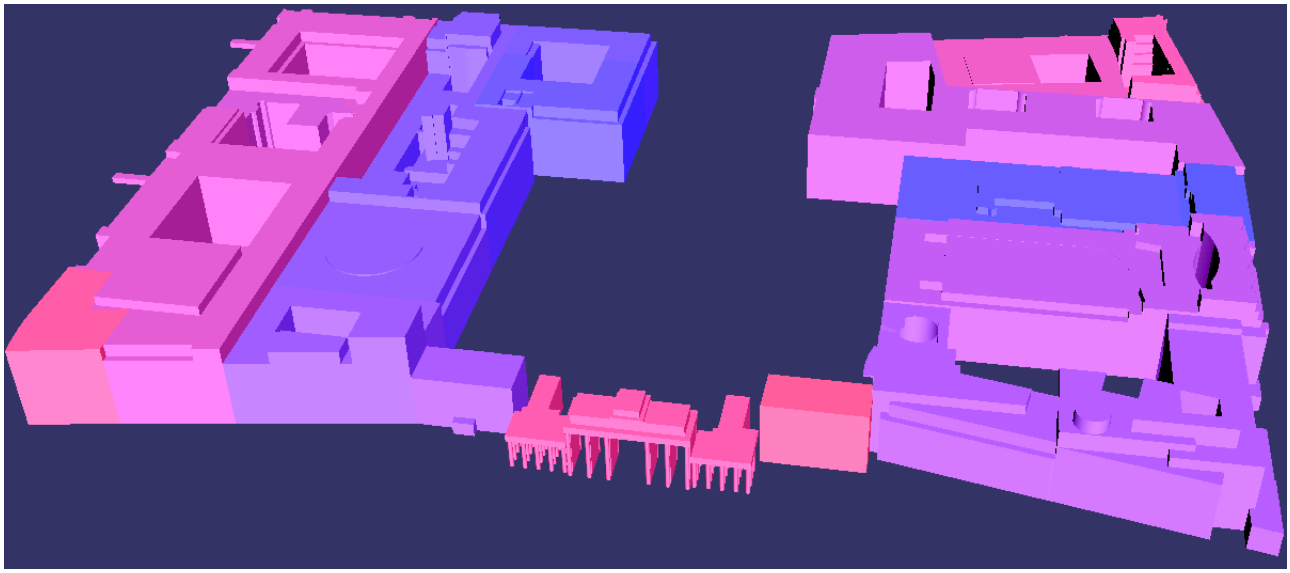
PGROU - VisDU3D

Rapport final

15/12/10



Alice Lan
Toinon Vigier
Elsa Arrou-Vignod
Florent Buisson
Robin Kervadec
Michael Lumbroso

Tuteur enseignant :
Guillaume Moreau

Table des matières

1 Présentation du projet Visdu3D

1.1 Objectifs et destinataires du document

Le projet est réalisé pour le CERMA, le Centre de Recherche Méthodologique d'Architecture, avec qui travaille en partenariat une équipe d'enseignants-chercheurs de l'Ecole Centrale de Nantes. Le projet Visdu3D a pour but de fournir un environnement de visualisation de données géographiques et des métadonnées (voir glossaire).

La problématique du projet est double : tout d'abord, permettre un affichage simultané de données géographiques 2D et 3D ; ensuite, proposer des solutions d'affichage des métadonnées de ces fichiers de façon ergonomique et lisible pour l'utilisateur. Le but du projet est de mener une réflexion sur des affichages adaptés au point de vue de la caméra, aux couleurs d'arrière-plan, à la taille des données, au type de données sémantiques représentées, etc.

