

Evolution d'une application web dans la Gestion Financière des Télécoms

Rapport d'alternance

Auteur: Arnaud.FELIX

Publication: 22/06/2009

Maitre d'alternance : Jean-Marie LAGARDE

Tutrice : Catherine COMPAROT

Entreprise d'accueil : Memobox – 2G Technologies

Formation : Master Nouvelles Technologies en
Informatique pour l'entreprise

REMERCIEMENTS

Je tiens à remercier en premier lieu l'entreprise MEMOBOX – 2G TECHNOLOGIES de m'avoir accueilli afin d'y réaliser mon année d'alternance et plus particulièrement Monsieur Denis MALLET, Directeur technique de l'entreprise, pour m'avoir permis de l'effectuer à ses côtés.

Qu'il me soit permis d'exprimer mon immense gratitude et mon profond respect à Monsieur Denis MALLET pour sa sympathie et son bon accueil. Cela m'a permis de m'intégrer rapidement.

Je voudrais remercier les collaborateurs du service Recherche & Développement pour leurs disponibilités et leurs conseils qu'ils m'ont prodigués au cours de cette année.

J'adresse mes remerciements à Madame Catherine COMPAROT, professeur tutrice, pour m'avoir suivie tout au long de cette année et le soutien qu'elle a apporté à ma candidature.

Je remercie également Monsieur Jean-Marie LAGARDE, maître d'alternance et responsable R&D, de m'avoir conseillé durant cette année.

Enfin je remercie tous les membres de l'Institut Universitaire Professionnel Nouvelles Technologies de l'Informatique pour l'Entreprise (NTIE).

SOMMAIRE

| | |
|---|-----------|
| REMERCIEMENTS | 3 |
| SOMMAIRE | 4 |
| INTRODUCTION..... | 5 |
| I. PRESENTATION DE L'ENTREPRISE | 6 |
| 1. L'HISTORIQUE | 7 |
| 2. LE SECTEUR D'ACTIVITE | 8 |
| 3. LES CONCURRENTS..... | 9 |
| 4. L'ORGANISATION..... | 10 |
| 5. L'ETAT FINANCIER..... | 11 |
| 6. LES PRODUITS | 12 |
| II. ORGANISATION ET ENVIRONNEMENT DE TRAVAIL | 13 |
| 1. MATERIEL | 14 |
| 2. POUR LE DEVELOPPEMENT..... | 15 |
| 3. POUR LE TRAVAIL COLLABORATIF..... | 15 |
| 4. POUR LA SYNCHRONISATION SERVEUR..... | 15 |
| 5. PERIODE TYPE..... | 16 |
| 6. ORGANISATION D'UNE EVOLUTION PRODUIT | 16 |
| III. PARTIE TECHNIQUE | 19 |
| 1. PLANIFICATION ANNUELLE | 20 |
| 2. ETUDE DE AUDITELCOM V6..... | 21 |
| a. <i>Descriptif</i> | 21 |
| b. <i>Détail technique</i> | 21 |
| c. <i>Analyse de la base de données</i> | 23 |
| 3. EVOLUTION MAJEUR DE L'OBJET ARBRE | 24 |
| a. <i>Etude de l'existant</i> | 25 |
| b. <i>Travail réalisé</i> | 29 |
| c. <i>Les Tests</i> | 37 |
| d. <i>Les Résultats</i> | 38 |
| 4. MODIFICATION DES WORKFLOW | 40 |
| a. <i>Etude de l'existant</i> | 40 |
| b. <i>Travail réalisé</i> | 44 |
| 5. ANALYSE D'OBJETS METIERS V7 | 46 |
| a. <i>Etude de l'existant</i> | 47 |
| b. <i>Travail réalisé</i> | 47 |
| c. <i>Classe AccessManager</i> | 49 |
| d. <i>Paquetage Profil</i> | 50 |
| e. <i>Classe DataAccessManager</i> | 50 |
| CONCLUSION..... | 51 |
| TABLE DES ILLUSTRATIONS..... | 52 |
| GLOSSAIRE | 53 |
| SOURCES DOCUMENTAIRES | 54 |
| INDEX | 55 |

INTRODUCTION

Effectuer une année d'alternance dans une entreprise représente une réelle possibilité d'acquérir de l'expérience pour des jeunes en formation. Le Master 2 Nouvelles Technologies de l'Informatique pour l'Entreprise offre cette opportunité.

