Lycée Jean Bart - Dunkerque
2011-2012

Comparaison de base de données et mise en place de Git

Stechele Julien

Entreprise d'accueil : IdentIt
1294 rue Achille Pérès
Petite-Synthe

Tuteur de stage : M.Anselin
Maître de stage : M.Dubourg

Remerciements

Je tiens à remercier :
- M.Dubourg pour le suivi qu'il m'a apporté pendant toute la durée du stage, le temps qu'il m'a consacré pour m'initier à la programmation orientée objet, les astuces techniques pour développer plus rapidement ainsi que l'enseignement des habitudes propres à l'entreprise ;
- La société IdentIt pour avoir accepté ma candidature, j'espère avoir été à la hauteur de leurs attentes ;
- L'équipe de développeurs pour m'avoir aiguiller quand j'étais en détresse ;
- L'équipe enseignante pour nous avoir appris les fondements du développement et l'univers informatique tout autour qui en découle.

Sommaire

1 Introduction **3**

2 La gestion de version **4**

2.1 Git : Le gestionnaire de code source 4

2.1.1 Les rudiments 4

2.1.2 L'installation 7

2.1.3 Le passage fatidique 7

2.2 Le tutoriel utilisateur 8

2.3 Les utilitaires 8

2.3.1 Lister les dépôts du serveur 8

2.3.2 Mise en production automatique 10

2.3.3 Mise à jour locale automatique 10

3 Script de comparaison **11**

3.1 Le besoin 11

3.2 Les prémices 11

3.3 Le script PHP 13

3.3.1 Comparaison des tables 13

3.3.2 Comparaison des champs 14

3.3.3 Affichage du résultat 14

4 Conclusion **16**

1 Introduction

Je m'appelle Julien Stechele, je suis actuellement en BTS [1] informatique de gestion option développeur d'applications et dans le cadre de mes études j'ai la chance d'effectuer deux stages durant cette formation. Mon stage de première année s'est déroulé du 16 mai 2011 au 8 juillet 2011 dans la société *IdentIt* qui se situe à Petite-Synthe. C'est une SARL [2] de quatre personnes, M.Dubourg et M.Lesage sont les fondateurs de cette structure, le premier s'occupe de la partie technique en tant que chef de projet et le second de la partie gestion de l'entreprise. Ils sont accompagnés par deux développeurs, l'un travaillant sur la partie application sur Windows Mobile ©, l'autre sur la partie internet.

Mon premier travail consistait à développer un utilitaire pour la partie web de comparaison de base de données qui permetterait d'améliorer le suivi des mises à jour d'une base obsolète à partir d'une base de référence et aussi de fournir les requêtes permettant cette mise à jour. Mon deuxième travail était de mettre en place un nouvel outil de gestion de version de code source plus évolué que l'existant.

Le premier jour en entreprise fut un peu spécial pour moi. En effet, mon entretien s'étant passé un vendredi après-midi, seuls les gérants de l'entreprise étaient présents, du coup cela a été pour moi l'occasion de rencontrer les deux développeurs, Cyril [3] tout juste arrivé dans l'entreprise et Ludovic, développeur expérimenté qui a de nombreuses années de programmation derrière lui.

Cette note de synthèse suit un plan spécifique, mais les événements ne ce sont pas passés dans l'ordre réel. En effet, à cause des difficultés rencontrées et des problématiques nouvelles données en cours de stage, je suis souvent passé d'une réalisation à une autre.

1. *Brevet de Technicien Supérieur.*
2. *Société À Responsabilité Limitée.*
3. Titulaire d'un BTS IG à Jean Bart promotion 2009.

2 La gestion de version

2.1 Git : Le gestionnaire de code source

Les logiciels de gestion de versions ou VCS [1] sont utilisés principalement par les développeurs. En effet, ils sont quasi exclusivement utilisés pour gérer des codes sources, car ils sont capables de suivre l'évolution d'un fichier texte *ligne de code par ligne de code.* Ces logiciels sont fortement conseillés pour gérer un projet informatique.

