



Logiciel Filo-Science

Manuel d'installation et de paramétrage

4/6/2019 - v1.0.1

Éric Quinton

IRSTEA - Centre de Bordeaux
50, avenue de Verdun, Gazinet
33612 CESTAS Cedex

Table des matières

1	Le logiciel Filo-Science	1
1.1	Présentation	1
1.2	Fonctionnalités générales	1
1.3	Technologie employée	1
1.3.1	Base de données	1
1.3.2	Langage de développement et framework utilisé	1
1.3.3	Liste des composants externes utilisés	1
1.4	Sécurité	2
1.5	Licence	3
1.6	Copyright	3
2	Installer le logiciel	4
2.1	Consultez la documentation du framework !	4
2.2	Configurer le serveur	4
2.2.1	Configurer Apache	4
2.2.2	Modules PHP nécessaires	5
2.2.3	Configurer l'hôte virtuel et SSL	5
2.2.4	Configurer le dossier d'installation	5
2.2.5	Droits à attribuer au serveur web	7
2.3	Configurer l'application	7
2.3.1	Connexion à la base de données	7
2.3.2	Identification des utilisateurs	8
2.3.3	Configuration de l'accès à l'annuaire LDAP	9
2.3.4	Paramètres spécifiques	10
2.3.5	Paramètres stockés en base de données	10
2.4	Créer la base de données	11
2.4.1	Login de connexion	11
2.4.2	Droits sur les tables	11
2.4.3	Scripts de modification	11
2.5	Mise en production	12
2.6	Installer une nouvelle version	12
2.6.1	Faites une sauvegarde de la base de données	12
2.6.2	Procédure automatique de mise à jour	12
2.6.3	Sauvegarder le fichier contenant les paramètres de l'application	12
2.6.4	Consultez le fichier news.txt	13
2.6.5	Mise à jour de la structure de la base de données	13
2.6.6	Reconfigurer les droits d'accès au serveur web	13
2.6.7	Supprimer les dossiers inutiles	13
2.6.8	Vérifier la configuration du chiffrement	13

3	Administrer l'application	14
3.1	Gérer les droits	14
3.1.1	Principe général	14
3.1.2	Créer un nouvel utilisateur	16
3.1.3	Créer un login utilisé dans la gestion des droits	17
3.1.4	Définir les groupes d'utilisateur	17
3.1.5	Créer une application	18
3.1.6	Définir les droits utilisables dans l'application	19
3.1.7	Cas particulier des groupes et des logins issus d'un annuaire LDAP	19
3.2	Droits spécifiques de l'application Filo-Science	19
3.2.1	Droits à positionner	19
3.3	Gestion des traces	20
4	Importations et exportations de données	21
4.1	Présentation	21
4.2	Importation de données tabulées	21
4.2.1	Types d'import	22
4.2.2	Types de fonctions	22
4.2.3	Créer un modèle d'import	23
4.2.4	Exécuter un import	24
4.3	Export de données au format JSON	25
4.3.1	Description des modèles d'export	25
4.3.2	Exporter des campagnes ou des opérations	26
4.3.3	Autres exportations	26
4.3.4	Importer un fichier JSON	26
5	Utiliser Qgis avec Filo-Science	27
5.1	Présentation	27
5.2	Utilisation avec Qgis	27
5.2.1	Formulaire Station	28
5.2.2	Formulaire Localisation manuelle des poissons	28
5.2.3	Ajouter un fond de carte OpenStreetMap	29
5.3	Utilisation en mode embarqué	29
5.3.1	Installer l'extension QFieldSync	29
5.3.2	Installer QField dans une tablette Android	29
6	Fonctionnement en mode autonome	31
6.1	Présentation	31
6.2	Installer l'application et la base de données dans un matériel autonome	32
	Bibliographie	33

Chapitre 1

Le logiciel Filo-Science

1.1 Présentation

Le logiciel Filo-Science a été conçu pour permettre l'enregistrement des informations concernant les poissons capturés lors des opérations de pêches scientifiques (pêches électriques, pêches au filet).

Il a été conçu pour l'unité de recherche *Écosystèmes aquatiques et changements globaux* d'IRSTEA, à Cestas (33).

Il a été écrit en PHP, les pages web sont générées en HTML et Javascript avec le composant Smarty.

1.2 Fonctionnalités générales

L'application est organisée autour de la notion de projet : un projet peut regrouper plusieurs campagnes, qui sont menées en réalisant des opérations de pêche. Les droits d'accès sont attribués par projet.

1.3 Technologie employée

1.3.1 Base de données

L'application a été conçue pour fonctionner avec Postgresql, en version 9.6.

1.3.2 Langage de développement et framework utilisé

Le logiciel a été écrit en PHP, en s'appuyant sur le framework *Prototypephp* [18], développé parallèlement par l'auteur du logiciel.

Il utilise la classe *Smarty* [21] pour gérer l'affichage des pages HTML. Celles-ci sont générées en utilisant *Jquery* [9] et divers composants associés. Le rendu général est réalisé avec *Bootstrap* [5].

1.3.3 Liste des composants externes utilisés

Nom	Version	Licence	Usage	Site
PrototypePHP	5/4/2019	LGPL	Framework	github.com/ equinton/ prototypephp
Smarty	3.1	LGPL	Générateur de pages HTML	www.smarty.net

Nom	Version	Licence	Usage	Site
Smarty-gettext	1.2.0	LGPL	Gestion du multi-linguisme avec Smarty	https://github.com/smarty-gettext/smarty-gettext
PHPCAS	1.3.5	Apache 2.0	Identification auprès d'un serveur CAS	wiki.jasig.org/display/CASC/phpCAS
Bootstrap	3.3.7	MIT	Présentation HTML	get.bootstrap.com
js-cookie-master	2.1.4	MIT	Gestion des cookies dans le navigateur	github.com/js-cookie/js-cookie
Datatables	1.10.15	MIT	Affichage des tableaux HTML	www.datatables.net
Datetime-moment		MIT	Formatage des dates dans les tableaux	datatables.net/plug-ins/sorting/datetime-moment
Moment	2.8.4	MIT	Composant utilisé par datetime-moment	momentjs.com
JQuery	3.3.1	≈ BSD	Commandes Javascript	jquery.com
JQuery-ui	1.12.1	≈ BSD	Commandes Javascript pour les rendus graphiques	jqueryui.com
Jquery-timepicker-addon	1.6.3	MIT	Time picker	github.com/trentrichardson/jQuery-Timepicker-Addon
Magnific-popup	1.1.0	MIT	Affichage des photos	dimsemenov.com/plugins/magnific-popup/
Smartmenus	1.1.0	MIT	Génération du menu HTML	www.smartmenus.org
OpenLayers	4.2.0	BSD 2	github.com/openlayers/openlayers	
AlpacaJS	1.5.23	Apache 2	Génération et saisie des métadonnées (pour une version future)	www.alpacajs.org

TABLE 1.1 – Table des composants externes utilisés dans l'application

1.4 Sécurité

L'application a été conçue pour résister aux attaques dites opportunistes selon la nomenclature ASVS v4 [12] de l'OWASP [13]. Des tests d'attaque ont été réalisés en mai 2019 avec le logiciel ZapProxy [14], et n'ont pas détecté de faiblesse particulière.

La gestion des droits est conçue pour qu'un utilisateur ne puisse accéder qu'aux projets auxquels il est rattaché.

1.5 Licence

Le logiciel est diffusé selon les termes de la licence GNU AFFERO GENERAL PUBLIC LICENSE version 3, en date du 19 novembre 2007 [7].

1.6 Copyright

L'application est en cours de dépôt auprès de l'Agence de protection des programmes [3].

Chapitre 2

Installer le logiciel

2.1 Consultez la documentation du framework !

Le logiciel a été conçu à partir du framework *Prototypephp*. La documentation associée [17] récapitule l'ensemble des informations nécessaires pour réaliser l'installation générale (configuration du serveur, définition des droits d'accès, etc.).

De nombreuses passages ont été repris ici, mais il n'est pas inutile de se référer au document d'origine.

2.2 Configurer le serveur

L'application est conçue pour fonctionner à partir d'une adresse unique de type : *https://monsie.com*. Le chiffrement est obligatoire (protocole https). Il n'est pas possible d'installer l'application dans un sous-dossier, par exemple : *https://monsie.com/collec-science* ne fonctionnera pas.

Un script d'installation quasi-automatique est disponible et permet :

- d'installer les paquetages nécessaires (Apache, PHP, Postgresql principalement) ;
- de télécharger la dernière version de l'application ;
- de créer la base de données, avec mise en place d'une sauvegarde automatique ;
- de pré-configurer le serveur pour qu'il soit prêt à être utilisé.

Pour déployer une nouvelle instance, une fois le serveur installé, dans un terminal, tapez les commandes suivantes :

```
wget https://github.com/Irstea/filo-science/install/deploy_new_instance.sh
sudo -s
./deploy_new_instance.sh
```

Suivez les messages affichés à l'écran. Vous devrez notamment modifier le fichier :

```
/etc/apache2/sites-available/filo-science.conf
```

pour indiquer l'adresse DNS utilisée pour accéder à l'application et le certificat de chiffrement associé.

La configuration a été réalisée pour un serveur Linux fonctionnant avec Ubuntu 16.04 LTS Server ou Debian 9. Elle peut bien sûr être adaptée à d'autres distributions Linux. Par contre, rien n'a été prévu pour faire fonctionner l'application directement dans une plate-forme windows, même si, en théorie, cela devrait être possible.

