

4 Table des matières

1	Résumé	I
	Mots-clés	I
2	Avant-propos	II
2.1	Contexte	II
2.2	Démarche	III
2.3	Convention typographique	III
3	Remerciements	IV
4	Table des matières	V
5	Liste des abréviations	XII
6	Définitions	XIV
7	Introduction	1
8	Organisation	2
8.1	Méthodologie	2
	Sprint 0	2
	Sprint 1	2
	Sprint 2	2
	Sprint 3	2
	Sprint 4	2
	Sprint 5	3
	Sprint 6	3
	Sprint 7	3
	Sprint 8	3
8.2	Journal de travail	3
9	Recherche	4
9.1	Résumé	4
9.2	ASP.NET	5

Connexion à SQL Server	5
Génération des modèles	5
Coût de la licence	5
9.3 Django	6
Connexion à MS SQL Server	6
Génération des modèles	6
Avantages	6
Désavantages	6
9.4 Symfony	7
Connexion à Microsoft (MS) SQL Server	7
Génération des modèles	7
Avantages	8
Désavantages	8
9.5 Proposition finale	9
10 Analyse	10
10.1 L'intranet	10
10.2 La base de données	10
10.3 Ancienne interface	11
11 Architecture	12
11.1 Environnement	12
11.2 Gestion du code source	12
11.3 Intégration Continue	12
12 Modélisation	13
12.1 Use Case	13
12.2 Diagramme de Classes	14
12.3 Diagramme réseau	15
13 Environnement de développement	16
13.1 Conventions de codage	16
13.2 Installation de SQL Server 2012	16

13.3	Configuration de SQL Server 2012	17
14	Environnement de déploiement	18
14.1	Docker	18
14.2	Doctrine	19
14.3	Symfony	20
14.4	Bootstrap	20
15	Importation des Entités	21
15.1	Notes	21
15.2	Génération	21
15.3	Liaisons non définies	22
15.4	Nommage des colonnes	23
15.5	Entité Demod	23
16	Formulaires	24
16.1	Génération automatisée	24
16.2	Chargement dynamique	24
16.3	Contrôle des champs	26
17	Create, Read, Update, Delete (CRUD)	27
18	Gestion des utilisateurs	28
18.1	Friends Of Symfony (FOS) User Bundle	28
18.2	Module de Sécurité	30
19	Toolbox	31
19.1	Node.js®	31
19.2	Whatsup	31
19.3	Installation	32
20	Optimisations	34
21	Héritage	35
22	Openweathermap	37

23 Méthodes généralisées	38
23.1 Classes abstraites	38
23.2 Automator	38
23.3 Twig	39
24 Ajax	40
25 Déploiement	41
25.1 Choix du système	41
Avantages	41
26 Documentation	43
27 Conclusion	45
27.1 Améliorations possibles	46
28 Références	47
Annexe I : Tendances des emplois aux Etats-Unis	49
Annexe II : Droit d’auteur	50
Annexe III : Journal de travail	51
Annexe IV : Product Backlog	58
Annexe V : Product Backlog	59
Annexe VI : Product Backlog	60
28 Déclaration sur l’honneur	61

Liste des tableaux

1	Récapitulatif des frameworks.	4
2	Prix de Visual Studio Professional.	5
3	Roadmap des versions de Symfony (Wikipedia)	20

Table des figures

1	Type de création de modèle dans Visual Studio (Samuel Dorsaz)	5
2	Générer les modèles à partir d'une base de données (Samuel Dorsaz)	6
3	Connexion à SQL Server à travers Doctrine	7
4	Générer les métadonnées de la BDD	7
5	Générer les entités avec Doctrine	7
6	Exemple de contenu d'une entité générée avec Doctrine	8
7	Deux visuels de l'ancienne interface tirée des procédures d'utilisation (New-sat)	11
8	Use Case (Samuel Dorsaz)	13
9	Diagramme de Classes (Samuel Dorsaz)	14
10	Diagramme réseau (Dorsaz Samuel)	15
11	Erreur de localisation de SQL Server (Samuel Dorsaz)	16
12	Configuration du port SQL Server (Samuel Dorsaz)	17
13	Exécute les instructions du fichier docker-compose.yml en mode service (-d)	18
14	Exécuter un terminal (-t) interactif (-i) de type bash	18
15	Ajouter la dépendance au module "leaseweb/doctrine-pdo-dlib"	19
16	Demander la signification d'un code d'erreur à SQL Server	21
17	Générer les entités par rapport à la base de données existante	21
18	Ajouter une contrainte de clef secondaire cause une erreur	22
19	Retrouver des identifiants manquants dans une liaison.	22
20	Utilisation du dièse avec Microsoft SQL Server (Samuel Dorsaz)	23
21	Exemple de génération automatisée du formulaire de la classe « Spectrum »	24
22	Tentative de chargement dynamique des classes	25
23	Résolution du chargement dynamique des classes	25
24	Exemple simple de l'utilisation d'un « Validator » pour contrôler l'adresse Internet Protocol (IP)	26
25	Résumé du processus de génération CRUD à partir d'une entité	27
26	Résultat de doctrine afin de mettre à jour la table User	29
27	Message d'erreur de l'initialisation de la boîte à outils (Samuel Dorsaz) . .	32
28	Requête SQL native au travers d'une connexion différente	32

29	Résultats des optimisations de la barre latérale (Samuel Dorsaz)	34
30	Héritage important au centre du système (Samuel Dorsaz)	35
31	Exemple d'héritage par table de classes	36
32	Code de jointure d'une sous-classe dans Doctrine.	36
33	Comparaison avec et sans le service Automator (Samuel Dorsaz)	39
34	Accès à la documentation du projet	44
35	Tendance du marché du travail aux Etats-Unis	50
36	Les icônes ajoutées au projet	51

5 Liste des abréviations

ACL Access Control List

API Application Programming Interface

ASP Active Server Pages

BDD Base De Données

BSD Berkeley Software Distribution

CE Community Edition

CI Countinous Integration

CRUD Create, Read, Update, Delete

CSS Cascading Style Sheets

DBAL Database Abstraction Layer

DNS Domain Name System

DQL Doctrine Query Language

FOS Friends Of Symfony

GNU GNU is Not Unix

HES-SO Haute Ecole Spécialisée de Suisse Occidentale

HTML HyperText Markup Language

HTTP Hypertext Transfer Protocol

IP Internet Protocol

ISO International Organization for Standardization

LTS Long Term Support

MIT Massachusetts Institute of Technology

MS Microsoft

MVC Modèle Vue Contrôleur

ORM Object-Relational Mapping

OS Operating System

PC Personal Computer

PEP Python Enhancement Proposal
PHP PHP Hypertext Preprocessor
PSR PHP Standards Recommendations
RMA Return Merchandise Agreement
SA Société Anonyme
Sàrl Société à responsabilité limitée
SCM Source Code Management
SGBD Système de Gestion de Base de Données
SI Système d'Information
SMS Short Message Service
SP Service Pack
SQL System Query Language
SSH Secure Shell
SSMS SQL Server Management Studio
TCP Transmission Control Protocol
TDS Tabular Data Stream
USB Universal Serial Bus
VPN Virtual Private Network
W3C World Wide Web Consortium
WUG WhatsUp Gold
XML Extensible Markup Language

6 Définitions

Action Symfony définit les actions comme des processus d'un contrôleur (en terme de Modèle Vue Contrôleur (*MVC*)). Ce sont les réactions du système par rapport à une action de l'utilisateur.

Entité Symfony considère l'Entité comme étant un modèle (en terme de *MVC*). C'est un objet métier qui représente un élément du système d'information de l'entreprise.

Framework Une base de travail structurée pour un langage et un champ d'application défini. Ils permettent de rassembler les développeurs autour de bonnes pratiques communes et évitent de redévelopper continuellement des outils généralistes.

Les licences permissives Ces licences sont très proches du domaine public. Elles ne requièrent que la mention du copyright initial. Elles offrent aussi la possibilité d'écrire un code propriétaire à partir d'un code publié sous ce type de licence (« Licences libres | AFUL », 2015). Les licences permissives les plus répandues sont les licences Berkeley Software Distribution (*BSD*), Massachusetts Institute of Technology (*MIT*) et Apache (« Open Source License usage on GitHub.com », 2015).

npm Ce logiciel permet de gérer les dépendances dans le cadre du développement d'une application Node.js[®]. J'ai cherché un acronyme lui correspondant mais le nom de npm est un rétroacronyme récursif qui veut dire « npm is not an acronym » (IHRIG, 2013).

Scrum Cette méthode de développement moderne permet un meilleur taux de réussite sur la réalisation d'un projet et surtout une bonne communication avec le client. Elle est organisée en « Sprints » qui sont des périodes régulières de livrables et de rendez-vous. Ces livraisons se font par rapport à un « Product Backlog » initialement défini avec le client, avant le début du projet.

WhatsUp Gold (WUG) Un service Web permettant de monitorer l'état de santé d'un réseau en analysant différents protocoles (Domain Name System (*DNS*), Transmission Control Protocol (*TCP*)/*IP*, telnet, Microsoft Exchange ...).

7 Introduction

Toutes les entreprises ont besoin de centraliser certaines informations quel que soit leur domaine d'activité. Il existe une multitude d'outils sur le marché et il arrive même que certaines sociétés en fassent développer sur mesure.

J'avais été approché, il y a trois ans, par une entreprise Sédunoise pour cette raison. Cette entreprise avait finalement décidé de travailler avec un prestataire de service ayant suffisamment ressources disponible dans le temps. Mais entre-temps, mes recherches m'ont montré que certains systèmes deviennent rapidement des « usines à gaz » lorsqu'ils se veulent trop généraliste.

A l'époque, je n'avais certainement pas toutes les compétences pour prendre en mains un projet de cette taille. Lorsque j'ai commencé à chercher un sujet pour mon travail de Bachelor, le cas de Newsat me semblait tout à fait intéressant. Ils avaient déjà une base technique à exploiter mais un intranet à revoir entièrement. C'était l'occasion pour moi de voir jusqu'où je pourrais aller en deux mois de développement dans le domaine.

L'une des contraintes de ce développement est de garder la base de données de l'ancien projet. Cette contrainte est intéressante car elle va me permettre d'expérimenter un cas réel de génération de code source à partir de la structure de son système et me permettre de voir à quel point le Framework que je vais utiliser se montrera flexible aux contraintes qu'on peut lui imposer. Après tout, la majorité des entreprises sont dépendantes d'anciens système interconnectés.

8 Organisation

8.1 Méthodologie

Dans un premier temps, j'ai tenté une approche Scrum mais en étant seul développeur, cela n'avait pas de sens et demandait trop d'investissement administratif. J'ai donc gardé la méthodologie Agile comme base de travail et prévu mon organisation en huit Sprints. Les directions de développement ont été prises avec le client, les éléments qui suivent ne reflètent donc pas nécessairement le Product Backlog du cahier des charges initial.

Sprint 0. Le Sprint 0 a été consacré à un état de l'art de trois des principaux Frameworks du marché afin de savoir lequel convenait le mieux au projet. Il a aussi permis d'établir un premier contact technique et de connaître les compétences en développement déjà présentes dans l'entreprise.

Sprint 1. Le premier Sprint a permis de mettre en place les éléments de gestion de code et d'intégration continue chez le développeur. Il a aussi permis d'analyser le code et la base de données déjà existante et enfin de récupérer le schéma réseau du bureau de Sion pour la mise en place de la machine de production.

Sprint 2. La première connexion avec la base de données du client a pu être établie. Les importations des modèles métiers se sont faites tant bien que mal. Une démonstration des premiers développements a été faite avec le client. L'installation de la machine en production dans le réseau va se faire durant le Sprint suivant afin de résoudre des problèmes d'intégration continue.

Sprint 3. La machine de production a été installée chez le client qui peut tester la plate-forme. Les outils de base de gestion des *Devices* et des *Site* ont déjà été implémentés et ont pu être présentés au client.

Sprint 4. Implémentation des deux menus latéraux terminée. Plusieurs tentatives de configurations de Docker avec Gitlab ont échoué au niveau du déploiement automatisé. Une installation standard d'Apache et de PHP sera fournie à la place, la priorité étant de montrer le projet.

Sprint 5. La gestion des utilisateurs est mise en place avec sa gestion des rôles. Le menu « Data-Link » est presque terminé et la notion de VSat et TLink y sont revues. Un menu « Documentation » est ajouté. Une « Toolbox » écrite en Node.js[®] fait son apparition. Une première implémentation des *Contact* est prête.

Sprint 6. La « Toolbox » est désormais déployable avec un logiciel d'installation, la maintenance est donc simplifiée. L'outil WUG a été ajouté à la « Toolbox » ce qui a tout de même demandé la connexion à deux bases de données distantes. La notion de « Terrestrial Link » fait son apparition. Les *Contact* sont désormais récupérés de trois emplacements différents.

Sprint 7. L'outil « Spectrum » à été mis en place dans la « Toolbox ». Tous les formulaires nécessaires pour gérer les *Satellite*, *Tranponder*, *Lease*, *Carrier* sont prêts.

Sprint 8. L'outil Spectrum a bénéficié d'une importante correction dans ses calculs et d'une optimisation conséquente. Un outil d'activation des *Modem* et *Demod* par Short Message Service (SMS) a été mis en place. Le menu de gestion des *Devices* non-attribués a gagné un champ de recherche dynamique et s'est vu complété de toutes les autres entités apparentées.

8.2 Journal de travail

Le projet totalise 340 heures de travail, répertoriées en annexe III. Ce travail sera complété entre le dépôt du rapport le 11 août 2016 et la défense du projet le 1er septembre 2016.

9 Recherche

Il existe des centaines de Frameworks dans une variété importante de langages. Je les ai donc volontairement filtrés selon deux types de critères : d'abord par la compatibilité avec les technologies utilisées par Newsat, puis par ma propre maîtrise des langages.

- Technologies actuellement utilisées et conservées.
 - Compatible avec Microsoft SQL Server.
- Langages maîtrisés.
 1. PHP, Python ou Active Server Pages (ASP).Net / C#.
 2. « HTML », Javascript et « Cascading Style Sheets (CSS) ».
- Outils permettant de faciliter la mise en oeuvre.
 - Architecture n-Tier.
 - Possibilité de générer automatiquement les modèles depuis une structure déjà existante en base de données.
 - Une gestion des rôles ou des Access Control List (ACL) native.
 - Eventuellement accès en tant qu'API.

9.1 Résumé

Framework	Technologies	MS SQL	n-Tier	Gén Modèles	ACL	API	Licence
ASP.NET	ASP, C#	Oui	Oui	Oui	Oui	Oui	Propriétaire
CakePHP	PHP	Oui	Oui	Oui	Oui	Oui	MIT
Django	Python	Oui	Oui	Oui	Oui	Oui ¹	BSD ¹
Symfony	PHP	Oui	Oui	Oui ²	Oui	Oui	MIT
Zend Framework	PHP	Oui	Oui	Oui	Oui	Oui	new BSD

TABLE 1. *Récapitulatif des frameworks.*

Comme on peut le constater avec la table 1, sur les cinq Frameworks que j'ai choisis, tous son capable de gérer les fonctionnalités minimum nécessaires à ce projet. Je vais donc pouvoir me concentrer sur ceux qui m'intéressent le plus personnellement.

