HAL
open science

Cartographie et analyse territoriale multiscalaire. Réingénierie des logiciels HyperAtlas et HyperAdmin

Benoit Le Rubrus

To cite this version:

Benoit Le Rubrus. Cartographie et analyse territoriale multiscalaire. Réingénierie des logiciels HyperAtlas et HyperAdmin. Vision par ordinateur et reconnaissance de formes [cs.CV]. 2011. dumas-00592177

HAL Id: dumas-00592177

https://dumas.ccsd.cnrs.fr/dumas-00592177v1

Submitted on 11 May 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.

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

CENTRE REGIONAL RHÔNE-ALPES

CENTRE D'ENSEIGNEMENT DE GRENOBLE

MEMOIRE

présenté par **Benoit Le Rubrus**

en vue d'obtenir

LE DIPLÔME D'INGENIEUR C.N.A.M.

en INFORMATIQUE

Cartographie et analyse territoriale multiscalaire
Réingénierie des logiciels HyperAtlas et HyperAdmin

Soutenu le 7 avril 2011

JURY

Président :	M. Eric Gressier-Soudan
Membres :	M. Jean-Pierre Giraudin
	M. André Plisson
	M. Mathias Voisin-Fradin
Tuteur :	M. Jérôme Gensel
	Mme Hélène Mathian

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

CENTRE REGIONAL RHÔNE-ALPES

CENTRE D'ENSEIGNEMENT DE GRENOBLE

MEMOIRE

présenté par **Benoit Le Rubrus**

en vue d'obtenir

LE DIPLÔME D'INGENIEUR C.N.A.M.

en INFORMATIQUE

Cartographie et analyse territoriale multiscalaire
Réingénierie des logiciels HyperAtlas et HyperAdmin

Soutenu le 7 avril 2011

Les travaux relatifs à ce mémoire ont été effectués dans l'équipe STEAMER du LIG (Laboratoire d'Informatique de Grenoble) sous la direction de M. Jérôme Gensel.

Remerciements

Je remercie avant tout les personnes participant au jury de ce mémoire : Monsieur Eric Gressier-Soudan, professeur au CNAM Paris, Monsieur Jean-Pierre Giraudin, professeur à l'université Pierre Mendès France de Grenoble, Monsieur André Plisson, directeur du centre d'enseignement CNAM de Grenoble et Monsieur Mathias Voisin-Fradin, directeur adjoint du CNAM Grenoble.

Je tiens tout particulièrement à remercier mon tuteur Monsieur Jérôme Gensel, Professeur à l'université Pierre Mendès France de Grenoble, pour m'avoir accueilli dans l'équipe qu'il dirige, STEAMER, et proposé de travailler sur des problématiques liées à la géomatique. Merci pour ses conseils et relectures ayant mené à l'achèvement de ce mémoire.

Le travail réalisé pendant ce mémoire est le résultat d'une collaboration avec les équipes du groupe de recherche HyperCarte, je remercie donc Madame Hélène Mathian, de l'équipe Géographie-Cités, et les membres de l'équipe RIATE de l'université Paris VII, dirigée par Monsieur Claude Grasland, professeur géographe.

Je remercie également les membres permanents de l'équipe STEAMER pour leur soutien, leurs encouragements, leur sympathie et leur accueil. Ainsi, je remercie Madame Paule-Annick Davoine pour m'avoir intégré à l'équipe en juin 2009 sur un contrat relatif au projet GenGHIS, et Madame Marlène Villanova-Oliver pour m'avoir proposé en novembre 2009 un contrat sur le projet *ESPON 2013 Database*.

Merci aussi aux membres non permanents de l'équipe STEAMER : Christine Plumejeaud pour la transmission d'une partie de ses connaissances sur les logiciels HyperAtlas et HyperAdmin, Anton Telechev et Bogdan Moisuc, avec qui j'ai pu collaborer sur des projets STEAMER connexes, comme GenGHIS et *ESPON 2013 Database*.

Je remercie également l'équipe Marvelig du LIG et plus particulièrement Alban Chazot, qui a non seulement partagé son bureau, mais aussi sa culture et son expérience transversale sur les équipes, outils et projets du LIG. Pour n'oublier personne, je remercie les habitués du troisième étage du bâtiment D de l'ENSIMAG (Ecole Nationale Supérieure en Informatique et Mathématiques Appliquées de Grenoble) pour leur chaleureux accueil.

En outre, je remercie l'association AI CNAM-PST (Association des Ingénieurs du Conservatoire National des Arts et Métiers et de la Promotion Supérieure du Travail) pour leurs services de soutien, de relecture et de soutenance à blanc.

Je remercie enfin tous les enseignants et personnels du CNAM qui ont tous contribué lors de mon cursus à un inestimable enrichissement culturel.

Sommaire

Remerciements i

Conventions d'écriture et acronymes xi

Introduction 1

I Présentation 3

1 Contexte 5
1.1 LIG STEAMER . . . 5
1.2 Le groupe de recherche HyperCarte . . . 6
1.2.1 HyperAtlas . . . 8
1.2.2 HyperAdmin . . . 9
1.3 Le programme ESPON . . . 9

2 Cahier des charges 11
2.1 Les enjeux . . . 11
2.2 Le contrat *ESPON HyperAtlas Update* . . . 11
2.3 Le projet HyperAtlas v2 . . . 13

3 Organisation du mémoire 15
3.1 Calendrier . . . 15
3.2 Plan du mémoire . . . 16

II Etat de l'art 17

4 Méthodes et outils de visualisation 19
4.1 Philcarto . . . 19
4.2 Les cubes espace-temps . . . 22
4.3 SOLAP . . . 24

5 Solutions spatio-temporelles en ligne 29
5.1 Vizzuality . . . 29
5.2 GapMinder . . . 30
5.3 Google Public Data Explorer . . . 33
5.4 OECD eXplorer . . . 34

5.4.1 Vue générale . . . 35
5.4.2 Architecture . . . 35
5.4.3 Principaux outils . . . 38

6 Synthèse de l'état de l'art **39**

III Réalisation **43**

7 Démarche **45**

8 Analyse de l'existant **49**
8.1 HyperAtlas . . . 49
8.1.1 Carte de la zone d'étude . . . 50
8.1.2 Cartes des indicateurs . . . 51
8.1.3 Carte du ratio . . . 52
8.1.4 Cartes d'écarts . . . 52
8.1.5 Carte de synthèse ternaire . . . 54
8.2 HyperAdmin . . . 54
8.3 Technologies . . . 55
8.4 Configurations . . . 57
8.5 Gestion des évènements et singletons . . . 57

9 Génie logiciel **59**
9.1 Objectif . . . 59
9.2 Gestion des sources et des versions . . . 59
9.3 Construction automatique . . . 60
9.4 Tests unitaires . . . 61
9.5 Documentation . . . 62
9.5.1 DocBook . . . 62
9.5.2 Javadoc . . . 64
9.5.3 Outils de conception UML et de bases de données . . . 64
9.6 Intégration continue . . . 65
9.7 Synthèse de l'environnement . . . 66

10 L'application Web **69**
10.1 Objectif . . . 69
10.2 Cas d'utilisation . . . 69
10.3 Architecture . . . 70
10.4 Principales fonctionnalités . . . 74
10.4.1 Styles de mise en page . . . 74
10.4.2 Internationalisation . . . 75
10.4.3 Jeux de données . . . 76
10.4.4 Applet HyperAtlas . . . 76
10.4.5 HyperAdmin Web : particularité de la version ESPON . . . 78
10.4.6 Menu aide . . . 81
10.4.7 Insertion de nouveaux jeux de données . . . 82
10.4.8 Menu administration . . . 82

10.5 Synthèse sur l'application Web . . . 83

11 HyperAtlas v2 **85**
11.1 Objectifs . . . 85
11.2 Carte de synthèse binaire . . . 85
11.2.1 Implémentation . . . 87
11.3 Mode expert . . . 91
11.3.1 Passage au mode expert . . . 91
11.3.2 Cartes de redistribution . . . 92
11.3.3 La mesure des inégalités . . . 94
11.3.4 Autocorrélation spatiale . . . 99
11.4 Intégration du temps . . . 102
11.5 Définition d'une nouvelle aire d'étude . . . 104
11.6 Zoom et déplacement de la carte . . . 105
11.7 Habillage des cartes . . . 105
11.8 Gel des cartes et comparaison . . . 107
11.9 Synthèse d'HyperAtlas v2 . . . 109

12 HyperAdmin v2 **111**
12.1 Objectifs . . . 111
12.2 Extraction de la couche métier . . . 112
12.3 Enrichissement de la couche métier . . . 113
12.3.1 Métadonnées du jeu de données . . . 113
12.3.2 Analyseurs des fichiers d'entrée . . . 114
12.3.3 Redéfinition des fichiers de stocks . . . 115
12.3.4 Intégration des villes . . . 121
12.4 Synthèse d'HyperAdmin v2 . . . 121

IV Conclusion et annexes **123**

Conclusion **125**
Rappel des objectifs . . . 125
Synthèse des réalisations . . . 125
Perspectives . . . 129
Ce dont il n'a pas été question... . . . 130
Bilan personnel . . . 130

A Docbench **133**

B Schéma de la base de données **135**

C Glossaire **137**

Bibliographie et références **142**

A propos **143**

Liste des figures

1.1 Entrées sorties HyperAdmin et HyperAtlas . . . 7

3.1 Intégration du stage dans le calendrier STEAMER . . . 15

4.1 Carte à demi-cercles affrontés . . . 20
4.2 Carte de liens en oursins . . . 21
4.3 Cube espace-temps . . . 23
4.4 L'équation SOLAP . . . 25
4.5 JMap Spatial OLAP . . . 27
4.6 Architecture SOLAPLayers 2.0 . . . 28

5.1 Capture d'écran : Vizzuality . . . 30
5.2 Onglet graphique de Gapminder . . . 31
5.3 Onglet carte de Gapminder . . . 32
5.4 Google Public Data Explorer . . . 33
5.5 Vue générale de OECD eXplorer . . . 35
5.6 Architecture GAV . . . 36
5.7 Le cube 3D des données dans GAV . . . 37
5.8 Le flot de données dans eXplorer . . . 37
5.9 Création d'une sauvegarde XML avec eXplorer . . . 38

7.1 Vue schématique de la démarche . . . 45
7.2 Roue de Demming . . . 46

8.1 Vue générale d'HyperAtlas . . . 50
8.2 Arbre des hiérarchies entre unités . . . 51
8.3 Exemples de cartes de la zone d'étude . . . 51
8.4 Cartes à disques des indicateurs . . . 51
8.5 Carte de ratio et options de sa légende . . . 52
8.6 Cartes des écarts . . . 52
8.7 Légendes et options des cartes . . . 54
8.8 Synthèse ternaire . . . 55

9.1 Architecture de l'intégration continue . . . 65
9.2 Architecture de l'environnement . . . 67

10.1 Cas d'utilisations de l'application Web . . . 71
10.2 Vue générale de l'architecture . . . 72
10.3 MVC Web . . . 72

10.4 Architecture MVC2 avec Struts 2 . . . 73
10.5 Tuiles . . . 74
10.6 Feuiles de styles CSS en fonction du client . . . 74
10.7 Jeux de données disponibles . . . 76
10.8 Applet ESPON HyperAtlas . . . 77
10.9 Modèles de structure/géométrie version ESPON HyperAdmin Web . . . 79
10.10 ESPON HyperAdmin : choix d'un modèle de structure/géométrie . . . 80
10.11 ESPON HyperAdmin : soumission du fichier de statistiques . . . 80
10.12 ESPON HyperAdmin : nom et description . . . 80
10.13 ESPON HyperAdmin : fichier généré avec succès . . . 80
10.14 Menu Aide de l'application Web . . . 81
10.15 Formulaire de soumission d'un nouveau jeu de données . . . 82
10.16 Extrait de la base de données . . . 83
10.17 Echec des tests de l'application Web . . . 83
10.18 Succès des tests de l'application Web . . . 83

11.1 Légende de la carte de synthèse binaire . . . 86
11.2 Exemple de la carte de synthèse binaire . . . 87
11.3 Implémentation des quadrants . . . 88
11.4 Diagramme de classes : carte de synthèse binaire . . . 89
11.5 Intégration d'un composant repère . . . 90
11.6 Diagramme de séquence : passage au mode expert . . . 92
11.7 Mise en page du mode expert . . . 92
11.8 Carte de redistribution . . . 93
11.9 Spécialisation et généralisation : cartes de redistribution . . . 94
11.10 Courbe de Lorenz . . . 96
11.11 Diagramme de classes pour le calcul des indices . . . 98
11.12 Boîte statistique de l'onglet écart général . . . 98
11.13 Signes de l'autocorrélation spatiale . . . 99
11.14 Représentation d'autocorrélation spatiale attendue . . . 99
11.15 Exemple du graphique d'autocorrélation spatiale . . . 101
11.16 Sous-menu temporel des indicateurs . . . 103
11.17 Définition d'une nouvelle aire d'étude . . . 104
11.18 Modèle relationnel pour le style des unités . . . 107
11.19 Exemple d'utilisation du sélecteur de fenêtres . . . 108
11.20 Diagramme de classes : sélecteur de fenêtres . . . 109

12.1 Diagramme de classes couche modèle HyperAdmin . . . 112
12.2 Diagramme de composition des jeux de données . . . 113
12.3 Spécialisation de la classe `SerialElement` . . . 114
12.4 Diagramme de classes des analyseurs de fichiers . . . 115
12.5 Flot de données autour de la base ESPON . . . 118
12.6 Chronologie des réalisations . . . 126

B.1 Modèle physique de la base de données : internationalisation . . . 135
B.2 Modèle physique de la base de données : relations entre les entités . . . 136

Liste des tableaux

1.1 Terminologies des écarts 8

2.1 Echéancier des délivrables pour le projet ESPON 13

6.1 Comparaison des outils Web présentés dans l'état de l'art 40
6.2 Comparaison des outils présentés vs HyperAtlas et HyperAdmin 40

10.1 Exemple de fichiers pour l'internationalisation 75

11.1 Inéquations des différentes régions du repère 89

12.1 Onglet `About` du tableur de stocks 116
12.2 Exemple du contenu de l'onglet `Data` du tableur de stocks 116
12.3 Exemple du contenu de l'onglet `Default` du tableur de stocks 117
12.4 Exemple du contenu de l'onglet `Label` du tableur de stocks 117
12.5 Exemple du contenu de l'onglet `Metadata` du tableur de stocks 119
12.6 Exemple du contenu de l'onglet `Provider` du tableur de stocks 119
12.7 Exemple du contenu de l'onglet `RatioStock` du tableur de stocks 120
12.8 Exemple du contenu de l'onglet `StockInfo` du tableur de stocks 121
12.9 Complétude des tâches réalisées 126
12.10 Nombre de fichiers `.java` dans les paquetages du projet 127
12.11 Matrice SWOT 127

Conventions d'écriture et acronymes

Dans la version électronique de ce document, les liens intra-document, les renvois vers les bas de pages, les liens vers les références bibliographiques et les liens vers des sites Internet sont colorés en bleu. Pour une meilleure lisibilité, ces liens sont affichés en noir dans la version papier.

Les expressions en langues étrangères sont indiquées en italique. Pour ne pas nuire à la compréhension mais sans tomber dans l'excès, quelques termes courants du jargon informatique sont cependant repris tels quels, sans traduction française : « Web » ou « bug », par exemple.

Les noms de fichiers, de variables ou de classes sont écrits selon une police qui les distinguent du corps de texte, par exemple : `HyperAtlas.jar`.

Les termes suffixés par une étoile (exemple : NUTS*) font l'objet d'une définition dans le glossaire, voir l'annexe C page 137.

Trouvez ci-dessous une liste alphabétique non exhaustive des acronymes les plus fréquemment utilisés dans ce document.

API	*Application Programming Interface*
ATM	Analyse Territoriale Multiscalaire
CNAM	Conservatoire National des Arts et Métiers
ESPON	*European Spatial Planning Observation Network, Territorial Development and Cohesion*
HTML	*HyperText Markup Language*
IDE	*Integrated Development Environment*
JDBC	*Java DataBase Connectivity*
JVM	*Java Virtual Machine*
JRE	*Java Runtime Environment*
LIG	Laboratoire d'Informatique de Grenoble
NUTS*	Nomenclature d'Unités Territoriales Statistiques
OECD	*Organisation for Economic Co-operation and Development*
OCDE	Organisation de Coopération et de Développement Economiques
PDF	*Portable Document Format*
PNG	*Portable Network Graphics*
SGBD	Système de Gestion de Bases de Données
SCM	*Source Content Management*
SIG	Système d'Information Géographique
STEAMER	*Spatio-TEmporal information, Adaptability, Multimedia and KnowlEdge Representation*
UML	*Unified Modeling Language*
XML	*eXtensible Markup Language*

Introduction

A la fin des années 1960, l'informatisation de la production cartographique donne lieu à la naissance d'une nouvelle technologie, la géomatique. Afin de les rendre pleinement exploitables, les données cartographiques s'organisent au cours des années 1980 au sein d'outils appelés systèmes d'information géographique (SIG). Les SIG reposent en général sur des systèmes de gestion de bases de données relationnelles (SGBDR). Ils proposent des outils pour la gestion et le traitement des relations spatiales entre les objets, des fonctionnalités pour la représentation sous forme de carte de l'information géographique. Ces outils sont destinés à des spécialistes.

Dans les secteurs de l'environnement, des catastrophes naturelles, de la santé, de la gestion des transports ou de l'aménagement du territoire, les décideurs manquent d'outils cartographiques simples et puissants pour intégrer, explorer, synthétiser, et analyser des données spatio-temporelles.

Les logiciels HyperAtlas et HyperAdmin, développés depuis 2000, viennent répondre à ce besoin. Ils proposent une interface conviviale pour réaliser des analyses territoriales multiscalaires (ATM), dont le concept et les bénéfices sont précisément décrits dans ce mémoire.

En vue d'obtenir le diplôme d'ingénieur CNAM en informatique, ce document présente le travail effectué pendant le stage, dédié à l'apport de nouvelles fonctionnalités aux logiciels HyperAtlas et HyperAdmin.

Ce document s'articule en quatre parties. La première partie complète cette introduction, en présentant le contexte d'exécution, et en précisant nos objectifs. La deuxième partie propose de situer nos logiciels parmi une gamme de solutions existantes. La troisième partie décrit les réalisations effectuées pendant le stage. Enfin, la quatrième partie synthétise le travail réalisé et propose deux annexes, un glossaire, et une bibliograhie.

Première partie

Présentation

Chapitre 1

Contexte

Ce chapitre présente les principales entités organisationnelles liées au contexte de ce mémoire : le stage est effectué dans l'équipe LIG STEAMER présenté dans la section 1.1, une des équipes engagées dans le groupe de recherche HyperCarte, (section 1.2), pour répondre à un projet du programme ESPON, présenté dans la section 1.3.

1.1 LIG STEAMER

Le LIG [1] a été créé en janvier 2007 au terme d'une restructuration de plusieurs laboratoires grenoblois. Il rassemble près de 500 chercheurs, enseignants, doctorants et techniciens répartis en 24 équipes de recherches. Ses partenaires académiques sont le CNRS (Centre National de la Recherche Scientifique), Grenoble INP (Institut National Polytechnique), l'INRIA (Institut National de Recherche en Informatique et Automatisme), les universités Joseph Fourier et Pierre Mendès France.

L'équipe STEAMER [2] a été créée en janvier 2007 en même temps que le LIG à partir de membres de l'équipe SIGMA (Systèmes d'Information : inGénierie et Modélisation Adaptables) du LSR (Logiciel Systèmes et Réseaux).

Les recherches de l'équipe STEAMER portent sur les systèmes d'information du Web et plus particulièrement sur le traitement d'informations de nature multimédia à références spatiales et temporelles.

L'objectif est de proposer des formalismes et des méthodes permettant de progresser dans la conception, la mise en oeuvre et l'utilisation de systèmes d'information multimédia spatio-temporelle.

Les problématiques sont abordées sous l'angle de l'adaptation, en prenant en compte la diversité des utilisateurs, des contextes d'utilisation (stations de travail, téléphonie mobile, localisation fixe ou variable, Web) pour proposer un service adapté en termes de contenus et de présentation.

1. Laboratoire d'Informatique de Grenoble http://www.liglab.fr (consulté le 8 mai 2010).
2. *Spatio-TEmporal information, Adaptability, Multimedia and KnowlEdge Representation* http://steamer.imag.fr/ (consulté le 8 mai 2010).

Les travaux de l'équipe consistent également à la conception et au développement de plates-formes reposant sur les systèmes de bases de connaissances à objets comme AROM (Allier Relations et Objets pour Modéliser), sur des formalismes structurés (ontologies objets) et des technologies Web. Ces travaux trouvent des applications dans les domaines de la géomatique et des risques naturels.

Il sera dans ce mémoire essentiellement question des travaux de l'équipe dans le domaine de l'analyse spatiale et de la cartographie interactive, dans le cadre des projets développés au sein du groupe de recherche HyperCarte, présenté dans la section suivante.

1.2 Le groupe de recherche HyperCarte

Le groupe de recherche HyperCarte [3] fondé en 1996 repose sur une coopération interdisciplinaire entre des équipes de chercheurs du CNRS RIATE (UMS 2414), Géographie-cités (UMS 8504), les équipes STEAMER et Mescal du LIG (UMR 5217). Ces équipes mettent en commun leurs expertises et compétences dans les domaines de la géographie, la sociologie, l'économie et l'informatique pour fournir des outils permettant une visualisation et une analyse spatiale interactive de phénomènes socio-économiques, environnementaux ou culturels.

L'objectif du groupe de recherche consiste à répondre à la complexité des demandes sociales et politiques adressées à la cartographie statistique, par la mise au point d'outils interactifs de production.

Comme le décrit Claude Grasland [GMV+05], le « *concept HyperCarte repose sur l'hypothèse que toute spatialisation d'un phénomène social peut faire l'objet d'un nombre infini de représentations. Au lieu de se limiter à l'univers des représentations individuelles, l'approche propose d'ouvrir le champ des possibilités de représentation cartographique en offrant à chacun la possibilité d'en construire des représentations multiples en fonction de son identité ou de ses objectifs* ».

La mise en œuvre des idées du groupe de recherche HyperCarte est concrétisée par le développement depuis 2001 de plusieurs logiciels, chronologiquement réalisés dans le cadre des stages mémoires CNAM de :

- Philippe Martin, en 2004 [Mar04] ;
- Olivier Cuenot, en 2005 [Cue05] ;
- Christine Plumejeaud, en 2006 [Plu07] ;
- Christophe Chabert, en 2007 [Cha07] ;
- Raphaël Thomas, en 2008 [Tho08].

Les noms de ces logiciels ayant évolué au cours du temps, il s'agira par convention dans ce mémoire de les distinguer suivant les dénominations actuelles suivantes :

HyperAtlas : nommé à l'origine Hypercarte [Mar04], ce logiciel est présenté comme un module d'analyse territoriale multiscalaire (ATM ou en anglais, MTA, pour *Multiscalar Territorial Analysis*). Le logiciel permet de visualiser sur différentes cartes des valeurs d'indicateurs socio-économiques, les résultats d'analyses spatiales statistiques de ratios et écarts à des contextes références.

HyperAdmin : ce module est destiné à l'intégration de données géographiques et statistiques pour générer un jeu de données manipulable par l'application HyperAtlas. Le fichier `.hyp`* généré en sortie d'HyperAdmin constitue un fichier d'entrée pour le logiciel HyperAtlas.

3. Groupe de Recherche HyperCarte http://hypercarte.imag.fr (consulté le 8 avril 2010).

HyperSmooth : ce module vient compléter la palette d'analyse cartographique du projet HyperCarte en proposant à l'utilisateur une représentation continue des phénomènes spatiaux basée sur l'utilisation de cartes de potentiel. Non maintenu depuis 2007, il ne sera pas question dans ce mémoire de ce module.

La figure 1.1 illustre le flot de processus et les entrées-sorties liant HyperAdmin et HyperAtlas suivant les quatre étapes suivantes :

1. la préparation des données, l'utilisateur doit fournir à HyperAdmin trois ensembles de fichiers décrivant :
   - la géométrie des unités élémentaires suivant le format d'échange standard MIF-MID de MapInfo ;
   - la structure de l'espace divisé en une hiérarchie d'unités territoriales. L'utilisateur doit ici fournir un fichier tableur `xls` composé d'un ensemble de feuilles respectant un modèle défini, ou un ensemble de fichiers au format texte, équivalant aux différentes onglets du tableur, le caractère tabulation remplaçant les colonnes ;
   - les stocks, c'est-à-dire les statistiques valorisées sur les unités territoriales élémentaires. L'utilisateur a le choix entre un fichier tableur `xls` composé de feuilles respectant un modèle attendu ou l'équivalent sous la forme de fichiers au format texte.
2. le traitement des données, réalisé par HyperAdmin : suivant le nombre d'unités élémentaires fournies en entrée, HyperAdmin peut nécessiter des fonctions topologiques du SGBD (Système de Gestion de Bases de Données) PostgreSQL et de son extension pour les types géométriques PostGIS ;
3. l'export des données : HyperAdmin a pour objectif de fournir en sortie un jeu de données binaire au format `hyp`* ;
4. l'exploitation des données : le fichier `hyp` est le format attendu en entrée d'HyperAtlas. L'utilisateur peut alors réaliser ses analyses à l'aide de cartes interactives et générer des rapports au format HTML.

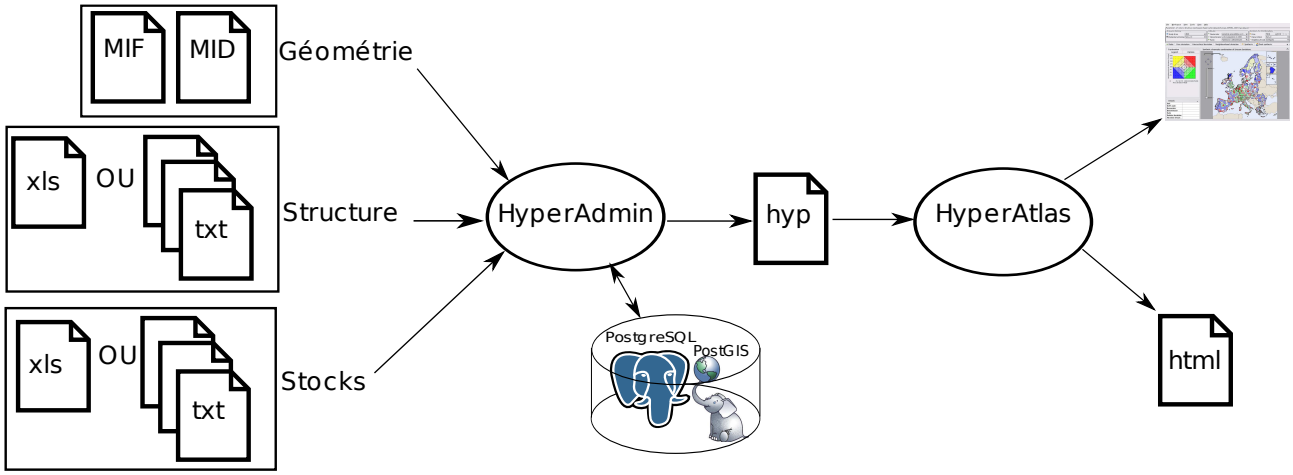


FIGURE 1.1 – Entrées et sorties de HyperAdmin et HyperAtlas.

Les sous-sections suivantes présentent le concept d'analyse territoriale multiscalaire telle qu'elle est proposée dans HyperAtlas et HyperAdmin. Une plus conséquente description de ces logiciels sera donnée dans le chapitre 8.

1.2.1 HyperAtlas

L'analyse territoriale multiscalaire est une méthode de recherche proposée par Claude Grasland et Liliane Lizzi. La méthode repose sur l'hypothèse selon laquelle l'étude d'une unité territoriale n'a de sens que comparée à ses voisines et mise en perspective à différentes échelles géographiques, en prenant en considération à la fois les caractéristiques des territoires voisins et les différentes entités territoriales de plus ou moins grandes dimensions auxquelles elle appartient [GL04b].

Le terme « multiscalaire » souligne la possibilité donnée à l'utilisateur d'évaluer une ou plusieurs données thématiques suivant différents contextes.

Considérant qu'une entité territoriale peut appartenir à plusieurs territoires de dimensions plus étendues, la méthode considère ainsi l'agrégation et à la désagrégation des éléments constitutifs des territoires.

Dans HyperAtlas, l'ATM consiste à prendre en compte pour chaque entité élémentaire trois types d'écarts : l'écart à un contexte général, l'écart à un contexte territorial et l'écart à un contexte spatial.

A noter que la version française de l'interface utilisateur d'HyperAtlas a jusqu'en novembre 2010 souffert de l'anglicisme inaproprié « déviation » pour le terme anglais *deviation*. La correction par l'utilisation du terme « écart » ayant été apportée depuis peu, certaines copies d'écran dans ce mémoire peuvent laisser figurer ce faux-ami. Les termes utilisés pour les trois contextes de références ayant été l'objet de maintes discussions et changements successifs, le tableau 1.1 récapitule les différentes terminologies utilisées en anglais et en français jusqu'à celle arrêtée en novembre 2010 en s'appuyant sur [DJP+09].

	Langue	**Terme arrêté**	**Dénominations précédentes**
1	anglais	*general deviation*	*global deviation*, *free deviation*
	français	écart général	déviation globale, déviation libre
2	anglais	*territorial deviation*	*medium deviation*, *hierarchical deviation*
	français	écart territorial	déviation medium, déviation hiérarchique
3	anglais	*spatial deviation*	*local deviation*, *neighbourhood deviation*
	français	écart spatial	déviation locale, déviation de voisinage

Table 1.1 – Historique des terminologies.

Pour réaliser une analyse multiscalaire, l'utilisateur constitue donc d'abord le ratio de deux indicateurs statistiques choisis aux numérateur et dénominateur. Un écart consiste à mesurer le rapport entre :

- le ratio de chaque unité territoriale d'une aire d'étude et à un niveau de maillage choisis ;
- le ratio des unités appartenant au contexte de référence pour les mêmes indicateurs.

Ces écarts sont calculés sous la forme d'un indice en pourcentage pour lequel 100 est la valeur pivot, la valeur de l'espace de référence. Pour une région donnée, un écart de 200 indique que la valeur du ratio pour cette région est le double de la moyenne du contexte de référence.

L'originalité de l'ATM telle qu'elle est implémentée dans HyperAtlas est donc d'obtenir trois vues différentes sur les mêmes données en fonction d'un espace de référence variable. Selon Philippe Waniez [Wan10], auteur du logiciel Philcarto (voir section 4.1 page 19), l'avancée scientifique est réelle et permet enfin de prendre en considération l'idée selon laquelle « *la réalité apparaît différente en fonction de l'échelle d'analyse* ».

1.2.2 HyperAdmin

HyperAdmin est un logiciel d'intégration de données permettant à un utilisateur de générer des fichiers `.hyp`* exploitables ensuite par HyperAtlas.

L'interface utilisateur de HyperAdmin est un clone de HyperAtlas : mêmes panneaux de paramètres et mêmes onglets cartographiques. La différence entre les deux logiciels réside dans la disponibilité dans HyperAdmin du menu « Assistant création de projet » (*Wizard*, en anglais : l'interface de HyperAdmin n'est disponible qu'en anglais).

Sous l'angle de l'interface graphique, la création d'un jeu de données passe donc par l'activation de ce menu « Assistant », ouvrant une fenêtre modale avec laquelle l'utilisateur va interagir pour fournir ses fichiers d'entrée, et visualiser, au fur et à mesure des différentes étapes du processus, des messages indiquant le bon déroulement des opérations ou, au contraire, les erreurs survenues et parfois bloquantes.

Quatre étapes principales constituent le processus de création d'un nouveau projet de jeu de données :

1. l'utilisateur indique le répertoire de son disque dans lequel il aura placé les fichiers d'entrée attendus par HyperAdmin (fichier structure, fichier statistique et fichiers de géométrie) ;
2. HyperAdmin associe les unités du plus bas niveau de maillage à leur géométrie et aux valuations statistiques des stocks, puis agrège ces données jusqu'au niveau supérieur ;
3. si l'utilisateur le désire, HyperAdmin calcule les matrices de contiguïté entre les unités territoriales, étape la plus longe et coûteuse en ressources, nécessitant le recours, si le nombre d'unités élémentaires dépasse le millier, aux calculs topologiques du SGBD (Système de gestion de bases de données) PostgreSQL et de son extension PostGIS, pour la gestion des types géométriques ;
4. si tout s'est bien passé, HyperAdmin propose à l'utilisateur de sauvegarder le projet dans la base de données, ou sous la forme d'un fichier `.hyp` qu'il peut enregistrer sur son disque.

1.3 Le programme ESPON

L'acronyme ESPON [4] trouve un équivalent en français dans l'acronyme ORATE, pour Observatoire en Réseau de l'Aménagement du Territoire Européen.

Le programme ESPON est soutenu et financé par les Etats membres de l'UE (Union Européenne) et des Etats partenaires comme l'Islande, le Liechtenstein, la Norvège et la Suisse. Lancé en 2006, le programme s'ouvre pour sa version 2013 aux états candidats et voisins.

Le programme vise une compréhension commune des tendances et des perspectives en matière de développement territorial pour l'UE et ses états voisins. Un des objectifs du programme consiste à mettre sur pied un système durable pour l'observation du territoire européen, à stocker et à diffuser les connaissances, les données statistiques et l'information géographique.

Dans le cadre du programme ESPON 2013, le groupe de recherche HyperCarte intervient sur plusieurs fronts :

4. *European Spatial Planning Observation Network, Territorial Development and Cohesion* http://www.espon.eu (consulté le 8 mai 2010).

- le projet *ESPON 2013 Database* ;
- le projet *ESPON HyperAtlas Update*.

L'objectif du projet *ESPON 2013 Database* est de collecter, d'unifier, d'harmoniser et de diffuser les données statistiques issues de sources variées. Le système doit faciliter la combinaison d'indicateurs et le suivi des évolutions tant sur les données géographiques que statistiques. Le projet a pour vocation de considérer différentes échelles : intercommunalités ou NUTS* par exemple, puis à terme des nomenclatures variées comme les UMZ (*Urban Morphological Zones*), la nomenclature des bassins versants ou des nomenclatures mondiales à partir des bases de l'ONU (Organisation des Nations Unies).

La base de données ESPON vise donc à fournir une SDI (*Spatial Data Infrastructure*, infrastructure de données spatiales) évolutive pour des données et leurs métadonnées associées. Tout ou presque varie :

- les indicateurs statistiques : leur nom, la méthode de calcul et, bien sûr, les valeurs pour les unités ;
- les fournisseurs de données ;
- les unités territoriales : leur nom, leur géométrie. Les unités peuvent par exemple fusionner entre elles, ou au contraire se diviser, ou encore disparaître.

Le projet *ESPON HyperAtlas Update* a, quant à lui, pour objectif de proposer des outils d'analyse spatiale à partir des formes sociales du territoire européen. A partir de données statistiques (population active, population totale, etc.), la notion d'« hyper-cartes » consiste à proposer différentes représentations de la distribution spatiale d'un phénomène social. Le logiciel *ESPON HyperAtlas* doit être tout à la fois d'un usage simple pour le non spécialiste désirant réaliser des cartes, mais aussi proposer des outils d'analyse complexe pour l'expert en géographie et sciences sociales.

Historiquement, une première version du logiciel HyperAtlas a été développée et livrée dans le cadre du programme ESPON 2006. Reconduit par les membres de l'Union Européenne, le programme ESPON 2013 et son comité de direction *ESPON Monitoring Committee* décide en 2009 de financer les évolutions sur le logiciel HyperAtlas, décrites dans le cahier des charges, objet du chapitre suivant.

Chapitre 2

Cahier des charges

2.1 Les enjeux

Le cœur de la mission répond aux besoins émis par ESPON quant à des évolutions attendues et contractées sur les logiciels HyperAtlas et HyperAdmin, décrites dans la section 2.2.

Le projet *ESPON HyperAtlas Update* est donc l'occasion de l'enrichissement des logiciels par un ensemble de nouvelles fonctionnalités.

A noter que le groupe de recherche HyperCarte reste propriétaire des sources des logiciels et entend pouvoir diffuser ces derniers dans d'autres contextes que celui du programme ESPON.

La section 2.3 page 13 complète donc le cahier des charges du projet *ESPON HyperAtlas Update* par la conception d'une architecture permettant la génération de versions personnalisées en fonction du client final.

La section 3.2 page 16 précise finalement le contenu de ce mémoire en tenant compte du calendrier, des enjeux et perspectives sous-jacents à la fois au projet *ESPON HyperAtlas Update*, et au projet *ESPON Database*.

2.2 Le contrat *ESPON HyperAtlas Update*

Cette section reprend les éléments du cahier des charges contracté entre l'unité de coordination ESPON et le groupe de recherche HyperCarte vis-à-vis des évolutions attendues sur le logiciel HyperAtlas livré en 2006.

Le contrat spécifie l'ensemble des attentes sur un calendrier de douze mois à compter du 1er mars 2010. La période est découpée en trois durées de six, trois et trois mois, à l'issue desquelles sont posés des jalons, respectivement nommés échéances intermédiaire, pré-finale et finale. Les délivrables attendus pour chacune de ces échéances sont exposés à la suite de la description de l'ensemble des évolutions à fournir. Afin de les référencer plus facilement dans la suite de ce document, ces différentes évolutions sont présentées ci-dessous précédées d'un titre identifiant :

Applet : ESPON souhaite une version client-serveur du logiciel HyperAtlas sous la forme d'une *applet* Java. Cette évolution consiste donc à rendre disponible l'ensemble des fonctionnalités du logiciel HyperAtlas depuis un navigateur Web standard.

Intégration du temps : en plus des dimensions spatiales et thématiques, les jeux de données doivent intégrer une dimension temporelle. Depuis l'interface, l'utilisateur doit disposer d'un outil pour visualiser l'évolution des indicateurs au cours du temps.

Extension des zones d'études : cette évolution consiste à proposer à l'utilisateur de définir de nouvelles zones d'études à partir d'unités territoriales sélectionnées sur une carte. Ces unités territoriales doivent être contiguës, et appartenir au même niveau de maillage.

Mode expert : HyperAtlas propose jusqu'ici trois cartes statistiques représentant les écarts relatifs à des contextes respectivement général, territorial et spatial (voir le nom des écarts dans le tableau 1.1). Le mode expert consiste à proposer, sur demande de l'utilisateur, de nouveaux outils d'analyse spatiale et statistique :

Redistribution : en mode expert, l'application doit proposer trois nouvelles cartes permettant de visualiser les redistributions pour les contextes général, territorial et spatial.

Nouveaux indices statistiques : en mode expert, le logiciel doit également fournir le calcul et la visualisation de l'écart type, du coefficient de variation, de l'indice de Hoover, du coefficient de Gini, et la représentation de la courbe de Lorenz.

Synthèse binaire : HyperAtlas propose jusqu'ici une carte de synthèse représentant les unités de l'aire d'étude suivant un mode ternaire, en fonction de leur positionnement relatif aux trois contextes. La synthèse binaire consiste à présenter les unités de l'aire d'étude en croisant non plus les valeurs de trois écarts, mais en permettant à l'utilisateur de choisir deux écarts. Une partition en diamant permet de synthétiser les valeurs et positions relatives des unités vis-à-vis des deux écarts choisis.

HyperAdmin : la disponibilité d'un outil d'intégration des données : le logiciel HyperAdmin devra générer à partir de données géographiques et statistiques des fichiers `.hyp`*.

Nouveaux jeux de données : la mise à jour des fichiers `.hyp`* tenant compte de l'espace européen EU27+4, capables de prendre en compte entre cinq et dix indicateurs autant que possibles pertinents pour ESPON, sur plusieurs années.

Interface : l'interface d'HyperAtlas doit être adaptée et réorganisée conformément aux nouvelles directives et règles fixées par ESPON quant aux respects de l'ergonomie, du style des pages et composants, de la mise en page des cartes exportées. Le logiciel doit explicitement intégrer les conditions d'utilisation. En outre, la taille de la carte doit être élargie, les noms des principales villes doivent apparaître en passant la souris sur la carte, les codes des NUTS* doivent compléter le tableau des détails, une information condensée des metadonnées doit apparaître dans les boîtes de sélection d'indicateurs.

Manuel utilisateur : cette évolution consiste à fournir une aide en ligne améliorée, plus interactive, constituée d'exemples.

Dualité d'utilisation : finalement, le logiciel doit constituer un outil facile d'accès aux non-spécialistes, et offrir des outils d'analyse complexe plutôt destinés aux utilisateurs avancés.

A noter que la liste précédente concerne l'aspect informatique du projet. Comme mentionné dans la présentation du groupe de recherche HyperCarte section 1.2 page 6, l'intérêt de l'association entre les différentes disciplines géographie, informatique, sciences sociales, constitue un avantage pour répondre de façon unitaire à un cahier des charges nécessitant des expertises dans ces différents domaines. Aussi la liste complète des tâches à effectuer pour répondre aux évolutions attendues sur ce projet a-t-elle été répartie en mars 2010 entre les équipes RIATE et STEAMER du groupe HyperCarte.

Le contrat du projet *ESPON HyperAtlas Update* stipule trois jalons nommés intermédiaire, pré-final et final, pour lesquels les délivrables attendus sont listés dans le tableau 2.1. A noter que les tâches associées incombent pour les unes aux géographes de l'équipe RIATE (colonne « Géo » du tableau), pour les autres à l'équipe STEAMER (colonne « Info »).

Délivrables	Géo	Info	Date échéance
Jalon intermédiaire			
Nouveaux jeux de données	*		31 août 2010
Proposition de nouveaux jeux de données	*		
Proposition de carte régions non contiguës	*		
Proposition d'interface graphique	*	*	
Proposition d'outils statistiques	*	*	
Jalon pré-final			
Version béta complète de HyperAtlas Web		*	31 décembre 2010
Version béta de HyperAdmin		*	
Version béta du manuel utilisateur	*	*	
Questionnaire d'évaluation	*		
Jalon final			
Version finale de HyperAtlas Web		*	28 février 2011
Version finale de HyperAdmin		*	
Version finale du manuel utilisateur	*	*	
Résultat du sondage d'évaluation du logiciel	*		
Rapport concis du travail effectué	*	*	
Documentation *ad hoc*	*	*	

TABLE 2.1 – Echéancier des délivrables pour le projet ESPON.

2.3 Le projet HyperAtlas v2

La section « Enjeux » (section 2.1 page 11) avance le fait que le groupe de recherche HyperCarte souhaite rester indépendant du programme ESPON et pouvoir proposer ses logiciels à d'autres éventuels clients. En 2008, à l'occasion de son stage d'ingénieur CNAM, Raphaël Thomas produit ainsi quatre versions personnalisées du logiciel HyperAtlas [Tho08] :

- *ESPON-HyperAtlas* pour le programme ESPON ;
- *EEA-HyperAtlas* pour l'Agence Européenne de l'Environnement ;
- *EP-HyperAtlas* pour le Parlement Européen ;
- *Nordic-HyperAtlas* pour NordRegio [1].

Ces versions se différencient essentiellement par le style de l'interface graphique, le nom de l'application, les logos, un jeu de données intégré par défaut personnalisé. La confidentialité des données statistiques d'origine implique l'acceptation par l'utilisateur de termes et conditions d'utilisation *ad hoc*, il/elle doit ainsi accepter une licence avant l'initialisation du logiciel. La rédaction de la licence est donc le résultat d'un accord entre le groupe de recherche HyperCarte et le client. De la version 1.2.6 (avril 2008) à la version 1.2.9 (novembre 2009), les différentes distributions proposées sur le site Web du groupe de recherche HyperCarte [2] sont également enrichies au fur et à mesure de nouvelles fonctionnalités..

Certaines versions de ces distributions d'HyperAtlas sont téléchargeables gratuitement sur le site, on parlera de « version standard ». D'autres distributions plus confidentielles sont également téléchargeables depuis la même page mais à condition de fournir un nom d'utilisateur et un mot

1. Nordic Centre for Spatial Development. http://www.nordregio.se/ (consulté le 18 mai 2010).
2. HyperCarte Réalisations - HyperAtlas http://hypercarte.imag.fr/realisations.hyperatlas.html (consulté le 30 septembre 2010).

de passe délivrés par le groupe de recherche.

La disponibilité et la diffusion d'une version standard constituent donc une stratégie de communication visant à susciter des demandes, des échanges, des opportunités de développement futur. Collectivités locales, enseignants, étudiants statisticiens ou géographes constituent un potentiel important d'utilisateurs.

En conséquence, le groupe de recherche HyperCarte tient à profiter des évolutions apportées aux logiciels dans le cadre de projets comme ESPON pour les rendre disponibles dans la version standard du logiciel.

La mise en place d'un système de gestion des configurations client ou standard constitue donc un élément clé du cahier des charges. Ce point est référencé dans ce document comme le « paramétrage de l'application ».

En termes de visibilité et de communication, la mise à jour du site Internet du groupe de recherche HyperCarte est également un point du cahier des charges à ne pas négliger.

Chapitre 3

Organisation du mémoire

Tenant compte à la fois du cahier des charges présenté dans le chapitre précédent et du calendrier de la convention de stage, ce chapitre récapitule les problèmes posés et présente le contenu de ce mémoire.

3.1 Calendrier

Ce mémoire s'appuie essentiellement sur le travail réalisé pendant les neuf mois de stage ingénieur CNAM. Comme indiqué sur la figure 3.1, la convention de ce stage a été signée entre le CNAM et le LIG pour la période de février à octobre 2010. Néanmoins, les tâches auxquelles j'ai été affecté pendant cette période ont dû être adaptées aux contraintes du calendrier auquel devait répondre l'équipe STEAMER. Bien qu'engagé dans l'équipe sur le projet *ESPON Database* depuis Décembre 2009, la majorité du travail décrit dans ce mémoire correspond à une partie du travail effectué dans le cadre d'un contrat relatif aux logiciels du groupe de recherche HyperCarte et étalé sur un an, de mars 2010 à mars 2011.

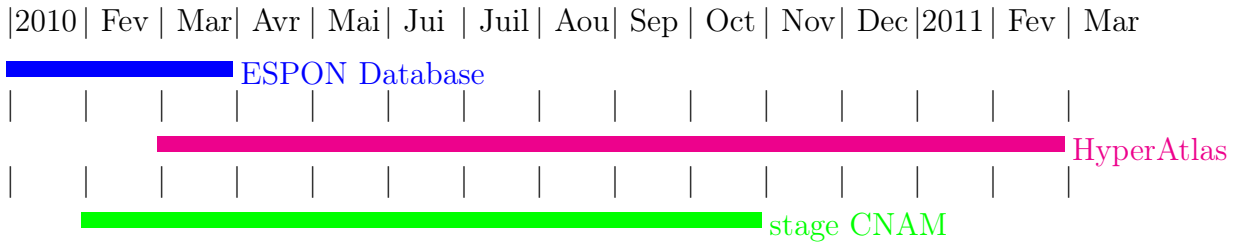


FIGURE 3.1 – Intégration du stage ingénieur dans le calendrier des projets STEAMER *ESPON Database* et *ESPON HyperAtlas Update*.

L'accomplissement des tâches du cahier des charges du projet *ESPON HyperAtlas* ayant été évalué à une charge de travail d'un an, la partie réalisation de ce mémoire ne rapporte donc qu'un sous-ensemble des évolutions du contrat complet, listées section 2.2 page 11.

Compte tenu du calendrier de la figure 3.1, de la charge de travail liée à la livraison de la livraison intermédiaire du 31 décembre 2010 (voir tableau section 2.1 page 13), la rédaction de ce mémoire a été finalisée début janvier 2011 pour présenter l'état du projet au 31 décembre 2010.

3.2 Plan du mémoire

Ce mémoire est composé de quatre parties principales. Cette première partie présentait le contexte d'exécution, les parties prenantes et le cahier des charges. La seconde partie propose un état de l'art de quelques modèles et solutions Internet existantes dédiées à la visualisation spatio-temporelle des statistiques. La troisième partie, « Réalisation », présente le travail effectué pendant ce stage. Enfin, la dernière partie conclut, synthétise, et dresse un bilan du travail effectué pendant ce stage. Un état des lieux vis-à-vis du cahier des charges et les perspectives quant à la suite des projets du groupe de recherche HyperCarte sont finalement abordés.

Deuxième partie

Etat de l'art

Chapitre 4

Méthodes et outils de visualisation

Ce chapitre présente tout d'abord quelques pistes pour la représentation des données spatiales et temporelles. La diversité des solutions actuelles dans le domaine de la géomatique est ensuite illustrée dans les section suivantes par quelques exemples d'outils en ligne intégrant la dimension temporelle.

L'objectif de cette partie est d'une part d'appréhender quelques points clés du sujet, d'en déceler les difficultés, et d'autre part, d'en extraire de l'inspiration pour des évolutions du modèle de données de HyperAdmin et de leur représentation dans HyperAtlas.

4.1 Philcarto

Philcarto[1] est un logiciel gratuit évoluant depuis 2001, et maintenu par son auteur, Philippe Waniez, du groupe de recherche en analyse de l'information territoriale, le GRANIT [Wan10].

L'objectif de l'auteur est de rendre accessibles des méthodes de cartographie et d'analyse des données géographiques. Le logiciel est destiné à l'apprentissage de la géographie dans l'enseignement secondaire et supérieur, afin d'offrir aux utilisateurs un outil facile d'accès, simple d'emploi tout en étant efficace sur le plan scientifique.

C'est essentiellement parce que Philcarto propose un nombre important de possibilités quant à la cartographie thématique et l'analyse des données qu'il a été retenu dans cet état de l'art. Ce logiciel permet de présenter ici une idée des possibilités de représentations de statistiques (valeurs ou classes) sur une carte. Comme précisé dans le manuel utilisateur, le logiciel permet également de tenir compte du temps pour des taux ou des indices de variation, tant que le maillage de l'espace est considéré comme invariant.

En outre, la version 5 de Philcarto, réécrite sous la plate-forme *Microsoft Visual Studio Express 2005*, est enrichie de nouvelles fonctionnalités dont la méthode d'analyse territoriale multiscalaire. S'appuyant sur les concepts du groupe de recherche HyperCarte, l'auteur propose une carte de synthèse des écarts pouvant être réalisée suivant différentes méthodes : analyse exploratoire des écarts considérés par paires sur un graphique bivarié, analyse en composantes principales et classification automatique sur les trois niveaux simultanément, analyse en composantes principales, ou classification ascendante hiérarchique.

La diversité des cartes pouvant être produites à l'aide de Philcarto ne répond qu'à une partie de l'information géographique, ne sont en effet prises en compte que les localisations suivant des

1. Logiciels Philcarto, Phildigit [en ligne] http://philcarto.free.fr/Philcarto.html (consulté le 9 juillet 2010).

coordonnées en deux dimensions x et y, points, lignes et polygones. Philcarto ne reconnaît pas les localisations dites topologiques.

Les cartes choroplèthes* pour représenter les variables discrètes ou continues et les cartes en cercles proportionnels* destinées à représenter les quantités et effectifs sont les deux types de cartes les plus utilisés. Les cartes à disques proposent des variantes comme les cartes en cercles proportionnels colorés : elles représentent simultanément des quantités ou des effectifs par le cercle, et des valeurs continues ou discrètes exprimant des pourcentages, ou des mesures, par la couleur. Philcarto permet aussi la superposition d'une carte en cercles proportionnels à une carte choroplèthe. Les inconvénients de ce type de carte cumulent ceux des deux types utilisés. De plus, la taille des disques peut gêner la lecture de la couleur du polygone, bien que ce problème peut être contourné en laissant vide la couleur des disques.

Les cartes en demi-cercles affrontés permettent de représenter simultanément deux variables portant sur des quantités ou effectifs, à l'exclusion des pourcentages taux mesures ou catégories. Ce type de carte permet notamment de comparer deux catégories de population, ou bien la même catégorie à deux moments donnés. La légende indique un échantillon pour calibrer les demi-cercles et deux caissons colorés pour identifier les deux variables. Comme le montre l'exemple de la figure 4.1, ce type de carte nécessite un diamètre assez grand pour visualiser les différences entre les demi-cercles affrontés, ainsi que des plages de valeurs propices pour les deux variables.

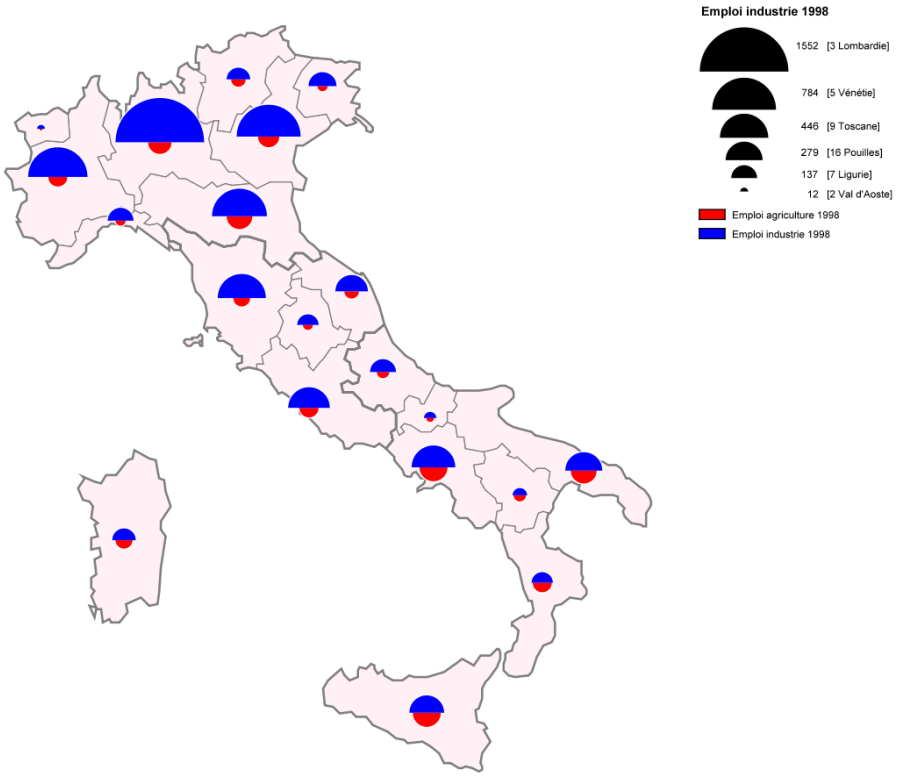


FIGURE 4.1 – Exemple de carte à demi-cercles affrontés, emploi agricole (rouge) et industriel (bleu) dans les régions d'Italie. Source : [Wan10].

Les cartes en diagrammes à secteurs (camemberts) permettent de représenter la composition d'une population en fonction de plusieurs catégories. Le cercle représente la somme des effectifs des catégories. La légende indique le calibrage des cercles représentés sur la carte et des caissons colorés permettent d'identifier chacun des secteurs. L'inconvénient majeur des cartes en diagrammes à

secteurs tient en la quasi impossibilité d'une lecture globale de la carte.

Philcarto permet également de réaliser des cartes en plages de niveaux. Ces cartes reposent sur l'hypothèse que la variation des mesures statistiques présente une certaine continuité dans l'espace (en météorologie par exemple). A partir d'un maillage ponctuel est dérivé un carroyage formant un maillage surfacique carré. A chaque centre de carré est estimée une valeur, on revient à une carte choroplèthe où les limites des classes sont les limites des niveaux basés sur un espacement régulier. La carte obtenue permet de s'affranchir des limites administratives. A partir de la même hypothèse, Philcarto permet aussi de réaliser des cartes en surfaces de tendances : la décomposition du phénomène continu dans l'espace s'opère en composantes d'échelles se rapportant à des espaces de plus en plus réduits, selon un carroyage où les valeurs des carrés sont évaluées à l'aide de fonctions polynomiales.

Pour obtenir un effet de densité, Philcarto permet de réaliser des cartes en semis de points. Celles-ci représentent des quantités ou des effectifs par un nombre de points proportionnel. Sur un maillage surfacique, le nombre de points est préalablement calibré, puis les points sont répartis aléatoirement sur le polygone. L'inconvénient majeur de ce type de carte est que les points ne sont pas localisés sur un lieu précis d'enregistrement des quantités. La carte en semis de points colorés est une variante permettant de représenter simultanément des quantités ou des effectifs par des points et des valeurs numériques exprimant des pourcentages ou des mesures par les couleurs de ces points.