Ils retiennent qui a effectué chaque modification de chaque fichier et pourquoi. Ils sont par conséquent capables de dire qui a écrit chaque ligne de chaque fichier et dans quel but ; si deux personnes travaillent simultanément sur un même fichier, ils sont capables d'assembler (de fusionner) leurs modifications et d'éviter que le travail d'une de ces personnes ne soit écrasé. Ces logiciels ont donc par conséquent deux utilités principales :

- suivre l'évolution d'un code source, pour retenir les modifications effectuées sur chaque fichier et être ainsi capable de revenir en arrière en cas de problème ;
- travailler à plusieurs, sans risquer de se marcher sur les pieds. Si deux personnes modifient un même fichier en même temps, leurs modifications doivent pouvoir être fusionnées sans perte d'information.

2.1.1 Les rudiments

Un commit correspond à un enregistrement des modifications dans le temps. Admettons un fichier qui contient un paragraphe, si nous ajoutons un deuxième paragraphe, le fichier sera considéré comme modifié par Git [2], pour enregistrer la modification on effectue un commit. L'analogie la plus simple est celle des jeux vidéos où vous sauvegardez votre progression à chaque étape franchie.

1. *Version Control System.*
2. Créé par Linus Torvalds, qui est entre autres l'homme à l'origine de Linux. Il est de type distribué.

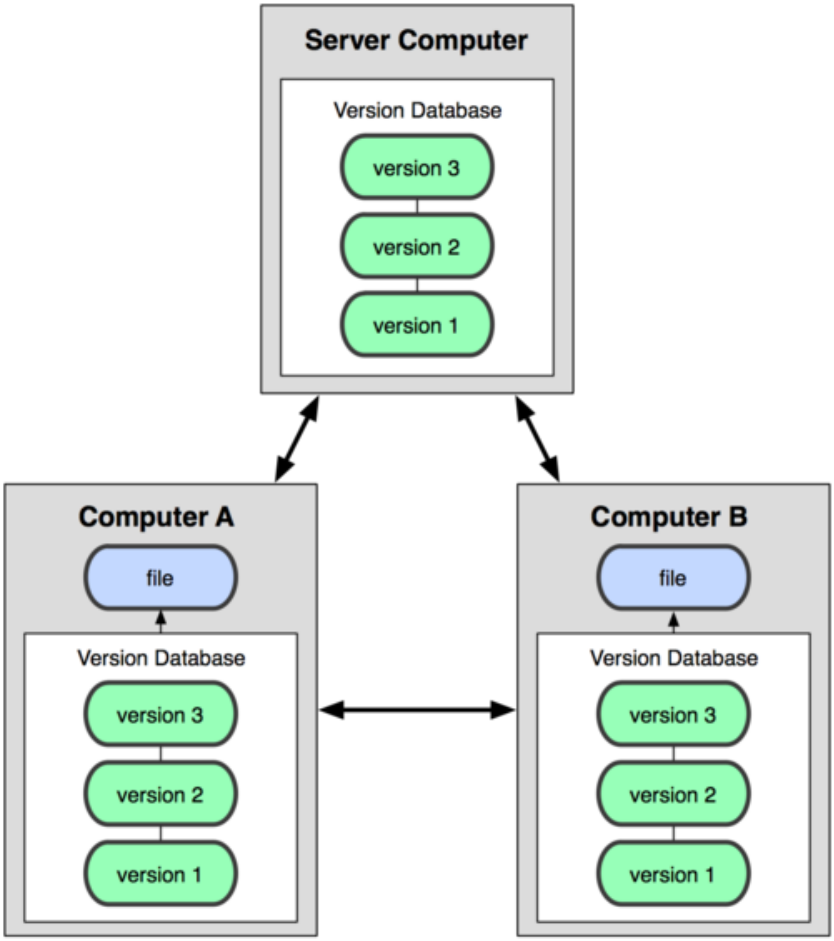

FIGURE 2.1 – Chaque utilisateur possède une copie du code source ainsi que toutes les versions précédentes de celui-ci, ce qui n'est pas le cas de l'existant. Le serveur sert de point de rencontre entre les développeurs et possède lui aussi l'historique des versions.