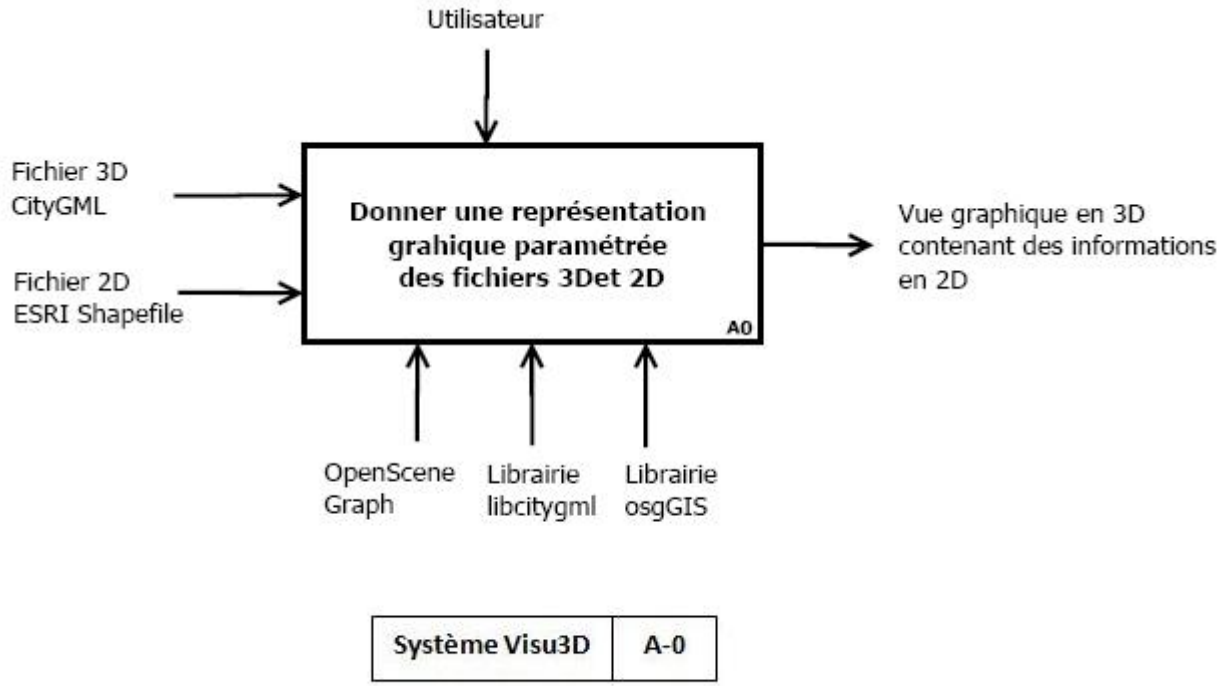


Fig. 1 – *Diagramme SADT au niveau 0 du projet Visdu3D.*

Les formats de fichiers à afficher sont des fichiers ESRI Shapefile pour la 2D, et CityGML pour la 3D. Une description plus détaillée de ces fichiers est disponible dans l'état de l'art.

1.2 Environnement du logiciel

Le développement du programme se fera en langage C++, en utilisant les systèmes d'exploitation Mac OS et Linux. Nous utiliserons Ubuntu pour les systèmes Windows. L'affichage en trois dimensions se fera à l'aide d'une bibliothèque C++ : Open Scene Graph. Le chargement des fichiers CityGML et ESRI Shapefile se fera respectivement grâce à deux bibliothèques : libcitygml et osgGIS.

1.3 Cahier des charges

Ce programme est destiné à être utilisé dans un système d'exploitation Linux ou Mac. Il pourra être complété et amélioré pour obtenir un plus large choix d'affichages. Il pourra être éventuellement repris pour créer une meilleure interface utilisateur.

Le cahier des charges que nous avons établi après réunion avec Guillaume Moreau, est le suivant :

- fournir un état de l'art de l'existant sur l'affichage de données 2D/3D et la sémiologie graphique
- afficher simultanément des fichiers CityGML (3D) et ESRI Shapefile (2D)
- proposer le plus possible de solutions pour afficher des métadonnées de façon ergonomique
- fournir des résultats de test et une analyse des solutions proposées

1.4 Livrables

2 Déroulement du projet

2.1 Plannings prévisionnel et effectué

Dates	01/10	15/10	05/11	19/11	03/12	15/12
Jalons	RA1	RA2	RA3	RA4	RA5	Rapport final
État de l'art						
Installations						
Affichage simultané						
Affichage d'informations						

Tab. 1 – *Planning prévisionnel de déroulement du projet. RA : rapport d'avancement.*

Dates	01/10	15/10	05/11	19/11	03/12	15/12
Jalons	RA1	RA2	RA3	RA4	RA5	Rapport final
État de l'art						
Installations						
Affichage simultané						
Affichage d'informations						

Tab. 2 – *Planning effectif du projet. RA : rapport d'avancement.*

2.2 Répartition du travail

2.2.1 Les différentes tâches du projet

Lancement du projet

Cette phase comprend la prise de connaissance du sujet avec Guillaume Moreau, la définition du cahier des charges et la familiarisation avec les bibliothèques que Guillaume Moreau nous suggérait d'utiliser.

État de l'art

Notre état de l'art comprend une description des formats de fichiers CityGML et Shapefile, une description des bibliothèques utilisées et les règles classiques de la sémiologie graphique.

Installations

Avant de démarrer la phase de développement du projet, il nous a fallu installer les différentes bibliothèques sur les machines sous Mac et sous Linux. Cette phase d'installations a été particulièrement laborieuse et longue.

2.2.2 Répartition des tâches entre les membres de l'équipe

Tâches	Alice	Toinon	Elsa	Florent	Robin	Michael
Lancement du projet	6	6	6	6	6	6
État de l'art	4	6	2			
Installations	4	20	10	30	6	4
Affichage simultané		10				
Affichage d'informations		15	20	25		
Rapports d'avancement	10 ?	5	5	5	5	
Rapport final	10 ?	5	13		10	>4
Total	34	67	56	61	37	12

Tab. 3 – *Nombre d'heures passées sur chaque partie du projet.*

3 Etat de l'art

3.1 Les formats CityGML et ESRI Shapefile

3.1.1 CityGML

Le format CityGML est un format de représentation de données urbaines qui dérive du format GML, lui-même dérivé du langage XML. Il est conçu pour permettre de stocker des métadonnées très variées. Ce type de fichier est optimisé pour représenter des données urbaines, et propose des représentations génériques déjà implantées dans le Schema pour les objets urbains les plus courants (routes, végétation, tunnels, ...).