1. BSD 3 Clauses (KAPLAN-MOSS, 2016)

2. A travers l'utilisation de Doctrine

9.2 ASP.NET

Connexion à SQL Server. Tout a fait compatible et même intégré dans l'outil de développement Visual Studio étant donné que ces produits sont développés par Microsoft.

Génération des modèles. Visual Studio permet de créer des modèles d'entités, à la main ou à partir d'une base de données préexistante (MICROSOFT, 2013).

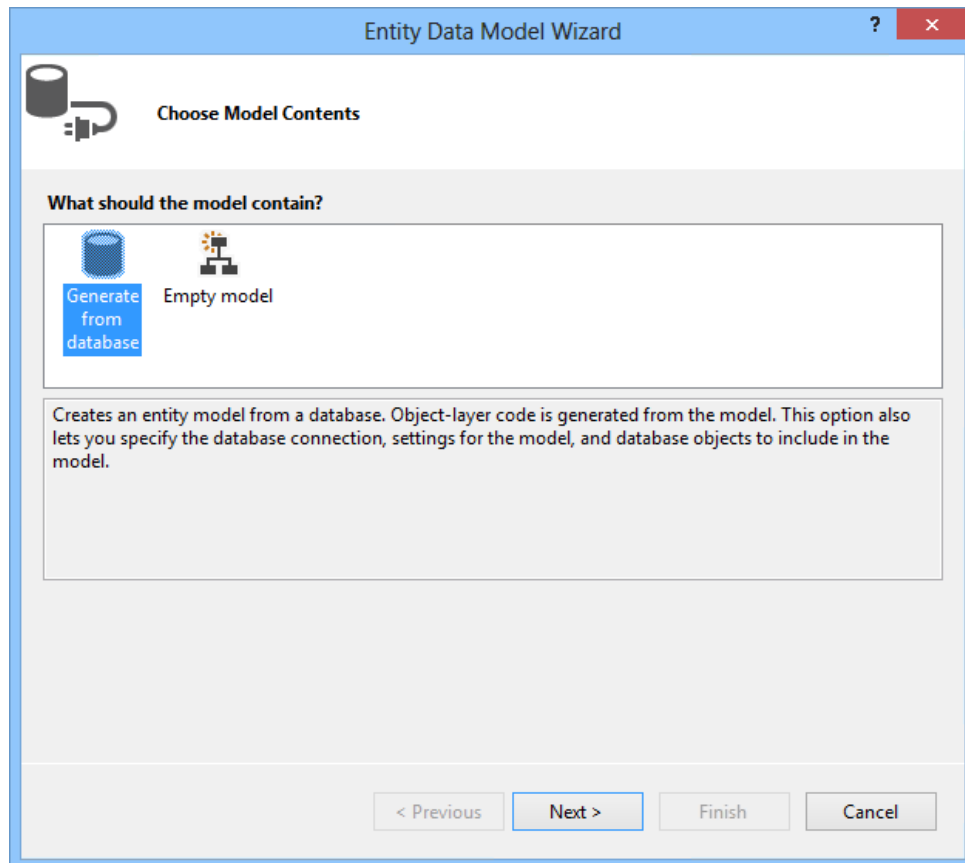


FIGURE 1. Type de création de modèle dans Visual Studio (Samuel Dorsaz)

Coût de la licence. Selon les prix officiels de Microsoft, coûts de l'environnement de développement (MICROSOFT, p.d.).

Licence	Prix
Visual Studio Professional 2015	499\$
Visual Studio Professional avec MSDN	1199\$, puis 799\$ de renouvellement
Version Cloud Annuelle	533\$/an
Version Cloud Mensuelle	45\$/mois

TABLE 2. Prix de Visual Studio Professional.

9.3 Django

Connexion à MS SQL Server. Le module `django-mssql` permet de fournir un connecteur vers des bases de données Microsoft SQL Server. Celui-ci sera activé et configuré dans le fichier `settings.py` (« Django MSSQL Documentation », 2015).

Génération des modèles. Celle-ci peut être effectuée à partir d'une seule ligne de commande comme le montre la première ligne de la figure 2 (« Django Documentation », 2016). La classe métier générée se trouve ici en ligne 6 à 11 de la figure 2.

```
1 $ python manage.py inspectdb
2 from __future__ import unicode_literals
3
4 from django.db import models
5
6 class ArticleArticle(models.Model):
7     id = models.IntegerField(primary_key=True) # AutoField
8     ?
9     name = models.CharField(max_length=30)
10    price = models.FloatField()
11    supplier = models.ForeignKey('SupplierSupplier', models
12    .DO_NOTHING, blank=True, null=True)
13    stared = models.BooleanField()
14
15    class Meta:
16        managed = False
17        db_table = 'article_article'
```

FIGURE 2. Générer les modèles à partir d'une base de données (Samuel Dorsaz)

Avantages.

- Indentation correcte obligatoire (c'est une spécificité de Python).
- Une très bonne documentation et très complète.
- Django active les sécurités par défaut, le développeur peut les désactiver par la suite (DJANGO COMMUNITY, p.d.)

Désavantages.

- Moins de compétences Python sur le marché (la tendance change, voir annexe I).

9.4 Symfony

Connexion à MS SQL Server. Doctrine est l'Object-Relational Mapping (ORM) choisi par Symfony. Toute problématique avec la gestion de la BDD passe à travers cet outil. Ce dernier prend en charge les formats courant dont Microsoft SQL Server. Il faut avoir préalablement installé le module `apache pdo_sqlsrv`. Une configuration de ce type permettra ensuite de communiquer avec le serveur (« 3. Configuration - Doctrine DBAL 2 documentation », p.d.) :

```
1 <?php
2 $config = new \Doctrine\DBAL\Configuration();
3 //...
4 $connectionParams = array(
5     'url' => 'mssql://user:secret@localhost/mydb',
6 );
7 $conn = \Doctrine\DBAL\DriverManager::getConnection(
8     $connectionParams, $config);
```

FIGURE 3. Connexion à SQL Server à travers Doctrine

Génération des modèles. Il est aussi tout à fait possible de générer les entités à partir de celui-ci. A commencer par les métadonnées décrivant ces dernières comme le montre la figure 4

```
1 $ php bin/console doctrine:mapping:import --force
   AcmeBlogBundle xml
```

FIGURE 4. Générer les métadonnées de la BDD

Puis en créer la classe à proprement parler avec les commandes shell en figure 5

```
1 $ php bin/console doctrine:mapping:convert annotation ./src
2 $ php bin/console doctrine:generate:entities AcmeBlogBundle
```

FIGURE 5. Générer les entités avec Doctrine

La figure 6 est un exemple d'une entité générée à partir de la table d'un blog. (SENSIOLABS, 2016)

```
1 // src/Acme/BlogBundle/Entity/BlogComment.php
2 namespace Acme\BlogBundle\Entity;
3 use Doctrine\ORM\Mapping as ORM;
4 /**
5  * Acme\BlogBundle\Entity\BlogComment
6  *
7  * @ORM\Table(name="blog_comment")
8  * @ORM\Entity
9  */
10 class BlogComment
11 {
12     /**
13      * @var integer $id
14      *
15      * @ORM\Column(name="id", type="bigint")
16      * @ORM\Id
17      * @ORM\GeneratedValue(strategy="IDENTITY")
18      */
19     private $id;
20
21     /**
22      * @var BlogPost
23      *
24      * @ORM\ManyToOne(targetEntity="BlogPost")
25      * @ORM\JoinColumn(name="post_id", referencedColumnName="id")
26      */
27     private $post;
28 }
```

FIGURE 6. Exemple de contenu d'une entité générée avec Doctrine

Avantages.

- Beaucoup de développeurs PHP sur le marché.
- Une documentation efficace à l'image de Django.
- Une communauté francophone importante étant donnée l'origine française de son concepteur (NEBRA, 2014).

Désavantages.

- Le PHP est un « langage » qui manque cruellement de structure (EEVÉE, 2012)

9.5 Proposition finale

Les bons Frameworks du marché proposent tous les fonctionnalités de base nécessaire au projet moyennant des coûts ou des configurations plus ou moins importantes.

Pour faire une proposition réaliste vis-à-vis du client, je me suis donc fié à leur capacité à reprendre le projet et au budget à leur disposition. Pour ce dernier, le zéro dépenses est de rigueur, il faudra donc abandonner l'utilisation de Visual Studio.

Enfin, si Django est un Framework qui me plait beaucoup, je garde en tête que trouver des compétences en PHP sur le marché est toujours plus évident que de trouver des développeurs Python. Qui plus est, certains de leurs employés connaissent déjà des bases de PHP.

Même si mon premier chez n'était pas Symfony, j'ai tout de même accepté d'apprendre un nouveau Framework pour le confort du client et aussi par curiosité personnelle.

10 Analyse

10.1 L'intranet

Le serveur Web a été installé grâce à la suite EasyPHP. La racine publique fournie est une instance de Joomla mais l'intranet réel se trouve dans un sous dossier nommé `MsSQL/test/`. Le système communique avec un service de monitoring réseau nommé WUG.

Après réception des sources, j'ai tenté de monter un serveur local afin d'en étudier le fonctionnement. Quelques constats sont très vite arrivés :

- L'installation du serveur SQL dans une machine virtuelle demande quelques configurations spécifiques pour communiquer avec l'hôte (NAVED, 2013) (WEBECS HOSTING, 2012).
- Le connecteur SQL Server fourni par Microsoft n'est pas compatible GNU is Not Unix (GNU)/Linux. Son alternative n'est malheureusement pas identique à l'utilisation et a rendu l'analyse plus difficile.
- L'intranet est dépendant d'une connexion à la BDD de WUG pour afficher des pages importantes mais cette dernière est trop volumineuse pour m'être transmise.

10.2 La base de données

La base de données sous Microsoft SQL Server contient 86 tables dont la structure n'est pas nécessairement aussi logique que ce qu'on attendrait d'une modélisation efficace. Toutes les tables ne sont par contre pas nécessaires au projet, il convient donc d'isoler celles traitant du matériel, des sites, des clients et des employés.

10.3 Ancienne interface

Une partie des éléments comme l'affichage en arbre sur le côté et les boutons d'outils seront gardés car ils permettront aux techniciens de se retrouver dans le nouvel environnement. Par contre des pages comme « Manage Devices for » ou des éléments tel que « Database Transaction Status » disparaîtront au profit d'une ergonomie repensée.

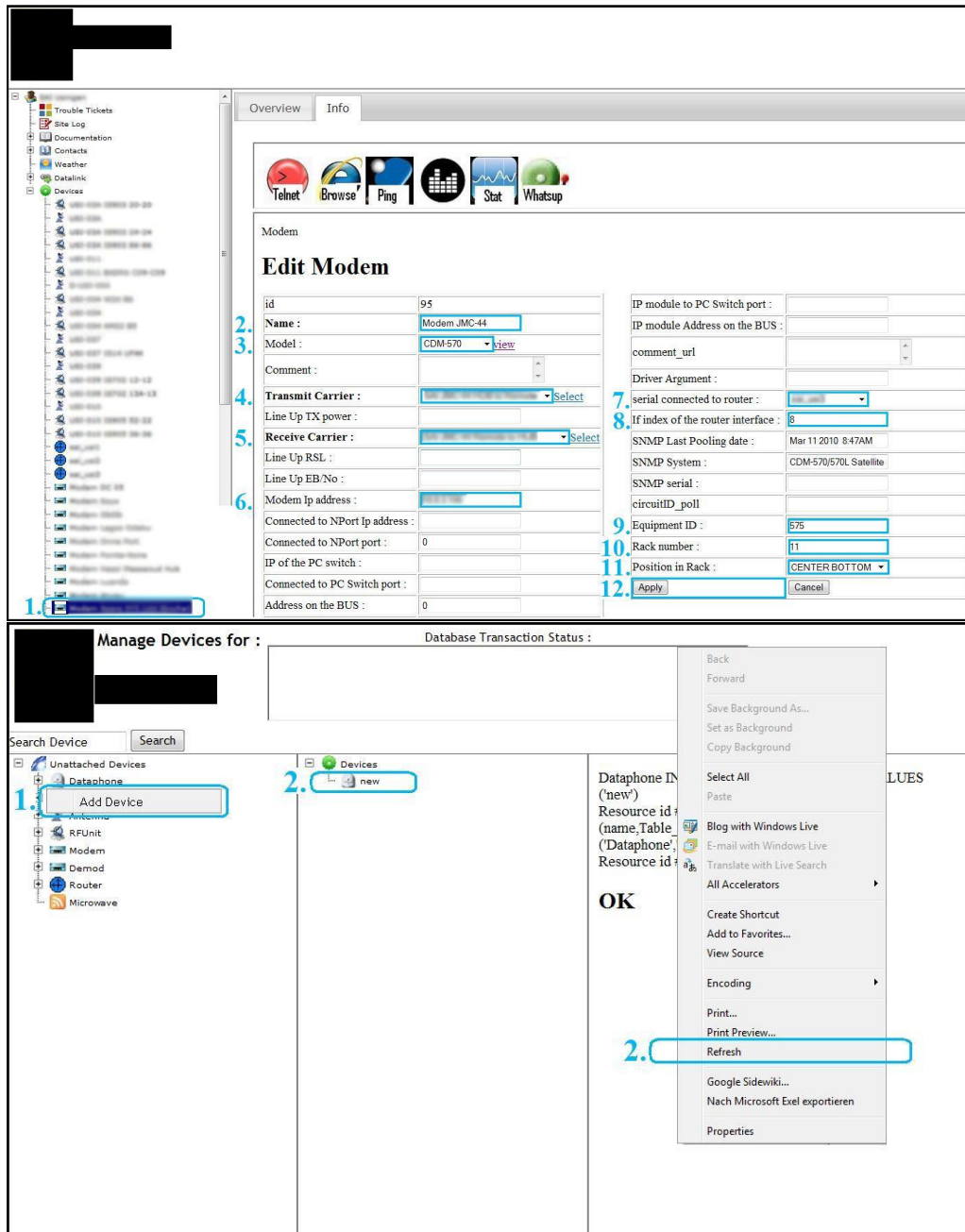


FIGURE 7. Deux visuels de l'ancienne interface tirée des procédures d'utilisation (Newsat)

11 Architecture

11.1 Environnement

La machine de production est un Personal Computer (PC) dédié au projet. Celui-ci intégrera une distribution GNU/Linux, accueillera le projet et le Source Code Management (SCM) qui assurera la pérennité du code et son déploiement. Si nécessaire ce dernier contiendra aussi une copie de la base de données de production afin de permettre à l'entreprise d'effectuer ses tests sans risques.

11.2 Gestion du code source

L'utilisation de git a l'avantage, par rapport aux autres SCM d'être décentralisé. Cela signifie que je peux préparer les « commits » de mes développements sans être connecté au serveur de réception. Ce paramètre est particulièrement important dans le cadre de ce projet, car il va me permettre de transmettre mon travail sur un réseau privé et de ne pas avoir recours à un Virtual Private Network (VPN).