Une branche représente une « copie virtuelle » du dossier contenant les fichiers sources. En effet, il est possible de cloner virtuellement un dépôt et de basculer de l'original à la copie pour développer, par exemple, sur la branche principale les nouvelles fonctionnalités et sur l'autre branche les corrections d'erreurs comme le montre la figure 2.2.

La fusion est l'opération de rassemblement des deux branches en une, c'est-à-dire dans le cas précédent de regrouper les corrections d'erreurs avec les nouvelles fonctionnalités.

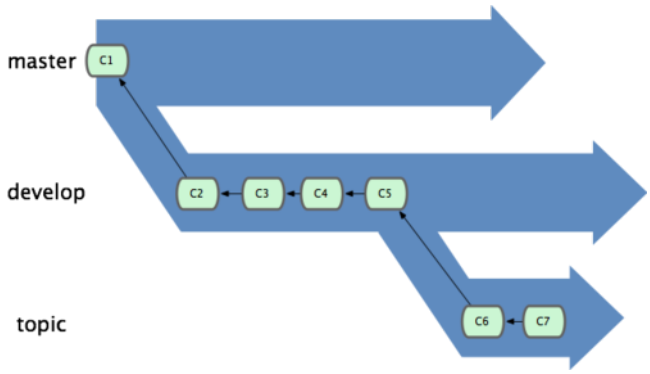

FIGURE 2.2 – master, develop et topic, trois branches d'un même dépôt.

Les fondamentaux ayant été acquis, j'ai trouvé plusieurs solutions à la problématique principale qui est la suivante :

> *Peut-on fusionner des branches tout en choisissant de ne pas fusionner tout les commits ?*

1. Tout dabord la commande `git chery-pick nomducommit` permet de « cueillir » le commit que l'on veut fusionner ;
2. Ensuite on peut aussi rassembler les deux branches puis exécuter la commande `git revert nomducommit` pour inverser les modifications du commit après la fusion ;
3. Enfin faire trois branches distinctes pour ne pas avoir à faire les opérations ci-dessus [3].

Git répondant au besoin de l'entreprise, il a fallu que j'effectue des recherches pour que la migration Subversion [4] vers Git se fasse sans perte.

3. Cette solution à été choisie.
4. Le logiciel de gestion de versions le plus utilisé à l'heure actuelle. Il est de type centralisé.

2.1.2 L'installation

À peu près au milieu du stage, nous nous sommes heurtés à un problème technique. Le serveur de l'entreprise ne contenait pas de version récente de Git. En sachant que celui-ci est mutualisé [5], nous n'avions pas les autorisations nécessaires pour installer une nouvelle version.

Les logiciels libres sont réputés pour être rétro-compatibles, mais en toute logique il ne dispose pas des nouveautés sans les mettre à jour. Le problème est qu'on ne pouvait pas modifier des branches distantes dites « de suivi ». Cette fonctionnalité permet à plusieurs développeurs de travailler en même temps sur une branche différente de celle par défaut, par exemple pour expérimenter de nouvelles implémentations à plusieurs.

À ce moment-là j'ai été très déçu et pensais mon travail inutilisable. Cependant, mes efforts n'ont pas été vain puisque M.Dubourg après quelques recherches, a trouvé sur internet un utilisateur ayant réussi à installer Git sur un serveur mutualisé. Nous n'avions pas les droits d'accès aux dossiers des programmes mais rien ne nous empêchait de compiler Git à partir des sources pour nous permettre de créer en local le logiciel [6]. Ceci étant fait, en redéfinissant la variable système qui stocke les chemins des applications, nous avons pu utiliser notre compilé dernier cri.