Voici un exemple de fichier GML :

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- File: cambridge.xml -->
<CityModel xmlns="http://www.opengis.net/examples" ... >

 <gml:name>Cambridge</gml:name>
 <gml:boundedBy>
  <gml:Box srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
   <gml:coord><gml:X>0.0</gml:X><gml:Y>0.0</gml:Y></gml:coord>
   <gml:coord><gml:X>100.0</gml:X><gml:Y>100.0</gml:Y></gml:coord>
  </gml:Box>
 </gml:boundedBy>

 <cityMember>
  <River>
   <gml:description>The river that runs through Cambridge.</gml:description>
   <gml:name>Cam</gml:name>
   <gml:centerLineOf>
    <gml:LineString srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
     <gml:coord><gml:X>0</gml:X><gml:Y>50</gml:Y></gml:coord>
     <gml:coord><gml:X>70</gml:X><gml:Y>60</gml:Y></gml:coord>
     <gml:coord><gml:X>100</gml:X><gml:Y>50</gml:Y></gml:coord>
    </gml:LineString>
   </gml:centerLineOf>
  </River>
 </cityMember>
...
```

Fig. 2 – *Fichier gml*

Un fichier CityGML est composé, comme tous les fichiers XML, d'une racine (CityModel) et de noeuds fils (CityObjectMember) qui contiennent les informations géométriques et sémantiques sur les différents objets urbains.

Ce format permet de décrire des entités simples :
- Informations géographiques et descriptives
- Informations géométriques simples (points, lignes,...)

CityGML définit notamment plusieurs niveaux de détails (Level of Detail, LoD). Ces niveaux de détails sont étiquetés de 0 à 4. Voici un exemple de ce que l'on peut obtenir grâce à CityGML, en LoD 3 :

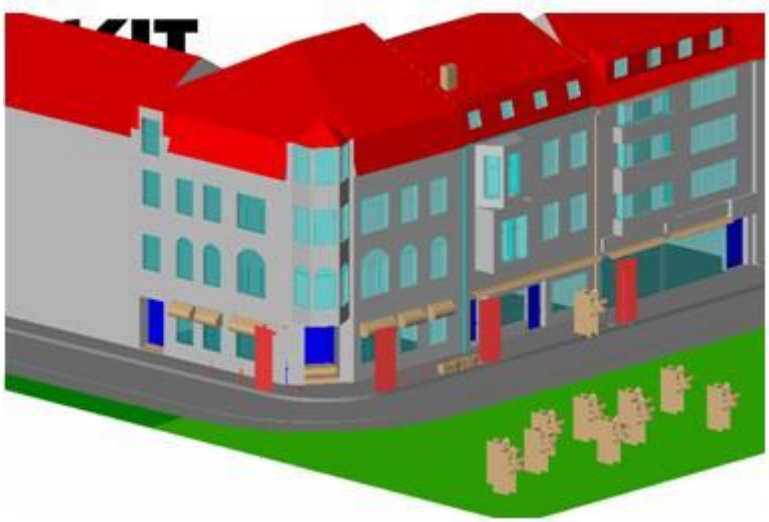


Fig. 3 – *Exemple de fichier CityGML en LoD 3*

[Arr10]

3.1.2 ESRI Shapefile

Le Shapefile est un format de fichier issu du milieu des Systèmes d'Information Géographiques (SIG). Il a été inventé et développé par ESRI (Environmental Systems Research Institute), une entreprise américaine qui développe et distribue des logiciels SIG. Le format de fichier Shapefile est cependant aujourd'hui utilisé par de nombreux logiciels libres.

Avantages du Shapefile :
C'est un fichier vectoriel, il est donc :
- plus facile à lire et à écrire
- moins gourmand en mémoire
- plus rapide à dessiner

qu'une image de type raster ou matriciel (les photos classiques par exemple).

Contenu :
Un fichier Shapefile contient toute l'information liée à la géométrie des objets décrits, par exemple :
- Points
- Multipoints
- Lignes
- Polygones

– ...

Description technique :