Du moment que je pars sur des outils libres, on peut travailler avec Gitlab Community Edition (CE) qui à l'avantage d'être auto-hébergeable, ainsi les outils de gestion du projet et le projet pourront tourner sur une même machine au sein même de l'entreprise.

11.3 Intégration Continue

Gitlab propose son propre outil d'intégration continue nommé Gitlab Countinous Integration (CI). Celui-ci permet, depuis la version 7.12 d'utiliser un fichier de commandes `.gitlab-ci.yml` directement dans le dossier du projet (van der VOORT, 2015).

12 Modélisation

12.1 Use Case

Le diagramme de « Use Case », en figure 8, est utile à la représentation des interactions des rôles dans le système d'information. Il me permettra de m'assurer de la bonne écriture du « Product Backlog »

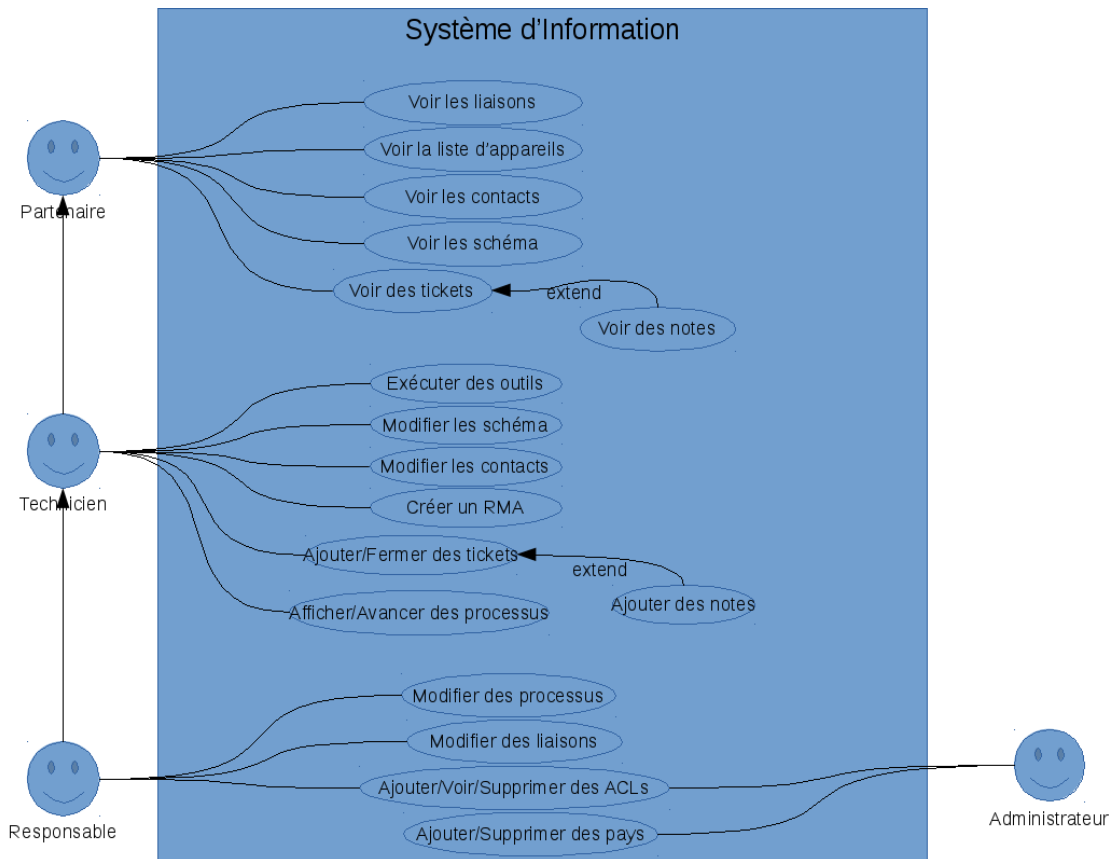


FIGURE 8. Use Case (Samuel Dorsaz)

12.2 Diagramme de Classes

Le diagramme de « Classes » me permettra d'avoir une visibilité sur les objets métiers et leurs interactions durant le projet. Seuls les objets importants et épurés de leurs paramètres et méthodes sont affichés car ce diagramme me sera surtout utile après les générations automatisées pour comprendre comment récupérer certaines données.

J'ai coloré quelques zones pour définir des contextes généraux :

- En verre : les appareils de l'entreprise.
- En bleu et azure : les éléments en rapport avec les satellites.
- En jaune : les éléments administratifs.
- En violet : la gestion des zones de déploiement.
- En rouge : les rapports de dépannage.
- En blanc : une table ajoutée pour la gestion des droits d'accès.

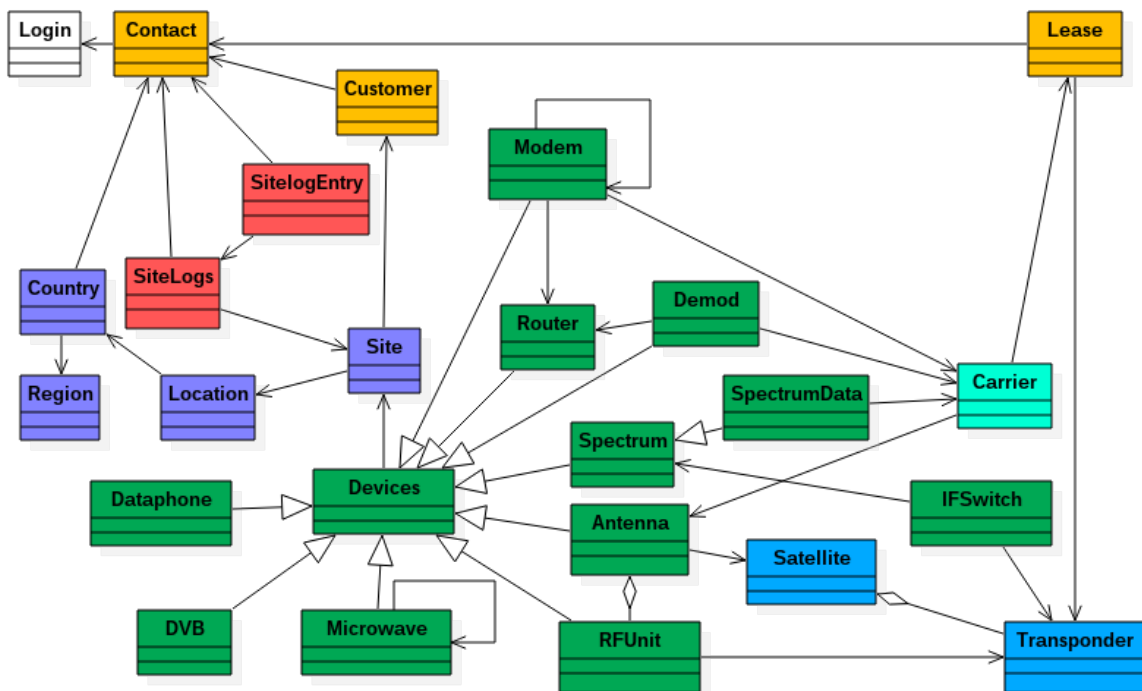


FIGURE 9. Diagramme de Classes (Samuel Dorsaz)

12.3 Diagramme réseau

Le diagramme réseau représente les interactions entre les services sur la machine de production et son intégration dans le réseau. On peut y voir un réseau interne 10.0.2.x qui fournira une base de données de test et un autre constitué de noms intranet2.local accessible par les employés de Newsat.

Interaction des services

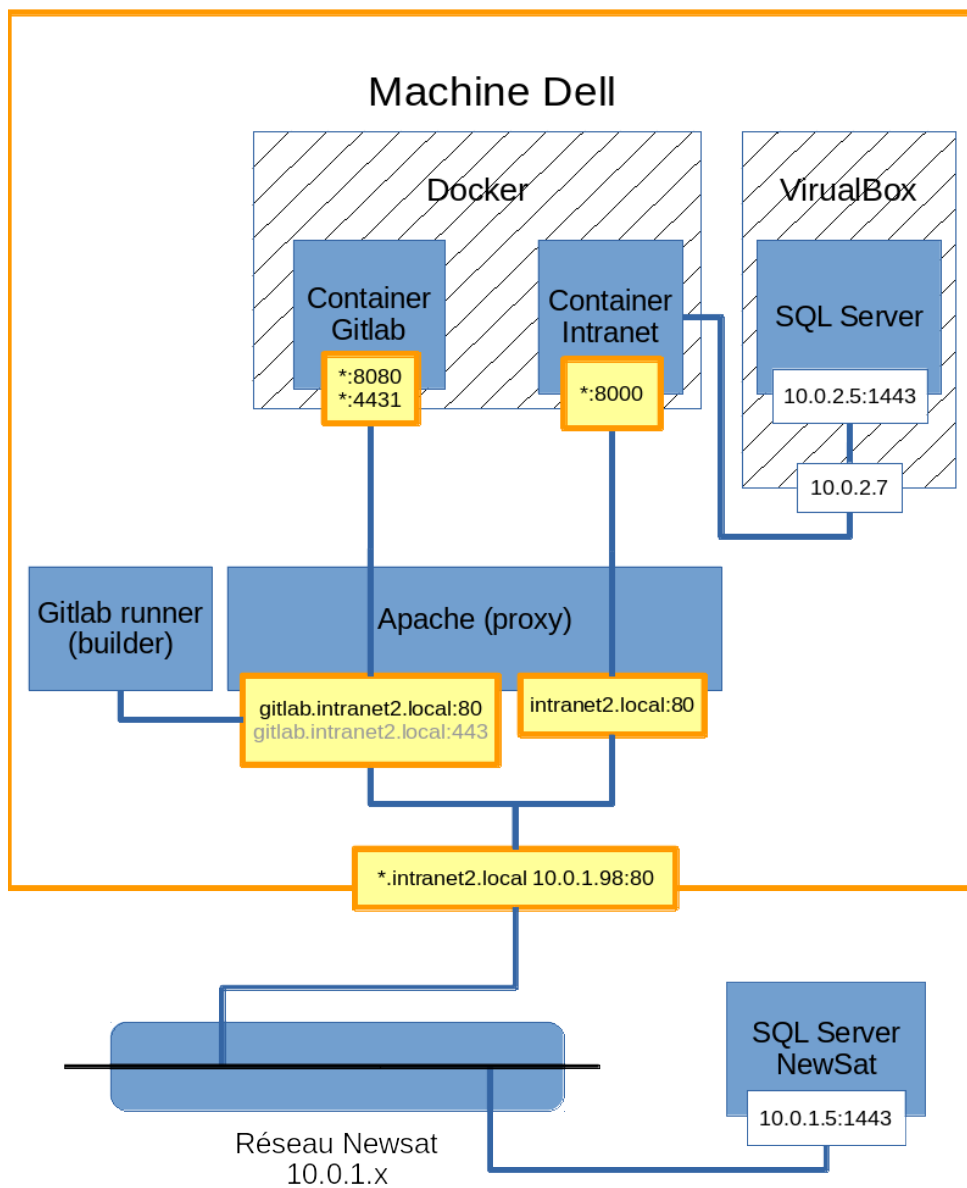


FIGURE 10. Diagramme réseau (Dorsaz Samuel)

13 Environnement de développement

13.1 Conventions de codage

Durant ce projet, j'ai tenté de suivre au mieux les deux premières PHP Standards Recommendations (PSR).

13.2 Installation de SQL Server 2012

L'installation a pris plus de temps que je ne l'aurais pensé au premier abord. J'avais une machine légère sous Windows Seven fraîchement installée. J'ai d'abord voulu installer Visual Studio 2015 mais ce dernier était incompatible avec Windows Seven. J'ai de ce fait, opté pour une version 2013 avant de me rendre compte que seule la version 2012 était compatible (la version la plus vieille téléchargeable sur DreamSpark durant le projet).

J'ai ensuite tenté de l'installer avec SQL Seveur en français vu que l'Operating System (OS) avait été préalablement installé sous cette langue. Mais cela pose apparemment problème au niveau de la localisation.

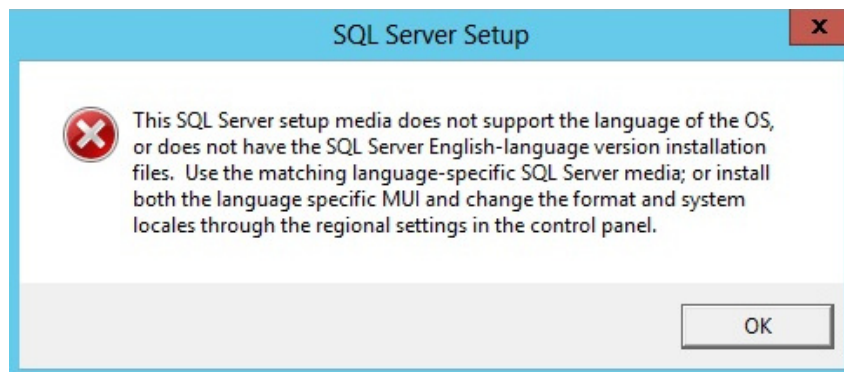


FIGURE 11. Erreur de localisation de SQL Server (Samuel Dorsaz)

J'ai décidé d'installer SQL Serveur en anglais, mais pour pouvoir le faire, il a fallu télécharger le Service Pack 1. Enfin pour pouvoir importer les données, l'installation de l'outil SQL Server Management Studio (SSMS) était nécessaire.

Windows Seven Pro	7.9 Go
Avec Visual Studio 2012	26 Go
Avec le Service Pack (SP)1	28.3 Go
Avec SQL Server 2012 et SSMS	29.7 Go

14 Environnement de déploiement

14.1 Docker

J'ai profité de mes connaissances sur Docker pour fournir un environnement de déploiement automatisé à ma plate-forme Symfony. J'ai ainsi suivi diverses instructions sur internet et préparé un conteneur qui peut se construire entièrement sur la base d'une image GNU/Linux Debian ¹

La mise en service de la machine pourra se faire avec la commande en figure 13. Sa première exécution pourra prendre du temps pour l'installation de base.

```
1 docker-compose up -d
```

FIGURE 13. Exécute les instructions du fichier docker-compose.yml en mode service (-d)

En cas de besoin, la maintenance pourra se faire en appelant un terminal interactif avec la commande

```
1 docker exec -t -i newsat_web_1 bash
```

FIGURE 14. Exécuter un terminal (-t) interactif (-i) de type bash

Un autre avantage à l'implémentation de Docker dans le dossier du projet, est d'avoir toutes les instructions nécessaires pour remonter un serveur ainsi que ses fichiers de configuration sous la main.

Newsat aimerait quelque chose de simple à administrer et émet des doutes à l'utilisation de Docker. Toutefois, rien ne les empêche de travailler sans le dossier `docker` et sans le fichier `docker-compose.yml` et d'effectuer une installation traditionnelle. Deux manuels d'installation leur seront fournis à ce sujet.