2.1.3 Le passage fatidique

Trois jours avant mon départ, le grand déménagement s'impose. Toute la partie étude et confection du tutoriel Git prend forme car j'ai basculé tous les projets qui étaient au préalable sur Subversion vers Git tout en gardant l'historique des changements provenant de SVN. Après quelques recherches, une commande Git m'a permis de répondre à ce besoin.

```
git-svn clone http://svn/repo/here/trunk
```

5. L'hébergement mutualisé est un concept d'hébergement internet destiné principalement à des sites web, dans un environnement technique dont la caractéristique principale est d'être partagé par plusieurs utilisateurs. L'administration du ou des serveurs est assurée par un intervenant tiers tel qu'OVH.

6. Le problème ne se serait pas posé si j'avais lu le chapitre traitant de la compilation dans mon livre Linux, je n'ai découvert cette méthode qu'après le stage...

2.2 Le tutoriel utilisateur

Pour aider les développeurs a intégré le nouveau gestionnaire de version, j'ai élaboré un document Microsoft Word © de quatorze pages qui explique les bases du logiciel, la mise en place de l'outil dans leurs environements de travail réspectifs et l'utilisation de celui-ci avec le gratuiciel TortoiseGit. Pour être au plus près des problèmatiques qui pourraient être rencontrés, j'ai conçu l'intégralité du guide sous forme de questions et de réponses. Sans rentrer dans les détails, je liste ici les chapitres qui le compose :

1. Les bases.
2. Comment mettre en place Git sur Windows ?
3. Comment gérer les dépôts Git ?
4. Comment utiliser Git ?
5. Comment utiliser les branches ?
6. Comment consulter l'historique des modifications ?
7. Quelle est la methode de travail ?
8. Annexe.

2.3 Les utilitaires

Nous avons longuement discuté sur le côté technique de Git. En effet cet outil est à la base un logiciel libre provenant de Linux et n'a pas d'interface graphique, ce choix est établi sur le fait qu'un développeur n'a pas forcément besoin de cela pour travailler [7] et aussi que ce logiciel regorge de fonctionnalités et donc très difficile à simplifier à travers une interface ne pouvant fonctionner qu'avec une souris. L'équipe n'étant pas très férue de ligne de commande et le logiciel TortoiseGit reprenant que les fonctions essentielles de Git, il a fallu que je conçois des petits programmes qui fonctionnent en un clic pour simplifier les choses répétitives.

2.3.1 Lister les dépôts du serveur

Mon travail suivant consistait à lister les dépôts Git dans une page PHP [8]. Le serveur OVH centralise les codes sources de la société, les développeurs,

7. Sans nul doute que pour un graphiste, une souris est un élément indispensable pour ses créations, mais pas pour produire du code.

8. *Hypertext Preprocessor* est un langage de scripts libre principalement utilisé pour produire des pages Web dynamiques.

