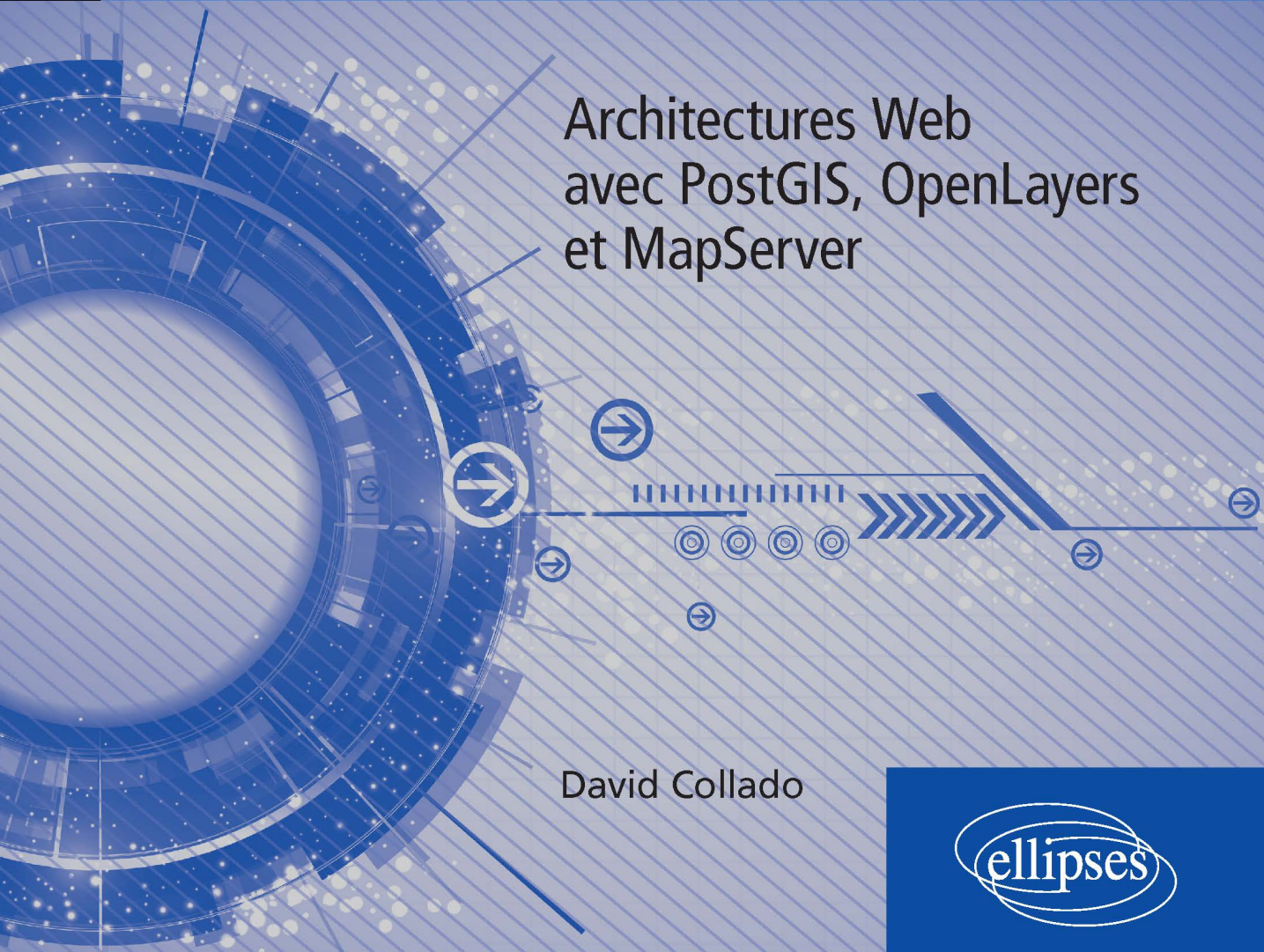
Formations & Techniques

GÉOMATIQUE WEBMAPPING EN OPEN SOURCE

Architectures Web avec PostGIS, OpenLayers et MapServer



David Collado

ellipses

Formations & Techniques

GÉOMATIQUE WEBMAPPING EN OPEN SOURCE

Architectures Web avec PostGIS, OpenLayers et MapServer

David Collado

ellipses

Collection Formations & Techniques

dirigée par Paul de Laboulaye

paul.delaboulaye@editions-ellipses.fr

Retrouvez tous les titres de la collection et des extraits sur www.editions-ellipses.fr