Pour représenter des relations entre des points de l'espace, Philcarto permet de réaliser des cartes de liens dites aussi en « oursins », dont un exemple est donné figure 4.2. Les relations sont figurées par des lignes joignant les points deux à deux. Les lignes peuvent être coloriées en fonction d'une variable nominale. Ce genre de carte est principalement utilisé pour exprimer une organisation spatiale polarisée.

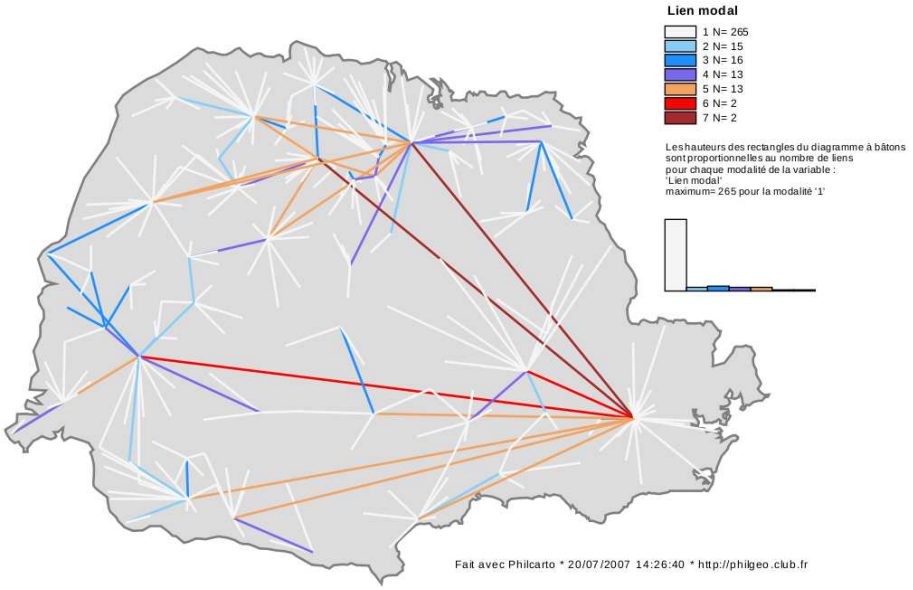


FIGURE 4.2 – Exemple de carte de liens entre les lieux centraux hiérarchisés dans l'Etat du Parana au Brésil. Source : [Wan10].

Pour information, Philcarto étend également ses possibilités à la réalisation de cartes de réseaux et de cartes de gravitation. Les cartes de réseaux sont basées sur des lignes représentant des tronçons, les effectifs (flux par exemple) sont représentés par des épaisseurs de lignes proportion-

nelles ou des variations de couleurs. Les cartes de gravitation donnent une représentation simplifiée des aires d'attraction. Une masse (population par exemple) est affectée à chaque lieu central. Un maillage ponctuel positionne les lieux centraux à l'aide de coordonnées. Un modèle de gravitation permet le tracé des lignes d'iso-attraction entre les centres deux à deux.

En conclusion, Philcarto est un logiciel riche et très didactique pour l'apprentissage de la géographie et la réalisation de cartes. La grande variété des types de cartes possibles est impressionnante, mais la complexité des options nuit à sa facilité d'utilisation pour un utilisateur non averti. La disponibilité du logiciel en un `.exe` limite sa portabilité à Windows en tant qu'application de bureau.

Preuve de ses capacités, notons que Philcarto a souvent été utilisé par les géographes du groupe de recherche HyperCarte pour la réalisation de cartes dans différents rapports des programmes ESPON, notamment dans le cahier des charges, pour illustrer les évolutions à implémenter dans HyperAtlas (cartes à disques colorés).

4.2 Les cubes espace-temps

Parmi les techniques de représentation des phénomènes ayant une emprise à la fois spatiale et temporelle, Adrienko et son équipe implémentent en 2004 une solution basée sur la technique du cube espace-temps [GAA04]. Cette solution a été retenue dans cet état de l'art pour la réflexion qu'elle suscite vis-à-vis de la sémiologie graphique, et des interrogations nécessaires en préalable d'une analyse spatio-temporelle.

Les auteurs s'intéressent plus particulièrement à l'analyse spatiale d'évènements passagers, comme par exemple les tremblements de terre ou l'observation d'animaux rares. Aussi leur technique cherche-t-elle à répondre à la question de la distribution de tels évènements dans l'espace et le temps. Elle s'inspire d'une vue graphique en trois dimensions proposée au début des années 1970 par Hägerstand [Hä70] : les deux dimensions de la base d'un cube représentent l'espace, la troisième dimension du cube étant réservée au temps. Le cube espace-temps représente en quelque sorte le temps comme une dimension spatiale supplémentaire. La figure 4.3 représente trois possibilités d'observations à l'aide d'un tel cube :

- en I, les déplacements d'une personne sur une journée, la graduation temporelle sur l'axe z s'étend de 5h10 en bas à 19h55 en haut ;
- en II, la ligne verticale indique le temps pendant lequel la personne est restée au même endroit ;
- en III, le prisme espace-temps STP (*spatio-temporal prism*) indique les positions qui peuvent potentiellement être atteintes dans un intervalle de temps donné PPS (*potention path space*), la projection de ce prisme sur la base du cube montre l'aire potentiellement atteinte PPA (*potential path area*).

Dans le cadre d'une analyse spatiale, il est avant tout nécessaire de considérer les données à disposition et les questions pour lesquelles on attend une réponse. Ainsi se doit-on de distinguer dans l'ensemble des données disponibles :

- les données spatiales, elles se distinguent les unes des autres en considérant généralement une des deux typologies suivantes :
  1. une typologie basée sur les primitives géométriques du point, de la ligne, du polygone ou du volume ;
  2. une typologie basée sur deux dimensions orthogonales :

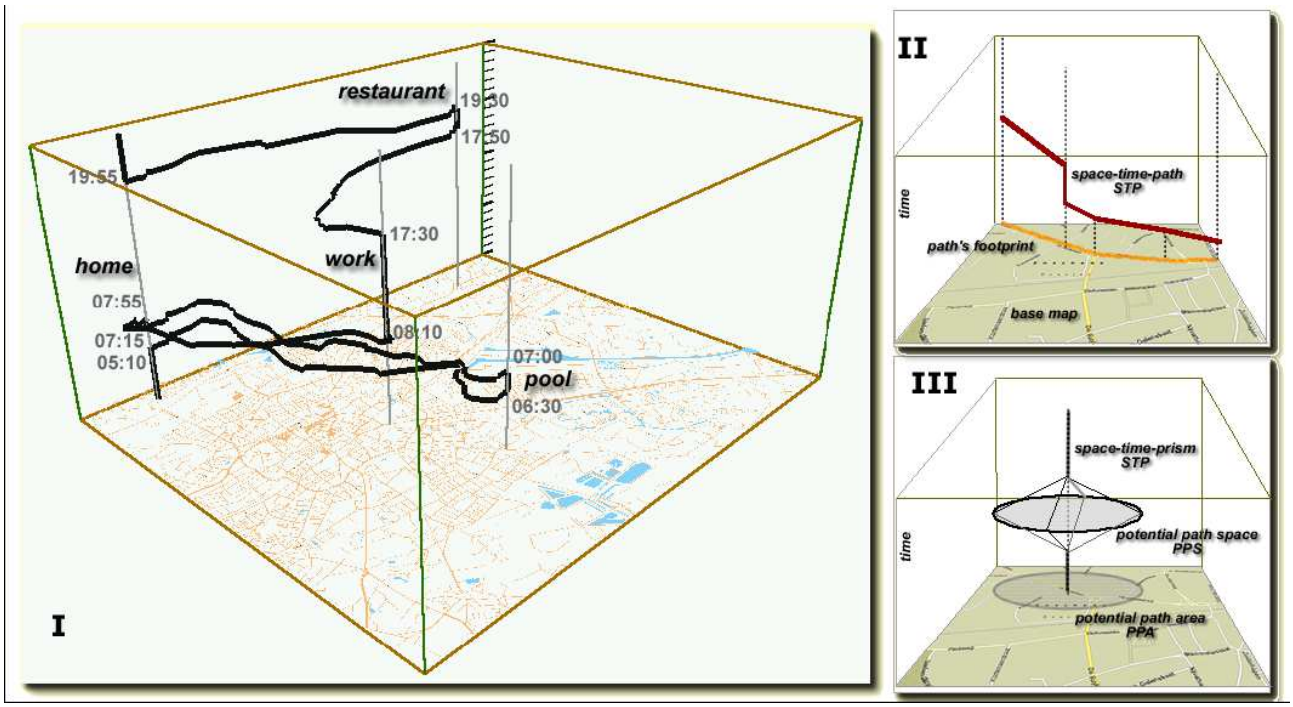


FIGURE 4.3 – Application du cube espace-temps pour visualiser les déplacements d'une personne au cours d'une journée. Source de l'image : [Kra03].

(a) une dimension considérant la contiguïté spatiale qui peut être discrète, la localisation du phénomène est alors isolée, ou au contraire continue, le phénomène est alors rencontrée partout ;

(b) une dimension considérant la dépendance spatiale : cette dimension indique si le phénomène change de manière graduelle, ou au contraire de façon abrupte.

– les données temporelles : les phénomènes géographiques peuvent changer au cours du temps de façon existentielle (apparition, disparition), changer de propriétés spatiales (taille, endroit, forme, altitude par exemple) ou de propriétés thématiques ;
– les données thématiques : suivant le le type de leurs attributs, on distingue les classes nominales, ordinales, selon un intervalle ou un ratio, une valeur numérique.

Selon Peuquet [Peu94] et Bertin [Ber67], les questions auxquelles on souhaite répondre lors d'une analyse spatio-temporelle se répartissent en trois classes de questions-types, suivant les données dont on dispose et le résultat que l'on souhaite observer. On considère ainsi l'équation $a+b \rightarrow x$ où x est l'inconnue :

– « quand et où » pour déterminer le « quoi » ;
– « quand et quoi » pour déterminer le « où » ;
– « où et quoi » pour déterminier le « quand ».

Outre ces questions-types, Jacques Bertin propose également de répartir les questions d'une analyse selon trois niveaux élémentaire, intermédiaire et général. Le niveau élémentaire considère par exemple un instant, un objet ou une localisation. Les niveaux intermédiaire et général considèrent des caractéristiques plus générales d'un phénomène, comme par exemple sa distribution dans l'espace ou sa fréquence dans le temps. Dans le cas d'analyses spatio-temporelles, le schéma $a + b \rightarrow x$ de questions-types peut donc encore se raffiner en fonction du niveau d'entendement élémentaire, intermédiaire ou général des paramètres a et b.

Les travaux d'Adrienko spécialisent la question « quand + où → quoi » dans laquelle le « quand » et le « où » considèrent le niveau général. La question à laquelle il souhaite répondre peut être formulée ainsi : quel est le modèle de la distribution spatiale et temporelle des évènements et leurs propriétés thématiques ?

La représentation des évènements sur une carte est indispensable à la reconnaissance d'évènements récurrents. Différentes méthodes permettent de telles observations. Une carte statique peut par exemple représenter tous les évènements indépendamment du temps, ou seulement pour un instant ou un intervalle de temps donné. Des étiquettes peuvent être placées sur les endroits où ont eu lieu les évènements, ou des couleurs différentes peuvent représenter des instants ou intervalles de temps distincts. La lisibilité est acceptable pour une faible quantité d'évènements seulement, et tant que le chevauchement des couleurs n'entraîne une occlusion des symboles.

La juxtaposition de plusieurs cartes permet de comparer des clichés correspondant à des instants ou des tranches de temps différentes. L'utilisateur peut ainsi être invité à interagir pour choisir des périodes.

L'animation sur des cartes dynamiques ou les requêtes demandent à l'utilisateur une concentration importante pour intégrer mentalement les changements, outre les difficultés à choisir les intervalles pertinents, et à observer un phénomène sur un territoire entier. A noter que les valeurs numériques peuvent être agrégées et surexposées par des symboles proportionnels aux valeurs. L'agrégation temporelle peut par exemple considérer un jour de la semaine particulier, ou une date, et être représentée par un diagramme à barres. Cette méthode d'agrégation pour un des types de données (spatial ou temporel), entraîne néanmoins la perte de l'information sur l'autre type de données. L'agrégation n'est donc intéressante que dans le cas où l'analyse consiste à observer une distribution dans le temps ou alors dans l'espace, mais chacune des dimensions prises séparément.

L'idée du cube espace-temps est donc une réponse au besoin de représentation simultanée et globale de l'espace et du temps.

L'équipe d'Adrienko explique en 2004 avoir pu exploiter cette idée du cube espace-temps à l'élaboration d'un modèle de zones à forte probabilité de tremblement de terre, en observant, grâce au cube, des séquences d'évènements survenues de façon rapprochée, à la fois dans le temps et dans l'espace : des clusters. Dans [GAA04], les auteurs indiquent avoir cependant eu des difficultés à repérer ces clusters : la principale difficulté consiste pour l'utilisateur à sélectionner la fenêtre temporelle qui permette d'observer la concentration du phénomène...

En conclusion, le concept des cubes espace-temps ouvre le champ des possibles quant à la représentation de la combinaison à la fois de l'espace et du temps. Bien que porteuse, la représentation en trois dimensions représente un pas difficilement franchissable pour l'HyperAtlas actuel, d'autant plus mal adaptée compte tenu des objectifs du logiciel qui doit permettre la visualisation d'indicateurs et écarts sur de grandes étendues comme l'Europe.

4.3 SOLAP

SOLAP, acronyme de *Spatial On-Line Analytical Processing*, définit à la fois le concept et la technologie combinant les capacités cartographiques des systèmes d'information géographique (SIG) aux outils d'aide à la décision OLAP.

Issu du domaine de l'informatique décisionnelle, Edgar Frank Codd introduit en 1993 le terme OLAP qui « *désigne une catégorie d'applications et de technologies permettant de collecter, stocker,*

traiter et restituer des données multidimensionnelles à des fins d'analyse » [Lup10]. Le concept de « processus d'analyse interactif pour l'aide à la décision » OLAP, désigne à la fois des bases de données multidimensionnelles et un modèle virtuel de représentation des données appelé *hypercube*. Le système OLAP offre des opérateurs pour explorer les données (forage, « remontage », forage latéral, pivot), il fournit aux décideurs un outil rapide et simple pour effectuer des analyses complexes.

Les outils OLAP offrent à l'utilisateur la représentation des données sous forme de diagrammes et tableaux, mais il leur manque généralement la puissance cognitive de la représentation logique et intuitive des données géographiques sous une forme cartographique. Pour répondre à ce besoin, une équipe de recherche de l'Université Laval Québec, dirigée par Dr Yvan Bédard, développe depuis 1995 une technologie décisionnelle, le processus interactif d'analyse spatiale, plus communément appelé Spatial OLAP ou SOLAP. Yvan Bédard définit un outil SOLAP comme « *une plate-forme visuelle facile et rapide pour l'analyse spatio-temporelle des données selon une approche multidimensionnelle, à plusieurs niveaux d'agrégation, via un affichage cartographique, tabulaire ou en diagramme statistique.* » [BRP06].

Comme la figure 4.4 le résume, cette solution cherche à combiner la puissance de l'OLAP aux capacités cartographiques des SIG.

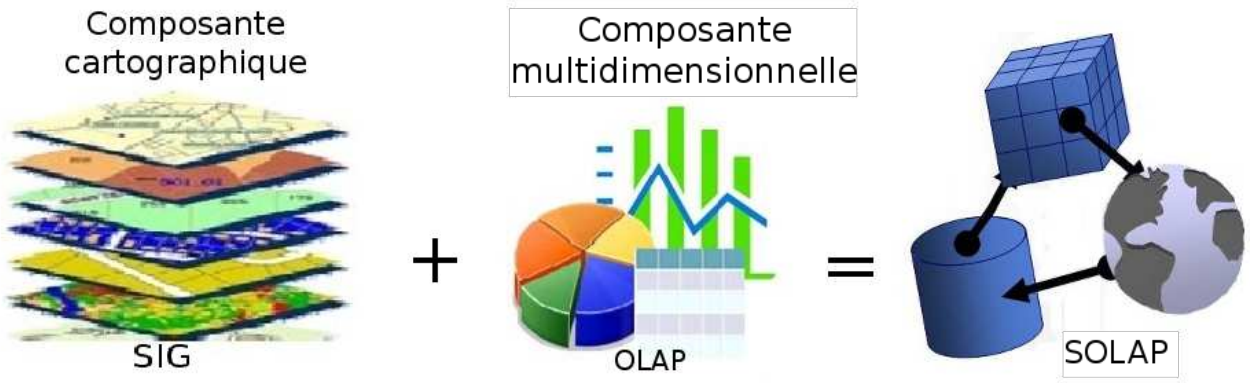


FIGURE 4.4 – L'équation SOLAP [LR09].

Deux des objectifs majeurs de la technologie SOLAP sont d'offrir un temps de réponse rapide, de l'ordre de quelques secondes, et, pour être adoptée par les décideurs et analystes, de proposer une interface graphique simple, sans connaissance particulière d'un langage de requête.

Un système OLAP s'intègre typiquement dans une architecture à trois niveaux : une base de données multidimensionnelle (l'entrepôt de données), un serveur OLAP, un client OLAP. Le serveur fournit une vue multidimensionnelle des données, il gère les données agrégées et détaillées, et l'ensemble des métadonnées. Le client OLAP offre une interface utilisateur simple, sans connaissance d'un langage de requête pour explorer les données, il affiche les données sous forme de graphiques ou de tableaux. La rapidité de l'OLAP repose sur le principe d'agrégation des membres d'une même dimension. Une dimension correspond à un thème de l'analyse (lieu, produit, temps, par exemple). Les opérations d'agrégation les plus fréquentes sont la somme et la moyenne, elles permettent de calculer rapidement les valeurs associées aux positions parents. Par exemple, on peut décomposer la dimension « temps » suivant une hiérarchie à trois niveaux : les années au niveau 1, les trimestres au niveau 2, les mois au niveau 3. L'agrégation des données multidimensionnelles permet de mémoriser le résultat des calculs dans la base et d'augmenter ainsi les performances lors de l'interrogation. En contrepartie, l'agrégation nécessite un espace de stockage conséquent.

L'*hypercube*, combinaison des différentes dimensions, permet donc d'organiser le stockage agrégé des données selon une granularité (un niveau de détails) plus ou moins importante. Le concept OLAP propose des mécanismes dits « opérateurs », pour naviguer dans les hiérarchies. Grâce à leur facilité d'utilisation et leur rapidité, l'utilisateur se concentre sur son analyse et non sur les moyens utilisés à cette fin.

SOLAP ne vise pas à remplacer les SIG, mais à proposer des fonctions supplémentaires en supportant la structure multidimensionnelle. Un SIG et un SOLAP se distinguent essentiellement par la convivialité et la vitesse d'exécution des requêtes. Le bénéfice du SOLAP sur l'OLAP repose sur la possibilité d'agrégation de données spatiales, qu'elles soient descriptives, géométriques ou mixtes.

En plus des diagrammes et tableaux alphanumériques disponibles nativement dans le système OLAP, le système SOLAP va donc permettre à l'utilisateur une nouvelle représentation des données sous la forme de cartes. Grâce à l'intégration des dimensions spatiales, l'utilisateur dispose des opérateurs OLAP (forage, remontage , forage latéral et pivot) pour interagir avec ces données, par des simples clics de souris. Les différentes représentations sont synchronisées entre elles.

Aboutissement de plusieurs prototypes développés initialement à l'Université de Laval au Québec, le logiciel commercial JMap Spatial OLAP, maintenu ensuite par la société Kheops devenue K2 Geospatial [2], est la solution commerciale SOLAP aujourd'hui certainement la plus avancée.

Développée en Java, disponible en version Web (une *applet* pour les navigateurs standard et JSon pour les navigateurs mobiles), la figure 4.5 montre l'interface graphique de JMap Spatial OLAP, proposant à l'utilisateur différentes représentations d'un phénomène sous forme de cartes, tableaux et diagrammes.

L'offre SOLAP Open Source n'est pas en reste, on retient notamment le projet Spatialytics [3], issu en partie des travaux du professeur Thierry Badard de l'Université Laval Québec. Présenté à la conférence FOSS4G 2010 (*Free Open Source Software for Geospatial*), le projet fusionne les solutions suivantes :

- GeoKettle : outil ETL (*Extract Transform Load*) d'intégration des données sous la forme d'une application de bureau intégrant les données géographiques ;
- GeoMondrian : héritant du moteur OLAP Mondrian, GeoMondrian permet de stocker des données spatiales dans la structure *hypercube*, il intègre le type spatial dans la structure d'*hypercube* OLAP ;
- SOLAPLayers : clients interagissant avec le serveur GeoMondrian à l'aide de la librairie olap4j [4], la version 2 se base sur ExtJS/GeoExt pour une intégration facilitée dans divers cadriciels dédiés à la présentation de données sous forme tabulaires, diagrammes ou cartes.

Ces trois logiciels gratuits et Open Source forment une suite complète d'outils d'informatique décisionnelle géospatiale, ou *GeoBI* (*GeoSpatial Business Intelligence*), dont l'architecture est illustrée figure 4.6. Selon Thierry Badard, une version stable de la suite devrait être disponible en janvier 2011 [Bad10].

En conclusion, le SOLAP introduit la dimension spatiale dans le monde décisionnel pour essentiellement bénéficier en retour des opérateurs OLAP et d'outils de représentations multiples.

2. K2 GEOSPATIAL - Systèmes d'informations géographiques en ligne [en ligne] http://www.k2.geospatial.com (consulté le 5 janvier 2010).

3. *Open Source Geospatial Business Intelligence* [en ligne] http://www.spatialytics.org (consulté le 5 janvier 2010).

4. Open Java API for OLAP

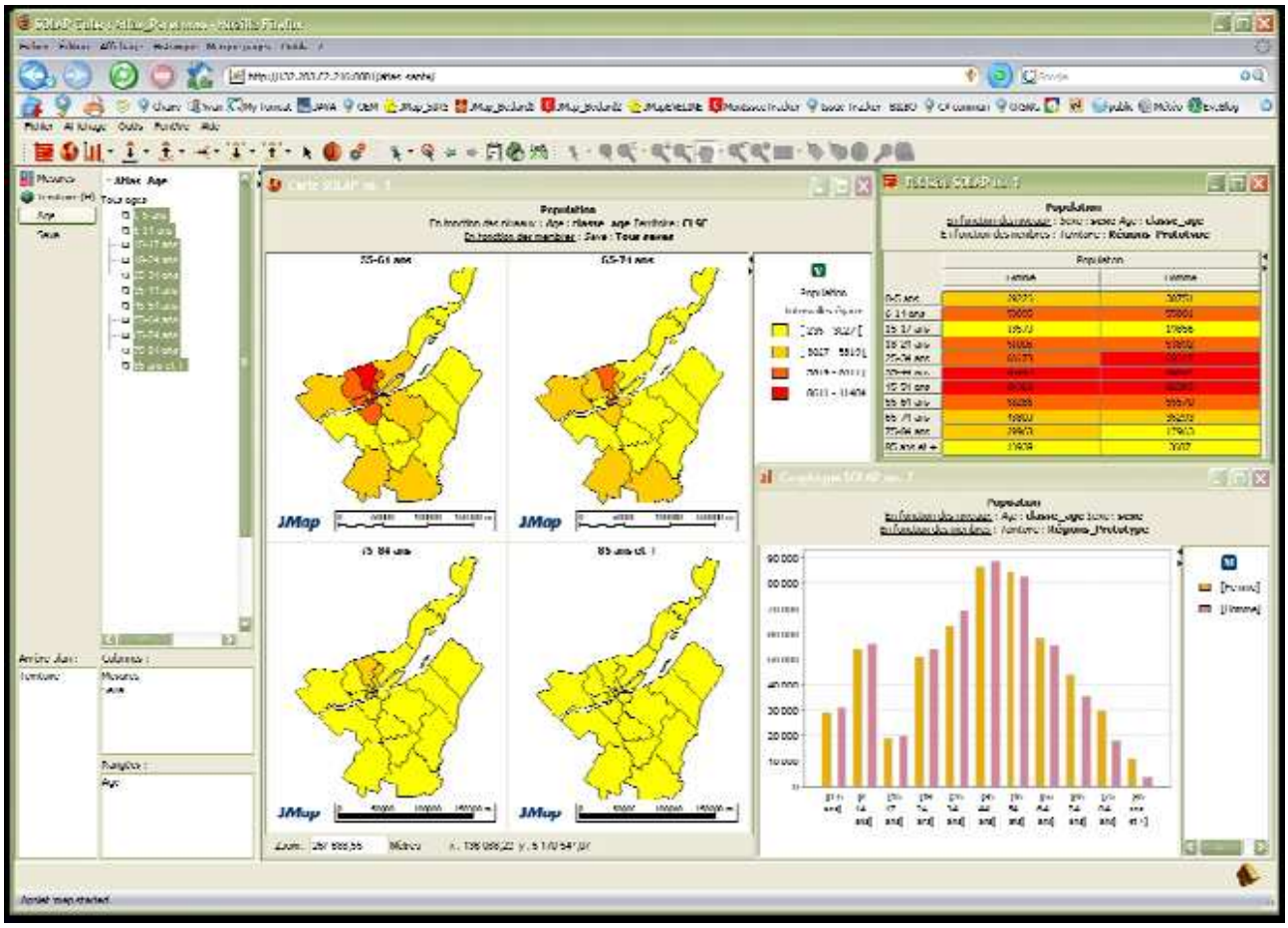


FIGURE 4.5 – L'interface de JMap Spatial OLAP. Source : [Poi08].

STEAMER a investi sur plusieurs travaux de recherche dans ce domaine, mais la maturité des outils SOLAP Open Source ne permet pas encore aujourd'hui de parier sur leur intégration assurée dans un délai comme celui imparti pour le projet *ESPON HyperAtlas Update*.

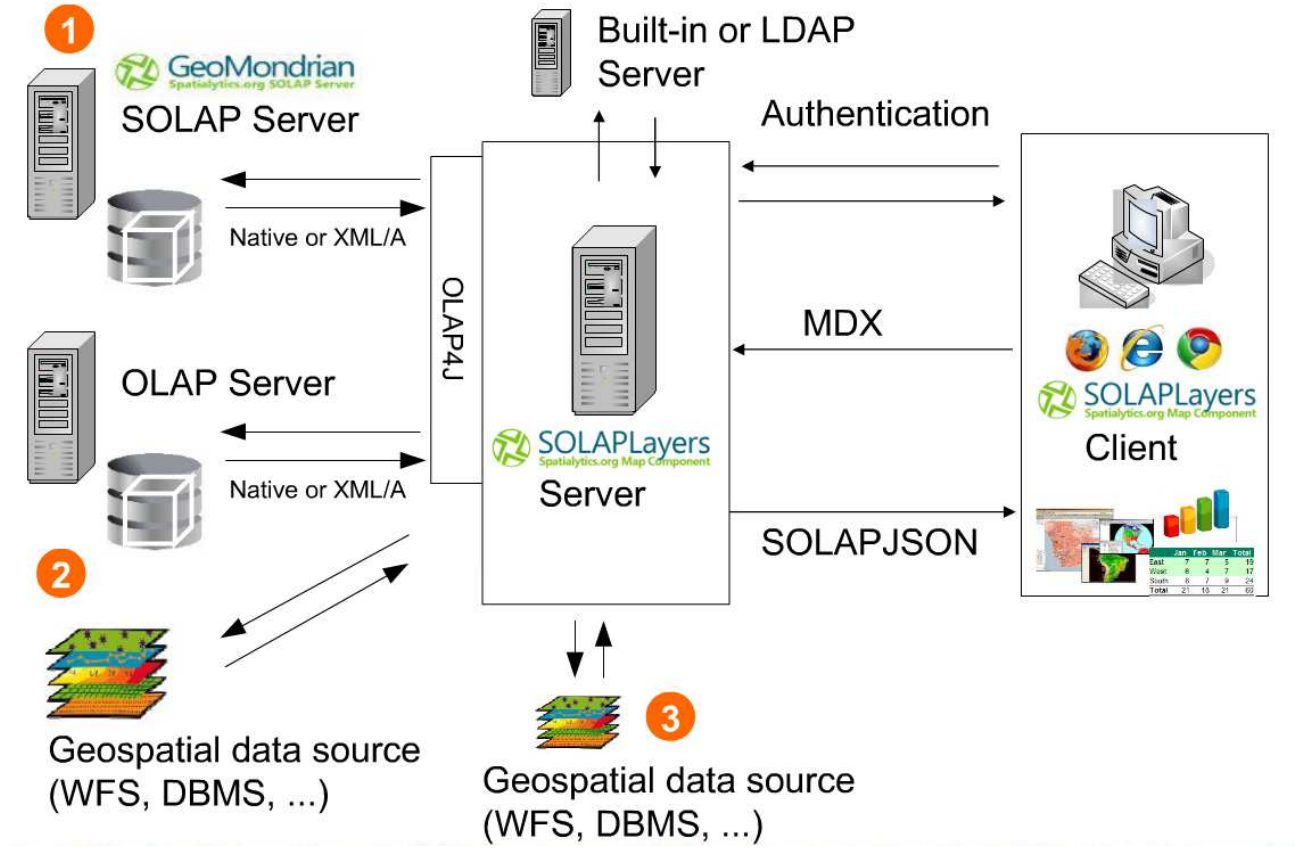


FIGURE 4.6 – Architecture SOLAPLayers 2.0. Les requêtes multidimensionnelles géospatiales ne sont disponibles qu'avec la le serveur SOLAP, en 1. Les connexions avec les sources de données OLAP ou services Web géographiques (en 2), ou des systèmes transactionnels (en 3) sont néanmoins possibles. Source : [Bad10].

Chapitre 5

Quelques solutions spatio-temporelles sur le Web

La représentation du temps dans les systèmes d'information géographique est une thématique de recherche actuelle [PGVO+09]. Ce chapitre présente plusieurs solutions.

5.1 Vizzuality

Vizzuality [Viz10] est une entreprise madrilène qui conçoit et développe des *Rich Internet Applications* (applications Internet riches). Les projets développés s'appliquent à expérimenter des outils de visualisation et d'analyse des données, essentiellement dans les domaines liés à la biodiversité, comme par exemple les espaces protégés marins et terrestres, la migration des espèces.

Une démonstration en ligne[1] propose notamment de suivre la part cumulée d'information disponible dans la base de données du GBIF Data Portal[2] au sujet de l'étourneau sansonnet. Outre les outils cartographiques classiques de zoom et de déplacement de la carte, l'utilisateur est invité à déplacer un curseur temporel situé en bas de la fenêtre. De 1880 à 2010, la figure 5.1 montre qu'en déplaçant la souris au dessus du curseur, l'application affiche des informations complémentaires provenant de Wikipedia. Quand il/elle déplace le curseur, la carte affiche la quantité cumulée d'information disponible pour la date courante.

Cette application illustre la mise en œuvre de différentes techniques de manipulation des images *raster* : les données proviennent d'une base de données (GBIF) au 14 décembre 2009 et affichées à l'aide du couplage de Google Maps et de Flash. Présents à la conférence internationale FOSS4G 2010[3], les auteurs précisent sur leur blog (`http://biodivertido.blogspot.com/`, consulté le 7 septembre 2010) qu'ils partageront le code et les détails techniques bientôt...

Cette démonstration est intéressante pour son esthétisme. Le couplage à Google Maps offre les capacités cartographiques classiques et la démonstration fournit un exemple simple (l'utilisateur n'a pas le choix dans le type de données à afficher) mais efficace d'interaction entre l'utilisateur et les données spatio-temporelles.

On regrette néanmoins que Vizzuality se présente comme une entreprise Open Source mais

1. *Expansion of the european starling* [en ligne] `http://vizzuality.s3.amazonaws.com/starling/index.html` (consulté le 7 septembre 2010).

2. Le portail Global Biodiversity Information Facility propose un accès libre et gratuit à des données sur la biodiversité. [en ligne] `http://data.gbif.org/welcome.htm` (consulté le 7 septembre 2010).

3. *Free Open Source Software for Geomatics*, du 6 au 9 septembre 2010 à Barcelone.

FIGURE 5.1 – Capture d'écran d'une application Vizzuality. Cette figure illustre les possibilités graphiques de Flash couplé à Google Maps pour la visualisation sur une carte de l'évolution de données au cours du temps à l'aide d'un curseur temporel.

qu'elle ne livre pas son code en même temps que ses démonstrations. En outre, au niveau thématique, si l'échelle et un titre apparaissent sur la fenêtre, l'utilisateur ne dispose d'aucune légende.

5.2 GapMinder

Gapminder [RRRR10] est une société fondée à Stockolm en 2005 et rachetée par Google en 2007. Gapminder fournit des services et réalise des projets en collaboration avec des universités, l'ONU (Organisation des Nations Unies), des agences publiques et des organisations non gouvernementales. L'activité principale de Gapminder consistait initialement au développement du logiciel Trendalyzer, renommé depuis 2006 Gapminder World [4]. Ce logiciel propose un service Web animé et interactif pour l'observation de séries temporelles relatives aux statistiques. L'équipe travaille aujourd'hui à la mise à jour et à la mise à disponibilité des séries temporelles, à la production de vidéos, de graphiques, de diagrammes et de présentations Flash sous la forme de graphiques

4. Gapminder World http://www.gapminder.org/world/ (consulté le 7 septembre 2010).

statistiques animés. L'intention de Gapminder est aujourd'hui de proposer un « réservoir de faits » (*fact tank*) et de « dévoiler la beauté des statistiques pour une vision du monde basée sur des faits » (*unveiling the beauty of statistics for a fact based world view*).

Les figures 5.2 et 5.3 montrent successivement les deux onglets de l'application pour observer des données statistiques selon un graphique ou une carte.

FIGURE 5.2 – Onglet graphique de Gapminder. L'axe des abscisses considère le revenu par personne, l'axe des ordonnées considère l'espérance de vie à la naissance. La taille des cercles correspond à la population totale de chaque pays référencé. Le panneau en bas à droite indique la valeur correspondante pour la bulle au dessus de laquelle la souris se place. Les données correspondent ici à l'année sélectionnée sur le curseur temporel en bas de la fenêtre, ici, 2009.

Cette application a été retenue dans cet état de l'art pour la qualité de l'animation affichée en cliquant le bouton *Play* à gauche du curseur temporel, disponible sur les deux onglets graphique et carte. L'animation commence à l'année où est placé le curseur et se déroule jusqu'en 2009. Sur la carte, les cercles apparaissent, grossissent ou diminuent en fonction d'une discrétisation du temps où une unité correspond à une année. Sur le graphique, les cercles (un par pays) changent de taille et de position. La vitesse de l'animation est paramétrable, les indicateurs peuvent être modifiés.

Le logiciel est essentiellement adapté, et c'est le but recherché, à l'observation des tendances globales : ainsi ne peut-on pas zoomer sur la carte, ni observer un niveau de maillage autre que celui du pays. L'utilisateur est contraint d'observer les données disponibles dans l'application, il ne peut éditer ses propres statistiques. De plus, le logiciel ne propose pas de méthodes d'analyses statistiques avancées comme le calcul de moyennes, d'indices particuliers (Gini, Hoover, etc.), de

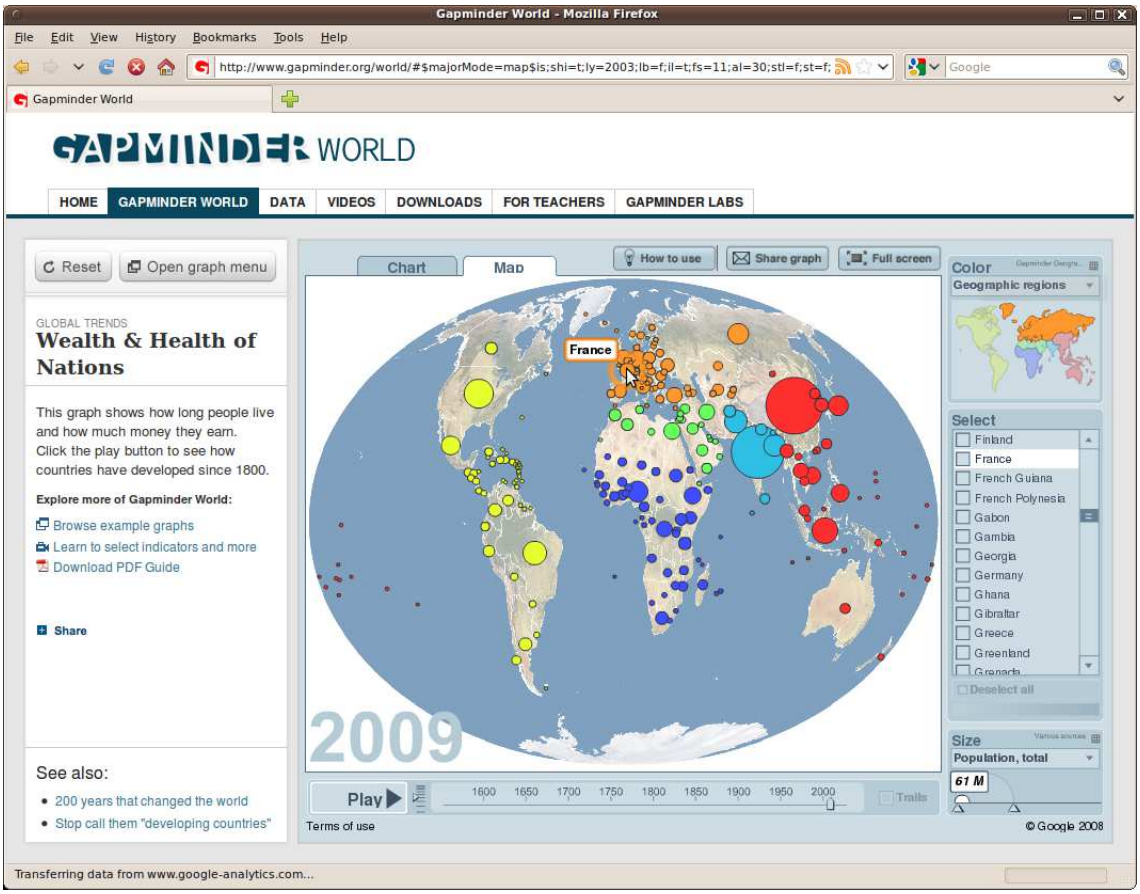


FIGURE 5.3 – Onglet carte de Gapminder. Cette carte à disques proportionnels indique pour chaque pays et à la date choisie la population totale. La couleur des disques indique la région géographique à laquelle chaque pays est rattaché.

ratio ou de l'écart.

Outre le service Web, l'application existe aussi en version application de bureau. Gapminder Desktop est gratuit et installable sur Windows, Mac et Linux. L'application requiert la technologie Adobe Air pour pouvoir exécuter des applications Flash . Après avoir téléchargé et installé l'application (testé avec succès sous Ubuntu), l'utilisateur peut ainsi observer des données sans connexion à l'Internet.

L'application de bureau a l'avantage de pouvoir sauvegarder les graphes et cartes selon un système d'onglets. En fermant et en réouvrant l'application, l'utilisateur retrouve les onglets qu'il avait sauvés. Si ce concept de sauvegarde des onglets personnalisés est proche de l'outil « Configuration » de HyperAtlas, l'utilisateur de Gapminder Desktop a l'avantage de pouvoir ouvrir plusieurs onglets à la fois, l'utilisateur peut ainsi passer en un clic d'une analyse à une autre.

La promotion de Gapminder Desktop insiste sur la possibilité de l'utiliser pour une présentation personnalisée des statistiques contenues dans l'application. Il n'existe malheureusement pas d'outil d'export pour sauver les graphiques individuellement, vers des fichiers image, ou comme l'outil « générer un rapport html » de HyperAtlas.

Enfin, bien que globalement impressionnant, l'utilisateur peut reprocher à Gapminder une certaine lenteur d'exécution : le passage de l'onglet carte à l'onglet graphique, ou inversement, et toute modification de paramètre nécessitent entre 10 et 15 secondes avant d'obtenir un résultat

visible. Bien qu'une barre de progression soit parfois affichée pour les plus longues opérations, avertissant ainsi l'utilisateur du coût de calcul de l'opération courante, la plupart des actions nécessitent un temps de latence où rien ne se passe à l'écran pendant plusieurs secondes.

5.3 Google Public Data Explorer

FIGURE 5.4 – Exemple d'export de Google Public Data Explorer sur un site personnel. Capture d'écran du site http://gemtice.blogspot.com/2010/03/de-la-cartographie-en-ligne-exploitable.html (consulté le 16 septembre 2010).

Google Public Data Explorer [5] est un site proposant la visualisation de différents indicateurs et thématiques sur des données publiques issues de plusieurs organismes comme la Banque Mondiale, l'OCDE (Organisation de coopération et de développement économiques), EUROSTAT.

Google exploite son API (*Application Programming Interface*) *Motion Chart gadget* et les possibilités du logiciel Trendalizer dont il est propriétaire pour permettre la visualisation des données selon un graphique en lignes, un diagramme en barre, une carte ou dans un repère (*Bubbles*). Le

5. *Google public data explorer in labs.* [en ligne] http://www.google.com/publicdata/home (consulté le 16 septembre 2010).

type de carte est déclinable selon les quatre possibilités de Google Maps, Plan, Satellite, Mixte ou Relief. Les représentations sont dynamiques, un curseur permet de voir l'évolution des données dans le temps et l'utilisateur peut choisir une période sur l'échelle du temps.

La figure 5.4 montre la possibité d'exporter une partie de l'outil et un jeu de données sur un site personnel. Le curseur temporel permet de voir évoluer la taille des disques en fonctions des valeurs des données. Les données manquantes sont représentées par des étoiles. En déplaçant la souris au dessus de la carte, la valeur de la région survolée (ici l'Estonie) est indiquée sur la droite (*No size data*).

La FAQ (foire aux questions) du site avertit l'utilisateur de l'état expérimental de l'outil. Certaines données manquantes peuvent laisser l'utilisateur perplexe devant un graphique ou une carte vides. Les données ne sont pour l'instant pas exportables, et l'utilisateur doit consulter le site du fournisseur pour les retrouver.

Néanmoins, l'outil est séduisant pour sa sobriété et sa simplicité d'utilisation. Quatre icônes permettent de passer d'une représentation à une autre. Une fois le fournisseur et le jeu de données choisis, le choix des indicateurs est organisé par catégories et sous-catégories. Chaque indicateur est accompagné d'une description fournissant un ensemble de métadonnées.

5.4 OECD eXplorer

L'OCDE (Organisation de coopération et de développement économiques, *OECD* en anglais), propose depuis 2008 un outil graphique interactif destiné à l'analyse des statistiques régionales de l'OCDE [OEC09a] : OECD eXplorer [OEC09c].

Le développement de OECD eXplorer est le résultat d'un coopération entre l'OCDE et le NCVA [NCV10] (*National Center for Visual Analytics*, Centre national pour l'analyse visuelle) de l'Université de Linköping en Suède. En novembre 2008 [JBL09], le Professeur Mikael Jern du NCVA met à disposition sur le site internet de l'OCDE une version préliminaire, une nouvelle version est disponible depuis mars 2009 pour répondre aux besoins exprimés par les utilisateurs.

La base de données régionales de l'OCDE [OCD09] constitue un ensemble de statistiques et d'indicateurs sur quelques 2000 régions de 30 pays. Elle englobe actuellement des séries chronologiques pour une quarantaine d'indicateurs dans des domaines tels que la démographie, les comptes économiques, le marché du travail et les thèmes sur la situation sociale et l'innovation dans les 30 pays membres de l'OCDE.

Les régions des pays membres de l'OCDE sont classées en deux niveaux territoriaux (TL pour *Territorial Level*) pour faciliter les comparaisons internationales :

- le niveau supérieur TL2 se compose de macro-régions ;
- le niveau inférieur TL3 se compose de micro-régions ;
- une distinction géographique est ensuite opérée au niveau TL3 pour marquer les régions essentiellement rurales, urbaines ou intermédiaires.

La base de données régionales de l'OCDE peut être affichée et analysée grâce à l'outil OECD eXplorer qui combine des cartes interactives et d'autres éléments graphiques.

OECD eXplorer est une instance personnalisée de la plate-forme eXplorer générique du NCVA. L'eXplorer est ainsi présenté comme une application Internet riche d'un ensemble d'outils d'analyse statistique et géographique supportant un large éventail de composants pour l'analyse, la génération d'hypothèses, la communication et la collaboration sur de larges jeux de données issus de sources variées. La page Web du NCVA présente quelques réalisations pour différents clients

comme l'OCDE, Eurostat, la ville de Göteborg ou l'organisme de statistiques suédoises SCB *Statistics Sweden*.

5.4.1 Vue générale

Quatre outils principaux de visualisation remplissent la fenêtre de l'application, ils sont représentés sur la figure 5.5. A l'ouverture de l'outil en ligne [OEC09c], le panneau inférieur PAC (*Parallel Axes Chart*) est minimisé. Chacun d'entre eux peut être redimensionné à la convenance de l'utilisateur.

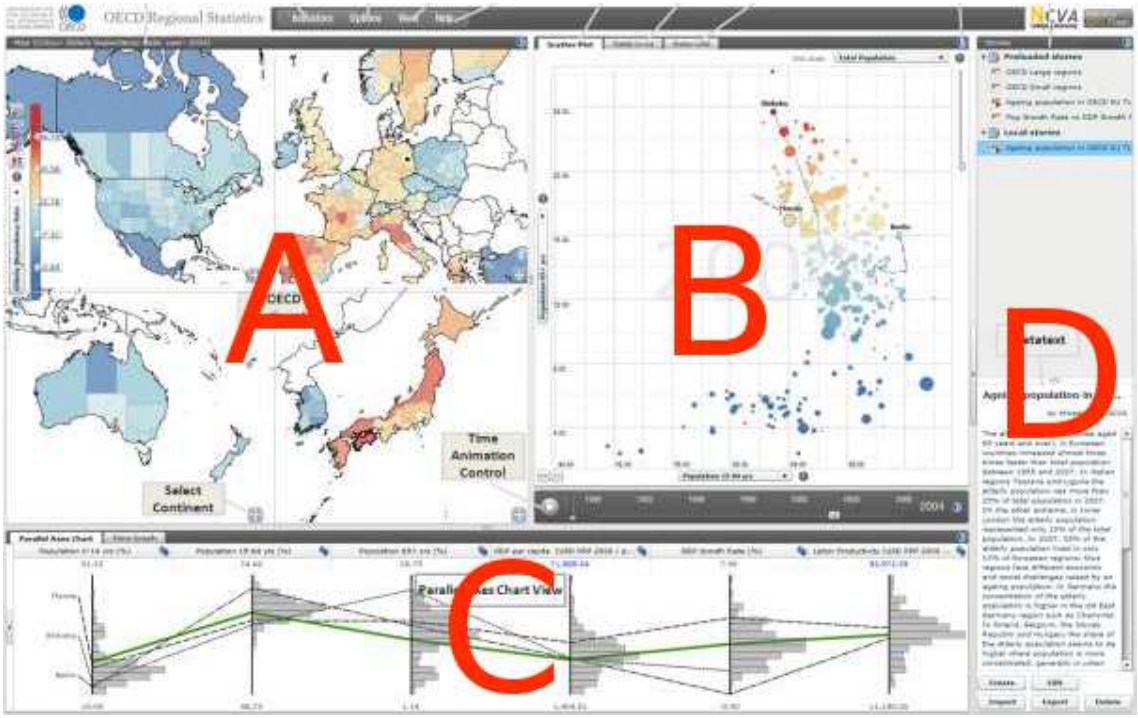


FIGURE 5.5 – La fenêtre de l'application est composée de quatre panneaux principaux : (A) une représentation cartographique selon une ou plusieurs cartes choroplèthes ; (B) la vue *Scatt Plot* (graphique de dispersion) aussi nommée *Bubble view*, vue graphique à bulles ; (C) le PAC *Parallel Axes Chart*, graphique à axes parallèles, aussi nommé le PCP *Parallele Coordinate Plot*, est optionnel et est iconifié à l'initialisation ; (D) un panneau *Story Telling Panel* fournissant du texte à propos de l'analyse en cours. Image réalisée à partir de [OEC09b].

Les trois panneaux carte, PAC et bulles sont synchronisés et dynamiques, ils partagent le même modèle de données et le même schéma de couleurs. La sélection d'une région par l'utilisateur sur une des vues met en valeur cette même région sur les autres vues.

5.4.2 Architecture

OECD eXplorer repose sur les outils GAV *GeoAnalytics Visualization toolkit* développés par le NCVA [NCV10]. GAV inclut des composants issus de la visualisation d'information InfoViz, de la visualisation géographique GeoViz et de la visualisation scientifique SciViz. Les composants GAV sont développés en technologie Microsoft C# et utilisent à l'origine la librairie graphique DirectX pour générer des applications de bureau.

GAV a été ensuite adapté pour un déploiement sur Internet en utilisant Adobe Flash et Adobe-Flex pour l'interface graphique. Programmée en language objet ActionScript d'Adobe, l'application Internet finale nécessite le greffon Adobe Flash Player version 10 installé sur le navigateur client.

L'architecture GAV est divisée en trois couches représentées sur la figure 5.6 :

- une couche d'accès aux données ;
- une couche de transformation (projection, dessin) et de filtre des données récupérées de la couche précédente ;
- une couche de composants visuels.

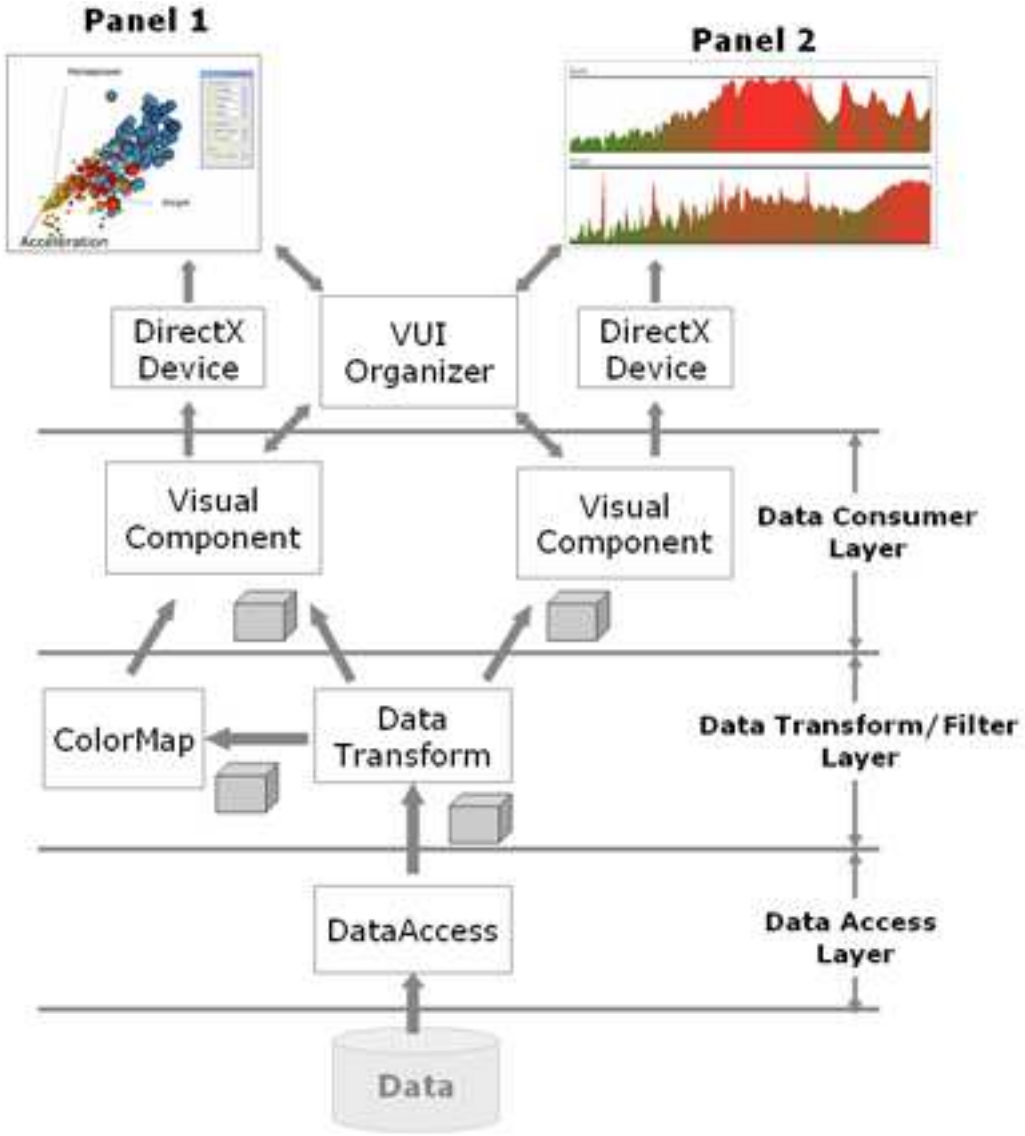


FIGURE 5.6 – L'architecture en trois couches de GAV : accès aux données, transformation/filtre, composants fonctionnels. Source : [NCV10].

Le stockage des jeux de données en entrée de GAV peut être représenté par un cube (figure 5.7) dont les trois dimensions permettent de manipuler indépendamment la composante spatiale (les régions), la composante attributaire (les indicateurs statistiques) et le temps (les années).

Le cube de données est au centre de l'architecture de l'eXplorer. Les services d'import export

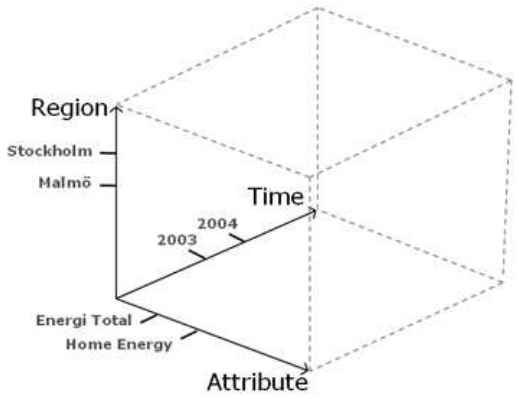


FIGURE 5.7 – Le modèle 3D des données dans l'architecture de GAV. Source : [NCV10].

chers aux concepteurs quant aux possibilités d'échange et de collaboration fournies par l'outil se déclinent actuellement suivant cinq modes représentés sur la figure 5.8 :

1. la base de données régionale de l'OCDE est chargée par défaut dans l'OECD eXplorer ;
2. une feuille de données Excel dans laquelle l'utilisateur aura saisi des données statistiques peut être également chargée. Les statistiques fournies doivent correspondre aux régions de l'OCDE ;
3. une interface SDMX (*Statistical Data and Metadata eXchange*) en collaboration avec la base OECD ;
4. un fichier XML *storyXml* précédemment sauvegardé depuis l'outil permet de partager et recharger une configuration d'analyse.
5. une fonction d'export permet de générer du code HTML pouvant ensuite être publiée sur des blogs ou des pages Web.

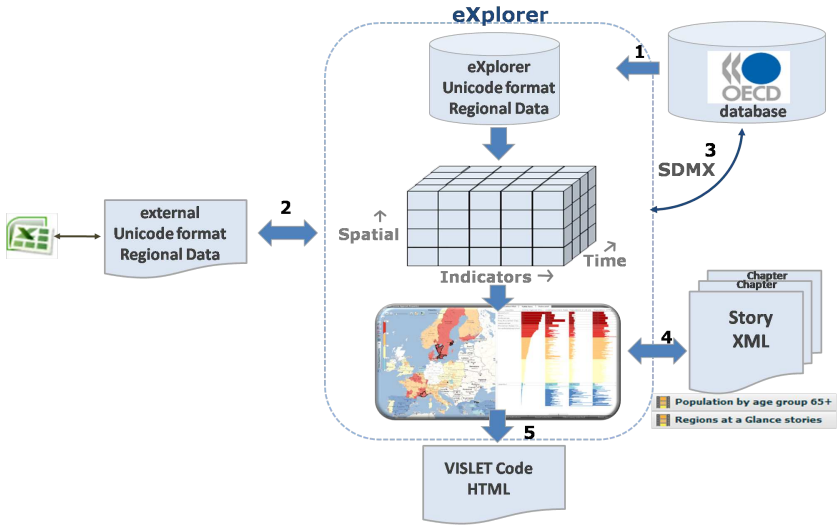


FIGURE 5.8 – Cette figure représente les cinq fonctions d'import export de OECD eXplorer. Source : [NCV10].