Ayant suivi cette année de formation durant l'année 2008–2009, j'ai pu effectuer cinq cycles d'alternances, sous la forme de cinq semaines en entreprise et trois semaines d'IUT. Cette année a commencé le 21 septembre 2008 pour se terminer le 31 aout 2009.

J'ai eu la possibilité de réaliser cette alternance dans l'entreprise « MEMOBOX - 2G TECHNOLOGIES » qui possède une division sur Toulouse, plus précisément dans la zone de Montaudran. Elle se place sur le marché de la Gestion Financière des Télécoms.

Etant une entreprise avec un nombre de collaborateurs très restreint, elle m'a attiré par son côté humain. Cela m'a laissé envisager la possibilité de m'épanouir le côté technique et le plan relationnel de l'autre côté. Tout en y rajoutant une valeur ajoutée à mes compétences.

Durant cette année, j'ai effectué les missions qui m'étaient confiées au service Recherche & Développement. J'ai travaillé en étroite collaboration avec le responsable technique de la société: Monsieur Denis MALLET. J'ai été suivi par Madame Catherine COMPAROT en ce qui concerne la partie cours universitaire et par Monsieur Jean-Marie LAGARDE, responsable R&D, pour la partie entreprise.

L'entreprise crée et édite des logiciels pour de nombreux clients. Ils sont composés de grands Comptes et d'une multitude d'entreprises de type PME.

Ma mission fut de prendre en main l'application existante AUDITELcom, tout en y apportant des évolutions. Cela exigeait de moi d'avoir de bonnes connaissances dans le domaine de la conception Web Objet ; tout en y ajoutant un sens aigu de l'analyse et du développement de qualité.

On trouvera dans ce dossier une description plus détaillée de mon entreprise d'accueil, ainsi que le développement des points techniques que j'ai abordé cette année.

I. Présentation de l'entreprise



MEMOBOX est une entreprise présente depuis 1994 sur le marché des télécoms en entreprise. Crée à l'origine pour apporter aux professionnels des solutions de communication sécurisée entre les PABX et les outils micro-informatiques de gestion. Elle a pris un virage important en 1997, en s'orientant vers le service, grâce à des solutions hébergées de taxation et d'analyse du trafic téléphonique.

L'évolution du marché et les solutions techniques lui ont permis de développer une gamme complète de services de Gestion Financière des Télécoms (GFT), couvrant aujourd'hui l'ensemble du domaine financier des télécoms en entreprise : téléphonie fixe, téléphonie mobile, VoIP, ToIP, données, équipements, contrats associés, abonnements, consommations ...

En 2007, l'entreprise réalise sa première opération de croissance externe en prenant le contrôle de 2G TECHNOLOGIES, société spécialisée dans l'édition et la vente de logiciels de gestion financière des télécoms. Elle réalise également une levée de fonds d'un million d'Euros, ce qui lui permet d'accroître encore son avantage concurrentiel en accentuant ses investissements en R&D renforçant ses équipes et déployant une stratégie commerciale adaptée aux attentes de ses clients en France et à l'étranger.

1. L'*historique*

- 1994: création de MEMOBOX.

A l'origine de MEMOBOX, deux spécialistes, l'un des services de télécommunications: Christophe Fornès et l'autre du service réseaux: Denis Mallet. Ensemble, ils décident en 1994 de développer et commercialiser leur propre ligne d'interfaces de communication pour PBX: EdelBox. Ces interfaces sont commercialisées auprès des éditeurs de logiciels de taxation et d'analyse de trafic téléphonique. MEMOBOX est né et ne compte aucun salarié.

- 1995: développement commercial.

Poussée par ses premiers clients, MEMOBOX étend sa gamme en créant un système de taxation téléphonique autonome pour les TPE-PME: Taxabox, décliné également en version hôtelière (Hotelbox). Pour se faire, elle embauche son premier salarié pour étendre ses capacités de développement.