1. <http://blog.webnet.fr/docker-pas-a-pas-sur-une-application-symfony2-part3/>

14.2 Doctrine

La préparation de Doctrine pour se connecter à la base de données SQL Server a été plus longue et plus compliquée que prévu. La connexion avec le driver sqlsrv devait être évidente mais n'est pas compatible avec un environnement GNU/Linux.

Afin de pouvoir faire ce travail, il aura fallu indiquer à Database Abstraction Layer (DBAL) de passer par la librairie FreeTDS. Cette dernière est une implémentation open-source du protocole Tabular Data Stream (TDS) pour pouvoir communiquer avec la base de données. Le module LeaseWeb/LswDoctrinePdoDbliib va permettre d'installer rapidement la librairie dans Symfony (TUDORICA, 2013).

Pour information, l'installation de modules dans Symfony se fait au travers de l'outil Composer. Celui-ci fonctionne comme un gestionnaire de paquets mais avec des modules PHP. Comme le module à installer est connu, je peux tout de suite l'installer comme une dépendance (voir figure 15).

```
1 $ composer require "leaseweb/doctrine-pdo-dbliib"
```

FIGURE 15. Ajouter la dépendance au module "leaseweb/doctrine-pdo-dlib"

L'installation de FreeTDS est décrite dans le manuel d'installation. Sa configuration doit se faire avant celle de Symfony car c'est DBAL qui va appeler FreeTDS.

14.3 Symfony

J'ai décidé d'installer la version 2.8 de Symfony. La version 3.0 existe déjà mais est encore trop jeune pour être utilisée en toute sécurité. De plus la version 2.8 est la dernière Long Term Support (LTS) qui sera supportée jusqu'en novembre 2018 (SYMFONY DOCUMENTATION TEAM, 2015).

Version	Release date	Support	PHP version	End of maintenance	Notes
2.8	November 2015	Three years	$\geq 5.3.9$	November 2018	LTS release.
3.0	November 2015	Eight months	$\geq 5.5.9$	July 2016	
3.1	May 2016	Eight months	$\geq 5.5.9$	January 2017	
3.2	November 2016	Eight months	$\geq 5.?.?$	July 2017	
3.3	May 2017	Eight months	$\geq 5.?.?$	January 2018	
3.4	November 2017	Three years	$\geq 5.?.?$	November 2020	LTS release.

TABLE 3. *Roadmap des versions de Symfony (Wikipedia)*

14.4 Bootstrap

Bootstrap est une librairie pratique pour monter un visuel correct dans le cadre d'une interface « backend » ou « intranet ». Si elle me permettra de gagner du temps, il faut garder en tête que le World Wide Web Consortium (W3C) ne promeut pas cette pratique pour des raisons de propreté sémantique (LESIEUR, 2014).

J'ai d'abord tenté de l'intégrer avec Composer mais ce dernier n'est pas des plus approprié pour du code Javascript. Pour ce faire, il faudra utiliser Bower qui est, lui, un gestionnaire de librairies dédié au « front-end » (SHAFIK, 2014). Je voulais éviter d'utiliser une énième application uniquement pour gérer Bootstrap. J'ai donc décidé de copier ces fichiers manuellement dans le dossier statique (`src/web`) de Symfony.

15 Importation des Entités

15.1 Notes

Il peut arriver que certaines erreurs soient relevées par le serveur `SQL` avant de l'être par Doctrine ou Symfony. Le problème dans ce cas, est de ne pas savoir ce qu'il se passe en amont car je ne reçois que des messages de type `SQLSTATE[HY000]: General error: 319 General SQL Server error: Check messages from the SQL Server [319] (severity 15) [(null)]` .

Dans cet exemple, une information peut tout de même m'aider : le serveur `SQL` contient la réponse qui est stockée sous l'id numéro 319. Une simple requête `SQL` sur ce dernier permettra de connaître l'origine de l'erreur.

```
1 SELECT * FROM sys.messages WHERE message_id=319
```

FIGURE 16. Demander la signification d'un code d'erreur à `SQL Server`

15.2 Génération

La commande suivante génère la structure Extensible Markup Language (`XML`) à partir de la base de données, la deuxième ligne converti ces fichiers `XML` en entités annotées et la troisième génère les getters et les setters. Toutefois, cette dernière reste limitée si la structure de la base de données n'est pas suffisamment correcte. Dans notre cas, certaines tables n'avaient pas de clefs, il faudra donc les importer manuellement plus tard si nécessaire.

```
1 $ php app/console doctrine:mapping:import --force AppBundle
  xml
2 $ php app/console doctrine:mapping:convert annotation ./src
3 $ php bin/console doctrine:generate:entities AppBundle
```

FIGURE 17. Générer les entités par rapport à la base de données existante

15.3 Liaisons non définies

Si les entités ont été générées correctement, les liaisons entre celles-ci sont parfois incomplètes. Le problème dans ce cas est que Doctrine utilise ces relations afin de gérer ses objets.

Il est possible de générer ces relations du côté serveur avec une commande de type :

```
1      ADD CONSTRAINT FK_Site_Location FOREIGN KEY (
        Location_id) REFERENCES [Newsat].[dbo].[Location] (
        id);
```

FIGURE 18. Ajouter une contrainte de clef secondaire cause une erreur

La création des relations posera des problèmes car certains identifiants sont manquants dans quelques relations. Il est possible de le voir en utilisant la commande en figure 19 qui retourne deux résultats sur la base de données de Newsat ².

```
1  SELECT Location_id
2     FROM Site s
3     WHERE
4         Location_id IS NULL OR
5         NOT EXISTS (SELECT 1 FROM Location l WHERE s.
                     Location_id=l.id)
```

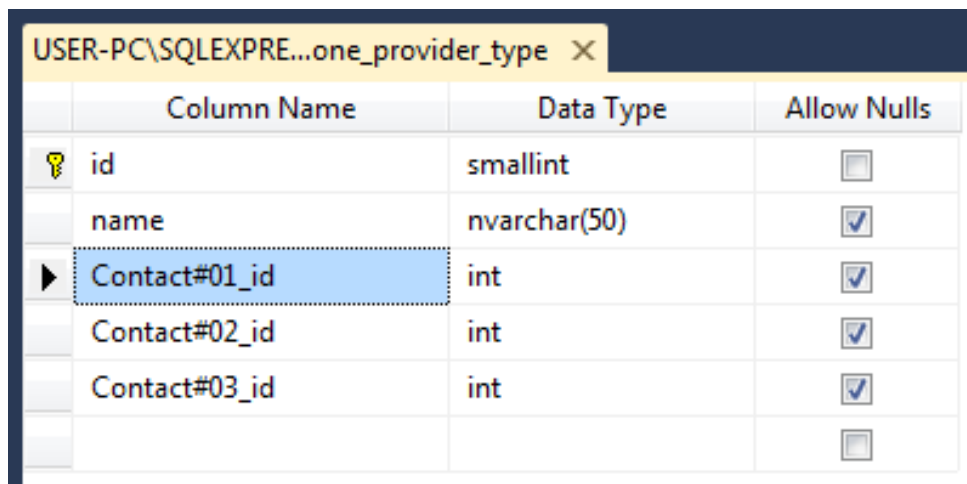
FIGURE 19. Retrouver des identifiants manquants dans une liaison.

Toutefois, comme le stipule le projet, je ne m'attarderai pas à essayer de toucher la structure de la base de données. La solution aura été de retoucher les entités afin de forcer manuellement la liaison et de répercuter les changements dans les fichiers XML s'y rapportant.

2. <http://stackoverflow.com/questions/11324433/foreign-key-fails-to-create>

15.4 Nommage des colonnes

Microsoft SQL Server est particulièrement permissif au niveau du nommage. Il est par exemple admis d'utiliser un « # » ou un « [» dans les colonnes de ses tables comme j'ai pu le constater en figure 20. Comme Doctrine génère les paramètres des entités à partir des noms de ces colonnes, il a essayé de faire passer `Contact#01_id` comme nom de variable, ce qui n'est bien entendu pas autorisé en PHP (« PHP : Essentiel - Manual », p.d.).



The screenshot shows a table in Microsoft SQL Server Enterprise. The table has four columns: 'id', 'name', 'Contact#01_id', and 'Contact#02_id', and 'Contact#03_id'. The 'id' column is a primary key (smallint). The 'name' column is nvarchar(50). The 'Contact#01_id', 'Contact#02_id', and 'Contact#03_id' columns are integers (int). The 'Allow Nulls' column shows checkboxes for each column, with 'name', 'Contact#01_id', 'Contact#02_id', and 'Contact#03_id' having checked boxes, and 'id' having an unchecked box.

	Column Name	Data Type	Allow Nulls
🔑	id	smallint	<input type="checkbox"/>
	name	nvarchar(50)	<input checked="" type="checkbox"/>
▶	Contact#01_id	int	<input checked="" type="checkbox"/>
	Contact#02_id	int	<input checked="" type="checkbox"/>
	Contact#03_id	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

FIGURE 20. Utilisation du dièse avec Microsoft SQL Server (Samuel Dorsaz)

Dans certains cas, le Système de Gestion de Base de Données (SGBD) fournira ou attendra un nom échappé de crochets. Ce cas est arrivé une fois et il a fallu retoucher le nom de champ lié dans les annotations de l'entité.

15.5 Entité Demod

Cette entité a une particularité intéressante car il est possible de lui connecter 12 *Carrier* différentes mais dans 12 champs différents. Le problème est donc de reconstruire les liaisons et renommer tous les getters et setters.

Cette modification est fastidieuse, c'est pourquoi je préconise de supprimer les getters et setters générés, puis de les reconstruire sur la base des paramètres corrigés. On économisera environ 35 minutes, mais surtout des erreurs humaines sur cette opération.

16 Formulaires

16.1 Génération automatisée

Doctrine permet de générer des formulaires à partir des entités créées. Dans ce travail, j'ai pu générer les huit formulaires relatifs à la création des *Devices* en quelques minutes.

```
1 php app/console generate:doctrine:form AppBundle:Spectrum
```

FIGURE 21. Exemple de génération automatisée du formulaire de la classe « Spectrum »

Selon l'exemple en figure 21, un fichier `AppBundle/Form/SpectrumType.php` sera créé dans `src/AppBundle` du projet Symfony.

16.2 Chargement dynamique

Dans le cadre des *Devices*, j'avais une dizaine de formulaires différents à charger en fonction du type de *Device* correspondant. Afin de ne pas avoir à écrire un grand nombre de contrôleurs ne servant qu'à rapatrier où traiter un formulaire, j'ai essayé de les charger dynamiquement à partir de la méthode `editAction` du contrôleur *DevicesController*.

Dans l'exemple de la figure 22, j'ai essayé de charger dynamiquement l'objet *Microwave* et son formulaire *MicrowaveType*. Ceci n'est malheureusement pas possible car Symfony optimise le chargement des fichiers en excluant ceux qui ne sont pas référencés dans le code même s'ils sont explicitement inclus avec `use`. Hors, comme ici, l'appel la classe *Microwave* est faite par une variable en lignes 8 et 11 de la figure 22, Symfony ne peut pas savoir quelle classe sera réellement utilisée (les classes sont déclarées en ligne 2 et 4).

L'astuce simple est de donner le chemin complet de cette classe à Symfony au moment de son chargement, ainsi je n'ai plus besoin d'utiliser les méthodes `use` pour ces classes au début du fichier comme on peut le voir en figure 23

```

1  [...]
2  use AppBundle\Entity\Microwave;
3  [...]
4  use AppBundle\Form\MicrowaveType;
5  [...]
6  // Dynamically load the correct device object
7  $className = ucfirst($type);
8  $device = new $className();
9  [...]
10 // Dynamically load the form corresponding on the device
11 $form = $this->createForm(ucfirst($type) . 'Type', $device)
    ;
12 $form->handleRequest($request);
13 [...]

```

FIGURE 22. Tentative de chargement dynamique des classes

```

1  [...]
2  // Dynamically load the correct device object
3  $className = 'AppBundle\Entity\\' . ucfirst($type);
4  $device = new $className();
5  [...]
6  // Dynamically load the form corresponding on the device
7  $form = $this->createForm('AppBundle\Form\\' . ucfirst(
    $type) . 'Type', $device);
8  $form->handleRequest($request);
9  [...]

```

FIGURE 23. Résolution du chargement dynamique des classes

16.3 Contrôle des champs

Chez les développeurs, il est courant de dire « Never trust user input ! » en référence au fait que chaque champ de formulaire doit être traité comme si on tentait une injection. L'utilisation d'un Framework comme Symfony permet d'automatiser le processus de contrôle.

Toutefois, je me suis penché sur la question en termes d'allègement des tests unitaires. En effet, on peut imaginer faire des tests pour s'assurer qu'une adresse IP est entrée correctement avant son traitement mais il est aussi possible de s'assurer que tout formulaire enregistre les bons types de données en intégrant un contrôle automatique de ces dernières.

Après quelques recherches, je suis rapidement tombé sur les informations intéressantes qui montrent que le langage PHP fournit des solutions automatisées simples pour contrôler une chaîne de caractères avec des types spécifiques à travers la méthode `filter_var(...)`. Cette recherche m'a fait prendre conscience qu'une bonne partie de ces tests étaient certainement déjà écrits au niveau de Symfony et c'est comme ça que j'ai trouvé la validation d'adresse IP en deux lignes de code.

```
1 // src/AppBundle/Entity/Author.php
2 namespace AppBundle\Entity;
3
4 use Symfony\Component\Validator\Constraints as Assert;
5
6 class Author
7 {
8     /**
9      * @Assert\Ip
10     */
11     protected $ipAddress;
12 }
```

FIGURE 24. Exemple simple de l'utilisation d'un « Validator » pour contrôler l'adresse IP

Si un contrôle est tellement spécifique qu'il n'existe pas dans Symfony, il est aussi possible de créer des « Validators » personnalisés en ajoutant deux fichiers dans le « Bundle » comme en exemple dans le dossier `AppBundle/Validator/Constraints/`.

17 CRUD

C'est en générant les formulaires que l'on remarque qu'il existe plusieurs commandes disponibles et dont l'une a le nom `doctrine:generate:crud`. Celle-ci est très intéressante car elle permet, en plus de générer le formulaire, d'en générer tous les affichages de base, le contrôleur et préparer l'environnement de tests unitaires.

J'ai donc profité de devoir travailler sur l'entité `Country` pour en voir l'efficacité. Le résultat brut est visible dans le commit numéro

5c87800a030ad36a1a0fee389c43ba18a9d6a714

```
1 # php app/console doctrine:generate:crud AppBundle:Country
2
3 Do you want to generate the "write" actions [no]? yes
4
5 Configuration format (yml, xml, php, or annotation) [
  annotation]: annotation
6
7 Routes prefix [/country]:
8
9   Summary before generation
10
11 You are going to generate a CRUD controller for "AppBundle:
  Country"
12 using the "annotation" format.
13
14 Do you confirm generation [yes]?
15
16   CRUD generation
17
18 Generating the CRUD code: OK
19 Generating the Form code: OK
20 Updating the routing: OK
21
22 Everything is OK! Now get to work :).
```

FIGURE 25. Résumé du processus de génération CRUD à partir d'une entité

Cette génération m'a permis de voir que les routes sont souvent définies dans les annotations directement ce qui évite d'avoir un fichier `routing.yml` énorme.