5.4.3 Principaux outils

OECD eXplorer permet à l'utilisateur un nombre important d'options et d'outils, cette section reprend les plus novateurs d'entre eux en s'appuyant sur le manuel utilisateur [OEC09b] :

- les couches GoogleMap sur les cartes ;
- la configuration des couleurs et des légendes (progressions discrète ou linéaire, édition de seuils, gestion des percentiles) ;
- vue des détails sur les unités territoriales (Menu - View - Selection Details) ;
- sélection des indicateurs, filtres dans la vue *scatter plot* ;
- sélection des indicateurs, configuration des échelles de valeur (filtre de valeurs extrêmes) dans la vue PAC ;
- animation temporelle sur les vues carte, PAC, *scatter plot* et un graphe temporel, sélection pas à pas des années ou en mode animation ;
- vue tabulaire des statistiques ;
- export vers un fichier XML d'une configuration d'étude (*snapshot*) aggrémentée éventuellement d'une analyse textuelle et des captures d'écran souhaitées. Le scenario de création d'une telle étude est représenté sur la figure 5.9.

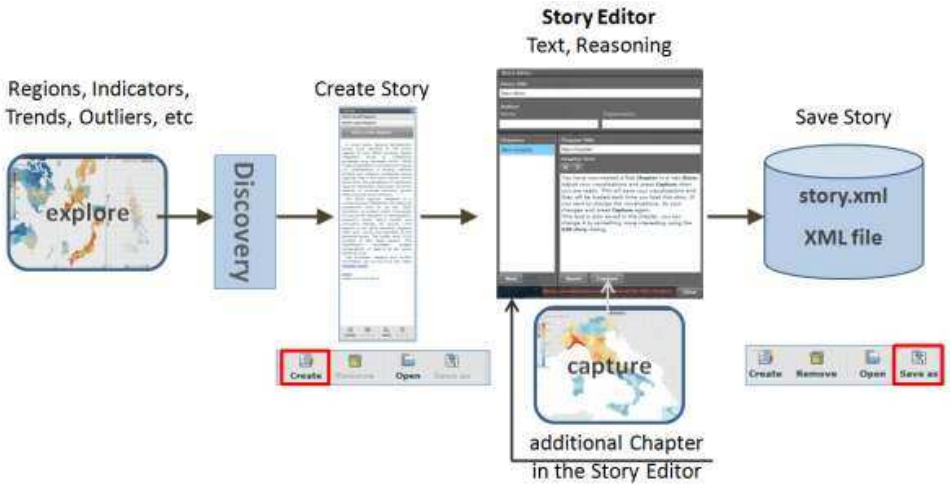


FIGURE 5.9 – Cette figure représente les étapes de création d'un fichier XML permettant de sauvegarder et d'échanger une analyse aggrémentée de commentaires. Le fichier sauvegardé sur le disque utilisateur peut être réouvert ultérieurement. Source : [OEC09b].

En conclusion, OECD eXplorer est une solution riche en possibilités. Peut-être cette richesse constitue-t-elle également son principal défaut : au démarrage, l'utilisateur pourrait s'attendre à une plus sobre et intuitive présentation, qu'il pourrait ensuite enrichir à sa convenance, progressivement. Si le paramétrage des outils est très avancé, il requiert en conséquence un recours systématique à la documentation. L'utilisation de cette solution nécessite donc une importante période d'apprentissage, il s'adresse globalement à des utilisateurs spécialisés. Ceux-ci auront par contre la possibilité de générer des rapports pour échanger et communiquer leurs analyses.

Chapitre 6

Synthèse de l'état de l'art

Le nombre de solutions pour la géo-visualisation de statistiques sur Internet augmente de jour en jour, traduisant un besoin réel de disposer d'outils de visualisation pour la quantité phénoménale de données emmagasinées dans les bases de données.

De plus en plus d'outils actuels accompagnent une représentation cartographique par des représentations tabulaires ou sous la forme de diagrammes et graphiques.

Ces outils proposent de plus en plus souvent à l'utilisateur un curseur temporel, pour visualiser l'évolution des données au cours du temps. Le graphique à bulles, par exemple dans les solutions GapMinder et OECD eXplorer, permet de visualiser d'une façon élégante l'évolution relative des unités, au prix, par contre, d'un paramétrage conséquent. Le paramétrage des filtres pour l'observation d'un phénomène et son analyse devient en effet vite nécessaire, pour ne pas être noyé dans la quantité d'information fournie par défaut. En outre, si la disponibilité d'un curseur temporel devient courante, elle dissimule à l'utilisateur des problèmes comme les évolutions des territoires ou des méthodes de calculs des indicateurs, et les données manquantes.

Malgré tous ces outils, l'analyse multiscalaire et les cartes de synthèse proposées par HyperAtlas semblent demeurer une exclusivité (hormis Philcarto).

Le tableau 6.1 propose une comparaison des différents outils Web étudiés dans cet état de l'art. Les notes, données en toute subjectivité, sur une échelle de 0 (non disponible) à 10 (excellent), traduisent l'impression générale de l'outil pour le critère courant. Les critères pris en compte sont décrits ci-dessous :

Technologie : indique la technologie utilisée pour le développement de l'outil.

Performance : réactivité aux actions de l'utilisateur.

Ergonomie : ce critère tient essentiellement compte de la facilité d'utilisation de l'outil.

Visualisation : quatre critères de visualisation prennent en compte la disponibilité d'outils de représentation des données selon les modes suivants :

- **Cartographie** : visualisation des données sur une carte ;
- **Tableaux** : visualisation tabulaire des données ;
- **Diagrammes** : visualisation des données dans des repères à deux axes ;
- **Bulles** : dans un repère, chaque entité géographique est positionnée suivant deux indicateurs (les axes), deux autres indicateurs peuvent être pris en compte en représentant chaque entité par un disque de taille ou de couleur différentes. En déplaçant un curseur temporel, on obtient un effet « bulles de savon ».

Curseur temporel : possibilité d'observer l'évolution des données dans le temps.
Import des données : possibilité donnée à l'utilisateur de visualiser ses propres données.
Export des données : possibilité de récupérer ou de réutiliser l'analyse courante.

Critère		**Solution**		
		OECD eXplorer	**GapMinder**	**Vizzuality**
Technologie		Flash	Flash	Flash
Performance		7	2	9
Ergonomie		2	5	8
V	Cartographie	7	7	7
I	Tableaux	0	1	0
S	Diagrammes	9	0	0
U	Bulles	10	10	0
Curseur temporel		10	10	10
Import des données		5	2	0
Export des données		5	0	0

Table 6.1 – Comparaison des outils Web présentés dans cet état de l'art.

Le tableau 6.2 compare finalement les outils étudiés avec HyperAtlas et HyperAdmin selon une typologie de bonus-malus.

Solution	**Bonus / Hyper-Atlas-Admin**	**Malus / Hyper-Atlas-Admin**
Philcarto	Diversité des cartes.	Public visé spécialiste. Application de bureau uniquement sous Windows. Nécessite des fonds de carte au format Adobe Illustrator.
SOLAP	Stockage multidimensionnel. Outils de forages, pivot, rapports, tableaux de bord.	Maturité : pas d'outil libre clé en main.
Vizzuality	Convivialité, facilité. Menus contextuels multimedia.	Un seul indicateur à la fois. Pas d'analyse.
GapMinder	Bulles	Performances, ergonomie. Plutôt destiné à la présentation qu'à l'analyse.
OECD eXplorer	Multiples diagrammes. Interactivité et options. Partage des analyses.	Performances. Ergonomie. Import des données. Interopérabilité. Apprentissage.

Table 6.2 – Comparaison des outils présentés avec HyperAtlas et HyperAdmin.

Pour conclure, notons que le contenu de cet état de l'art aurait certainement été différent s'il avait été écrit ne serait-ce que quelques mois plus tard. Les doctorants et chercheurs de l'équipe STEAMER suggèrent régulièrement de nouvelles ressources et solutions, parmi lesquelles, récemment :

- posté par Jérôme Gensel à l'équipe, les logiciels *Spatial Data Mining and Visual Analytics Lab* [en ligne] http://www.spatialdatamining.org/software (consulté le 24 janvier 2011) ;
- posté par Mouna Snoussi à l'équipe, une carte publiée par un service hongrois d'information en temps-réel sur les urgences, accidents, catastrophes et désastres : [en ligne] http://hisz.rsoe.hu/alertmap/index2.php (consulté le 25 janvier 2011).

Ces remarques visent à susciter la vigilance et la nécessité dans le domaine du spatio-temporel d'une veille technologique active.

Troisième partie

Réalisation

Chapitre 7

Démarche

Ce chapitre introduit la partie « réalisations » de ce mémoire en justifiant l'ordre choisi des chapitres suivants en fonction de la démarche suivie lors du stage. Les intentions de conception modulaire quant aux nouvelles fonctionnalités sont ensuite explicitées par un rappel des points clés et idées directrices d'une telle démarche.

La figure 7.1 représente schématiquement les grandes étapes de la démarche mise en œuvre pour répondre au cahier des charges et aux problématiques énoncées dans la partie « présentation » :

1. étude de l'existant, objet du chapitre 8 suivant ;
2. mise en place du cadre et de l'environnement de développement, objet du chapitre 9 « Génie logiciel » ;
3. réorganisation (*refactoring*) des paquetages, des configurations (activité récurrente) : les paquetages et classes liés à HyperSmooth sont par exemple supprimés des sources du projet ;
4. portage des logiciels vers une application Web décrite chapitre 10 ;
5. développement de nouvelles fonctionnalités : objets des chapitres 11 pour HyperAtlas et 12 pour HyperAdmin.

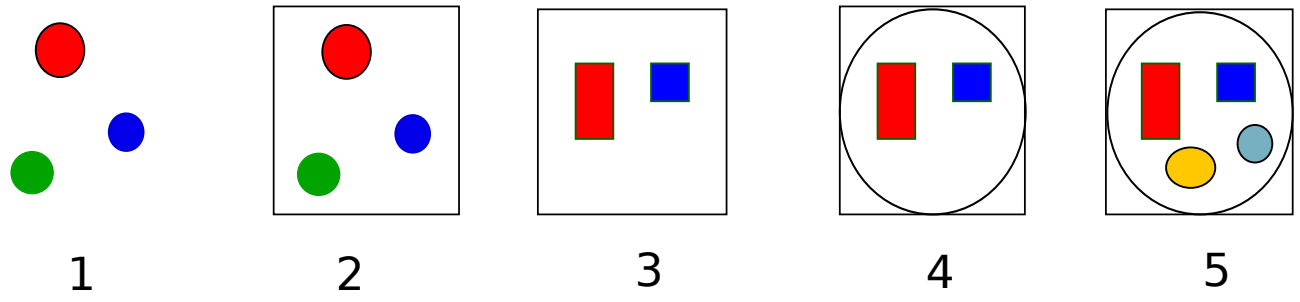


FIGURE 7.1 – Vue schématique des principales étapes.

D'une façon générale, la gestion du projet s'opère suivant une démarche itérative selon la roue de Deming, illustrée sur la figure 7.2, en suivant sur chaque itération les quatre étapes suivantes (en anglais, *Plan - Do - Check - Act*) :

planification : on planifie les tâches à accomplir sur cette itération ;

action : on tente d'accomplir les tâches définies précédemment dans le délai imparti ;

vérification : on vérifie si le travail réalisé correspond aux besoins exprimés dans les délais et les coûts précisés à la première étape ;

ajustement : on recherche les améliorations à apporter au sujet.

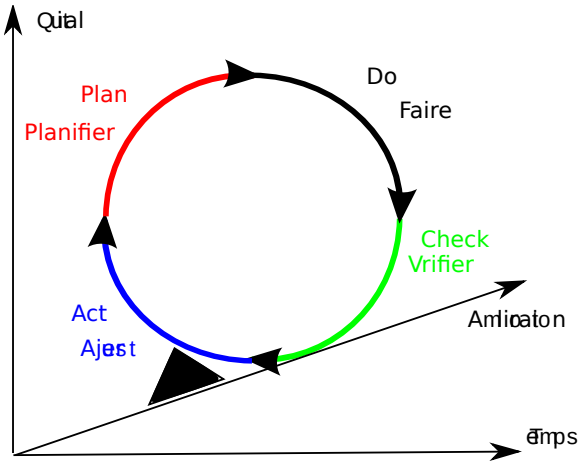


FIGURE 7.2 – Les quatre étapes d'une démarche itérative selon la roue de Deming.

Un cycle de la roue s'accomplit sur une période d'environ un mois, à l'issue duquel une réunion STEAMER est organisée, pour à la fois faire le point sur les tâches réalisées (ajustement), et établir les priorités pour le cyle suivant (planification). Deux réunions rassemblant tous les membres du groupe de recherche HyperCarte (RIATE, Géographie-Cités et STEAMER) auront lieu pendant ce stage, en mars et novembre 2010.

Parallèlement à cette approche générale, la réalisation d'une tâche s'opère selon une imbrication de sous-itérations, démarche s'inspirant en quelque sorte des cycles de vie et de quelques bonnes pratiques définies par l'XP (*Extreme Programming*) [Wel09], du RAD (*Rapid Application Development*, développement rapide d'applications) [1] et des méthodes Agile. La réalisation d'une tâche peut ainsi se décomposer selon les cinq étapes suivantes :

1. préparation, étude du besoin ;
2. recherche éventuelle de documentation sur le sujet, apprentissage, bibliographie ;
3. conception et spécifications ;
4. développement (tests compris) et écriture de la documentation technique (guide du développeur, Javadoc) simultanés ;
5. finalisation, déploiement d'une version intégrant la nouvelle fonctionnalité sur un serveur de tests.

La mise à jour régulière des prototypes intégrant petit à petit les nouvelles fonctionnalités permet théoriquement des retours rapides par les utilisateurs. Idéalement, dans l'esprit XP/Agile, le client final est ici impliqué. Dans le cadre de ce projet, ce sont tantôt les autres membres de STEAMER, ou du groupe de recherche HyperCarte, qui apportent leurs commentaires. M. Gensel valide le résultat, décide des priorités. Des ajustements en conséquence sont ainsi mis en œuvre au plus tôt pour relancer une sous-itération de conception, spécification, réalisation et intégration.

1. Site historique de la méthode RAD. [en ligne] http://www.rad.fr (consulté le 10 décembre 2010).

Le cycle itératif décrit précédemment s'inscrit dans la démarche interne à STEAMER, en calant les cycles et les productions sur les trois jalons du projet, contractés dans le cahier des charges et décrits dans le tableau 2.1 page 13 : 31 août 2010, 31 décembre 2010 et 28 février 2011.

Conception modulaire

Tant que possible, les nouvelles fonctionnalités du logiciel seront abordées en tenant compte des principes directeurs et des critères de qualité de la conception modulaire. Cette section propose une revue des principes majeurs d'une telle approche.

L'autonomie d'un module est une de ses particularités essentielles : un module peut ainsi être compilé, testé et archivé séparément. L'approche modulaire facilite en outre le prototypage.

La qualité d'une conception modulaire repose sur les critères suivants [Aou03] :

Décomposabilité : la décomposabilité consiste en l'éclatement d'un problème logiciel en un nombre de sous problèmes moins complexes, connectés entre eux par des mécanismes simples, et suffisamment indépendants pour permettre de travailler séparément sur chacun d'eux. Un des avantages de la décomposabilité est la possibilité éventuelle de distribuer le travail sur chacun des sous systèmes à des personnes ou groupes différents.

Composabilité (ou compatibilité) : une approche satisfait à l'exigence de composabilité modulaire si elle facilite la production d'éléments logiciels. Ils peuvent ensuite être combinés avec d'autres pour produire de nouveaux systèmes, éventuellement dans des environnements différents.

Compréhensibilité modulaire : chaque acteur humain peut comprendre chaque module sans connaître les autres. A cette fin, on fournit au minimum les signatures des méthodes. Les contraintes fournissent une information complémentaire.

Continuité modulaire : un changement mineur dans la spécification du problème ne provoque un changement que dans un seul ou peu de modules. L'indépendance modulaire est obtenue par la décomposabilité. La continuité oblige la décomposabilité.

Protection : une condition anormale à l'exécution dans un module sera confinée à ce module, ou ne se propagera qu'à ses modules voisins.

Généricité modulaire Aptitude d'un module à traiter des paramètes de types arbitraires.

Finalement, tout en tenant compte de ces précédents critères, la démarche de conception modulaire suivra les cinq principes directeurs suivants :

Unité linguistique : renforce l'exigence fondamentale de compréhensibilité modulaire.

Peu d'interface : un module ne doit communiquer qu'avec le plus petit nombre possible de modules. L'architecture fait en sorte que chaque module ne communique qu'avec ses voisins immédiats.

Couplage faible : deux modules communiquant doivent échanger aussi peu d'information que possible.

Interface explicite : chaque communication entre deux modules doit être clairement exprimée.

Abstraction : il s'agit du masquage d'information. Les caractéristiques essentielles d'un concept sont clairement explicitées, les frontières conceptuelles sont rigoureusement définies. Le masquage d'information est le principe fondamental de l'approche orientée objet.

Chapitre 8

Analyse de l'existant

En complément à l'approche générale des logiciels décrits section 1.2.1 page 8, ce chapitre présente plus précisément les principaux outils mis à disposition de l'utilisateur dans les versions originales de HyperAtlas et HyperAdmin (les sources récupérées correspondent à la version 1.2.9, à l'issue du stage de Raphaël Thomas fin 2008).

L'analyse du code existant est une activité quotidienne. Ce chapitre ne résume que quelques points-clés et très généraux quant aux technologies, modèles de données, configuration et éléments de conception tels qu'ils ont été conçus, implémentés et décrits dans les mémoires CNAM de mes prédécesseurs. Ces sections visent essentiellement à introduire les adaptations de l'existant qui seront initiées et décrites dans les chapitres suivants.

8.1 HyperAtlas

L'application est archivée dans un fichier exécutable Java nommé `HyperAtlas.jar`. Cette archive incorpore un jeu de données par défaut (fichier `.hyp`) qui est désérialisé et chargé en mémoire au démarrage de l'application.

L'interface graphique utilisateur (GUI, pour *Graphical User Interface*) de l'application de bureau HyperAtlas est principalement composée de deux panneaux interactifs : un panneau de paramètres globaux et un panneau central composé de huit onglets. Chacun de ces onglets d'analyse (un objet `Frameset`) est composé d'un panneau cartographique et d'un panneau latéral. Le panneau latéral est lui-même composé de trois onglets pour afficher respectivement la légende de la carte, ses options et une explication. La figure 8.1 illustre le premier onglet « Aire d'étude », permettant à l'utilisateur de visualiser l'aire d'étude et le maillage élémentaire choisis dans le panneau de paramètres.

L'utilisateur peut choisir un nouveau jeu de données depuis le menu « Fichier - Ouvrir ». Au chargement d'un jeu de données, le panneau de paramètres de l'interface graphique est mis à jour pour proposer à l'utilisateur un ensemble de choix, sous la forme de listes de sélection regroupées sur trois colonnes :

- une colonne pour le choix d'une aire d'étude géographique et d'un niveau de maillage ;
- une colonne pour le choix de deux indicateurs statistiques, ou directement d'un ratio prédéfini ;
- une colonne pour le choix des contextes général, territorial et spatial pris en références pour le calcul des écarts associés.

Les huit onglets cartographiques du panneau central sont regénérés en fonction des modifica-

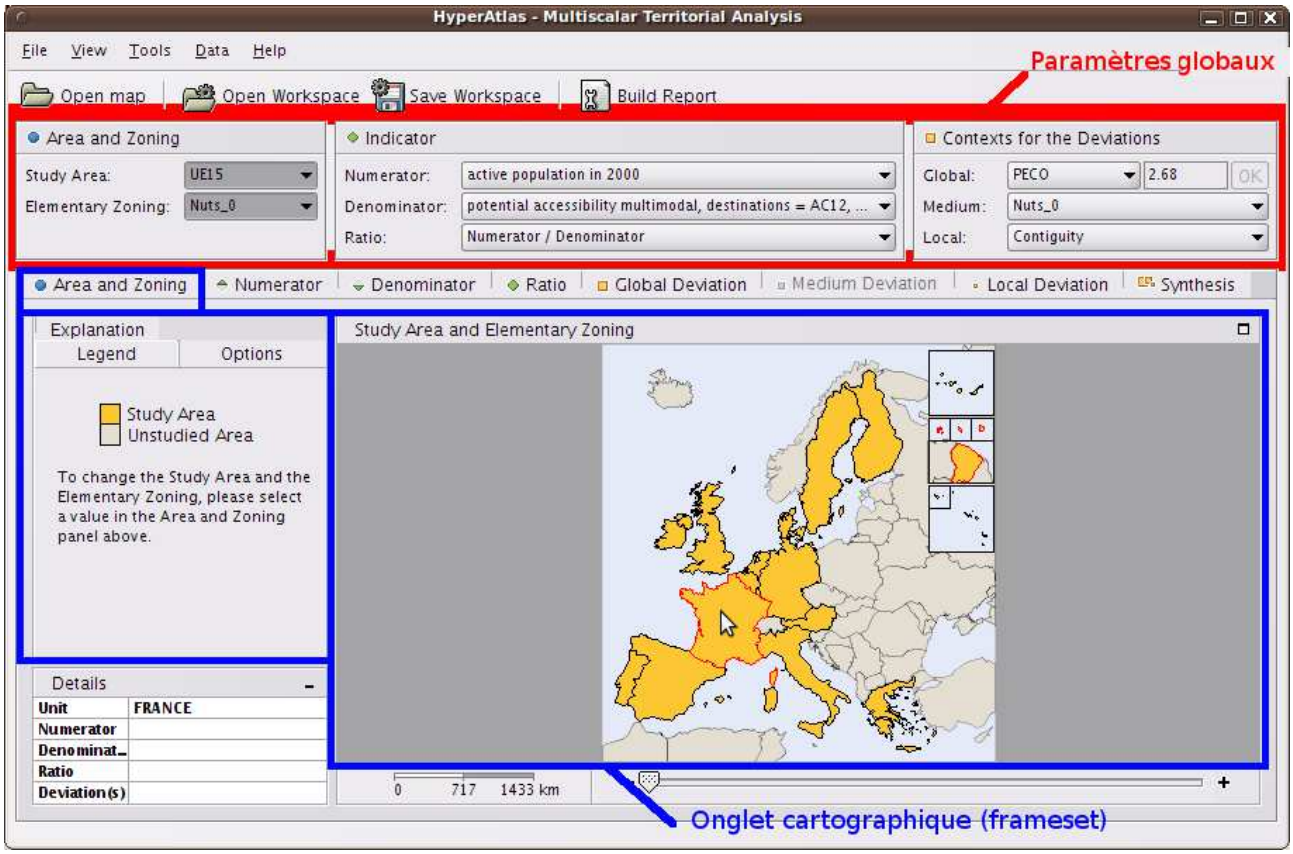


FIGURE 8.1 – Principaux composants de l'interface d'HyperAtlas.

tions apportées sur ces paramètres. La modification de ces paramètres déclenche des évènements dits « globaux ».

Les cartes des huit onglets réagissent également à des évènements « indexés » : par exemple, la modification des options de la légende de la carte (couleurs des disques, seuils des classes de valeurs, etc.).

Les différents outils d'analyse cartographique de l'application sont décrits successivement dans les sous-sections suivantes.

8.1.1 Carte de la zone d'étude

Ce premier onglet cartographique propose à l'utilisateur de visualiser les paramètres choisis dans le premier groupe : l'aire d'étude et le niveau de maillage*.

Rappelons que le maillage peut comporter différents niveaux d'imbrication, une maille peut être composée de plusieurs unités territoriales. Le maillage définit la hiérarchie d'unités territoriales par un arbre dont un exemple est donné sur la figure 8.2.

Comme chacune des autres cartes, la carte de la zone d'étude est accompagnée d'un panneau interactif sur la gauche composé d'un onglet « Légende », d'un onglet « Options », d'un onglet « Description » et d'un tableau « Détails ».

L'onglet des options permet à l'utilisateur de définir les couleurs de l'aire d'étude et de la zone hors étude. Le contenu du tableau de détails en bas à gauche réagit aux positions de la souris sur la carte, en fonction de l'unité territoriale survolée. En fonction de l'onglet cartographique courant, y sont mentionnés le nom de l'unité, les valeurs des indicateurs, du ratio, des écarts.

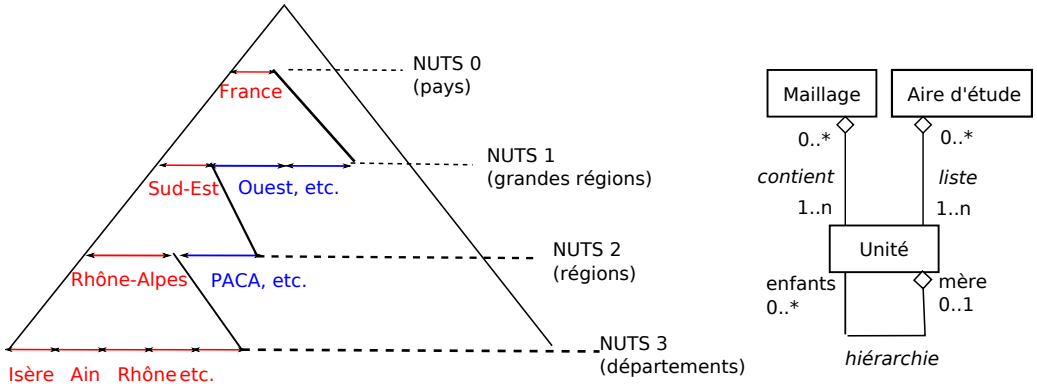


FIGURE 8.2 – Représentations de l'arborescence hiérarchique entre les unités et du modèle objet entre les entités unité territoriale, maillage et zone d'étude (images d'après [Plu07]).

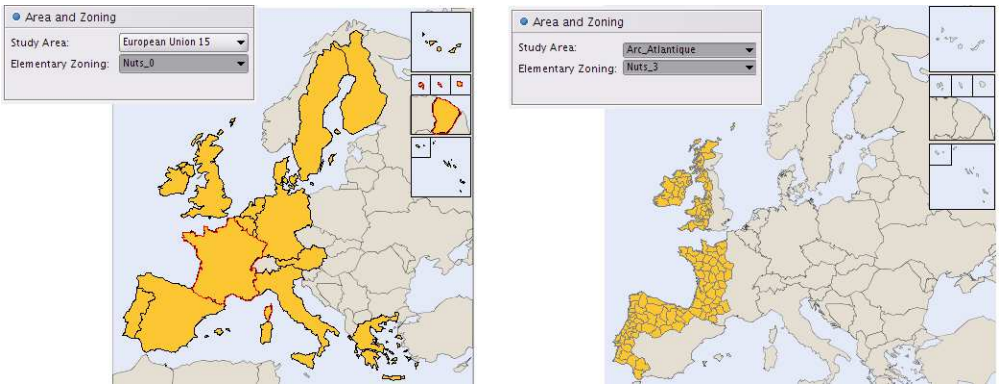


FIGURE 8.3 – Deux exemples de choix de zone d'étude et de niveau de maillage différents pour le même jeu de données.

Le premier onglet cartographique permet donc à l'utilisateur de visualiser sur une carte ses choix de la zone d'étude et du maillage choisis dans le panneau de paramètres. Deux exemples sont fournis sur la figure 8.3.

8.1.2 Cartes des indicateurs

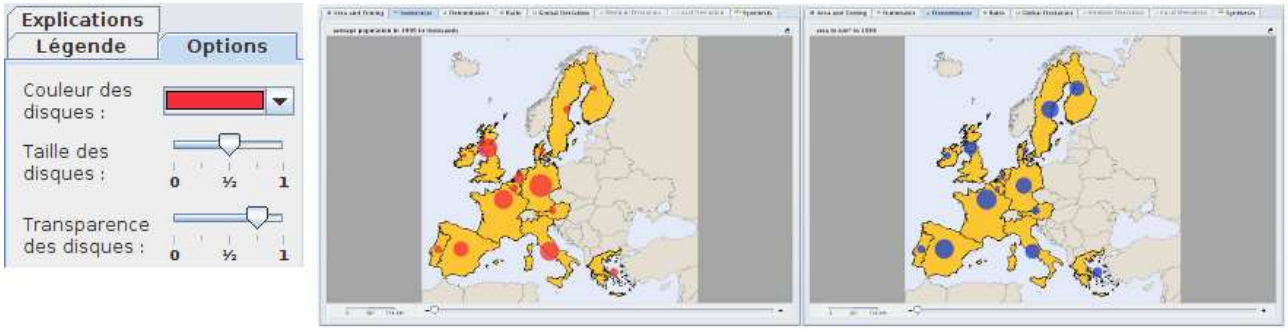


FIGURE 8.4 – Options de la légende des cartes à disques et cartes associées aux indicateurs numérateur et dénominteur (maillage NUTS 0).

Les indicateurs statistiques choisis en paramètres dans les listes de sélection pour le numérateur et le dénominateur sont représentés par des cartes à disques. Les options de la légende permettent de calibrer la taille des disques, leur couleur et le niveau de transparence.

La copie d'écran 8.4 juxtapose l'onglet des options de la légende pour le type de cartes à disques, puis les deux cartes associées aux valeurs des indicateurs choisis.

8.1.3 Carte du ratio

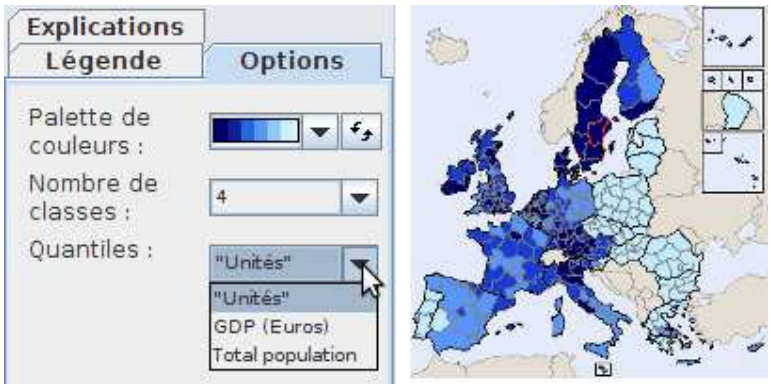


FIGURE 8.5 – La carte de ratio et les options de sa légende.

La carte de ratio est une carte choroplèthe*, sur laquelle la couleur de fond de chaque unité correspond à un intervalle de valeurs pour le ratio, calculé en divisant la valeur de l'indicateur statistique choisi pour le numérateur, par la valeur de l'indicateur choisi pour le dénominateur. La légende interactive de la carte de ratio permet à l'utilisateur de choisir la palette de couleurs sur laquelle il/elle définit le nombre de classes (figure 8.5).

8.1.4 Cartes d'écarts

FIGURE 8.6 – Onglets cartographiques des écarts aux contextes de références, général, territorial et spatial.

La figure 8.6 montre les trois onglets pour observer les écarts des unités territoriales de l'aire d'étude choisie par rapport, respectivement, aux trois contextes choisis dans les paramètres :

- le contexte général : l'utilisateur choisit une des aires d'étude disponibles (EU15, EU27), ou une valeur de son choix ;
- le contexte territorial (ou hiérarchique) propose à l'utilisateur de choisir un niveau de maillage supérieur à celui du maillage élémentaire courant. La carte d'écart territorial est désactivée

quand aucun niveau de maillage supérieur ne peut être choisi pour le maillage élémentaire courant ;
- le contexte spatial (ou local) propose par défaut à l'utilisateur une référence considérant les unités entretenant une relation d'adjacence avec l'unité territoriale courante. De plus amples options quant au choix de ce contexte spatial peuvent être également disponibles dans le jeu de données : des matrices de distance-temps (exemple : à moins de deux heures d'avion) peuvent ainsi être définies pour lier les unités les unes aux autres selon différents opérateurs de comparaison.

Le calcul de l'écart relatif E d'une unité territoriale i pour un contexte $Contexte$ peut être exprimé par la formule 8.1, en considérant les variables suivantes :
- la variable Num_x est la valeur de l'indicateur choisi au numérateur pour l'unité x ;
- la variable Den_x est la valeur de l'indicateur choisi au dénominateur pour l'unité x ;
- $C_{i,j}$ est un booléen prenant la valeur 0 ou 1, son calcul dépend du contexte de référence :
  - contexte général : $C_{i,j} = 1$ si l'unité territoriale j appartient au contexte général choisi (exemple, la France dans l'Europe des 15), 0 sinon ;
  - contexte territorial hiérarchique : $C_{i,j} = 1$ si les unités territoriales i et j vérifient la relation choisie en paramètre pour le contexte territorial, une relation hiérarchique régions-pays par exemple ;
  - contexte spatial de voisinage : $C_{i,j} = 1$ si l'unité territoriale j vérifie la relation choisie pour le contexte spatial avec l'unité i (exemple : à moins de deux heures d'avion), 0 sinon.

$$E_i^{Contexte} = 100 \cdot \frac{\frac{Num_i}{Den_i}}{\frac{\sum_j C_{i,j} \cdot Num_j}{\sum_j C_{i,j} \cdot Den_j}} \tag{8.1}$$

Dans HyperAtlas, l'écart d'une unité territoriale à un contexte de référence correspond donc au rapport entre le ratio de deux indicateurs pour cette unité courante, sur le ratio de ces deux mêmes indicateurs calculés sur l'ensemble des unités du contexte de référence. Le contexte général de référence permet de choisir une aire d'étude (exemple : l'Europe des 15) ou une valeur arbitraire. L'écart relatif territorial est le rapport entre le ratio de l'unité courante sur le ratio pour les unités du niveau supérieur choisi. L'écart relatif spatial est le rapport entre le ratio de l'unité courante sur le ratio pour les unités contiguës.

Les onglets cartographiques associés aux trois écarts relatifs sont illustrés sur la figure 8.6. Les options des légendes pour ces trois cartes d'écarts sont représentées sur la figure 8.7 : l'utilisateur a la possibilité de choisir une palette proposant des caissons sur deux couleurs (une couleur pour les valeurs au dessus de celle du contexte de référence, une couleur pour les valeurs inférieures). Il/elle peut également éditer les seuils des classes de valeurs, et la répartition de ces classes selon une progression géométrique ou arithmétique.

Les différentes cartes d'écarts de l'analyse territoriale multiscalaire mettent en évidence que les unités ne sont pas isolées les unes des autres, il y a interaction spatiale. Les positions relatives des différentes unités entrent en jeu pour l'analyse : plus deux zones sont proches, plus la chance d'interaction entre elles est importante. La notion de contiguïté* du contexte de référence spatial permet d'observer ces interactions. Dans HyperAtlas, la relation d'adjacence est la relation par défaut pour le contexte spatial, mais d'autres relations de contiguïté peuvent être définies pour un jeu de données : l'utilisateur d'HyperAdmin peut ainsi fournir des matrices dans lesquelles les valeurs entre deux unités sont basées sur une distance géométrique ou une distance-temps-véhicule

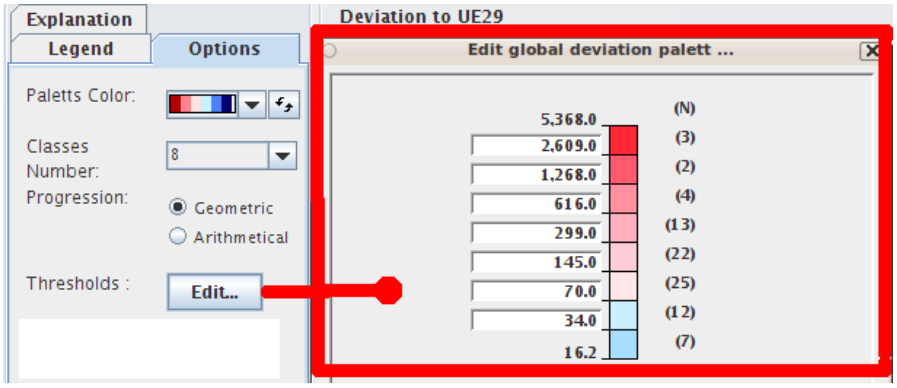


FIGURE 8.7 – Options de la légende des cartes d'écarts.

proportionnelle aux temps de parcours entre deux points.

On formalise ainsi les effets de la proximité pour l'écart spatial en recourant à la notion de distance : distance géométrique, distance-temps-véhicule (moins de deux heures de route en camion par exemple), proportionnelle aux temps de parcours entre deux points.

Aussi la contiguïté est-elle représentée par une matrice carrée booléenne. Il y a autant de lignes et de colonnes que d'unités territoriales. L'élément $C_{i,j}$ de la matrice prend pour valeur 1 si l'unité i est contiguë avec l'unité j, 0 sinon.

8.1.5 Carte de synthèse ternaire

La carte de synthèse ternaire permet de combiner les résultats des trois écarts relatifs afin de mettre en évidence leurs relations. L'onglet de synthèse dresse une typologie cartographique de ces relations en mode ternaire, les unités sont classées selon huit classes en fonction des trois positions de leurs écarts.

Afin de réduire les combinaisons possibles, l'utilisateur peut définir dans un onglet « options de la légende » le point de vue sur lequel il/elle souhaite concentrer son analyse : un paramètre « Critère » permet de spécifier un comparateur entre les écarts et une valeur seuil (100 % par défaut, c'est-à-dire le ratio du contexte de référence). En fonction des indicateurs étudiés, la synthèse peut ainsi viser à identifier les territoires avantagés ou au contraire défavorisés.

En cliquant droit sur une unité de la carte de synthèse, un menu contextuel permet à l'utilisateur d'ouvrir une fenêtre montrant précisément les valeurs des écarts pour cette unité dans un histogramme. La figure 8.8 illustre les possibilités de cette carte de synthèse ternaire.

8.2 HyperAdmin

La GUI de HyperAdmin est un clone de HyperAtlas. HyperAdmin a en effet été pensé comme une version de HyperAtlas proposant à l'utilisateur un menu supplémentaire lui permettant d'intégrer des données afin de générer un `.hyp`. Hormis le logo du démarrage, le lancement de l'application de bureau HyperAdmin renvoie une fenêtre identique à HyperAdmin, composée du panneau de paramétrage et des onglets cartographiques. La différence entre les deux applications tient dans le menu « Assistant de création de projet » (*Wizard*, plus précisément, HyperAdmin n'est pas internationalisé et son interface n'est disponible qu'en anglais). Objet du stage de Raphaël Thomas [Tho08], l'activation de ce menu ouvre une fenêtre modale dont le contenu évolue au fur et à me-

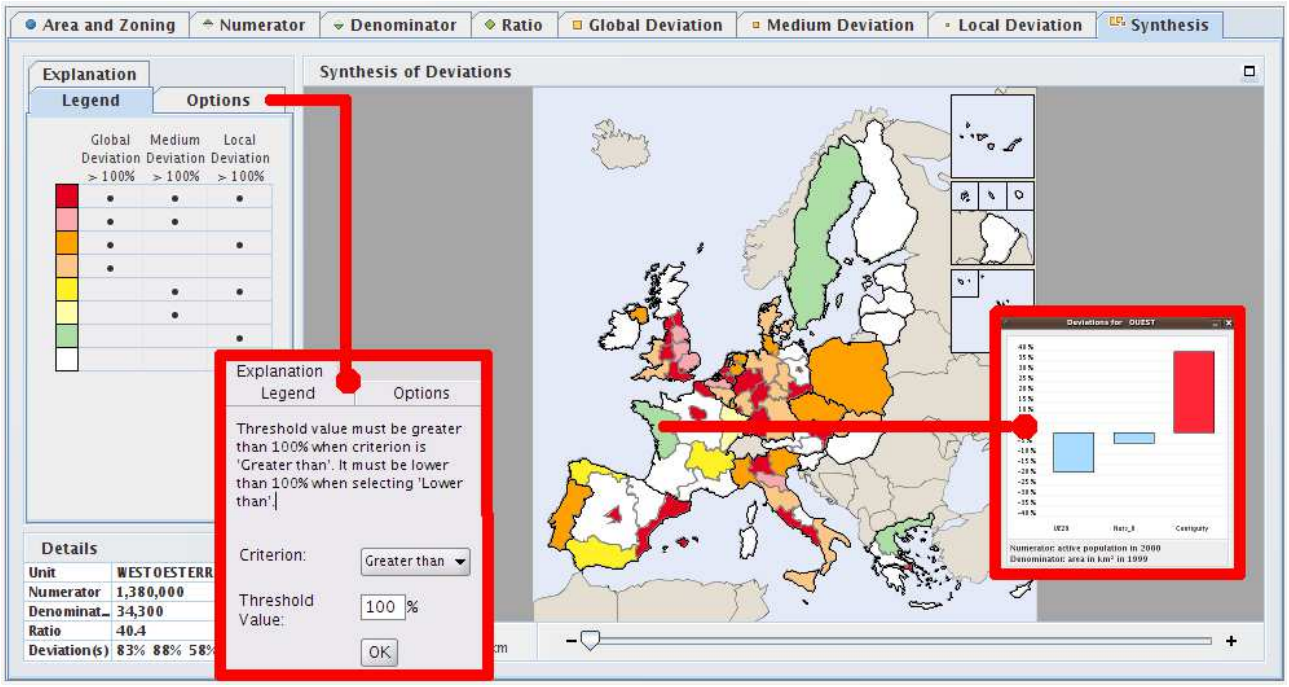


FIGURE 8.8 – Onglet cartographique de la carte de synthèse ternaire, options de la légende et histogramme des écarts d'une unité.

sure des différentes étapes, guidant l'utilisateur jusqu'à l'enregistrement sur son disque du jeu de données qu'il souhaite créer.

La création d'un `.hyp` passe typiquement par les étapes suivantes :

1. soumission du choix d'une géométrie sous la forme d'une paire de fichiers MIF-MID (format MapInfo) ;
2. soumission d'un répertoire dans lequel doivent être présents :
   - un fichier tableur (ou son équivalent en fichiers au format texte, un par onglet) pour décrire la structure du jeu de données : les unités, leurs relations hiérarchiques, etc. ;
   - un fichier tableur (ou son équivalent en fichiers texte) pour décrire les données statistiques au plus plus bas niveau des unités territoriales définies dans le fichier structure ;
3. invitation à calculer ou non les contiguïtés simples (adjacence) entre les unités, pour la disponibilité des écarts spatiaux, et à fournir d'éventuelles matrices supplémentaires. Selon le nombre d'unités, l'utilisateur peut être invité à fournir les paramètres de connexion à une base de données qui s'avère nécessaire et obligatoire lorsque ce nombre dépasse le millier ;
4. invitation à sauvegarder le jeu de données en base ou à le sauver sous la forme d'un fichier sérialisé `.hyp`.

8.3 Technologies

Les applications HyperAtlas et HyperAdmin sont développées en Java. L'avantage premier de cette technologie est la possibilitée d'exécuter ces logiciels sur toute plate-forme munie d'un environnement d'exécution Java (JRE pour *Java Runtime Environment*). Le plan de tests qui sera mis en œuvre lors de ce projet tendra autant que possible à vérifier l'ensemble des fonctionnalités sur les systèmes d'exploitation Linux, Macintosh et Windows.

Le portage de l'application de bureau HyperAtlas en version *applet* est également facilitée par Java. Ce langage permet relativement facilement l'ambivalence d'exécution du même code en version application de bureau ou depuis une *applet* embarquée dans une page HTML.

La version *applet* de HyperAtlas nécessite cependant une adaptation conséquente pour passer outre les restrictions de sécurité de la JRE : le logiciel nécessite en effet des entrées-sorties en écriture sur le disque de l'utilisateur, afin d'enregistrer les jeux de données modifiés, le rapport généré ou les cartes exportées au format image. L'accès en écriture au disque client depuis une *applet* embarquée dans une page Web est, par défaut, refusée par la JRE cliente. Cette possibilité d'écriture est cependant rendue possible en signant l'*applet* à l'aide d'un certificat électronique. Au moment du téléchargement de l'*applet*, le navigateur de l'utilisateur l'avertit du contenu signé et potentiellement dangereux du programme qu'il/elle s'apprête à exécuter. L'utilisateur est ainsi systématiquement invité à valider, vérifier ou réfuter l'exécution de l'*applet* signée qu'on lui propose.

L'application HyperAdmin requiert, quant à elle, la disponibilité d'une base de données PostgreSQL [1] munie de son extension PostGIS [2] pour manipuler des types géographiques.

Cette base de données n'est cependant pas nécessaire si les données fournies en entrée concernent moins de 1000 unités territoriales élémentaires, pour lesquelles les contours géographiques et contiguïtés sont à calculer. Dans ce cas de petits jeux de données, d'après les tests de performances exécutés par Christine Plumejeaud [Plu07], les performances du logiciel pour le calcul des topologies sont même meilleures en travaillant directement en mémoire, à l'aide de la librairie JTS [3].

Néanmoins, non seulement l'application devient plus performante à partir de 1000 unités en bénéficiant des calculs topologiques de PostGIS, mais la sollicitation de la mémoire est telle qu'il devient impossible de manipuler plusieurs milliers d'unités dans la JVM sans bloquer l'application (*out of memory error...*).

En toute généralité, la disponibilité d'une base de données est donc nécessaire pour HyperAdmin. Quand la quantité de données à traiter le nécessite, le contrôleur de l'application engage donc la couche vue, via l'assistant de projet, et invite l'utilisateur à fournir les paramètres de connexion JDBC (*Java Data Base Connectivity*) à une base de données `hyperadmin` : adresse IP du serveur de bases de données, nom de la base, nom d'utilisateur et mot de passe.

Notons qu'une étape importante en termes de temps a consisté en début de projet à faire fonctionner HyperAdmin. Avant de se familiariser avec les attentes de l'application elle-même, l'installation de la base de données attendue par le logiciel n'a pas été aisée. Nous partions d'une sauvegarde de la base incorporant structure, données, fonctions plPgsql spécifiques à HyperAdmin, et fonctions géométriques PostGIS. Une analyse a été nécessaire pour disposer de la base attendue sur une version plus récente du SGBD. Cette analyse a mené à l'écriture de scripts SQL isolant la structure et les fonctions spécifiques à la base attendue par HyperAdmin. Cette étude a également été l'occasion de la rédaction d'une documentation conséquente, destinée aux développeurs, et aux clients recevant une distribution. Nous pouvons désormais considérer le script d'installation créé, `hypercarte_structure.sql`, indépendant de la version du SGBD, du moins pour les versions couramment utilisées aujourd'hui, puisqu'il a été validé sur les environnements suivants :

- PostgreSQL version 8.4 et PostGIS version 1.4.0 sous Linux Ubuntu version 9.10 (poste de développement) ;

1. PostgreSQL *The world's most advanced open source database* [en ligne] http://www.postgresql.org/ (consulté le 17 juillet 2010).
2. *PostGIS Home* [en ligne] http://postgis.refractions.net/ (consulté le 17 juillet 2010).
3. JTS *Toplogy Suite* [en ligne] http://www.vividsolutions.com/jts/jtshome.htm (consulté le 17 juillet 2010).

- PostgreSQL version 8.4.2 et PostGIS version 1.3.6 sous Linux CentOS (poste de déploiement) ;
- PostgreSQL version 8.3 et PostGIS version 1.3.0 sous Mac OS X.5 Leopard ;
- PostgreSQL version 9.0 et PostGIS version 1.5 sous Windows XP 32 bits. L'extension PostGIS n'est actuellement pas disponible pour la version 9.0 de PostgreSQL installée sur une machine 64 bits.

Pour information, le schéma physique de la base de données est fourni dans l'annexe B.

8.4 Configurations

Suivant le client cible, le binaire exécutable livré sous la forme d'une archive Java `.jar` incorpore un jeu de données par défaut spécifique, des icônes personnalisées, et une mise en page particulière. Bien qu'un fichier de propriétés permet de regrouper quelques variables (nom du client pour le titre des fenêtres, icônes à lancer au démarrage, jeu de données par défaut), ces valeurs sont ignorées et la configuration de l'application est codée en dur dans les fichiers sources Java.

En fonction du nom du jeu de données `.hyp` chargé, le code Java effectue également des traitements particuliers, pour répondre à des services ou bugs connus, liés par exemple à des géométries particulières. Le dessin des cartes va par exemple considérer que des unités ayant un identifiant entre 0 et 9 correspond à un cartouche carré à représenter avec un fond bleu : cette solution permet de traiter le cas des cartes européennes où les unités territoriales d'outre-mer de la France (Antilles, Guyane), de l'Espagne (Iles Canaries), et du Portugal (Madère), sont déportées dans un cartouche au dessus de la Russie. Le dessin des cartes et le style des unités dépend donc fortement de conventions de nommage et de contraintes fortes, non documentées, lors de la phase de préparation des données, en amont d'HyperAdmin.

Ces quelques exemples nuisent à la généricité de l'application et à son utilisation par tout utilisateur non averti.

8.5 Gestion des évènements et singletons

Les actions de l'utilisateur sur l'interface sont associées à deux types d'évènements gérés par l'application :

- des évènements dits globaux, qui affectent l'ensemble des composants graphiques : le choix d'un nouveau paramètre, comme par exemple le maillage élémentaire, nécessite la mise à jour de toutes les cartes ;
- des évènements dits indexés, qui affectent les onglets du panneau d'analyse cartographique seulement : par exemple, la couleur des cercles pour les cartes à disques, peut être modifiée par l'utilisateur. Ce changement n'affecte que la carte associée.

Les messages retournés dans des boîtes de dialogue sont gérés en interne, comme un troisième type d'évènements, dits « évènements de message » (`MessageEvent`). Ils sont uniquement déclenchés, quant à eux, depuis le code Java.

Le traitement de tous ces évènements est contrôlé par la classe singleton `Dispatcher`, conçue selon le patron de conception « Ecouteur-Observateur ». Dans HyperAtlas, selon Olivier Cuénot [Cue05], le système mis en place répond plus exactement au concept de communication inter-composants évènementielle anonyme (CICEA) : la propagation des événements est anonyme, un émetteur ne connait pas la liste des récepteurs de l'évènement, et les récepteurs n'en connaissent

pas la provenance. Le principe est également largement décrit dans le mémoire de Christophe Chabert [Cha07].

Les classes peuvent toutes s'enregistrer auprès du `Dispatcher` pour être averties des évènements globaux ou indexés. Ce dernier maintient deux listes, pour les deux types d'évènements, globaux et indexés.

Chaque évènement est associé à une constante entière unique. Un évènement est déclenché en suscitant le `Dispatcher`. Lorsque ce dernier reçoit un évènement, il avertit toutes les classes qui se seront enregistrées auprès de lui. Ces classes reçoivent donc la notification de l'évènement, et réagissent en conséquence, ou pas, selon la valeur de l'évènement.

Le principal inconvénient de cette gestion des évènements est l'impossibilité de contrôler l'ordre d'exécution par les différents écouteurs d'un évènement donné. Il est également impossible pour le `Dispatcher` de connaître les écouteurs qui vont traiter effectivement l'évènement. En outre, le traitement d'un évènement par un écouteur peut lui-même déclencher un autre évènement, les évènements s'empilent les uns dans les autres comme des poupées russes, l'utilisateur peut regretter que l'application entre parfois dans des boucles infinies. L'exemple type de ce comportement est une boîte de dialogue que l'application ré-ouvre quand l'utilisateur la ferme. Dans un tel cas de figure, l'utilisateur n'a comme seule issue que de fermer brutalement l'application.

Sur le même principe de singleton que le `Dispatcher`, plusieurs classes sont conçues sur ce patron de conception. Par exemple, la classe `Logic` centralise tous les calculs, la classe `Zoom` contrôle le facteur de zoom des cartes, la classe `HCResourceBundle` centralise la gestion de l'internationalisation.

Christophe Chabert introduit également le singleton `Settings`, à l'origine pour centraliser les paramètres graphiques [Cha07], et les partager entre différentes composants. L'alternative consiste à propager un paramètre dans les méthodes. Cette classe `Settings` s'est petit à petit enrichie de multiples attributs, méthodes et accesseurs publiques. Son instance unique en mémoire centralise presque tout : les paramètres d'analyse choisis par l'utilisateur, leur code, leur nom, le jeu de données, etc. Pour illustration de son omniprésence dans le code, une commande Unix[4] montre que `Settings` est importée depuis 133 classes de l'application. Son utilisation excessive nuit à la modularité du projet.

Plus généralement, la présence de multiples singletons et classes statiques rend plus difficile la gestion des opérations lors du chargement d'un nouveau jeu de données : il s'agit en effet de tous les réinitialiser. Cette réinitialisation incomplète semble la raison principale des incohérences constatées lors du chargement successif de jeux de données, spécialement lorsqu'ils sont basés sur des géométries différentes. Pour l'utilisateur, l'application nécessite parfois d'être redémarrée pour passer d'un jeu de données existant à un autre, de Rhône-Alpes ou du Cameroun à l'Europe, par exemple.

4. La commande Unix `grep -r "import hypercarte.hyperatlas.config.Settings" src | wc -l` retourne le nombre d'occurences de l'import de la classe `Settings` depuis les sources Java.

Chapitre 9

Génie logiciel

Ce chapitre présente la méthodologie de développement suivie pour la résolution du problème.

Après une introduction décrivant les objectifs souhaités, les méthodes et outils mis en œuvre dans le cadre de ce projet sont passés en revue.

9.1 Objectif : architecture de l'environnement de développement

L'objectif est de mettre en place une architecture multiplate-forme et de fournir une description suffisament détaillée quant à l'environnement de développement et aux outils utilisés pour permettre à tout développeur de collaborer sur le projet.

L'idée principale est de disposer d'une architecture standard s'affranchissant de la « machine magique » des développeurs. Un tel environnement permet ainsi de s'assurer que toute version du projet et de ses délivrables peut être reconstruite, à partir du moment où les quelques outils prérequis sont installés sur la machine courante.

Ce chapitre présente de façon synthétique les principaux outils de développement et l'architecture de déploiement du projet afin d'en assurer une plus grande qualité.

Les techniques utilisées s'inspirent de quelques « meilleures pratiques » (*best practises*) recommandées par les méthodes d'*Extreme Programming* [Wel09] et Agile [Agi06].

9.2 Gestion des sources et des versions

L'acronyme SCM pour *Source Control Management* est le terme désignant les outils proposant un appui logiciel pour l'activité complexe consistant à maintenir et maîtriser l'ensemble des versions et révisions d'un logiciel. L'outil concerne essentiellement le code source à partir duquel l'application est reconstruite.

Si un outil SCM favorise le développement collaboratif, il facilite la gestion et le tracé des évolutions avec la possibilité, par exemple, de comparer plusieurs versions pour en extraire les modifications.

De nombreux logiciels SCM existent, parmi les plus populaires : CVS [1], Perforce [2].

1. *Concurrent versions system* http://www.nongnu.org/cvs/ (consulté le 2 juillet 2010).
2. *Perforce Software* http://www.perforce.com/ (consulté le 2 juillet 2010).

La plate-forme LIGforge[3] met à disposition des membres du LIG un ensemble d'outils de gestion de projets dont le célèbre SCM **Subversion**[4], légitime héritier de CVS.

Profitant de cette forge de logiciels, les sources existantes du projet HyperCarte (classes Java, librairies et fichiers de configuration) sont importées le premier jour du stage en tant que projet Subversion. L'étiquettage des versions appose la révision 0 à cet ensemble de ressources dans l'état original (en considérant ce stage) du projet.

Tout au long du projet, nous profiterons également de la boîte à outils essentiels de la forge dont notamment la liste des tâches et la traçabilité des bugs. Le projet LIGforge HyperCarte[5] est déclaré privé, la console d'administration du projet restreint actuellement l'accès à quelques membres de l'équipe STEAMER. L'accès en ligne à travers un simple navigateur à cet ensemble d'outils constitue une réelle facilité de suivi et de visibilité du développement.

9.3 Construction automatique

Les développeurs Java disposent aujourd'hui d'IDE (*Integrated Development Environment*, environnement de développement intégré) puissants pour éditer, compiler, exécuter et déboguer des applications.