ISBN 9782340-091573
©Ellipses Édition Marketing S.A., 2019
32, rue Bargue 75740 Paris cedex 15

DANGER LE PHOTOCOPILLAGE TUE LE LIVRE

Le Code de la propriété intellectuelle n'autorisant, aux termes de l'article L. 122-5.2° et 3°a), d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective », et d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause est illicite » (art. L. 122-4).
Cette représentation ou reproduction, par quelque procédé que ce soit constituerait une contrefaçon sanctionnée par les articles L. 335-2 et suivants du Code de la propriété intellectuelle.

www.editions-ellipses.fr

Introduction

Nous allons tout au long de ce manuel évoquer l'architecture client-serveur. Elle est au centre du fonctionnement du web. D'un côté l'internaute qui vient chercher de l'information sur un site via son navigateur est un exemple de client ; de l'autre, l'application hôte et gestionnaire du site est un serveur. Par extension, la machine hébergeant ce dernier prend aussi le nom de serveur. Mais il existe d'autres exemples de telles architectures : dès lors qu'un programme est « à l'écoute » d'un réseau et répond à des requêtes qui lui sont adressées, alors ce programme peut être qualifié de serveur. Le client est l'émetteur de la requête.

La communication entre un client et un serveur est parfois complexe car chacun possède sa propre architecture logicielle : les systèmes d'exploitation et les logiciels peuvent être différents mais doivent être capables d'échanger de l'information.

La communauté informatique utilise ainsi des protocoles permettant à chacun de comprendre l'autre. Dans cet ouvrage, nos applications bénéficieront des protocoles TCP/IP et HTTP. Cela s'effectuera de manière transparente car ces protocoles sont toujours pris en charge par les systèmes d'exploitation.

Une fois que deux programmes savent communiquer, il suffit de récupérer les requêtes de l'un et de les transférer au serveur pour qu'une réponse adaptée soit fournie. Plus les requêtes sont riches en informations et plus la réponse peut l'être aussi.

Les sites web qui proposent des services cartographiques ne dérogent pas à cette règle : s'il est facile de publier une carte statique, c'est-à-dire avec laquelle l'internaute n'aura guère d'interactions, il l'est tout autant, aujourd'hui, de doter cette carte d'outils permettant de l'interroger, d'interagir avec la donnée représentée, y compris en la modifiant.

Dans cet ouvrage, nous allons nous intéresser à OpenStreetMap (OSM) dont l'objectif n'est pas moins que cartographier le monde entier de manière collaborative, et qui propose à tout un chacun une utilisation gratuite de la donnée, mais surtout une collaboration à sa confection à l'aide d'outils qui le rapprochent d'un véritable Système d'information géographique (SIG) en ligne.

Ce n'est plus maintenant qu'une question de temps avant que la mutation vers le « tout web » ne s'opère complètement : bientôt il sera banal d'effectuer des opérations de « géotraitement », de jointures, bref toutes les fonctionnalités d'un SIG, directement en ligne.

L'avantage évident d'une telle approche est la mutualisation : toute mise à jour est immédiatement partagée et publique. De plus, avec la profusion d'images aériennes et

satellites accessibles à tous, il est d'ores et déjà possible, là encore quasiment en temps réel, de fournir un service cartographique publiant le résultat d'algorithmes de détection.

Prenons l'exemple de l'agriculture pour illustrer ceci. Si, côté serveur tourne un programme de récupération automatique d'une image satellite dès sa publication, celle-ci peut tout aussi automatiquement être traitée afin de déterminer la vigueur végétative des cultures, et cette information peut être cartographiée et publiée sur le web dans la foulée. En chargeant cette information dans le GPS de son tracteur, l'agriculteur peut agir rapidement et de manière adaptée à chaque parcelle et surtout au niveau intra-parcellaire. Si en retour l'agriculteur fournit les informations issues de ses propres mesures, les modèles qui traitent l'image satellite s'en trouveront mieux calibrés, et cela bénéficiera à tout le monde.