- 1997: le pari du service avec AUDITELcom®.

A cette date, MEMOBOX compte 3 salariés. Après le succès rencontré auprès des éditeurs de logiciels et intégrateurs en télécoms, MEMOBOX pressent le besoin des entreprises en matière de gestion des coûts télécoms et pour s'adapter à son marché, décide de lancer AUDITELcom® le premier service de GFT (gestion financière des télécoms) en mode hébergé. Une augmentation de capital accompagne ce nouvel axe de développement.

- 2003: MEMOBOX renforce sa position.

Le déploiement des 800 premiers sites nationaux du ministère de l'Intérieur est bouclé en 4 mois et le service satisfait pleinement le client. Pour réaliser cette prouesse, MEMOBOX a consolidé ses équipes de développement et d'exploitation. L'effectif est porté à 13 personnes.

- 2004: les deux fondateurs de MEMOBOX rachètent la totalité de l'entreprise.

La bonne santé de MEMOBOX en 2004 (+23,96% de chiffre d'affaires) permet à Christophe Fornès et Denis Mallet de racheter à Global Concept la totalité des parts que ce dernier détenait dans MEMOBOX. La société est désormais filiale à 100% du groupe TEL&COMS FM (Facilities Management) dont Christophe Fornès et Denis Mallet sont les deux seuls actionnaires. AUDITELcom® compte maintenant plus de 2500 sites raccordés et représente désormais 91% du chiffre d'affaire de la société. Un ingénieur commercial a été recruté pour promouvoir le service auprès des grands-comptes, en s'appuyant sur les fonctionnalités et l'expérience acquises grâce au ministère de l'Intérieur. Le service doit évoluer pour répondre au plus près à la demande de ce nouveau profil de clients.

- 2007: MEMOBOX renforce sa stratégie.

Forte de sa politique de développement, MEMOBOX décide de renforcer sa stratégie :

- En octobre 2007, elle rachète 2G TECHNOLOGIES à sa maison mère, QUESCOM.
- En décembre 2007, elle lève 1 million d'Euros auprès d'ALTO Invest.

2G TECHNOLOGIES a été créée en 2002, lors de la reprise de la société GENIE TELECOM par la société GEOSOFT. Ces deux structures historiques travaillaient depuis une dizaine d'année à l'édition et la commercialisation de logiciels d'analyse de trafic téléphonique et de CTI (Computer Telephony Intégration).

2. Le secteur d'activité

En misant résolument sur le service, la société française MEMOBOX est accès sur le service, elle a une place à part sur le marché français des solutions de GFT (Gestion Financière des Télécoms en France, TCM : Telecom Cost Management à l'international) à destination des entreprises. AUDITELcom®, son fer de lance, s'y impose aujourd'hui comme la seule offre de service commercialisée en mode hébergé SaaS (Software As A Service) et vaut à MEMOBOX, forte de 3200 clients, de compter parmi les leaders dans son secteur.

MEMOBOX a anticipé l'augmentation de la consommation et du budget télécom des entreprises. C'est ainsi qu'elles disposent d'outils d'analyse, de contrôle et d'aide à la décision... sans avoir pour autant à se préoccuper de gérer matériels ou logiciels supplémentaires.

MEMOBOX a mis au point avec AUDITELcom® un outil qui passe au crible les données de l'autocom et les rend facilement lisibles. Consommation par poste, centre de frais, établissement ou rapports d'alertes signalant les situations inhabituelles sont autant d'éléments indispensables pour prendre des mesures correctives, qu'il s'agisse de modifier les prestations d'un opérateur ou les habitudes au sein de l'entreprise.

Aujourd'hui, MEMOBOX affine encore AUDITELcom® Elle intègre la gestion des factures électroniques (EDI) des opérateurs, la gestion de flotte de mobiles ou la gestion des actifs et des budgets, de plus en plus demandées par les entreprises comme par les services publics et collectivités locales.