Si les IDE facilitent énormément le développement, ils dissimulent cependant un piège dont il est capital de tenir compte : le caractère « intégré » de ces logiciels sous-entend en effet une dépendance forte au système sur lequel travaille le développeur. Produire des applications Java pour bénéficier de l'atout « développer une fois, exécuter sur toute plate-forme » nécessite donc une vigilance, d'une part dans le code, et, d'autre part, dans la prise en compte de la variété d'environnements cibles sur lesquels seront exécutés les délivrables.

A l'heure actuelle, deux IDE libres dominent parmi les développeurs Java : Eclipse[6] (ma préférence) et NetBeans[7]. Utiliser l'un ou l'autre et s'en remettre entièrement à leurs assistants de projet pour gérer les librairies externes mènent au piège de la « machine magique » du développeur.

Afin de s'affranchir de l'IDE utilisé et de maîtriser entièrement son projet, il est donc grandement préférable de confier la fabrication des délivrables à un outil dédié pour le processus de construction (*build process*). Maven[8] et Ant[9] sont deux logiciels écrits en Java et donc multiplateformes permettant respectivement la gestion des projets et leur construction automatique. A partir d'un script XML écrit par le développeur, Ant est un langage procédural qui permet de définir des cibles (*target* dans la terminologie Ant) constituées de tâches à exécuter. Les différentes tâches à accomplir pour fournir un délivrable sont ainsi organisées hiérarchiquement dans les différentes cibles.

Typiquement, la génération d'une distribution d'un logiciel avec Ant passera par les cibles suivantes, chacune devant être achevée avec succès pour accomplir la suivante : nettoyage de la

3. LIGForge http://ligforge.imag.fr/ (consulté le 2 juillet 2010).
4. Apache Subversion http://subversion.apache.org/ (consulté le 2 juillet 2010).
5. LIGforge hypercarte [en ligne, accès restreint] https://ligforge.imag.fr/projects/hypercarte/ (consulté le 3 juillet 2010).
6. *Eclipse.org home* [en ligne] http://www.eclipse.org/ (consulté le 3 juillet 2010).
7. *Welcome to NetBeans* http://netbeans.org/ (consulté le 3 juillet 2010).
8. *Welcome to Apache Maven* [en ligne] http://maven.apache.org/ (consulté le 10 octobre 2010).
9. Apache Ant http://ant.apache.org (consulté le 3 juillet 2010).

construction précédente, récupération des sources, compilation, génération d'un `.jar`, exécution des tests unitaires, génération des documentations, archivage.

Non seulement Ant offre, par défaut, un ensemble très riche de tâches courantes fournies sous la forme de balises XML dont on renseigne les attributs, mais la création et l'utilisation de tâches spécifiques est également facilement intégrable.

Largement maintenu et utilisé, Ant est une brique logicielle indispensable d'un environnement de développement en Java. Il est d'ailleurs complètement intégré dans les IDE et autres outils de gestion de projet comme Maven.

Une fois le projet versionné dans Subversion (voir section section 9.2 page 59), il s'agissait de pouvoir disposer des applications exécutables à partir de l'état original des sources. En corrigeant et en améliorant un script Ant existant, j'en suis donc venu à configurer et mettre au point le processus complet de construction automatique du projet.

Le script de construction évolue lui aussi avec les sources du logiciel, il est donc archivé sous Subversion. Il constitue une brique du développement quasiment indispensable pour automatiser et partager le processus de construction automatique des délivrables, indépendamment de l'environnement du développeur. Il est également le processus clé pour une plate-forme d'intégration continue dont le principe est présenté section 9.6 page 65.

Une étape supplémentaire consisterait à faire bénéficier des avantages de Maven pour la gestion du projet, notamment en terme d'informations générées automatiquement par cet outil. Contrairement à Ant, Maven est déclaratif et respecte des conventions. Il facilite le processus de construction à partir d'une définition d'un modèle objet de projet (POM, pour *Project Object Model*). La mise à jour des sources d'HyperAtlas et HyperAdmin pour le respect des conventions Maven requiert une revue complexe du projet et une réelle expertise Maven. Le coût en termes de délai pour une mise en place opérationnelle de cet outil sur le projet a été estimé trop important pour répondre au cahier des charges dans les temps. Le projet se contentera pour l'instant de Ant.

9.4 Tests unitaires

Le test unitaire est un procédé permettant de s'assurer du fonctionnement correct d'une partie déterminée d'un logiciel, d'un module, mais plus généralement d'une méthode. Le test unitaire permet de confronter la réalisation d'une méthode à sa spécification. Le développeur est donc encouragé à écrire des tests pour s'assurer qu'un module, indépendamment du reste du programme, répond aux spécifications fonctionnelles. Etape clé lors de la construction du logiciel, l'exécution automatique des tests unitaires permet, en outre, de vérifier la non-régression de l'application à la suite des modifications des sources. Selon les principes de l'*Extreme programming*, le test unitaire est même écrit avant la méthode qu'il va vérifier, ou du moins en même temps.

Communément utilisé dans le monde Java, JUnit[10] propose sous la forme d'une librairie, `junit.jar`, un cadriciel permettant d'implémenter des classes de tests. Pour information, notons que Erich Gamma est un des auteurs de JUnit, il a également participé au GoF (« le gang des quatre ») dont les travaux sont notamment reconnus dans le domaine des patrons de conception et de la modélisation orientée objet [GHJV95].

10. *Welcome to JUnit.org* http://www.junit.org/ (consulté le 3 juillet 2010).

Typiquement, le principe du test unitaire consiste à créer une classe `MyClassJunitTest`, pour vérifier les méthodes de sa classe associée, `MyClass`. `MyClassJunitTest` spécialise la classe `TestCase`, issue du paquetage `junit.framework`, fournie dans la librairie `junit.jar`. Dans la classe de tests, on ajoute des méthodes pour valider fonctionnellement les méthodes de `MyClass`. Le code de `MyClass` reste propre, non pollué de traces inutiles et de sorties sur la console qui freinent l'exécution du logiciel.

L'intégration des tests unitaires dans le projet constitue une évolution majeure. Il est dommage de constater qu'aucun moyen n'ait été jusque là mis en œuvre pour valider les fonctionnalités du logiciel. En outre, les tests aident à mieux comprendre les méthodes. Constatant une documentation technique inexistante, une Javadoc très peu utilisable, l'écriture des tests unitaires *a posteriori* est d'autant plus difficile... Tenant compte de cette lacune, une attention particulière sera donc menée lors de ce projet à l'écriture quasi systématique de tests JUnit pour les nouvelles classes. Facilitant à la fois la compréhension des algorithmes, le développement, la pérennité, le maintien et la non-régression, l'intégration des tests unitaires dans l'architecture améliore grandement la qualité de l'application.

9.5 Documentation

Les travaux menés par le groupe de recherche HyperCarte ont été l'occasion d'une publication importante d'articles de recherche[11]. Les différents mémoires CNAM (voir section 1.2 page 6) constituent également une ressource importante (plus de 1000 pages), quant à la description des logiciels développés.

Ces mémoires CNAM n'ont cependant pas pour vocation de constituer une documentation technique, et le nouveau développeur peut souffrir à son arrivée sur le projet d'un manque de repères, quant aux attentes précises des services offerts par l'application.

Les spécifications ne sont que très peu décrites, les attentes fonctionnelles des méthodes inexistantes. L'outil Javadoc, dont la génération partielle est automatique dans les IDE, est pourtant d'une aide précieuse pour décrire la signature des méthodes : au minimum, ce que la méthode attend en entrée, ce qu'elle fournit en sortie, et comment elle le réalise. La Javadoc n'a malheureusement jusque là été que très rarement exploitée dans le projet, ou alors elle est obsolète, invalide, mais la plupart du temps inexistante.

Ce stage est donc l'occasion de mettre en place plusieurs supports et types de documentation décrits ci-dessous.

9.5.1 DocBook

DocBook est un langage de balisage développé à l'origine par l'éditeur O'Reilly, maintenu et standardisé aujourd'hui par le *DocBook Technical Comittee* du consortium OASIS[12]. La grande force de DocBook est de séparer le contenu de la présentation : comme le décrit un de

11. HyperCarte : publications [en ligne] http://hypercarte.imag.fr/publications.html (consulté le 10 juin 2010).

12. *Organization for the Advancement of Structured Information Standards* [en ligne] http://www.oasis-open.org/home/index.php (consulté le 16 décembre 2010).

ses auteurs, Norman Walsh [Wal99], le format d'écriture XML est totalement orienté sémantique, il n'inclut aucune information de mise en forme. Des feuilles de sytles XSLT (*eXtensible Stylesheet Language Transformations*) permettent de transformer les sources XML vers les formats HTML, PDF, etc. Selon Wikipedia, DocBook est en train de s'imposer comme le format standard pour la documentation logicielle, non seulement dans la communauté du logiciel libre, mais également dans l'industrie.

Je développe depuis 2004 un projet personnel multiplate-forme de génération DocBook basé sur Ant et Java. Largement inspiré du guide complet sur DocBook XSL [13] de Bob Stayton [Sta05], je mets à disposition de STEAMER depuis mon entrée dans l'équipe mes compétences en ce domaine par la création du projet LIGforge `docbench` [14]. Centralisant les sources et la méthode de génération en HTML et PDF, ce projet permet de partager entre développeurs les documentations écrites pour les projets de l'équipe, notamment ceux sur lesquels je suis intervenu : GenGHIS, *ESPON Database* et les projets d'HyperCarte. La structure des sources XML des différentes documentations (guides du développeur, manuels utilisateurs, etc.) permet de partager des sections ou chapitres entre plusieurs documents et de générer des documents cohérents quant à leur style et mise en page pour tous les travaux de l'équipe. De plus amples détails sur ce projet transversal `docbench` sont fournis dans l'annexe A.

Dans le cadre du projet du groupe de recherche HyperCarte, plusieurs documentations DocBook sont ainsi écrites et sauvegardées sous `docbench` :

Le guide du développeur : ce document technique est réservé aux développeurs de l'équipe. Il est enrichi quotidiennement par les spécifications, les implémentations, mais aussi des informations sur l'utilisation des outils, des détails techniques et des références pour tout ce qui peut concerner le projet.

Le manuel utilisateur : constituant une requête du contrat ESPON, un manuel utilisateur en ligne et interactif est demandé. Dans la version précédente de HyperAtlas, le menu d'aide renvoie un manuel utilisateur au format Microsoft Word, non adapté à une solution multiplate-forme et Web. Ce document original est donc repris au format DocBook en début de projet. Ce portage est notamment l'occasion de découvrir et d'explorer les logiciels du point de vue de l'utilisateur. A noter que DocBook est là encore d'une aide précieuse pour satisfaire la génération de plusieurs manuels utilisateurs en fonction du client cible. Plusieurs versions du manuel utilisateur sont donc créées en partageant la plupart des pages mais en séparant ce qui les distingue : la déclinaison permet de créer un manuel utilisateur de la version standard `hypercarte_userManual`, un manuel utilisateur de la version ESPON (`ESPON_HyperCarte_userManual`) adapté avec des styles, des logos *ad hoc*.

Le guide d'installation : destiné à l'administrateur recevant la livraison du logiciel, ce guide fournit des instructions pour l'installation de la base de données et la configuration nécessaire au déploiement de l'application Web livrée. Comme le manuel utilisateur, ce document est structuré de façon à partager la plupart des pages tout en factorisant le contenu commun aux versions standard (`hypercarte_adminGuide`) et ESPON (`ESPON_HyperCarte_adminGuide`).

13. *DocBook XSL : The Complete Guide* [en ligne] http://www.sagehill.net/docbookxsl/index.html (consulté le 16 décembre 2010).
14. *Java Workbench for STEAMER DocBook Documentations* [en ligne - accès réservé] https://ligforge.imag.fr/projects/docbench/ (consulté le 16 décembre 2010).

9.5.2 Javadoc

La Javadoc constitue également un point important de la documentation technique. En s'appuyant sur les recommandations de Sun [SM04], l'écriture de la Javadoc permet de disposer au format HTML d'une documentation riche sur l'API (*Application Programming Interface*) du projet.

Une attention particulière est donc également apportée sur ce point, en respectant autant que possible les consignes d'écriture pour les nouvelles classes développées, et en tentant, dans la mesure du possible et du temps imparti, de renseigner le minimum vital sur le source existant. Nous profiterons de l'IDE Eclipse pour la génération automatique de la signature des méthodes et des en-têtes de fichier (version du fichier dans le SCM Subversion, auteur, description).

L'objectif souhaité en début de projet est de diminuer le nombre d'erreurs et avertissements lors de l'exécution de la commande `javadoc`. La commande retourne 417 avertissements sur la révision 0 du projet (version 1.2.9). Encore une fois, l'utilisation de Ant et l'exécution automatique de la cible `javadoc` permettent de suivre au jour le jour cet objectif.

9.5.3 Outils de conception UML et de bases de données

Le guide du développeur décrit précédemment est enrichi au fur et à mesure du maximum de détails techniques sur le projet, les spécifications, la conception et l'implémentation des évolutions. Lors des phases de conception, le langage de modélisation graphique UML (*Unified Modeling Language*) est utilisé pour ce projet. Parmi les treize diagrammes disponibles dans la version 2 de UML, les diagrammes de classes et de cas d'utilisation sont les deux principaux types qui enrichissent aujourd'hui le guide du développeur.

Si l'offre est riche en outils pour l'édition graphique de tels diagrammes, la difficulté est de choisir l'outil qui permet aux développeurs de partager et maintenir l'existant. Si mes prédécesseurs ont fourni des diagrammes UML dans leurs mémoires CNAM, il est dommage qu'on ne puisse les réutiliser puisqu'ils ne sont récupérables qu'au format image, les sources n'ayant pas été sauvegardées par l'équipe.

Pour remédier à ce problème, un répertoire `doc/uml/` est ajouté au projet Subversion. Notamment pour sa disponibilité sur les systèmes d'exploitation Linux et Windows, le logiciel BOUML [15] a été choisi pour éditer les nouveaux diagrammes UML du projet. Les sources sont sauvegardées sur le SCM, la possibilité d'exporter les diagrammes au format PNG et SVG est exploitée pour les incorporer en tant qu'images dans le guide du développeur ou d'autres documents.

Comme pour les diagrammes UML, le choix d'un outil pour la conception des bases de données est problématique. L'équipe STEAMER achète en octobre 2010 deux licences du logiciel MicroOLAP [16] pour Windows. MicroOLAP a l'avantage de prendre en compte les serveurs PostgreSQL, son extension PostGIS pour les fonctions géométriques, et de permettre la rétro-ingénierie sur une base existante. Pour la maintenance et les évolutions futures du projet, les projets sources ouvrables avec MicroOLAP sont désormais sauvegardés dans le répertoire `doc/db/microOlap` du projet Subversion.

15. *BOUML, a free UML tool box* [en ligne] http://bouml.free.fr (consulté le 10 octobre 2010).
16. *MicroOLAP Database Designer for PostgreSQL* [en ligne] http://www.microolap.com/ (consulté le 10 octobre 2010).

9.6 Intégration continue

Si les outils précédents constituent une base méthodologique pour l'environnement de développement, le concept d'intégration continue assure finalement la non régression de l'application et permet une qualité logicielle dont les principaux avantages sont les suivants :

- les problèmes d'intégration sont détectés et réparés de façon continue, on évite les problèmes de dernière minute ;
- la méthode détecte le code incompatible ou manquant et s'affranchit de la « machine magique » des développeurs ;
- une version de démonstration est toujours disponible ;
- idéalement, un rapport d'avancement est généré quotidiennement.

L'intégration continue prend tout son sens quand elle est automatique : des outils comme Cruise Control [17] permettent de valider toute la chaîne de construction du logiciel à chaque modification dans le SCM.

Aussi, l'intégration continue nécessite-t-elle idéalement une architecture illustrée sur la figure 9.1 et composée de cinq environnements [Agi06] :

1. les postes des développeurs ;
2. l'environnement de développement cible de l'intégration continue : l'application y est déployée de façon automatique, quotidiennement, à chaque modification des sources du SCM ;
3. l'environnement de test destiné au client : ce dernier peut consulter la dernière version stable mise à jour régulièrement ;
4. l'environnement de pré-production : identique à l'environnement final, on y exécute des tests de charge ;
5. l'environnement de production : pour le déploiement de la distribution finale.

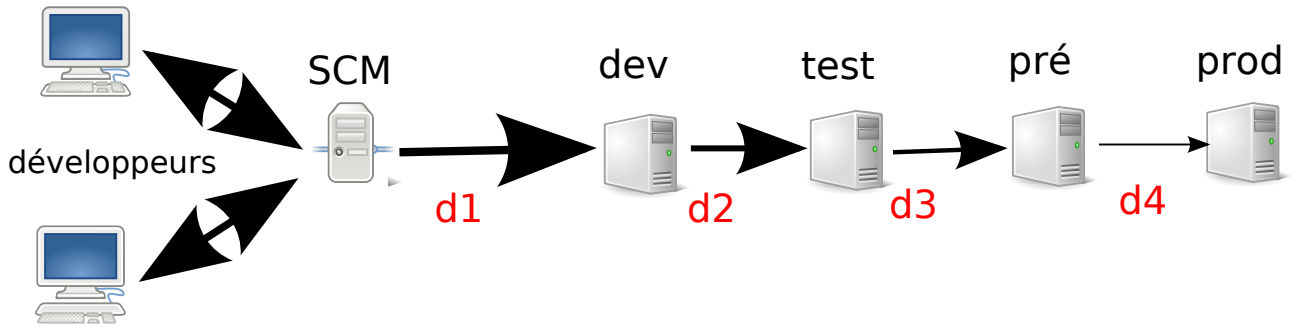


FIGURE 9.1 – Cette figure illustre la chaîne d'environnements et de déploiements dans une architecture idéale d'intégration continue. Les flèches représentent les différents déploiements, leur épaisseur indique la fréquence. Le déploiement d1 entre le SCM et l'environnement de développement est automatique, les déploiements d2, d3 et d4 entre les environnements de test, de pré-production et de production sont manuels [Agi06].

A noter que compte tenu de mon expérience professionnelle, la mise en place et la maintenance de l'intégration continue automatique au sein d'un projet requiert une ressource à temps plein.

17. CruiseControl Home. [en ligne] http://cruisecontrol.sourceforge.net/index.html (consulté le 12 octobre 2010).

En attendant du temps nécessaire à son automatisation, l'intégration continue manuelle mise en place au sein du projet est composée de deux environnements : l'environnement développeur et l'environnement test.

L'environnement développeur joue également le rôle de l'environnement de développement en exécutant quotidiennement la mise à jour des sources et le déploiement de A à Z (*from scratch*) grâce au script Ant (compilation, exécution des tests, mise à jour des documentations). L'environnement test est mis à jour régulièrement sur la plate-forme Marvelig [18], l'accès est réservé par mot de passe aux membres du projet.

9.7 Synthèse de l'environnement

L'architecture de l'environnement de développement mise en place sur le projet est représentée sur la figure 9.2. Nous y retrouvons les principales entités et outils décrits précédemment :

- suivi du projet et tâches à réaliser sont accessibles en ligne sur la plate-forme LIGforge : tous les membres de l'équipe peuvent consulter l'avancement du projet, l'outil génère des diagrammes de Gantt ;
- archivage des sources du projet et des documentations sur le SCM Subversion de la plate-forme LIGforge ;
- construction automatique des délivrables avec des tâches Ant sous la forme d'archives compressées :
  - `tout.zip` sur la figure 9.2 représente une distribution complète avec les sources, les documentations internes (Javadoc et guide du développeur) et les logiciels, archive réservée à l'équipe ;
  - `client.zip` sur la figure 9.2 représente la distribution livrée au client avec les documentations externes (manuel utilisateur, guide de l'administrateur) et les logiciels.

  Ces distributions sont accessibles en ligne aux membres de l'équipe sur un serveur de fichiers.
- déploiement régulier de versions stables à tester sur la plate-forme de prototypes Marvelig.
- les bugs constatés sont remontés sur le gestionnaire de bugs du projet, dans le même environnement de projet offert par LIGforge.

L'évolution majeure à apporter à cet environnement consiste à mettre en place une intégration continue automatique sur l'environnement de test de la plate-forme Marvelig. Pour l'instant, ce serveur consiste à héberger les versions stables en les déployant de façon manuelle. A l'heure de l'écriture de ces lignes, la mise en place sur cette plate-forme d'un déploiement quotidien, par exemple à minuit *night build*, à partir des dernières sources récupérées sur le SCM, est en cours. La disponibilité d'une version dâtée du jour courant est ainsi l'assurance que les dernières sources ont passé avec succès les tests d'intégration.

18. Prototype HyperCarte sur Marvelig. [en ligne] https://marvelig.liglab.fr/doku.php/prototypes/hypercarte_descriptif (consulté le 28 octobre 2010).

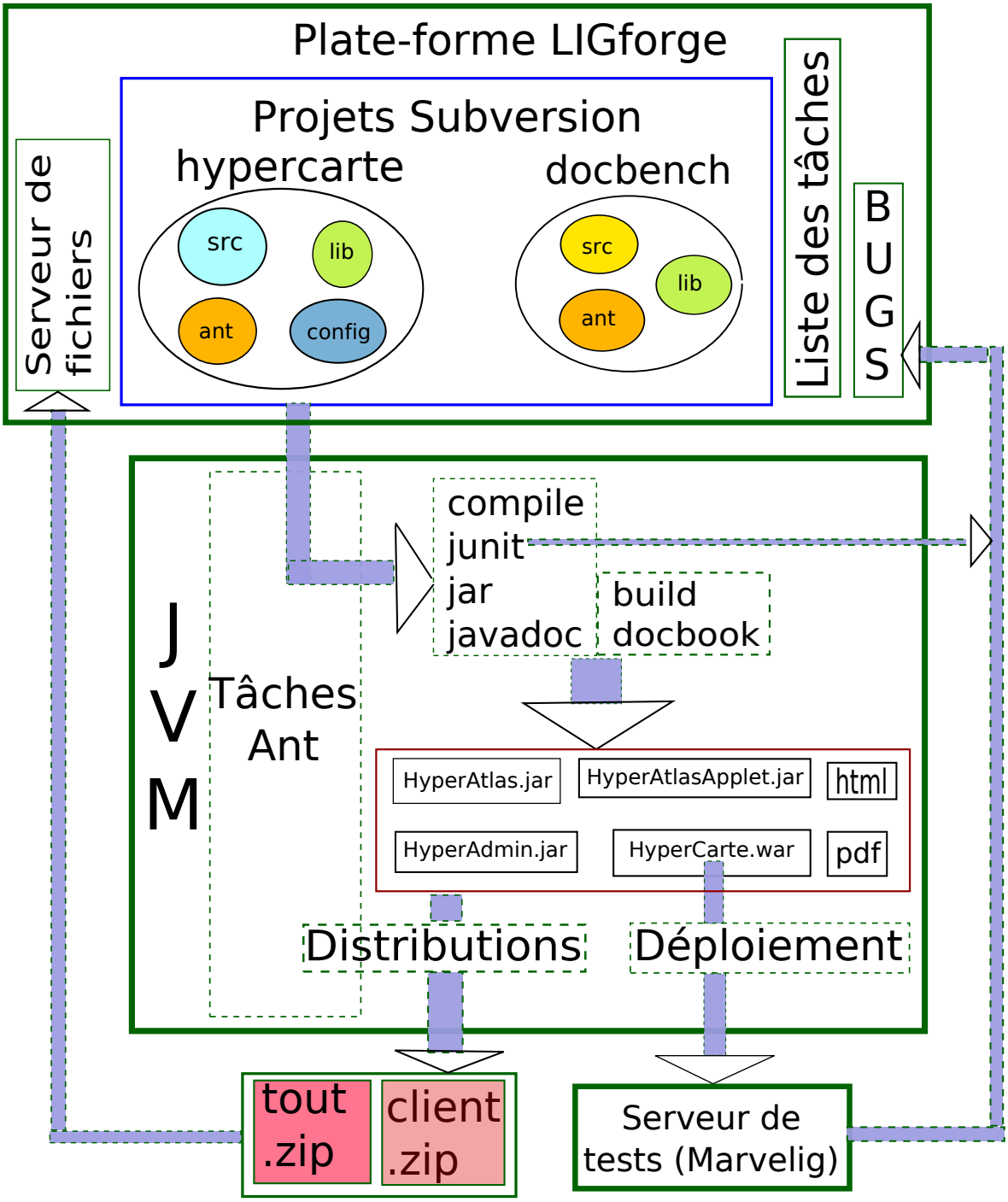


FIGURE 9.2 – Ce diagramme illustre les outils utilisés et le flux de données entre les différentes entités de l'environnement de développement.

Chapitre 10

L'application Web

10.1 Objectif

Le contrat *ESPON HyperAtlas Update* demande une version Web du logiciel HyperAtlas sous la forme d'une *applet*. Le contrat stipule non seulement de développer mais aussi d'intégrer cette *applet* sur le site d'ESPON selon des contraintes de style et de mise en page en concordance avec le site existant. Si l'*applet* HyperAtlas est finalement intégrable dans une page Web HTML via une balise et trois lignes de code, d'autres demandes du cahier des charges visent à compléter la livraison finale par un ensemble de pages : une page listant les jeux de données disponibles, une aide utilisateur en ligne, un outil d'intégration HyperAdmin.

Le contrat *ESPON Database* auquel répond également STEAMER ayant été l'occasion de développer une application Web en Java satisfaisant le client, l'équipe du groupe de recherche HyperCarte décide de capitaliser son expérience en livrant l'« *ESPON HyperAtlas* v2 » sous la forme d'une archive application Web, un fichier `.war`.

Ce chapitre présente cette application réalisée pour englober selon une architecture client-serveur les services demandés par le contrat *ESPON HyperAtlas Update*. La section 10.2 décrit les attentes de cette application, la section 10.3 présente une vue générale de l'architecture proposée en justifiant les choix techniques entrepris.

10.2 Cas d'utilisation

L'application Web du groupe de recherche HyperCarte s'adresse à plusieurs catégories d'utilisateurs. Un utilisateur quelconque peut accéder via son navigateur à un certain nombre de services de l'application Web. L'application propose également des pages réservées à des utilisateurs qui doivent s'authentifier et disposer d'un compte. Pour distinguer les utilisateurs et les pages auxquelles ils ont un droit d'accès, un système de statuts est mis en place au sein de l'application, les statuts s'organisent selon une hiérarchie ordonnée où le statut du niveau N permet toutes les actions du niveau N-1. Les différents statuts sont :

1. « Anonyme » : ce niveau permet essentiellement d'exécuter HyperAtlas (cas d'utilisation `exécuter hyperatlas` sur la figure 10.1), à condition d'avoir accepté les conditions (cas d'utilisation `accepter les conditions`), à partir des jeux de données disponibles (cas d'utilisation `lister les jeux de données`) ;
2. « Enregistré » : ce niveau permet, une fois authentifié, d'exécuter HyperAdmin.

3. « Avancé » : ce niveau permet, une fois authentifié, de gérer les jeux de données disponibles dans l'application (fichiers .hyp).
4. « Administrateur » : ce niveau permet toutes les actions précédentes, plus la gestion des utilisateurs et des paramètres de l'application.

A noter que les statuts supplémentaires suivants sont prévus dans l'application bien qu'ils ne soient pas utilisés pour l'instant : « Testeur », « Editeur » et « Développeur ».

La figure 10.1 page 71 illustre selon le formalisme des cas d'utilisation UML les services disponibles dans l'application Web en fonction du statut de l'utilisateur. Sont représentés sur ce schéma l'héritage entre les différents statuts d'utilisateurs, et les actions disponibles pour chacun d'entre eux. Les cas d'utilisation du schéma sont décrits ci-dessous.

- Les actions disponibles pour un utilisateur anonyme sont :
  - **accueillir** : point d'entrée de l'application ;
  - **lister les jeux de données** : page présentant les `.hyp` disponibles et pouvant être ouverts avec HyperAtlas ;
  - **exécuter hyperatlas** : exécution de l'*applet* HyperAtlas, une fois que l'utilisateur a validé le formulaire d'acceptation des termes et condition d'utilisation **accepter les conditions** ;
  - **accepter les conditions** : formulaire affichant une licence que l'utilisateur doit accepter et valider avant d'exécuter l'*applet* ;
  - **demander un compte** : formulaire permettant à l'utilisateur de demander un compte utilisateur.
- Les actions disponibles pour les utilisateurs enregistrés (les pages de formulaires pour les actions « s'authentifier » et « demander un mot de passe oublié » sont proposées à tous sur l'interface, mais ne sont utiles que pour les utilisateurs enregistrés) :
  - **demander un mot de passe oublié** : depuis la page d'authentification, un lien permet à l'utilisateur de taper son adresse de messagerie électronique dans un formulaire pour récupérer par courriel son nom d'utilisateur et son mot de passe.
  - **s'authentifier** : formulaire d'authentification pour plus de services.
  - **se déconnecter** : simple lien du menu pour quitter la session authentifiée courante.
  - **exécuter hyperadmin** : permet d'utiliser le logiciel HyperAdmin afin de créer un nouveau jeu de données à partir de fichiers Excel et MapInfo MIF-MID.
- Le statut avancé permet le cas d'utilisation **gérer les .hyp**, et notamment de soumettre un nouveau jeu de données qui sera disponible à tous les utilisateur sur la page **jeux de données**).
- Le statut administrateur permet d'accéder au cas d'utilisation **administration** composé des extensions suivantes :
  - **gérer les paramètres** : permet de gérer les paramètres de l'application, comme par exemple les paramètres JDBC (*Java DataBase Connectivity*) ou le niveau des logs générés par l'application ;
  - **gérer les utilisateurs** : permet d'accepter ou de refuser les nouvelles demandes de comptes, d'éditer modifier ou supprimer les comptes existants.

10.3 Architecture

La réalisation de l'application Web embarquant HyperAtlas et HyperAdmin profite de l'expérience acquise lors du développement de l'application Web *ESPON Database Extraction Tool*.

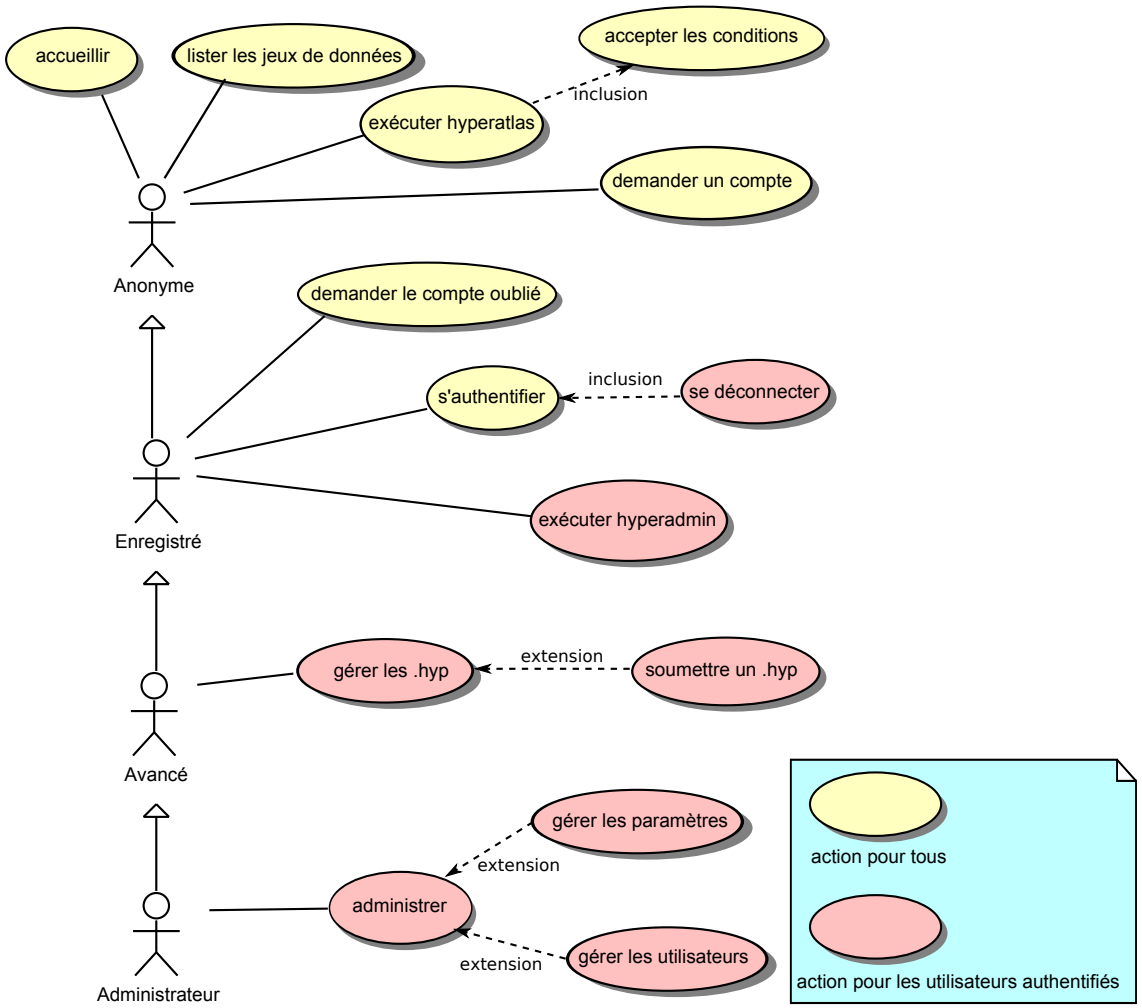


FIGURE 10.1 – Cas d'utilisation de l'application Web en fonction du statut de l'utilisateur.

Réalisée dans le cadre du projet *ESPON Database*, une première version en est livrée en mars 2010. Destinées au même client, partageant un ensemble commun de contraintes et fonctionnalités, les deux applications Web sont ainsi conçues de manière identique, selon une architecture décrite ci-dessous.

La figure 10.2 illustre tout d'abord les différentes entités de l'architecture : le navigateur Web du client, un serveur Web (exemple : Apache) qui reçoit les requêtes HTTP (*HyperText Transfer Protocol*) et les transmet au serveur d'applications Web Java (Tomcat, Glassfish). Ce dernier héberge l'application et interagit avec une base de données PostgreSQL PostGIS.

Le principal objectif du modèle d'architecture MVC (Modèle Vue Contrôleur) mis ensuite en œuvre est de séparer les données, leur présentation et les traitements. Le modèle constitue le cœur du comportement de l'application, la couche « métier ». Il assure les calculs et la gestion des données, typiquement en relation avec un SGBD (Système de Gestion de Base de Données). Le modèle se veut indépendant de la présentation des données.

L'utilisateur interagit avec la vue qui présente les données et reçoit les actions de l'utilisateur (soumissions de formulaires, mouvements de souris, ...).

Le contrôleur gère les évènements, il synchronise la vue et le modèle. Typiquement, le contrôleur analyse la requête du client, appelle le modèle approprié qui réalise le traitement et renvoie le

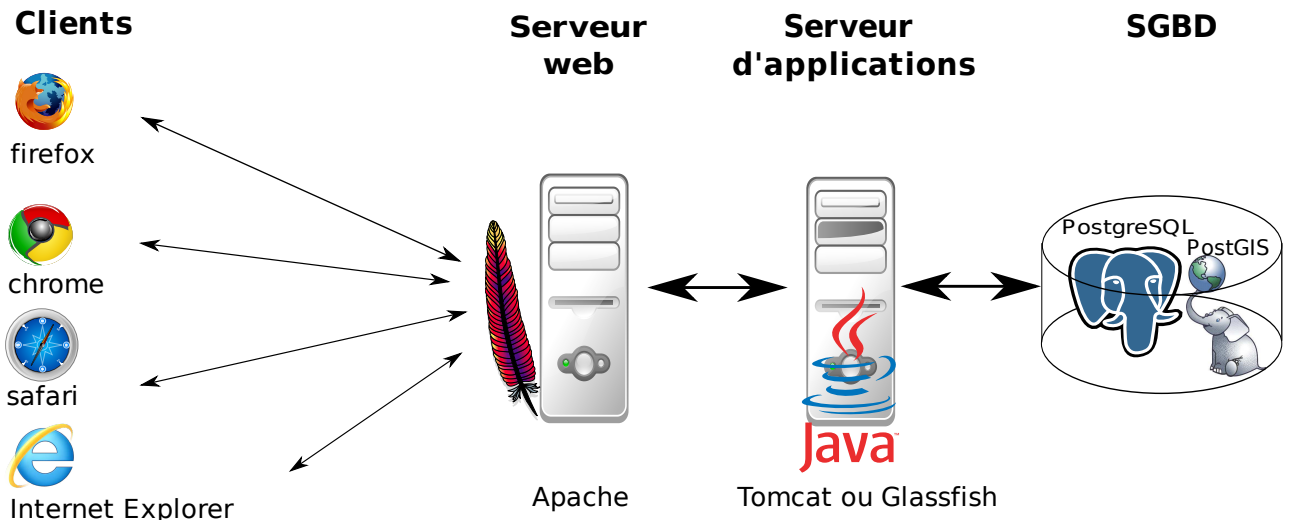


FIGURE 10.2 – Les différentes entités de l'architecture.

résultat brut au contrôleur. Ce dernier transmet finalement à la vue notifiée de se mettre à jour et de présenter à l'utilisateur le résultat de sa requête.

Si MVC permet théoriquement le maintien et l'évolution des couches indépendamment, dans le cadre des applications Web, le patron d'architecture MVC s'est vu petit à petit préférer le patron MVC2. Dans ce dernier, un seul contrôleur gère l'ensemble des actions. Une vue générale de cette architecture décrite par Sun est résumée sur la figure 10.3 [SM02].

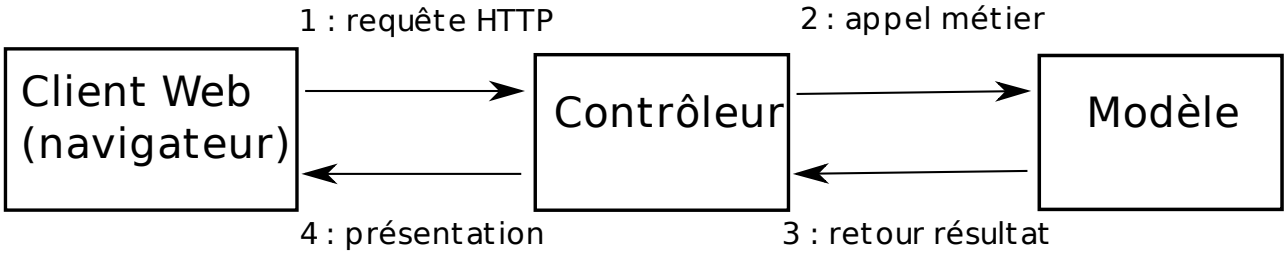


FIGURE 10.3 – Utilisation d'un contrôleur pour la gestion des interactions du navigateur client.

Définies en 1998, les API (*Application Programming Interface*) Java servlets et JSP (*Java Server Pages*) permettent de créer dynamiquement des pages dans un serveur HTTP. Les servlets écrites en Java s'exécutent sur le serveur et gèrent typiquement les traitements, elles représentent l'aspect contrôle et modèle du MVC. Les pages JSP s'occupent de l'affichage et génèrent du code HTML, elles représentent la couche vue. Servlets et JSP sont donc le socle de base de l'application Web. Afin d'accélérer le développement de telles applications, mais aussi pour les tester plus facilement, les maintenir et faire évoluer, des cadriciels (*framework*) Open Source émergent bientôt pour standardiser la conception des applications Web selon une architecture multi-niveaux MVC.

Craig R. McClanahan, à l'origine des spécifications Servlets, JSP et du serveur Tomcat, crée ainsi en 2000 le projet Apache Struts 1 [Rou06], devenu rapidement très populaire. Parallèlement, le cadriciel WebWork dispose d'atouts complémentaires sur l'architecture générale, la possibilité de tests et la gestion des JavaBeans*. Fin 2005, les deux projets fusionnent pour fonder le projet Struts 2 [Apa09].

Struts 2 est un cadriciel orienté actions : selon Ian Roughley [Rou06], Struts 2 est plus précisément un *pull MVC2 framework* : le *pull* suggère que la vue peut interagir avec l'action pour récupérer les données, MVC2 sous-entend que l'action prend plutôt le rôle de modèle que de contrôleur. Struts 2 repose sur une déclaration de l'architecture sous forme de fichiers XML ou d'annotations Java. Les actions encapsulent le traitement à réaliser, elles permettent de manipuler automatiquement les données des requêtes à l'aide de JavaBeans. Struts détermine quel résultat doit être retourné à l'utilisateur en réponse à un traitement. Résolvant plusieurs problèmes de conception en proposant une norme, Struts aide le développeur à organiser la logique de son application en fournissant notamment les services suivants :

- gestion de la navigation ;
- validation des formulaires ;
- internationalisation ;
- bibliothèque de balises (itérateur sur des collections de résultats par exemple) ;
- tuiles de mises en page.

Bien documenté, maintenu par une large communauté de développeurs, le cadriciel Struts 2 a pour principaux concurrents JavaServerFaces et Tapestry. Pour capitaliser le développement réalisé dans le cadre du projet *ESPON Database*, STEAMER reprend donc Struts 2 pour l'application Web du projet *ESPON HyperAtlas Update*. Néanmoins, conscient de la rapidité d'évolutions des différents cadriciels, Struts 2 est utilisé à minima, pour le bénéfice seul des services précédemment cités. Le cœur de métier de l'application est ainsi isolé au maximum pour s'affranchir de l'utilisation de tel ou tel outil. Le portage éventuel de l'application vers un autre cadriciel est ainsi facilité. Bien que conséquente, la réécriture de la vue et du contrôleur reste ainsi indépendante du traitement métier et de la couche d'accès aux données. La figure 10.4 illustre finalement l'architecture retenue pour bénéficier des avantages de Struts 2 tout en prévoyant la possibilité d'une future évolution des couches contrôle et vue : le maximum de logique est externalisé dans la couche *Business Logic*.

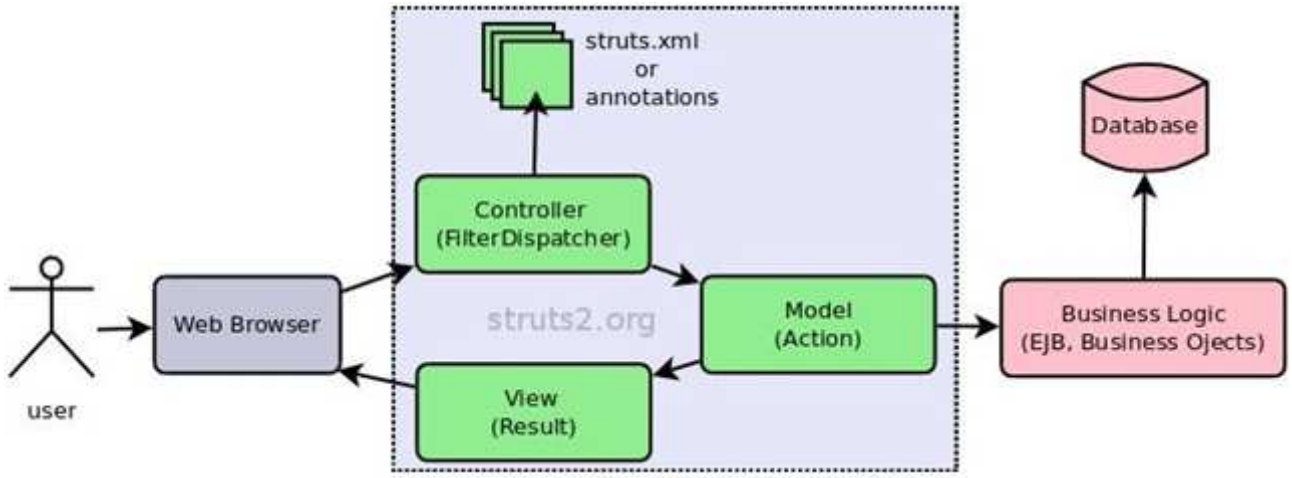


FIGURE 10.4 – Architecture MVC2 avec Struts 2. Source de l'image : [Apa09].

10.4 Principales fonctionnalités

10.4.1 Styles de mise en page

Parmi les possibilités de greffons possibles au cadriciel Struts, la librairie Tiles [1] permet d'organiser toutes les pages selon le même modèle. Pour limiter la duplication des éléments identiques, les pages complètes sont ainsi assemblées à partir de fragments, ou tuiles. La figure 10.5 illustre les différentes tuiles de l'application Web qui composent l'ensemble de la page retournée à l'utilisateur.

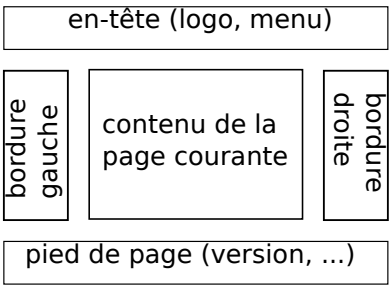


FIGURE 10.5 – Les tuiles composant les pages de l'application.

Pour répondre au besoin introduit dans la partie présentation de disposer d'une application paramétrable à souhait en fonction du client final (version standard, version ESPON), un simple fichier de propriétés composé de paires de clés-valeurs permet d'externaliser la configuration souhaitée. Les valeurs de ce fichier de propriétés sont mis à jour dynamiquement lors de la construction de l'application par le script Ant. Y sont notamment valués le nom du client apparaissant dans les messages et la feuille de style CSS (*Cascading Style Sheet*) à utiliser. La figure 10.6 illustre la personnalisation du style des pages pour les versions standard et ESPON de l'application, ici sur la page de bienvenue. Si cette démarche permet d'associer un style spécifique pour chaque configuration client, elle requiert néanmoins le même modèle de tuiles.

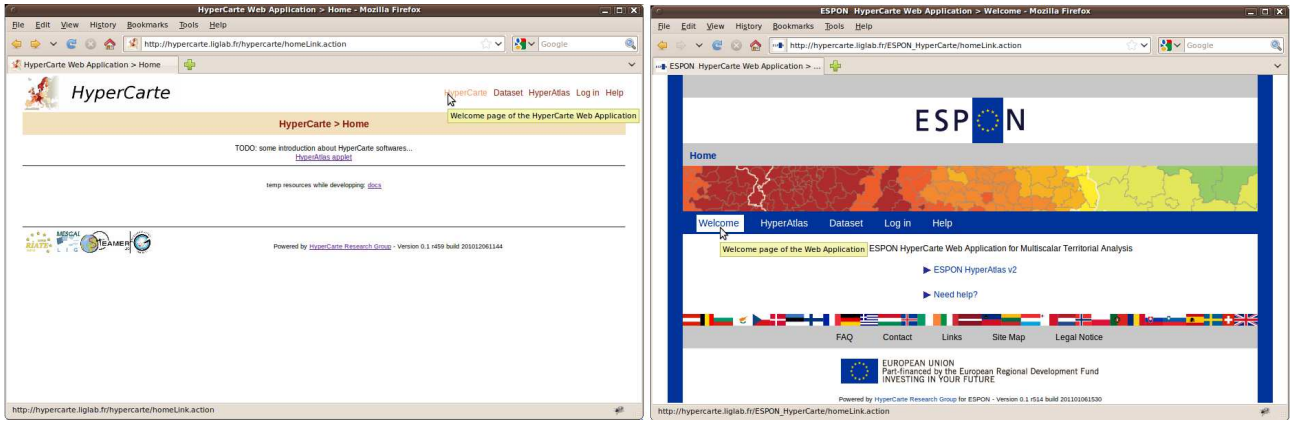


FIGURE 10.6 – Les deux versions de l'application standard (à gauche) et ESPON (à droite) se différencient par les feuilles de styles CSS.

A noter que si différentes feuilles de styles facilient énormément la personnalisation des pages en fonction du client, il serait malhonnête de déclarer l'application Web complètement indépendante

1. Apache Tiles 2 [en ligne] `http://tiles.apache.org/` (consulté le 10 janvier 2010).

de ce dernier : quelques tests sur le nom du client subsistent ainsi dans le code pour adopter des comportements très spécifiques à la version ESPON (`if (espon)...else ...`). La section 10.4.5 décrit la particularité de cette version.

10.4.2 Internationalisation

Destinée au programme ESPON, l'interface utilisateur de l'application Web est disponible aujourd'hui en anglais seulement. L'internationalisation est néanmoins prévue dans la conception. Sur les différentes pages de l'application, tout texte affiché est ainsi référencé dans les JSP par une clé dont la valeur est récupérée par le cadriciel Struts dans un fichier de propriétés. Un fichier de propriétés par langue centralise les paires de clé-valeur pour chaque message textuel : `message_en.properties` et `message_fr.properties` par exemple pour, respectivement, les messages en langue anglaise et en français. Il est également possible de raffiner la localisation par un pays : `message_fr_CAN.properties` pour les messages en français nuancé par les variantes du canadien. Pour afficher le label « Aide » dans la barre de menu, la page JSP utilise la balise Struts `s:text` en référençant une clé pour la valeur de l'attribut `name` :

```
<s:text name="menu.help"/>
```

Suivant la localisation courante du navigateur de l'utilisateur, stockée dans un paramètre de session, Struts va tenter de récupérer la valeur de la clé `menu.help` dans le fichier de propriétés correspondant le mieux à cette localisation. Si le fichier de propriétés localisé n'existe pas, ou si la clé n'est pas trouvée, le message affiché sera la clé elle-même : « menu.help ». Les clés sont donc définies pour chaque langue dans des fichiers de propriétés séparés comme illustré pour cet exemple dans le tableau 10.1.

Les valeurs des fichiers de propriétés peuvent attendre un ou plusieurs paramètres, cette fonctionnalité s'intègre en intercalant des chiffres entre accolades, le chiffre représentant l'index du paramètre à envoyer depuis la page JSP grâce à la balise `param`. Le code suivant appelle la clé `data.number` en lui passant en paramètre le nombre de résultats, lui même récupéré dynamiquement par la balise `s:property` fournie par Struts :

```
<s:text name="data.number" >
   <s:param><s:property value="hypFilesList.size"/></s:param>
</s:text>
```

message_en.properties	message_fr.properties
data.number=There are {0} results menu.help=Help	data.number={0} solutions possibles menu.help=Aide

TABLE 10.1 – Exemple du contenu des fichiers de propriétés pour l'internationalisation.

Les fichiers de propriétés référencent les caractères accentués selon le système Unicode : le caractère accentué « é » est par exemple noté « UE00e9 ».

Par contre, aucune disposition n'a pour l'instant été prise pour tenir compte des langues comme l'arabe s'écrivant de droite à gauche. Les encodages de caractères non occidentaux et dispositions particulières du clavier chinois, par exemple, n'ont pour l'instant pas été étudiés non plus.

Les captures d'écran de l'application Web fournies dans ce document sont donc en anglais uniquement. L'interface utilisateur de l'*applet* HyperAtlas est internationalisée en anglais, français et roumain suivant le même principe de clé-valeur dans des fichiers de propriétés distincts. L'utilisateur peut changer la langue de l'interface graphique de l'*applet* depuis le menu « Outils - langue », mais il ne peut pas changer la langue des pages Web.

10.4.3 Jeux de données

Comme décrit dans les cas d'utilisation de la figure 10.1, l'utilisateur peut consulter les jeux de données disponibles à ouvrir dans l'*applet* HyperAtlas. C'est encore une fois le processus de construction Ant qui permet d'incorporer les fichiers `hyp` à l'application Web livrée, en fonction de la valeur du paramètre `config` fourni en entrée à Ant. Le code de l'application établit la liste des jeux de données retournée à l'utilisateur en parcourant dynamiquement le répertoire où sont attendus ces fichiers `hyp`.

Le nom et la description des jeux de données sont stockés en base, ils sont récupérés en fonction du nom du fichier `hyp` trouvé physiquement sur le serveur. En cas d'invalidité de la connexion avec la base de données, la date de dernière modification du fichier est renvoyée à l'utilisateur. Aussi l'application est-elle livrée avec un ensemble de scripts SQL permettant d'insérer les enregistrements correspondant aux jeux de données `hyp` fournis.

Les utilisateurs enregistrés dont le statut est « avancé » ont la possibilité de compléter la liste des jeux de données disponibles grâce à une page qui leur permet de transférer un fichier `hyp` depuis leur disque vers le serveur. Cette option est décrite section 10.4.7 page 82.

La figure 10.7 montre les jeux de données disponibles :

- dans la version standard, trois fichiers `hyp` sont présents dans le répertoire `resources/datasets/` de l'application : Cameroun, Algérie, Rhône-Alpes. Ces jeux de données ne sont pas enregistrés en base, leur description n'est pas disponible ;
- dans la version ESPON, cinq jeux de données sont fournis avec l'application. L'administrateur reçoit un script SQL pour créer la structure de la base dans laquelle sont ensuite insérés un nom et une description pour chacun des jeux de données livrés.

Find here the list of available datasets (as hyp files)

Hyp name	Last modified	Description
Cameroun.hyp	2010-12-06 11:45:36	No description available...
Algerie.hyp	2010-12-06 11:45:36	No description available...
Rhone-Alpes-V4.hyp	2010-12-06 11:45:36	No description available...

Find here 5 available datasets that can be loaded by ESPON HyperAtlas on clicking the name.

Name	Last modified	Description
ESPON 2007	2007-01-01	Basic indicators for the ESPON area in the NUTS 2003 delineation
Economy and Social Affairs	2010-12-07	ESPON DB and DEMIFER data on EU27+4 area. Lower level: NUTS2 (delination revision 2006)
Land Use	2010-12-07	Derived from Corine Land cover, ESPON 2013 DB data about the land use in 2000 in the NUTS 2006 delineation, includes Croatia and FYRM
Euromed	2010-12-07	The Euromed area covers the ESPON space with its Southern and Eastern Neighborhoods at State level (73 countries). This dataset provides demographic, economic and environmental data
Demography	2010-12-07	ESPON DB, DEMIFER and Map Update on Demography and Migration data about demography and migration on EU27+4 area. Lower level: NUTS2 (delination revision 2006)

FIGURE 10.7 – La page « Jeux de données » liste les jeux de données disponibles, pour la version standard de l'application, à gauche, et pour la version ESPON, à droite. Un lien sur le nom permet de lancer l'*applet* HyperAtlas en chargeant le fichier `hyp` correspondant.

10.4.4 Applet HyperAtlas

La disponibilité d'une version Web de HyperAtlas sous la forme d'une *applet* Java est le premier terme du contrat *ESPON HyperAtlas Update*. Le chapitre 11 décrit comment cette étape a pu relativement facilement être accomplie à partir de la version application de bureau.

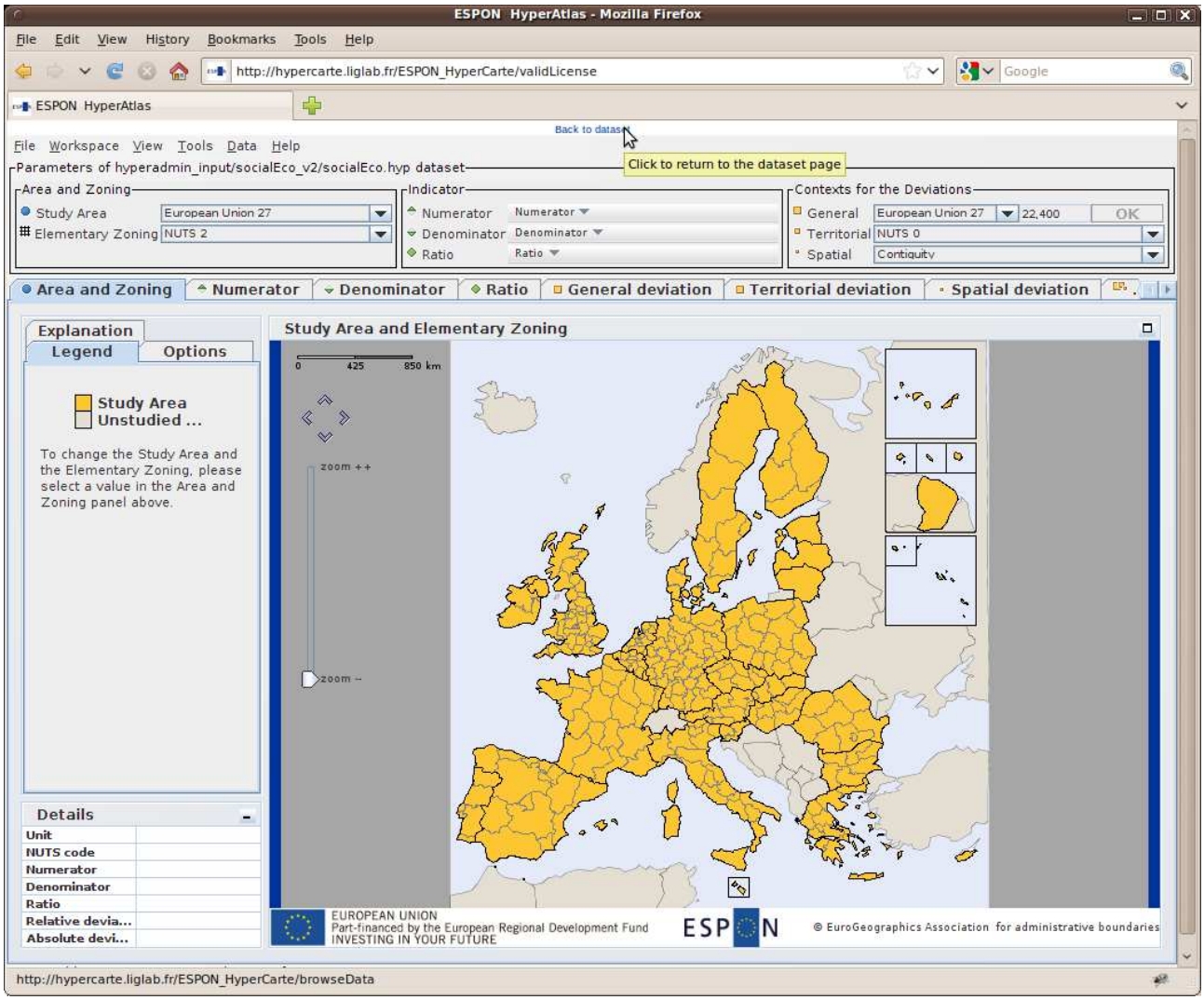


FIGURE 10.8 – Applet *ESPON HyperAtlas* au démarrage. Le lien en haut de la page permet de revenir au menu principal.