De plus les types Hypertext Transfer Protocol (HTTP) peuvent être utilisés afin de simplifier l'écriture des routes et faire abstraction des instructions `/new/`, `/edit/`, `/delete/` par exemple.

18 Gestion des utilisateurs

18.1 FOS User Bundle

Pour gérer des utilisateurs sous Symfony, il est recommandé d'installer et d'utiliser le module FOS User Bundle. Ce module nécessite de créer une entité qui héritera d'un objet préconstruit. Le problème, à ce stade du développement, est que Doctrine aimerait mettre à jour certaines tables alors qu'on doit éviter au maximum de toucher la structure de la base de données (92 requêtes sont demandées dans notre cas).

Si l'on veut éviter d'avoir à le faire, il est tout à fait possible de demander à Doctrine de créer un dump des requêtes qu'il prévoit (comme affiché en figure 26) et de sélectionner celles qui sont intéressantes.

Dans cette figure, seul les lignes 11 à 14 sont intéressantes. Toutefois, elles ont aussi posé problème en raison de la création de plusieurs index uniques. En effet, le driver GNU/Linux de SQL Server, ne permet pas de le faire si la table a été créée à l'origine sur le serveur de Microsoft (« DBAL-1068 : Microsoft SQL Server issues with ANSI_NULLS=OFF · Issue #1002 · doctrine/dbal », 2014).

```

1 # php app/console doctrine:schema:update --dump-sql
2 The Schema-Tool would execute "92" queries to update the
  database.
3 Please run the operation by passing one - or both - of the
  following options:
4     doctrine:schema:update --force to execute the command
5     doctrine:schema:update --dump-sql to dump the SQL
      statements to the screen
6 root@8d7db8d222ba:/var/www/symfony# php app/console
  doctrine:schema:update --dump-sql
7 CREATE TABLE [User] (id INT IDENTITY NOT NULL, username
  NVARCHAR(255) NOT NULL, username_canonical NVARCHAR(255)
  NOT NULL, email NVARCHAR(255) NOT NULL, email_canonical
  NVARCHAR(255) NOT NULL, enabled BIT NOT NULL, salt
  NVARCHAR(255) NOT NULL, password NVARCHAR(255) NOT NULL,
  last_login DATETIME2(6), locked BIT NOT NULL, expired
  BIT NOT NULL, expires_at DATETIME2(6),
  confirmation_token NVARCHAR(255), password_requested_at
  DATETIME2(6), roles VARCHAR(MAX) NOT NULL,
  credentials_expired BIT NOT NULL, credentials_expire_at
  DATETIME2(6), PRIMARY KEY (id));
8 CREATE UNIQUE INDEX UNIQ_2DA1797792FC23A8 ON [User] (
  username_canonical) WHERE username_canonical IS NOT NULL
  ;
9 CREATE UNIQUE INDEX UNIQ_2DA17977A0D96FBF ON [User] (
  email_canonical) WHERE email_canonical IS NOT NULL;
10 EXEC sp_addextendedproperty N'MS_Description', N'(DC2Type:
  array)', N'SHEMA', dbo, N'TABLE', [User], N'COLUMN',
  roles;
11
12 [...]

```

FIGURE 26. Résultat de doctrine afin de mettre à jour la table User

18.2 Module de Sécurité

Quelques recherches ont montré que FOS User Bundle n'est pas une couche de sécurité mais une structure permettant d'utiliser la gestion des utilisateurs fournie par Symfony. En réalité la quantité de travail nécessaire à la création des pages et des contrôleurs de base semble égale à celle qu'il faudrait pour configurer et préparer le FOS User Bundle.

Une entité *Login* a donc été créée afin d'enregistrer les données de connexion chez Newsat et une liaison avec l'entité *Contact* peut être établie par l'administrateur pour fournir un profil à ses employés. Chaque collaborateur pourra librement mettre à jour sa fiche de *Contact* en éditant son profil.

Comme Newsat n'avait besoin que d'une prise en charge simple des rôles, j'ai défini quatre précis :

- Anonymous : Toute personne non connectée n'accèdera qu'à la page de login.
- User : Toute personne connectée peut voir les données techniques de l'intranet.
- Technicians : Ces utilisateurs peuvent ajouter, modifier et supprimer des données techniques.
- Admin : Les admin peuvent accéder à la gestion des utilisateurs et des satellites.

Les droits sont cumulatifs ce qui veut dire qu'un administrateur qui n'est pas technicien pourra voir les données des satellites mais ne pourra pas les modifier.

19 Toolbox

La boîte à outils des techniciens ne devait pas être quelque chose de conséquent à implémenter du moment qu'elle était déjà existante chez le client. Mais cela était sans compter qu'elle n'était fonctionnelle que sur Internet Explorer et uniquement sur d'anciennes versions.

Ces scripts révèlent deux problèmes que sont l'abandon des ActiveX dans le nouveau navigateur Edge de Microsoft (CHARLES MORRIS, 2015) mais surtout l'amélioration du contrôle d'accès HTTP qui a été implémenté sur tous les navigateurs récent (NETWORK, 2015).

19.1 Node.js®

La solution proposée pour venir à bout de ce problème est un petit serveur Node.js® qui sera installé chez tous les clients. Celui-ci fonctionnera comme une API et acceptera les requêtes venant de l'intranet. Une interface Web sera aussi montée afin de fournir une documentation de l'API.

L'utilisation de Node.js® requière une bonne compréhension du Javascript et la prise en compte de plusieurs concepts sur la gestion des évènements et la création d'un service (VLADUTU, 2014).

Afin que l'utilisateur soit accompagné correctement dans la démarche d'installation (si elle s'avérait nécessaire en cours de route), la boîte à outils annonce une erreur de connexion dès la première tentative d'initialisation.

19.2 Whatsup

L'un des éléments de la boîte est un appel au service WUG. Ce service est un outil de monitoring du matériel réseau. Le problème de cet outil est qu'il ne peut pas communiquer à l'extérieur à travers une API (COMMUNITY, 2015).

L'analyse de l'ancien intranet, montre que celui-ci se connecte directement à la base

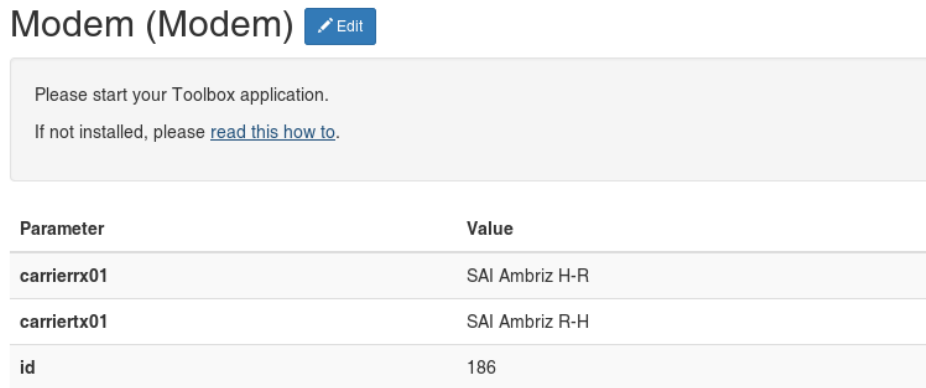


FIGURE 27. Message d'erreur de l'initialisation de la boîte à outils (Samuel Dorsaz)

de données de Whatsup afin d'en chercher les correspondances entre les adresses IP qu'il connaît et les identifiants matériels du service. Puis, une iframe est appelée afin de visionner les statistiques du matériel dont l'identifiant est transmis par variable HTTP GET.

Fort heureusement Doctrine accepte sans trop de problème la connexion à plusieurs bases de données différentes. Il aura donc fallu revoir légèrement l'écriture du fichier `configuration.yml` à ce sujet. En termes d'utilisation, il est possible de choisir la connexion à utiliser par défaut, puis de spécifier à l'ORM laquelle utiliser dans certains cas particuliers. L'exemple suivant illustre une connexion avec une requête SQL native afin d'éviter d'avoir à enregistrer les « Entités » du service WUG.

```

1 $WUGem = $this->getDoctrine()->getManager('whatsup');
2 $stmt = $WUGem->getConnection()->prepare("
3     SELECT *
4     FROM Device
5 ");
6 $stmt->execute();
7 $rows = $stmt->fetchAll();

```

FIGURE 28. Requête SQL native au travers d'une connexion différente

Dans la figure 28, je n'utilise pas de syntaxe Doctrine Query Language (DQL) simplement parce que celle-ci serait dépendante des entités du WUG qui n'ont pas été définies.

19.3 Installation

Il existe une multitude de méthodes pour déployer un serveur Node.js[®] sur un ordinateur. Dans notre cas, le passage par un service Windows semblait la plus sûre et la plus

cohérente à mettre en place. Toutefois le chargement visuel de terminaux et d'applications à travers ce dernier n'était pas possible.

L'alternative à ce problème est de créer un fichier .BAT pouvant être exécuté au démarrage avec un raccourci dans le « menu démarrer » de Windows.

L'utilisation du logiciel « Inno Setup » permet d'automatiser la création d'un logiciel d'installation « tout-en-un » qui permettra un déploiement propre d'une application Javascript (O'NEAL, 2013). Les étapes suivantes seront opérées durant l'installation :

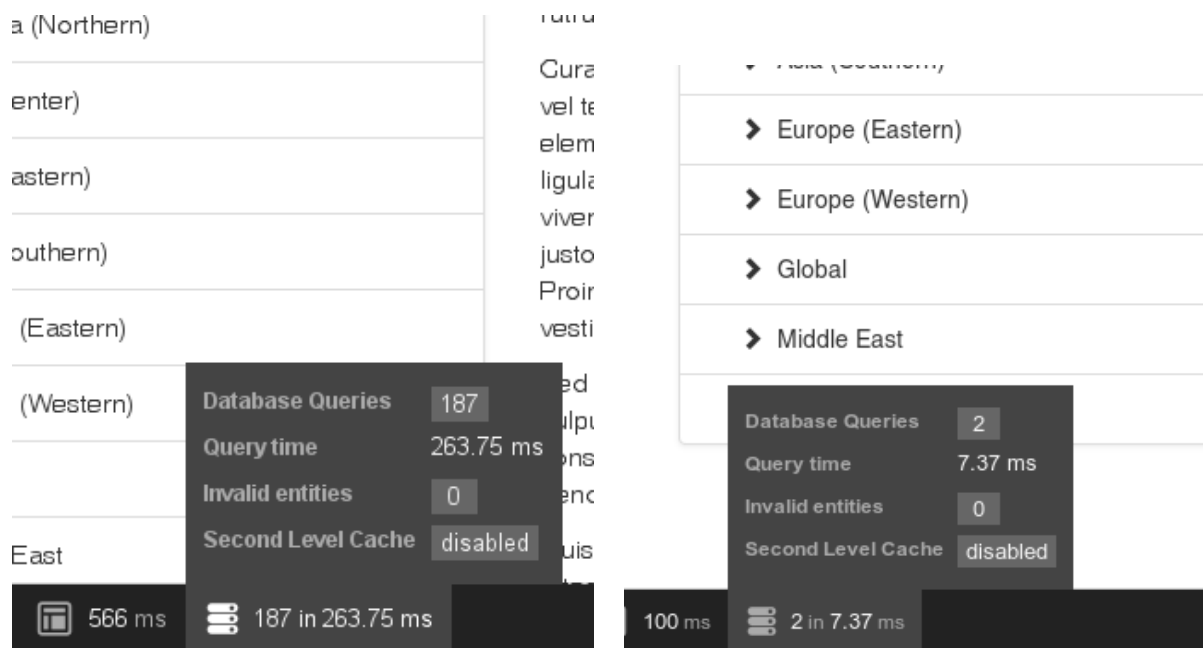
- Copie des fichiers nécessaires au client dans son dossier « Program Files ».
- Création d'une entrée dans le « Menu Démarrer » sous Startup / Toolbox.
- Création des entrées dans le « Menu Démarrer » sous Newsat / Toolbox.
 - Toolbox afin de démarrer le programme si nécessaire.
 - Toolbox (mode debug) permet d'afficher les messages de debug.
 - Uninstall Toolbox afin de pouvoir désinstaller plus facilement l'application.
- Installation automatisée de Node.js[®].
- Mise à jour automatique de npm.
- Installation des dépendances du projet.
- Gestion des règles du Firewall pour Node.js[®].
- Choix de l'utilisateur pour exécuter l'application après l'installation.

20 Optimisations

La barre latérale contenant la localisation des sites, lieux, régions, pays, zones était particulièrement gourmande en terme de requêtes SQL. En effet, j'avais, dans un premier temps, laissé le soin au moteur de templates « Twig » de gérer la récupération des données.

Celui-ci envoyait plus de 150 requêtes afin de lier les informations présentes dans cinq tables. Le chargement des données prenait plus de 200ms et la page se chargeait en plus de 500ms. Cette barre latérale étant présente sur la plupart des pages du site internet, il était primordial de l'optimiser.

J'ai ainsi déporté la récupération des données directement dans le contrôleur prévu à cet effet et demandé à Symfony d'importer l'ensemble du contenu de ces tables en une seule connexion. Le résultat est saisissant, voyez par vous-même en figure 36.



(a) Résultats avant traitement (b) Résultats après traitement
FIGURE 29. Résultats des optimisations de la barre latérale (Samuel Dorsaz)

21 Héritage

Au milieu du schéma de la base de données se trouve la table « Devices ». Cette table est une table de référence pour connaître l'ensemble du matériel lié à un site (hormis la partie satellite). La personne qui a conçu cette base de données à été maligne car elle a su que, pour une collection polymorphique, elle aurait intérêt à travailler de la sorte.

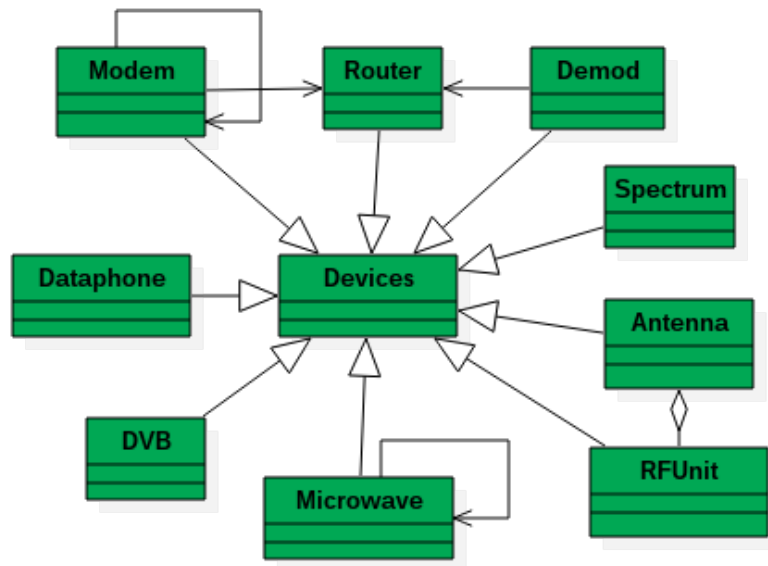


FIGURE 30. Héritage important au centre du système (Samuel Dorsaz)

De là découlent une bonne et une mauvaise nouvelle. La bonne c'est que ce type de structure est prise en charge par Symfony. Dans notre cas, il suffirait d'écrire une annotation de type « Class Table Inheritance » comme dans l'exemple en figure 31

Malheureusement, dans la réalité, cette forme est très peu utilisée et son implémentation n'est pas aussi avancée que ce qu'elle pourrait. En effet la mauvaise nouvelle est que l'héritage sous-entendrait qu'on utilise un index partagé mais dans le cadre de Newsat, une colonne supplémentaire `Table_id` rend cette annotation impossible à utiliser.