2.2.1 Configurer Apache

Les modules suivants doivent être activés :


```
a2enmod ssl
a2enmod headers
a2enmod rewrite
```

2.2.2 Modules PHP nécessaires

Modules complémentaires nécessaires :

- *php-mbstring*
- *php-pgsql*
- *php7.0-xml*
- *php-xdebug* pour les phases de mise au point
- *php-curl* pour l'identification via un serveur CAS.

Le stockage et l'affichage des photos nécessite :

- *php-imagick*

2.2.3 Configurer l'hôte virtuel et SSL

L'application ne fonctionne qu'en mode SSL, les cookies de session n'étant pas transmis sur des liens non chiffrés. Vous devrez modifier le paramétrage proposé dans le fichier `install/apache2/filo-science.conf`, pour indiquer notamment le nom du certificat utilisé et le nom du site.

Cas particulier de l'identification en mode HEADER

Si vous identifiez vos utilisateurs derrière un proxy d'identification, comme LemonLdap par exemple, vous devrez limiter l'accès de l'application uniquement au proxy. La commande *Directory* devient donc :

```
<Directory /var/www/html>
    Options FollowSymLinks MultiViews
    AllowOverride all
    Order allow,deny
    allow from 10.1.2.3
</Directory>
```

10.1.2.3 correspond à l'adresse IP du serveur proxy d'identification.

2.2.4 Configurer le dossier d'installation

Le principe général est que le dossier contenant l'application contient, dans son nom, le numéro de version (v2.0 par exemple), et un lien virtuel (*filo-science*) pointe vers celui-ci. C'est le lien qui est la cible de l'adresse web : ainsi, à chaque nouvelle version, il suffit de mettre à jour le code de l'application et de faire pointer le lien vers le nouveau dossier pour que celle-ci soit opérationnelle.

Des scripts seront fournis pour réaliser automatiquement les mises à jour (dans le cas d'installations mono-instances).

Cas général : une seule instance hébergée dans le serveur

Utilisez le script fourni, qui créera automatiquement les dossiers nécessaires.

Cas particulier : faire cohabiter plusieurs instances avec le même code

Il est possible d'utiliser le même code applicatif pour alimenter des bases de données différentes (ou des données stockées dans des schémas différents). Cette fonctionnalité est basée sur l'attribution d'entrées DNS différentes.

Le mécanisme est décrit dans la figure 2.1 *Schéma général d'implémentation pour utiliser le même code avec des noms d'application et des jeux de données différents*, page 6.

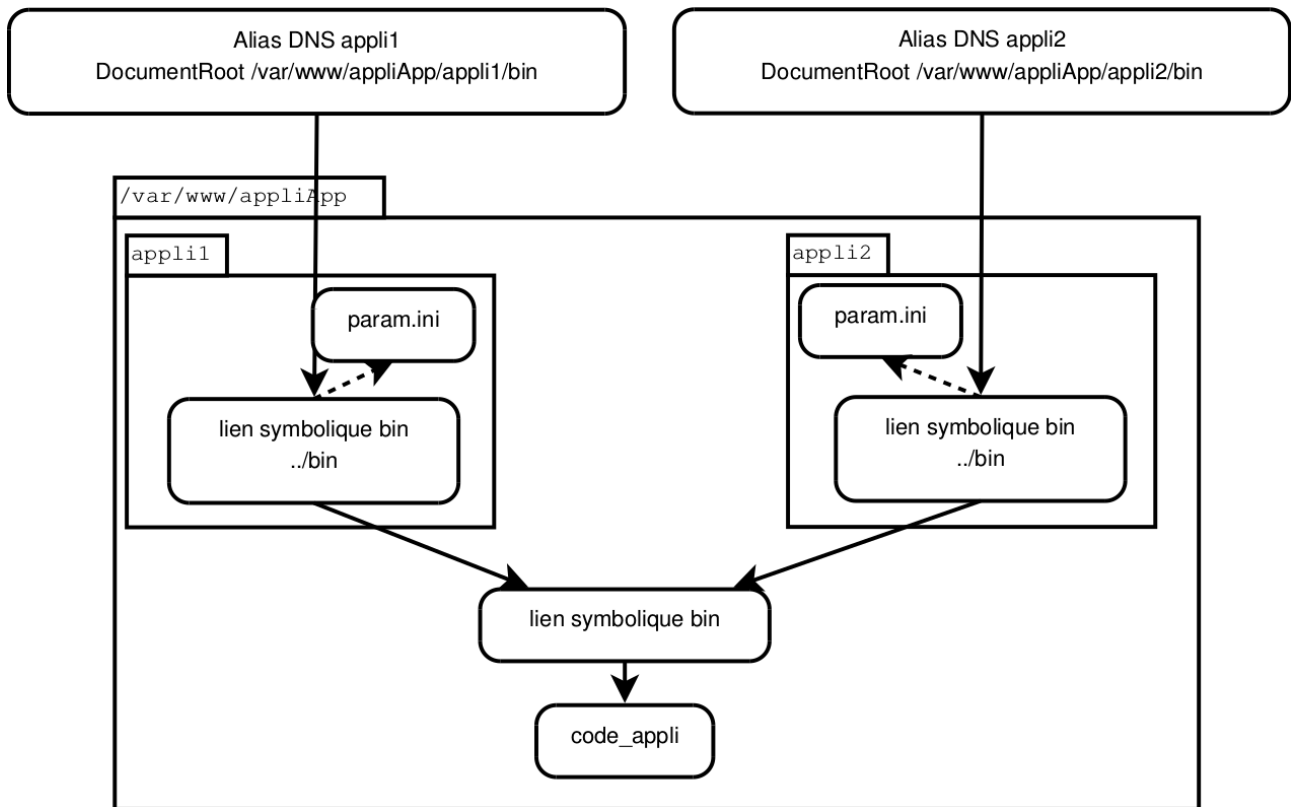


FIGURE 2.1 – Schéma général d'implémentation pour utiliser le même code avec des noms d'application et des jeux de données différents

Dans le paramétrage de l'alias DNS (en principe, dans */etc/apache2/sites-available*), l'application pointe vers le dossier */var/www/appliApp/appli1/bin*. */var/www* correspond à la racine du site web, *appliApp* au dossier racine de l'application, *appli1* au dossier spécifique de l'alias DNS. Ce dossier *appli1* ne contient que deux fichiers : *param.ini*, qui contient les paramètres spécifiques, et *bin*, qui est un lien symbolique vers le dossier *../bin*.

Le dossier *../bin* (donc, dans */var/www/appliApp*) est lui aussi un alias qui pointe vers le code réel de l'application, ici *code_appli*. Le fichier *param.inc.php* doit contenir les commandes suivantes pour que le fichier *param.ini* soit correctement chargé selon le contexte :

```
$chemin = substr($_SERVER["DOCUMENT_ROOT"],0, strpos($_SERVER["DOCUMENT_ROOT"],"/bin"
));
$paramsIniFile = "$chemin/param.ini";
```

Le fichier *param.ini* sera cherché dans le dossier parent du code de l'application, c'est à dire soit dans *appli1*, soit dans *appli2* dans cet exemple. Il suffit qu'il contienne les paramètres adéquats pour rendre l'application utilisable dans des contextes différents à partir du même code initial.

Le fichier *param.ini* est le dernier qui est traité par l'application pour récupérer les paramètres. Ceux-ci sont lus dans l'ordre suivant :

param/param.default.inc.php → param/param.inc.php → ../param.ini

param.ini contiendra les entrées spécifiques liées au DNS utilisé pour accéder à l'application, en principe tout ou partie de celles-ci :

```
BDD_schema=filo-science, public, gacl
BDD_login=compte_de_connexion
BDD_passwd=mot_de_passe_de_connexion
```

```
BDD_dsn=pgsql:host=serveur;dbname=base_de_donnees;sslmode=require
GACL_aco=filo
```

Si un libellé contient une apostrophe, la chaîne doit être insérée dans des guillemets doubles, comme ici pour la variable *APPLI_titre*.

2.2.5 Droits à attribuer au serveur web

Le serveur web doit pouvoir accéder en lecture à l'ensemble des fichiers de l'application, et en écriture à deux dossiers :

- *display/templates_c* : fichier utilisé par Smarty pour compiler les modèles de documents HTML ;
- *img* : dossier de génération des images et des fichiers temporaires.

Deux scripts sont fournis pour attribuer les droits :

- **install/apache2/upgrade_rights.sh** : positionne les droits en utilisant les droits standards Linux (owner, group)
- **install/apache2/upgrade_rights_with_acl.sh** : positionne les droits à partir des ACL.

Les scripts doivent être lancés ainsi :

```
filo-2.0/install/apache2/upgrade_rights.sh v2.0
```

ou

```
filo-2.0/install/apache2/upgrade_rights_with_acl.sh v2.0
```

2.3 Configurer l'application

L'application est configurable par l'intermédiaire de trois fichiers :

param/param.default.inc.php → **param/param.inc.php** → **../param.ini**

Le premier fichier contient les paramètres par défaut. Il est systématiquement fourni à chaque nouvelle version de l'application.

Le second est spécifique de l'implémentation. Il comprend notamment les informations liées à la connexion à la base de données, à la méthode d'identification, ou à la recherche des attributs dans l'annuaire LDAP.

le troisième est destiné à offrir la possibilité d'accéder, à partir du même code applicatif, à plusieurs bases de données différentes (cf. 2.2.4 *Cas particulier : faire cohabiter plusieurs instances avec le même code*, page 5).