Côté application Web, le lancement de HyperAtlas est possible de deux façons : en cliquant le menu HyperAtlas sur la barre de navigation (le jeu de données par défaut incorporé dans l'archive est alors chargé) ou en cliquant le nom d'un jeu de données listé dans le tableau de la page *Dataset*. Dans les deux cas, l'utilisateur est avant tout invité à lire les termes et conditions d'utilisation. En cliquant « J'accepte », son choix est enregistré dans un paramètre de session, il/elle n'a plus à repasser par ce formulaire tant que sa session est active. L'utilisateur est alors redirigé sur une simple page HTML composée d'un lien lui permettant de revenir sur la page précédente et d'une balise `<applet>`, référençant l'archive exécutable Java `HyperAtlas.jar`.

A noter que le processus de construction Ant construit un `.jar` personnalisé client et l'incorpore à l'application Web correspondante (standard ou ESPON). Le code de l'application Web ne se soucie donc pas de la configuration client, le processus est délégué au développeur à travers le script Ant.

Contrairement aux autres pages, l'*applet* est chargée sans tenir compte du style CSS et des tuiles de mise en page. Pour des raisons d'ergonomie, l'*applet* est, en effet, chargée de façon à remplir tout l'espace disponible, cent pour cent de la largeur et de la hauteur de la fenêtre du

navigateur. Le redimensionnement de la fenêtre du navigateur déclenche dynamiquement celui du conteneur Java `JApplet`.

Une *applet* incorporée dans une page Web est exécutée dans le moteur Java (la JRE, *Java Runtime Environment*) du client, une fois téléchargée. Comme indiqué dans le chapitre « Analyse de l'existant », l'*applet* permet, en effet, de lire et d'écrire sur le disque où elle s'exécute. L'*applet* HyperAtlas est signée pour pouvoir passer outre les restrictions standards de sécurité de la JRE : l'utilisateur est donc averti par son navigateur du contenu potentiellement dangereux du code qu'il va exécuter, aussi est-il/elle invité(e) à vérifier le certificat ayant signé l'*applet*.

La figure 10.8 montre la fenêtre du navigateur de l'utilisateur ayant lancé l'*applet ESPON HyperAtlas* depuis le menu principal. Le jeu de données par défaut est chargé.

10.4.5 HyperAdmin Web : particularité de la version ESPON

Une partie de l'application est réservée aux utilisateurs enregistrés en base. La barre de navigation propose le menu *Log in* pour inviter l'utilisateur à entrer son nom d'utilisateur et son mot de passe. Une fois reconnu, la page de bienvenue lui indique qu'il/elle est authentifié(e) et les nouveaux menus auxquels il/elle a maintenant accès : le simple fait d'être authentifié (statut « enregistré ») permet de générer de nouveaux jeux de données, le statut « avancé » permet de soumettre de nouveaux jeux de données, le menu « administrateur » ne permet actuellement que d'exécuter les tests embarqués, mais il est censé à terme permettre la gestion des utilisateurs et des paramètres. Cette section s'intéresse plus particulièrement à l'outil d'intégration *ESPON HyperAdmin* embarqué dans l'application Web.

Comme décrit dans la partie présentation, HyperAdmin attend en entrée trois composantes : une géométrie, une structure et un fichier de statistiques. Le groupe de recherche HyperCarte décide en novembre 2010 d'alléger le processus pour la version ESPON : tenant compte de la difficulté à fournir un ensemble valide de données d'une part, et d'autre part décidé à brider l'application ESPON pour la génération de jeux de données sur l'espace européen seulement, le groupe de recherche propose à ESPON une version intermédiaire de l'outil HyperAdmin. La création d'un `.hyp` depuis l'application Web ESPON substitue les deux premières étapes de soumission d'une géométrie, puis d'une structure, par la possibilité donnée à l'utilisateur de choisir un modèle embarquant à la fois une géométrie et une structure.

La figure 10.9 montre la géométrie et la structure des deux deux jeux de données patrons actuellement proposés à l'utilisateur d'*ESPON HyperAdmin* :

- *eu31nuts2* propose une géométrie selon la projection standard européenne EPSG 3035 * (voir glossaire), une structure composée de 31 pays de la communauté Européenne ou candidats, selon la délinéation NUTS* 2 millésime 2006 ;
- *euromed* propose une géométrie dont la projection particulière est centrée sur la Méditérannée pour englober l'Europe, les pays d'Afrique du Nord et du Moyen-Orient. La structure décompose l'espace selon la délinéation WUTS* dont le plus bas niveau correspond aux pays.

Ces deux modèles ont été suffisament testés aujourd'hui pour en assurer la validité. Une fois le patron de structure choisi (figure 10.10), l'utilisateur est alors invité à télécharger le modèle de fichier stocks associé. Il lui faut remplir ce fichier tableur en définissant correctement les indicateurs qu'il souhaite intégrer, leur nom et description éventuellement dans plusieurs langues, les ratios pertinents éventuels, et l'ensemble des valeurs pour toutes les unités au plus bas niveau de la hiérarchie de la structure correspondante. Le contenu de ce fichier stocks est détaillé dans la section 12.3.3 page 115.

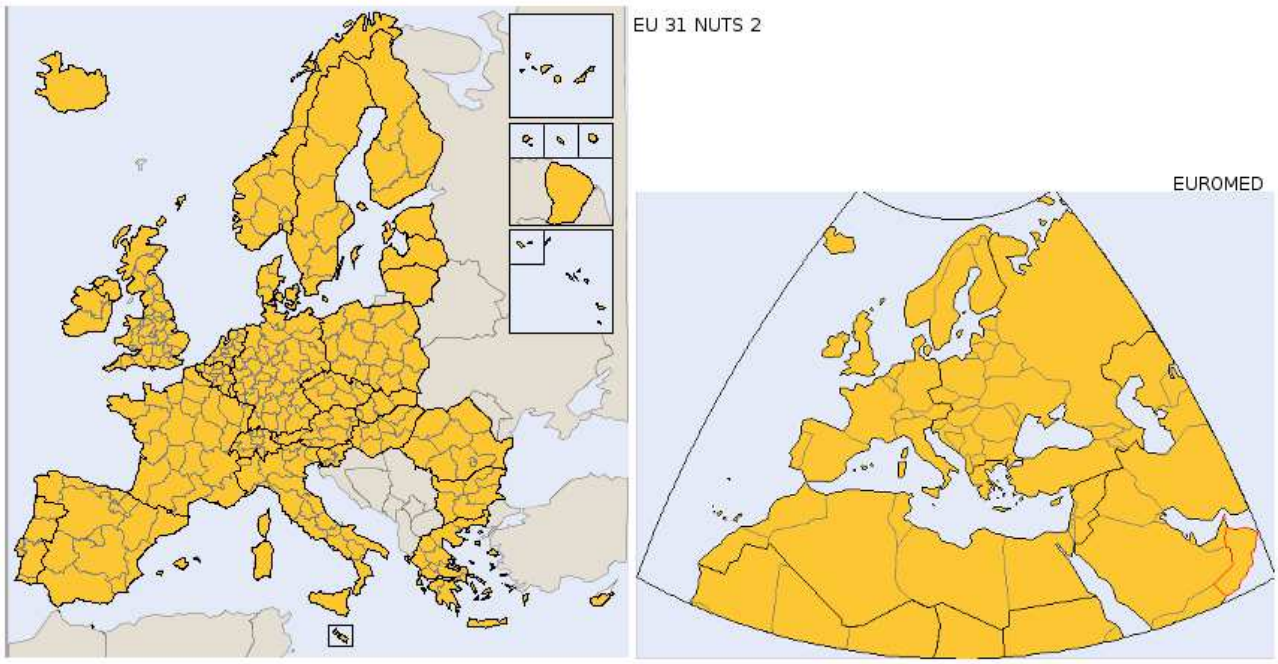


FIGURE 10.9 – Les deux modèles de structure/géométrie proposés dans la version ESPON Hyper-Admin Web.

Cet exercice de saisie est à lui seul un défi... Une fois rempli, l'utilisateur soumet son fichier de stocks à l'application (figure 10.11) qui vérifie que le fichier est bien formé (pas de cases vides, tous les onglets présents, etc.).

L'analyseur renvoie les éventuels messages d'erreur ou redirige l'utilisateur sur un formulaire où il/elle fournit un nom et une description pour le jeu de données qu'il/elle souhaite créer (figure 10.12). L'application construit alors le jeu de données, renvoie les erreurs éventuelles qui ont empêché sa finalisation ou au contraire un lien permettant à l'utilisateur de télécharger le `.hyp` créé.

Les écrans successifs pour la création d'un fichier `hyp` sont illustrés par les captures d'écran suivantes :

1. figure 10.10 : l'utilisateur doit choisir un des modèles de structure/géométrie disponibles ;
2. figure 10.11 : l'utilisateur récupère le modèle de fichiers de statistiques attendu pour le modèle de structure/géométrie choisi précédemment. Le même écran lui permet de charger son fichier de statistiques vers le serveur ;
3. figure 10.12 : le fichier de statistiques fourni par l'utilisateur a été chargé sur le serveur et le document est bien formé (onglets et titres des colonnes). Il/elle est alors invité(e) à fournir un nom et une description pour le jeu de données à créer ;
4. figure 10.13 : la génération du jeu de données a été réalisée avec succès. Quelques messages résumant les différentes étapes de la construction précèdent un lien permettant à l'utilisateur de télécharge le fichier `.hyp` généré.

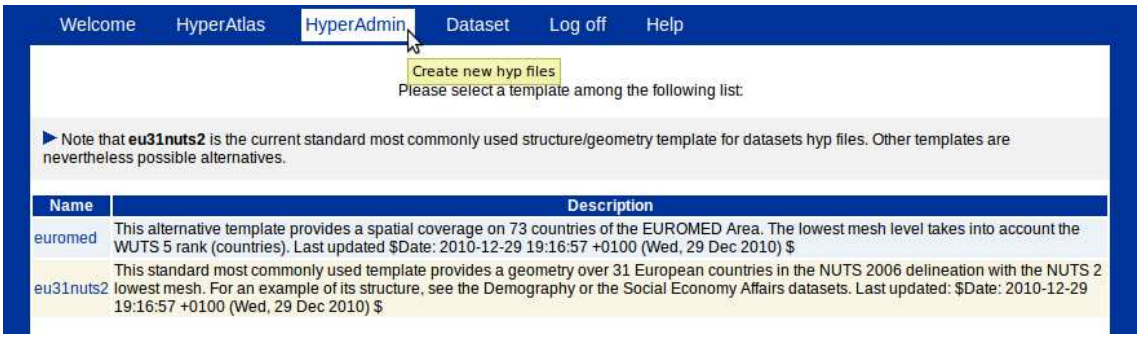


FIGURE 10.10 – Première étape pour la création d'un `.hyp` : choix du modèle de structure/géométrie.

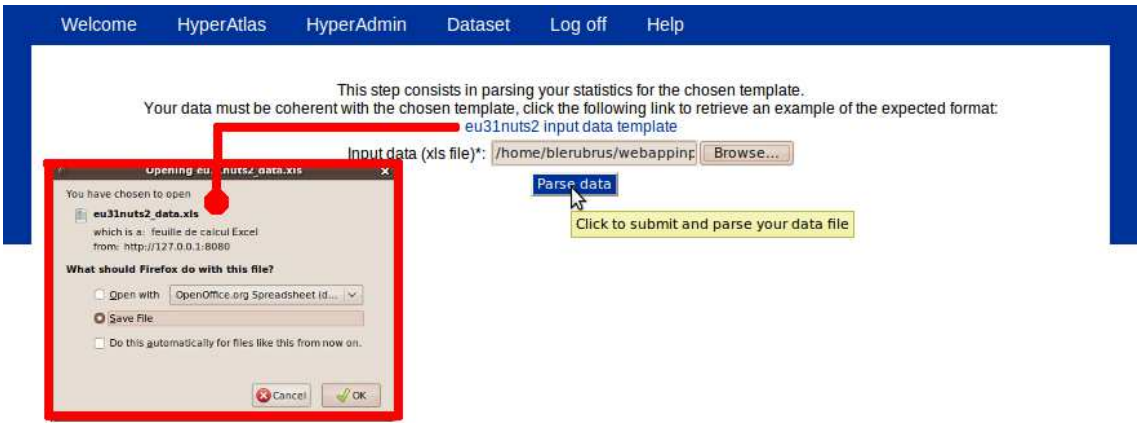


FIGURE 10.11 – Récupération et soumission du fichier de statistiques correspondant au modèle choisi.

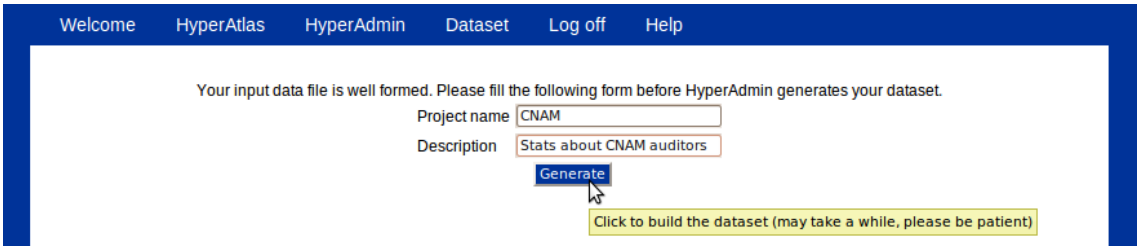


FIGURE 10.12 – Nom et description du jeu de données à générer.

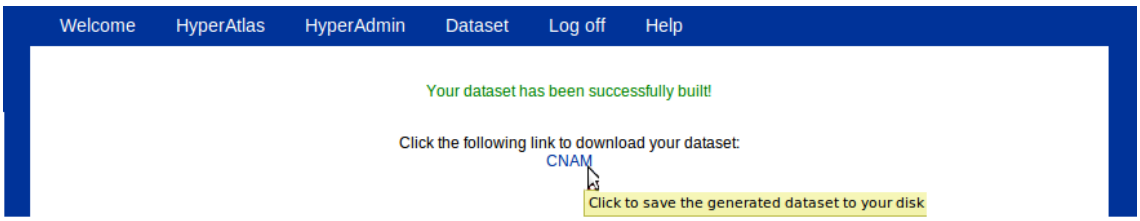


FIGURE 10.13 – Le jeu de données généré peut être téléchargé sous la forme d'un fichier `.hyp`.

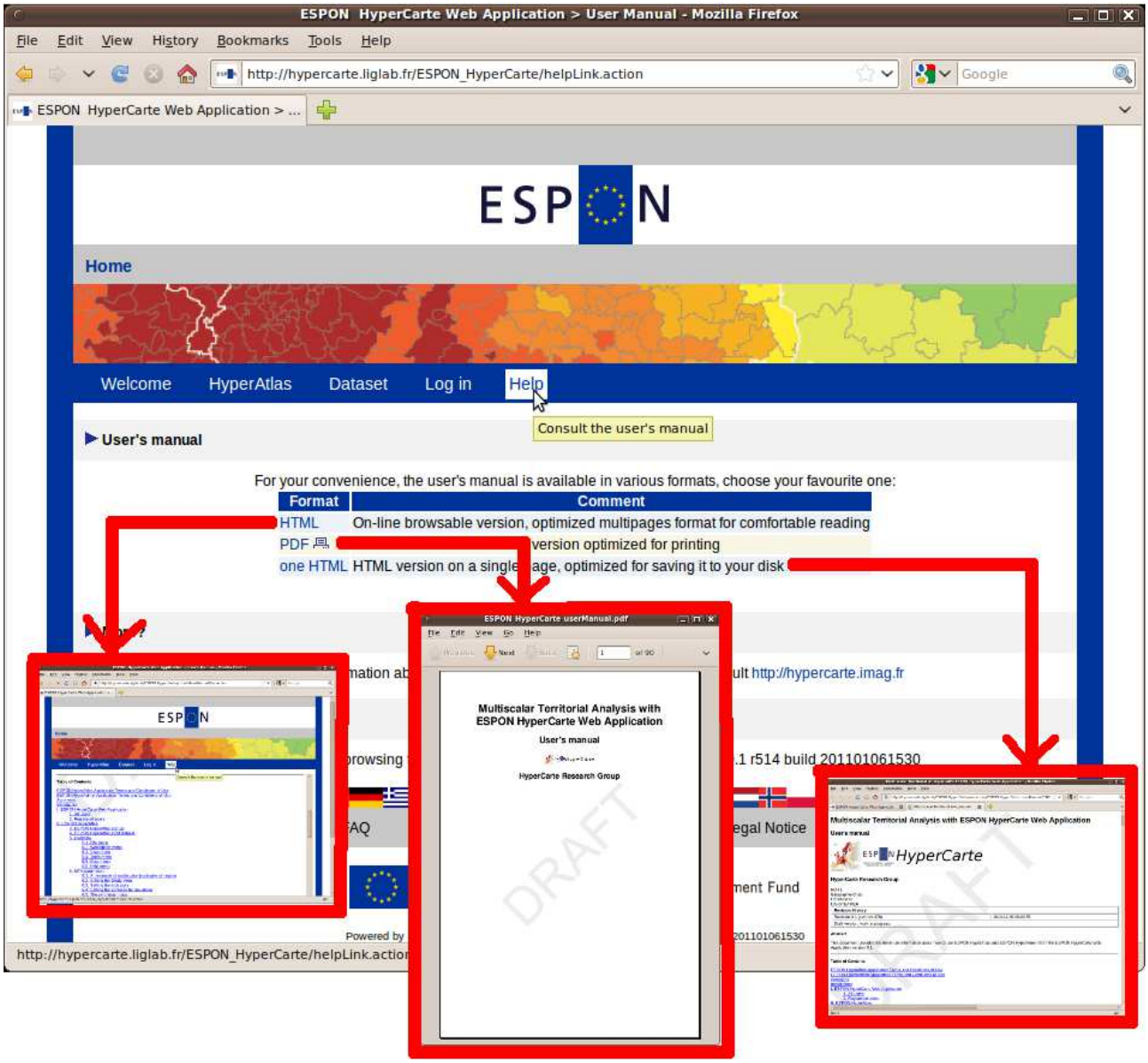


FIGURE 10.14 – Le menu Aide et un aperçu des formats du manuel utilisateur.

10.4.6 Menu aide

Constituant une requête du cahier des charges, un manuel utilisateur a été intégré à l'application Web. Comme décrit dans la section 9.5.1 page 62, le manuel utilisateur est disponible selon plusieurs formats. La figure 10.14 illustre le menu « Aide » de l'application invitant l'utilisateur à choisir le format qui lui convient :

- la version HTML multipage est la plus confortable pour la lecture interactive, elle est intégrée dans la mise en page en tuiles de l'application ;
- la version PDF est optimisée pour l'impression ;
- la version HTML sur une page est optimisée pour le téléchargement, la lecture hors-ligne ou la recherche d'un mot dans l'ensemble du document.

Partant de la documentation obsolète et incomplète au format Microsoft Word, le portage au format DocBook a été l'occasion de mettre à jour les copies d'écran, d'ajouter une information

conséquente pour l'utilisation des nouveaux outils d'HyperAtlas, et une description détaillée et accompagnée d'exemples des fichiers attendus par HyperAdmin.

La documentation pleinement interactive propose 56 figures et copies d'écran parmi les 81 pages de la version multipage HTML et les 90 pages du document PDF générées à partir des sources.

10.4.7 Insertion de nouveaux jeux de données

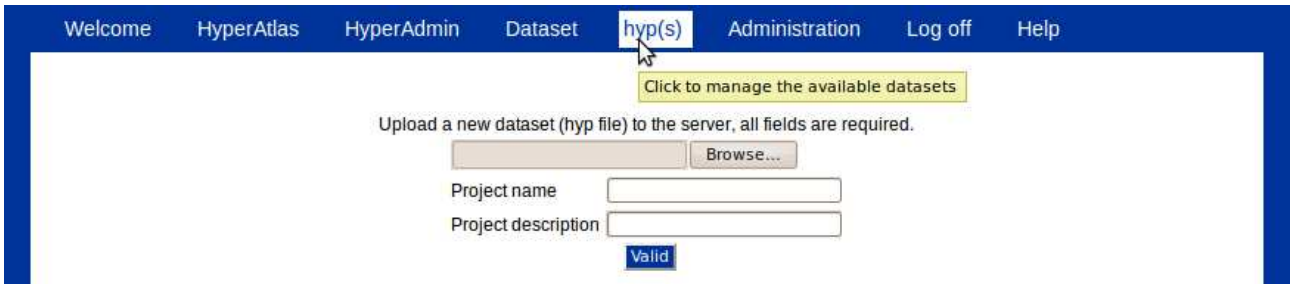


FIGURE 10.15 – Formulaire de soumission d'un nouveau jeu de données.

Réservée aux utilisateurs enregistrés et dont le statut est « avancé », la soumission d'un nouveau jeu de données consiste à enrichir la liste déjà existante et proposée à tous les utilisateurs sur la page « Jeux de données » (voir figure 10.7).

La soumission s'effectue à travers un formulaire composé de trois champs obligatoires (figure 10.15). L'utilisateur doit fournir un nom, une description et transférer un fichier `hyp` de son poste vers le serveur (*upload*). Lors de la validation du formulaire, l'action Struts associée vérifie qu'aucun fichier du même nom n'existe déjà dans le répertoire où sont localisés les fichiers `hyp` déjà disponibles. Si tel est le cas, le fichier est copié à l'endroit attendu, les enregistrements du nom et de la description du jeu de données sont insérés en base. Un extrait du schéma physique de la base de données est donné figure 10.16. La clé étrangère `submitter_id` est renseignée en récupérant l'identifiant de l'utilisateur courant qui a dû s'authentifier pour accéder à cette page.

Un message conséquent indique ensuite à l'utilisateur si l'opération a réussi ou échoué. Le nouveau jeu de données est alors listé dans le tableau de la page « Dataset » et tous les utilisateurs bénéficient de la possibilité d'ouvrir ce nouveau jeu de données avec la version Web de HyperAtlas (voir section 10.4.3 page 76).

Comme suggère le message contextuel au menu de la figure 10.15, il est prévu que cette page soit à l'avenir enrichie d'un formulaire plus complet pour la gestion complète des jeux de données disponibles, avec des possibilités de modification du nom, de la description, de la visibilité, et de la suppression irréversible des fichiers `hyp`.

10.4.8 Menu administration

Le menu « Administration » est réservée aux utilisateurs disposant du statut « Administrateur ». Non prévue dans le contrat *ESPON HyperAtlas Update*, cette fonctionnalité stimule le client quant à des évolutions futures pour la gestion des utilisateurs ou des paramètres d'installation de l'application par exemple.

Actuellement, la page d'administration permet d'exécuter les tests unitaires JUnit embarqués dans l'application. La fonction est conçue essentiellement pour l'administrateur de l'application qui peut en un clic de souris vérifier que la configuration et les fonctionnalités élémentaires sont

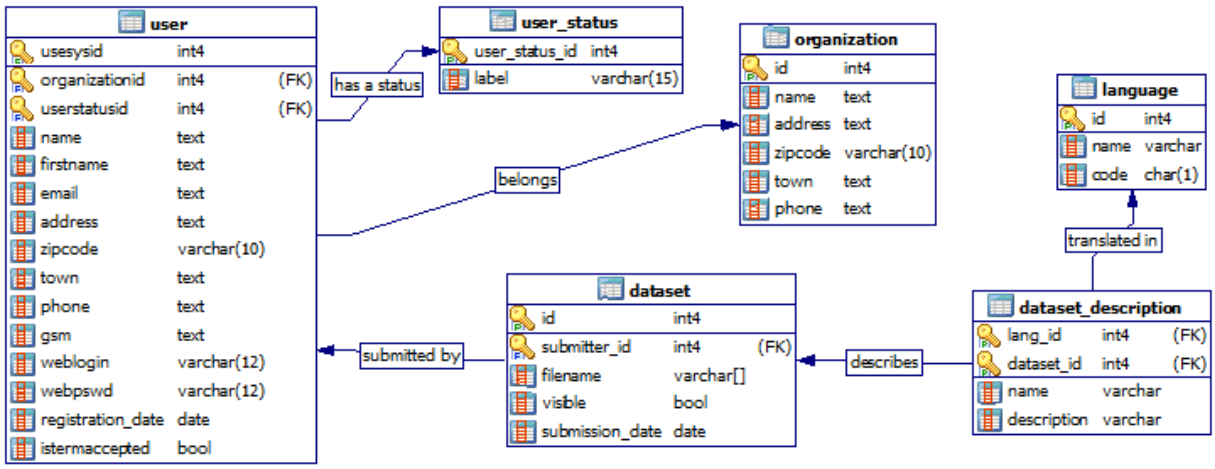


FIGURE 10.16 – Extrait du schéma physique de la base de données : tables concernant les jeux de données disponibles depuis l'application Web.

correctes. Un tableau indique les éventuelles erreurs qui ont empêché les tests de s'exécuter avec succès, comme le montre la figure 10.17, ou, au contraire, que l'installation est correcte, figure 10.18.

1 Failure(s) while executing JUnit tests :-(
17 run tests in 388 milliseconds
Failures details are shown in the below table:

Test name (JUnit class name)	Exception
testGetHypInfo_espon2007(org.steamer.hypercarte.web.model.db.access.DBAccessDatasetJunitTest)	junit.framework.ComparisonFailure: bad en desc expected:<[Provides stocks for ESPON area in the NUTS revision 2003 zoning]> but was:<[Basic indicators for the ESPON area in the NUTS 2003 delineation]>

FIGURE 10.17 – Message d'erreur explicite lors de l'exécution des tests embarqués dans l'application Web.

Successful JUnit tests execution!
24 run tests in 395 milliseconds

FIGURE 10.18 – Succès lors de l'exécution des tests embarqués dans l'application Web.

10.5 Synthèse sur l'application Web

L'application Web permet de répondre à plusieurs besoins exprimés dans le cahier des charges du projet *ESPON HyperAtlas Update* :

- encapsuler une version Web de HyperAtlas sous la forme d'une *applet* Java ;
- fournir aux utilisateurs l'ensemble des jeux de données disponibles ;
- permettre aux utilisateurs enregistrés d'intégrer leurs données statistiques grâce à une version Web de HyperAdmin ;

- permettre à certains utilisateurs de soumettre de nouveaux jeux de données afin de les rendre disponibles à tous ;
- le tout en respectant les styles et mises en page en vigueur sur le site du programme ESPON, les feuilles de styles CSS ayant été récupérées depuis http://www.espon.eu.

L'application Web propose donc à l'utilisateur un ensemble d'outils du groupe de recherche HyperCarte via une interface simple d'utilisation, accessible depuis tout navigateur standard moderne.

Côté développement, l'application repose sur un socle lui permettant de pouvoir *a priori* évoluer facilement tant au niveau de l'interface graphique qu'au niveau d'éventuelles nouvelles fonctionnalités. L'utilisation du framework Struts 2 a permis une implémentation multi-couches pour séparer tant que possible la logique application du contrôle et de la vue.

Si les critères contractuels sont remplis pour la version attendue fin décembre 2010, des besoins supplémentaires évidents restent à implémenter, comme par exemple :

- une interface d'administration des utilisateurs : ce module devrait profiter prochainement des développements effectués dans ce sens dans le projet *ESPON Database*, pour lequel une application Web a été développée sur les mêmes technologies ;
- une interface d'adminisration CRUD (*Create Remove Update Delete*) des jeux de données disponibles : actuellement seule la soumission de nouveaux jeux de données n'est possible ;
- une interface de gestion des paramètres de l'application permettant, par exemple, de modifier dynamiquement les paramètres de connexion à la base de données sans redémarrer l'application, les niveaux de logs, de supprimer les fichiers temporaires téléchargés. Bref, des outils simples qui facilitent les tâches de l'administrateur ;
- la possibilité donnée aux utilisateurs de demander un compte ou de récupérer leurs paramètres de connexion dans le cas où ils/elles les ont oubliés.

A plus long terme, l'application Web ouvre la voie du portage des applications du groupe de recherche HyperCarte vers des solutions tournées vers le partage sur l'Internet : services Web, adaptation pour appareils mobiles, les logiciels existants HyperAtlas et HyperAdmin doivent au plus vite s'adapter aux nouveaux besoins et aux nouvelles technologies (voir le mémoire CNAM de Laurent Poulenard).

Chapitre 11

HyperAtlas v2

11.1 Objectifs

Les évolutions attendues dans le cadre du projet *ESPON HyperAtlas Update* ont été décrites dans le chapitre « Cahier des charges ». Nous partons d'un ensemble de fichiers sources et de librairies dont la version est estampillée 1.2.9. L'ensemble est archivé dans le SCM Subversion LIGforge au premier jour du stage. Outre la maintenance corrective et la reprise de l'existant pour le passage effectif à la dualité application de bureau/*applet*, le passage à la version 2 d'HyperAtlas consiste essentiellement à intégrer de nouveaux outils d'analyse spatiale. La descritption, la conception, l'implémentation, et l'intégration de ces nouvelles fonctionnalités sont l'objet des premières sections de ce chapitre.

La prise en compte du paramètre temporel des indicateurs statistiques constitue une seconde justification de l'incrémentation de la version du logiciel. Comme pour la disponibilité d'une version *applet*, cette évolution part d'un existant implémenté en partie dans la version 1.8.1 : en mai 2008, une distribution pour l'EEA (l'Agence Européenne pour l'Environnement) amorçait en effet la prise en compte de la dimension temporelle dans le panneau de paramétrage, lors du choix des ratios pertinents. La piste était ouverte, bien que codée en dur pour un jeu de données créé spécialement à cet effet. En parallèle à la modification du modèle des jeux de données générés par HyperAdmin pour prendre en compte le temps, un objectif majeur de la version 2 d'HyperAtlas est donc de proposer à l'utilisateur un moyen simple de réaliser des analyses multiscalaires territoriales spatio-temporelles.

Enfin, comme énoncé dans le chapitre « Génie logiciel », les objectifs précédents sont à considérer dans une démarche visant à améliorer la qualité globale du produit, du moins pour les nouvelles évolutions (tests, documentation), et, dans la mesure du possible, sur le code existant.

11.2 Carte de synthèse binaire

HyperAtlas dispose à l'origine de l'onglet cartographique « Synthèse » dans lequel il est présenté à l'utilisateur une carte choroplèthe* correspondant à une typologie des différentes unités territoriales de l'aire d'étude en fonction de la valeur de leurs trois écarts général, territorial et spatial.

L'objectif de la nouvelle carte de synthèse, dite binaire, est de permettre à l'utilisateur de confronter deux écarts parmi les trois disponibles. Cette carte choroplèthe* permet d'observer pour chaque unité territoriale la combinaison des valeurs des deux écarts choisis.

Pour réaliser cette carte, le système chromatique proposé par l'UMS RIATE [GL04a] s'inspire d'une solution introduite précédemment par l'ÖIR [1] (*Österreichisches Institut für Raumplanung*). La légende correspondante, semblable à un diamant, est présentée sur la figure 11.1. Cette légende repose sur un système de coordonnées dans un repère dont les axes X et Y correspondent aux écarts choisis, ils sont gradués de 0 à 200. Rappelons qu'un écart de 100 correspond à un ratio de l'unité territoriale courante égal au ratio pour l'ensemble des unités de l'aire d'étude. L'espace des positions est divisé en quatre quadrants de couleurs principales différentes (rouge, bleu, vert, jaune). Chaque quadrant propose ensuite une intensité graduelle de sa couleur principale. La saturation diminue en s'approchant du point de coordonnées $(100, 100)$. Le carré central, en blanc, correspond à des valeurs proches de celles du contexte de référence, pour les deux écarts.

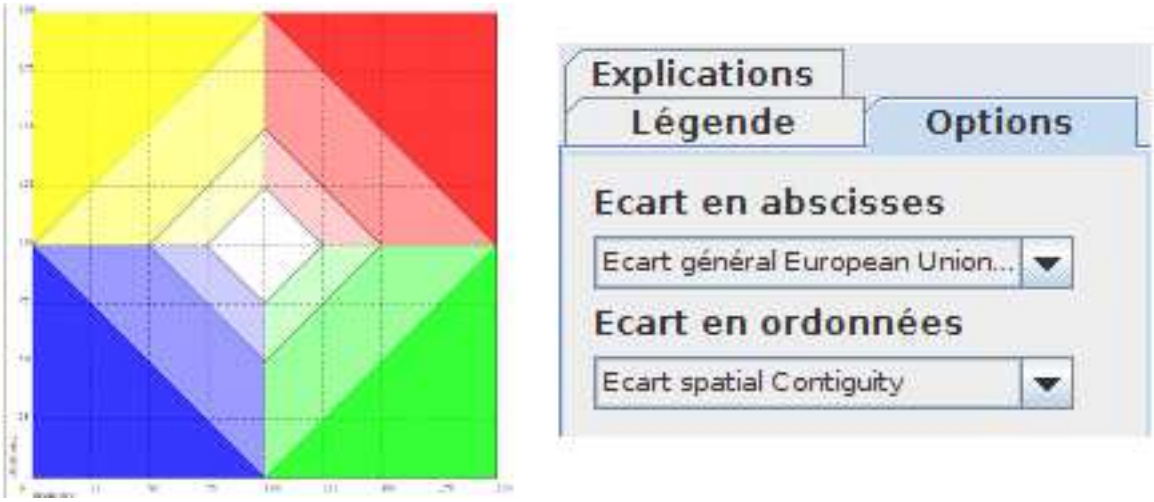


FIGURE 11.1 – Légende de la carte de synthèse binaire et onglet « Options » pour la définition des axes.

L'utilisateur peut choisir les écarts à considérer pour les axes du repère dans un onglet « Options ». La carte de synthèse associée à cette légende présente donc la position de chaque unité en fonction de la typologie des valeurs des deux écarts choisis. La couleur principale définit une situation qualitative d'une région par rapport aux contextes de référence. L'intensité de la couleur définit la situation quantitative en fonction du caractère commun ou exceptionnel de ces valeurs.

La typologie introduite grâce à cette légende suggère une table de contingence montrant les positions relatives des deux écarts pour chaque unité territoriale :

- dans le quadrant bleu, « $--$ », les valeurs des deux écarts sont inférieures aux contextes de références ;
- dans le quadrant vert, « $+-$ », la valeur de l'écart sur l'axe des abscisses est supérieure à celle du contexte de référence, mais la valeur de l'écart sur l'axe des ordonnées est inférieure ;
- dans le quadrant rouge, « $++$ », les valeurs des deux écarts sont supérieures à celles du contexte de référence ;
- dans le quadrant jaune, « $-+$ », la valeur de l'écart sur l'axe des abscisses est inférieure à celle du contexte de référence, mais la valeur de l'écart sur l'axe des ordonnées est supérieure.

Un avantage de cette carte de synthèse est de rapidement identifier les unités présentant une situation paradoxale : une position dans les quadrants verts et jaunes ($+-$ et $-+$) révèlent typiquement des situations contradictoires. La figure 11.2 illustre l'exemple de cette carte de synthèse

1. *Österreichisches Institut für Raumplanung* [en ligne] `http://www.oir.at/` (consulté le 8 juillet 2010).

calculée par HyperAtlas en considérant l'écart général en abscisses (les 31 pays européens) et l'écart spatial (contiguïtés) en ordonnées, le ratio est formé du PIB sur la population totale en 2000. Un simple coup d'oeil à la carte permet d'exprimer que la région parisienne (rouge foncé) est fortement avantagée par rapport à la fois à l'ensemble des pays et à ses voisins, contrairement à la situation de quelques unités d'Europe de l'Est (bleu foncé).

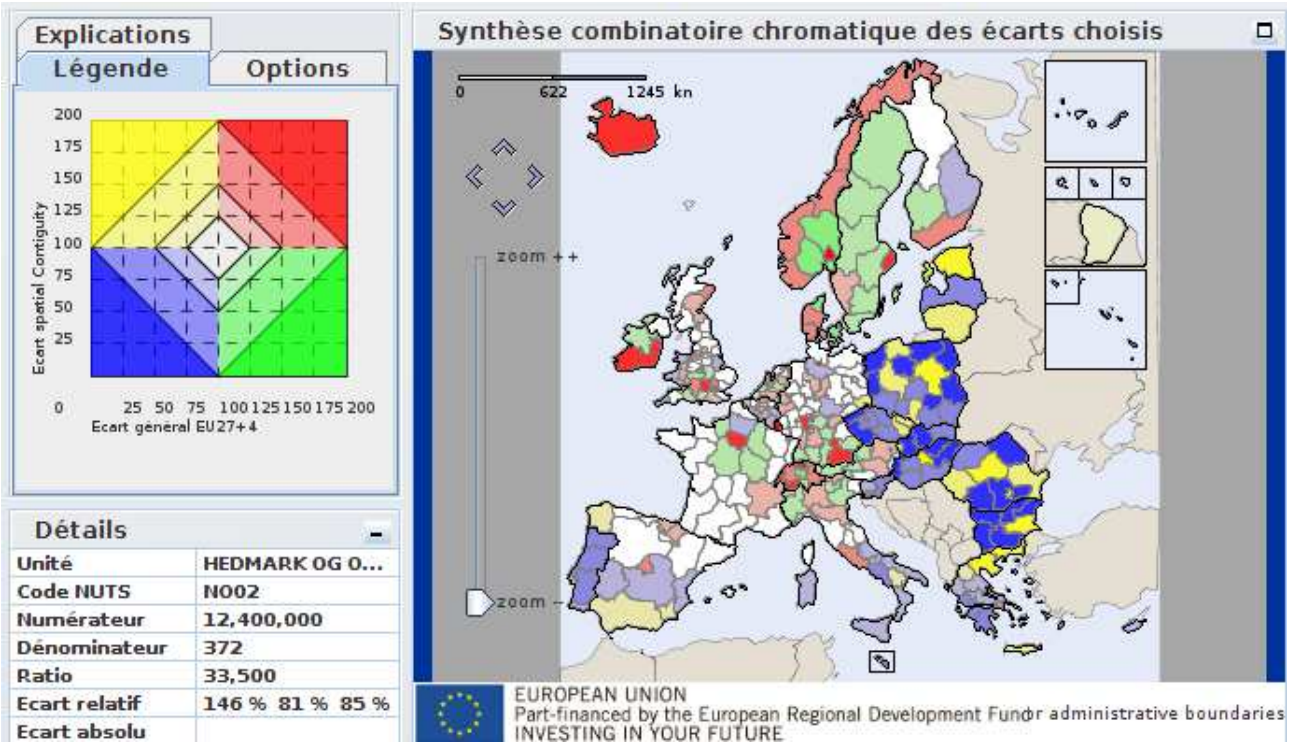


FIGURE 11.2 – Synthèse binaire appliquée au PIB/population en 2000.

11.2.1 Implémentation

Pour implémenter la légende, deux classes essentielles sont utilisées : `DualDeviationLegendPanel` représente la couche vue, `DualDeviationLegendControl` joue le rôle de contrôleur et fournit les paramètres :

- la couleur principale des quadrants, pour l'instant définies comme des couleurs basiques `Color.BLUE`, `Color.GREEN`, `Color.RED` et `Color.YELLOW` de l'API `java.awt` ;
- la composante alpha des couleurs RVB (rouge vert bleu) pour distinguer les différentes intensités des trois couleurs des quatre polygones de chaque quadrant : foncé, moyen, léger et blanc ;
- les maxima du repère (identiques en abscisse et ordonnées, réglés par défaut à 200).

`DualDeviationLegendPanel` utilise l'API Java Graphics2D et son constructeur prend le contrôleur en paramètre. Le dessin du repère en lui-même fait appel à une classe `CoordSysPanel` décrit dans la section 11.2.1 page 90.

Le contrôleur permet de répondre aux questions d'appartenance d'un point à telle ou telle région en implémentant l'interface `DualDeviationControlInterface` proposant les méthodes suivantes prenant toutes en paramètres les coordonnées x et y d'un point (c'est-à-dire ici les valeurs des deux écarts d'une unité territoriale) :

- le point de coordonnées x y est-il dans le repère ? `boolean isInRepere(float x, float y)` ;
- le point est-il le centre ? `boolean isInCenter(float x, float y)` ;
- le point est-il dans le quadrant en bas à gauche ? `boolean isInBottomLeftQuadrant(float x, float y)` ;
- en bas à droite ? `boolean isInBottomRightQuadrant(float x, float y)` ;
- en haut à droite ? `boolean isInTopRightQuadrant(float x, float y)` ;
- en haut à gauche ? `boolean isInTopLeftQuadrant(float x, float y)` ;
- quelle couleur pour ces coordonnées ? `Color getColor(float x, float y)` ;

La politique d'appartenance des axes limitrophes a été arbitrarement définie de façon à ce que chaque quadrant inclue les points situés sur trois de ses quatre côtés. Ce système d'appartenance implique que le centre n'appartient à aucun quadrant. L'algorithme lance donc la méthode `boolean isInCenter(float x, float y)`, avant de déterminer à quel quadrant appartient un point donné.

Dans le cas où un point n'est pas dans le repère, on fait appel à la méthode `getOutOfRepere(x, y)`. Cette méthode retourne noir si une des coordonnées est négative (*a priori* impossible). Quand les deux valeurs des écarts sont positives, elle renvoie la valeur maximale (255) pour le paramètre alpha de la couleur du quadrant auquel le point appartient, au dessus de la première moitié horizontale, au dessus ou à droite du quadrant en haut à gauche, à droite du quadrant en bas à droite.

Une fois le quadrant déterminé, l'algorithme répond à la question : « ce point est-il dans la région foncée, intermédiaire ou claire ? ».

Quatre classes sont dédiées respectivement au quadrant en bas à gauche, à droite, en haut à droite et à gauche. Elles implémentent toutes différemment les méthodes de l'interface `QuadrantInterface` pour déterminer dans quelle région (sombre, intermédiaire ou claire) est située un point donné. Comme illustré sur la figure 11.3, elles héritent toutes les quatre d'une classe abstraite laquelle factorise les propriétés communes (maxima sur les axes du repère et couleur principale).

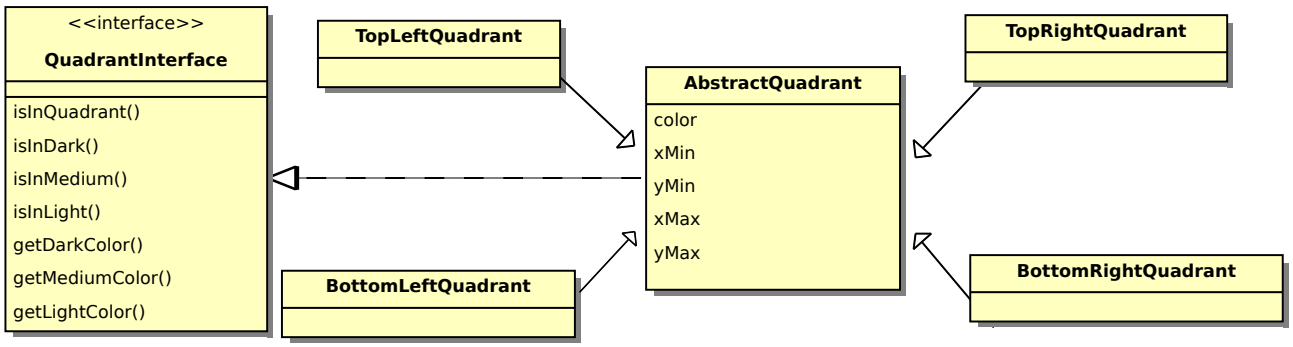
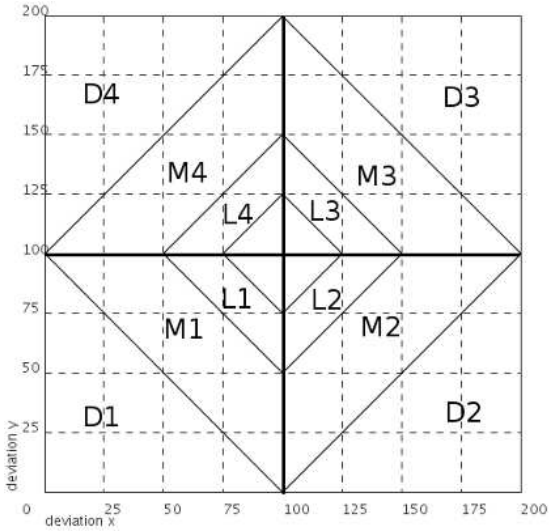


FIGURE 11.3 – Implémentation des quatre quadrants de la légende pour la carte de synthèse binaire.

Pour déterminer dans quelle région du quadrant se situe un point, l'algorithme teste simplement si les coordonnées données vérifient l'inéquation élémentaire de chaque région. Les inéquations utilisent les équations linéaires des segments de droite, elles sont données dans le tableau 11.1 (D pour *dark*, M pour *medium* et L pour *light*).

Afin de débrayer les actions sur l'interface utilisateur de la construction de la légende, la figure 11.4 illustre les nouvelles classes créées à cet effet et comment elles s'intégrent à l'existant.

La classe centrale (en rouge) `DualDeviationSettings` du paquetage `hypercarte.hyperatlas.config` constitue en quelque sorte le pivot contrôleur entre l'interface utilisateur d'HyperAtlas



$D1$	$y \leq 100 - x$
$M1$	$100 - x < y \leq 150 - x$
$L1$	$150 - x < y \leq 175 - x$
$D2$	$y \leq -100 + x$
$M2$	$-100 + x < y \leq -50 + x$
$L2$	$-50 + x < y \leq -25 + x$
$D3$	$y \geq 300 - x$
$M3$	$250 - x \leq y < 300 - x$
$L3$	$225 - x \leq y < 250 - x$
$D4$	$y \geq 100 + x$
$M4$	$50 + x \leq y < 100 + x$
$L4$	$25 + x \leq y < 50 + x$

TABLE 11.1 – Inéquations des différentes régions du repère.

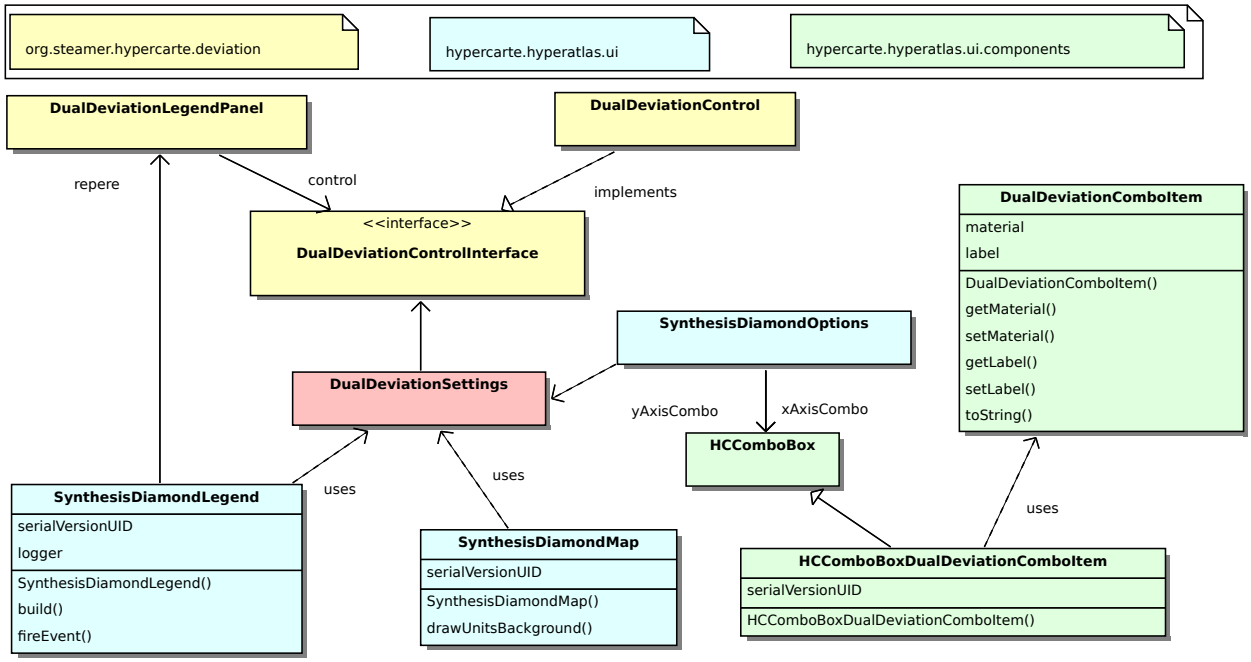


FIGURE 11.4 – Classes intervenant pour la gestion de la carte de synthèse binaire.

(légende, carte et options) du paquetage `hypercarte.hyperatlas.ui` et la gestion/construction de la légende (paquetage `org.steamer.hypercarte.deviation`). La classe `DualDeviationSettings` est un attribut de la classe singleton `Settings`, singleton généraliste de l'application.

L'utilisateur choisit les déviations grâce à deux listes déroulantes (une pour l'axe des X, une pour les Y) d'une instance de `SynthesisDiamondOptions`. A cette occasion, un simple composant JavaBean* `DualDeviationComboItem` a été créé pour nourrir ces boîtes de sélection particulières représentées par la classe `HCComboBoxDualDeviationComboItem`, spécialisation d'une classe déjà existante dans le projet.

Les changements mettent à jour les attributs de `DualDeviationSettings`. Un changement d'options déclenche un évènement « axe modifié ».

La légende `SynthesisDiamondLegend` réagit aux évènements « axe modifié » et met à jour la légende sur les axes en récupérant les attributs de `DualDeviationSettings`.

La carte `SynthesisDiamondMap` réagit également aux évènements « axe modifié » et redessine la couleur de fond des unités en conséquence.

Composant repère orthonormé

Plusieurs nouvelles fonctionnalités d'HyperAtlas v2 font appel à l'affichage d'un repère orthonormé. A cette fin, la classe `CoordSysPanel` a été créée pour répondre de façon générique aux mêmes besoins exprimés pour dessiner la légende de la carte de synthèse binaire, la courbe de Lorenz et le diagramme d'auto-corrélation spatiale.

Cette classe `CoordSysPanel` factorise le dessin du repère et des éléments communs comme les marges, les graduations, les axes, les légendes le long des axes et optionnellement une grille en pointillés.

Un objet `CoordSysPanel` hérite du composant Java `JPanel` et surcharge la méthode `void paintComponent(Graphics g)` pour afficher les différents composants à l'aide de la librairie Java `Graphics2D`. L'ensemble occupe le maximum de place disponible dans le `JPanel` alloué et réagit automatiquement au redimensionnement. La dimension des marges est calculée dynamiquement en fonction de la taille de la police allouée aux caractères formant les légendes et graduations.

La classe `CoordSysPanel` fournit une méthode pour traduire les coordonnées d'un point du panneau Java (l'origine est le coin en haut à gauche, les coordonnées sont exprimées en nombre de pixels) vers les coordonnées du point dans le repère « mathématique » dessiné dans le panneau. Réciproquement, `CoordSysPanel` fournit une méthode pour renvoyer les coordonnées d'un point dans le panneau Java à partir de ses coordonnées dans le repère « mathématique ». Les différentes parties du composant et la conversion de coordonnées entre les deux repères « math » et `JPanel` sont illustrées sur la figure 11.5.

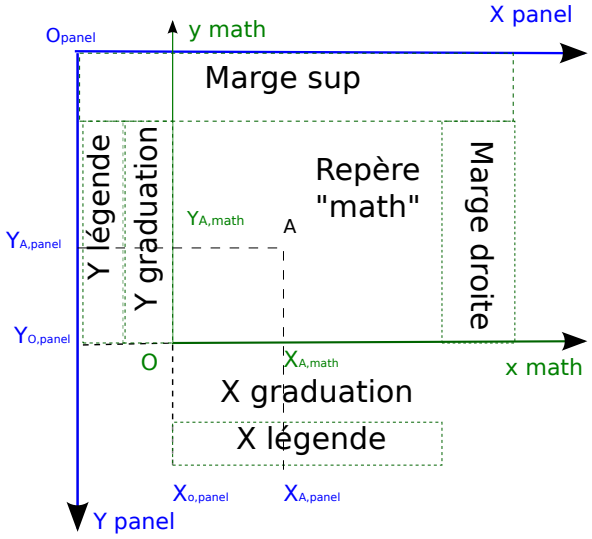


FIGURE 11.5 – Classe `CoordSysPanel` : intégration d'un repère mathématique et de ses attributs dans un panneau Java `JPanel`.

11.3 Mode expert

Un objectif du groupe de recherche HyperCarte est de fournir des outils graphiques simples d'utilisation et accessibles à tous. Le mode expert consiste à proposer aux utilisateurs avancés les nouveaux outils d'analyse spatiale et statistique demandés dans la mise à jour du projet *ESPON HyperAtlas Update*.

Si l'application est proposée par défaut en mode standard, la version 2 de HyperAtlas propose un nouvel élément de menu intitulé « mode expert ». En activation le mode expert, les trois onglets cartographiques des écarts général, territorial, spatial, se divisent en trois fenêtres internes pour proposer à l'utilisateur, en plus de la carte d'écart initiale, les outils supplémentaires suivants :

- une carte de redistribution (dite aussi d'équi-répartition), relative suivant l'onglet choisi aux contextes général, territorial ou spatial ;
- une boîte d'outils statistiques :
  - dans l'onglet général, cette boîte propose le calcul d'indices statistiques et une courbe de Lorenz ;
  - dans l'onglet territorial, des boîtes à moustaches ;
  - dans l'onglet spatial, une représentation dans un repère de l'auto-correlation spatiale entre l'écart spatial et l'écart territorial.

Ces nouveaux outils du mode expert permettent essentiellement la mesure des inégalités. Après la description théorique de ces nouveaux outils, les sous-sections suivantes détaillent leur implémentation et leur intégration dans l'application HyperAtlas.

11.3.1 Passage au mode expert

Au lancement de l'application sont instanciés trois objets de type `DeviationFrameset` initialement composés uniquement de la carte d'écart (mode standard). Un nouvel élément de menu a été rajouté dans le menu : « Activer le mode expert ». La figure 11.6 illustre selon un diagramme de séquence UML le comportement de l'application lorsque l'utilisateur active le mode expert :

1. l'instance de `HCMenuBar` bascule la valeur d'un booléen de la classe singleton `Settings` (de faux à vrai ou inversement lors du passage en mode standard ou expert) ;
2. le label du menu est modifié en conséquence (de « Activer... » à « Désactiver... » ou inversement) ;
3. l'instance singleton `Dispatcher` est averti de l'évènement ;
4. selon le principe de l'observateur écouteur, les trois instances de la classe `DeviationFrameset` vont agir en conséquence en recevant cet évènement. En fonction de la valeur du booléen `statsExpertMode` stockée dans l'objet `Settings`, les objets `DeviationFrameset` modifient les fenêtres internes qui les composent :
   - en passant du mode standard au mode expert, la mise en page est modifiée : la carte de l'écart relatif est diminuée pour lui juxtaposer la carte de redistribution et la boîte statistique. Ces trois fenêtres internes représentées sur la figure 11.7 héritent du composant Java `JInternalFrame` pour bénéficier des possibilités de maximisation, minimisation et iconification ;
   - du mode expert au mode standard, la carte de redistribution et la boîte statistique sont désactivées et la carte d'écart relatif reprend tout l'espace disponible de l'onglet.