Rien de nouveau cependant dans cette démarche, à part son déploiement en réseau. En effet, depuis quelques décennies, l'ensemble de ces opérations était déjà effectué : via des logiciels de traitement d'images, puis via des SIG, et enfin via des mails voire dans le meilleur des cas via des sites web précurseurs pour partager l'information. Seule l'échelle de temps change : ces opérations pouvaient prendre plusieurs jours et être laborieuses. Aujourd'hui, elles sont quasi-instantanées.

Ainsi, pour construire une architecture de SIG déployé sur le web - ou « webmapping » -, devons-nous nous intéresser à son côté client d'une part, celui de l'internaute, et à son côté serveur de l'autre, celui du créateur du service cartographique.

Côté client, il est nécessaire de se doter d'outils exploitant et complétant le HTML, langage de construction des pages web. Ces outils doivent gérer les interactions entre l'internaute et la page web. Si JavaScript a été construit afin de répondre à de tels besoins, il sera nécessaire de doter ce langage de programmation d'outils spécifiquement cartographiques.

Côté serveur, nous avons besoin d'une technologie permettant de récupérer les requêtes des clients et de gérer les réponses, mais aussi d'un outil de stockage et d'exploitation de la donnée.

Enfin, des outils de génération de cartes sont évidemment indispensables. Ceux-ci sont déployables côté client ou serveur, bien que des solutions exploitant les avantages de chacune de ces deux parties soient judicieuses : la confection de l'image cartographique peut s'effectuer côté client à partir d'images récupérées par des protocoles standardisés côté serveur.

A travers cet ouvrage, nous présenterons des solutions logicielles Open Source qui permettent de construire de telles architectures. Dès lors, il nous a fallu faire des choix car même dans le monde du logiciel libre, il existe une offre pléthorique.

Côté serveur, nous avons opté pour Apache pour prendre en charge les relations avec les clients. Sa notoriété et sa robustesse ne nous ont laissé aucun doute.

En terme de gestion de la donnée physique, nous utiliserons un serveur de base de données, PostGIS, qui est doté de toutes les fonctionnalités d'un SIG (sauf la visualisation de la donnée). PostGIS est basé sur une architecture client-serveur, accessible depuis le web ; et le SQL utilisé pour effectuer les requêtes, quasi indépendant des versions du logiciel, offre une résolution fonctionnelle de tout problème. Autrement dit, le programmeur peut se concentrer sur la logique de la solution sans perdre de temps sur sa programmation impérative qui en détaille toutes les séquences.

Par ailleurs, le système d'information d'une entreprise peut être concentré dans une telle base de données. Cette architecture permet de disposer, en tout lieu connecté au web, de toute l'information en sus de celle à caractère spatiale. Les mises à jour effectuées par les utilisateurs enrichissent alors immédiatement et automatiquement le système d'information, de manière centralisée, rigoureuse et partagée.

Nous nous attarderons sur l'étude de ce logiciel afin de prendre soigneusement en main le SQL spatial.

Nous utiliserons aussi un serveur cartographique dont le rôle est la confection de cartes. Nous avons hésité entre MapServer et GeoServer. MapServer est totalement mature, très efficace, y compris en temps de calcul, simple à installer et rapidement productif. De plus, Mapserver peut être utilisé tel quel, mais peut aussi être intégré au sein de scripts plus globaux dans lesquels il sert à la confection des cartes. Geoserver en revanche, bien que très efficace, est programmé en Java, et aurait nécessité l'installation d'un serveur Tomcat (ce qui nous aurait égaré relativement à notre volonté de simplicité).

C'est donc sur MapServer que nous avons porté notre dévolu.

Nous avons choisi OpenLayers plutôt que LeafLet en guise de client JavaScript pour sa plus grande richesse de fonctionnalités. L'utilisation de ce type de librairies permet de doter des pages web de fonctionnalités SIG, comme l'édition de données, l'affichage, la modification de leurs attributs, à moindre coût et pour un résultat spectaculaire.

Enfin, il est indispensable de se doter d'un logiciel SIG classique, mono-poste, pour visualiser les données, les gérer et les mettre en forme rapidement. Nous avons opté pour QGIS qui est en plein essor et qui s'est imposé tout naturellement de ce fait.