Voici les principaux paramètres utilisés :

2.3.1 Connexion à la base de données

Dans la pratique, deux connexions sont nécessaires : l'une pour accéder à la base des droits, l'autre aux données proprement dites. Voici les paramètres à définir :

Variable	Signification
BDD_login	compte de connexion à la base de données
BDD_passwd	mot de passe associé
BDD_dsn	adresse de la base de données sous forme normalisée
BDD_schema	schéma utilisé (plusieurs schémas peuvent être décrits, en les séparant par une virgule - fonctionnement propre à Postgresql)
GACL_dblogin	compte de connexion à la base de données des droits
GACL_dbpasswd	mot de passe associé
GACL_dsn	adresse normalisée

Variable	Signification
GACL_schema	schéma utilisé
GACL_aco	nom du code de l'application utilisé dans la gestion des droits

TABLE 2.1 – Variables utilisées pour paramétrer les connexions

2.3.2 Identification des utilisateurs

Variable	Signification
ident_type	Type d'identification supporté. L'application peut gérer BDD (uniquement en base de données), LDAP (uniquement à partir d'un annuaire LDAP) LDAP-BDD (d'abord identification en annuaire LDAP, puis en base de données), CAS (serveur d'identification <i>Common Access Service</i> ¹), et enfin HEADER (identification derrière un proxy qui fournit le login dans une variable d'entête HTTP)
CAS_plugin	Nom du plugin utilisé pour une connexion CAS
CAS_address	Adresse du serveur CAS
CAS_port	Systématiquement 443 (connexion chiffrée)
CAS_CApth	chemin d'accès au certificat du serveur CAS
LDAP	tableau contenant tous les paramètres nécessaires pour une identification LDAP
ident_header_login_var	par défaut, AUTH_USER. Nom de la variable qui contiendra le login dans le cas d'une identification en mode HEADER (le radical HTTP_ ne doit pas être indiqué)
privateKey	clé privée utilisée pour générer les jetons d'identification (ré-identification automatique après une première connexion)
pubKey	clé publique utilisée pour générer les jetons d'identification
tokenIdentityValidity	durée de validité, en secondes, des jetons d'identification
MAIL_enabled	Si à 1, l'envoi de mail est géré par l'application
CONNEXION_max_attemps	nombre maximum d'essais de connexion avant blocage temporaire du compte
CONNEXION_blocking_duration	durée de blocage du compte
APPLI_mailToAdminPeriod	intervalle de temps entre l'envoi d'un mail de notification de blocage de compte à un administrateur
APPLI_admin_ttl	durée de vie d'une session d'administration (temps maximum entre deux accès à une page d'administration avant réidentification)
APPLI_lostPassword	Si à 1, autorise la récupération du mot de passe perdu, par envoi d'un mail avec un lien chiffré. Nécessite également que MAIL_enabled soit positionné à 1

TABLE 2.2 – Variables utilisées pour paramétrer l'identification

Ré-identification par jeton

L'application permet de conserver l'identification plus longtemps que celle définie dans le serveur, en jouant la connexion avec un jeton d'identification chiffré. Cela évite, par exemple, de devoir se

1. serveur externe gérant l'identification des utilisateurs, et renvoyant à l'application le login utilisé

ré-identifier toutes les heures si on accède au logiciel à partir d'un terminal mobile (smartphone ou tablette, par exemple).

Les trois dernières variables permettent de configurer ce mode d'identification.

Le framework peut générer un jeton chiffré après la première identification, qui sera analysé pour savoir si l'utilisateur peut être ré-identifié automatiquement.

Pour que ce mécanisme fonctionne, il faut :

- que le paramètre *tokenIdentityValidity* ait une durée de validité supérieure à la durée de vie de la session. Il est raisonnable de ne pas fixer une durée de vie supérieure à une journée de travail (10 heures). Le cookie transmis est protégé;
- que les clés privée et publique, utilisées pour le chiffrement du jeton, soient accessibles au serveur web (variables *privateKey* et *publicKey*).

Les clés sont générées automatiquement avec le script d'installation automatique du serveur et de l'application.

Le jeton est chiffré avec la clé privée, ce qui lui permet d'être lu, le cas échéant, par l'application. Il contient le login et la date d'expiration.

Si l'utilisateur déclenche une déconnexion, le jeton est supprimé.

Pour plus d'informations, consultez comment fonctionne le mécanisme de ré-identification par jeton [19].

Identification par HEADER

Dans ce mode d'identification, le serveur web est placé derrière un serveur d'identification, appelé proxy d'identification. L'adresse de l'application pointe vers ce dernier.

Le proxy gère la connexion de l'utilisateur, et fournit à l'application le login dans une variable configurable. Cette variable est accessible dans le tableau `$_SERVER`, par exemple `$_SERVER["HTTP_AUTH_USER"]`.

Pour activer ce mécanisme, il faut modifier les paramètres suivants dans le fichier *param.ini.php* :

```
$ident_type = "HEADER";
$ident_header_login_var = "AUTH_USER";
```

la variable ne doit pas contenir la racine `HTTP_` (une fonction l'extrait automatiquement).

2.3.3 Configuration de l'accès à l'annuaire LDAP

Les paramètres LDAP sont stockés dans un tableau :

```
$LDAP = array(
    "address"=>"localhost",
    "port" => 389,
    "rdn" => "cn=manager,dc=example,dc=com",
    "basedn" => "ou=people,ou=example,o=societe,c=fr",
    "user_attr" => "uid",
    "v3" => true,
    "tls" => false,
    "groupSupport"=>true,
    "groupAttr"=>"supannentiteaffectation",
    "commonNameAttr"=>"displayname",
    "mailAttr"=>"mail",
    'attributgroupname' => "cn",
    'attributloginname' => "memberuid",
    'basedngroup' => 'ou=example,o=societe,c=fr'
);
```

L'application peut non seulement identifier les utilisateurs auprès de l'annuaire LDAP, mais également récupérer les groupes auxquels ils appartiennent dans celui-ci.

Voici les paramètres à indiquer dans ce cas de figure (valable en principe pour tout annuaire compatible OpenLdap) :

Variable	Signification
address	adresse de l'annuaire
port	389 en mode non chiffré, 636 en mode chiffré
rdn	compte de connexion, si nécessaire
basedn	base de recherche des utilisateurs
user_attr	nom du champ contenant le login à tester
v3	toujours à <i>true</i>
tls	<i>true</i> en mode chiffré
groupSupport	true si l'application recherche les groupes d'appartenance du login dans l'annuaire
groupAttrib	Nom de l'attribut contenant la liste des groupes d'appartenance
commonNameAttrib	Nom de l'attribut contenant le nom de l'utilisateur
mailAttrib	Nom de l'attribut contenant l'adresse mail de l'utilisateur
attributgroupname	Attribut contenant le nom du groupe lors de la recherche des groupes (cn par défaut)
attributloginname	attribut contenant les membres d'un groupe
basedngroup	base de recherche des groupes

TABLE 2.3 – Variables utilisées pour paramétrer l'accès à l'annuaire LDAP

2.3.4 Paramètres spécifiques

Variable	Signification
APPLI_photoStockage	dossier contenant les photos générées par l'application, avant transmission au navigateur. Par défaut : <i>img</i>
APPLI_maxfilesize	Taille maximale des photos téléchargeables. Par défaut : 100000000

TABLE 2.4 – Variables spécifiques

2.3.5 Paramètres stockés en base de données

À partir de la version 2, certains paramètres peuvent être stockés dans la base de données, pour éviter qu'ils ne soient dépendants de la configuration du serveur.

Ces paramètres sont accessibles depuis le menu *administration*, item *Paramètres de l'application*.

Voici la liste des paramètres actuellement décrits :

Variable	Signification
APPLI_title	Titre de l'application tel qu'il figure entre l'icône et le menu
mapDefaultLat	Latitude de positionnement par défaut des cartes
mapDefaultLong	Longitude de positionnement par défaut des cartes
mapDefaultZoom	Niveau de zoom des cartes par défaut

TABLE 2.5 – Paramètres stockés dans la base de données

2.4 Créer la base de données

La base de données est composée de deux schémas : l'un pour stocker les informations d'identification, les droits d'accès et les traces (*gacl*), l'autre pour les données proprement dites (*filo*).

Le schéma *public* ne devrait jamais être utilisé pour stocker l'information : réservez-le pour les composants communs, comme Postgis.

Les tables de gestion des droits peuvent être communes à plusieurs jeux / applications différentes : la variable *GACL_aco* permet de séparer la gestion des droits pour chaque application, tout en travaillant à partir des mêmes utilisateurs (répartis le cas échéant dans des groupes différents selon le jeu de données considéré).

Les tables sont créées à partir du script *install/pgsql/create_db.sql*.

Un compte d'administration par défaut est créé automatiquement :

— login : **admin**

— mot de passe : **password**

Il devra être supprimé quand un autre compte d'administration aura été créé.

2.4.1 Login de connexion

Si la base de données et l'application sont hébergés dans deux serveurs différents, il est fortement conseillé de créer deux logins de connexion, un pour le schéma des droits, l'autre pour les schémas applicatifs. Ces logins ne doivent pouvoir être utilisés que depuis le serveur web hébergeant l'application.

Cette opération est possible en modifiant le fichier */etc/postgresql/9.5/main/pg_hba.conf* selon ce principe :

```
# Connexions pour les serveurs web
host nom_database userGacl adresse_serveur/32 md5
host nom_database userData adresse_serveur/32 md5
```

Le login utilisé dans *userGacl* correspond à la variable *\$GACL_dblogin*, et *userData* à *\$BDD_login*. et en rechargeant ensuite la configuration de Postgresql avec la commande :

```
service postgresql reload
```

2.4.2 Droits sur les tables

Le compte utilisé pour la connexion au schéma des droits doit pouvoir modifier les informations présentes dans l'ensemble des tables de *gacl*. Il ne devrait pas pouvoir accéder aux autres schémas (hormis *public*), sauf en cas d'installation mono-serveur (base de données hébergée dans la même machine et connexion à distance impossible).