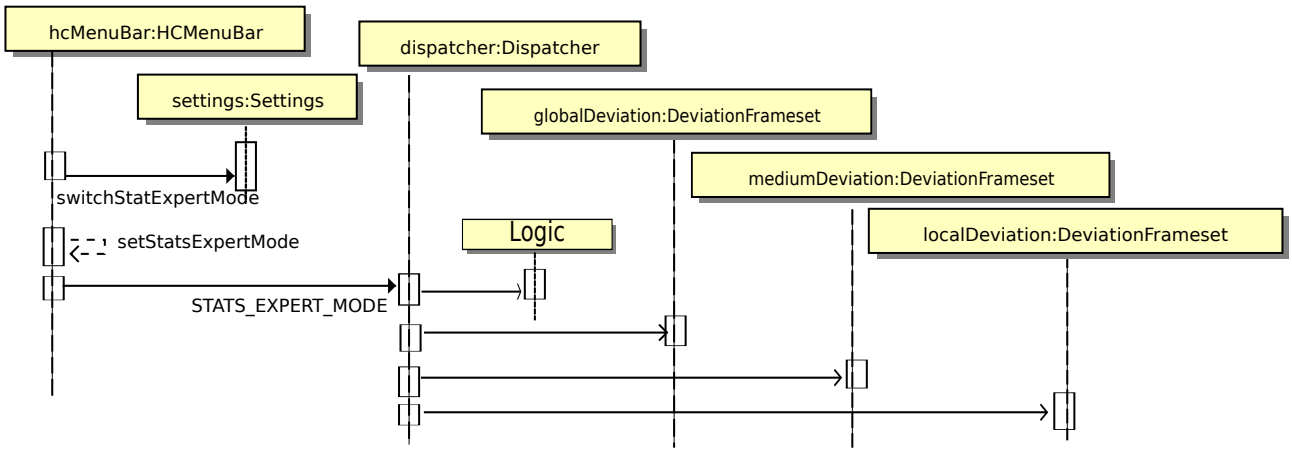


FIGURE 11.6 – Diagramme de séquence illustrant l'activation ou la désactivation du mode expert.

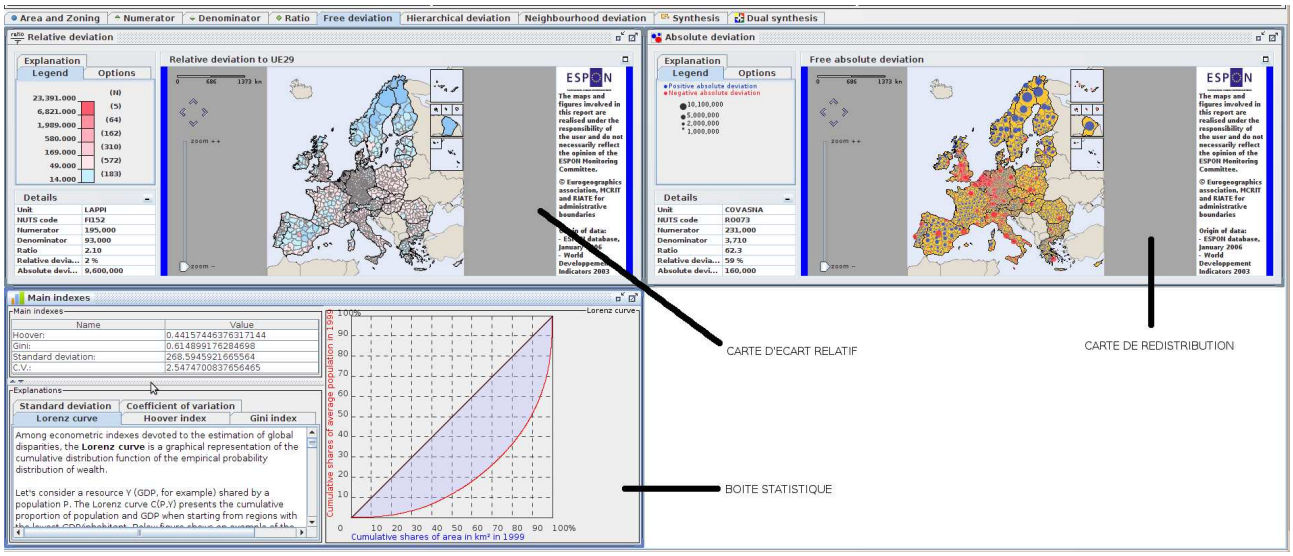


FIGURE 11.7 – Les trois fenêtres internes de l'onglet « écart général » en mode expert.

11.3.2 Cartes de redistribution

La figure 11.8 juxtapose des captures d'écran de la carte de redistribution pour le contexte de référence général et les différents onglets latéraux de la carte présentant sa légende, les options et une explication.

Les cartes des écarts du mode standard d'HyperAtlas représentent pour chaque unité territoriale des valeurs relatives, sans dimension. Rappelons, en effet, que le calcul de cet écart relatif provient de la division entre le ratio de l'unité territoriale (PIB/population, par exemple) et le ratio du contexte de référence (PIB/population de l'Europe des 15, par exemple).

Le mode expert introduit un nouveau type de carte considérant un écart absolu : pour chaque unité territoriale, la carte de redistribution représente la quantité à ajouter au paramètre choisi au numérateur (le PIB par exemple) pour atteindre l'équirépartition parfaite. L'écart absolu obtenu peut être négatif (régions plus riches que le contexte de référence) ou positif (régions plus pauvres).

Les trois cartes d'équi-répartition indiquent donc quelle redistribution il s'agirait d'opérer afin d'atteindre la convergence des unités aux niveaux général, territorial, et local.

Selon l'expertise en sémiologie des géographes de RIATE, la carte *ad hoc* pour la représentation

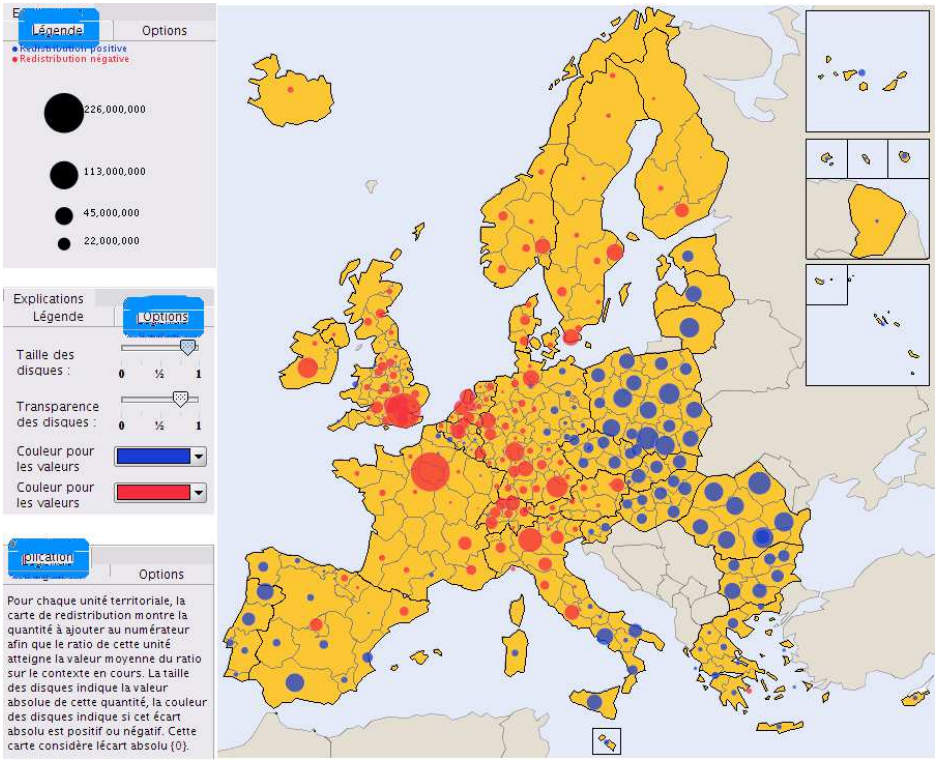


FIGURE 11.8 – Carte de redistribution appliquée au contexte de référence général considérant l'espace ESPON, au niveau de maillage NUTS 2, pour le PIB et la population totale en 2005. Les onglets latéraux pour la légende, ses options et les explicatinos sont juxtaposés.

de cette redistribution est du type à disques proportionnels colorés : la surface du disque indique la quantité de cet écart absolu, sa couleur indique son signe.

La conception de ces nouvelles cartes reprend l'architecture introduite par mes prédécesseurs [Plu07] :

- chaque carte est associée à un index correspondant à un onglet du panneau cartographique ;
- chaque onglet cartographique est composé d'un objet `Map` (le dessin de la carte), et d'un panneau latéral géré par un objet `SidePanel` composé de trois onglets :
  - la légende ;
  - les options de la légende ;
  - une description textuelle de la carte.

Pour la carte de redistribution, l' onglet « Options » permet à l'utilisateur de modifier le calibrage des disques, leur couleur et leur niveau de transparence. Par défaut, les disques sont rouges pour une valeur positive et bleus pour une valeur négative.

L'implémentation de ces cartes d'équi-répartition (une pour chaque contexte de référence : général, territorial, et spatial) spécialise les cartes à disques existantes dans l'application pour représenter les valeurs des indicateurs choisis au numérateur et dénominateur du ratio. Comme représenté sur la figure 11.9, les classes créées pour représenter les panneaux de légende et d'options de la légende spécialisent également les classes existantes conçues pour les cartes à disques.

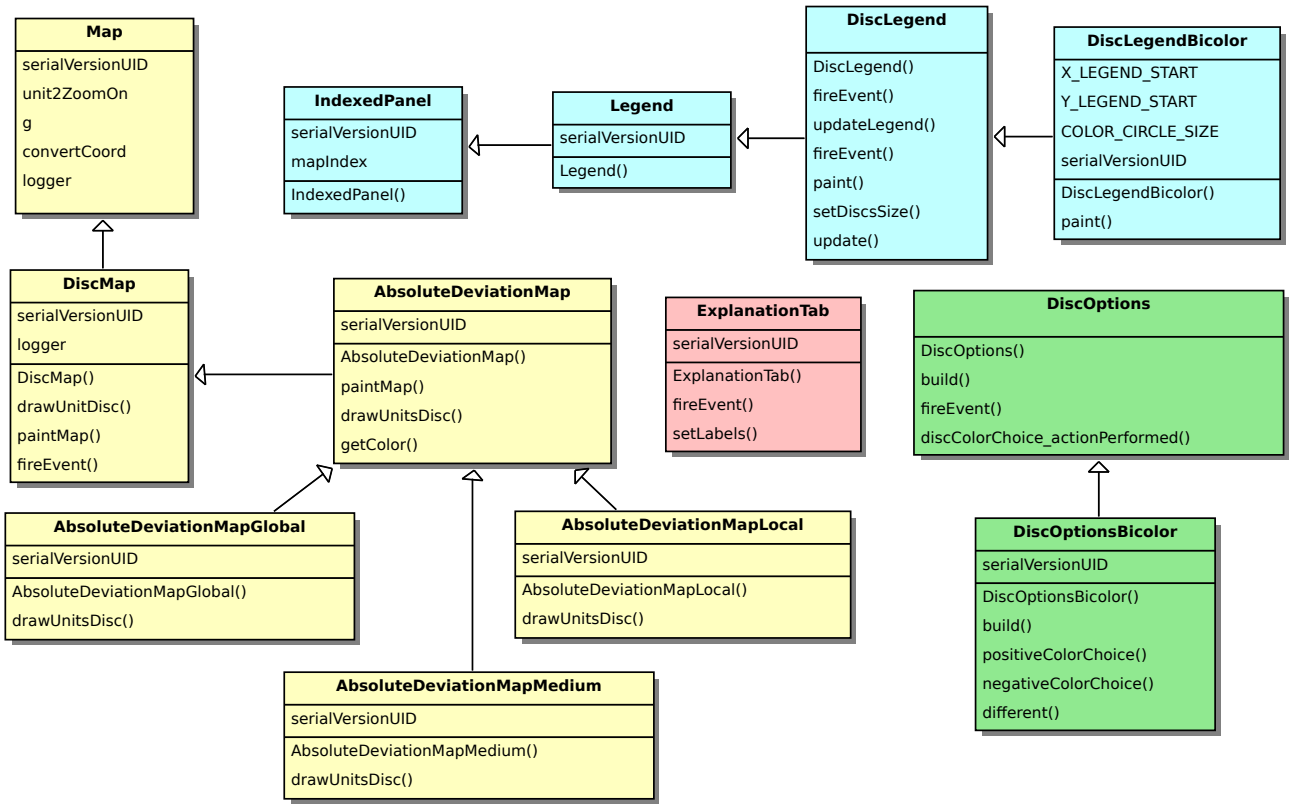


FIGURE 11.9 – Spécialisation et généralisation entre les principaux composants créés pour les cartes de redistribution.

11.3.3 La mesure des inégalités

Plusieurs indices statistiques permettent de mesurer les inégalités. Cette section décrit ceux qui ont été proposés dans la réponse du groupe de recherche HyperCarte au projet *ESPON HyperAtlas Update*.

Les cours en ligne de Claude Grasland [Gra97a] constituent la ressource documentaire majeure du contenu de cette section.

Les propos suivants considèrent une ressource quantifiable (PIB par exemple) et une population répartie sur un territoire. La mesure des inégalités cherche à représenter la concentration d'une ressource à l'aide d'un indice. Comme décrit ci-dessous, certains indices ont été définis de façon à être bornés par 0 et 1. Un indice à 0 correspond à la répartition la plus égalitaire, dite équirépartition : chaque personne de la population dispose de la même quantité de ressources. Un indice à 1 correspond à la situation extrême opposée : une seule personne détient 100 % de la ressource.

Dispersion relative

La dispersion statistique consiste à évaluer la tendance qu'ont les valeurs d'une distribution d'un caractère à s'étaler de part et d'autre d'une valeur centrale de référence, ou à s'éloigner les unes des autres. Si les paramètres de dispersion absolue sont mesurés dans l'unité de mesure du caractère (l'écart type, par exemple), les paramètres de dispersion relative renvoient un nombre sans unité. Ces derniers permettent notamment de comparer la dispersion de deux distributions qui ont des unités de mesure ou des ordres de grandeur différents.

Le coefficient de variation (CV) est une première solution pour déterminser la dispersion relative

d'une ressource. Le CV est défini comme le rapport entre l'écart type, noté σ, et la moyenne, notée $\bar{x}$. Le CV renvoie une valeur positive sans unité, il permet de comparer plusieurs ressources. A noter que si une valeur du CV de 0 correspond bien à l'équirépartition, sa valeur supérieure n'est pas bornée.

Pour rappel, l'écart-type est la racine carrée de la variance, la variance est définie comme la moyenne du carré des écarts à la moyenne. Une formule du calcul de l'écart type est présenté dans l'équation 11.2 en fonction de la moyenne, dont la formule est rappelée dans l'équation 11.1.

Formule du calcul la moyenne d'une ressource x dans une population composée de n individus :

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{11.1}$$

Formule du calcul de l'écart type de la ressource x en fonction de sa moyenne sur les n individus :

$$\sigma_x = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{11.2}$$

Expression du cofficient de variation en fonction de l'écart type et de la moyenne :

$$CV = \frac{\sigma_x}{\bar{x}} \tag{11.3}$$

Indice d'équirépartition de Hoover

L'indice d'équirépartition de Hoover est aussi nommé indice de disparité. La formule du calcul de l'indice de Hoover est donnée par l'équation 11.4 (page Wikipedia en anglais).

Cet indice a l'avantage d'être borné par 0 et 1. Il indique la quantité minimale de ressource à transférer d'un individu à un autre pour obtenir l'équirépartition. Par exemple, un indice de Hoover de 0.155 dénote qu'au moins 15.5 % du PIB devrait être transféré des régions les plus riches vers les régions les plus pauvres, alternativement, qu'au minimum 15.5 % de la population la plus pauvre devrait migrer vers des régions riches afin d'atteindre une équirépartition parfaite.

L'inconvénient de l'indice de Hoover est qu'il ne permet pas de distinguer si la population est composée d'un individu très riche et de beaucoup d'individus pauvres, ou si la population est composée de deux groupes à revenus différents.

L'équation 11.4 donne la formule du calcul de l'indice de Hoover en fonction des variables suivantes :

n : nombre de quantiles dans la population à étudier.

E_i : quantité de ressource détenue par le quantile i.

E_{total} : quantité totale de ressources détenue par l'ensemble de la population.

A_i : effectif d'individus dans le quantile i.

A_{total} : effectif total de la population.

$$H = \frac{1}{2} \sum_{i=1}^{n} \left| \frac{E_i}{E_{total}} - \frac{A_i}{A_{total}} \right| \tag{11.4}$$

La valeur des écarts absolus introduits dans les cartes de redistribution (section 11.3.2 page 92) est en étroite relation avec l'indice de Hoover. En effet, le calcul de l'écart absolu EA pour une unité territoriale i par rapport à une référence ref est donné par la formule 11.5 :

$$EA_{i,ref} = \frac{V_i}{V_{ref}} \cdot \frac{(Z_i - Z_{ref})}{Z_{ref}} \tag{11.5}$$

où Z_i et Z_{ref} représentent respectivement les ratios de l'unité i et du contexte de référence, et V le numérateur ou le dénominateur du ratio d'indicateurs considérés. Considérons pour la suite le cas du dénominateur, $\frac{V_i}{V_{ref}}$ devient $\frac{den_i}{\sum den_{ref}}$.

Les ratios de i et du contexte de référence peuvent s'exprimer en fonction des indicateurs choisis au numérateur et dénominateurs :

$Z_i = \frac{num_i}{den_i}$ et $Z_{ref} = \frac{\sum num_{ref}}{\sum den_{ref}}$.

En remplaçant ces valeurs dans l'équation 11.5 et en simplifiant on obtient :

$$EA_{i,ref} = \frac{num_i}{\sum num_{ref}} - \frac{den_i}{\sum den_{ref}} \tag{11.6}$$

En sommant les écarts absolus EA sur l'ensemble des unités i de l'aire d'étude, puis en divisant par deux, on retrouve l'expression de la formule de Hoover introduite dans l'équation 11.4.

Courbe de Lorenz et indice de Gini

La courbe de Lorenz et l'indice de concentration de Gini peuvent compléter l'analyse des inégalités en permettant de répondre à des questions du type « *quel est le pourcentage de ressources détenu par un certain pourcentage des individus les plus pauvres ?* ».

La courbe de Lorenz est la représentation graphique de la fonction qui à la part x de la population la moins riche associe la part y de la quantité de ressource dont elle dispose. Pour construire la courbe de Lorenz, on répartit généralement la population par déciles. On gradue l'axe des abscisses de 0 à 100 % pour représenter la part cumulée de la population, on gradue l'axe des ordonnées de 0 à 100 % pour représenter la part cumulée de la ressource (figure 11.10). La diagonale représente l'équirépartition parfaite, le cas où l'ensemble de la population disposerait de la même quantité de ressources.

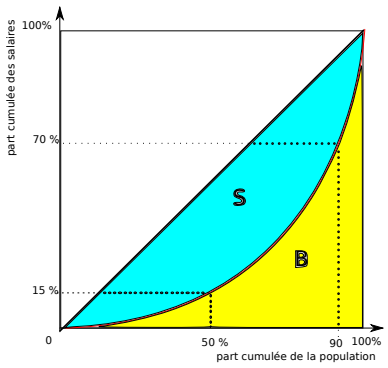


FIGURE 11.10 – Exemple de lecture de la courbe de Lorenz : 50 % des plus pauvres ne disposent que de 15 % du revenu total, pendant que 10 % des plus riches disposent de 70 % des revenus.

L'indice de Gini est une synthèse de la courbe de Lorenz, il fournit une valeur mesurant la globalité des disparités.

L'indice de Gini se déduit à partir de la courbe de Lorenz en considérant le double de la surface S entre la diagonale d'équirépartition et la courbe de Lorenz. Sur la figure 11.10, l'indice de Gini $G = 2 \cdot S$, ou encore, puisque $S + B = \frac{1}{2}$, $G = \frac{S}{S+B}$. On montre ainsi aisément que l'indice de Gini

est borné entre 0 et 1. Un indice de Gini à 0 correspond à l'équirépartition parfaite, la courbe de Lorenz est confondue avec la diagonale. Dans le cas d'un indice de Gini à 1, un individu dispose de 100 % des ressources : la courbe de Lorenz est confondue avec l'axe des absicces, la surface B est nulle.

En toute rigueur, l'indice de Gini G se calcule à partir de l'intégrale sur la courbe de Lorenz :

$$G = 1 - 2\int_0^1 L(x)dx$$

Néanmoins, la courbe de Lorenz sera généralement construite à partir d'une approximation linéaire sur chaque intervalle, à partir des points connus consécutifs. On pourra obtenir une approximation de l'indice de Gini en calculant l'aire par la méthode des trapézoïdes[2] et déterminer G par la formule 11.7 :

$$G = 1 - \sum_{i=1}^{n}(X_i - X_{i-1}) \cdot (Y_i + Y_{i-1}) \tag{11.7}$$

C'est cette formule qui sera utilisée pour l'implémentation du calcul de l'indice de Gini dans l'application.

Notons que l'indice de Hoover propose une valeur synthèse de la courbe de Lorenz : graphiquement, l'indice de Hoover correspond à la distance verticale maximale entre la courbe de Lorenz et la diagonale d'équirépartition, autrement dit le pourcentage de ressources qu'il s'agirait de transférer des plus riches aux plus pauvres pour obtenir une parfaite équirépartition.

Implémentation du calcul des indices

Afin d'en faciliter la réutilisation, l'ensemble des outils statistiques introduits pour le mode expert font l'objet d'un module indépendant d'HyperAtlas, isolé dans un paquetage nommé `org.steamer.stat`. On peut en effet imaginer que le calcul des indices statistiques soit nécessaire pour une autre application de l'équipe. La création d'une API sous la forme d'une archive `jar` constituée de ce paquetage est ainsi triviale.

Les données d'entrée pour le calcul des différents indices sont stockées dans un JavaBean* implémentant une interface `QuantilResourceInterface`. Deux méthodes accesseurs récupèrent un effectif (la population) et une ressource (le PIB) pour fournir les variables de la formule 11.4 de l'indice de Hoover : les valeurs retournées par `getEffectif()` et `getResource()` fournissent respectivement les valeurs des paramètres E_i et A_i de la formule.

La classe `QuantilResource` a été créée pour tester les calculs indépendamment de HyperAtlas. Depuis HyperAtlas, on dispose d'une classe identique permettant de stocker les valeurs des indicateurs choisis respectivement pour le numérateur et le dénominateur du ratio à considérer. L'accès aux calculs statistiques depuis HyperAtlas s'opère en instanciant une implémentation de la classe `StatDataInterface`.

Le diagramme de classes de la figure 11.11 illustre les différentes classes et interfaces qui ont été créées dans le paquetage `stat` afin de répondre chacune à un calcul particulier. Les constructeurs des classes prennent en paramètre de leur constructeur une liste d'objets `QuantilResourceInterface`. Pour plus de lisibilité, les classes de tests JUnit (une par classe de calcul) ont été omises sur ces schémas.

2. Page Wikipedia en anglais sur l'indice de Gini : http://en.wikipedia.org/wiki/Gini_coefficient (consulté le 28 mai 2010).

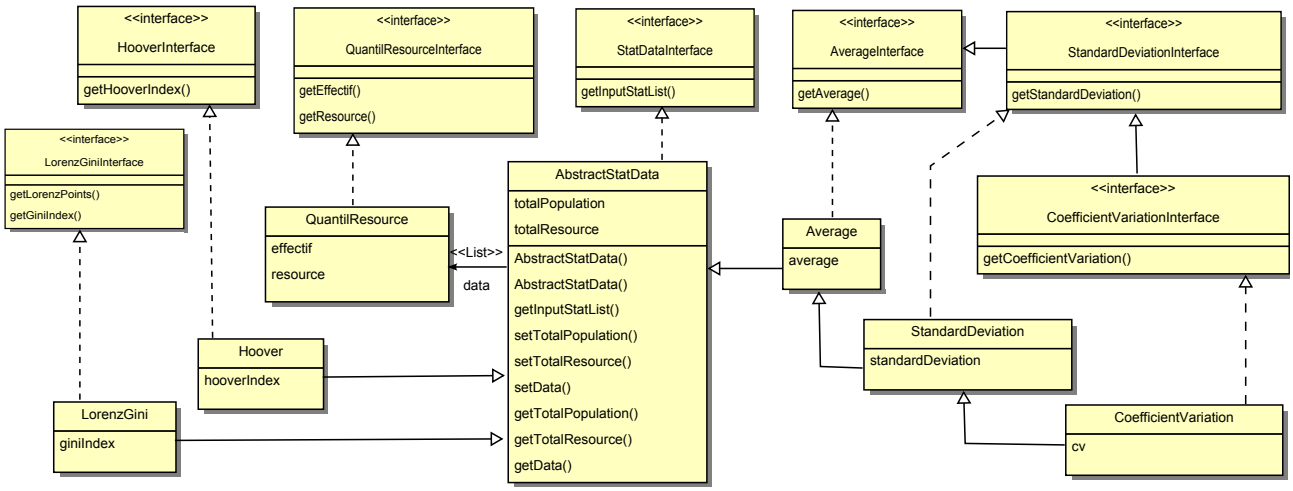


FIGURE 11.11 – Diagramme des classes intervenant dans le calcul des indices statistiques.

La figure 11.12 illustre le rendu final de la boîte statistique créée dans l'onglet cartographique « écart général », où trois sous-panneaux redimensionnables à souhait présentent à l'utilisateur :

- un tableau avec les valeurs des indices de Hoover, Gini, de l'écart type et du coefficient de variation ;
- la courbe de Lorenz (à droite) ;
- un panneau composé d'onglets fournit une description (textes et images) pour chacun des indices.

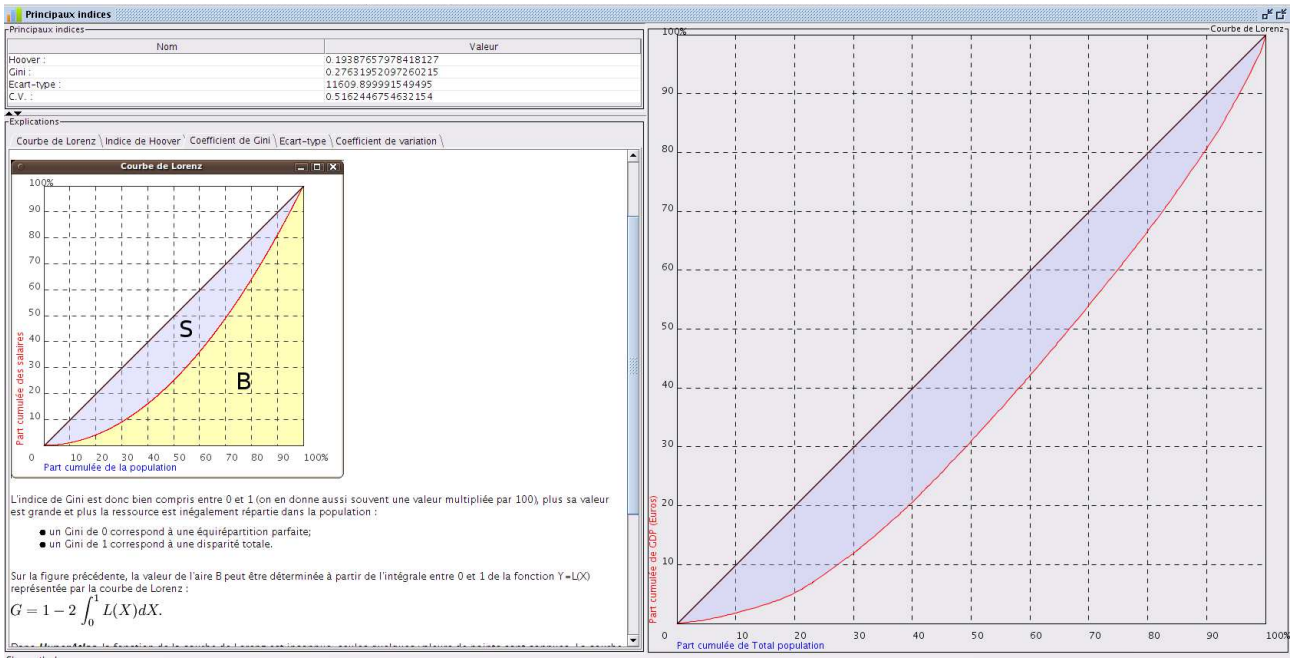


FIGURE 11.12 – La boîte statistique de l'onglet « écart général » affiche un tableau d'indices en haut à gauche, la courbe de Lorenz à droite et des explications en bas à gauche.

11.3.4 Autocorrélation spatiale

La courbe de Lorenz et le calcul des principaux indices statistiques sont mis à disposition de l'utilisateur dans la boîte statistique introduite dans l'onglet « écart général » en mode expert. Pour la boîte statistique de l'onglet « écart spatial », le groupe de recherche HyperCarte propose un graphique représentant l'autocorrélation spatiale entre l'écart spatial et l'écart territorial.

L'autocorrélation spatiale est définie comme la corrélation positive ou négative d'une variable avec elle-même provenant de la disposition géographique des données [LG00]. Elle illustre en quelque sorte la première loi de la géographie introduite par Waldo Tobler en 1970 selon laquelle « *everything is related to everything else, but near things are more related than distant things* ».

Considérant l'observation d'une variable, une autocorrélation spatiale positive traduit que des lieux proches se rassemblent davantage que des lieux éloignés. Une autocorrélation spatiale négative indique que des lieux proches sont plus différents que des lieux éloignés. Une absence d'autocorrélation indique que la répartition spatiale des observations est aléatoire. Ces trois cas sont schématisés sur la figure 11.13 [MSJ06].

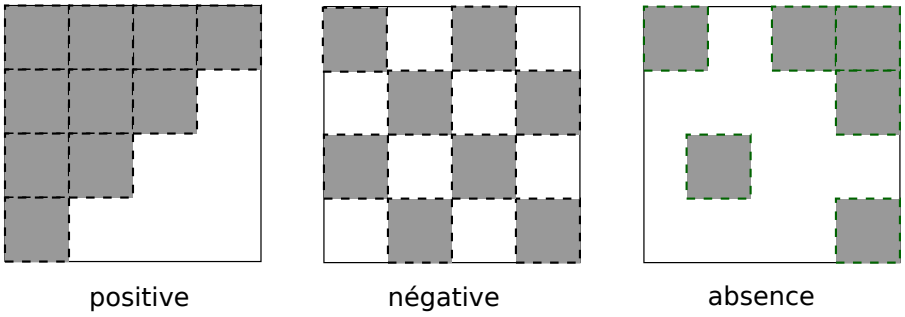


FIGURE 11.13 – Autocorrélations spatiales positives, négatives et absence d'autocorrélation.

Dans le cadre d'HyperAtlas en mode expert, l'équipe RIATE propose un nouveau graphique (figure 11.14) permettant à l'utilisateur de visualiser l'autocorrélation spatiale en prenant en compte l'écart spatial et l'écart hiérarchique territorial pour chaque unité de l'espace d'étude. Par la méthode des moindres carrés, le graphique illustre également la droite de régression caractérisant l'autocorrélation spatiale pour l'ensemble des unités territoriales.

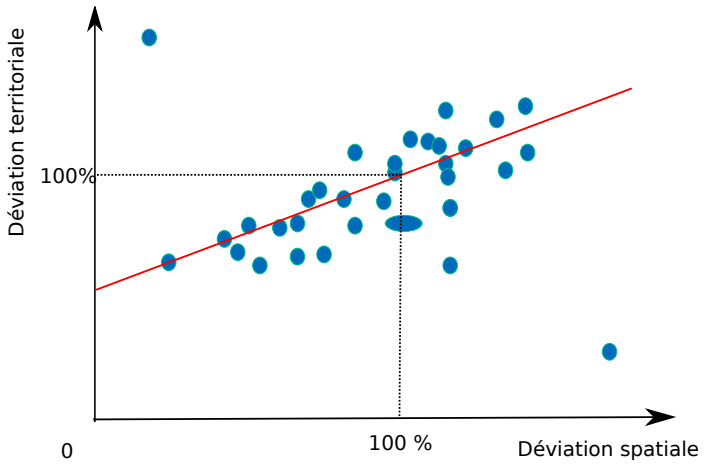


FIGURE 11.14 – La représentation d'autocorrélation spatiale proposée par RIATE.

Pour calculer l'équation de la droite de régression caractérisant au mieux le nuage de points, HyperAtlas s'appuie sur la méthodes des moindres carrés : l'objectif est de déterminer les coefficients a et b de l'équation $y = ax + b$ dite équation de la droite des moindres carrés [Ver07]. On considère l'hypothèse selon laquelle chaque individu i du nuage de points est tel que $y_i = ax_i + b + e_i$ avec e_i une certaine erreur appelée le résidu. On cherche à minimiser la somme quadratique des residus notée S pour tous les individus :

$S = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - ax_i - b)^2$

S est minimal si les dérivées partielles par rapport à a et b sont nulles. Dérivons partiellement S par a :

$\frac{\partial S}{\partial a} = \sum_{i=1}^{n} -2x_i(y_i - ax_i - b)$

Dérivons maintenant partiellement S par b :

$\frac{\partial S}{\partial b} = -2\sum_{i=1}^{n} (y_i - ax_i - b)$

On obtient le système de deux équations à deux inconnues a et b suivant :

$$\begin{cases} -2\sum_{i=1}^{n} x_i(y_i - ax_i - b) &= 0 \quad (1) \\ -2\sum_{i=1}^{n} (y_i - ax_i - b) &= 0 \quad (2) \end{cases}$$

On peut isoler b grâce à l'équation (2) :

$2nb = 2\sum_{i=1}^{n} y_i - 2a\sum_{i=1}^{n} x_i$

En considérant les moyennes $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ et $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$, on en déduit :

$$b = \bar{y} - a\bar{x} \tag{11.8}$$

Remplaçons cette valeur de b dans le système d'équations :

$$\begin{cases} -2\sum_{i=1}^{n} x_i((y_i - \bar{y}) - a(x_i - \bar{x})) &= 0 \quad (3) \\ -2\sum_{i=1}^{n} y_i - \bar{y} - a(x_i - \bar{x}) &= 0 \quad (4) \end{cases}$$

En soustrayant l'équation (3) et l'équation (4) préalablement multipliée par $\bar{x}$, on obtient :

$\sum_{i=1}^{n} (x_i - \bar{x})((y_i - \bar{y}) - a((x_i - \bar{x})) = 0$

Si la somme des écarts de x à la moyenne $\bar{x}$ n'est pas nulle on peut isoler a :

$$a = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{11.9}$$

Le système est résolu, les équations 11.9 et 11.8 nous fournissent les valeurs de a et b. On peut

néanmoins exprimer plus synthétiquement la valeur de a. La covariance empirique de x et y notée $\sigma(x, y)$ est par définition :

$$\sigma(x, y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

La définition de la variance empirique de x notée $\sigma^2(x)$ (carré de l'écart type) donne :

$$\sigma^2(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Le coefficient a de l'équation de la droite des moindres carrés s'exprime ainsi plus synthétiquement en fonction de la covariance empirique de x et y et de la variance de x :

$$a = \frac{\sigma(x, y)}{\sigma^2(x)} \tag{11.10}$$

Implémentation

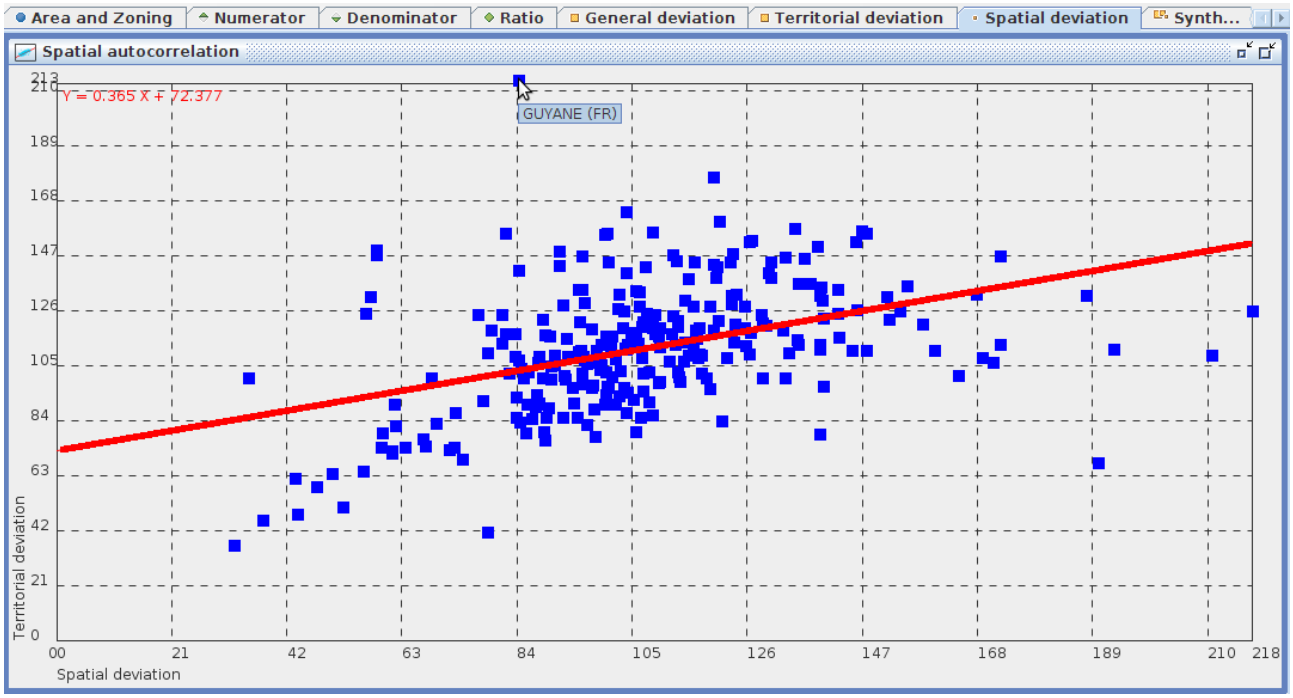


FIGURE 11.15 – Exemple du diagramme d'autocorrélation spatiale intégré dans HyperAtlas appliqué au PIB et à la population en 2005 pour les unités au niveau NUTS 2 de la zone d'étude EU27.

Le calcul des coordonnées des points représentant les unités territoriales dans le repère d'autocorrélation spatiale attendu réutilisent les classes du paquetage `stat`. Pour la vue, la boîte statistique affichant le graphique d'autocorrélation est trivialement implémentée en spécialisant le composant `CoordSysPanel` dont il a déjà été question section 11.2.1 page 90 : il s'agit en effet de fournir au constructeur père (le repère) les maxima de graduations et les légendes à afficher le long des axes. La spécialisation du diagramme d'autocorrélation sur le repère du composant `CoordSysPanel` consiste :

- à ajouter une instance d'objet graphique, un carré bleu, pour chaque unité : connaissant les coordonnées représentant les valeurs des écarts d'une unité territoriale, la classe `CoordSysPanel`

traduit ces coordonnées dans le système de coordonnées du panneau Java (`JPanel`), l'API Java `Graphics2D` fait le reste ;
- à dessiner la droite de régression en rouge : là encore, `Graphics2D` fournit la méthode nécessaire pour dessiner une droite connaissant deux points, et lui affecter une épaisseur, une couleur, etc. ;
- à ajouter un écouteur d'évènement sur les mouvements de souris pour détecter le passage de celle-ci au dessus d'une unité : un message contextuel indique alors le nom de l'unité survolée.

Notons qu'afin d'optimiser l'affichage des messages contextuels, une table est créée lors de l'initialisation du composant pour associer à chaque instance d'objet graphique carré le nom de l'unité correspondante.

La figure 11.15 illustre le résultat intégré dans l'application HyperAtlas : la boîte statistique « autocorrélation spatiale » incorporée dans l'onglet « Ecart spatial » en mode expert est ici maximisée. Le repérage des valeurs exceptionnelles est aisé pour les unités qui se détachent du nuage de points (ici, la Guyane).

11.4 Intégration du temps

La gestion du temps dans HyperAtlas v2 consiste à donner à l'utilisateur la possibilité de naviguer aisément parmi les dates disponibles d'un même indicateur ou d'un ratio prédéfini, par exemple le PIB par habitant en 2000, en 2005, etc.

Notons que la gestion du temps ne tient compte, ni de l'évolution de la géométrie des territoires, ni de l'évolution des métadonnées attachées aux indicateurs.

Au niveau interface utilisateur, des maquettes de curseurs temporels double entrée (numérateur/dénominateur) ont été étudiées. La barre de zoom spatial, en bas des fenêtres de la version 1, a d'ailleurs été déplacée pour laisser place à un tel composant. Le curseur temporel n'a cependant pas été retenu pour cette version 2 : les indicateurs des jeux de données courants ne proposent, en effet, des valeurs que pour une, ou deux dates. Principalement pour des raisons ergonomiques, l'option du groupe de recherche HyperCarte a donc été de reprendre et perfectionner une évolution déjà partiellement intégrée dans la version 1.2.8.

Initiée lors d'une distribution pour l'EEA en 2008, la prise en compte de l'aspect temporel dans HyperAtlas 1.2.8 consiste à proposer à l'utilisateur un sous-menu, à partir de la boîte de sélection du ratio, dans le panneau de paramétrage. Pour un libellé de ratio donné, le sous-menu liste les dates pour lesquelles des valeurs ont été recensées. En fonction du nom du jeu de données, HyperAtlas 1.2.8 affiche des boîtes de sélection classiques, et dans le cas du jeu de données `eea.hyp`, affiche une boîte de sélection disposant du sous-menu dates.

Dans cette version 1.2.8, la récupération des dates est codée en dur, en analysant le code identifiant du ratio, et en respectant une convention de nommage lors de création du jeu de données : par exemple, les identifiants de ratios `r1_00` et `r1_05` représentent le ratio `r1`, pour respectivement 2000 et 2005.

Au niveau représentation, la version 2 d'HyperAtlas reprend le principe du sous-menu de dates. Le composant graphique développé initiallement, `HCComboMenuBar`, permettant d'incorporer un sous-menu en spécialisant le composant Java `JComboBox` (boîte de sélection de l'API Java Swing), a donc été repris et amélioré. Il supporte désormais l'internationalisation, et un menu contextuel apparaît au survol de la souris, pour afficher la description de l'indicateur courant. Il permet également d'afficher un intervalle de dates, ou une date discrète (l'année).

Ce composant a ensuite été généralisé aux paramètres numérateur, dénominateur, et ratio,

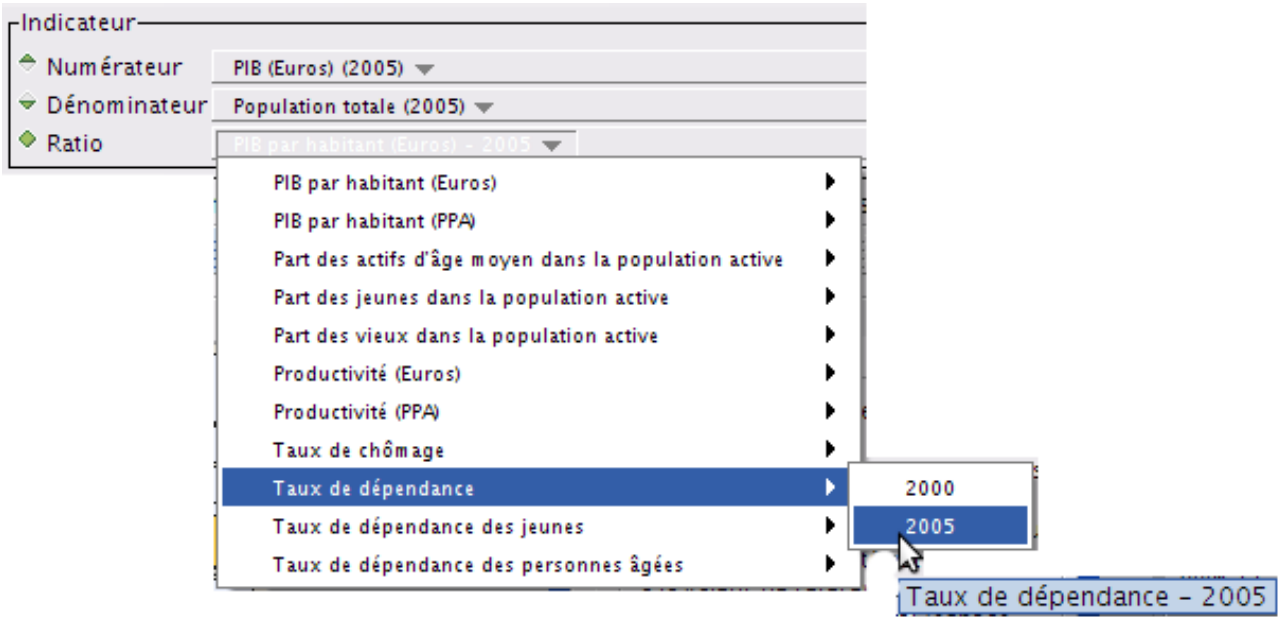


FIGURE 11.16 – Sous-menu temporel des indicateurs.

comme illustré sur la figure 11.16.

Pour la prise en compte du temps, l'innovation sur l'existant a essentiellement consisté à modifier les jeux de données générés par HyperAdmin. HyperAtlas peut maintenant exploiter l'éventuelle information temporelle des indicateurs intégrée dans le `.hyp`, sans conventions particulières de nommage.

Ainsi, au chargement d'un jeu de données, HyperAtlas tente d'en récupérer les métadonnées : quelle est la version de ce `.hyp` ? Cette information n'est pas disponible dans les jeux de données construit avant la version 2. Si le fichier `hyp` informe sur sa version, il informe aussi sur la prise en compte du temps (voir section 12.3.1 page 113). En fonction de la valeur du booléen `timeEnabled`, attribut des métadonnées du `.hyp`, HyperAtlas affiche dans le panneau des paramètres, tantôt des boîtes de sélection classiques (si `timeEnabled=false`), tantôt des boîtes de sélection munies de sous-menus dates (des composants `HCComboMenuBar`).

Dans le cas où le temps est activé, le premier niveau des boîtes de sélection propose la liste des indicateurs (ou ratio) dont les labels sont identiques.

Quant à l'attribut temporel, il est désormais incorporé dans la description associée à l'indicateur, implémentée par un composant JavaBean*, `SerialDescription`. Enrichi de deux nouveaux champs `startDate` et `endDate`, la description de l'indicateur permet à HyperAtlas de récupérer la date pour laquelle cet indicateur est défini (ou l'intervalle de dates). Ces deux attributs sont analysés pour remplir en conséquence le sous-menu dates du libellé de l'indicateur.

En conclusion, l'intégration du temps dans HyperAtlas a essentiellement consisté à profiter de l'existant. La représentation a repris une idée simple, introduite dans une version précédente du logiciel, et améliorée pour cette version 2. Le modèle de données des fichiers `hyp` a été modifié pour intégrer de façon générique l'attribut temporel des indicateurs. L'ajout de métadonnées sur le fichier `hyp` permet une compatibilité ascendante, descendante, et générique, du logiciel vis-à-vis des jeux de données chargés.

11.5 Définition d'une nouvelle aire d'étude

Le cahier des charges du projet *ESPON HyperAtlas Update* spécifie la possibilité donnée à l'utilisateur d'HyperAtlas de définir des nouvelles aires d'étude. Considérant la difficulté d'implémenter une telle requête de façon générale, le groupe de recherche HyperCarte décide de répondre à cette requête en proposant à l'utilisateur cette possibilité par la sélection d'unités territoriales au plus haut niveau de la hiérarchie des maillages seulement.

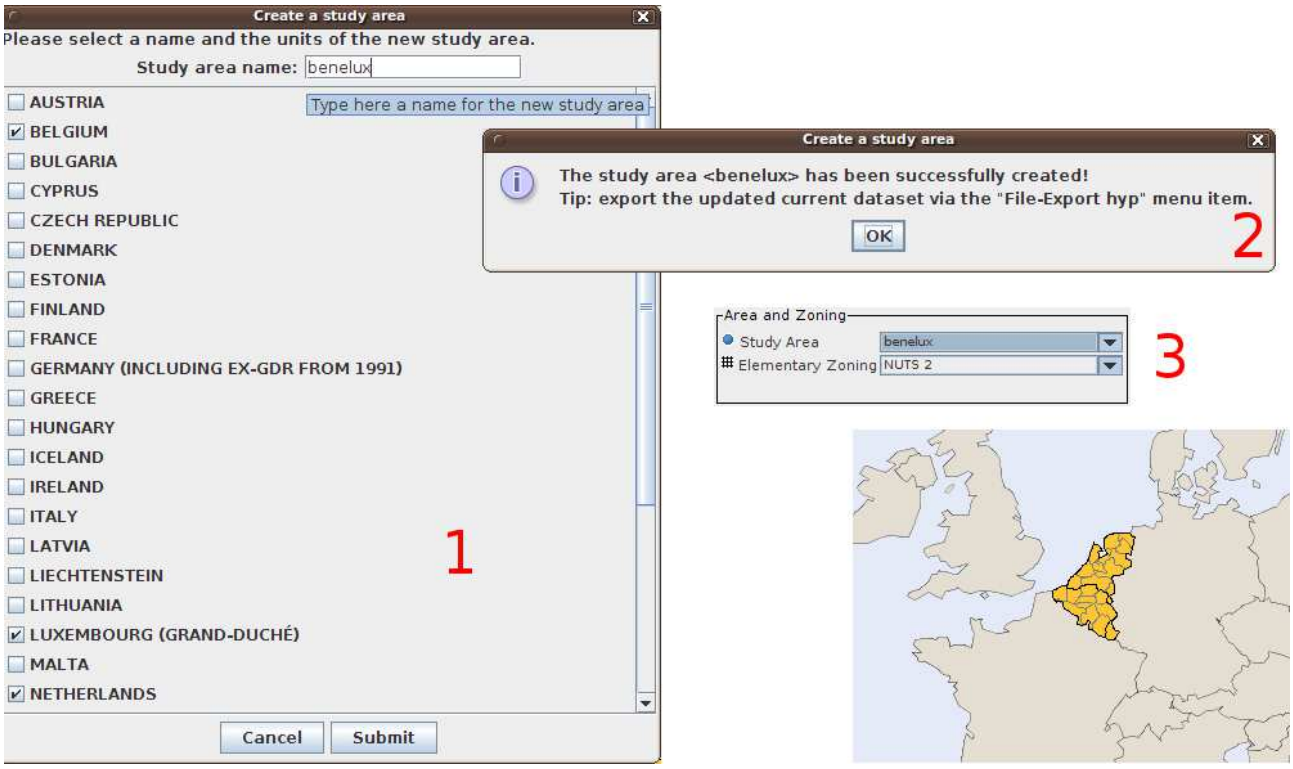


FIGURE 11.17 – Définition d'une nouvelle aire d'étude : 1 - formulaire : choix du nom (benelux) et des unités (Belgique, Luxembourg, Hollande) ; 2 - message de confirmation ; 3 - sélection de la nouvelle aire d'étude benelux dans le panneau de paramètres, visualisée sur la carte de l'onglet « Aire d'étude ».

La possibilité de sélectionner des unités sur différents niveaux de la hiérarchie était d'ores et déjà spécifiquement exclue du cahier des charges. La sélection d'unités à n'importe quel niveau de maillage a également été exclue pour s'affranchir des difficultés liées à l'éventuelle impossibilité d'agrégation. Le panneau de paramètres d'HyperAtlas est en effet conçu de telle façon que le choix d'une aire d'étude ne met pas à jour dynamiquement les niveaux de maillages disponibles pour ce choix. Autrement dit, les jeux de données sont conçus de façon à ce que l'utilisateur puisse choisir n'importe lequel des niveaux de maillage élémentaires disponibles (par exemple les niveaux NUTS 0, NUTS 1 et NUTS 2) pour tout choix d'aire d'étude.

La généralisation de la possibilité de définition d'une aire d'étude à partir d'unités sur tout niveau de la hiérarchie de maillages est une requête demandant une revue importante de l'application. Pour palier cette difficulté dans le temps imparti tout en offrant une réponse au cahier des charges, la définition d'une nouvelle aire d'étude n'est donc actuellement possible qu'à partir d'unités au plus haut niveau de maillage (les pays, c'est-à-dire le niveau NUTS 0 dans le cas des jeux de données basés sur la délinéation NUTS).

La définition d'une nouvelle aire d'étude s'effectue à travers un nouvel élément de la barre de menus, « Créer une aire d'étude ». Le clic de cet élément ouvre une fenêtre modale invitant l'utilisateur à donner un nom à sa nouvelle aire d'étude (la couche modèle vérifie que ce nom n'est ni vide ni déjà attribué à une aire d'étude existante dans le jeu données), puis à sélectionner les unités qui vont composer la nouvelle aire. La validation de ce formulaire associe récursivement toutes les unités filles des unités choisies à cette nouvelle aire d'étude, le jeu de données chargé en mémoire est modifié en conséquence puis la liste de sélection « aire d'étude » du panneau de paramètres est mise à jour avec ce nouvel élément. Les étapes sont résumées sur la figure 11.17.

11.6 Zoom et déplacement de la carte

La version 1 de HyperAtlas proposait un curseur en bas de la fenêtre pour agir sur le facteur de zoom spatial et le grossissement de la carte. A noter que l'utilisateur a également la possibilité de zoomer à l'aide de la molette de souris ou en cliquant sur deux éléments « Zoom + » et « Zoom - » depuis la barre de menu.

La visibilité du curseur de zoom suggère à l'utilisateur la disponibilité du service qui lui est offert. De même pour le déplacement de la carte sur l'écran, le « glisser-déplacer » à l'aide de la souris est ergonomiquement plus aisé que des clics successifs sur un bouton, mais la présence de ces boutons au dessus de la carte rappelle à l'utilisateur cette possibilité d'action et lui permet de se concentrer sur son analyse plutôt que sur les moyens pour y parvenir.

En outre, chacun a pu aujourd'hui utiliser un jour Google Maps, ou un autre outil cartographique sur le Web : l'habitude générale des utilisateurs est donc de retrouver un curseur de zoom et quatre boutons pour déplacer la carte (*Pan*) en haut à gauche de toute carte, en général.

Pour ces raisons mais également pour augmenter la taille des cartes, le curseur de zoom spatial a donc été déplacé sur un calque au dessus de la carte, à la position habituelle des autres outils cartographiques. Quatre composants graphiques particuliers (héritant des labels Java `JLabel` et écoutant les évènements de clics de souris) ont également été ajoutés pour déplacer la carte vers le haut, le bas, la gauche et la droite de la fenêtre. L'échelle a aussi été déplacée sur ce calque, son code a été considérablement allégé au passage.

11.7 Habillage des cartes

Les étapes intervenant dans le dessin et l'habillage des cartes ont été revues et adaptées pour tenir compte de la géométrie particulière de certaines régions à afficher, mais également pour quelques considérations politiques relatives aux nouveaux jeux de données livrés à ESPON. Cette section décrit tout d'abord les étapes par lesquelles passe HyperAtlas pour dessiner des cartes, puis les évolutions proposées et implémentées dans la version 2 d'HyperAtlas. On considère ici l'exemple de la construction de la carte des jeux de données sur l'Europe en prenant les pays candidats (la Turquie, par exemple) comme choix de l'aire d'étude.