Nous n'avons pas voulu imposer au lecteur l'installation d'un système Linux pourtant très commode, nous expliciterons donc les passages délicats pour chacun des systèmes Windows et Linux de sorte que chacun puisse tirer partie aisément des codes présentés. Ceux-ci ont été testés sur Ubuntu 16.04 et Windows 10.

A qui s’adresse ce livre?

Ce livre est une synthèse d'une quinzaine d'années d'enseignement de la géomatique à des étudiants de Licence professionnelle issus d'horizons très variés en terme de parcours initial (très peu disposaient d'un solide bagage scientifique). Enseigner le SQL ou un langage de programmation peut dès lors paraître rédhibitoire, pourtant cela s'est avéré possible en prenant un point de vue très appliqué.

C'est ce point de vue que nous nous sommes efforcés d'avoir tout au long de la rédaction de ce livre afin de montrer que les outils Open Source sont arrivés à un stade de maturité qui permet de stocker, gérer et publier sur un réseau de la donnée cartographique de manière simple et efficace.

Ce manuel se veut, en conséquence, abordable à qui s'est penché sur la confection de cartes à l'aide d'un logiciel SIG (Systèmes d’Information Géographique). Il y a fort à parier que dans ce cas, l’enjeu de l’acquisition de compétences en Webmapping se soit déjà imposé.

Ce livre peut aussi être utile à des informaticiens ou étudiants en informatique désireux de s'intéresser à la géomatique afin d'ajouter ce domaine en pleine expansion dans leurs compétences. Pour eux, un rappel des notions spécifiques de cette matière sera effectué, dès lors qu’elles sont nécessaires pour appréhender l’analyse d’un problème et de sa solution.

La finalité de l'ouvrage est de présenter une architecture logicielle complète qui permette à tout un chacun de publier de l'information cartographique sur le web à des fins informatives ou de production.

Cela nous entraînera dans l’acquisition de connaissances - au moins synthétiques - dans les domaines de l'architecture client-serveur, du HTML et de son corollaire le JavaScript avec ses librairies dédiées (dont OpenLayers), du langage SQL et aussi d’un langage de programmation comme Python.

L'expérience montre qu'avec un peu de pratique, on se convainc facilement que le jeu en vaut la chandelle. Il y a un côté jubilatoire à mettre en ligne une carte dynamique, qui plus est lorsque la donnée représentée, elle-même dynamique, est le fruit d'un calcul complexe effectué en temps réel.

Nous aborderons ces sujets sur leurs points essentiels afin de les introduire et de motiver le lecteur pour leur étude approfondie. A contrario le lecteur averti s'amusera de la simplicité avec laquelle il est possible de mettre en ligne de la donnée cartographique obtenue très finement.

Comment lire ce livre ?

Afin de tirer un maximum de profit de l'étude de cet ouvrage, le lecteur est invité à mettre en place les architectures évoquées tout au long des chapitres. Pour lui faciliter la tâche, il pourra télécharger l'ensemble des codes créés sur le site internet : www.ggat.fr. Les applications étudiées y sont par ailleurs déployées à titre démonstratif.

Si le novice en SIG ou en programmation orientée web gagnera à étudier et mettre en œuvre toutes les parties de l'ouvrage, le lecteur plus expérimenté pourra survoler certaines parties.

Nous avons voulu, tout au long de ce livre, travailler avec un même jeu de données, aussi l'intégration des données dans PostGIS, vue au chapitre 2, est-elle nécessaire à tous.

Cependant, le lecteur expérimenté en SQL peut aborder le chapitre 3 à partir du paragraphe 3.5 où nous employons PostGIS comme outil de télédétection.

Quant au lecteur maîtrisant Mapserver et le HTML, il peut lui, se dispenser du chapitre 5.

Le lecteur ayant une bonne maîtrise de JavaScript peut lui n'aborder le chapitre 6 qu'à partir de sa section 6.4 puisque nous étudions auparavant les bases du HTML, abordons les notions d'événements et manipulons les objets de la page web de manière classique. Il peut aussi, au chapitre 7, sauter les paragraphe 7.2 et 7.6, où nous étudions la géolocalisation et AJAX. Nous avons, en effet, voulu appréhender progressivement le JavaScript afin d'en montrer la richesse. Il nous est apparu qu'OpenLayers étudié seul en aurait masqué, par sa puissance, les notions basiques mais essentielles.