Un ESRI Shapefile est composé de : Un fichier principal Shapefile (.shp) Une dBase table (.dbf) qui contient les données attributaires relatives aux objets contenus dans le shapefile Un fichier index (.shx) qui stocke l'index de la géométrie Ces trois fichiers doivent avoir le même nom. D'autres fichiers peuvent également être fournis : .prj : contient des informations sur le système de coordonnées, utilise le format WKT

3.2 Les outils d'affichage 3D

Pour réaliser ce projet, nous avons eu besoin de bibliothèques pour afficher et manipuler des objets 3D, parser des fichiers CityGML et ESRI Shapefile et permettre d'exploiter les informations géométriques et sémantiques de ces fichiers.

3.2.1 Open Scene Graph

Open Scene Graph (OSG) est une bibliothèque graphique open-source permettant le développement d'applications graphiquement exigeantes, comme des simulateurs de vols, ou, ce qui va nous intéresser ici, de la réalité virtuelle.

Open Scene Graph est en fait une surcouche d'OpenGL , écrite en C++ et orientée objet. Elle fournit notamment des outils permettant de gérer la caméra, l'éclairage d'une scène...

OSG dispose d'un système de gestion de plugins dynamiques, ce qui nous a permis d'utiliser un loader 3D pour les fichiers CityGML : libcitygml.

3.2.2 libcitygml

3.2.3 osgGIS

osgGIS est une boîte à outils permettant le traitement de données d'informations géographiques dans OpenSceneGraph. Cette boîte à outil qui s'utilise directement en lignes de commande permet de visualiser des images géospatiales mais aussi de les manipuler et de faire de requêtes à travers des applications Open Scene Graph.

3.2.4 GDAL - OGR

GDAL (Geospatial Data Abstract Library) est une bibliothèque open source développée en langage C++ qui permet de traiter différents formats de stockage d'images géographique. Une sous-bibliothèque de GDAL est OGR qui permet de travailler en lecture et de manière plus minime en écriture sur les images de type vecteur. OGR supporte la plupart des formats courants (qu'ils soient porpriétaires ou non) de formes vectorielles dont le format ESRI Shapefile mais aussi Oracle Spatial, PostGIS ou encore MapInfo.

3.3 La sémiologie graphique

Il n'existe pas encore de norme pour la sémiologie graphique en 3D. Pour l'instant, l'affichage d'informations sémantiques se fait principalement par l'ajout de texte à côté des objets 3D. Nous nous sommes donc appuyés sur des règles d'affichage graphique en 2D pour la cartographie, que nous avons tenté d'appliquer à notre problème spécifique.