Une analyse approfondie du code de Doctrine (figure 32) permet de confirmer le problème³. Le Framework étant open-source, je pourrais proposer une modification mais elle prendrait trop de temps pour être appliquée d'ici la fin du projet.

3. <https://github.com/doctrine/doctrine2/blob/master/lib/Doctrine/ORM/Persisters/Entity/JoinedSubclassPersister.php#L623>

```

1  /**
2  * Devices
3  *
4  * @ORM\Table(name="Devices")
5  * @ORM\Entity
6  * @ORM\InheritanceType("JOINED")
7  * @ORM\DiscriminatorColumn(name="Table_ref", type="string")
8  * @ORM\DiscriminatorMap({"Spectrum"="Spectrum", "Antenna"="
9  *   Antenna"})
10 */
11 class Devices
12 {
13     [...]
14 }

```

FIGURE 31. Exemple d'héritage par table de classes

```

590  /**
591  * @param string $baseTableAlias
592  *
593  * @return string
594  */
595  private function getJoinSql($baseTableAlias)
596  {
597      $joinSql      = '';
598      $identifierColumn = $this->class->getIdentifierColumnNames();
599
600      // INNER JOIN parent tables
601      foreach ($this->class->parentClasses as $parentClassName) {
602          $conditions = array();
603          $parentClass = $this->em->getClassMetadata($parentClassName);
604          $tableAlias = $this->getSQLTableAlias($parentClassName);
605          $joinSql .= ' INNER JOIN ' . $this->quoteStrategy->getTableName($parentClass, $this->platform) . ' ' . $tableAlias
606
607
608          foreach ($identifierColumn as $idColumn) {
609              $conditions[] = $baseTableAlias . ' ' . $idColumn . ' = ' . $tableAlias . ' ' . $idColumn;
610          }
611
612          $joinSql .= implode(' AND ', $conditions);
613      }
614
615      // OUTER JOIN sub tables
616      foreach ($this->class->subClasses as $subClassName) {
617          $conditions = array();
618          $subClass = $this->em->getClassMetadata($subClassName);
619          $tableAlias = $this->getSQLTableAlias($subClassName);
620          $joinSql .= ' LEFT JOIN ' . $this->quoteStrategy->getTableName($subClass, $this->platform) . ' ' . $tableAlias . ' 0
621
622          foreach ($identifierColumn as $idColumn) {
623              $conditions[] = $baseTableAlias . ' ' . $idColumn . ' = ' . $tableAlias . ' ' . $idColumn;
624          }
625
626          $joinSql .= implode(' AND ', $conditions);
627      }
628
629      return $joinSql;
630  }
631 }

```

FIGURE 32. Code de jointure d'une sous-classe dans Doctrine.

22 Openweathermap

Newsat avait prévu de fournir une page météo à ses techniciens afin qu'ils aient une idée des possibilités d'interférences météorologiques avec les signaux envoyés et reçus des satellites.

Cette fonctionnalité n'était pas vraiment prévue dans le cahier des charges et n'était même pas réellement importante. Mais comme j'avais déjà travaillé avec openweathermap.org⁴, j'ai pensé que ce serait une bonne idée de leur fournir ce service.

Bien entendu cette initiative n'était pas dénuée d'intérêt. En effet, si un service comme WUG ne fournit pas d'API pratique à utiliser, il est fort probable que cela change dans un avenir proche.

Avec la mise en place de cette page, Newsat aura un exemple fonctionnel de communication par API qu'elle pourra copier et adapter à l'avenir.

4. <http://www.openweathermap.org>

23 Méthodes généralisées

23.1 Classes abstraites

J'ai créé deux entités abstraites. La première *Entity Abstract* implémente la méthode PHP `__toString` qui permet de retourner une chaîne de caractères lorsqu'un objet est appelé dans un texte. Cette méthode est simple à mettre en place et a l'avantage de fournir un affichage clair dans les listes de données techniques.

De plus, lorsqu'une entité est appelée dans un champ de formulaire de type sélection, Symfony aura automatiquement la bonne information de nommage des éléments à afficher.

La deuxième entité *DevicesAbstract* me permet de généraliser la récupération des icônes des différents matériels et de fournir deux paramètres temporaires de référence à la table *Devices*. Ces deux paramètres me fourniront des détails importants quand j'aurai à manipuler des informations dans cet héritage précis.

23.2 Automator

Automator est un « Service » Symfony. Un service, dans ce cas, est une classe que l'on peut instancier depuis n'importe quel contrôleur et qui peut fournir des outils pratiques. Le service *Automator* que j'ai implémenté me permet d'automatiser la récupération des paramètres d'un objet et de les transférer vers un tableau afin de pouvoir les utiliser dans les vues.

Sans ce petit service, chaque paramètre doit être appelé individuellement dans la vue. Avec ce service, il est possible de boucler sur le tableau et d'afficher toutes les données techniques à la volée comme en figure 33.

23.3 Twig

J'ai profité d'avoir créé *Automator* pour étendre les fonctionnalités de Twig. Je lui ai ajouté une extension nommée `html`. Cette dernière va utiliser Automator pour découvrir le nom d'une entité brute et le décorer d'un lien `HTML` vers sa vue. Ainsi, comme on peut le voir en figure 33, le nom du contact « Jose Augusto » est un lien qui permettra d'afficher sa fiche complète et permettre la modification selon les rôles de l'utilisateur courant.

Customer's Contacts

Jose Augusto (First technical contact for the site)

Jose Augusto

Address PETROMAR:
Rua Rodrigo Miranda Henriques 15B,
Bairro Maculusso
Luanda - Angola

Phones [+244 923 42 65 35](#)
[+244 222 35 19 90](#)
[+244 936667884](#)

Email Jose.Augusto@saipem.eni.it
jaugusto@Petromar.co.ao

DVB (DVB) Edit

Parameter	Value
audio	+870 783 025 243
data56k	+870 783 024 640
data64k	+870 783 024 907
id	233
imei	351918030071789
ip	10.67.44.163
macip	10.67.44.163

(a) Appel manuel

(b) Boucle sur un tableau

FIGURE 33. Comparaison avec et sans le service Automator (Samuel Dorsaz)

24 Ajax

Une section *Devices* non assignés à été ajouté au système. Cette section permet de retrouver les liaisons manquantes entre les *Devices* et un *Site* distant ou caché. Cette page sert en quelque sorte de gestion des stocks et permet de transférer du matériel à un nouveau client.

La particularité technique de cette page tient dans le fait que le champ de recherche envoie une requête asynchrone au server web qui répondra par un liste d'informations permettant de mettre à jour un contenu spécifique de la page.

Cette technique a plusieurs avantages :

- Elle est plus légère et plus réactive qu'une requête standard.
- Elle permet l'interaction avec l'utilisateur.
- Elle permet de transmettre des données en parallèle.

Ce type de communication est déjà présente dans la « Toolbox » mais l'environnement des *Devices* non attribués est plus simple à comprendre. C'est donc un bon exemple que je voulais fournir à Newsat pour ses prochains développements.

25 Déploiement

25.1 Choix du système

J'ai constaté que la dernière version *LTS* d'Ubuntu est sortie très récemment. C'est donc une bonne occasion de pouvoir l'utiliser car elle est simple à prendre en main, robuste pour de l'hébergement et intègre toutes les nouvelles fonctionnalités du moment. La version 16.04 est supportée jusqu'en 2021, donc les mises à jour de sécurité garanties seront garantie durant cette période.

Avantages.

- Support pour les 5 prochaines années.
- Un noyau récent mais sans être le dernier (compatibilité avec Docker).
- Prise en main facilitée par une distribution populaire (même en tant que serveur).
- La version serveur par défaut a moins de paquets préinstallés.

La procédure d'installation détaillée n'est peut-être pas nécessaire dans notre cas mais voici quelques étapes intéressantes :

1. Téléchargement de l'ISO sur le site de <http://www.ubuntu.com/download/>.
2. Création d'une clef Universal Serial Bus (USB) d'installation. Le X de sdX doit être remplacé par la lettre du media que vous allez effacer. La commande peut durer plusieurs bonnes minutes selon la vitesse de vos ports et de votre clef USB :

```
sudo dd if=ubuntu-16.04-server-amd64.iso of=/dev/sdX
```
3. Démarrer la machine avec la clef et installer le système.
4. J'ai demandé un partitionnement standard de l'ensemble du disque dur.
5. Il n'est pas recommandé d'installer toutes les mises à jour automatiquement, mais celles de sécurité sont bienvenues.
6. Open Secure Shell (SSH) à été installé durant la procédure, on pourra ainsi se connecter avec PuTTY.
7.

```
sudo apt-get install aptitude vim htop docker-compose ubuntu-desktop virtualbox.
```

 - aptitude : gestionnaire d'installation avancé.

- vim : amélioration de l'éditeur de text en console.
 - htop : outil de monitoring en console très pratique.
 - docker-compose : pour déployer les conteneurs.
 - ubuntu-desktop : interface graphique.
 - virtualbox : pour ajouter un serveur **SQL** de test (optionel).
8. Ajouter l'utilisateur intranet au group docker `sudo usermod -a -G docker intranet`.
 9. Redémarrer pour prendre le groupe en charge.
 10. Copier le projet.
 11. Installer gitlab `docker-compose -f gitlab-compose.yml up -d` (avec -d permet de lancer en arrière-plan).
 12. Installer gitlab-runner `docker-compose -f runner-compose.yml up -d`.
 13. Installer le projet `docker-compose up -d`.

26 Documentation

Le rachat « récent » d'ITC Global par Panasonic (PANASONIC CORPORATION, 2015) aura posé une contrainte supplémentaire durant le projet : celle de commencer l'installation des postes de travail sous Windows 10. Cette nouvelle a rendu ce travail de Bachelor d'autant plus important que ce nouvel intranet est la seule solution qui leur permettra de continuer de travailler avec leur base technique.

Ce travail de Bachelor ne permettra pas d'implémenter la totalité de l'intranet de Newsat mais sa documentation a été écrite de manière à ce que des développeurs puissent reprendre le flambeau en cours de route.

Pour assurer la pérennité de cette dernière, elle a été incluse directement dans le code source (dossier /doc) et écrite au format Markdown. Ce format a la particularité d'être interprétable comme un page [HTML](#) et de rester agréable à la lecture si aucune autre application que `notepad.exe` n'était disponible (GRUBER, p.d.). Ce dossier inclut les guides suivant :

- Un guide d'installation (du serveur web)
 - Installation avec Docker
 - Installation traditionnelle
- Un guide pour les développeurs
 - Déploiement d'une nouvelle version de la « Toolbox »

HES-SO / newsat

Project Activity **Repository** Graphs Issues 12

Files Commits Network Compare Branches Tags

master newsat / doc / **Installation.md**

#5 Markdown documentation architecture and basic installation instructions
Samuel Dorsaz authored 3 months ago

69ef18e7 Browse Files

Installation.md 3.18 KB Raw Blame History Permalink Edit Replace Delete

Manuel d'installation

Logiciels requis

Docker

Docker est un ensemble de logiciel tout en un permettant de créer un virtualisation légère au sein d'un environnement Linux. Son gros avantage dans ce projet est de faciliter la configuration et le déploiement avec une configuration centralisée et déployable en une commande. Pour l'installer sur un environnement GNU/Linux Debian, vous pouvez utiliser la commande:

```
sudo aptitude install docker
```

Docker-compose

Docker-compose est un outils facilitant l'exécution des conteneurs Docker. Il permet notamment de préparer les commandes de lancement dans le fichier docker-compose.yml et de lancer le tout avec commande simple: docker-compose up. Pour l'installer sur un environnement GNU/Linux Debian, vous pouvez utiliser la commande:

(a) Affichage de la documentation dans Gitlab

HES-SO / newsat

Project Activity **Repository** Graphs Issues 12

Edit File Preview

master doc/Installation.md text

```

1 Manuel d'installation
2 =====
3
4 Logiciels requis
5 -----
6
7 ### Docker
8
9 Docker est un ensemble de logiciel tout en un permettant de créer un virtualisation légère au sein d'un en
10
11 Pour l'installer sur un environnement GNU/Linux Debian, vous pouvez utiliser la commande:
12
13 ---
14 sudo aptitude install docker
15 ---
16
17 ### Docker-compose
18
19 Docker-compose est un outils facilitant l'exécution des conteneurs Docker. Il permet notamment de prépare
20
21 Pour l'installer sur un environnement GNU/Linux Debian, vous pouvez utiliser la commande:
22
23 ---
24 sudo aptitude install docker-compose
25 ---
26

```

(b) Documentation au format texte

FIGURE 34. Accès à la documentation du projet

27 Conclusion

Ce travail m'a montré, comme je l'espérais, les limites de Symfony mais aussi l'efficacité de son utilisation. Si j'avais tendance à l'époque à éviter la console dans le cadre d'un Framework. Je ne peux que constater son utilité et sa puissance à notre époque. On a donc pu voir à quel point il était rapide de générer du code et même des pages entières à partir d'une base de données et de trois commandes.

Bien entendu quelques erreurs se sont glissées mais le résultat général était vraiment d'une grande aide. Si j'en ai l'occasion, j'espère pouvoir remonter mon expérience à la communauté afin d'améliorer les produits que j'ai utilisé.

Par contre à un niveau au-dessus, j'ai pu me rendre compte du grand nombre de différences à installer un serveur PHP sous Windows ou sous GNU/Linux. Je savais que monter un serveur sous GNU/Linux était plus évident mais je ne pensais pas que les différences de systèmes de fichiers allaient me poser des problèmes.

Le cas du driver pour Microsoft SQL Serveur différent sous GNU/Linux et sous Windows était passé inaperçu durant la phase d'analyse. Cet exemple montre que malgré les analyses il y aura toujours des détails qu'on ne peut pas planifier.

Les héritages dans la gestion du matériel m'a montré qu'un élément qui parait simple en surface peut très vite se compliquer après avoir déposé le Product Backlog initial. Je pense qu'il est intéressant d'avoir au moins identifié ce genre de structure pour mes prochaines estimations. Dans la même idée, les objets tout-en-uns comme la « Toolbox » sont des pièges à creuser dès qu'on les voit apparaître.