Enfin, le chapitre 9 est une présentation générale d'OpenLayers, aussi peut-il être survolé, lorsque cette librairie JavaScript est déjà connue du lecteur.

Table des matières

Chapitre 1 - Installation des logiciels.....15
1.1 QGIS 3.0 : Logiciel de SIG.....15
1.2 PostGIS : serveur de bases de données spatiales.....15
1.3 Apache : serveur Web.....17
1.4 Un serveur cartographique orienté web : MapServer.....19
Chapitre 2 - Intégration des données dans PostGIS.....21
2.1 Présentation générale.....21
2.2 Récupération des données utilisées.....22
2.2.1 Données OpenstreetMap (OSM).....22
2.2.2 Données Sentinel 2.....23
2.3 Création d'une base de données spatiale.....28
2.4 Intégration des données téléchargées.....30
2.4.1 Données vectorielles.....31
2.4.2 Données Raster.....33
Chapitre 3 - Exploitation des données en SQL.....37
3.1 Requêtes SQL élémentaires.....37
3.1.1 Structure d'une clause Select.....37
3.1.2 Structure d'une clause update.....43
3.2 Les jointures.....44
3.2.1 Création de tables pédagogiques.....44
3.2.2 CROSS JOIN ou produit cartésien.....46
3.2.3 INNER JOIN ON.....47
3.2.4 LEFT OUTER JOIN ON.....47
3.2.5 FULL JOIN.....48
3.3 Les jointures spatiales.....49
3.3.1 Jointures de couches vectorielles.....49
3.3.2 Jointures de couches Raster.....51
3.4 Les « Common Table Expression » (CTE).....52
3.5 Jointures de Rasters et de vecteurs : détection de forêts.....53
3.6 Rastérisation et extractions de valeurs d'un pixel.....64

Chapitre 4 - Notions d'objets topologiques et implémentation dans PostGIS.....67
4.1 Définition et Intérêt d'une topologie.....67
4.2 Description d'une topologie.....67
4.3 Les topologies dans PostGIS.....70
4.3.1 pgRouting pour le calcul du plus court chemin.....71
4.3.2 Utilisation du schéma topologique.....77
Chapitre 5 - Prise en main de MapServer.....81
5.1 Description de l'architecture logicielle.....81
5.1.1 CGI, FastCGI , Node.js.....81
5.1.2 Anatomie Générale de MapServer.....82
5.1.3 Présentation du mode Dynamique de MapServer.....83
5.2 Le Mapfile.....84
5.2.1 La structure d'un Mapfile.....84
5.2.2 Construction d'un premier Mapfile.....86
5.2.3 Création d'une carte simple.....89
5.2.4 Cartographie des forêts « télédétectées ».....91
5.3 Notions essentielles de HTML.....92
5.3.1 Définition.....92
5.3.2 Structure d'une page HTML.....93
5.3.3 Les tableaux et leurs attributs.....93
5.3.4 Les formulaires.....94
5.3.5 Construction d'une page web.....96
5.4 Déploiement dynamique.....101
5.4.1 Spécificités du Mapfile en mode dynamique.....101
5.4.2 Insertion d'une carte dans une page web.....101
5.4.3 Légende, échelle et étiquettes.....104
Chapitre 6 - Création d'une page web de calcul d'itinéraire routier.....107
6.1 Objectif du chapitre.....107
6.2 Création de la page web.....107
6.3 Du JavaScript pour récupérer les points de départ et d'arrivée..107
6.4 Création d'un Mapfile ad-hoc.....111
6.5 Mise en route.....120