Le compte utilisé pour accéder aux schémas des données doit pouvoir modifier l'ensemble des informations dans les schémas de données, et lire la table *gacl.aclgroup*.

Le plus simple est d'utiliser le logiciel *pgAdmin* [15] pour attribuer les droits.

Le script d'installation automatique rend le compte *filo* propriétaire de la base de données, ce qui simplifie la gestion des droits sur les tables.

2.4.3 Scripts de modification

Lors de la livraison de nouvelles versions, il est possible que des scripts de modification soient livrés pour mettre à niveau la base de données. Ces scripts doivent être exécutés dans tous les schémas contenant des données applicatives (pour plus de détails, consultez ci-après *Installer une nouvelle version*).

2.5 Mise en production

Une fois l'application configurée, et après avoir créé un nouveau compte d'administration :

- supprimez le compte *admin*, livré par défaut, qui ne doit pas être conservé. Sa désactivation n'est pas suffisante : si pour une raison ou pour une autre le compte est réactivé, n'importe qui pourra récupérer les droits totaux ;
- supprimez le dossier *install* qui contient les scripts de création des tables ;
- déplacez le dossier *database*, qui contient la documentation d'installation et de configuration (elle n'a pas à rester accessible depuis le site web) ;
- faites une revue des droits, pour vous assurer que tout est correctement configuré.

Vous pouvez également tester si la configuration du serveur est correcte en recourant à *ZAPProxy* [14], qui analysera la communication entre le serveur et un navigateur et identifiera les problèmes éventuels de non conformité (mauvaise réécriture des entêtes HTML suite à une mauvaise configuration du serveur Apache, par exemple).

2.6 Installer une nouvelle version

2.6.1 Faites une sauvegarde de la base de données

Il arrive fréquemment que la structure de la base de données évolue. Avant toute opération, assurez-vous de disposer d'une sauvegarde, dans un autre support.

Un programme de sauvegarde est disponible dans *install/pgsql/backup.sh*. Vous pouvez l'exécuter manuellement ainsi :

```
su postgres -c "install/pgsql/backup.sh"
```

La sauvegarde sera stockée dans */var/lib/postgresql/backup*.

Si vous avez utilisé le script d'installation automatique, le programme est également présent dans */var/lib/postgresql*.

2.6.2 Procédure automatique de mise à jour

Si l'application est installée dans un serveur dédié à cet usage (installation réalisée avec le script https://github.com/Irstea/filo-science/blob/master/install/deploy_new_instance.sh), un script de mise à jour automatique peut être téléchargé. Il est disponible dans le dossier <https://github.com/Irstea/filo-science/tree/master/install>, et est sous la forme :

```
upgrade-2.0-2.1.sh
```

où **2.0** correspond à la version courante de votre installation, et **2.1** à la version cible.

Pour utiliser ce script :

```
sudo -s
wget https://github.com/Irstea/filo-science/raw/master/install/upgrade-2.0-2.1.sh
chmod +x upgrade-2.0-2.1.sh
./upgrade-2.0-2.1.sh
```

Le script va télécharger le code de la nouvelle version et mettra à jour la base de données, si c'est nécessaire.

2.6.3 Sauvegarder le fichier contenant les paramètres de l'application

Le fichier *param/param.inc.php* contient vos paramétrages spécifiques. Lors de l'installation d'une nouvelle version, il va être supprimé.

Faites-en une copie, et remettez-le en place après avoir installé la nouvelle version.

2.6.4 Consultez le fichier `news.txt`

Le fichier `param/news.txt` contient la description des modifications apportées au logiciel. Il précise notamment si une mise à jour de la base de données doit être appliquée.

2.6.5 Mise à jour de la structure de la base de données

Le dossier `install/pgsql` contient les scripts de création et de mise à jour de la base de données. Les scripts de mise à jour sont nommés ainsi :

```
alter_versionAnterieure-versionMiseAJour.sql
```

`versionAnterieure` correspond à la version la plus ancienne qui doit être mise à jour, `versionMiseAJour` la version cible. Par exemple :

```
alter_2.0-2.1.sql
```

indique que toutes les versions entre 2.0 et 2.1 doivent être mises à jour avec le script indiqué. Si vous avez « sauté » certaines versions du logiciel, il est possible que plusieurs scripts doivent être appliqués.

La mise à jour doit être appliquée dans tous les schémas contenant des données, notamment dans le cas où le même logiciel est utilisé pour gérer plusieurs jeux de données.

Avant d'exécuter les scripts, vérifiez leur contenu, et notamment le nom des schémas.

Ne relancez jamais l'exécution d'un script.

2.6.6 Reconfigurer les droits d'accès au serveur web

Après installation de la nouvelle version du code, n'oubliez-pas de reconfigurer les accès en lecture pour le compte utilisé pour faire fonctionner le serveur web, et en écriture pour les dossiers `img` et `display/templates_c` (cf. 2.2.5 *Droits à attribuer au serveur web*, page 7).

2.6.7 Supprimer les dossiers inutiles

Une fois la mise en production validée, supprimez le dossier `install` et déplacez le dossier `database`, et faites une revue des droits pour vous assurer qu'il n'y a pas eu de modification intempestive ou que la configuration est toujours correcte.

2.6.8 Vérifier la configuration du chiffrement

Avec un navigateur récent, ou en testant le site (s'il est accessible depuis internet) à partir de SSLLABS, vérifiez que l'application soit correctement configurée, notamment au niveau du serveur Apache.

Chapitre 3

Administrer l'application

3.1 Gérer les droits

3.1.1 Principe général

Les droits sont gérés selon le principe initialement utilisé dans la bibliothèque PHPGACL [4], aujourd'hui obsolète.

Les logins sont déclarés dans des groupes organisés de manière hiérarchique : un groupe hérite des droits attribués à ses parents.

Les droits utilisés dans le logiciel sont associés à des groupes. Il est possible d'attribuer plusieurs droits à un même groupe, et un droit peut être détenu par des groupes différents.

Si le paramètre `$LDAP["groupSupport"]` est positionné à `true`, les groupes dont fait partie le compte LDAP sont également récupérés. Si ces groupes se voient attribués des droits, les comptes associés les récupéreront automatiquement.

Voici le schéma des tables utilisées pour gérer les droits :

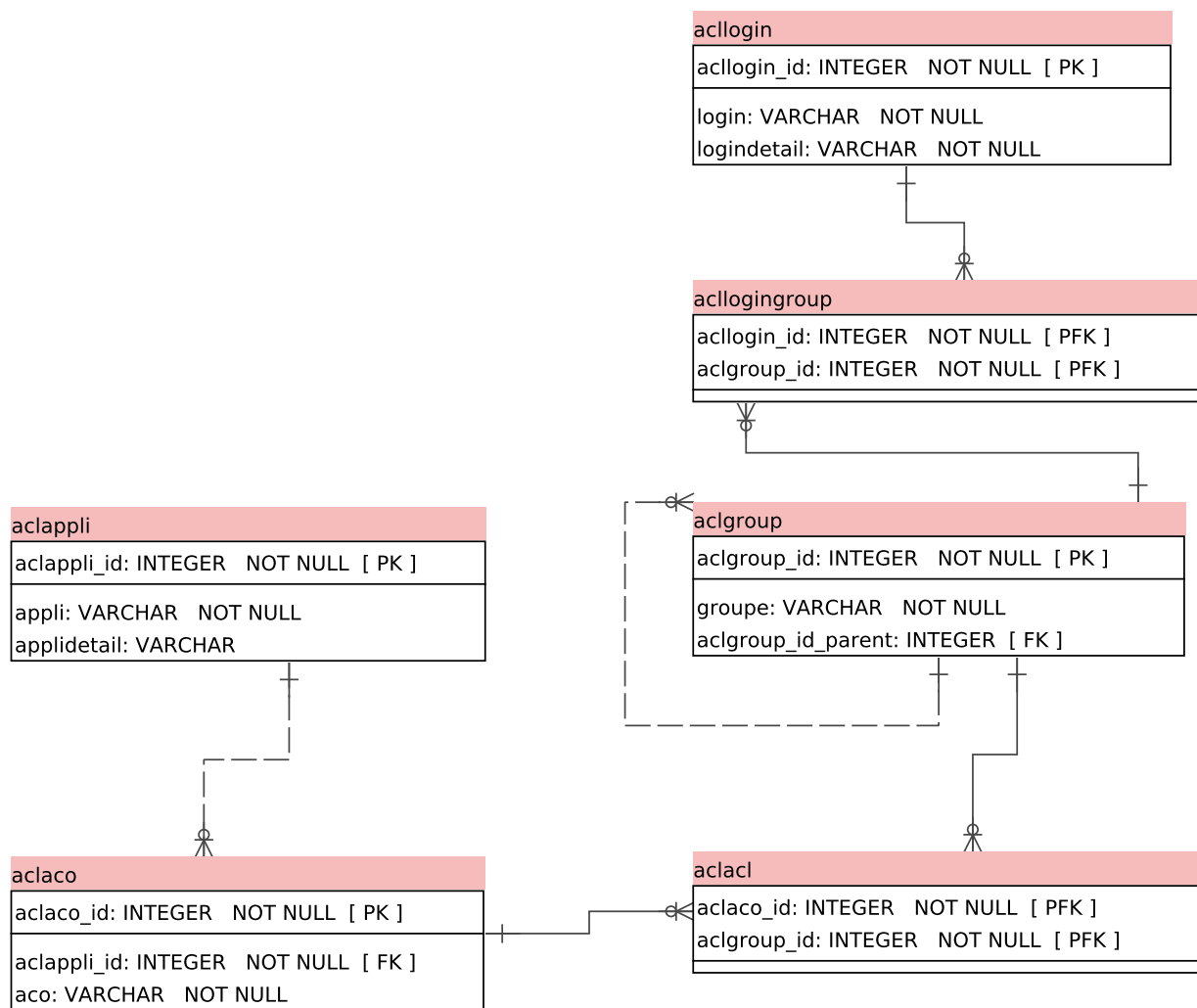


FIGURE 3.1 – Schéma des tables utilisées pour gérer les droits

Voici la description des tables :

acllogin : liste des logins utilisés. Si un compte est créé dans la base locale d'identification, un enregistrement est également créé dans cette table. Pour les identifications LDAP ou CAS, ils doivent être identiques. Si seuls les groupes LDAP sont utilisés pour un compte, il n'a pas besoin d'être décrit ici ;

aclappli : liste des applications gérées. Il est possible de gérer, à partir de la même base de données, plusieurs ensembles de droits, qui utilisent les mêmes logins.

aclaco : liste des droits déclarés dans l'application ;

aclgroup : liste des groupes contenant les logins, et qui détiennent les droits. Un groupe peut hériter d'un autre groupe. Les droits associés au groupe parent sont également attribués au groupe hérité ;

aclloggingroup : table permettant de déclarer les logins associés à un groupe ;

aclacl : table décrivant les droits détenus par un groupe.