Tout d'abord, la sémiologie graphique distingue deux types de données : l'information qualitative (par exemple des différences de climat, la présence d'eau potable, ...), et quantitative (nombre d'habitants, taille d'une ville, pourcentage de femmes, ...). Voir le cours de Gwendall Petit : [Pet11].

Parmi les différentes techniques d'affichage d'informations sémantiques, nous pouvons notamment relever les marqueurs : ce sont des pointeurs qui indiquent les endroits d'information. Ils sont par exemple utilisés dans le service Ville en 3D proposé par pagejaunes.fr (`http://v3d.pagesjaunes.fr/`)

Fig. 4 – *Exemple de marqueurs qui indiquent les points d'informations.*

Ces marqueurs ne sont plus visibles une fois que l'on est trop proche de l'objet afin de ne pas gêner la visualisation. L'information est affichée lorsque la souris survole le marqueur.

Fig. 5 – *Information lors du survol d'un marqueur.*

4 La démarche appliquée au projet

Affichage de la géométrie et des informations des fichiers Shapefile
Mettre un diagramme de classe ou d'objet

Pour pouvoir afficher les informations des formats ESRI Shapefile, nous nous avons utilisé l'outil osgGIS en particulier le programme osggis_viewer que nous avons modifié afin de ne pas passer par le terminal. L'affichage n'est donc possible que si osgGIS a été installé et compilé sur la machine. Cette installation n'est pas forcément évidente, vous trouverez en Annexe un tutorial explicitant la marche à suivre.

L'affichage se fait grâce à la méthode transformShapefile qui retourne, à partir d'un fichier ESRI Shapefile, une pointeur de osg : :Node qu'il est ensuite possible de visualiser grâce à la méthode viewer() de OSG. Cette implémentation permet de visualiser sur la même fenêtre des fichiers Shapefile et des fichiers cityGML.

L'utilisation d'osgGIS permet un affichage rapide et simplifié des formats Shapefile. Néanmoins, les codes sources étant opaques et la documentation pas très enrichie, il est difficile de réutiliser les méthodes de traitement sur les informations du Shapefile. Nous avons donc décidé de créer nos propres méthodes en utilisant la bibliothèque GDAL, elle-même utilisée par osgGIS.

4.1 Diagrammes de classes

vspace-2cm

4.2 Interface d'utilisation du programme

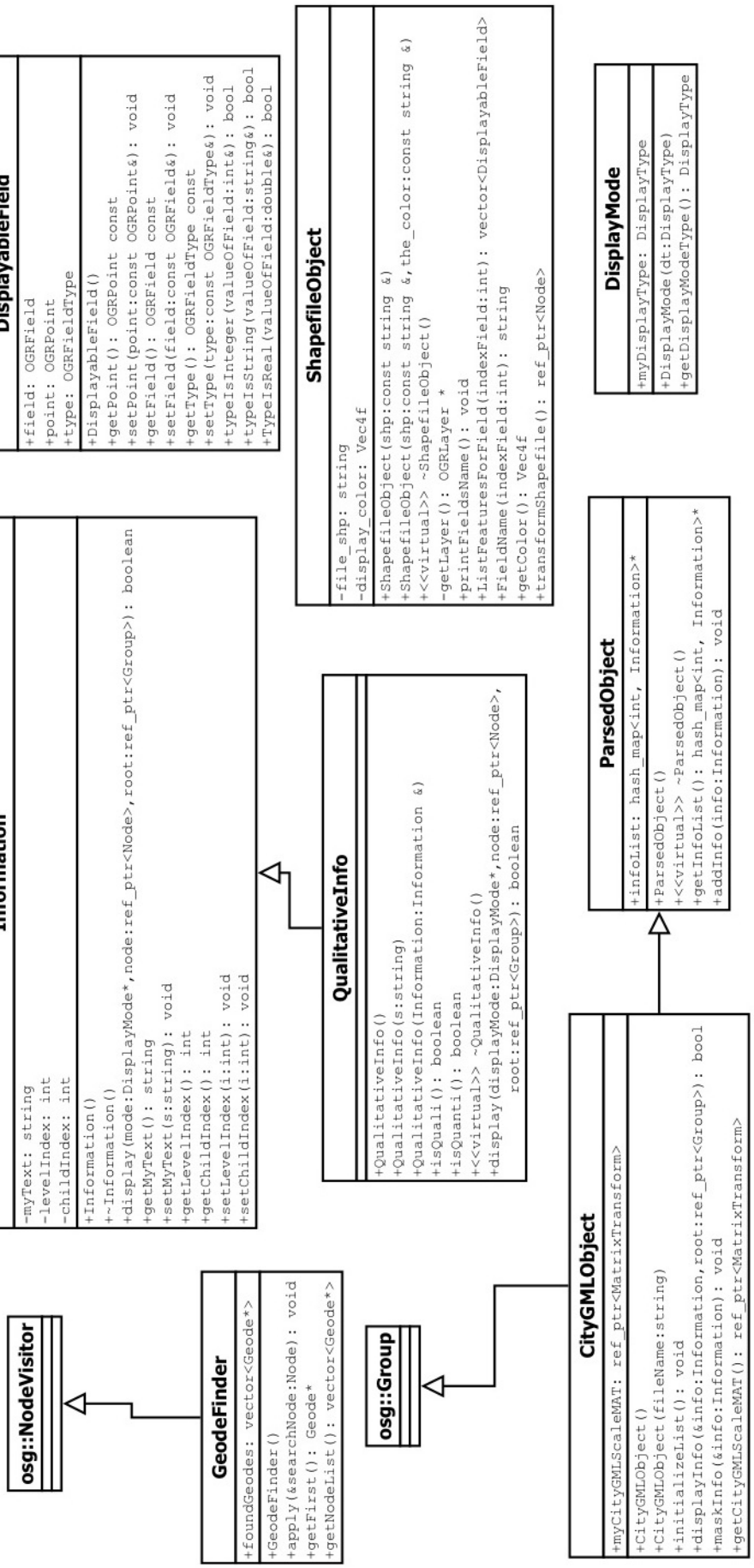


Fig. 6 – *Affichage d'une information quantitative sous forme d'un camaïeu de rouge et bleu.*

5 Résultats

5.1 Affichage simultané d'un fichier Shapefile et d'un fichier CityGML