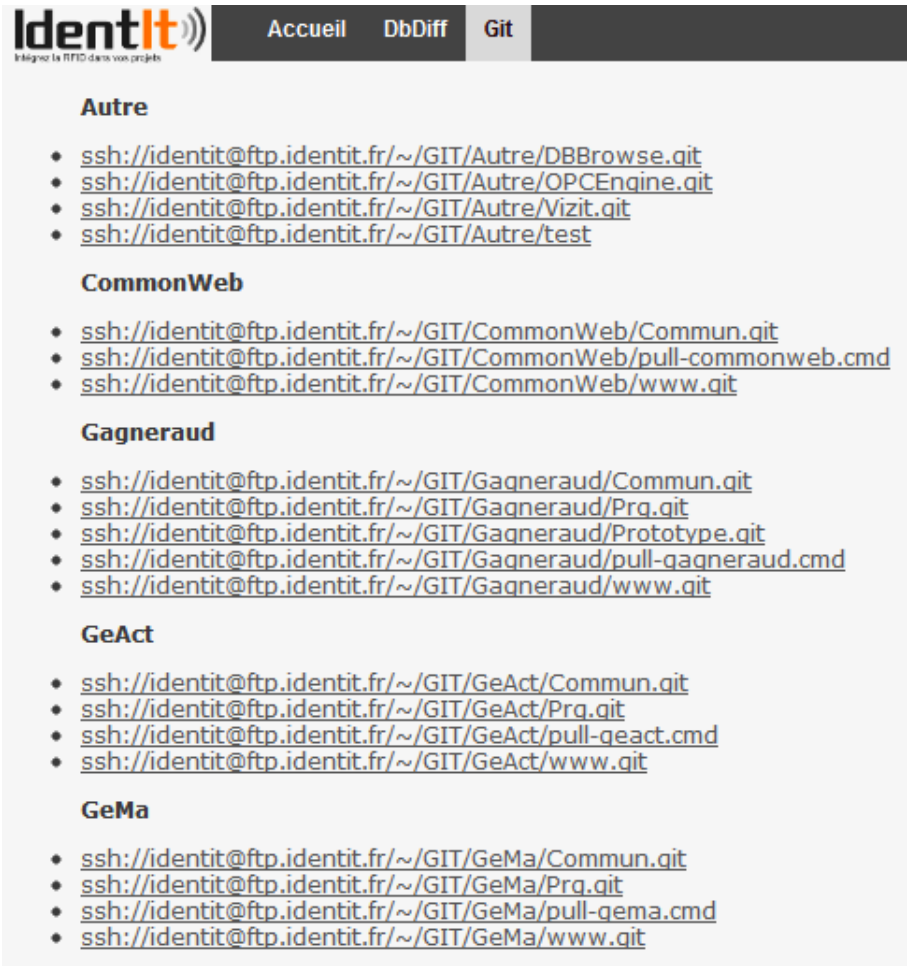

FIGURE 2.3 – La liste des dépôts cliquables.

eux doivent récupérer une copie des dossiers pour pouvoir travailler dessus. Seulement, lancer FilleZilla [9] pour connaître les noms des répertoires Git puis recopier le bon lien hypertexte avec le bon chemin d'arborescence est une tâche fastidieuse. J'ai donc, à l'aide de mes connaissances en système UNIX, créé un script qui analyse un dossier défini et qui liste les différents dépôts tout en leurs mettant les bonnes adresses de téléchargement en préfixe comme nous le montre la figure 2.3 en page 9. Du coup, un simple copier-coller du lien du dépôt voulu dans tortoiseGit suffit pour cloner. Gain de productivité et de simplicité réunie.

9. FileZilla est un logiciel gratuit qui permet de se connecter à distance sur un serveur pour y télécharger des fichiers.

2.3.2 Mise en production automatique

Une fois que les développeurs sont satisfaits de leurs modifications, ils doivent mettre à jour le dépôt distant pour partager leurs travaux. Une fois les modifications validées par le chef de projet, celui-ci doit mettre en ligne sur le dépôt de production. J'ai créé un script Batch « modèle » pour que cela se fasse en un clic.

```
set PATH=%HOMEPATH%;%PATH%
plink identit@ftp.identit.fr -l identit -pw MOTDEPASSE
"cd LEDOSSIER; git pull; exit;"
```

2.3.3 Mise à jour locale automatique

Les développeurs possèdent des clones des dépôts distants [10] en local pour bien évidemment maintenir le code source. Le fait qu'il y ait une multitude de dépôts implique une mise à jour des clones locaux pour récupérer les modifications des autres développeurs ce qui est très répétitif. J'ai donc créé un script Batch [11] qui parcourt tous les dossiers englobant les sources des dépôts et les mets à jour. Pour cela j'ai parcouru la documentation de la console windows et ça n'a pas été évident du tout. Pour l'anecdote, je devais à partir de mon MacBookPro [12], lancer une machine virtuelle Windows © [13] pour créer mon script Batch qui s'exécute en console, et qui lance enfin un terminal Linux pour faire la mise à jour...

```
"C:\Program Files\TortoiseGit\bin\pageant.exe"
C:%HOMEPATH%\.ssh\id_rsa.ppk
"C:\Program Files (x86)\git\bin\sh.exe" --login -i -c
"for i in $(find . -maxdepth 2 -mindepth 2 -type d);
do
    cd $i;
    echo $i;
    git remote -v;
    git pull; cd ../..;
done"
```

10. Ils se trouvent tous sur le serveur OVH qui sert de point de rencontre comme nous l'avons vu.

11. Désigne un fichier qui contient une suite de commandes qui seront traitées automatiquement par Windows ©.

12. Système de type UNIX.

13. À partir du logiciel VirtualBox.