Le module d'administration permet de saisir toutes ces informations. Il faut que l'utilisateur dispose du droit *admin*, c'est à dire qu'il fasse partie du groupe *admin* (configuration par défaut à l'initialisation de la base des droits) pour pouvoir accéder à ces fonctions.

3.1.2 Créer un nouvel utilisateur

Les utilisateurs peuvent être issus soit de l'annuaire LDAP, soit de la base interne. Pour créer un nouvel utilisateur dans la base locale :

- *Administration* → *Liste des comptes*
- *Nouveau login*
- renseignez au minimum le login.

FIGURE 3.2 – Écran de saisie d'un login de connexion

Pour créer le mot de passe, vous pouvez cliquer sur le bouton *Générer*, qui en générera un automatiquement. Envoyez-le par mël à son destinataire (par *copier-coller*), en lui demandant de le modifier à la première connexion (icône en forme de clé, dans le bandeau, en haut à droite).

Les mots de passe doivent respecter les règles suivantes :

- ils doivent avoir une longueur minimale de 10 caractères ;
- ils doivent comprendre trois types de caractères différents parmi les minuscules, majuscules, chiffres et caractères de ponctuation ;
- ils ne peuvent pas être réutilisés pour le même login ;
- les mots de passe n'expirent pas.

Les mots de passe sont stockés sous forme d'empreinte, calculée en rajoutant un sel¹ et encodés en SHA256 : ils ne peuvent pas être retrouvés en cas de perte.

L'application n'intègre pas de module permettant de régénérer automatiquement un mot de passe en cas de perte : c'est au responsable applicatif d'en fournir un nouveau.

1. chaîne de caractère rajoutée au mot de passe – en général le login ou un identifiant – qui permet d'éviter que deux mots de passe identiques, associés à deux logins différents, aient la même empreinte

La création d'un compte entraîne la création d'une entrée identique dans la table des *aclogin*, utilisée pour attribuer les droits.

Pour désactiver temporairement un compte, sélectionnez *non* dans la zone *actif*. Si le compte ne doit plus être utilisé, supprimez-le.

Attention : si le compte disposait des droits d'administration, assurez-vous que vous avez toujours un compte disposant des mêmes droits avant la suppression.

3.1.3 Créer un login utilisé dans la gestion des droits

Indépendamment du compte de connexion, qui peut être soit issu de la base interne, soit récupéré auprès d'un annuaire LDAP ou d'un serveur CAS, l'application a besoin de connaître les utilisateurs pour pouvoir leur attribuer des droits.

À partir du menu, choisissez *Administration* → *ACL - logins*.

Vous pouvez modifier un login existant ou en créer un nouveau. Dans ce cas, vous devrez indiquer au minimum le login utilisé (identique à celui qui est employé pour la connexion à l'application : base de données interne, annuaire LDAP, serveur CAS).

Modification d'un login (module de gestion des droits)

[Retour à la liste des logins](#)

Nom de l'utilisateur * :

Login utilisé * :

Valider

Supprimer

*Donnée obligatoire

Droits attribués

consult

param

FIGURE 3.3 – Écran de modification d'un login dans le module de gestion des droits

Sous l'écran de saisie figurent la liste des droits attribués à un login (en modification, le calcul n'est réalisé qu'à l'affichage de la page).

3.1.4 Définir les groupes d'utilisateur

Les groupes d'utilisateurs sont gérés selon un mécanisme d'héritage. Un groupe de haut niveau hérite des groupes précédents : si des droits ont été attribués à un groupe de niveau inférieur, un login associé à un groupe de niveau supérieur les récupère également.

Pour définir les groupes, dans le menu, choisissez *Administration* → *ACL - groupes de logins*.

Nouveau groupe racine...

Nom du groupe	Nombre de logins déclarés	Rajouter un groupe fils
consult		+
gestion	8	+
gestionCompte	3	+
admin	4	+

FIGURE 3.4 – Liste des groupes de logins

Ainsi, le login déclaré dans le groupe *admin* récupérera les droits attribués aux groupes *gestion-Compte*.

Pour créer un groupe, il existe deux possibilités :

- soit le groupe est à la base d'une nouvelle branche : utilisez alors *Nouveau groupe racine...* ;
- soit le groupe hérite d'un autre groupe : cliquez sur le signe + (*Rajouter un groupe fils*).

Vous pouvez indiquer les logins qui sont rattachés à ce groupe.

3.1.5 Créer une application

Le moteur utilisé pour faire fonctionner le logiciel Otolithe permet de gérer des droits différents pour des jeux de données différents, à partir du même code applicatif. Chaque couple *logiciel* ↔ *base de données* constitue donc une *application*, au sens de la gestion des droits.

Il est ainsi possible, à partir de la même base de données, de définir des droits différents selon les jeux de données utilisés (un jeu de données correspond à un schéma de base de données comprenant l'intégralité des tables applicatives).

À partir du menu, choisissez *Administration* → *ACL - droits* :


Modifier	Nom de l'application	Description
	otolithe	

FIGURE 3.5 – Liste des applications déclarées

Pour créer une nouvelle application, choisissez *Nouvelle application...*

[Retour à la liste des applications](#)

* Nom de l'application :

Description :

Valider Supprimer

*Donnée obligatoire

FIGURE 3.6 – Écran de saisie d'une application

Le nom de l'application doit impérativement correspondre à la valeur *\$GACL_appli* dans les fichiers de paramètres : c'est ce qui permet au framework de savoir quels droits appliquer.

3.1.6 Définir les droits utilisables dans l'application

À partir de la liste des applications, cliquez sur le nom de celle pour laquelle vous voulez définir les droits utilisables. À partir de la liste, sélectionnez *Nouveau droit...*



FIGURE 3.7 – Écran de saisie des droits associés à une application

Le nom du droit doit être celui défini dans le corps de l'application (les droits sont positionnés dans les fichiers *param/actions.xml*, qui contient la liste des modules utilisables, et *param/menu.xml*, qui sert à générer le menu – cf. table 3.1 *Droits à positionner*, page 20).

Indiquez les groupes d'utilisateurs qui seront associés au droit courant.

3.1.7 Cas particulier des groupes et des logins issus d'un annuaire LDAP

Si vous avez paramétré l'application pour qu'elle s'appuie sur un annuaire LDAP pour gérer l'affectation des utilisateurs dans les groupes, vous n'êtes pas obligés de les déclarer explicitement dans le module de gestion des droits.

Droits attribués à un groupe LDAP

Tous les utilisateurs d'un groupe héritent d'un droit dans l'application.

- définissez le nom du groupe (en respectant la casse) dans le tableau des groupes d'utilisateurs (par exemple, EABX) ;
- sélectionnez le nom de ce groupe dans les droits utilisables ;
- tous les utilisateurs de l'annuaire LDAP récupéreront automatiquement les droits attribués à ce groupe.

Droits attribués à un utilisateur particulier de l'annuaire LDAP

Un utilisateur s'identifie auprès de l'annuaire LDAP, mais dispose de droits particuliers.

- créez son login dans la gestion des droits ;
- rajoutez-le dans le groupe d'utilisateurs adéquat.

3.2 Droits spécifiques de l'application Filo-Science

3.2.1 Droits à positionner

Voici les droits nécessaires pour faire fonctionner correctement l'application :

Droit	Usage
admin	Gestion des utilisateurs, des droits, des paramètres globaux de l'application
param	Gestion des paramètres, création des projets, etc.

Droit	Usage
gestion	Création d'un protocole, saisie des opérations de pêche, pour les projets rattachés au groupe de l'utilisateur
consult	Droit technique, permettant la consultation des informations générales, sans possibilité de modification

TABLE 3.1 – Liste des droits utilisés

Ces droits doivent être définis pour chaque application (couple *logiciel* \leftrightarrow *base de données*) gérée par la base de gestion des droits.

3.3 Gestion des traces

Tous les appels lancés par les utilisateurs vers les modules de l'application sont enregistrés dans la table *gacl.log*, qui ne doit être accessible qu'aux personnes dûment autorisées. Les traces sont supprimées au bout d'un an (script de nettoyage exécuté lors de la connexion d'un utilisateur).