L'affichage simultané des deux types de fichiers a finalement été possible, grâce à libcitygml et osgGIS, et changeant le mode de projection dans le cas où les fichiers ne sont pas géolocalisés dans le même système de coordonnées.

Cependant, nous n'avons pas réussi à installer osgGIS sur les machines Mac : de fait, l'affichage simultané est possible seulement sous Linux pour l'instant. De plus, l'installation a nécessité quelques manipulations : en effet, osgGIS est un projet qui n'a pas été mis à jour depuis 2008, et l'installation est complexe et source d'erreurs de compilation. Il nous a notamment fallu écrire un script pour modifier les fichiers sources d'osgGIS car depuis 2008 une classe d'Open Scene Graph avait changé de nom.

5.2 Récupération des métadonnées dans les fichiers

5.2.1 Dans les fichiers ESRI Shapefile

La récupération des métadonnées dans les fichiers Shapefile est possible. Les données recueillies se présentent sous la forme de coordonnées (celles du point qualifié par l'information) et de la valeur de l'information elle-même.

5.2.2 Dans les fichiers CityGML

Le site qui présente la bibliothèque libcitygml indique que celle-ci permet de parser un fichier CityGML et récupérer les données géométriques, et précise que les métadonnées du fichier sont également traitées et stockées dans les Node créés. Cependant, après avoir parcouru les messages log du site et examiné soigneusement les fichier C++ de cette bibliothèque, nous nous sommes rendu compte que la récupération des métadonnées n'était pas encore totalement implantée.

En effet, nous avons remarqué qu'il existe des variables prêtes à accueillir ce type de données, mais que nulle part ces métadonnées ne sont lues dans le fichier CityGML, ni transmises au Node principal que nous exploitons dans Open Scene Graph. Nous en avons conclu que cette fonctionnalité n'était pas encore totalement développée et qu'il nous était impossible de récupérer les métadonnées avec libcitygml.

Ayant constaté cela tard durant le projet, et un parser fait à la main étant long et difficile à mettre en place pour un format aussi riche que le CityGML, nous n'avons donc pas pu utiliser pour nos tests de métadonnées tirées de fichiers CityGML. Les informations affichées visibles sur les captures d'écran sont donc inventées pour les tests.

5.3 Affichage des métadonnées

