conference on Web 3D Technology*, p. 175–184, New York, NY, USA, 2010. ACM.
- [82] SquareClock : Square clock, 2008. URL <http://www.squareclock.com/>.
- [83] N. Stephenson : *Snow Crash*. Bantam Books, 1992.
- [84] Symmetri : Search-cube. URL <http://search-cube.com/>.
- [85] A. Systems : Adobe flash. URL <http://www.adobe.com/products/flashplayer/>.
- [86] U. Technologies : Unity. URL <http://unity3d.com>.
- [87] E. Tolman : Cognitive maps in rats and men. *Image and Environment : Cognitive Mapping and Spatial Behavior*, 1973.
- [88] W3C : Document object model (dom) level 2 html specification, January 2003. URL <http://www.w3.org/TR/2003/REC-DOM-Level-2-HTML-20030109/>.
- [89] W3C : Html5 : A vocabulary and associated apis for html and xhtml, June 2010. URL <http://www.w3.org/TR/html5/>.
- [90] K. Walczak et W. Cellary : X-vrml - xml based modeling of virtual reality. *In SAINT '02 : Proceedings of the 2002 Symposium on Applications and the Internet*, p. 204–213, Washington, DC, USA, 2002. IEEE Computer Society.
- [91] K. Walczak, W. Cellary et M. White : Virtual museum exhibitions. *Computer*, 39(3):93–95, 2006.
- [92] Web3DConsortium : Web 3d consortium. URL <http://www.web3d.org/>.
- [93] WHITEvoid : Whitevoid portofolio. URL <http://www.whitevoid.com/portfolio/>.
- [94] A. Wilson, S. Izadi, O. Hilliges, A. Garcia-Mendoza et D. Kirk : Bringing physics to the surface. *In Proceedings of the 21st annual ACM symposium on User interface software and technology*, p. 67–76. ACM, 2008.
- [95] B. Witmer, J. Bailey, B. Knerr et K. Parsons : Virtual spaces and real world places : transfer of route knowledge. *International Journal of Human-Computer Studies*, 45(4):413–428, 1996.
- [96] W. Wiza, W. Cellary et K. Walczak : A method of holistic 3d visualization of arbitrarily large datasets. *In ISM '05 : Proceedings of the Seventh IEEE International Symposium on Multimedia*, p. 151–158, Washington, DC, USA, 2005. IEEE Computer Society.
- [97] W. Wiza, K. Walczak et W. Cellary : Periscope : a system for adaptive 3d visualization of search results. *In Web3D '04 : Proceedings of the ninth international conference on 3D Web technology*, p. 29–40, New York, NY, USA, 2004. ACM.
- [98] Yoowalk : Yoowalk, 2008. URL <http://www.yoowalk.com/>.
- [99] X. Zhang : A multiscale progressive model on virtual navigation. *Int. J. Hum.-Comput. Stud.*, 66(4):243–256, 2008.

Résumé

Le Web 3D atteint actuellement un nouveau stade de maturité grâce à la convergence de nouvelles technologies, comme par exemple de nouveaux standards permettant la visualisation native de contenus 3D dans les navigateurs Web. Malgré l'émergence de ces technologies, nous pouvons constater une faible pénétration des contenus 3D sur le Web, due entre autre au manque d'outils de création de haut niveau facilitant la production d'interfaces graphiques 3D pour le Web. Afin de palier ce manque, nous proposons un formalisme basé sur les technologies Web permettant de définir des métaphores complètes, attractives et efficaces pour la visualisation de grands ensembles de données, ainsi que la génération des interfaces graphiques 3D correspondantes. Ce formalisme doit être flexible en termes d'extensibilité, de modularité, de généralité concernant les formats de données en entrée et en sortie du système, ainsi qu'en terme de gamme d'interfaces graphiques 3D pouvant être définies par le concepteur d'une interface Web. Le pipeline de création que nous proposons permet la description de métaphores de visualisation d'information prenant en compte à la fois la structuration des données (la façon dont l'utilisateur explore et accède aux données), et la présentation (la géométrie et la représentation des données) et les interactions utilisateur. Ces métaphores permettent la visualisation d'information issue de grands ensembles de données. Les interfaces générées peuvent varier des interfaces graphiques 3D immersives à l'intérieur desquelles l'utilisateur final est représenté par un avatar et peut interagir avec d'autres utilisateurs, à des interfaces graphiques 3D non-immersives.

Abstract

Nowadays, the Web 3D is reaching a new milestone thanks to the convergence of various technologies, including new Web standards allowing Web browsers to natively display 3D content. Despite these factors, the lack of efficient and high level authoring tools to ease the creation and reduce the production cost of Web 3D graphical interfaces is still a string impediment to the large scale use of 3D Web contents. We thus propose a framework based on Web technologies to define complete, attractive and efficient 3D visualization metaphors of datasets, and generate the corresponding Web 3D graphical interfaces. This framework must reach a sufficient level of flexibility in terms of extensibility, of modularity, of genericity regarding the input and output file formats, as well as of range of graphical interfaces a 3D Web interface designer may define. Our creation pipeline allows the description of metaphor-based information visualization, taking into consideration both the data structure (how the user explores and discovers data) and the visual and interactive aspects of the computed 3D graphical interface (including the geometry and data representation). These metaphors may be used to visualize large volumes of information coming from databases or classic/semantic search results. The created graphical interfaces may be highly varying, from 3D immersive virtual worlds inside which the user can navigate with an avatar and interact with other users, to non-immersive 3D graphical interfaces.