Voici un exemple de trace générée :

```
log_id login nom_module log_date commentaire ipaddress
1 admin filo-operationDisplay 2019-04-30 17:23:01 ok 127.0.0.1
2 admin filo-projectList 2019-04-30 17:23:05 ok 127.0.0.1
3 admin filo-projectChange 2019-04-30 17:23:07 ok 127.0.0.1
4 admin filo-projectWrite 2019-04-30 17:23:15 ok 127.0.0.1
5 admin filo-project-write 2019-04-30 17:23:15 1 127.0.0.1
```

La colonne *commentaire* précise des informations concernant soit la connexion, soit l'écriture en base de données : dans ce cas, l'identifiant considéré est indiqué. L'adresse IP est en principe celle de l'utilisateur, y compris en prenant en compte le passage par un serveur Reverse-proxy².

Parallèlement, les messages d'erreur sont envoyés au processus Linux SYSLOG, qui enregistre les traces dans le fichier */var/log/apache2/error.log*.

2. serveur mis en entrée du réseau privé, qui permet de masquer les adresses internes et de contrôler les accès depuis Internet

Chapitre 4

Importations et exportations de données

4.1 Présentation

Le logiciel Filo-Science intègre deux modules génériques permettant de réaliser soit des importations, soit des exportations. Le premier vise à importer des données de pistage de poissons ou des données transmises par des sondes d'analyse physico-chimique. Les données fournies sont de type tabulé (fichiers ressemblant à des fichiers CSV). Le second sert à générer un export de données au format JSON, pouvant comprendre également les informations rattachées, pour pouvoir les réimporter dans une autre base de données.

Les tables qui permettent de décrire ces fonctions sont stockées dans le schéma *import* (figure 4.1).

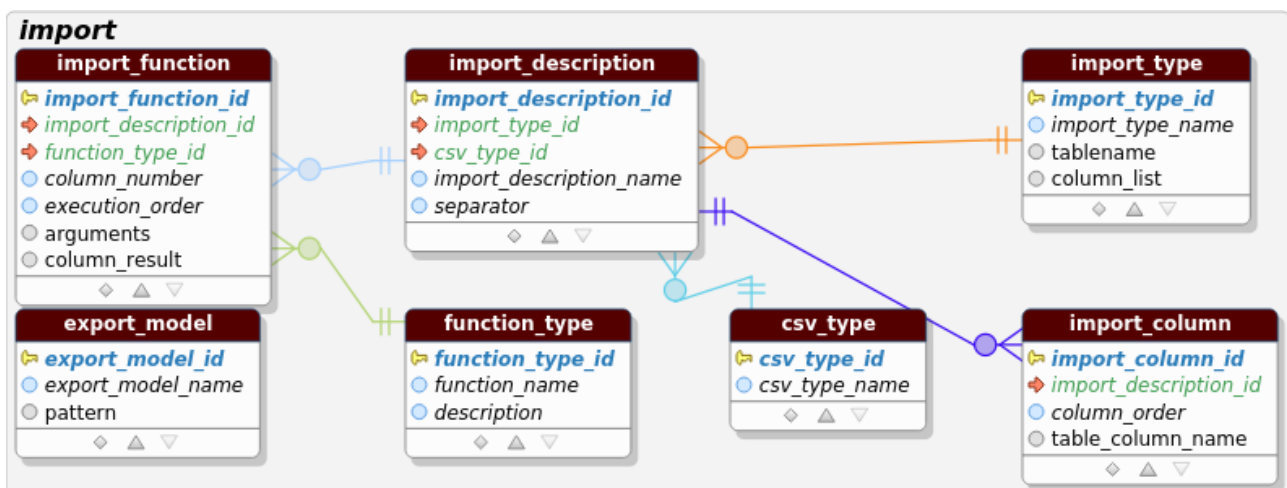


FIGURE 4.1 – Schéma des tables utilisées pour décrire les importations/exportations

4.2 Importation de données tabulées

Les structures des fichiers fournis par les systèmes d'acquisition automatique sont très variées, les données nécessitent souvent d'être reformatées ou recalculées avant de pouvoir être importées dans une table.

Le principe d'importation est le suivant :

- chaque champ (une colonne) du fichier à importer peut faire l'objet d'une ou plusieurs transformations ou contrôles ;
- une fois l'ensemble de ces opérations réalisées, les champs pertinents vont être assignés à des colonnes de la table cible, et l'enregistrement réalisé en base de données.

4.2.1 Types d'import

Trois types d'import sont décrits et fournis, qui permettront d'alimenter les tables *detection*, *probe_measure* et *location*. La liste des colonnes utilisables pour l'importation est décrite, chaque colonne étant séparée par une virgule.

Les types d'import ne doivent pas être modifiés.

4.2.2 Types de fonctions

Les fonctions, codées dans l'application, sont décrites dans la table *function_type*. Cette table ne peut pas être modifiée.

La table 4.1 présente la liste des fonctions utilisables pour préparer un import.

TABLE 4.1 – Liste des fonctions utilisées pour préparer un import.

Nom de la fonction	Description	Contrôle ?	Transformation ?
testValue	Teste si un champ contient la valeur indiquée	X	
getSecondsFromTime	Transforme un champ de type hh:mm:ss.u en ss.u		X
extractRightChar	Extrait les n caractères à droite du champ		X
concatenateDateAndTime	Concatène un champ date et un champ time. L'argument doit correspondre au numéro de la colonne time		X
transformJulianToDate	Transforme un nombre de jours depuis la date indiquée en référence (au format Y-m-d) en date exploitable au format Y-m-d		X
verifyTypeNumber	Vérifie si une valeur est numérique ou non	X	
testColumnsNumber	Vérifie que le nombre de colonnes de la ligne courante correspond bien au nombre attendu		X
getIndividualFromTag	Récupère l'identifiant du poisson à partir du tag (RFID)		X
getIndividualFromTransmitter	Récupère l'identifiant du poisson à partir du transmetteur (radio ou acoustique)		X
numericToHexa	Transforme une valeur numérique en valeur Hexa, si celle-ci ne l'est pas. La valeur Hexa doit comprendre au moins une lettre.		X
Suite sur la page suivante			

TABLE 4.1 – suite de la page précédente

Nom de la fonction	Description	Contrôle ?	Transformation ?
concatenate	Associe le contenu de colonnes ou du texte.		
matchingCode	L'argument doit être au format JSON, sous la forme : ["type" : "col", "val" : 4, type : "str", "val" : " :"] Remplace la valeur courante par une autre valeur, définie dans un argument JSON au format : "valueSearched" : correspondingValue, "2ndvalue" : corresp2. Pour la recherche des paramètres de sonde, valueSearched doit correspondre au libellé utilisé par la sonde, et correspondingValue à la valeur de la clé dans la table des paramètres		X
formatDateTime	Transforme une datetime dans un format reconnu par la base de données, à partir du format indiqué. Exemple : d/m/Y H :i :s pour une date de type 13/01/2019 08 :50 :00. La liste des formats autorisés est disponible ici : https://www.php.net/manual/fr/date-time.createfromformat.php		X
decodeAll	Transforme un jeu de caractère particulier en UTF-8. L'argument doit comprendre le jeu de caractère à transcoder, par exemple UTF-32. C'est l'ensemble de la ligne qui sera transcodé		X
transformDecimalSeparator	Transforme la virgule en point, pour les champs décimaux en français		X

4.2.3 Créer un modèle d'import

Ouvrez le menu *Téledétection > Paramètres > Modèles d'import*, puis cliquez sur le lien *Nouveau modèle*. Les informations à renseigner sont les suivantes :

— Type d'import :

- Détection : données de détections enregistrées depuis une station fixe
 - Données de sonde : relevés physico-chimiques réalisés par une sonde
 - Détections manuelles : données de détection réalisées sur le terrain.
 - Type de fichier CSV :
 - Data in columns : format classique des fichiers CSV. Une colonne contient toujours la même information
 - Data in lines : chaque ligne décrit le paramètre relevé et la valeur associée
 - Nom du modèle : nom mnémotechnique
 - Séparateur de champs : indiquez le séparateur utilisé (point-virgule, virgule, tabulation, espace).
- Une fois ces informations enregistrées, deux tableaux doivent être renseignés.

Fonctions de test et de transformation

Le premier contient la liste des fonctions de transformation ou de test à exécuter. Voici les paramètres à préciser pour chacune :

- Nom de la fonction à exécuter : une fois sélectionnée, un texte décrivant son fonctionnement et les paramètres attendus est affiché
- N° de colonne à traiter : indiquez le numéro de la colonne sur laquelle va porter la fonction. Si la fonction porte sur plusieurs colonnes (concaténation, par exemple), indiquez le numéro de la première colonne
- N° de la colonne récupérant le résultat : s'il s'agit d'une fonction de transformation, indiquez dans quelle colonne le résultat va être stocké. Pour les fonctions de contrôle, indiquez la valeur 0 (aucun résultat n'est stocké)
- ordre d'exécution de la fonction : pour une même colonne, plusieurs fonctions peuvent être appelées successivement. Renseignez l'ordre d'exécution pour être sûr que le résultat correspondra à ce que vous attendez
- argument complémentaire : le cas échéant, indiquez la valeur attendue par la fonction pour s'exécuter. Le détail de la valeur est décrit dans l'explication de la fonction.

Si une fonction de contrôle échoue, la ligne ne sera pas traitée et sera indiquée comme telle lors de l'importation.

Il est possible d'exécuter plusieurs fonctions successives sur la même colonne.

À noter que certaines fonctions sont utilisées pour rechercher des identifiants présents dans la base de données à partir des informations fournies (numéros des poissons à partir des numéros des émetteurs, par exemple).