Nous allons présenter ici des exemples d'affichage d'informations sémantiques sur des fichiers CityGML, Shapefile, et les deux formats simultanément. Nous avons fait le choix de présenter à la fois les exemples d'affichage qui semblent efficaces et ceux qui nous paraissent peu ergonomiques ou peu esthétiques.

Camaïeu de couleurs

Ici nous avons affiché une information quantitative (par exemple, la quantité de pertes thermiques par bâtiment) sous la forme d'un camaïeu de couleurs continu (absence de paliers). Il serait également envisageable de catégoriser les valeurs par paliers afin de mieux distinguer les différences de valeurs.

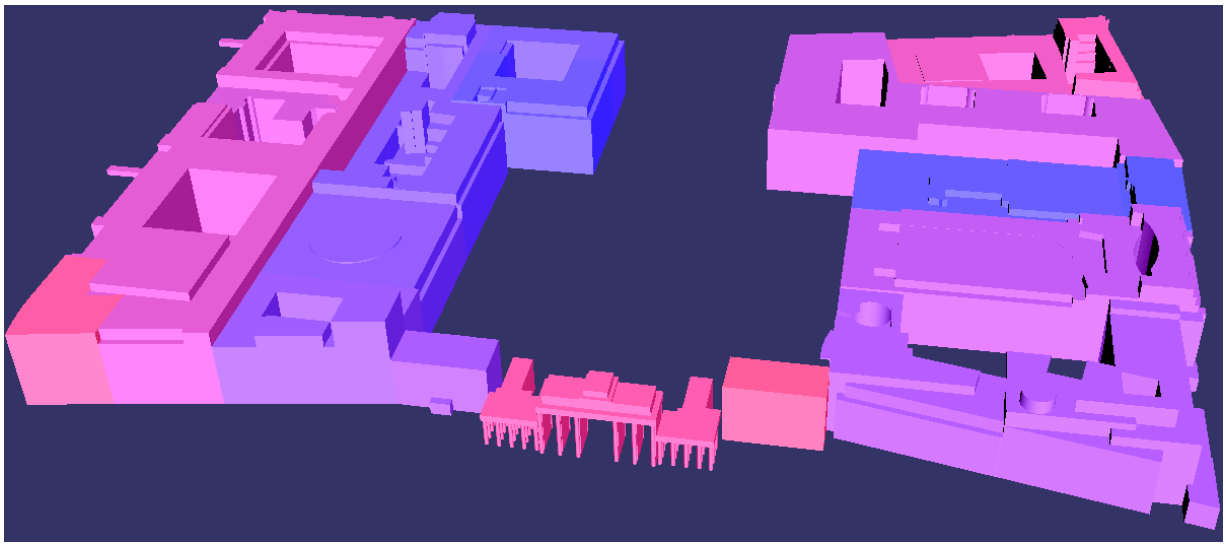


Fig. 7 – *Affichage d'une information quantitative sous forme d'un camaïeu de rouge et bleu.*

À noter qu'un affichage similaire est possible avec une variation de transparence, cependant le résultat est beaucoup moins appréciable et peu ergonomique :

Formes plus ou moins grandes

Une autre façon d'afficher des informations quantitatives est de placer des formes basiques dont la taille est l'élément qui va renseigner sur la valeur. Par exemple, pour deux bâtiments qui partagent un même type d'information, nous avons affiché deux cylindres de tailles différentes à la position des bâtiments qu'ils renseignent.

Les cylindres sont d'une couleur différente de celle des bâtiments pour pouvoir les distinguer et ne pas les confondre avec des informations de géométrie ; ils sont également transparents, afin de ne pas occulter complètement l'objet 3D de départ.

Remarque : cette capture d'écran ne permet pas de bien distinguer les bâtiments les uns des autres. Si le contexte d'utilisation ne le tolérait pas, une solution serait de

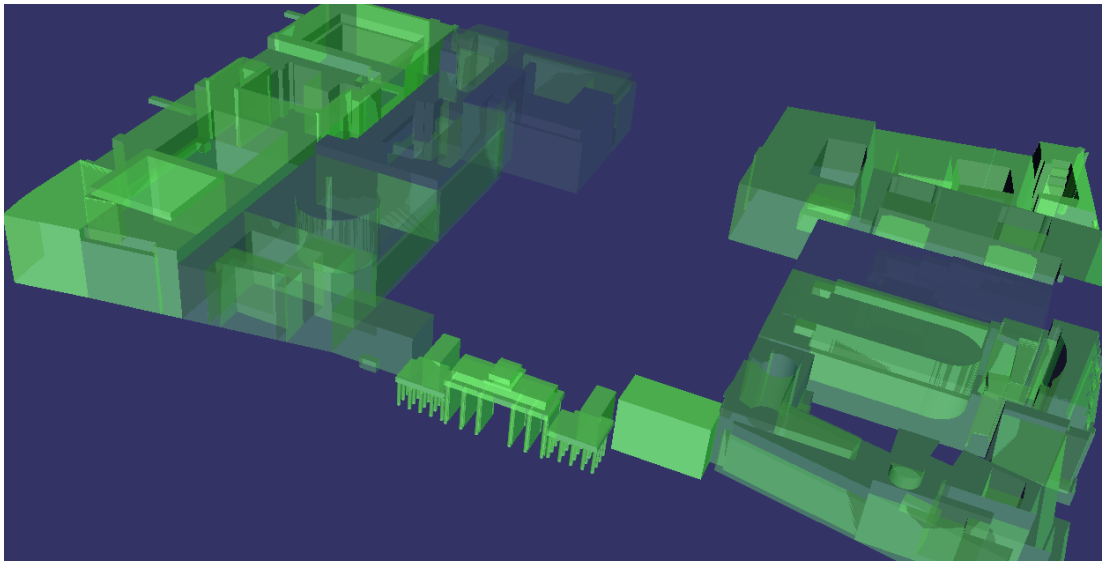


Fig. 8 – *Affichage d'une information quantitative sous forme d'une variation de transparence.*

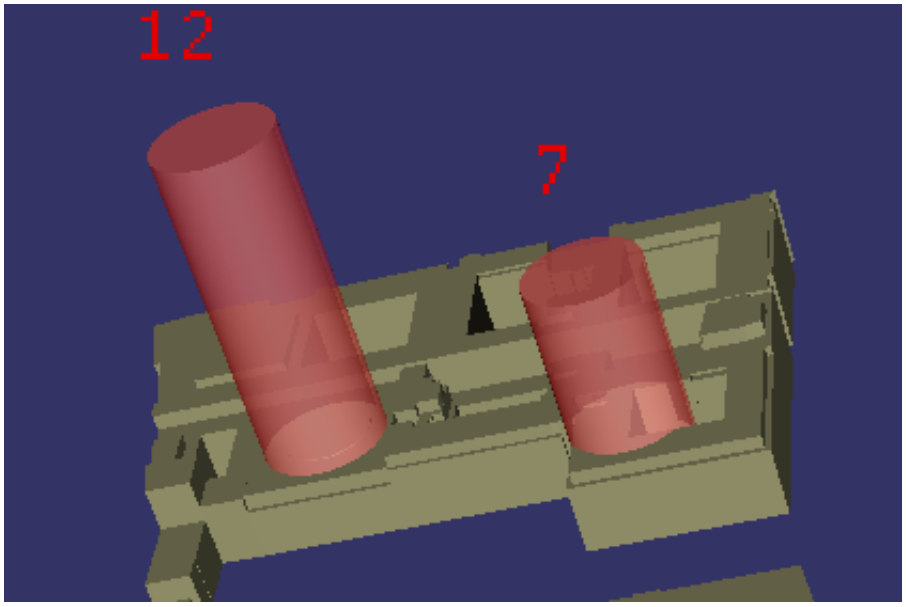


Fig. 9 – *Affichage d'une information quantitative sous forme de cylindres de tailles variables.*

mixer ce type d'affichage avec le camaïeu de couleurs présenté plus haut, où l'information sémantique serait donnée par les cylindres et le camaïeu servirait à optimiser l'ergonomie de l'affichage. On peut aussi imaginer une solution consistant à repérer les arêtes des bâtiments et les afficher de façon plus marquée.

Nous avons fait un autre essai avec des formes moins classiques que des cylindres, et qui conjugue variation de taille, affichage possible de texte ou dessin, de textures, et traits reliant l'objet signifiant à l'objet réel. Cela permettrait à un utilisateur averti d'Open Scene Graph de composer lui-même ses formes et de décider des éléments présents.