Quelque soit la carte à dessiner (ratio, écarts, disques, etc.), le processus passe par deux grandes étapes :

1. le dessin des unités hors de l'aire d'étude choisie en paramètre :
   (a) un fond bleu pouvant être interprété comme les océans ;
   (b) les unités hors aire d'étude sont peintes en beige : contour de la Russie, d'Afrique du

Nord, les îles d'outre-mer (Antilles françaises, Guyane, Canaries, Madeire) sont représentées au dessus de la Russie ;

(c) les cartouches carrés à fond bleu écrasent les îles précédemment dessinées au dessus de la Russie ;

(d) les unités spéciales d'Amérique du Sud, le Surinam et le Brésil entourant la Guyane ;

(e) les îles d'outre-mer, effacées par la couche des cartouches ;

(f) le contour des cartouches est repeint ;

2. les unités de l'aire d'étude choisie en paramètre sont ensuite dessinées, avec un fond jaune, par défaut, pour la carte de l'aire d'étude.

La classe abstraite `Map` factorise le code permettant de dessiner les cartes hors aire d'étude. Les classes `DiscMap` et `DeviationMap` spécialisent `Map` pour représenter les différents types de cartes (à disques, choroplèthes). Pour la carte de l'aire d'étude, la classe `StudyAreaMap` dessine les polygones des unités de l'aire d'étude.

Le principal inconvénient des étapes précédentes provient du fait que le dessin des unités particulières, Surinam, Brésil, pays comportant des îles d'outre-mer, et cartouches carrés, est réalisé en filtrant ces unités en fonction de leur identifiant. Nous cherchons donc un moyen de ne plus coder en dur dans HyperAtlas le filtre sur l'identifiant des unités pour leur affecter un habillage particulier. Ce filtre, d'une part, requiert une convention de nommage lors de la réalisation du fichier structure en entrée d'HyperAdmin, et, d'autre part, freine l'application, surtout pour des jeux de données comme Rhône-Alpes, par exemple, où les îles d'outre-mer sont hors propos.

En outre, pour les nouveaux jeux de données fournis à ESPON, est émis le souhait de représenter Chypre Nord en blanc : cette demande provient d'un désaccord sur le rattachement de ce territoire à la Turquie, décision non reconnue par les pays membres de la Communauté Européenne. Ce litige politique nécessite la prise en compte d'une nouvelle exception dans l'habillage des cartes.

Deux options ont été proposées lors d'une réunion du groupe de recherche HyperCarte pour intégrer ce genre d'exception :

- utiliser le fichier MID en entrée d'HyperAdmin pour stocker des styles sur les unités particulières (en séparant par une tabulation l'identifiant des unités et un code de style) ;
- modifier le fichier tableur `structure.xls` en entrée d'HyperAdmin pour spécifier la couleur des unités particulières.

La première option a été écartée afin de séparer le contenu purement géométrique fournie par le fichier MIF-MID de l'habillage à considérer. La proposition retenue consiste donc à enrichir de deux nouveaux onglets le contenu du tableur `structure.xls`, fourni par l'utilisateur en entrée d'HyperAdmin :

- un onglet définissant les associations entre les unités particulières et un style ;
- un onglet dédié à la définition des styles. Un style est défini par son identifiant, un nom et un ensemble de colonnes pouvant évoluer à l'avenir. Les champs retenus actuellement permettent de spécifier la couleur de fond du polygone, la couleur de sa frontière et l'épaisseur du trait de sa frontière. Les deux couleurs sont définies chacune par quatre champs afin de spécifier les composantes RVB (rouge vert bleu) et alpha (transparence) par des entiers entre 0 et 255.

Les deux onglets entretiennent une relation modélisée sur la figure 11.18.

HyperAdmin a donc un nouvel onglet à gérer en entrée. Il doit sauvegarder les styles et les associations dans le `.hyp` généré. Si ces nouveaux onglets n'existent pas dans la définition de la structure de données, aucun style particulier n'est à prendre en compte.

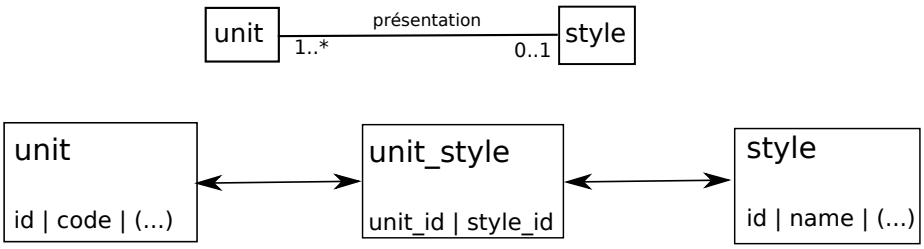


FIGURE 11.18 – Modèle relationnel entre les unités territoriales et les styles.

HyperAtlas, de son côté, doit assurer la compatibilité des anciens fichiers `.hyp`. On ne peut donc pas actuellement se passer des exceptions décrites précédemment pour l'habillage des unités particulières en fonction de leur identifiant. Aussi, pour déterminer la version du jeu de données chargé, HyperAtlas cherche à détecter l'existence d'un objet `SerialHypInfo` dans le fichier `hyp` désérialisé : si ces métadonnées n'existent pas, le jeu de données correspond à un ancien modèle. Sinon, un numéro de version permet à HyperAtlas de traiter en conséquence les nouvelles possibilités d'habillage pour les unités particulières.

Cette compatibilité descendante des jeux de données est actuellement l'option choisie. Une alternative consiste à supprimer définitivement le code en dur pour gérer les unités exceptionnelles, et de migrer les anciens fichiers `.hyp`, au moins ceux mis à disposition sur le site Internet du groupe de recherche HyperCarte, vers le format des jeux de données version 2. Cette alternative est écartée pour l'instant.

En conclusion, lors de la détection d'un jeu de données au format v2, HyperAtlas récupère les unités particulières et les affiche suivant le style défini par le géographe en entrée d'HyperAdmin. Le code en dur pour filtrer les unités des jeux de données v1 persiste mais a été isolé dans une classe `MapHack`. L'implémentation de cette fonctionnalité est en cours.

11.8 Gel des cartes et comparaison

L'objectif de cette évolution consiste à « geler » les onglets cartographiques, afin de pouvoir comparer, par exemple, différents indicateurs, ou, les mêmes indicateurs à différentes dates.

La mise en œuvre de cet objectif est encore expérimentale. Un nouvel élément de menu « Sélecteur de fenêtres », désactivé par défaut, a été introduit à cet effet. Le clic sur ce menu fait apparaître une barre d'outils dans laquelle sont listés les onglets cartographiques du panneau principal. Spécialisant le composant Java `JToolBar`, cette barre d'outils peut être déplacée sur les bords de la fenêtre principale, ou même extraite dans une fenêtre à part, comme sur la figure 11.19. Les différents éléments de la liste agissent comme des interrupteurs, ils permettent d'ouvrir ou de fermer la carte associée dans une nouvelle fenêtre externe. Sur la figure 11.19, l'utilisateur a ouvert les trois cartes d'écarts, il/elle consulte en même temps la carte de synthèse binaire sur la fenêtre principale.

Au moment de l'écriture de ce document, ce nouveau service est un prototype. Son instabilité provient du fait que les différentes cartes réagissent toutes simultanément aux évènements dits indexés : par exemple, la modification du facteur de zoom, ou le déplacement de la carte, sur un des onglets de la fenêtre principale, est répercuté sur toutes les autres cartes. La machine virtuelle Java a d'autant plus de mises à jour à réaliser que le nombre de cartes affichées est grand. Pour remédier à ce problème, l'équipe STEAMER suggère la possibilité d'intégrer le concept de modes

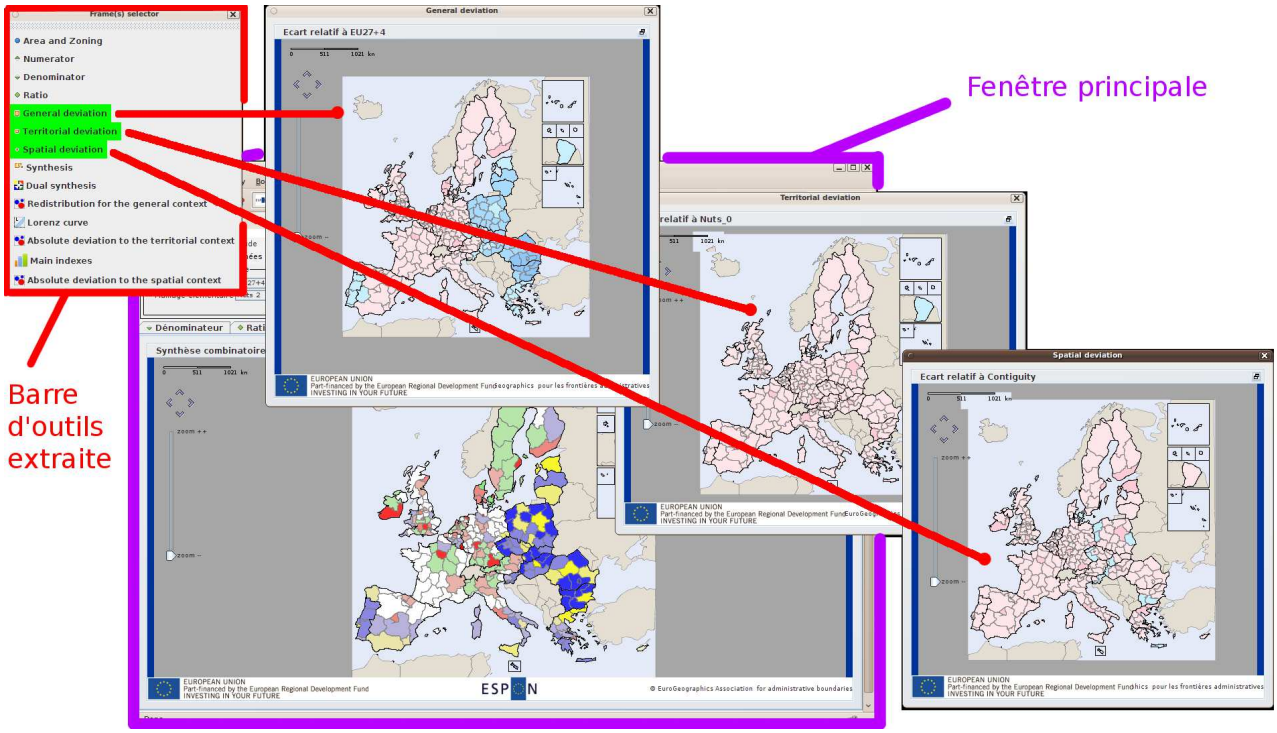


FIGURE 11.19 – Exemple d'utilisation du menu « Sélecteur de fenêtres » pour la comparaison simultanée de différentes cartes.

sur les fenêtres externes. Les différents modes régissent le comportement des fenêtres externes :

mode synchrone : les fenêtres externes réagissent toutes simultanément aux évènements indexés (zoom, déplacement) ;

mode asynchrone : le zoom et le déplacement sur une fenêtre externe ne sont pas répercutés sur les autres cartes ;

mode gelé : l'utilisateur n'a pas de possibilité de changer l'affichage de la carte représentée dans la fenêtre externe.

Proposée à ESPON lors du jalon pré-intermédiaire en août 2010, cette évolution suscite des intérêts à la fois techniques et ergonomiques. Estimant la complexité de son implémentation, ESPON a néanmoins considéré cette évolution comme secondaire pour cette version 2 du logiciel et du présent projet.

En attendant, l'implémentation de cette évolution prototype repose sur une modélisation innovant sur l'existant. Contrairement aux onglets cartographiques de la fenêtre principale, le sélecteur de fenêtres tente de s'affranchir au maximum des singletons de l'application, et de débrayer la couche vue de la couche contrôle via le recours à des interfaces. Si le modèle représenté figure 11.20 reste à enrichir, il initie le dégroupage des couches et des différents composants, notamment entre la barre d'outils et les fenêtres externes. Les différents composants sont physiquement répartis dans trois nouveaux paquetages, `model`, `view` et `control`, créés pour aider à la modélisation suivant le patron MVC.

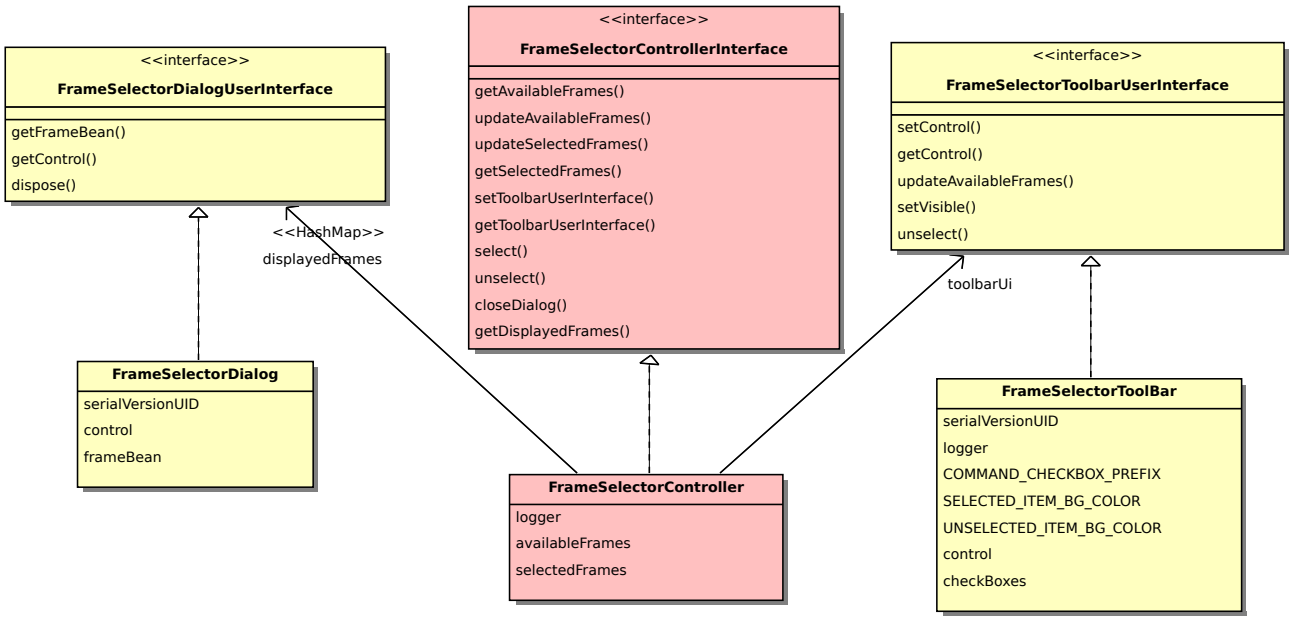


FIGURE 11.20 – Découplage via des interfaces entre les couches vue (en jaune) et contrôle (en rouge) des composants du « sélecteur de fenêtres ».

11.9 Synthèse d'HyperAtlas v2

Ce chapitre a essentiellement présenté l'intégration à l'existant de nouveaux outils d'analyse cartographique et statistique. Plutôt destiné aux statisticiens, un mode expert a été introduit dans l'interface utilisateur. Cette dualité d'exécution standard/expert permet de respecter une contrainte forte du logiciel, sa simplicité d'utilisation. L'utilisateur *lambda* n'est pas perturbé par ces nouveaux outils avancés. En activant le mode expert, le statisticien dispose, par contre, de ces nouveaux outils complémentaires pour réaliser des analyses plus complexes.

En évaluant dès le départ le temps nécessaire au développement des nouvelles fonctionnalités, la stratégie générale adoptée a été d'intégrer les nouveaux outils sur la logique applicative existante. En considérant que l'objectif de la livraison d'une première version a été atteint dans le temps imparti, in-extremis (30 décembre pour le 31), cette stratégie initiale peut sembler justifiée.

Si les neuf mois de stage ont permis d'appréhender globalement les différents points clés de la conception de l'application, ils n'ont cependant pas suffi pour à la fois intégrer les nouvelles fonctionnalités demandées dans le cahier des charges ESPON, et en même temps, de remédier aux faiblesses de la conception existante (singletons statiques).

Néanmoins, l'intégration modulaire des nouvelles fonctionnalités d'analyse, la dualité d'exécution application/*applet*, l'intégration générique du critère temporel, et surtout la démarche générale (tests, documentations, environnement de développement), ont permis d'améliorer significativement la qualité globale du logiciel.

Ainsi, 116 tests unitaires JUnit ont été écrits, ils sont lancés automatiquement par le script Ant lors de la construction des archives exécutables `HyperAtlas.jar`. Ces tests ont été développés en même temps que les nouvelles fonctionnalités. Seulement une quinzaine d'entre eux a été écrite pour vérifier des fonctionnalités qui existaient dans la version 1.2.9 du logiciel, les autres étant donc voués à tester les nouvelles méthodes intégrées dans la version 2. Le passage avec succès de tous ces tests est une condition *sine qua non* à la poursuite du processus de construction. Ils assurent la validité et la non-régression d'un ensemble important des fonctionnalités de l'application.

Avant la livraison finale d'*ESPON HyperAtlas* v2, prévue pour le 28 février 2011, des tests de performances sont en cours. Des corrections sont apportées sur les faiblesses de l'interface. L'optimisation des algorithmes devrait notamment permettre une plus grande réactivité, par exemple en mode expert. Pour information, l'activation du mode expert requiert actuellement entre cinq et vingt secondes, selon le niveau de maillage choisi. Cette durée est essentiellement consacrée aux calculs des points de la courbe de Lorenz et du coefficient de Gini. Le temps de calcul peut largement être amélioré.

Indiscutablement bénéfique quant à son évaluation, une enquête est actuellement en cours auprès d'utilisateurs externes au groupe de recherche HyperCarte. L'enquête porte sur les nouvelles fonctionnalités d'HyperAtlas v2 « *draft version* » (livraison intermédiaire du 31 décembre 2010). Prochainement diffusée en ligne par l'équipe RIATE, une vingtaine de partenaires du programme ESPON, parmi lesquels, statisticiens, chercheurs et politiques, sont ainsi invités à donner leur avis sur l'application. Les résultats du sondage, les remarques, seront alors pris en compte pour la suite des implémentations et des améliorations.

Les perspectives d'évolution d'HyperAtlas sont donc potentiellement multiples, tant au niveau des fonctionnalités, qu'au niveau conception. Le rythme des nouvelles idées apportées régulièrement par les chercheurs est extrêmement soutenu. Le temps imparti pour répondre au contrat *ESPON HyperAtlas Update* est largement insuffisant pour les implémenter toutes. De nombreuses évolutions font ainsi l'objet d'une rubrique « version 3 », sur l'interface en ligne de gestion des tâches du projet, parmi lesquelles :

- interactivité sur les diagrammes (zoom sur des intervalles dans les repères) ;
- carte de positionnement (mini-cadre replaçant les unités observées sur la carte globale) ;
- nouvelles couches de données (villes, fleuves, GoogleMaps) ;
- dynamisme et propositions pertinentes pour le choix des paramètres (tel numérateur a un sens, ou au contraire aucun, avec tels dénominateurs, sur telle aire d'étude) ;
- menus contextuels enrichis de métadonnées sur les unités, les indicateurs, de documents multimédia (photo, vidéo, liens Internet) ;
- jeux de données considérant des géométries multiples (maillage 2003, maillage 2006) ;
- curseur temporel « intelligent », ne proposant que des dates où les paramètres choisis sont valués ;
- etc.

Au niveau architecture, le groupe de recherche HyperCarte répond aujourd'hui à la demande d'une application Internet par une version *applet* d'HyperAtlas. Cette implémentation est parallèle aux travaux de Laurent Poulenard, entre avril 2010 et janvier 2011, pour son stage mémoire ingénieur CNAM, sur une nouvelle version du logiciel : HyperAtlas en mode services Web.

HyperAtlas vit depuis dix ans. Il s'enrichit, évolue, migre, et reste une solution originale et quasiment exclusive pour l'analyse territoriale multiscalaire.

Chapitre 12

HyperAdmin v2

12.1 Objectifs

Comme décrit dans la partie présentation, HyperAdmin est l'outil d'intégration dédié à la génération de fichiers sérialisés au format `.hyp` exploitables par HyperAtlas.

Nous partons de la version application de bureau du logiciel telle qu'elle est décrite dans le mémoire CNAM de Raphaël Thomas [Tho08].

Le groupe de recherche HyperCarte souhaite disposer d'un outil plus simple d'utilisation, plus rapide, plus fiable et vraiment dédié à l'intégration des données : HyperAdmin est, en effet, jusque là, conçu comme un HyperAtlas disposant d'un menu supplémentaire, l'assistant de création de projet. La complexité algorithmique de la création d'un projet, ajoutée à la complexité du processus d'exploitation des données comme le permet HyperAtlas, font d'HyperAdmin une application peu stable. Si les étapes de l'assistant de projet permettent de décomposer le processus complexe d'intégration, une quasi absence d'information laisse l'utilisateur non averti dépourvu devant les formulaires et écrans successifs.

En outre, l'application de bureau HyperAdmin nécessite la connexion à une base de données : l'utilisateur doit donc administrer un SGBD PostgreSQL, son extension PostGIS, installer la base de données attendue par l'application. La complexité de mise en place d'un tel environnement rend l'application peu utilisable par un non-informaticien, alors qu'elle est *a priori* destinée aux géographes et statisticiens.

Le contrat *ESPON HyperAtlas Update* spécifie la disponibilité d'un outil d'intégration : le passage à une version *applet* d'HyperAtlas et la proposition retenue par ESPON de la livraison de l'ensemble des services du projet sous la forme d'une application Web, mènent à l'objectif du portage complet de l'outil d'intégration HyperAdmin sous la forme client-serveur. Un objectif de cette tâche consiste donc à extraire la couche modèle de l'existant et de créer la couche vue (initialement les fenêtres modales de l'assistant de projet) selon des pages Web. Egalement requis dans le contrat ESPON, une documentation *ad hoc* doit permettre à l'utilisateur de pouvoir générer son jeu de données de façon autonome, sans recours à l'aide d'un membre du groupe de recherche HyperCarte.

Un autre objectif lié aux évolutions demandées sur l'interface HyperAtlas consiste à l'enrichissement du modèle de données généré par HyperAdmin : intégration du temps pour les indicateurs, disponibilité de métadonnées, styles pour des unités particulières (couleur, épaisseur du contour). Ces évolutions demandent, en conséquence, la revue du contenu des fichiers d'entrée à fournir par l'utilisateur.

12.2 Extraction de la couche métier

Un ensemble d'interfaces a été conçu afin d'envisager d'une façon plus souple les possibilités quant aux données d'entrée d'HyperAdmin. Comme illustré sur la figure 12.1, les entités attendues en entrée sont représentées par des interfaces :

- une interface fournissant les méthodes nécessaires à l'extraction de la structure ;
- une interface pour l'extraction des données statistiques ;
- une interface pour l'extraction de la géométrie.

Le système prévoit ainsi de manipuler des fichiers d'entrée au format texte, au format tableur Excel, ou via la récupération d'information depuis une connexion à une base de données.

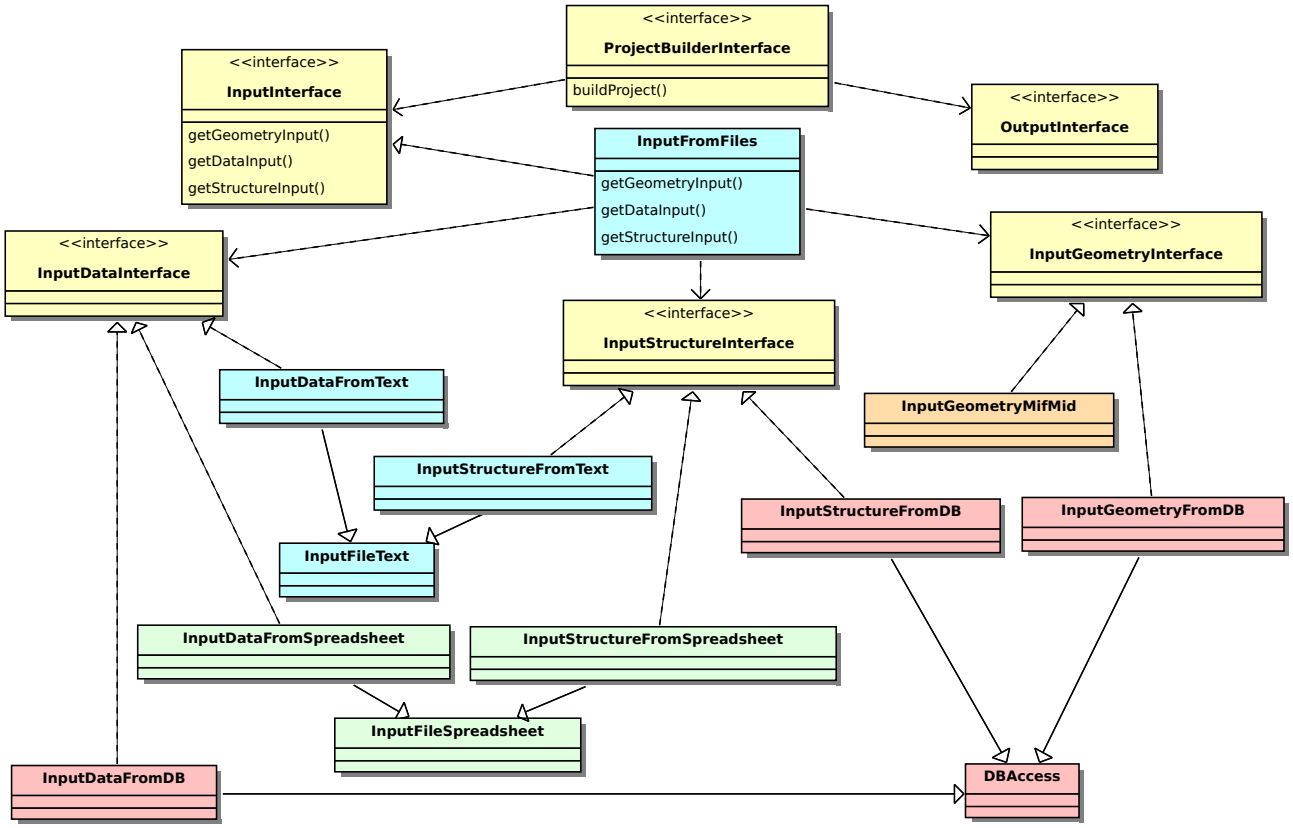


FIGURE 12.1 – Diagramme de classes de la couche modèle de HyperAdmin, interfaces débrayant les possibilités de fournir les données d'entrée via des fichiers texte (en bleu), tableur (en vert) ou via une base de données (en rouge).

Afin d'offrir à l'utilisateur la possibilité de générer des jeux de données via l'application Web, un important chantier est actuellement en cours pour extraire le cœur de métier du logiciel, jusqu'ici intégré dans une application de bureau. La difficulté tient principalement dans la récupération de la logique minimale nécessaire, encore aujourd'hui partiellement dépendante des composants graphiques Java Swing. Lors de la livraison fin décembre 2010, la simplification du processus de génération à l'aide de modèles structure/géométrie (voir section 10.4.5 page 78) a permis de temporiser sur la finalisation d'une version optimale. La librairie `hyperAdminModel.jar` appelée depuis l'application Web, et censée ne contenir que la couche métier, demande à être encore optimisée.

12.3 Enrichissement de la couche métier

Plusieurs évolutions décrites dans le chapitre consacré à HyperAtlas ont nécessité l'enrichissement des informations disponibles dans les fichiers `hyp`. La figure 12.2 illustre la composition des jeux de données tels qu'ils sont conçus dans la version 1 d'HyperAtlas.

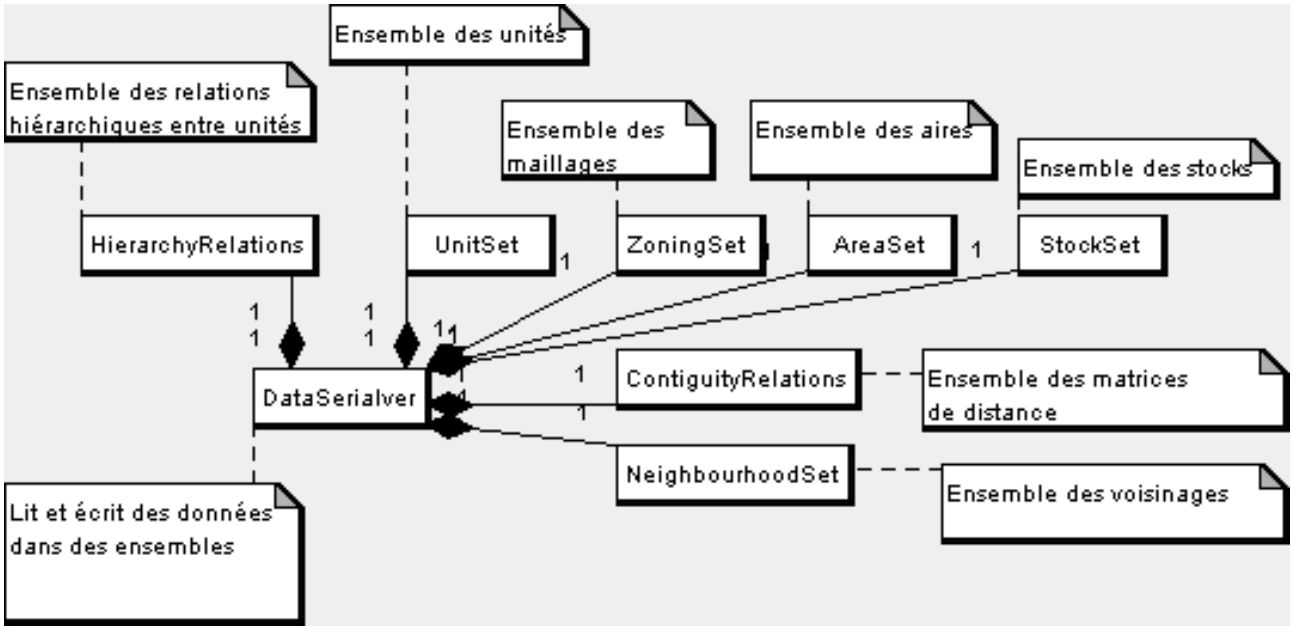


FIGURE 12.2 – Diagramme de composition des `.hyp` [Plu07].

12.3.1 Métadonnées du jeu de données

Afin de disposer d'informations générales sur le jeu de données, un simple composant JavaBean* nommé `SerialHypInfo` est ajouté au modèle de composition de la figure 12.2. Ce composant dispose des attributs suivants :

`version` indique la version de ce jeu de données en prévision de futures évolutions du modèle. Actuellement valué à 2.0 lors de la génération.

`author` indique par une chaîne de caractères le créateur de ce jeu de données. En créant un jeu de données depuis l'application Web, l'utilisateur a dû s'authentifier. Son compte utilisateur et son mot de passe renvoient à un enregistrement en base de données. La valeur de cet attribut `author` du jeu de données est donc instanciée en fonction du nom et prénom de l'utilisateur authentifié ;

`creationDate` indique la date de création du jeu de données, l'application se charge d'instancier cette valeur lors de la création ;

`timeEnabled` indique si les indicateurs proposent des valeurs pour différentes dates. La valeur booléenne est récupérée depuis le fichier `stock.xls` fourni par l'utilisateur. Ce dernier aura pris soin de renseigner le champ correspondant en fonction des valeurs et des indicateurs dont il dispose.

La classe `SerialHypInfo` hérite de la classe `SerialElement`, elle bénéficie ainsi des méthodes et attributs de sa classe mère, dont les classes spécialisées déjà existantes dans la version 1 sont

représentées sur la figure 12.3. Les différentes entités héritant de `SerialElement` représentent les unités, les maillages, les aires d'étude, les stocks, les voisinages et les matrices de distance. Ces entités sont destinées à être sérialisées avec leur identifiant et leur description, dans éventuellement plusieurs langues. La classe `SerialElement` factorise ces attributs et les méthodes permettant de créer ces entités, de les modifier, ou de les supprimer du jeu de données.

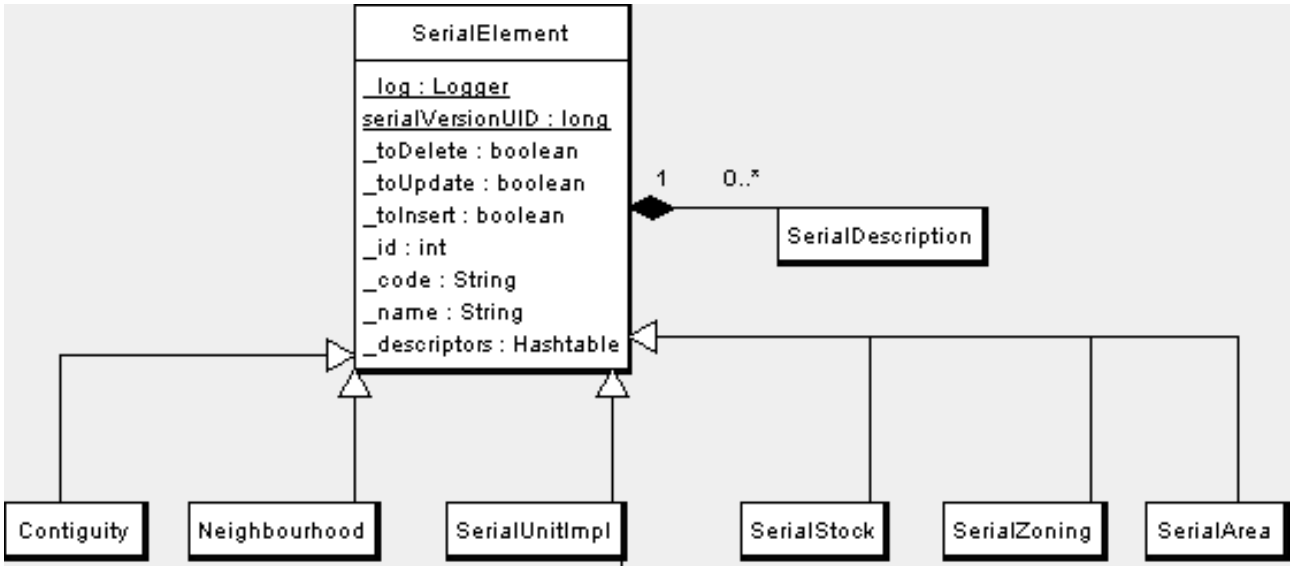


FIGURE 12.3 – Les entités spécialisant la classe `SerialElement` [Plu07].

La disponibilité d'informations sur le jeu de données est exploitée par HyperAtlas selon deux aspects : d'une part, les évolutions du format des jeux de données, indiquées par un numéro de version, peuvent être prises en compte par HyperAtlas, d'autre part, l'utilisateur est potentiellement mieux renseigné sur le jeu de données qu'il utilise, il peut plus facilement jongler entre plusieurs fichiers `.hyp` sans s'y perdre.

La classe `SerialHypInfo` est également utilisée pour stocker les paramètres par défaut à afficher au chargement du `.hyp` dans HyperAtlas. Lors de son ouverture, le jeu de données est ainsi analysé par HyperAtlas pour proposer les paramètres que le concepteur aura défini comme les plus appropriés au début de l'analyse. Actuellement en cours d'implémentation côté HyperAtlas, le nouveau format des fichiers en entrée d'HyperAdmin tiennent d'ores et déjà compte de cette information pour les indicateurs statistiques (voir section 12.3.3 page 117).

12.3.2 Analyseurs des fichiers d'entrée

Avant de stabiliser la définition du format des fichiers d'entrée, nécessitant quelques aménagements pour intégrer le temps, par exemple, le modèle d'analyse des fichiers en entrée d'HyperAdmin a été revu pour permettre des combinaisons de formats : la figure 12.4 montre l'interface principale, `UserDataSource`, utilisée pour extraire les informations fournies par l'utilisateur. Dans la version 1, deux classes, `ExcelDataSource` et `TextDataSource`, implémentent cette interface, pour respectivement traiter des fichiers tableur ou des fichiers au format texte. Pour cette version 2, le débrayage mis en place à l'aide d'interfaces permet de séparer le traitement des fichiers structure, du traitement des fichiers stocks, et du traitement des fichiers de contiguïté (éventuelles matrices distance-temps fournies par l'utilisateur).

La classe abstraite `AbstractDataSource` distribue les méthodes à implémenter à ses trois attributs interfaces conçues chacune pour s'occuper d'un type de donnée (structure stocks ou contiguïté). La figure 12.4 illustre les classes implémentant les différentes interfaces pour le traitement de fichiers tableurs Excel :

- `ExcelStructureParser` implémente les méthodes nécessaires à l'extraction des données d'un fichier structure « v2 » ;
- `ExcelStockParser` implémente les méthodes nécessaires pour l'extraction des statistiques ;
- `ExcelContiguityParser` implémente les méthodes pour l'extraction des matrices de contiguïté.

Les trois classes précédentes héritent de la classe abstraitre `ExcelAbstractParser` qui factorise les méthodes communes et utiles pour manipuler des fichiers Excel depuis Java. La librairie externe JExcelApi [1] est incorporée à l'application à cette fin : lire les différents onglets, parcourir le tableur par ligne ou colonne et récupérer les valeurs des cellules.

Pour la livraison du 31 décembre de l'application, le système mis en place a permis la construction de jeu de données à partir de fichiers Excel de structure inchangés et de fichiers Excel de stocks basés sur la nouvelle version décrite dans la section suivante. Considérant que les géographes et statisticiens utilisent principalement des fichiers au format tableur Excel, notons que le traitement de fichiers au format texte n'a pas été ré-implémenté pour cette version 2.

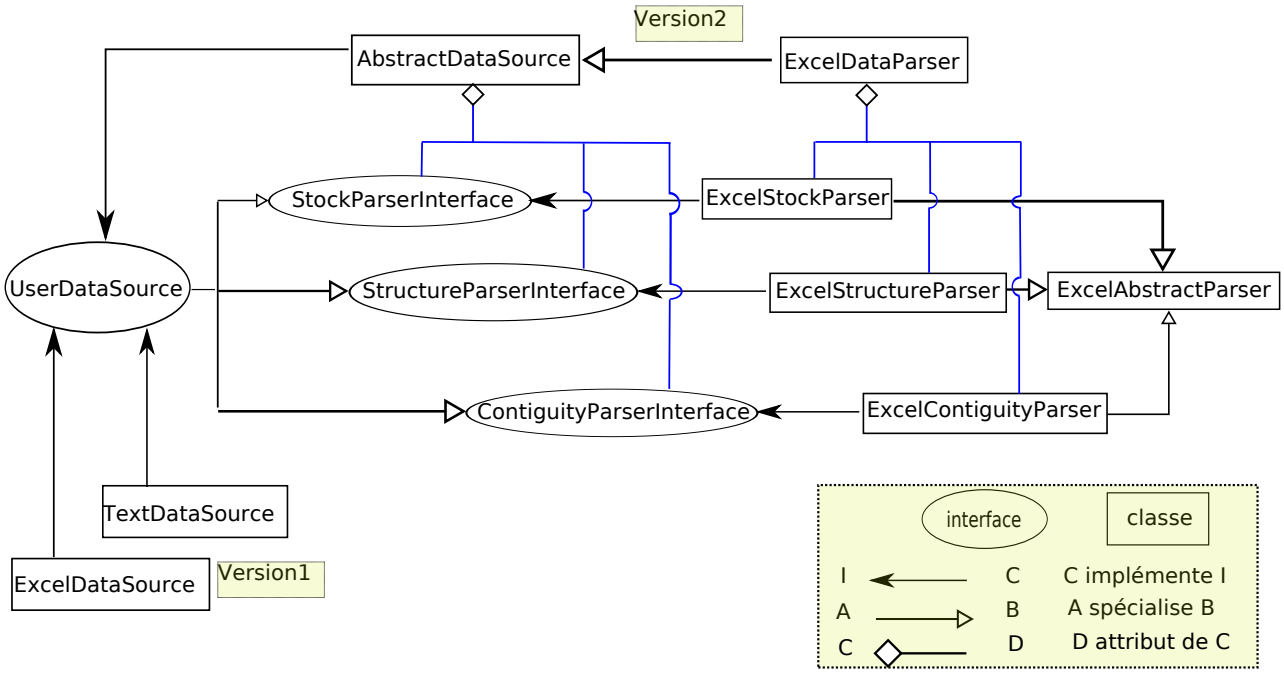


FIGURE 12.4 – Diagramme de classes pour l'analyse et l'extraction des informations fournies dans les fichiers en entrée d'HyperAdmin.

12.3.3 Redéfinition des fichiers de stocks

La prise en compte du temps, quant aux choix des indicateurs dans le panneau de paramètres de HyperAtlas, a nécessité l'enrichissement du modèle de données associé aux `.hyp` générés. La

1. *JExcelApi : A Java API to read, write and modify Excel spreadsheets* [en ligne] http://jexcelapi.sourceforge.net/ (consulté le 10 janvier 2011).

disponibilité de métadonnées sur les indicateurs a également été prise en compte pour le format des fichiers stocks « v2 ». Cette section décrit le format des fichiers tableurs de stocks attendus dorénavant par HyperAdmin v2, en énumérant dans l'ordre alphabétique les différents onglets qui composent ce fichier.

Avant tout, notons que le fichier de stocks fourni par l'utilisateur doit respecter les contraintes suivantes :

- le fichier doit être un tableur éditable par Microsoft Excel ou Open Office avec une extension `.xls`, le nom du fichier doit comporter le mot `data` : par exemple, `my_eu31_data.xls` ;
- les valeurs des stocks doivent être valuées pour toutes les unités territoriales au plus bas niveau de maillage.
- tous les onglets décrits ci-dessous sont obligatoires, le contenu de la première ligne de chaque onglet (excepté l'onglet `Data`) définissant le nom des colonnes est également obligatoire. Le nom des onglets n'est pas sensible à la casse.

Onglet `About`

Ce premier onglet du tableur de stocks permet d'identifier la version de son format et si le temps doit être considéré pour les indicateurs. Deux lignes et deux colonnes sont attendues, le tableau 12.1 montre l'exemple d'un jeu de données respectant le format 2.0, les indicateurs seront définis pour plusieurs dates (exemple : population en 2000, population en 2005).

VERSION	TIME_ENABLED
2.0	TRUE

Table 12.1 – Onglet `About` du tableur de stocks.

Onglet `Data`

Cet onglet doit être composé d'au moins trois colonnes : la colonne `UT_ID` liste les identifiants des unités territoriales au plus bas niveau de maillage de la structure et, au moins deux colonnes associent des valeurs pour chaque unité pour deux indicateurs statistiques. Le calcul des écarts nécessitent en effet un ratio de deux indicateurs. Le tableau 12.2 fournit un exemple du contenu attendu : la population en 2000 pour l'unité `AT113` est de 5 habitants.

Notons que la première ligne liste les identifiants des indicateurs qui doivent être définis dans l'onglet `StockInfo` décrit section 12.3.3 page 120.

UT_ID	pop2000	pop2002	area2000	gdp2000	gdp2002
AT111	1	15	2	7	10
AT112	3	16	4	8	11
AT113	5	17	6	9	12

Table 12.2 – Exemple du contenu de l'onglet `Data` du tableur de stocks.

Onglet `Default`

Cet onglet a pour objectif de définir les indicateurs à utiliser par défaut dans le panneau de paramètrage, au chargement du jeu de données.

Comme illusré par le tableau 12.3, cet onglet est composé de deux lignes et deux colonnes pour respectivement spécifier les identifiants des indicateurs à utiliser par défaut pour le paramètre numérateur (colonne `DEFAULT_NUM`) et pour le paramètre dénominateur (colonne `DEFAULT_DEN`). Les identifiants de stocks doivent être définis dans l'onglet `StockInfo`.

DEFAULT_NUM	DEFAULT_DEN
pop2000	area2000

TABLE 12.3 – Exemple du contenu de l'onglet `Default` du tableur de stocks.

Le contenu de cet onglet est actuellement considéré comme une partie des métadonnées du jeu de données, il est stocké dans le `.hyp` à l'aide du composant `SerialHypInfo` décrit section 12.3.1.

Onglet `Label`

Cet onglet permet l'internationalisation des noms et descriptions des différents indicateurs et ratios prédéfinis. Les deux premières colonnes `LABEL_ID` et `LANG_CODE` représentent en quelque sorte les index de cette table : pour un identifiant de label donné peuvent exister plusieurs traductions. Dans le tableau exemple 12.4, l'identifiant `LABEL_ID = 1` est disponible en anglais (`LANG_CODE = EN`) et en français (`LANG_CODE = EN`). Dans l'onglet `StockInfo`, chaque indicateur référence un identifiant de label. Cet onglet permet de factoriser les noms et descriptions se rapportant à plusieurs indicateurs, ou au même indicateur défini à plusieurs dates.

LABEL_ID	LANG_CODE	NAME	DESC
1	EN	Total population	Total population in thousands
1	FR	Population totale	Population totale en milliers
2	EN	Area	Total area
2	FR	Superficie	Superficie totale
3	EN	GDP	Gross domestic product
3	FR	PIB	Produit intérieur brut
4	EN	GDP/Inh	Gross domestic product per inhabitant
4	FR	PIB/Hab	PIB par habitant
5	EN	Density	Density of population
5	FR	Densité	Densité de population

TABLE 12.4 – Exemple du contenu de l'onglet `Label` du tableur de stocks.

Onglets `Metadata` et `Provider`

Ces deux onglets visent à répondre à la requête d'ESPON quant à la disponibilité dans HyperAtlas de l'affichage d'information sur les métadonnées* des indicateurs. Avant de présenter le contenu de ces onglets, cette section décrit l'objectif et l'intérêt des métadonnées.

La directive INSPIRE[2] (*Infrastructure for Spatial Information in the European Community*), approuvée par la Communauté Européeene, vise à favoriser l'échange des données au sein de l'UE. Parmi les obligations de cette directive, la fourniture des données doit respecter des règles de mise en œuvre communes, et des métadonnées doivent être constituées pour en faciliter l'accès et l'interopérabilité. Pour le partage de l'information géographique à référence spatiale et temporelle, dans le domaine des métadonnées, INSPIRE préconise l'usage de la norme ISO 19115[3].

Aussi, dans le cadre du projet *ESPON Database*, une extension de la norme ISO 19115, le profil `esponMD` [PVOG10], a été développée par STEAMER. La figure 12.5 résume le flot de données autour de la base ESPON, accessible via le portail développé par STEAMER. La base stocke toutes les informations sur les données et les métadonnées dans le respect du standard ISO.

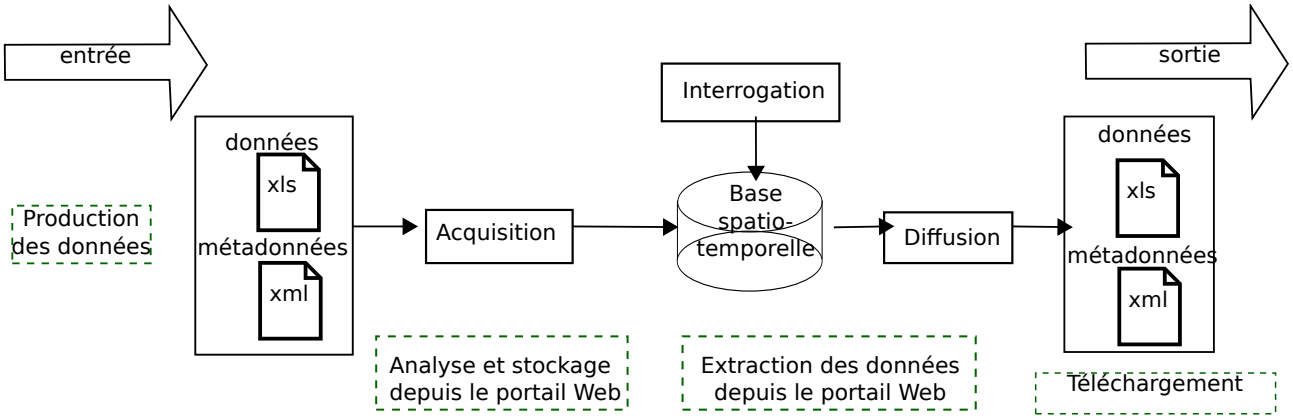


FIGURE 12.5 – Flot de données autour de la base ESPON.

L'éditeur de métadonnées au sein du portail Web, développé par l'équipe pour le projet *ESPON Database*, rencontre un succès grandissant. L'intérêt des métadonnées est enfin pris en compte par les fournisseurs et utilisateurs de données. Aussi comprend-on aisément la requête d'ESPON quant à la disponibilité de métadonnées dans HyperAtlas. La vocation première d'HyperAtlas n'est cependant pas de représenter une information aussi riche, quant aux métadonnées, que le portail Web, développé à cette fin, le permet. Si la perspective d'établir un pont direct entre la base de données ESPON et HyperAtlas ou HyperAdmin paraît pertinente, la proposition retenue pour l'heure actuelle consiste à enrichir les jeux de données générés d'une information minimale sur les indicateurs : d'une part, en exploitant les descriptions des indicateurs (voir onglet `Label`), et, d'autre part, en introduisant deux nouveaux onglets `Metadata` et `Provider`.

Les nouveaux jeux de données fournis à ESPON fin décembre 2010 ont ainsi été générés à partir des fichiers tableurs sur ce nouveau modèle, ils initient d'ores et déjà l'intégration de métadonnées. L'information disponible actuellement en termes de métadonnées consiste à associer le fournisseur des données (par exemple, EUROSTAT, la base de données ESPON, l'INSEE), en fonction de l'indicateur et de l'unité territoriale.

Comme illustré dans le tableau 12.5, l'onglet `Metadata` est composé de trois colonnes, il représente une table d'association entre les trois entités « unité territoriale », « indicateur » et « fournisseur des données » :

2. *European Commission INSPIRE Geoportal* [en ligne] http://www.inspire-geoportal.eu/index.cfm (consulté le 10 janvier 2011).

3. *ISO 19115 :2003 - Geographic information - Metadata* [en ligne] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=26020 (consulté le 10 janvier 2011).

– `UT_ID` est la première colonne dont les cellules peuvent être vides, ou au contraire contenir l'identifiant d'une unité territoriale ;
– `STOCK_ID` liste les identifiants des indicateurs pour lesquels des métadonnées sont disponibles (les valeurs des identifiants doivent être définies dans l'onglet `StockInfo`) ;
– `PROVIDER_ID` référence des identifiants de fournisseurs de données, lesquels doivent être définis dans l'onglet `Provider`.

L'onglet `Provider`, dont un exemple est donné dans le tableau 12.6, liste les fournisseurs des données. Chaque entrée indique le nom d'un organisme, un contact et une URL. Ces trois attributs composent, pour l'instant, l'information basique disponible à propos des fournisseurs de données.

Ces deux onglets permettent donc pour l'instant d'associer une information minimale, le fournisseur, pour un indicateur, relativement à une unité territoriale, ou globalement, pour toutes les unités. Les valeurs des indicateurs peuvent, en effet, provenir de différentes sources en fonction des unités qu'ils décrivent. En prenant l'exemple du tableau 12.5, les valeurs de la population en 2000 pour les unités `AT111` et `AT112` sont issues de deux fournisseurs différents. Par contre, toutes les valeurs de l'indicateur `area` proviennent du fournisseur 2, quelle que soit l'unité.

UT_ID	STOCK_ID	PROVIDER_ID
AT111	pop2000	1
AT112	pop2000	2
	area	2
	pop2002	1

Table 12.5 – Exemple du contenu de l'onglet `Metadata` du tableur de stocks.

PROVIDER_ID	NAME	CONTACT	URL
1	Eurostat	toto@eurostat.eu	http ://www.eurostat.eu
2	INSEE	tata@insee.fr	http ://www.insee.fr

Table 12.6 – Exemple du contenu de l'onglet `Provider` du tableur de stocks.

Le principe de l'intégration de métadonnées des indicateurs dans les jeux de données `.hyp`, générés par HyperAdmin, est donc amorcé, pour un cas simple, l'indication du fournisseur de données pour un stock et une unité. L'intégration de cette information dans les `.hyp` est en cours, son exploitation dans HyperAtlas selon un affichage simple et ergonomique est en phase d'étude. Compte-tenu des travaux de l'équipe STEAMER réalisés dans le cadre du projet *ESPON Database*, on peut cependant considérer cette intégration comme temporaire. La capitalisation du modèle de données tel qu'il est conçu pour les métadonnées de la base ESPON pointe une perspective de mise en place d'une relation plus forte entre HyperAdmin/HyperAtlas et cette base ESPON.

Onglet `RatioStock`

Cet onglet facultatif a pour objectif de proposer des ratios prédéfinis. Dans HyperAtlas, si le jeu de données chargé contient des ratios prédéfinis, ceux-ci remplissent la boîte de sélection « Ratio », dans le panneau de paramètrage. Un ratio prédéfini est un raccourci pour la sélection automatique d'un indicateur au numérateur et au dénominateur.

Cet onglet est composé des champs suivants :

- `RATIO_ID` est un identifiant unique pour un ratio prédéfini ;
- `LABEL_ID` référence l'identifiant d'un label (nom et description dans différentes langues) défini dans l'onglet `Label` (voir section 12.3.3 page 117) ;
- `NUM_ID` fournit l'identifiant d'un indicateur à considérer au numérateur de ce ratio ;
- `DEN_ID` fournit l'identifiant d'un indicateur à considérer au dénominateur de ce ratio ;
- `VALIDITY_START` donne la date de départ de l'intervalle temporel à laquelle se rattache ce ratio ;
- `VALIDITY_END` donne la date de fin de l'intervalle temporel à laquelle se rattache ce ratio.

Les champs `VALIDITY_START` et `VALIDITY_END` ne sont considérés que si le temps est activé pour le jeu de données. Si `VALIDITY_START = VALIDITY_END`, le ratio est défini pour une date et non un intervalle.

En s'appuyant sur les tableaux 12.4 et 12.8, le tableau 12.7 donne un exemple de quatre ratios prédéfinis pour le jeu de données à créer :

- le PIB par habitant,
  - en 2000 (deuxième ligne) ;
  - en 2002 (troisième ligne) ;
- la densité de population,
  - en 2000 (quatrième ligne) ;
  - en 2002 (cinquième ligne) ;

RATIO_ID	LABEL_ID	NUM_ID	DEN_ID	VALIDITY_START	VALIDITY_END
1	4	gdp2000	pop2000	2000	2000
2	4	gdp2002	pop2002	2002	2002
3	5	pop2000	area2000	2000	2000
4	5	pop2002	area2000	2002	2002

TABLE 12.7 – Exemple du contenu de l'onglet `RatioStock` du tableur de stocks.

Onglet `StockInfo`

Le tableau 12.8 fournit un exemple de cet onglet destiné à définir les indicateurs grâce aux champs suivants :

- `STOCK_ID` est l'identifiant unique de l'indicateur ;
- `LABEL_ID` référence l'identifiant d'un label (nom et description) défini dans l'onglet `Label` ;
- `MEASURE_UNIT` fournit l'unité de mesure des valeurs fournies dans l'onglet `Data` ;
- `VALIDITY_START` donne la date de départ de l'intervalle temporel à laquelle se rattache cet indicateur ;
- `VALIDITY_END` donne la date de fin de l'intervalle temporel à laquelle se rattache cet indicateur ;
- `VISIBLE_FLAG` indique par un booléen si cet indicateur doit apparaître dans les boîtes de sélection du numérateur et dénominateur d'HyperAtlas. Si la valeur est à faux, cet indicateur est uniquement utile pour les ratios prédéfinis, mais n'a pas une signification en tant que tel intéressante. C'est le cas par exemple pour l'espérance de vie, définie comme ratio, mais dont les numérateur et dénominateur n'ont pas d'intérêt pour d'autres analyses.

Comme pour l'onglet `RatioStock`, les champs `VALIDITY_START` et `VALIDITY_END` ne sont considérés que si le temps est activé pour le jeu de données. Si `VALIDITY_START = VALIDITY_END`,

l'indicateur est défini pour une date et non un intervalle.

STOCK_ID	LABEL_ID	MEASURE	VALIDITY_START	VALIDITY_END	VISIBLE
pop2000	1	*1000	2000	2000	TRUE
pop2002	1	*1000	2002	2002	TRUE
area2000	2	km2	2000	2000	TRUE
gdp2000	3	euros	2000	2000	TRUE
gdp2002	3	euros	2002	2002	TRUE

TABLE 12.8 – Exemple du contenu de l'onglet `StockInfo` du tableur de stocks.