Chapitre 7 - Création d'une page web de suivi d'un trajet.....123
7.1 Un peu de Python : des librairies utiles.....123
7.1.1 Librairie CGI.....123
7.1.2 Librairie MapScript.....124
7.1.3 Psycopg2.....125
7.2 Les outils de géolocalisation proposés par HTML 5 et JavaScript 128
7.3 Mise en route.....131
7.4 Un fond de carte en WMS.....137
7.5 Transformer MapServer en serveur WMS.....140
7.6 Une alternative aux formulaires: AJAX.....144
7.6.1 Le XML : Extensible Markup Language.....145
7.6.2 L'objet XHR: XMLHttpRequest.....147
7.6.3 L'API Fetch.....150
7.6.4 Transmettre une nouvelle position à PostGIS via AJAX....151
Chapitre 8 - Du satellite à la page web en un clic.....155
8.1 Présentation.....155
8.2 Mise en œuvre.....155
Chapitre 9 - Un client cartographique : OpenLayers.....161
9.1 Présentation.....161
9.2 Création d'une carte.....162
9.3 Les couches.....164
9.3.1 Vecteur.....164
9.3.2 Raster.....165
9.4 Les contrôles.....166
9.5 Les interactions.....168
9.6 Les styles.....171
9.7 Les projections cartographiques.....174
Chapitre 10 - La géolocalisation avec OpenLayers.....179
10.1 Suivi d'un trajet avec OpenLayers : côté client.....179
10.2 Suivi d'un trajet avec OpenLayers : côté serveur.....183
Chapitre 11 - Calcul d'itinéraire avec OpenLayers.....187
11.1 Une couche WMS dynamique.....188
11.2 Openlayers et le WMS dynamique.....190

Chapitre 12 - Création d'un éditeur d'entités en ligne....195
12.1 Présentation du projet.....195
12.2 Un serveur WFS-T : TinyOWS.....197
12.3 OpenLayers : affichage d'une couche WFS.....201
12.4 OpenLayers : édition et sauvegarde dans PostGIS.....203
Bibliographie…….... 209
Index …….211

Chapitre 1 - Installation des logiciels

Dans cette partie nous nous focaliserons sur l'installation des logiciels nécessaires tout au long de cet ouvrage.

1.1 QGIS 3.0 : Logiciel de SIG

L'installation de QGIS sous Windows est directe et se fait depuis la page https://www.qgis.org/fr/site/forusers/download.html ou en recherchant « QGIS 3 download » depuis votre moteur de recherche préféré.

Sous Ubuntu, la version par défaut au moment d'écrire cet ouvrage est la 2.8. Nous profitons de la récente sortie de la version 3.0 pour l'installer via les quelques lignes de code suivantes. Il s'agit d'ajouter un dépôt non standard : « ubuntugis-nightly-release » au fichier répertoriant les dépots :

```
sudo gedit /etc/apt/sources.list
```

Et d'ajouter les deux lignes suivantes en fin de fichier :

```
deb https://qgis.org/ubuntugis-nightly-release xenial main
deb-src https://qgis.org/ubuntugis-nightly-release xenial main
```

Avant de mettre à jour les dépots par :

```
sudo apt-get update
```

Et d'installer Qgis et GRASS (qui permet de faire de la télédétection mais que nous n'utiliserons pas dans le cadre de cet ouvrage)

```
sudo apt-get install qgis python-qgis qgis-plugin-grass saga
```

1.2 PostGIS : serveur de bases de données spatiales

PostGIS est l'extension spatiale du système de gestion de base de données (SGBD) PostgreSQL. Il faut donc commencer par installer ce dernier.

Pour les utilisateurs de Windows l'installation est directe depuis le lien suivant : https://www.enterprisedb.com/downloads/postgres-postgresql-downloads.

Vous pouvez installer la version 10.3 de PostgreSQL même si les codes de cet ouvrage fonctionnent aussi sur les versions 9.x.

De même, nos codes sont compatibles avec les versions 2.x de PostGIS dont l'installation est effectuée dans un second temps en utilisant le « Stack Builder », utilitaire installé conjointement à PostgreSQL. Vous le trouverez dans le menu :

```
Démarrer →Programmes→PostgreSQLx.x→Application Stack Builder.
```

Un assistant vous proposera de vous connecter à votre nouvelle base de données, et affichera alors une série d'applications supplémentaires installables. Dans l'onglet « Extension spatiale », choisissez la dernière version de PostGIS comme indiqué sur l'image suivante :