Table d'équivalence entre les colonnes et les champs de la base de données

Le second tableau à renseigner permet d'indiquer quelles colonnes doivent être insérées dans la base de données. Voici les informations à indiquer :

- le numéro de la colonne, telle qu'elle a été transformée après application de l'ensemble des fonctions décrites précédemment
- le nom du champ dans la base de données qui recevra l'information
- s'il s'agit d'un fichier où chaque élément est différent d'une ligne à l'autre (fichier de type Entité/relation), indiquez également si :
 - le champ sert d'identifiant pour la valeur mesurée
 - le champ contient la valeur mesurée

4.2.4 Exécuter un import

Choisissez le menu *Téledétection > Importation*, puis indiquez :

- le fichier à importer

- le type d'import à réaliser
- le projet concerné
- s'il s'agit d'un import de données réalisées à partir d'une station fixe (antenne ou sonde), indiquez l'antenne ou la sonde concernée
- le mode *test* permet d'exécuter toutes les opérations d'importation, sans réaliser la mise en base de données. Cela permet d'identifier les problèmes potentiels dans un fichier avant de réaliser l'opération de manière définitive
- en mode *test*, le nombre de lignes à traiter doit être indiqué, ce qui permet de limiter la taille de l'échantillon à tester.

En mode *test*, deux tableaux sont produits : le premier présente les données prêtes à être importées, le second l'ensemble des messages d'erreur.

En mode normal, le premier tableau affiche le résultat de l'importation. Celui contenant les messages d'erreurs est identique.

4.3 Export de données au format JSON

Il est possible d'exporter un lot de données, réparties sur plusieurs tables, pour les réimporter dans une autre base de données. Par défaut, trois exports sont disponibles et intégrés directement dans l'application : export d'une campagne, d'une opération et de leurs données associées, et un export technique des modèles d'exports.

Les données sont exportées au format JSON.

4.3.1 Description des modèles d'export

Choisissez le menu *Paramètres > Modèles d'export*. Hormis le nom du modèle, toutes les autres informations permettent de décrire les tables à exporter et les relations entre elles.

La saisie d'une nouvelle table passe par l'ajout d'un nouvel item dans la partie *Description du modèle*. Chaque item peut être déplacé après création, si nécessaire.

Pour chaque table à exporter, voici les informations à renseigner :

- nom de la table, telle qu'il figure dans la base de données
- alias de la table : si une même table peut être reliée à des tables parentes différentes, cette colonne devra être renseignée avec un nom différent pour chaque instance. C'est le cas notamment pour la table *ambiance*, qui peut être reliée soit à la table *operation*, soit à la table *sequence*.
- clé primaire : indiquez la clé primaire utilisée dans la table. Elle ne doit pas être renseignée dans le cas d'une table porteuse d'une relation n-n, dont la clé est composée des clés des deux tables parentes (cas de la table *operation_operator*).
- clé métier : il s'agit de la colonne qui permet de retrouver de manière univoque un enregistrement dans la table. Selon les cas de figure, il peut s'agir :
 - du libellé, pour une table de paramètres
 - de la clé primaire elle-même, pour certaines tables de paramètres dont la clé est significative. Cela permet de conserver la valeur de cette clé, même si le libellé change
 - du champ UUID, qui est un identifiant technique généré avec un algorithme garantissant qu'il est unique au niveau mondial. Cette valeur sera utilisée chaque fois qu'elle est disponible
- clé étrangère : le nom du champ porteur de la relation avec le parent, qui contient donc la clé du parent
- la liste des champs de type booléen, en raison d'une particularité lors des importations liée à ceux-ci
- la liste des alias (ou du nom des tables) des tables filles

- dans le cas d’une table porteuse d’une relation n-n, c’est à dire dont la clé est composée des clés des deux tables parentes, il faudra indiquer :
 - le nom du champ comprenant la seconde clé étrangère
 - l’alias (ou le nom de la table) de la seconde table parente

La première table présente dans la liste doit être la table principale de l’export. Les tables filles doivent être créées après leurs tables parentes.

Il est possible d’indiquer plusieurs tables principales (sans parents) dans le même modèle.

4.3.2 Exporter des campagnes ou des opérations

Depuis la liste des campagnes, ou depuis une campagne, cochez les lignes que vous voulez exporter, puis cliquez sur le bouton d’exportation. Le fichier JSON sera généré.

4.3.3 Autres exportations

Depuis la liste des modèles, il est possible de réaliser un export pour l’ensemble des données décrites dans un modèle.

Dans la liste, affichez le détail d’un des modèles, puis cliquez sur le bouton *Exporter toutes les données concernées par le modèle*.

Attention : c’est une opération qui va traiter l’ensemble d’une table et des données rattachées. Elle est à manipuler avec précaution, le volume des données générées peut être important.

4.3.4 Importer un fichier JSON

L’importation de campagnes ou d’opérations s’effectue au même endroit que leur exportation. Il suffit de sélectionner le fichier considéré pour qu’il soit importé.

Pour ces deux importations, le programme va travailler en mode « remplace ou insert » : si un enregistrement est trouvé (à partir du champ métier), il est mis à jour, sinon il est créé.

Il est également possible de réaliser une importation rapide depuis le menu de création des modèles d’exportation, depuis le détail d’un modèle. Cette fonction peut être pratique pour mettre à jour des tables de référence.

Chapitre 5

Utiliser Qgis avec Filo-Science

5.1 Présentation

Le module de télédétection des poissons peut s'interfacer facilement avec Qgis, pour corriger le positionnement d'une station d'écoute ou de mesure de paramètres physico-chimiques, ou pour enregistrer la détection des poissons, quand celle-ci est faite manuellement le long du cours d'eau.

Pour que la mise à jour des informations puisse fonctionner, il faut toutefois utiliser les bonnes tables ou vues, prévues à cet effet dans la base de données.

5.2 Utilisation avec Qgis

La table 5.1 récapitule les tables et vues utilisables avec Qgis, soit pour visualiser des informations, soit pour les modifier.

TABLE 5.1 – Liste des tables et des vues utilisables avec Qgis.

Nom de la table	Type	Description
v_station_tracking	vue	Liste des stations d'enregistrement des poissons. La liste peut être limitée au projet en rajoutant les critères de sélection adéquats.
v_individual_tracking	vue	Liste des poissons pouvant être détectés. Ceux-ci devraient être filtrés sur le projet.
location	table	Liste des détections de poissons. La liste devrait être filtrée sur les poissons présents dans le projet courant et, pour de nouvelles détections, en rajoutant un filtre sur la date (postérieure aux anciennes détections pour le projet considéré)
antenna_type	table	Liste des types d'antennes, pour affichage dans les formulaires
project	table	Liste des projets, pour affichage dans les formulaires. La table devrait être filtrée sur le projet courant.
station_type	table	Liste des types de stations, pour affichage dans les formulaires

Il est possible de modifier ou de créer une station depuis Qgis et de saisir une localisation manuelle d'un poisson. Deux formulaires peuvent être créés à cet effet :

5.2.1 Formulaire Station

Le formulaire travaille avec la vue *v_station_tracking*.

Champs à afficher :

- station_name
- station_long
- station_lat
- station_pk
- station_type_id
- project_id

Caractéristiques particulières de certains champs :

- station_name : éditable
- station_long : éditable, valeur par défaut : \$x, et cocher *Appliquer la valeur par défaut sur la mise à jour*
- station_lat : éditable, valeur par défaut : \$y, et cocher *Appliquer la valeur par défaut sur la mise à jour*
- pk : éditable
- station_type_id : éditable, type d'outil : *Liste de valeurs*, Charger des données depuis la couche *station_type*, contraintes : *non nul*, valeur par défaut : 2 (à adapter le cas échéant)
- project_id : éditable, type d'outil : *Liste de valeurs*, charger des données depuis la couche *project*, valeur par défaut : le numéro du projet

5.2.2 Formulaire Localisation manuelle des poissons

Le formulaire travaille avec la table *location*.

Champs à afficher :

- detection_date
- individual_id
- antenna_type_id
- project_id
- location_long
- location_lat
- signal_force
- observation

Caractéristiques particulières de certains champs :

- antenna_type_id : éditable, type d'outil : *Liste de valeurs*, charger des données depuis la couche *antenna_type*
- location_pk : éditable, type d'outil : *édition de texte*
- location_long : éditable, valeur par défaut : \$x, et cocher *Appliquer la valeur par défaut sur la mise à jour*
- location_lat : éditable, valeur par défaut : \$y, et cocher *Appliquer la valeur par défaut sur la mise à jour*
- individual_id : éditable, type d'outil : *Liste de valeurs*, charger les données depuis la couche *v_individual_tracking*. Dans cette couche, chargez comme valeur *individual_id*, et comme description soit *tag*, soit *transmitter*, selon le type de détection réalisé
- signal_force : éditable, type d'outil : *plage, éditable, pas 1*
- observation : éditable, type d'outil : *édition de texte*
- detection_date : éditable, type d'outil : *Date/Heure*, Format du champ : *Date & Heure*, Affichage : *Personnalisation* : *dd/MM/yyyy HH:mm:ss*, cocher *Calendrier*, valeur par défaut : *now()*

5.2.3 Ajouter un fond de carte OpenStreetMap

Dans l’explorateur, ajoutez une nouvelle connexion *XYZ Tiles*, avec les paramètres suivants :

- Nom : OpenStreetMap
- URL : <https://tile.openstreetmap.org/z/x/y.png>
- Niveau de zoom min. : positionnez à 5 (c’est en général suffisant)
- Niveau de zoom max. : 19

Ajoutez ensuite la couche au projet.

5.3 Utilisation en mode embarqué

Il est possible de charger le projet dans une tablette android, pour permettre la saisie directe sur le terrain, puis de le synchroniser ensuite avec la base de données. La solution proposée est basée sur le produit QField (<https://qfield.org>).

Une documentation complète de la configuration d’un projet est présente sur le site de QField.

QField ne fonctionne qu’avec des objets géographiques de type *point*. Il est adapté au positionnement d’un point sur une carte à partir du GPS intégré ou connecté.