27.1 Améliorations possibles

L'automatisation du déploiement avec Docker et Gitlab-CI est un échec partiel. La problématique de gestion des droits ne m'était pas arrivée de cette manière avec d'autres Framework. Ce serait un sujet intéressant à approfondir afin de pouvoir activer cette fonctionnalité correctement.

Il reste beaucoup d'objets métiers secondaires pour lesquels il faudra créer des formulaires. Peut-être même qu'ils n'ont pas une réelle utilité et devraient être nettoyés du système.

La gestion Return Merchandise Agreement (RMA) et des tickets n'ont pas été implémentées mais ce sont deux sujets qui pourraient faire partie d'un prochain travail. L'autre partie pourrait être une migration de la base de données pour une structure plus propre.

La compatibilité avec les différentes versions d'Internet Explorer n'est pas assurées car la gestion des requêtes inter-domaines est assez complexe.

28 Références

3. Configuration - Doctrine DBAL 2 documentation. (nodate). Récupérée à partir de <http://doctrine-orm.readthedocs.org/projects/doctrine-dbal/en/latest/reference/configuration.html>
- CHARLES MORRIS, J. R. (2015, mai 6). A break from the past, part 2 : saying goodbye to activex, vbscript, attachevent... | microsoft edge dev blog. Récupérée à partir de <https://blogs.windows.com/msedgedev/2015/05/06/a-break-from-the-past-part-2-saying-goodbye-to-activex-vbscript-attachevent/>
- COMMUNITY, I. (2015, janvier 8). Whatsup gold restful-api and cli support - ipswitch community. Récupérée à partir de https://community.whatsupgold.com/network_server_monitoring/whatsup_gold/whatsup_gold_ideas/whatsup_gold_restful_api_and_cli_support
- DBAL-1068 : Microsoft SQL Server issues with ANSI_NULLS=OFF · Issue #1002 · doctrine/dbal. (2014, décembre 9). Récupérée à partir de <https://github.com/doctrine/dbal/issues/1002>
- DJANGO COMMUNITY. (nodate). La sécurité dans django | django documentation | django. Récupérée à partir de <https://docs.djangoproject.com/fr/1.9/topics/security/>
- Django documentation*. (2016, février 27). Django Software Foundation.
- Django mssql documentation*. (2015, novembre 10).
- EEVEE. (2012, avril 9). Php : a fractal of bad design. Récupérée à partir de <https://eevee/blog/2012/04/09/php-a-fractal-of-bad-design/>
- GRUBER, J. (nodate). Daring fireball : markdown syntax documentation. Récupérée à partir de <https://daringfireball.net/projects/markdown/syntax>
- IHRIG, C. J. (2013). *Pro node.js for developers*. Apress.
- KAPLAN-MOSS, J. (2016, février 29). Django/license at master · django/django. Récupérée à partir de <https://github.com/django/django/blob/master/LICENSE>
- LESIEUR, B. (2014, mai 17). Bootstrap est une régression pour un développement front-end de qualité. Récupérée à partir de <http://blog.lesieur.name/bootstrap-est-une-regression-pour-un-developpement-front-end-de-qualite/>

- Licences libres | AFUL. (2015, février 20). Récupérée à partir de <https://aful.org/ressources/licences-libres>
- MICROSOFT. (nodate). Comment acheter visual studio. Récupérée à partir de <https://www.visualstudio.com/fr-fr/products/how-to-buy-vs.aspx>
- MICROSOFT. (2013, février 18). Asp.net mvc 4 models and data access | the asp.net site. Récupérée à partir de <http://www.asp.net/mvc/overview/older-versions/hands-on-labs/aspnet-mvc-4-models-and-data-access>
- NAVED, I. (2013, septembre 28). Configuration for connecting to mssql server 2008 on virtualbox guestos from ubuntu 12.04 hostos using pyodbc 3.0.8 | iqbal nouyed. Récupérée à partir de <https://iqbalnaved.wordpress.com/2013/09/28/configuration-for-connecting-to-mssql-server-2008-on-virtualbox-guestos-from-ubuntu-12-04-hostos-using-pyodbc-3-0-8/>
- NEBRA, M. (2014, septembre 3). 5 questions à fabien potencier, créateur de symfony | openclassrooms : le blog. Récupérée à partir de <http://blog.openclassrooms.com/blog/2014/09/03/5-questions-a-fabien-potencier-createur-de-symfony/>
- NETWORK, M. D. (2015, mai 4). Contrôle d'accès http - http | mdn. Récupérée à partir de https://developer.mozilla.org/fr/docs/HTTP/Access_control_CORS
- O'NEAL, A. (2013, avril 22). How to create a windows (innosetup exe) installer (for nodejs apps). Récupérée à partir de <https://coolaj86.com/articles/how-to-create-an-innosetup-installer.html>
- Open Source License usage on GitHub.com. (2015, mars 9). Récupérée à partir de <https://github.com/blog/1964-open-source-license-usage-on-github-com>
- PANASONIC CORPORATION. (2015, mars 16). Panasonic to acquire itc global. Récupérée à partir de <http://www.panasonic.aero/Portals/0/Media/Press/PR-PDF/2015/31615-PanasonicAcquiresITCGlobal.pdf>
- PHP : Essentiel - Manual. (nodate). Récupérée à partir de <http://php.net/manual/fr/language.variables.basics.php>
- SENSIOLABS. (2016, février 29). *Symfony the cookbook*. 3^e éd.
- SHAFIK, D. (2014, juillet 9). Managing frontend dependencies & deployment part 1 : bower. Récupérée à partir de <https://blog.engineyard.com/2014/frontend-dependencies-management-part-1>

SYMFONY DOCUMENTATION TEAM. (2015, décembre 14). Symfony-docs/releases.rst at 4cd6dc2825924c9569621bf749f168a7ba2a235d · symfony/symfony-docs. Récupérée à partir de <https://github.com/symfony/symfony-docs/blob/4cd6dc2825924c9569621bf749f168a7ba2a235d/contributing/community/releases.rst>

TUDORICA, R. (2013, mai 10). Symfony2 mssql support with pdo_dblib - leaseweb labs. Récupérée à partir de https://www.leaseweb.com/labs/2013/05/symfony2-mssql-support-with-pdo%5C_dblib/

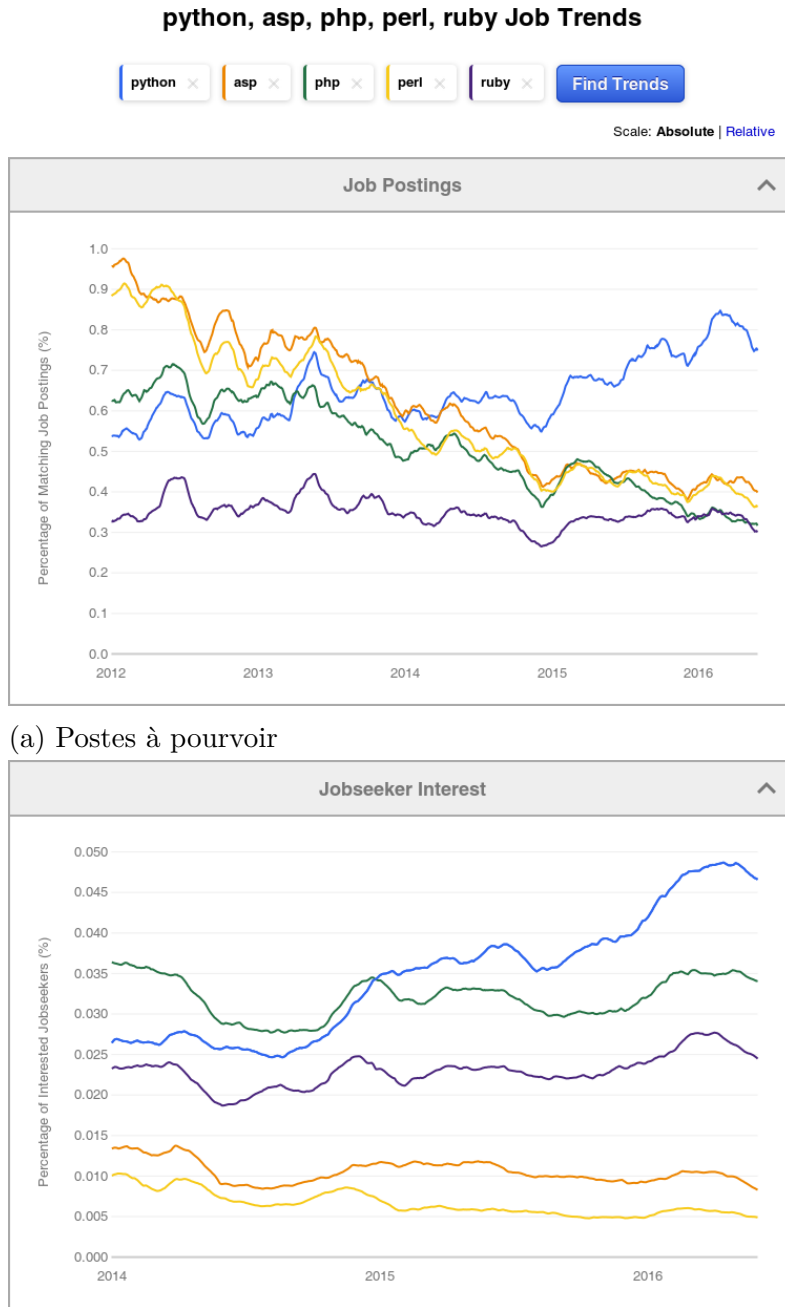
van der VOORT, J. (2015, septembre 22). Gitlab 8.0 released with new looks and integrated ci! | gitlab. Récupérée à partir de <https://about.gitlab.com/2015/09/22/gitlab-8-0-released/>

VLADUTU, A. (2014). Top 10 mistakes node.js developers make. Récupérée à partir de <https://www.airpair.com/node.js/posts/top-10-mistakes-node-developers-make>

WEBECS HOSTING. (2012, décembre 28). How do i configure sql server express to allow remote tcp/ip connections on port 1433? Récupérée à partir de <http://support.webecs.com/kb/a868/how-do-i-configure-sql-server-express-to-allow-remote-tcp-ip-connections-on-port-1433.aspx>

WEBECS.COM. (2012, décembre 28). How do i configure sql server express to allow remote tcp/ip connections on port 1433? Récupérée à partir de <http://support.webecs.com/kb/a868/how-do-i-configure-sql-server-express-to-allow-remote-tcp-ip-connections-on-port-1433.aspx>

Annexe I : Tendances des emplois aux Etats-Unis



(a) Postes à pourvoir

(b) Intérêt des demandeurs d'emploi

FIGURE 35. Tendance du marché du travail aux Etats-Unis

Annexe II : Droit d'auteur

Les drapeaux des pays ont été téléchargés sur www.365icon.com⁵. Ils sont libres d'utilisation pour des projets personnels et commerciaux. Je les ai utilisés pour remplacer les drapeaux actuels car le nom des fichiers est standardisé International Organization for Standardization (ISO) 3166-1 alpha-2.

Le menu supérieur a été étoffé de deux icônes supplémentaires. Celle d'un satellite dont les droits d'utilisation sont libre pour une utilisation non-commerciale⁶ et celle du matériel en stock dont les droits sont libres pour une utilisation personnelle⁷.

Enfin l'icône de l'application Toolbox à été téléchargée sur www.iconfinder.com. Elle est libre de droits pour une utilisation personnelle⁸.



FIGURE 36. Les icônes ajoutées au projet

5. <http://365icon.com/icon-styles/ethnic/classic2/>

6. <http://www.iconarchive.com/show/real-vista-networking-icons-by-iconshock/satellite-icon.html>

7. http://www.easyicon.net/language.en/1077571-add_files_to_archive_icon.html

8. https://www.iconfinder.com/icons/86483/toolbox_icon

Annexe III : Journal de travail

Jour	Durée	Description
2016-02-25 15 :00	45	Première réunion avec Alain Duc afin de se mettre d'accord sur l'organisation du travail.
2016-02-27 17 :07	14	Première création de Backlog
2016-02-27 17 :25	23	Ajout de données dans le Backlog
2016-02-29 18 :00	60	Analyses
2016-02-29 20 :43	29	Affiner les informations sur Django/Symfony
2016-02-29 22 :30	161	Détails sur Symfony
2016-03-01 01 :11	63	Ebauches UML des UC et Diagramme de Classe
2016-03-01 22 :33	94	Organisation
2016-03-03 15 :00	105	Discussion du Backlog
2016-03-04 07 :00	60	Remise en état du Backlog par rapport à la réunion de hier
2016-03-05 15 :01	93	"Ajout des priorités et des Story Points
2016-03-07 23 :10	100	Finalisation du Backlog initial
2016-03-08 21 :30	120	Comparaison de différents Frameworks
2016-03-13 20 :00	120	Tentative de mise en place d'une machine virtuelle avec Visual Studio et SQL Server
2016-03-14 22 :00	120	Fin de l'installation de MS SQL Server et analyse rapide de la base de données
2016-03-17 00 :15	105	Réalisation du cahier des charges
2016-03-17 13 :45	45	Préparation de la première Sprint review
2016-03-17 14 :45	80	Sprint review 0 -> 1
2016-03-17 19 :30	45	Réunion en cours de projet avec Alain
2016-03-20 08 :01	131	Analyses
2016-03-21 21 :00	60	Préparation de la machine virtuelle afin de servir la base de données et premières analyses du code
2016-03-21 22 :00	90	Préparation de l'environnement de développement
2016-03-22 07 :43	42	Analyses

2016-03-24 22 :30	46	Intégration continue
2016-03-26 09 :00	180	Analyses
2016-03-26 14 :00	120	Analyses
2016-03-29 07 :48	62	Analyse de la structure de la base de données
2016-03-29 18 :00	60	Analyse de la structure de la base de données
2016-03-29 22 :00	135	Création d'un diagramme de classe UML
2016-03-30 22 :00	144	Création d'un diagramme de classe UML
2016-03-31 15 :00	120	Réunion
2016-03-31 22 :00	294	Création du conténaire Docker
2016-04-01 08 :45	255	Création du conténaire Docker
2016-04-01 14 :30	90	Création du conténaire Docker
2016-04-02 09 :30	210	Création du conténaire Docker et installation de Symfony
2016-04-02 14 :30	150	Installation de Symfony
2016-04-03 17 :30	60	Installation de Symfony
2016-04-04 00 :30	120	Préparation des données
2016-04-04 07 :00	90	Création de la vue des appareils quelque soit leur type
2016-04-04 17 :40	80	Ajout de jQuery et Bootstrap à travers Assetics
2016-04-04 19 :20	70	Préparation de l'affichage des Sites
2016-04-04 21 :45	238	Création d'un menu hiérarchisé complet des Sites
2016-04-05 01 :43	20	Mise à jour du planning
2016-04-05 07 :30	88	Création de la vue des Site
2016-04-05 17 :30	30	Gestion de la sidebar pour les Sites
2016-04-06 21 :44	110	Gestion de la sidebar pour les Sites
2016-04-13 02 :17	88	Optimisation de la requête de la sidebar
2016-04-13 12 :36	0	Optimisation de la requête de la sidebar
2016-04-21 16 :45	116	Conciliation des problèmes de liaisons avec Doctrine
2016-04-21 19 :30	40	Pareil que précédemment
2016-04-23 15 :30	257	Résolution des problèmes de liaisons sur les Sites
2016-04-25 08 :15	175	"Formulaire d'édition des sites complété