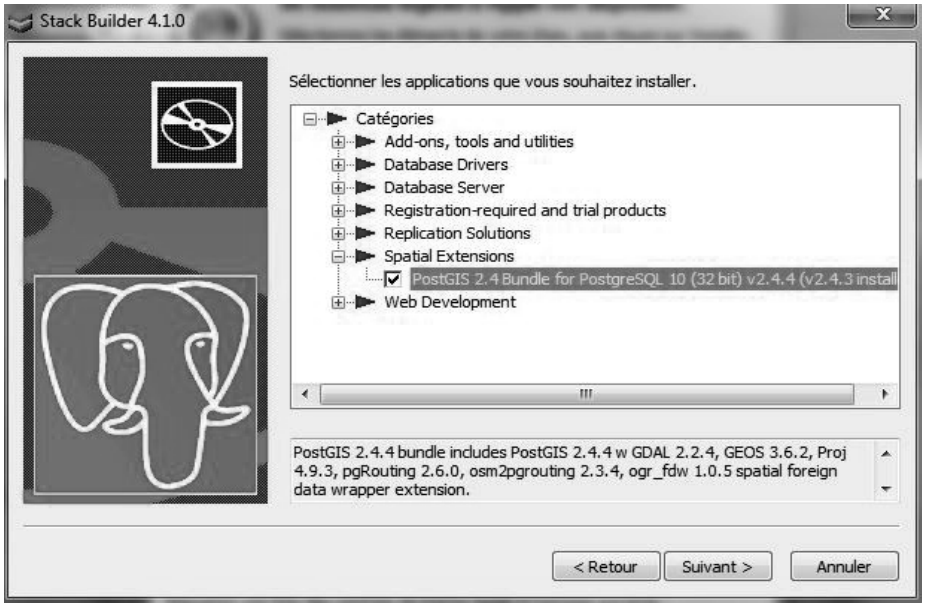


Figure 1.1 : Le gestionnaire d'extension PostgreSQL: Stack Builder

Une fois PostGIS téléchargé, vous serez invité à créer une base de donnée spatiale « Create spatial database », ce que nous vous recommandons : celle-ci servira de modèle aux bases que vous pourrez créer ainsi très rapidement. Il vous restera alors à saisir votre mot de passe pour que l'installation se termine.

Sous Ubuntu, l'installation requiert quelques lignes de code depuis une invite de commande. Voici un exemple de code simple qui permet d'obtenir la version 10 de PostgreSQL : nous ajoutons un dépôt, récupérons la clef de celui-ci avant d'installer notre application.

```
sudo add-apt-repository 'deb http://apt.postgresql.org/pub/repos/apt/ xenial-pgdg main'
wget --quiet -O- https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
sudo apt-get update
sudo apt-get install postgresql-10
```

A partir de là, il est possible de se connecter à PostgreSQL en prenant l'identité « postgres », super-utilisateur par défaut du système installé :

```
sudo su -postgres
```

pour exécuter le programme « psql » qui vous permet de communiquer avec le serveur de base de données :

```
psql
```

et choisir un mot de passe pour l'utilisateur « postgres »:

```
\password postgres
```

Dès lors nous pouvons installer PostGIS 2.4, à partir d'un dépôt « instable » à l'heure d'écrire ces lignes, du fait de sa sortie récente.

```
sudo add-apt-repository ppa:ubuntugis/ubuntugis-unstable
sudo apt update
sudo apt install PostGIS postgresql-10-PostGIS-2.4
```

Enfin nous pouvons créer une base de données, « webmapping » avec toutes les extensions spatiales dont nous aurons besoin :

```
sudo -u postgres createdb -O postgres webmapping
sudo -u postgres psql -c "CREATE EXTENSION PostGIS; CREATE EXTENSION
PostGIS_topology;" webmapping
```

Et nous sommes alors prêts à travailler, même si pour administrer confortablement le serveur, il peut être intéressant d'installer pgAdmin IV.

Sous Windows, l'installation de ce logiciel s'est effectuée conjointement à l'installation de PostgreSQL. Sous Ubuntu, nous vous renvoyons à la documentation en ligne. Le lecteur trouvera un excellent tutoriel à cette adresse :

https://kiahosseini.github.io/help/2016/10/18/installing-pgadmin4-ubuntu-16.04.html

ou encore celle-ci :

http://yallalabs.com/linux/how-to-install-pgadmin-4-in-desktop-mode-on-ubuntu-16-04/

Cette installation requiert un environnement virtuel en Python, la détailler ici nous ferait sortir du cadre de cet ouvrage.

1.3 Apache : serveur Web

Apache est un serveur HTTP Open Source. Il répond aux requêtes d'un client du réseau respectant le protocole HTTP, « Hyper Text Transfer Protocol ». C'est ce protocole qui gère les échanges de pages web à travers Internet.