5.3.1 Installer l’extension QFieldSync

Pour faciliter les échanges avec QField, il est préférable d’installer l’extension QFieldSync, disponible dans les extensions Qgis.

Pour préparer le projet, plusieurs étapes sont nécessaires. Elles sont accessibles depuis le menu *Extension > QFieldSync*.

À partir du menu *Préférences*, définissez les chemins par défaut de stockage des projets. Les dossiers d’exportation (vers la tablette) et d’importation (depuis la tablette) sont volontairement différents, pour éviter d’écraser par erreur une saisie réalisée sur le terrain.

Dans la configuration du projet, indiquez, pour chaque couche, comment elle sera gérée : *édition hors ligne*, *supprimée du projet* ou *aucune action*. Toutes les couches qui pourront être modifiées doivent être positionnées en *édition hors ligne*. À noter que la couche *OpenStreetMap* doit être de type *Aucune action*. Indiquez également que vous créez un fond de carte à partir soit du thème de la carte (si vous en avez créé un), soit à partir de la couche OpenStreetMap. Sauf si vous savez ce que vous faites, ne modifiez pas la taille des tuiles ou les unités de carte par pixel.

Enfin, depuis le menu, choisissez l’option *Paquet pour QField*. Le fond de carte va être généré à la taille de la fenêtre : redimensionnez votre projet pour qu’il inclut toute la zone concernée. Dans le cas contraire, vous n’aurez pas accès au fond de carte. Le projet sera alors créé dans le dossier défini préalablement.

5.3.2 Installer QField dans une tablette Android

QField est disponible dans le magasin d’applications de Google, l’installation est sans difficulté particulière.

Une fois Qfield installé, il faut recopier le projet Qgis (l’ensemble du dossier généré), dans le dossier *files* de l’application.

Pour cela, connectez la tablette à votre ordinateur avec le câble USB, et autorisez (sur la tablette) l’ordinateur à accéder aux données.

Attention : il existe deux emplacements possibles, l’un dans la tablette elle-même, l’autre dans la carte externe. Il faut impérativement recopier le dossier dans la carte externe. Avec une tablette Samsung, le chemin où déposer les fichiers est ici : *Card/Android/data/ch.opengis.qfield/files*

Il faut ensuite lancer l’application QField, puis ouvrir le projet précédemment recopié.

Une fois les saisies réalisées, pensez à bien fermer l'application QField pour être sûr que toutes les modifications ont bien été sauvegardées. Recopiez ensuite le dossier contenant les données dans le dossier *import* de QFieldSync.

Relancez Qgis, ouvrez le projet que vous venez de récupérer, puis lancez la mise à jour de la base de données depuis le menu : *Extension > QFieldSync > Synchroniser depuis QField*.

Chapitre 6

Fonctionnement en mode autonome

6.1 Présentation

L'application Filo-Science peut fonctionner en mode autonome, c'est à dire sans connexion internet. Le principe est décrit dans la figure 6.1.

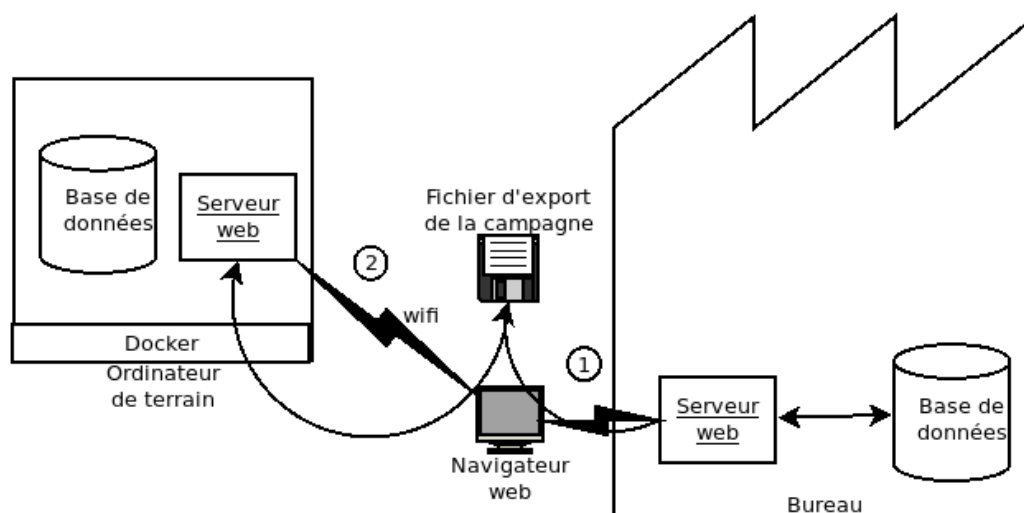


FIGURE 6.1 – Schéma général de fonctionnement du mode autonome.

Dans un premier temps, les informations générales sur la campagne sont saisies dans l'application centrale (bureau). La campagne est ensuite exportée dans un fichier au format JSON.

Dans un second temps, l'application est installée dans une machine destinée à aller sur le terrain (ordinateur portable Windows, ou couple Raspberry¹ – tablette Android). Un navigateur se connecte au serveur web embarqué qui contient une copie de l'application. À partir de celle-ci, il suffit d'importer le fichier généré précédemment pour récupérer les paramètres de la campagne.

La saisie s'effectue alors à partir du système autonome. Une fois la campagne terminée, il suffit de réaliser les opérations dans l'autre sens : les saisies réalisées sur le terrain sont exportées au format JSON depuis l'application embarquée, puis importées dans l'application centrale.

1. Les Raspberry sont des ordinateurs de très petite taille (ils tiennent dans une poche), fonctionnant sans écran ni clavier, destinés à être utilisés un peu partout, et peu chers (quelques dizaines d'euros, moins de 150 euros avec une batterie). Ils fonctionnent avec un système d'exploitation basé sur Linux, et embarquent un émetteur wifi permettant de connecter une tablette, par exemple.

6.2 Installer l'application et la base de données dans un matériel autonome

L'installation a été prévue pour utiliser un composant particulier, *Docker*, qui permet d'installer n'importe quel système sur n'importe quel autre, tout en gardant le paramétrage particulier du premier.

L'ensemble de la procédure d'installation, à la fois de Docker, de l'application et de la base de données est décrite (en anglais et en français) dans un projet *Github* : <https://github.com/Irstea/filo-docker>. Des instructions particulières concernent l'installation dans un Raspberry, avec l'activation du wifi pour pouvoir connecter une tablette.

Bibliographie

- [1] ANSSI. Recommandations de sécurité relatives à tls, 2016. URL http://www.ssi.gouv.fr/uploads/2016/09/guide_tls_v1.1.pdf.
- [2] Apache-FOP. The apache fop project, 2016. URL <http://xmlgraphics.apache.org/fop/>.
- [3] APP. Agence pour la protection des programmes, 2016. URL <http://www.app.asso.fr>.
- [4] Mike Benoit and Dan Cech. Phpgacl, generic access control lists, 2006. URL <http://phpgac1.sourceforge.net>.
- [5] bootstrap. Bootstrap is the most popular html, css, and js framework for developing responsive, mobile first projects on the web, 2016. URL <http://getbootstrap.com>.
- [6] Justin Ellingwood. How to set up master slave replication on postgresql on an ubuntu 12.04 vps, 2013. URL <https://www.digitalocean.com/community/tutorials/how-to-set-up-master-slave-replication-on-postgresql-on-an-ubuntu-12-04-vps>.
- [7] Free Software Foundation. Gnu affero general public license, 2007. URL <https://www.gnu.org/licenses/agpl.html>.
- [8] Nils Hamerlinck. Configurer la rÉplication d'un serveur postgresql, 2015. URL <http://connect.ed-diamond.com/GNU-Linux-Magazine/GLMF-184/Configurer-la-replication-d-un-serveur-PostgreSQL>.
- [9] JQuery. Site officiel, 2015. URL <http://jquery.com/>.
- [10] Stanislas Lange. Installer php 7 sous debian 8 jessie via le dépôt dotdeb, 2016. URL <https://angristan.fr/installer-php-7-debian-8-jessie-depot-dotdeb/>.
- [11] Mozilla. Mozilla ssl configuration generator, 2016. URL <https://mozilla.github.io/server-side-tls/ssl-config-generator/>.
- [12] OWASP. Application security verification standard (2014), 2014. URL https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf.
- [13] OWASP. Site institutionnel, 2015. URL <https://www.owasp.org>.
- [14] OWASP. Zed attack proxy project, 2015. URL https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project.
- [15] pgAdmin. pgadmin postgresql tools, 2016. URL <https://pgadmin.org>.
- [16] postgresql. Binary replication tutorial, 2015. URL https://wiki.postgresql.org/wiki/Binary_Replication_Tutorial.

- [17] Eric Quinton. Documentation d'utilisation du framework prototype php, 2016. URL <https://github.com/equinton/prototypephp/blob/bootstrap/database/documentation/prototypephp-documentation.pdf>.
- [18] Eric Quinton. Prototypephp, 2016. URL <https://github.com/equinton/prototypephp/tree/bootstrap>.
- [19] Eric Quinton. Ré-identification par jeton, 2016. URL <http://www.linux-professionnel.net/programmation/php---codes-divers/re-identification-par-jeton>.
- [20] Greg Reinacker. Zero to postgresql streaming replication in 10 mins, 2013. URL <http://www.rassoc.com/gregr/weblog/2013/02/16/zero-to-postgresql-streaming-replication-in-10-mins>.
- [21] smarty. Smarty, php template engine, 2016. URL <http://www.smarty.net>.
- [22] The Gimp Team. Gimp - gnu image manipulation program, 2018. URL <https://www.gimp.org/>.