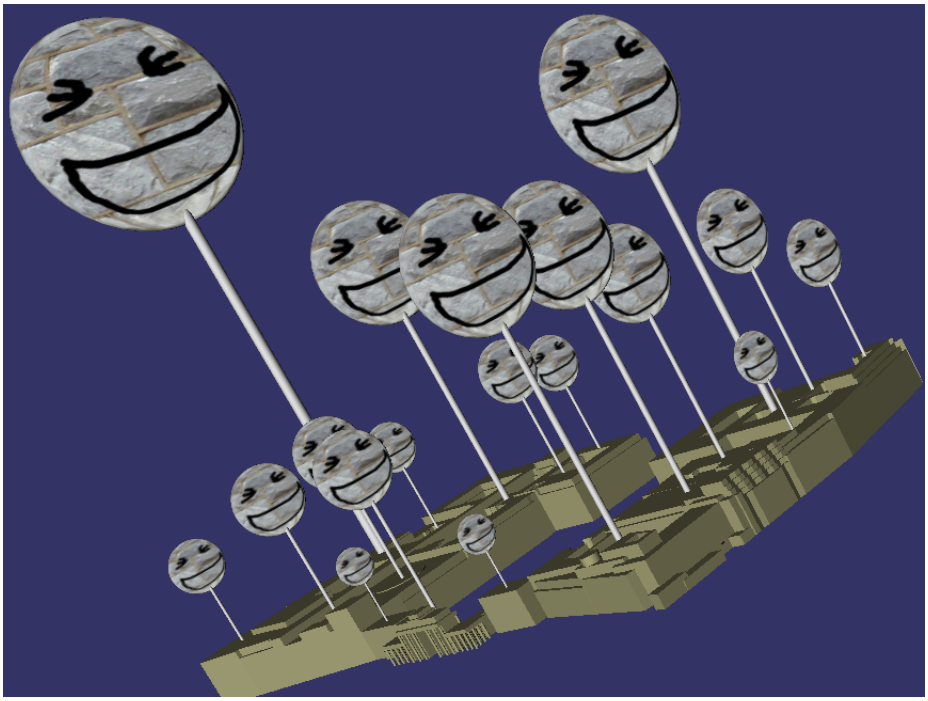


Fig. 10 – *Affichage d'une information quantitative avec des formes personnalisées conjuguant variation de taille, affichage d'un dessin, présence d'une texture.*

Distinction que quelques bâtiments parmi les autres

Pour signifier qu'un bâtiment a une caractéristique qui le distingue des autres, il faut qu'il soit distinctible à l'affichage également. Par exemple, on peut le marquer avec une couleur et/ou une transparence différente de celle des autres :

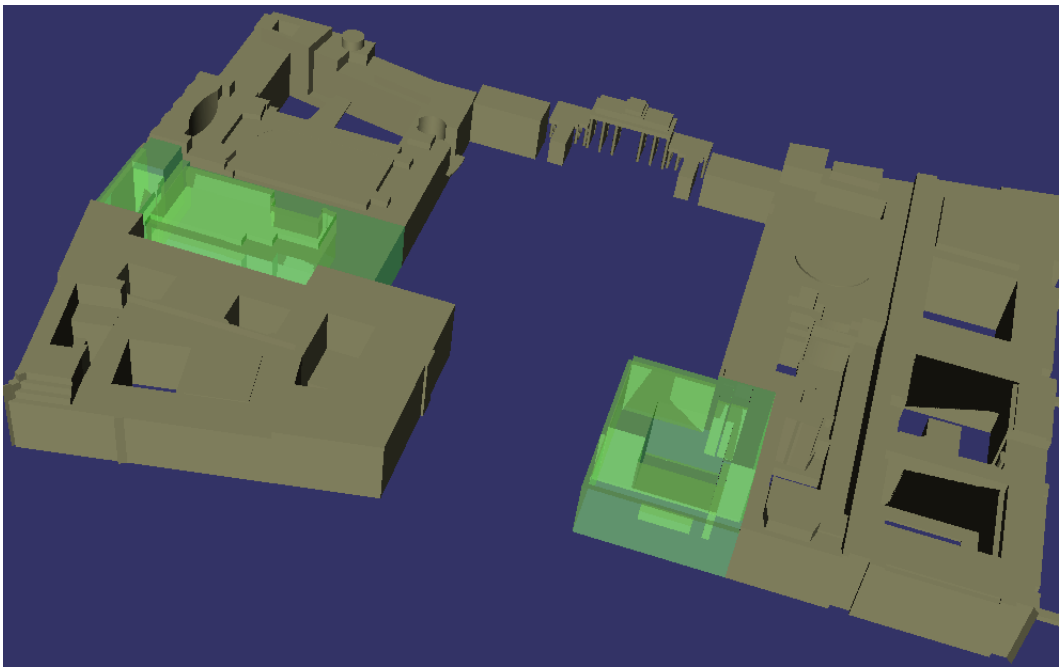


Fig. 11 – *Deux bâtiments se distinguent des autres par leur couleur et leur transparence.*

Nous avons également imaginé placer des formes au-dessus des bâtiments pour le signaler :

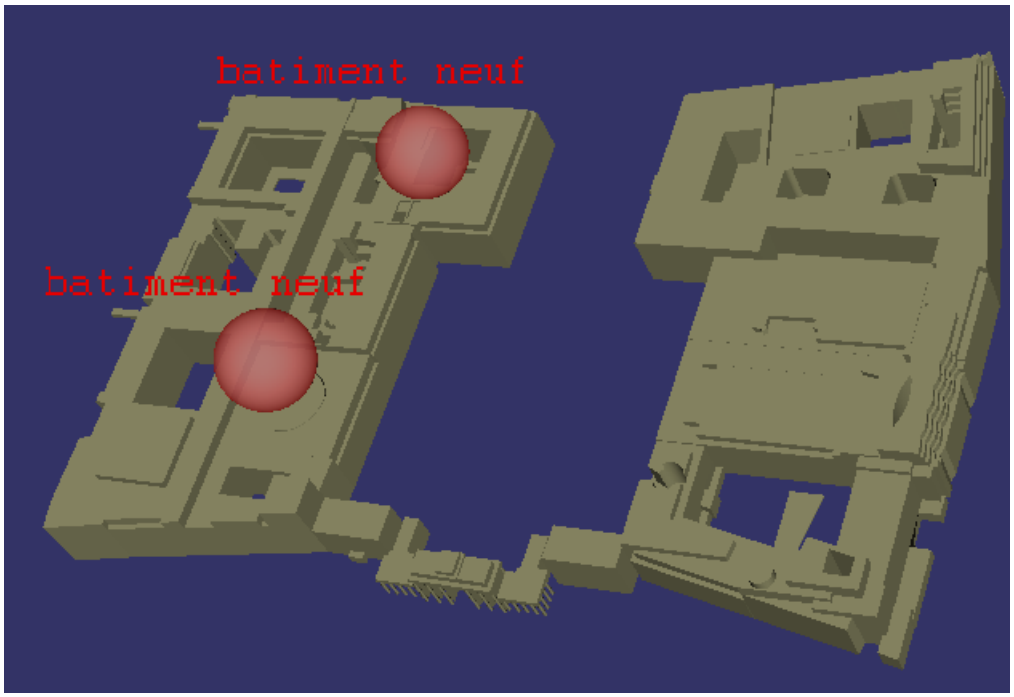


Fig. 12 – *Deux bâtiments se distinguent des autres par une sphère placée au-dessus d'eux et un texte.*

Ce type d'affichage n'est pas pleinement satisfaisant : en effet, il est difficile pour l'utilisateur de savoir quels sont les bâtiments qui sont marqués par cette sphère, d'autant plus sous cet angle de vue ; en outre, l'affichage n'est pas vraiment esthétique. Encore une fois, les limites de chaque bâtiment étant difficilement reconnaissables sur ce fichier, il faudrait conjuguer la présence des sphères avec un changement de couleur ou un affichage plus marqué des lignes de construction des objets 3D.

5.4 Fonctionnalités utiles

Tout d'abord, nous avons mis en place la possibilité de changer l'affichage instantanément grâce à une sélection de touches au clavier. Par exemple, il est possible de sélectionner les bâtiments les uns après les autres et de modifier leur couleur selon le besoin.

6 Conclusion

7 Bibliographie

Références

[Arr10] Arrou-Vignod E. **Rapport de Stage Ingénieur 2010**. *Ecole Centrale de Nantes*, 2010.

[Pet11] Petit G. **Cours de Master 2 STEU**. *IRSTV / Ecole Centrale de Nantes*, 2010–2011.

8 Glossaire

Métadonnées : données sur la donnée. Dans le cas de notre projet, ce sont des informations sémantiques qualifiant les objets graphiques manipulés : par exemple, le nom des bâtiments, la densité de population d'un quartier, le fait qu'une rue soit à sens unique ou non, ...

Sémiologie graphique : ensemble de règles qui indiquent comment afficher des informations sur une carte à travers des éléments visuels, dans l'objectif d'en faciliter le plus possible la lecture et la compréhension.

9 Annexes