3 Script de comparaison

3.1 Le besoin

Le but de ce script est de consulter les différences qui existent entre une base de référence est une base à mettre à jour. Dans le cas de l'entreprise, il permettrait un suivi des mises à jour des applications fournies au client et pour moi, m'initier à la programmation orientée objet [1] dès le premier stage.

N'étant pas à l'aise avec la programmation orientée objet [2] je n'ai pas pu rejoindre les développeurs de l'entreprise dans l'application qu'ils développaient, cela dit on m'a confié une autre tâche qui est la comparaison de base de données. Le schéma 3.1 en page 12 nous en illustre une structure basique.

3.2 Les prémices

À l'aide d'un cours sur internet, j'ai commencé à créer ma première classe. Cette classe après instanciation représenterait l'objet « base de données » sous forme de tableau. L'image 3.2 qui ce trouve en page 12 est plus parlante.

Passer de la programmation fonctionnelle [3] à l'objet fut vraiment difficile. Me rendant compte que je bloquais énormément, je fis des recherches sur internet pour trouver un programme équivalent sur lequel je me suis appuyé pour commencer. M.Dubourg m'a mis sur la voie en me disant d'utiliser des classes et des méthodes toutes prêtes de l'entreprise ce qui me fit gagner beaucoup de temps, car je savais pas comment transcrire une structure de base de données en objet.

Pour utiliser les sources de la société, j'ai dû mettre en place un répertoire

1. La programmation orientée objet est un paradigme de programmation qui consiste à utiliser des objets ; un objet représente un concept, une idée ou toute entité du monde physique, comme une voiture, une personne ou encore une page d'un livre.
2. Ce qui sera enseignée en deuxième année.
3. La programmation fonctionnelle est un paradigme de programmation qui repose sur l'utilisation majoritaire des fonctions et procédures.

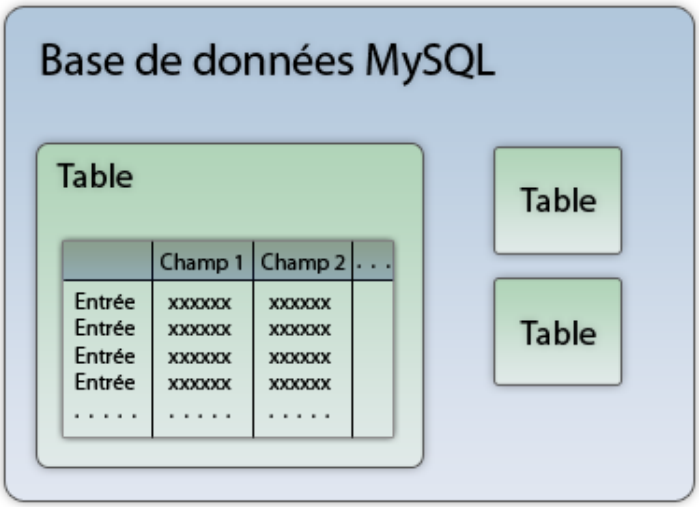

FIGURE 3.1 – Un schéma de base de données simple.