D'autres protocoles existent, comme le FTP (File Transfer Protocol), permettant par exemple l'échange de fichiers quelconques à travers un réseau.

Apache est installé en général sur une machine connectée à Internet et sert ainsi des clients distants. Cependant il est possible de simuler une architecture similaire sur votre propre ordinateur. C'est ce que nous ferons ici, afin d'effectuer notre prise en main des logiciels étudiés. Ainsi notre propre machine se comportera d'une part comme un client : lorsque depuis notre navigateur nous enverrons des requêtes HTTP. D'autre part elle se comportera comme un serveur : Apache, que nous allons installer maintenant, prendra en charge ces requêtes.

La conséquence de cela est que depuis notre navigateur nous aurons accès à notre propre site web à l' URL : http://localhost

Pour effectuer cette installation sous Windows, il suffit de se rendre sur le site web de Gateway Geomatics et de télécharger MS4W, une suite de logiciels qui comprend entre autres Apache. Voici le lien de téléchargement : https://ms4w.com/.

L'installation se fait très simplement en ne modifiant pas les paramètres par défaut (Installation de MS4W à la racine C:\ et sur le port 80). Si vous souhaitez modifier ces paramètres, cela ne pose aucun problème à condition de vous souvenir de votre choix ultérieurement. Notamment si vous modifiez le port, qui fait le lien physique entre le réseau et votre machine, et donc là où Apache est « à l'écoute », vous aurez accès à votre site via : http://localhost:port

Sous Ubuntu, l'installation là encore, tient en une ligne :

```
sudo apt install apache2
```

Une fois l'installation terminée, vous pouvez la tester.

Saisissez l'URL de votre site local dans votre navigateur, par défaut http://localhost.

Cette URL fait référence à l'adresse physique d'un répertoire qui par défaut est :

```
/var/www/html/
```

sous Ubuntu et qui est sous Windows :

```
C:\ms4w\Apache\htdocs\
```

Nous placerons toutes nos pages web dans ce répertoire.

Nous aurons aussi besoin d'un répertoire qui contiendra les scripts que nous serons amenés à créer :

```
/var/www/cgi-bin/
```

et

```
C:\ms4w\Apache\cgi-bin\
```

sous Windows.

Les codes de cet ouvrage ont été testés sur les deux systèmes. Ainsi, que vous soyez sur un système ou un autre, il vous suffira d'adapter les chemins vers ces répertoires lorsqu'ils figurent dans les codes.

1.4 Un serveur cartographique orienté web : MapServer

MapServer est un projet Open Source porté à l'origine par l'Université du Minnesota. Écrit en C, il excelle à créer des cartes rapidement tout en offrant un grand panel de paramètres et outils permettant un rendu d'excellente qualité. Il peut accéder à de la donnée cartographique sous de nombreux formats, y compris ceux de l'OGC (WMS, WFS..). Ses formats de sorties cartographiques sont tout aussi variés.

Il est utilisable en tant que programme CGI ou en utilisant ses librairies dans votre langage de prédilection : Python, PHP, Java, .NET. Nous aborderons ces notions au chapitre 5.

Son installation est très simple. Nous l'avons réalisée à l'étape précédente en installant Apache sous Windows et sous Ubuntu il suffit d'exécuter :

```
sudo apt-get install cgi-mapserver
```

puis de créer un lien symbolique, placé dans le répertoire :

```
/var/www/cgi-bin/
```

vers l'exécutable.

A son issue et pour vérifier votre installation, saisissez l'URL suivante dans votre navigateur :

http://localhost/cgi-bin/mapserv.exe? sous Windows ou http://localhost/cgi-bin/mapserv? sous Ubuntu. Là encore, il sera indifféremment fait référence dans la suite à l'une ou l'autre de ces URL, le lecteur n'aura qu'à utiliser celle relative à son système.

En retour vous devriez avoir dans votre navigateur un message :

« No query information to decode. QUERY_STRING is set, but empty. » qui indique le succès de votre installation tout en vous signifiant que vous ne demandez rien à faire.

Vous trouverez de nombreuses informations utiles à tout stade de maîtrise du logiciel sur le site :

http://mapserver.org/fr/index.html