12.3.4 Intégration des villes

La tâche consistant à afficher dans *ESPON HyperAtlas* les capitales des pays n'a pas encore été implémentée. Lors d'une réunion en novembre 2010, le groupe de recherche HyperCarte a estimé que l'intégration de ce service ne pouvait être effectuée de façon générique dans le délai imparti pour la livraison du 31 décembre 2010. L'intégration des capitales peut en effet être entendue comme la possibilité de créer des jeux de données suffisament compatibles pour pouvoir y ajouter des couches de données géométriques. Nous préférons donc reporter cette évolution pour considérer une évolution plus conséquente, à savoir la possibilité générique de superposer n'importe quel type de couche : les fleuves, les réseaux routiers, les villes principales, etc.

Avant de concevoir une telle possibilité générique, une étape intermédiaire a néanmoins été abordée pour intégrer les capitales. La méthode repose sur l'idée de fournir en entrée d'HyperAdmin un nouveau fichier géométrique, en complément de la paire MIF-MID initiale décrivant les polygones des unités territoriales. MapInfo met en en effet à disposition un format permettant de localiser des points dans un référentiel de coordonnées, sous la forme d'un fichier texte.

Une telle intégration requiert la prise en compte des points suivants :
- un analyseur du fichier géométrique MapInfo décrivant des points ;
- l'incorporation des points au fichier sérialisé représentant le jeu de données ;
- côté HyperAtlas, l'affichage des points du jeu de données doit tenir compte du facteur de zoom courant et du niveau de généralisation.

L'ensemble de cette évolution est en phase d'étude.

12.4 Synthèse d'HyperAdmin v2

En conclusion des réalisations effectuées sur HyperAdmin, le processus d'intégration des données est en pleine mutation. L'approche initiale de l'outil, conçu comme un HyperAtlas application de bureau, disposant d'un menu supplémentaire, a été bouleversée pour s'adapter à une version Web.

A cette fin, la couche métier, le cœur de l'intégration, est encore en cours de séparation de la couche de visualisation, et des composants Java Swing auxquels elle est encore fortement intégrée.

A cette occasion, le processus d'intégration des données est devenu plus modulaire, pour répondre parallèlement aux besoins d'HyperAtlas, aux évolutions nécessaires et introduites progressivement dans les fichiers d'entrée.

Pour le géographe ou statisticien, utilisateurs ciblés par HyperAdmin, l'évolution la plus significative réside donc dans les modifications introduites sur le format des fichiers tableurs à fournir en entrée. Aussi avons-nous consacré dans ce chapitre une part importante à la description du nouveau format du fichier « stocks ». Sa description a permis de compléter l'information, abordée dans le chapitre HyperAtlas, quant à l'intégration du temps ou des métadonnées.

Le format des fichiers « structure » devrait subir moins d'évolutions. Un onglet `Default` est ajouté pour définir les paramètres à afficher par défaut lors du chargement dans HyperAtlas, pour les boîtes de sélection « aire d'étude » et « maillage élémentaire ». Décrits dans le chapitre HyperAtlas, deux onglets sont également ajoutés pour associer des styles à quelques unités particulières (Chypre Nord en blanc, par exemple).

L'objectif premier était de maintenir une version d'HyperAdmin capable de générer les nouveaux jeux de données, attendus lors de la livraison du 31 décembre, et pourvus de capacités répondant aux évolutions d'HyperAtlas (gestion du temps, notamment). Les quatre nouveaux jeux de données attendus par ESPON ont pu être générés, à partir de fichiers « structure » version 1 et de fichiers « stocks » version 2. Le travail des géographes et statisticiens de l'équipe RIATE est donc totalement validé par la disponibilité de ces nouveaux jeux de données, portant sur des thématiques et géométries variées :

- selon une délinéation des NUTS millésime 2006 :
  - un jeu de données dont la thématique porte sur la démographie et les migrations de population, le maillage élémentaire considère le niveau NUTS 2 (niveau des régions administratives françaises), l'aire d'étude peut s'étendre jusqu'à 31 pays européens (EU + 4 pays partenaires), ou considérer les pays candidats ;
  - un jeu de données portant sur l'économie, mêmes maillage et aires d'étude que le jeu de données « démographie » ;
  - un jeu de données portant sur l'utilisation des sols et l'environnement, les données sont issus de l'EEA (Agence Européenne de l'Environnement), basées sur les CLC (données *Corine Land Cover*), le maillage élémentaire est le NUTS 3 (départements), les aires d'étude considèrent les pays candidats, EU+5, ou les pays de l'ex-Yougoslavie ;
- une géométrie WUTS*, couvrant l'espace européen et ses voisins au Sud et à l'Est, du Sahara Occidental au Moyen-Orient. Ce jeu de données, bâptisé *EUROMED*, illustre brillament l'intérêt de l'analyse multiscalaire territoriale pour une compréhension globale des phénomènes, dont celui de la place du continent européen parmi ses proches voisins.

Côté performances, la quantité d'information fournie dans les fichiers en entrée pour les nouveaux jeux de données ne permet pas une comparaison équitable avec la version précédente d'HyperAdmin. Les nouveaux jeux de données sont maintenant conçus de façon plus thématique, ils incorporent moins d'indicateurs. Mais surtout, le jeu de données ESPON 2007 considérait les niveaux élémentaires de maillage NUTS 2-3 (intermédiaire aux niveaux 2 et 3) et NUTS 3. L'intégration de tels jeux de données nécessitait environ vingt minutes. La quantité d'information en entrée est disproportionnée pour établir une comparaison, mais pour information, la génération des jeux de données de décembre 2010 avec HyperAdmin v2, dépouillé de son interface graphique, nécessite entre vingt secondes (NUTS 2) et une minute (NUTS 3).

Doivent cependant être bientôt (janvier-février 2011) fournies par RIATE des données aux niveaux de maillages LAU* 2 et LAU 3 : les performances d'HyperAdmin v2 pourront alors être évaluées avec plus de pertinence.

Quatrième partie

Conclusion et annexes

Conclusion

La conclusion de ce mémoire dresse un bilan des travaux effectués pendant le stage. Après une évaluation quantitative puis qualitative de ces réalisations, quelques perspectives sont proposées pour les versions futures des logiciels HyperAtlas et HyperAdmin.

Suivent quelques considérations qui n'ont pas été abordées dans ce mémoire, et enfin, un bilan personnel.

Rappel des objectifs

L'objectif principal de ce stage mémoire ingénieur était de répondre au cahier des charges du projet *ESPON HyperAtlas Update*. La majeure partie du travail a donc consisté à concevoir et implémenter un ensemble d'évolutions sur les versions existantes des logiciels HyperAtlas et HyperAdmin. En outre, un objectif d'amélioration de la qualité globale des logiciels était souhaitée.

Synthèse des réalisations

L'état de l'art a positionné la version initiale d'HyperAtlas vis-à-vis de quelques unes des solutions existantes actuellement dans le domaine de la représentation des données statistiques à référence spatiale et temporelle. Ce mémoire a ensuite décrit la démarche, les moyens et outils mis en œuvre pour la réalisation des tâches demandées.

Les chapitres de la partie « Réalisation » ont développé et décrit plus précisément les objectifs des évolutions, leur conception et leur implémentation. Pour chacune de ces évolutions, une capture d'écran a permis au lecteur d'entrevoir leur intégration dans l'interface.

Le tableau 12.9 propose une synthèse des réalisations, en reprenant les évolutions majeures contractées dans le cahier des charges. Un pourcentage indique la complétude d'achèvement de chacune des principales tâches lors de la livraison intermédiaire du projet au 31 décembre 2010. Selon un diagramme de Gantt, la figure 12.6 illustre de façon indicative la part chronologique consacrée à ces différentes réalisations.

Rappelons que le stage de neuf mois ne couvrait qu'une partie du temps imparti à la livraison finale du projet complet, s'étalant sur un an. Aussi ce document ne porte-t-il que sur la version intermédiaire du projet, livrée aux dix douzièmes de sa finalisation.

Notons que la réalisation de ces tâches est à considérer selon une démarche et un environnement de développement dont la mise en œuvre a été décrite dans le chapitre 9 : partage des sources, tests unitaires, construction automatique, plate-forme de tests, une documentation technique est désormais maintenue, la Javadoc a été corrigée (0 erreur, 0 avertissement), l'ensemble permettant une collaboration efficace.

Tâche	Complétude	Commentaire
Version Web	100 %	Corrigée, améliorée et intégrée dans une application Web
Temps	100 %	Représentation améliorée, modèle générique
Métadonnées	70 %	Pas encore affichées dans HyperAtlas
Zones d'étude	100 %	Pour les unités au plus haut niveau de la hiérarchie
Redistribution	100 %	Cartes disponibles en mode expert
Statistiques	100 %	Module indépendant pour le calcul des statistiques
Synthèse binaire	100 %	Nouvelle typologie synthétique des écarts
Interface	90 %	Capitales non encore intégrées, mais styles mis à jour
HyperAdmin	80 %	En cours d'optimisation du modèle pour un accès Web
Manuel utilisateur	95 %	Largement amélioré et complété
Configuration	100 %	Elimination du code en dur

TABLE 12.9 – Complétude des tâches réalisées.

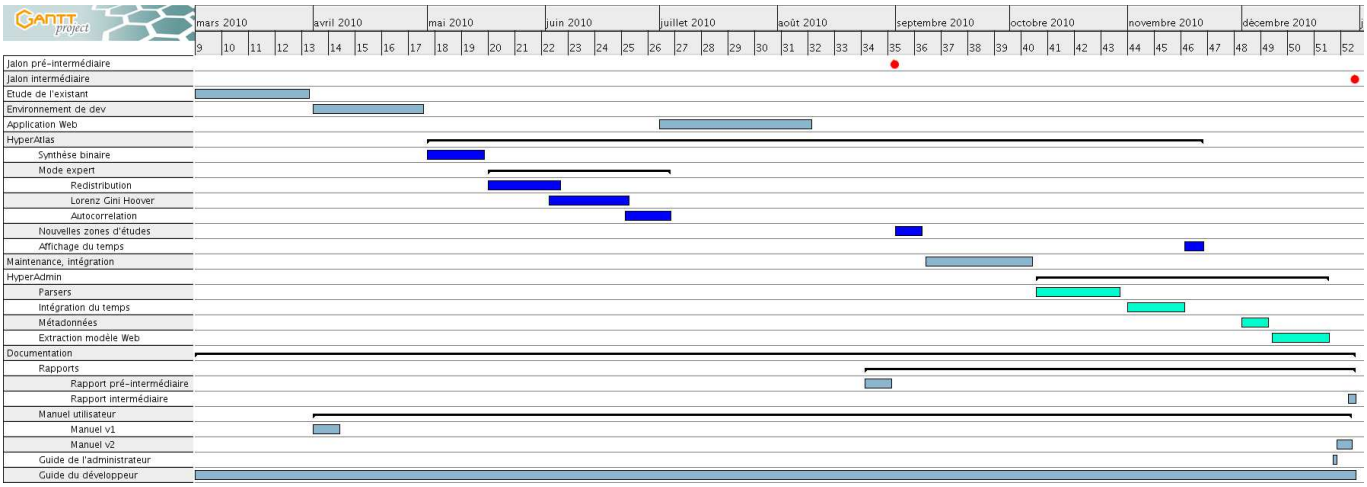


FIGURE 12.6 – Chronologie indicative des principales réalisations selon un diagramme de Gantt.

Agenda

La livraison finale des réalisations attendue par le projet *ESPON HyperAtlas Update* est prévue pour le 28 février 2011. Le délai entre les livraisons intermédiaire et finale est essentiellement alloué aux tâches suivantes :

- la correction des bugs répertoriés sur l'outil proposé par LIGforge, dont deux critiques : la réinitialisation des composants de l'interface graphique lors du chargement d'un nouveau jeu de données, la discrétisation des classes de valeurs pour les cartes d'écarts ;
- la revue de l'interface proposée en mode expert : après avoir testé le système de fenêtres internes, les géographes de RIATE préfèrent désormais un système de sous-onglets ;
- l'affichage des capitales des pays européens sur les jeux de données : les coordonnées des points seront intégrées dans le fichier structure en entrée d'HyperAdmin ;
- la génération améliorée des rapports d'analyse au format HTML.

Le manuel utilisateur sera alors revu et complété de scenarios typiques quant à l'élaboration d'une analyse, pour un utilisateur *lambda*, et un utilisateur avancé.

L'objectif est de fournir dans le temps imparti une distribution stable des logiciels *ESPON HyperAtlas* et *ESPON HyperAdmin*.

Evaluation quantitative

Le tableau 12.10 confronte le nombre de fichiers `.java` dans la révision originale du projet et dans la révision 534, fin décembre 2010, en fonction des paquetages.

paquetage	révision 0	révision 534
`com.jgoodies`	110	110
`hypercarte`	0	7
`hypercarte.data`	24	24
`hypercarte.hyperadmin`	94	108
`hypercarte.hyperatlas`	203	291
`hypercarte.hypersmooth`	35	-
`org.steamer.hypercarte.deviation`	-	12
`org.steamer.hypercarte.hyperadmin.model`	-	21
`org.steamer.hypercarte.stat`	-	60
`org.steamer.hypercarte.web`	-	44
`org.steamer.util`	-	10
Total	465	687

TABLE 12.10 – Nombre de fichiers `.java` dans les paquetages du projet.

Parmi les nouvelles classes ajoutées, notons que 73 d'entre elles sont des classes de tests unitaires JUnit. La sortie console suivante indique le nombre de tests unitaires lancés automatiquement à chaque construction des logiciels, pour l'ensemble du projet (HyperAtlas et HyperAdmin) :

```
junit:
    [junit] Running hypercarte.MainJunitTest
    [junit] Tests run: 132, Failures: 0, Errors: 0, Time elapsed: 42.815 sec
```

Evaluation qualitative

Afin d'évaluer ces réalisations, cette section s'appuie sur l'outil de diagnostic SWOT (*Strengths, Weaknesses, Opportunities, Threats*), pour confronter respectivement des facteurs internes (forces, faiblesses) et externes (opportunités et menaces), selon une matrice représentée dans le tableau 12.11.

	Positif	**Négatif**
Interne	**Forces**	**Faiblesses**
Externe	**Opportunités**	**Menaces**

TABLE 12.11 – Matrice SWOT.

Avant d'énumérer quelques éléments pour chacune des cellules de cette matrice, rappelons quatre critères permettant une mesure de la qualité d'un logiciel [Aou03] :

- validité : le logiciel répond-il exactement aux tâches définies ?
- réutilisabilité : le logiciel peut-il être réutilisé en tout ou en partie ?
- extensibilité : avec quelle facilité le logiciel peut-il être adapté aux changements de ses spécifications ?
- compatibilité : quelle est l'aptitude du logiciel à être combiné avec d'autres modules ou logiciels ?

Les réponses à ces questions sont réparties et argumentées dans les cellules de la matrice SWOT, décrites ci-dessous.

Forces

Pour le critère de validité, les tests unitaires aident à valider la non-régression des fonctionnalités introduites. HyperAtlas et HyperAdmin démontrent depuis dix ans une extensibilité convaincante, tant au niveau de l'architecture relativement adaptable, que des fonctionnalités. Le concept d'analyse territoriale multiscalaire est une quasi exclusivité d'HyperAtlas. L'intégration de ses propres données avec HyperAdmin est difficile mais possible. Les nouveaux outils d'analyse statistique sont modulaires et indépendants. L'ensemble est maintenant disponible depuis l'Internet.

Côté développement, la maintenabilité a été améliorée par la factorisation du code commun à HyperAtlas et HyperAdmin. La rédaction des documentations, la correction de la Javadoc, la mise en place d'un environnement multiplate-forme, fournissent une base solide pour la poursuite du développement d'évolutions par de nouveaux développeurs.

Une attention particulière a également été portée sur l'archivage systématique et versionné des documents produits pendant le projet (compte-rendus de réunions, distributions, bibliographie, données).

Faiblesses

L'intégration des données avec HyperAdmin reste complexe : la préparation des fichiers tableurs et des géométries MIF-MID n'est pas accessible à tous. L'intégration doit être facilitée.

Pour le développeur, la Javadoc et le guide du développeur s'appliquent essentiellement aux nouveaux éléments. La documentation technique pour des points antérieurs est pauvre. Les singletons nuisent au critère de modularité. Les interfaces sont peu explicites.

Opportunités

La place du groupe de recherche HyperCarte dans le programme ESPON devrait permettre une diffusion européenne. La mise à jour de démonstrations gratuites sur les sites Web devrait également aider à promouvoir HyperAtlas dans d'autres contextes, collaborations inter-universitaire par exemple.

Menaces

La concurrence est très forte en termes de solutions Internet pour la représentation des statistiques à référence spatiale et temporelle. Beaucoup d'applications basées sur les technologies Flash offrent des interfaces conviviales, proposant des outils multiples : diagrammes divers, tableaux, outils de rapports, curseurs temporels et animations.

Perspectives

Pour pouvoir visualiser les données de son choix dans HyperAtlas, l'utilisateur est aujourd'hui contraint de collecter des données statistiques, puis de les formater selon le modèle de tableur attendu par HyperAdmin. Passée cette étape d'édition, il peut alors générer un `.hyp` exploitable par HyperAtlas.

Le flot de données d'un tel processus, illustré par les figures 1.1 et 12.5, devrait être considérablement facilité. Suggérée plusieurs fois dans ce document, l'évolution certainement la plus pertinente pour les utilisateurs des applications ESPON consisterait donc à pouvoir plus facilement visualiser les données de la base de données ESPON à travers HyperAtlas. La possibilité de génération de `.hyp` depuis l'interface d'extraction des données de la base ESPON devrait ainsi être prochainement étudiée. Solution d'autant plus facilitée par le fait que les deux solutions sont conçues et maintenues par le groupe de recherche HyperCarte. Le lien entre les deux projets constituerait un bénéfice considérable, tant pour l'utilisateur, que sous l'angle de la conception, réalisation et implémentation des outils partagés entre les différents projets STEAMER.

La question de cette opportunité sera d'ailleurs prochainement soumise à travers le sondage réalisé par RIATE, visant à récupérer l'avis des utilisateurs partenaires européens susceptibles d'utiliser les deux applications *ESPON Database* et *ESPON HyperAtlas*.

Gardons cependant à l'esprit qu'HyperAtlas et HyperAdmin restent la propriété du groupe de recherche HyperCarte. Si la base de données ESPON peut être considérée comme une source importante d'information pour l'Europe, HyperAdmin doit pouvoir vivre indépendamment, et permettre l'intégration de données et géométries diverses et variées. Le fichier tableur et le format d'échange d'information géographique MapInfo demeurent des moyens simples et efficaces pour leur interopérabilité et relative facilité d'utilisation. Néanmoins, pour ouvrir la possibilité d'intégration aux non-géographes, non-statisticiens, non-informaticiens, la convivialité d'HyperAdmin pourra nettement être améliorée.

Quant à l'architecture, grâce à l'API Java Swing, HyperAtlas est maintenu en tant qu'application de bureau et en version *applet*. HyperAdmin migre en version client-serveur pour son accessibilité depuis l'Internet. Son modèle a été revu pour permettre une ouverture plus facile vers des sources de données variées. En outre, grâce au stage ingénieur CNAM de Laurent Poulenard, une version serveur WMS (*Web Map Service*) d'HyperAtlas et un client WMS en une application de bureau sont aujourd'hui disponibles. STEAMER a donc déjà anticipé sur les probables demandes futures pour cette suite de logiciels.

Enfin, la technologie et les outils SOLAP devraient prochainement constituer un axe de recherches et d'études pour la réalisation d'un « HyperAtlas^3 ». Ce nouvel objectif pourrait permettre l'alliance des fonctionnalités OLAP (outils de forage, tableaux de bord, diagrammes, rapports) aux outils d'analyse courants.

Le groupe de recherche HyperCarte souhaite donc capitaliser ses réalisations, poursuivre la recherche et le développement, et décliner les versions en fonction des futures opportunités de projets. L'intégration de jeux de données considérant des bassins versants, ou la personnalisation d'une interface ciblant des utilisateurs collégiens ne sont que deux exemples parmi les multiples perspectives du groupe de recherche.

Ce dont il n'a pas été question...

Ce mémoire ne constitue pas un rapport exhaustif au sujet du projet *ESPON HyperAtlas Update*. Notre approche a principalement considéré l'aspect informatique du projet, sous l'angle de vue de la conception et de l'implémentation de nouvelles fonctionnalités. Ce document reflète plutôt fidèlement les tâches auxquelles j'ai été affecté pendant ce stage.

Ainsi n'a-t-il pas été question par exemple du sommet « Coût », du triangle « Coût Qualité Délai », inhérent à la gestion d'un projet. Les aspects budgétaires et économiques ont, en effet, été intégralement pris en charge par une collaboration entre Madame Villanova-Oliver pour STEAMER et Madame Salmon pour RIATE.

Monsieur Gensel, en tant que *Lead Partner*, a, entre autres, assuré les communications avec ESPON, la coordination du projet, et la responsabilité de l'ensemble. Cette répartition des tâches a donc facilité ma concentration sur les aspects techniques.

Bilan personnel

A titre personnel, je retire de ce stage un bénéfice extrêmement positif, sur de multiples points.

Appliquer l'informatique au domaine de la géographie était un souhait de longue date. Grâce à Monsieur Gensel, j'ai pu aborder la géomatique quand il m'a proposé un sujet pour mon épreuve TEST (Travail d'Etude et de Synthèse Technique) UEENG 111, sur les « *capacités cartographiques autour des technologies SOLAP* ». STEAMER m'a ensuite proposé successivement deux contrats sur deux projets de l'équipe : GenGHIS, puis *ESPON Database*. Le projet *ESPON HyperAtlas Update* comme sujet de mémoire m'a permis de diversifier encore mes connaissances sur une des thématiques de l'équipe, la visualisation cartographique. L'envergure européenne de diffusion de nos travaux fut également un élément non négligeable de motivation.

Avec beaucoup de plaisir, je côtoie ainsi depuis deux ans le monde de la recherche. D'autant plus qu'entre informatique et géographie, ce domaine d'application représente à mes yeux un terrain d'exploration riche, vaste, et passionnant. L'interdisciplinarité du groupe de recherche HyperCarte et les confrontations d'idées, de vocabulaire, de connaissances, notamment avec les géographes de RIATE, ont constitué un ensemble d'éléments motivant et moteur pour la réalisation du projet.

Une invitation à l'école d'été du groupement de recherche MAGIS (Méthodes et Applications pour la Géomatique et l'Information Spatiale), organisée par Madame Paule Annick Davoine, m'avait, entre temps, définitivement convaincu d'un univers propice à un enrichissement magistral.

Côtoyer les enseignants chercheurs m'a également permis d'appréhender l'enseignement : j'ai pu dispenser une quinzaine d'heures de travaux pratiques auprès d'étudiants LP ESSIG (Licence Professionnelle d'Etudes des Statistiques et des Systèmes d'Information Géographique), sur les bases de données spatiales PostgreSQL PostGIS. Ce qui ce conçoit bien s'énonce bien, la pratique de cette activité requiert d'approfondir ses connaissances, de les clarifier afin d'en retenir l'essentiel pour les transmettre au mieux. Humainement et techniquement enrichissante, j'aspire à renouveler l'expérience.

Quant à l'aspect informatique, le stage m'a notamment donné l'occasion de porter une attention particulière à la modélisation. L'extensibilité et la modularité ont été deux objectifs majeurs de la démarche suivie.

Le développement de l'interface utilisateur de l'application Web a été l'occasion de découvrir quelques aspects de la librairie JQuery, d'approfondir mes connaissances sur l'usage des feuilles de style CSS (*Cascading Style Sheets*).

En Java, j'ai également pu découvrir des API comme JTS (*Java Topology Suite*), ou le dessin vectoriel à l'aide de l'API Graphics2D de Java.

Ce stage m'a permis de pratiquer régulièrement l'anglais, écrit, et occasionnellement oral, lors de démonstrations à des chercheurs anglophones invités par STEAMER. Dans le cadre de présentations régulières du travail effectué, ce stage a donc été propice à exercer ma communication.

Enfin, j'ai profité de la rédaction de ce mémoire pour découvrir l'univers LaTeX. L'apprentissage de ce langage et l'achèvement d'un document d'une centaine de pages renforcent mes compétences et ma confiance pour, je l'espère, à l'avenir, répondre efficacement à l'opportunité, sinon à l'envie, de nouvelles productions écrites.

Annexe A

Docbench

Le projet LIGforge `docbench`[1] vise à fournir un environnement multiplate-forme pour la génération de documents HTML et PDF à partir de sources XML validant la DTD (*Document Type Definition*) standard DocBook version 4.5 CR3. Basé sur Ant et Java, ce projet permet de disposer de l'environnement complet permettant l'écriture et le partage de documents au sein d'une équipe.

L'intérêt de DocBook étant de se concentrer sur le fond sans se soucier de la forme, le projet `docbench` permet à l'équipe STEAMER de produire des documentations techniques qui respectent toutes la même charte graphique.

Le coeur du projet `docbench` est archivé sur le SCM Subversion de LIGforge suivant la hiérarchie de fichiers et répertoires suivante :

```
docbench/
|- build.xml
|- build.properties
|- lib/
|  |- DTD DocBook, *.jar
|- src/
|  |- common/
|  |  |- doc.css
|  |- images/
|  |  |- *.gif, jpg, png, svg
```

Les fichiers `build.xml` et `build.properties` fournissent le script Ant et ses paramètres pour la construction automatique.

Le répertoire `lib` fournit la DTD DocBook référencée depuis les sources et les librairies Java appelées pour la réalisation de quelques tâches Ant (récupération des sources depuis le serveur Subversion, par exemple).

Le répertoire `src` est composé de sous-répertoires, un par documentation. Le répertoire `src/common/` fournit des fichiers partagés, notamment la feuille de style `doc.css` permettant aux versions HTML des documentations générées d'adopter un style commun et cohérent. Toutes les images sont centralisées dans le répertoire `src/images`.

Le répertoire `src` de la hiérarchie de répertoires précédente est enrichi avec les documentations relatives au projet du groupe de recherche HyperCarte décrites section 9.5.1 page 62 :

1. *LiGforge docbench : Java Workbench for DocBook Documentation*. [en ligne - accès réservé] https://ligforge.imag.fr/projects/docbench/ (consulté le 10 octobre 2010).

```
docbench/
|- src/
|  |- hyper/
|  |  |- *.xml
|  |- hypercarte_userManual/
|  |  |- *.xml
|  |- hypercarte_adminGuide/
|  |  |- *.xml
|  |- ESPON_HyperCarte_userManual/
|  |  |- *.xml
|  |- ESPON_HyperCarte_adminGuide/
|  |  |- *.xml
|  |- (...)
```

Le format texte XML est adapté à la sauvegarde sur le SCM. La hiérarchie de répertoires permet de partager des chapitres ou sections entre les différentes documentations : les instructions pour l'installation de la base de données sont par exemple disponibles dans le guide du développeur et le document d'installation livré au client.

En simple prérequis, l'utilisateur doit installer Java et Ant sur son poste. A partir du script de construction automatique `build.xml` et `build.properties`, il/elle peut ensuite récupérer toute la hiérarchie de répertoires du projet. Il/elle peut alors générer sur son poste la documentation souhaitée au format voulu, HTML (en tronçons ou en sur une page unique) et PDF, par exemple.

Pour générer une documentation, le script Ant est conçu de façon à simplifier la vie du développeur. Il lui suffit, en effet, de taper le nom identifiant la documentation à construire et éventuellement le format de sortie désiré. Par défaut, les trois formats HTML PDF et HTML sur une seule page sont construits. Exemple pour construire le manuel utilisateur :

```
ant -Ddoc_name=hypercarte_userManual
```

La commande précédente va générer le répertoire `dist` suivant sur le disque de l'utilisateur :

```
docbench/
|  |- dist/
|  |  |- hypercarte_userManual/
|  |  |  |- hypercarte_userManual.pdf
|  |  |  |- hypercarte_userManual_html/
|  |  |  |  |- *.html
|  |  |  |- hypercarte_userManual_onehtml/
|  |  |  |  |- index.html
|  |  |  |- images/
```

En phase d'écriture, l'utilisateur dispose des cibles Ant `valid`, `html`, `pdf`, pour vérifier chacune des étapes de la construction.

La distribution complète des documentations relatives au projet du groupe de recherche HyperCarte est mise à jour et incorporée automatiquement lors de la génération de l'application Web. Pour pouvoir reproduire une distribution, cette mise à jour automatique reconstruit les documents à partir d'un export des sources depuis Subversion à la révision souhaitée (dernière révision, par défaut).

Annexe B

Schéma de la base de données

Les figures suivantes ont été exportées depuis le logiciel MicroOLAP Database Designer for PostgreSQL Version 1.2.11. Le logiciel permet d'éditer un schéma physique d'une base existante à travers un menu « rétro-ingénierie ».

Pour plus de lisibilité, le schéma de la base est scindé suivant deux figures :

- la figure B.1 montre le modèle mis en œuvre pour internationaliser les noms et descriptions des différentes entités ;
- la figure B.2 montre les relations entre les entités du modèle.

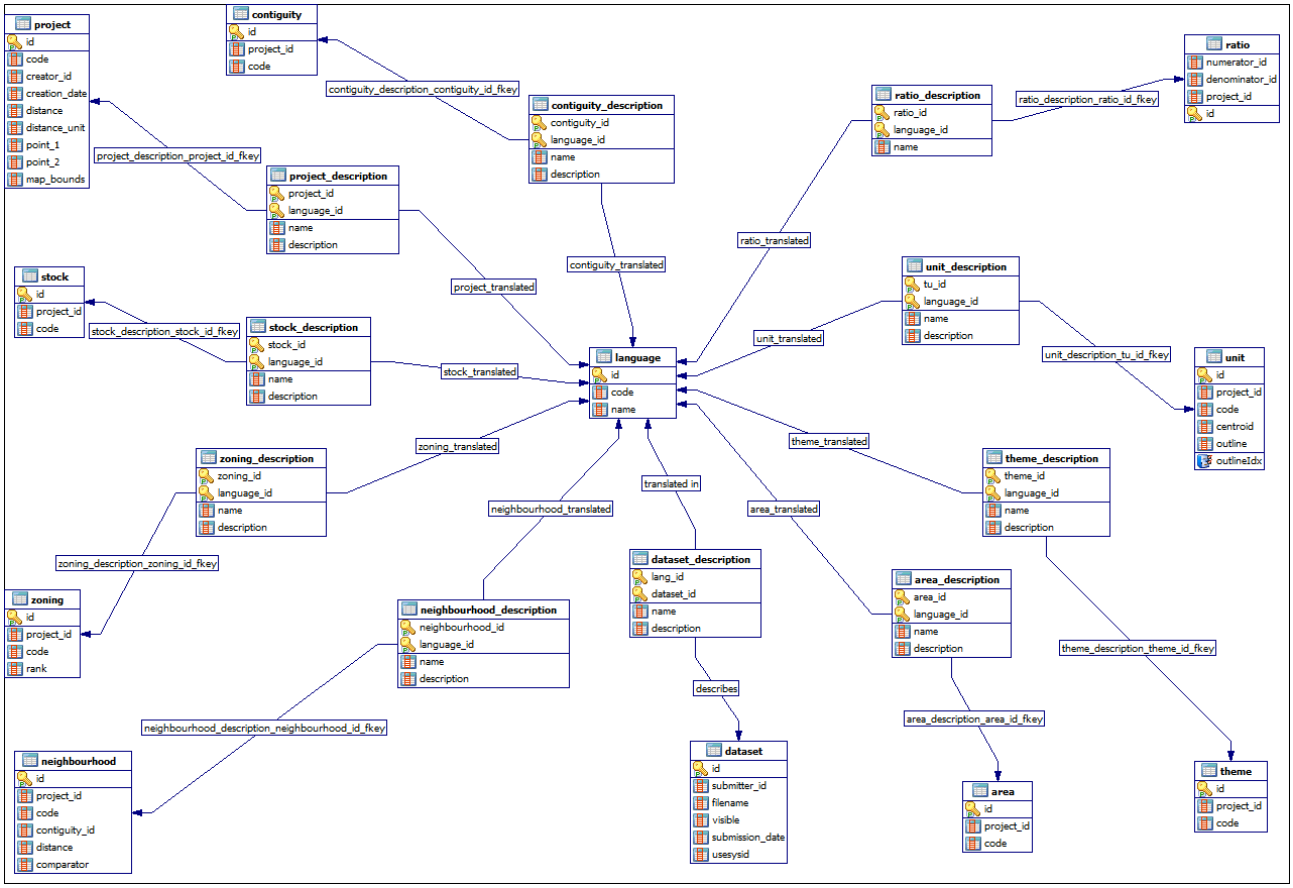


FIGURE B.1 – Extrait du modèle physique de la base de données `hypercarte` pour l'internationalisation des noms et descriptions des entités.

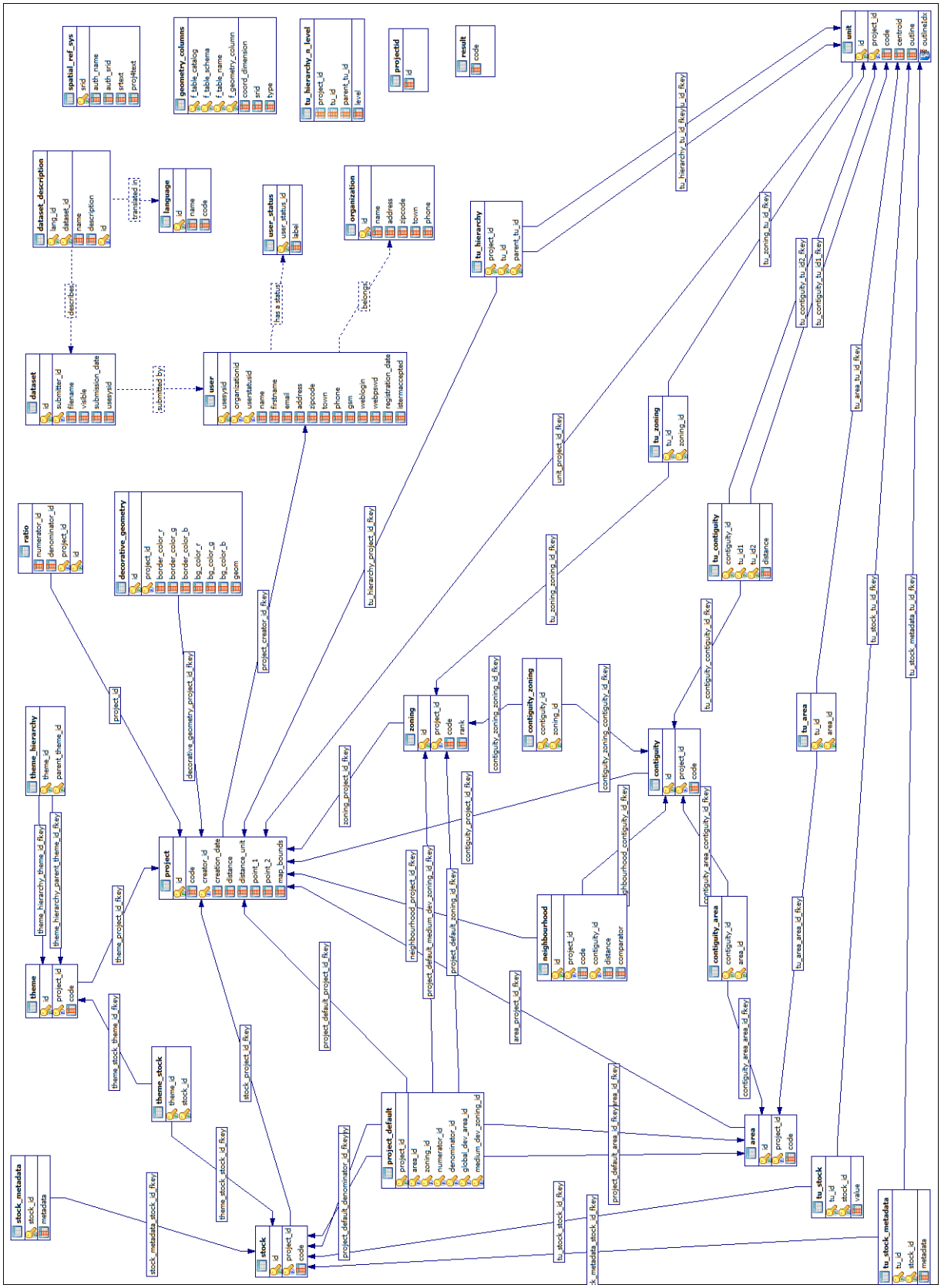


FIGURE B.2 – Extrait du modèle physique de la base de données `hypercarte` : relations entre les entités.

Annexe C

Glossaire

Carte à cercles proportionnels Représente des quantités ou des effectifs par un disque dont la surface est proportionnelle à ces quantités ou effectifs, à l'exclusion des pourcentages, taux, mesures ou catégories. Le maillage est ponctuel : un point caractéristique de chaque polygone accueille le centre. Les cercles sont préalablement calibrés afin que le plus grand cercle n'occupe pas une surface trop importante et que le plus petit reste visible. Les cercles sont remplis par une couleur unique ou vide. La légende est composée d'un échantillon de cercles qui correspondent aux valeurs présentes sur la carte. L'inconvénient principal est le chevauchement possible des disques pouvant géner la lecture.

Choroplèthe Du grec *khore* évoquant l'espace et de *plethos* qui exprime la multitiude, la carte choroplèthe représente la variation d'une variable continue ou discrète sur un maillage surfacique, à l'exclusion des quantités ou des effectifs. Les variables continues sont discrétisées afin d'affecter une couleur à chaque classe (une dizaine maximum). Chaque polygone composant le maillage est colorié afin de traduire graphiquement la variation géographique de la variable représentée. La légende se compose de caissons colorés permettant de qualifier chaque couleur par un intervalle de valeurs ou une modalité. L'inconvénient principal réside dans le risque d'une mauvaise interprétation dans le cas de grands polygones [Wan10].

Contiguïté Deux zones géographiques sont contiguës lorsqu'elles partagent une frontière commune [Jay93].

Ecart L'écart relatif d'une unité à un contexte de référence considère le rapport entre le ratio de deux indicateurs statistiques pour cette unité sur un ratio considérant ces deux mêmes indicateurs pour toutes les unités de l'aire d'étude du contexte de référence choisi.

$$Ecart_u^{Ref} = 100 \cdot \frac{\frac{Indicateur1_u}{Indicateur2_u}}{\frac{\sum_j Ref_{u,j} \cdot Indicateur1_j}{\sum_j Ref_{u,j} \cdot Indicateur2_j}} \tag{C.1}$$

EPSG 3035 L'EPSG (*European Petroleum Survey Group*) définit depuis 1985 une liste des systèmes de coordonnées géoréférencées identifiés par leur code (Wikipédia). Le 3035 correspond à la projection ETRS-LAEA, aussi référencée comme ETRS 89 par l'OGC (*Open Geographical Consortium*), système standard européen actuel conservant les surfaces, notamment utilisé par l'EEA (l'Agence Européenne de l'Environnement) et dans la cartographie de statistiques, connu également comme le système de coordonnées géoréférencées Lambert azimuthal à surfaces égales, il prend comme origine la latitude 52° Nord et la longitude 10° Est (près de Hannovre en Allemagne) [ZLR10].

.hyp Extension utilisée pour le nom du fichier binaire contenant un jeu de données géographiques et statistiques généré par l'application HyperAdmin et manipulable par le logiciel HyperAtlas. On utilise le terme *.hyp* pour désigner un jeu de données.

JavaBean Un composant JavaBean est une simple classe Java respectant des conventions de nommage pour ses méthodes accesseurs et mutateurs de ses attributs privés. Les composants JavaBeans sont typiquement utilisés dans les applications Web pour la soumission de formulaires.

LAU (*Local Administrative Unit*) Les deux niveaux d'unités administratives locales LAU 1 et LAU 2 correspondent respectivement aux anciens niveaux NUTS 4 et NUTS 5. Le niveau LAU 1 est défini pour certains pays et correspond à des unités d'envergure locale, comme les *districts* anglais, le niveau LAU 2 correspond à des unités équivalentes aux communes.

Maillage Malgré l'ambiguïté du terme [Gra97b], le maillage est considéré dans ce document comme une organisation hiérarchique du territoire, Le maillage est constitué de maillages imbriqués à partir de mailles élémentaires. Il s'agit donc d'une partition de l'espace constituant une échelle d'observation.

Métadonnées Du préfixe grec *meta* indiquant l'auto-référence, les métadonnées sont des données qui décrivent d'autres données. Les métadonnées décrivent à la fois le contenu de l'information, sa fiabilité et sa disponibilité, et permettent d'établir la qualité des données au sens le plus large du terme [SLL06].

NUTS (Nomenclature d'Unités Territoriales Statistiques) Norme statistique créée par Eurostat, office de statistique de l'Union Européenne. Une hiérarchie à plusieurs niveaux de NUTS facilite la comparaison des unités et l'étude de la répartition territoriale : NUTS 0 (pays), NUTS 1 (grandes régions), NUTS 2 (régions) et NUTS 3 (départements). Les niveaux inférieurs NUTS 4 et NUTS 5 ont été renommés LAU 1 et LAU 2 (*Local Administrative Unit*) [RIA07].

UAL (Unité Administrative Locale) : voir l'acronyme anglais **LAU**.

WUTS Equivalent au NUTS pour le niveau mondial, acronyme créé par Claude Grasland pour *World Unit Territorial Statistics*. Le WUTS niveau 5 correspond au pays.

Bibliographie

[Agi06] AgileSwiss.org. Accueil. [En ligne] http://www.agile-swiss.org/wiki/index.php/Accueil (consulté le 10 octobre 2010), 2006.

[Aou03] Djamel Aouane. Cours de génie logiciel. DEST CNAM Grenoble, 2003.

[Apa09] Software Foundation Apache. Struts 2 welcome. [en ligne] http://struts.apache.org/2.x/index.html (consulté le 16 décembre 2010), 2009.

[Bad10] Thierry Badard. Open source geospatial business intelligence in action with geomondrian and solaplayers. [en ligne] http://www.slideshare.net/tbadard/ (consulté le 10 janvier 2011), Octobre 2010.

[Ber67] Jacques Bertin. *Sémiologie graphique*. Paris Mouton, 1967.

[BRP06] Yvan Bédard, Sonia Rivest, and Marie-josée Proulx. Spatial on-line analytical processing (solap) : Concepts, architectures, and solutions from a geomatics engineering perspective. In *Data Warehouses and OLAP : Concepts, Architecture, tools and remaining challenges*. Press, 2006.

[Cha07] Christophe Chabert. Hyperatlas et hyperadmin : des outils cartographiques pour l'analyse de phénomènes sociaux. Mémoire ingénieur CNAM, mars 2007.

[Cue05] Olivier Cuenot. Modélisation spatiale multiscalaire de phénomènes sociaux. Mémoire ingénieur CNAM, mars 2005.

[DJP+09] Gregory Derek, Ron Johnston, Geraldine Pratt, Michael Watts, and Sarah Whatmore. *The Dictionary of Human Geography*. Wiley-Blackwel 5th Edition, April 2009.

[GAA04] Peter Gatalsky, Natalia Andrienko, and Gennady Adrienko. Interactive analysis of event data using space-time cube. [en ligne] http://geoanalytics.net/and/papers/iv04.pdf (consulté le 7 décembre 2010), 2004.

[GHJV95] Erich Gamma, Richard Helm, Ralph E. Johnson, and John Vlissides. *Design Patterns : Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA, 1995.

[GL04a] Claude Grasland and Liliane Lizzi. *ESPON Project 3.1 Integrated Tools for European Spatial Development.Annex A Multiscalar Territorial Analysis 4th. version*. CNRS UMS RIATE UMR GEOGRAPHIE-CITES, 2004.

[GL04b] Claude Grasland and Liliane Lizzi. Third interim report espon project 3.1 integrated tools for european spatial development. annex a : Multiscalar territorial analysis. [en ligne] http://www.ums-riate.fr/documents/MTA_4th%20version-livret2.pdf (consulté le 2 décembre 2010), 2004.

[GMV+05] Claude Grasland, Hervé Martin, Jean-Marc Vincent, Jérôme Gensel, Hélène Mathian, Saïd Oulahal, Olivier Cuenot, Euloge Edi, and Liliane Lizzi. Le projet hypercarte : analyse spatiale et cartographie interactive. In *SAGEO*, juin 2005.

[Gra97a] Claude Grasland. Go251_2 : Indices de concentration. http://grasland.script.univ-paris-diderot.fr/go251/go251_2/go251_2.html (consulté le 28 mai 2010), 1997.

[Gra97b] Claude Grasland. Un cadre théorique et méthodologique pour l'étude des maillages territoriaux. [en ligne] http://census.web.ined.fr/debat/Contributions/Avant-Fevrier-1999/Grasland-2.html (consulté le 17 mai 2010), 1997.

[Hä70] Torsten Hägerstrand. What about people in Regional Science ? *Papers in Regional Science*, 24(1) :6–21, December 1970.

[Jay93] Hubert Jayet. *Analyse spatiale quantitative, une introduction*. Paris Economica, 1993.

[JBL09] Mikael Jern, Monica Brezzi, and Thygesen Lars. A web-enabled goevisual analytics tool applied to oecd regional data. [En ligne] http://www.oecd.org/dataoecd/29/40/42711835.pdf (consulté le 10 octobre 2010), 2009.

[Kra03] M.J. Kraak. The space-time cube revisited from a geovisualization perspective. [en ligne] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.102.9231&rep=rep1&type=pdf (consulté le 10 décembre 2010), 2003.

[LG00] Julie Le Gallo. Econométrie spatiale, autocorrélation spatiale. [en ligne] http://www.univ-lille1.fr/bustl-grisemine/pdf/rapports/G2000-152.pdf (consulté le 15 novembre 2010), 2000.

[LR09] Benoit Le Rubrus. Capacité cartographique autour des technologies solap. Epreuve TEST CNAM UE ENG 111, Juin 2009.

[Lup10] Bernard Lupin. Osez olap : les bases de données olap par l'exemple. [en ligne] http://bernard.lupin.pagesperso-orange.fr/index.htm, 2010.

[Mar04] Philippe Martin. Interface cartographique pour l'analyse territoriale multiscalaire de phénomènes sociaux. Mémoire ingénieur CNAM, décembre 2004.

[MSJ06] Hélène Mathian and Thérèse Saint-Julien. Statistiques spatiales : analyse d'un semis de points et autocorrélation spatiale. [en ligne] http://www.ums-riate.fr/ecoleyaounde2006/documents/fascicules/semis.pdf (consulté le 15 novembre 2010), 2006.

[NCV10] Linköping University NCVA. exp : ncva : Linköping university. [En ligne] http://ncva.itn.liu.se/explorer?l=en (consulté le 10 octobre 2010), 2010.

[OCD09] OCDE. Statistiques et indicateurs régionaux. [En ligne] http://www.oecd.org/gov/regional/statistiquesindicateurs (consulté le 10 octobre 2010), 2009.

[OEC09a] OECD. Oecd explorer : interactive maps for regional statistics. [en ligne] http://www.oecd.org/gov/regional/statisticsindicators/explorer (consulté le 7 septembre 2010), 2009.

[OEC09b] OECD. What does oecd explorer enable you to do ? an introduction to its main features. [En ligne] http://www.oecd.org/dataoecd/55/47/44084514.pdf (consulté le 10 octobre 2010), 2009.

[OEC09c] Regional Statistics OECD. Démonstration de l'outil oecd explorer. [En ligne] http://stats.oecd.org/OECDregionalstatistics/ (consulté le 10 octobre 2010), 2009.

[Peu94] DJ Peuquet. It's about time : a conceptual framework for the representation of temporal dynamics in geographic information systems. *Anns of the Association of American Geographers*, page 441, 1994.

[PGVO+09] Christine Plumejeaud, Jérôme Gensel, Marlène Villanova-Oliver, Maher Ben Rebah, and Guillaume Vergnaud. Modélisation des hiérarchies territoriale multiples vers la gestion d'informations spatio-temporelles évolutives. In *SAGEO*, 2009.

[Plu07] Christine Plumejeaud. Acquisition de données et cartes de potentiel pour l'analyse spatiale. Mémoire ingénieur CNAM, juin 2007.

[Poi08] Audrey Poissonnier. Rapport de master 2 institut de géographie alpine de grenoble. 2008.

[PVOG10] Christine Plumejeaud, Marlène Villanova-Oliver, and Jérôme Gensel. Opérationnalisation d'un profil iso 19115 pour des métadonnées socio-économiques. [en ligne] http://hypercarte.imag.fr/doc/publis/Metadonnees_Socio-Economiques-v11.pdf (consulté le 10 janvier 2011), 2010.

[RIA07] UMS RIATE. Lexique de l'aménagement du territoire européen. [en ligne] http://www.ums-riate.fr/lexique/modeleterme.php?id=28 (consulté le 17 mai 2010), 2007.

[Rou06] Ian Roughley. Starting struts 2 productivity and integration strategies. [en ligne] http://www.infoq.com/minibooks/starting-struts2 (consulté le 10 janvier 2011), 2006.

[RRRR10] Ola Rosling, Anna Rosling Rönnlund, and Hans Rosling. Gapminder - unveiling the beauty of statistics for a fact based world view. [en ligne] http://www.gapminder.org/ (consulté le 7 septembre 2010), 2010.

[SLL06] Sylvie Servigne, Nicolas Lesage, and Thérèse Libourel. Spatial data quality components, standards and metadata, March 2006. Fundamentals of Spatial Data Quality. International Scientific and Technical Encyclopedia. ISBN 1905209568. Pp. 179-210.

[SM02] Inc Sun Microsystems. Designing enterprise applications with the j2ee platform, second edition. [en ligne] http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/DEA2eTOC.html (consulté le 12 novembre 2010), 2002.

[SM04] Inc Sun Microsystems. How to write doc comments for the javadoc tool. [en ligne] http://java.sun.com/j2se/javadoc/writingdoccomments/ (consulté le 3 juillet 2010), 2004.

[Sta05] Bob Stayton. *Docbook Xsl*. Sagehill Enterprises, City, 2005.

[Tho08] Raphaël Thomas. Evolutions d'outils dédiés à l'analyse territoriale et à l'analyse spatiale dans le cadre du projet hypercarte. Mémoire ingénieur CNAM, juin 2008.

[Ver07] Thierry Verdel. Décisions et prévisions statistiques. [en ligne] http://tice.inpl-nancy.fr/modules/unit-stat/chapitre7/Chapitre7.pdf (consulté le 15 novembre 2010), 2007.

[Viz10] Vizzuality. Vizzuality - envisionning life. [en ligne] http://www.vizzuality.com (consulté le 7 septembre 2010), 2010.

[Wal99] Norman Walsh. *Docbook : the Definitive Guide*. O'Reilly, Sebastopol, 1999.

[Wan10] Philippe Waniez. Cartographie thématique et analyse des données avec philcarto 5.xx pour windows. http://philcarto.free.fr/Logiciels/DOCdeGRANITn1.zip (consulté le 9 juillet 2010), 2010.

[Wel09] Don Wells. Extreme programming : a gentel presentation. [en ligne] http://www.extremeprogramming.org/ (consulté le 28 octobre 2010), 2009.

[ZLR10] Christine Zanin, Nicolas Lambert, and Ysebaert Ronan. *Mapping Guide Cartography for ESPON Projects*. Technical Report ESPON 2013 Database, 2010.

A propos

Ce document a été généré le 6 mars 2011, à 14:05, à partir de la révision svn rev 163 du projet `bulex`[1].

La version électronique au format PDF et les sources de ce document sont disponibles sur le site personnel de l'auteur à l'adresse suivante : http://blerubrus.free.fr/cnam/memoire/.

En guise de crédits, trouvez ci-dessous une liste des principaux outils utilisés pour l'écriture de ce mémoire :

- compilateur pdflatex et éditeur de références JabRef[2] ;
- Ant[3] pour l'automatisation de génération des versions du document ;
- greffon TeXlipse[4] pour l'édition de documents `tex` sous l'IDE Eclipse[5] ;
- BOUML[6] pour la création des diagrammes UML ;
- Inkscape[7] pour la création d'images vectorielles et leur export au format PDF ;
- The Gimp[8] pour la capture d'écrans et les retouches d'images ;
- OpenOffice[9] pour l'édition de la couverture imposée par le CNAM et son export au format PDF.

1. LiGforge : BuLeX [en ligne - accès réservé] https://ligforge.imag.fr/projects/bulex/
2. *JabRef references manager* [en ligne] http://jabref.sourceforge.net/ (consulté le 2 décembre 2010).
3. *Apache Ant - Welcome* [en ligne] http://apache.ant.org (consulté le 23 octobre 2010).
4. *TeXlipse homepage : LaTeX for Eclipse* [en ligne] http://texlipse.sourceforge.net/ (consulté le 23 octobre 2010).
5. *Eclipse.org home* [en ligne] http://www.eclipse.org/ (consulté le 23 octobre 2010).
6. *BOUML - a free UML tool box.* [en ligne] http://bouml.free.fr/ (consulté le 23 octobre 2010).
7. *Inkscape. Draw Freely.* [en ligne] http://inkscape.org/ (consulté le 23 octobre 2010).
8. *The Gimp - The GNU Image Manipulation Program.* [en ligne] http://www.gimp.org/ (consulté le 23 octobre 2010).
9. fr : OpenOffice.org. [en ligne] http://fr.openoffice.org/ (consulté le 23 octobre 2010).

MEMOIRE D'INGENIEUR C.N.A.M. en INFORMATIQUE

Cartographie et analyse territoriale multiscalaire

Réingénierie des logiciels HyperAtlas et HyperAdmin

Benoit Le Rubrus **Grenoble, le 7 avril 2011**

Résumé

L'analyse territoriale multiscalaire repose sur l'hypothèse selon laquelle l'étude d'une unité territoriale n'a de sens que mise en perspective à différentes échelles géographiques, en la comparant aux territoires voisins et aux entités territoriales de plus ou moins grandes dimensions auxquelles elle appartient. Le groupe de recherche HyperCarte développe le logiciel HyperAtlas depuis dix ans. A partir de cartes interactives, les utilisateurs sont invités à réaliser de telles analyses multiscalaires. Plutôt destiné aux géographes et statisticiens, le logiciel HyperAdmin est, quant à lui, l'outil d'intégration de jeux de données visualisables par HyperAtlas.

Une version application de bureau d'HyperAtlas fut livrée au programme européen ESPON en 2007. Dans le cadre d'un nouveau programme, fin 2009, l'unité de coordination d'ESPON contracte avec le groupe de recherche HyperCarte un ensemble d'évolutions pour le logiciel, parmi lesquelles, notamment, le portage en version Web de l'application, la prise en compte de la dimension temporelle, de nouveaux outils statistiques et cartographiques (courbe de Lorenz, autocorrélation spatiale, cartes de redistribution et de synthèse).

A partir du cahier des charges de ce projet, ce document décrit l'étude, la conception, la démarche logicielle et le développement de ces nouvelles fonctionnalités, l'ensemble constituant une version 2 des logiciels HyperAtlas et HyperAdmin.

Mots-clés : géomatique ; cartographie ; SIG (Systèmes d'information géographique) ; analyse territoriale multiscalaire ; statistiques ; Java.

Abstract

Multiscalar Territorial Analysis is based on the assumption that it is not possible to evaluate the situation of a given territorial unit without taking into account its relative situation and localization. Based on this principle, the HyperAtlas application has been developed for ten years by the HyperCarte Research Group. From a set of user-friendly and fully interactive maps, the user is invited to evaluate indicators according to various territorial contexts, and to compare units with their neighbours. Besides, HyperAdmin is the integration tool that aims at generating datasets that can be visualized by HyperAtlas.

In 2007, the European ESPON Programme was delivered a previous version of HyperAtlas, as a desktop stand-alone application. At the end of 2009, the ESPON Coordination Unit requests to transform it into a Web-based application, and to include new functionalities, like the handling of time, the availability of advanced statistics and expert cartographic tools : Lorenz curve, spatial autocorrelation chart, equi-repartition and synthesis maps, for example.

Based on the call for tender of this project, this document describes the study, the design, the software engineering approach that has guided the implementation of these new functionnalities, upgrading this software towards a version 2 of HyperAtlas and HyperAdmin.

Keywords : Geomatic ; Cartography ; GIS (Geographical Information Systems) ; Multiscalar Territorial Analysis ; Statistics ; Java.