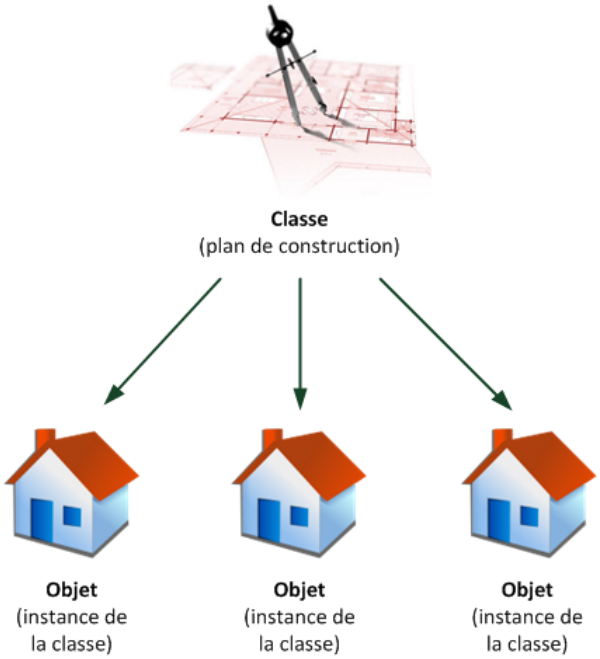

FIGURE 3.2 – Un plan à partir duquel on crée des objets.

de travail et récupérer les sources via leur ancien gestionnaire de version de code source. Ceci étant fait il s'agissait maintenant de réussir à faire fonctionner le site web principal en local sur ma machine. Comme mon logiciel MAMP [4] m'affichait plein d'erreurs, j'ai décidé d'installer manuellement chacun des logiciels présents dans celui-ci. Même après cette manipulation le problème n'avait pas disparu, mon tuteur de stage m'est venu en aide après de longues heures à chercher en vain. Le problème était tout d'abord au niveau de la configuration de PHP, qui affichait tous les avertissements de manière trop stricte, ce qui entravait le lancement de la page principale. Il était aussi au niveau du cache de mon navigateur. En effet, j'ai du vider celui-ci pour faire enfin apparaitre le site proprement. De nombreuses heures de recherche juste à cause d'un cache internet non vidé fut extrêmement frustrant...

3.3 Le script PHP

3.3.1 Comparaison des tables

Il a fallu dans un premier temps que je recherche comment extraire le nom d'une base ainsi que le nom de ses tables. La réponse à cette question est dans la documentation de MySQL. Une base de données est fournit à l'installation et s'appelle « information schéma » . En résumé, c'est une base de données qui contient toutes les autres.

Ensuite, j'ai transformé le programme procédural sur plusieurs semaines en objet, le fait de passer par cette étape intermédiaire m'a permis d'abord de résoudre le problème algorithmiquement, puis de me consacrer sur la façon de l'écrire.

La requête récupérant les données utiles, j'ai dû concevoir un algorithme capable de comparer les deux tableaux d'objets par rapport à leurs noms :

- Si la base de données de référence a une table qui n'est pas dans la base de données à mettre à jour, on l'ajoute ;
- Si la base de données à mettre à jour contient une table qui n'est pas dans la base de données de référence, on l'enlève ;
- Si les tables comparées sont toutes les deux dans les bases de données, on compare l'intérieur des tables.

J'ai dû revoir mon algorithme plusieurs fois, car la fonction de comparaison de chaine de caractère `strnatcmp` compare les mots comme un humain le ferait, alors que mySQL trie les tables de manière binaire grâce aux codes

4. *Macintosh Apache MySQL PHP* est une combinaison de logiciel.

Classement	Ordre standard	Ordre naturel
1er	table1	table1
2e	table10	table2
3e	table12	table10
4e	table2	table12

TABLE 3.1 – La différence entre les tris de chaînes.

ASCII [5] des caractères ce qui faisait que la première fonction donnait des résultats erronés comme le montre le tableau 3.1.

3.3.2 Comparaison des champs

J'étais dans la situation des tables ayant les mêmes noms, il fallait que je compare le contenu à partir du nom des champs. Il ne m'a pas fallut longtemps pour comprendre que l'algorithme était exactement le même, le plus dur étant de savoir comment j'allais faire pour ne pas me répéter, ce qui a bloqué énormément mon avancé pour pas grand-chose. M.DUBOURG constatant que je stagnais, m'a conseillé de faire comme j'avais appris plutôt que d'essayer de faire de la POO [6]. Une fois le script fonctionnel, qui était cependant mal optimisé et peu lisible, mon tuteur me guida avec des explications poussées sur la marche à suivre. De fil en aiguille le code source passa d'environ trois-cent cinquante lignes à cent cinquante.