2016-05-02 18 :30	75	Correction des problèmes d'assets
2016-05-02 20 :30	75	Correction de certains noms de colonnes dans les entités
2016-05-07 12 :15	45	Création de la documentation d'installation en Latex
2016-05-07 14 :36	219	"Ajout de la documentation
2016-05-11 18 :43	57	Installation de la machine de production
2016-05-11 21 :45	91	Installation de la machine de production
2016-05-12 12 :49	131	Installation de Gitlab
2016-05-12 16 :00	90	Installation de Gitlab
2016-05-16 20 :00	75	Improve some UI elements
2016-05-18 18 :00	60	Entities refactor
2016-05-19 12 :30	120	Entities refactor
2016-05-22 18 :30	60	Update Docker configuration
2016-05-23 19 :56	124	Update Docker configuration
2016-05-24 22 :30	182	Add first Devices to Sites
2016-05-26 10 :39	111	
2016-05-26 15 :00	90	Manage assets and Sidebar
2016-05-29 14 :00	270	
2016-06-02 15 :00	60	Fixing Docker configurations and working on Devices
2016-06-03 20 :34	56	Fixing Docker configurations
2016-06-04 22 :00	246	"Fixing Docker configurations
2016-06-06 20 :00	180	
2016-06-08 22 :00	60	Sidebar improvements
2016-06-09 14 :00	150	Mise en place du serveur web sans docker ni déploiement et présentation du projet
2016-06-09 16 :40	155	Add weather support
2016-06-12 00 :15	109	Configuration du user bundle
2016-06-12 14 :30	90	Rencontre avec un technicien de l'entreprise
2016-06-12 16 :00	120	Préparation des éléments pour la gestion des utilisateurs (teplate FOS user pour le login)

2016-06-14 21 :00	150	Proof of concept d'un serveur NodeJS
2016-06-15 19 :30	120	Implémentation de la sideBar de la page Satellites
2016-06-16 14 :30	120	Présentation de l'avancement
2016-06-16 15 :50	40	Implémentation de la sideBar de la page Satellites
2016-06-16 23 :30	120	Première implémentation du menu Data-link
2016-06-17 16 :30	30	Reprise du menu datalink pour avoir les connexions dans les deux sens
2016-06-17 17 :00	117	Fix view data render automatiser and create auto-mator service
2016-06-27 19 :45	135	try to have a working registration
2016-06-30 14 :00	270	Analyse des problématiques liées au FOS UserBundle
2016-07-01 13 :55	297	Analyse des problématiques liées au FOS UserBundle
2016-07-03 15 :00	270	Revue complète de la gestion des utilisateurs
2016-07-05 13 :00	60	Création des appels cURL pour l'API NodeJS
2016-07-06 18 :45	45	Création des appels cURL pour l'API NodeJS
2016-07-06 20 :30	195	Tests et résolution des problèmes liés au requêtes cross domaines
2016-07-07 14 :30	150	Déploiements et tests de la gestion des utilisateurs et de la première version de la ToolBox
2016-07-07 18 :00	44	Création du package Toolbox
2016-07-08 08 :00	60	Suivi du projet avec Alain Duc
2016-07-08 10 :00	150	Implémentation des premiers outils fonctionnels de la Toolbox
2016-07-08 13 :30	210	Implémentation des premiers outils fonctionnels de la Toolbox
2016-07-08 17 :30	47	Implémentation des premiers outils fonctionnels de la Toolbox
2016-07-10 10 :00	150	Update Sitelog structure
2016-07-10 13 :52	38	Update Sitelog structure
2016-07-10 19 :30	150	Add SitelogEntry

2016-07-11 19 :00	54	First implementation for Profiles and Roles
2016-07-11 21 :00	333	Improve user managements
2016-07-13 21 :30	120	Amélioration du datalink
2016-07-15 07 :00	300	Add Documentation menu with mounted network folder
2016-07-15 13 :00	345	"Add Documentation menu and manage file read/opening
2016-07-17 14 :00	240	Manage network folder exception
2016-07-17 23 :30	142	"Add newsat Contacts on Site
2016-07-18 20 :00	150	Documentation au sujet des relation hierarchiques au niveau de Symfony
2016-07-19 18 :45	60	Création d'un installer pour la Toolbox
2016-07-19 20 :30	60	Création d'un installer pour la Toolbox
2016-07-20 00 :00	147	Création d'un installer pour la Toolbox
2016-07-20 19 :30	60	Création d'un installer pour la Toolbox
2016-07-20 21 :30	261	"Finalisation d'un installer pour la Toolbox
2016-07-21 14 :15	255	"Create Remote tool for Toolbox
2016-07-21 22 :05	96	Improve Toolbox Installer
2016-07-22 07 :00	300	"Ajout de l'outil Whatsup
2016-07-22 12 :00	314	Ajout de l'outil Whatsup
2016-07-23 14 :00	120	Fix some contact links
2016-07-24 15 :00	360	Amélioration de la prise en charge des Contacts en ajoutant la gestion des Countries entre autre
2016-07-25 20 :00	225	Recherche d'une solution d'implémentation de la hiérarchie
2016-07-26 19 :30	388	Ecriture du rapport
2016-07-27 18 :52	98	"Add CRUD for Countries
2016-07-28 13 :30	300	Improve Data-link menu by a better filter and devices names
2016-07-28 19 :00	30	Improve Data-link performances

2016-07-29 08 :00	240	"Improve specific methods for Modem and Demod
2016-07-29 13 :00	210	Add Spectrum in Toolbox
2016-07-29 17 :00	60	Suivi du projet avec Alain Duc
2016-08-01 09 :00	210	Improve Spectrum tool
2016-08-01 13 :00	270	"Gestion des filtres de la vue DVB
2016-08-03 18 :30	60	Corrections diverses
2016-08-03 20 :30	120	"Gestions d'erreurs spécifiques lors de la lecture des Contacts
2016-08-04 14 :30	150	Corrections du Spectrum
2016-08-04 19 :00	210	Corrections du rapport
2016-08-05 08 :03	257	Corrections du Spectrum
2016-08-05 13 :20	192	"Corrections du Spectrum
2016-08-05 20 :30	120	Ajout d'un search Ajax pour les Devices non-assignés
2016-08-07 09 :00	180	Mise à jour du rapport
2016-08-07 13 :30	360	Mise à jour du rapport
2016-08-08 18 :45	475	Adaptations du rapport
2016-08-09 18 :30	60	Adaptations du rapport
2016-08-09 20 :15	120	Adaptations du rapport
	20437	340 heures



Backlog

#	Thème	En tant que	Je voudrais	Afin de	Validation	MuSCoW	Sprint	Priorité	SP
10	Recherches	Développeur	Analyser les différents frameworks existant	Se remettre à jour par rapport à ce qui se fait en ce moment, garder les avantages et éliminer les inappropriés	* Fournir le détail de 3-4 technologies * Proposer l'une d'entre elle	Must	0	100	18
20	Analyse	Développeur	Connaître la structure logicielle de l'entreprise	Proposer la meilleure technologie possible	* Établir une liste des systèmes et logiciels utilisés * Rapide analyse du choix des techno développées	Should	1	99	2
30	Analyse	Développeur	Connaître la structure matérielle de l'entreprise	Proposer d'installer le serveur web sur la machine la plus pratique	* Rapide analyse de l'emplacement possible du serveur web	Should	1	95	2
40	Analyse	Développeur	Analyser la base de données	S'assurer que le product backlog contienne bien tous les éléments avec une charge correcte	* Etablir un schema de la base de données * Modifications du Backlog	Must	1	94	8
50	Réalisation	Développeur	Proposer et monter une architecture pour le projet	Pouvoir présenter le projet au séances suivantes et proposer un espace de validation du code	* Monter un serveur de gestion de code * Configurer un système d'intégration continue * Avoir une base de données de test exploitable * Monter une plate-forme web de test	Must	1	90	8
60	Réalisation	Technicien	Pouvoir entrer des Appareils supplémentaire	Que les Techniciens puissent ajouter et modifier des Appareils réseau	* Avoir un formulaire création des Appareils * Pouvoir éditer ces Appareils	Must	2	88	4
70	Réalisation	Technicien	Pouvoir entrer des Sites supplémentaires	Afin d'ajouter les détails d'un Site	* Avoir un formulaire d'ajout de Site * Pouvoir y lier du Matériel	Must	2	88	8
80	Réalisation	Technicien	Pouvoir afficher des Sites	Recherche un Site en vue d'un dépannage	* Avoir une liste des Sites sur le côté gauche * La liste doit être maintenant hiérarchisée Site / Appareils * Pouvoir afficher les détails du Site	Must	2	86	4
90	Réalisation	Technicien Partenaire	Consulter le plan de connexion des appareils	Comprendre le réseau à dépanner	* Ayant la possibilité d'afficher un shéma réseau * Ce Schema doit être récupéré directement sur un système de fichiers du réseau	Must	2	85	4
100	Réalisation	Technicien	Pouvoir lier un Appareil à un Site	Afin de pouvoir accéder aux Appareils d'un Site	* Pouvoir ajouter des Appareils depuis le formulaire de gestion du Site	Must	3	84	4
110	Réalisation	Partenaire	Afficher la liste des Appareils	Afin de pouvoir les lier à des Sites / Téléports par la suite	* En ayant une liste déroulante d'Appareils * Pouvoir faire une recherche textuelle	Must	3	83	6
120	Réalisation	Technicien	Pouvoir connecter deux Sites ensemble	Afin de pouvoir connecter des Sites de type « Standard » à des Sites de type « Téléport »	* Avoir la possibilité de lier des sites de différents types	Must	3	82	2
130	Réalisation	Administrateur	Ajouter des outils	Que les technicien puissent les afficher et les utiliser	Ajouter des outils Permettre de lier au Matériel et/ou à ses paramètres	Must	3	81	4
140	Réalisation	Technicien	Afficher les outils	Connaître la panoplie d'outils à disposition en fonction des Appareils et de leur configurations	* Afficher la liste des outils avec leurs icônes * La liste doit être spécifique au Matériel et de ses configurations	Must	3	80	4



150	Réalisation	Technicien	Exécuter les outils	Pouvoir résoudre une panne et gagner du temps sur la configuration et le lancement	* Un cliquer sur l'icône doit permettre l'exécution Les paramètres doivent déjà être configurés	Must	4	79	8
160	Réalisation	Administrateur	Créer et modifier des droits d'accès selon des groupes de Techniciens	Pouvoir limiter certains groupes comme les Partenaires	* Avoir un formulaire de gestion des droits * Proposer la modification de la BDD pour les acl * attention à la hiérarchie et type (permissive / restrictive)	Must	4	78	16
170	Réalisation	Technicien	Pouvoir affiner les droits des Partenaires en fonction des Sites auxquels ils se rapportent	Empêcher un Partenaire d'accéder à trop de données	* Depuis un Site, pouvoir en donner le droit d'accès à un Partenaire	Must	5	77	12
180	Réalisation	Administrateur	Afficher une liste des droits en cours sur le serveur	Pouvoir identifier rapidement qui à les droits et s'il y a des erreurs	* Afficher une liste de droits effectifs sur le système	Must	5	76	4
190	Réalisation	Technicien	Afficher des données techniques d'un Matériel	En savoir d'avantage sur le Matériel à dépanner	* Ayant une page affichant les données technique d'un Matériel	Must	5	73	4
200	Réalisation	Partenaire	Afficher des données techniques d'un matériel	Fournir une aide aux Techniciens	* Afficher les données techniques	Must	6	70	2
210	Réalisation	Technicien	Créer et Modifier les contacts	Fournir les dernières informations aux techniciens	* Formulaire d'ajout / modification / suppression des contacts	Must	6	64	6
220	Réalisation	Technicien	Afficher les données d'un contact	Pouvoir contacter rapidement un Technicien sur place ou un Client	* Afficher un contact	Must	6	63	4
230	Réalisation	Technicien	Lier plusieurs contacts à un Site	D'avoir les bons contact sous la main lorsqu'un Site est affiché	* Pouvoir lier un Contact à un Site * Avoir une liste de contact pertinent dans les détails du Site	Must	6	61	8
240	Réalisation	Technicien	Ajouter / modifier un Log	Suivre le dépannage d'un Site	* Avoir un formulaire de création de Log	Should	6	55	8
250	Réalisation	Technicien	Créer des Notes dans un Log	Informé du traitement en cours du Log et des actions effectuées	* Avoir un formulaire d'ajout de Note * Choisir le Technicien ayant écrit la Note	Should	7	54	6
260	Réalisation	Technicien	Afficher un Log et ses Notes	Visionner les informations de dépannage	* Afficher les détails de base du Log * Afficher les notes qui vont avec	Should	7	53	4
270	Réalisation	Technicien	Pouvoir gérer un formulaire de retour de marchandise	Pouvoir retourner un produit sous garantie	* Afficher le formulaire RMA * Créer / Modifier un RMA	Would	7	45	8
280	Réalisation	Technicien	Ajouter un RMA à des Logs	Visionner les informations de retour rapidement depuis le Log	* Lier un RMA à un Log * Afficher le RMA dans le Log	Would	7	43	4
290	Réalisation	Administrateur	Ajouter modifier supprimer les processus métiers	Afin de les adapter pour les techniciens	Formulaire d'ajout de Processus Chargement des données existantes (édition) Bouton de suppression	Would	8	25	8



300	Réalisation	Technicien	Afficher un processus métier (BPM) et activer les étapes en cours	Afin de rester informer sur l'avancement d'un processus	* Affichage du fichier SVG * Possibilité d'activer une étape	Would	8	23	8
310	Réalisation	Technicien	Lier un BPM à un RMA	Afin d'accéder rapidement à l'avancement du processus à partir du Log	* Formulaire permettant d'attacher le BPM à un RMA	Would	8	21	2
320	Réalisation	Technicien	Ajouter modifier supprimer des Pays	Maintenir les données quasi constantes	* Formulaire d'ajout de Pays * Chargement des données existantes (édition) * Bouton de suppression	Could	8	10	4

28 Déclaration sur l'honneur

Je déclare, par ce document, que j'ai effectué le travail de Bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées. Je ne donnerai aucune copie de ce rapport à un tiers sans l'autorisation conjointe du Responsable de filière et du professeur chargé du suivi du travail de Bachelor, y compris au partenaire de recherche appliquée avec lequel j'ai collaboré, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail et que je cite ci-après :

- M. Alain Duc
- M. Yann Bocchi
- M. Olivier Mayoraz

Sierre, le 10 août 2016

Samuel Dorsaz