Ensuite, je devais comparer le contenu des champs des tables lorsque les champs parcourus avait le même nom. Je comparais tout simplement le contenu des champs pour connaitre l'issue finale, à savoir que je ne pouvais pas décider si les tables étaient similaire sans avoir balayé tous les champs et toutes les caractéristiques de ceux-ci.

3.3.3 Affichage du résultat

Après tout ceci fini et optimisé, j'implémente une nouvelle fonctionnalité qui permettrait d'afficher un texte lisible qui énonce les différences des deux bases de données. En fonction des résultats obtenus, je génère des balises

5. *American Standard Code for Information Interchange.* ou « Code américain normalisé pour l'échange d'information » est la norme de codage de caractères en informatique la plus connue, la plus ancienne et la plus largement compatible.

6. *Programmation Orientée Objet.*

HTML[7] et du texte pour que cela soit compréhensible à l'utilisateur. Ceci étant fait assez rapidement je suis passé à la génération des requêtes SQL[8] permettant de mettre à jour la base obsolète depuis la base de référence. C'est à ce stade que j'ai constaté que je ne pourrais pas prendre en compte les clés étrangères dans mon code à moins de revoir totalement tout le script. M.Dubourg m'a rassuré sur le fait que l'entreprise n'utilisait pas le type de base de données myISAM[9] et que donc aucune de leurs tables comportait de clé étrangère, cependant les clés primaires concaténées restent problématiques, car pas imaginer pendant la conception. Nous avons décidé que la création de ce genre de cas se fera à la main pour ne pas repartir de zéro.

Après avoir analysé mes méthodes de génération de phrases lisibles et de requêtes SQL, M.Dubourg m'a présenté un outil nommé « Smarty » qui permet de dissocier la partie traitement de la partie affichage, car il est vrai que mes méthodes étaient vraiment très sales. Pour résumer, j'écrivais des chaînes de caractères dans une seule variable en écrivant les noms des tables ou des champs, en concaténant le tout avec des balises HTML de saut de ligne un peu n'importe où. J'ai dû parcourir la documentation de Smarty pour comprendre son fonctionnement, il s'avère que la syntaxe est très particulière mais très efficace. J'ai terminé par faire de l'agencement sur ma page pour que chaque ligne lisible par un néophyte soit en face de la ligne traduite en SQL dans un tableau à deux colonnes.

7. L' *Hypertext Markup Language* est un langage de balisage conçu pour représenter les pages web.

8. *Structured Query Language* est un langage informatique normalisé qui sert à effectuer des opérations sur des bases de données.

9. *Indexed Sequential Access Method* ou L'organisation séquentielle indexée, est un mode d'organisation des fichiers dans les bases de données.

4 Conclusion

La première chose qui me vient à l'esprit est la progression que m'a apporté ce stage. Je peux dire en toute humilité que j'en ressors grandi. L'utilitaire de comparaison m'a donné beaucoup de fil à retordre car à chaque fois qu'il fonctionnait, M.Dubourg me poussait à être de plus en plus critique sur ce que je faisais et à recommencer en voyant les choses de manière différente. J'ai pu connaitre enfin le travail de développeur en entreprise et je dois dire que ça me plait beaucoup cependant ce n'est pas toujours évident. Bloquer sur un problème toute la journée sans avancer est très pénible intellectuellement, j'ai connu beaucoup de maux de tête et de matins difficiles. Travailler toute la journée a concevoir des algorithmes est éprouvant, et encore je ne subissais aucune pression quant à ma productivité...

J'ai été très satisfait des échanges que j'ai pu avoir avec l'équipe. Développer est une chose mais proposer, débattre, trouver les meilleures solutions sont ce que j'ai préféré. L'entreprise m'a beaucoup apporté, mais je pense aussi avoir apporté des choses et c'est très gratifiant.

Cette période m'a conforté dans mes choix et dans ma poursuite d'étude. Ma passion pour les logiciels libres me pousse vers la licence professionnelle « logiciels libres et propriétaires pour systèmes, réseaux et bases de données », par contre ma force de proposition m'orienterais plus vers la « spécialité assistant chef de projet informatique ».