En 2007, MEMOBOX a développé une stratégie établie autour de deux axes:

- une première opération de croissance externe en rachetant à QUESCOM sa filiale d'édition de logiciels de GFT 2G TECHNOLOGIES (installée à Toulouse) pour enrichir son offre de service avec des produits logiciels.
- une levée de fonds de 1 million d'Euros est réalisée ceci afin de renforcer son leadership en France et conquérir le marché européen.

Depuis la reprise de 2G TECHNOLOGIES en octobre 2007, MEMOBOX a délégué sa R&D à Toulouse, implantation historique de l'entreprise sous la responsabilité de Denis MALLET.

3. Les concurrents

Il existe plusieurs catégories de fournisseurs sur le marché :

- Les fournisseurs plutôt destinées aux opérateurs ou grosses multinationales. Ces derniers sont elles-mêmes de grosses structures qui développent des solutions lourdes, faites pour gérer des chaînes complexes de facturation et de suivi clients. Une seule société française est présente sur ce marché, il s'agit d'HIGHDEAL, spin-off de FRANCE TELECOM, qui s'est fait une place au niveau international.
- Les fournisseurs de solutions destinées aux entreprises. Ce marché couvre toutes les solutions, du simple système de « taxation », chargé de vérifier les communications sur un PABX et destiné aux petites entreprises, aux systèmes plus complexes capables de centraliser les informations de plusieurs sites, de prendre en compte des coûts fixes, d'intégrer les coûts des mobiles, etc. Au niveau mondial, il n'existe que quelques sociétés présentes (AVOTUS, MTS, TELSOFT, CONTROL POINT SOLUTION), les autres sont plutôt présentes sur des marchés locaux ou domestiques.

4. L'organisation

Voici l'organigramme du service Recherche & développement dans lequel j'ai été affecté.

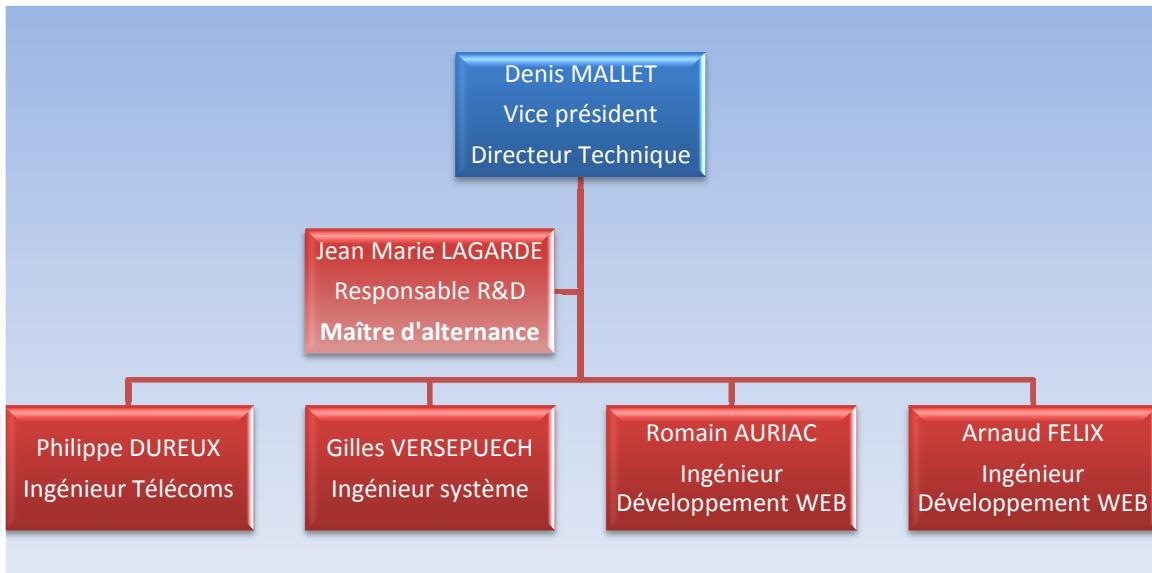


Figure 1 Organigramme du service R&D

En ce qui me concerne, ce fut Jean-Marie LAGARDE qui m'a suivi administrativement. En revanche sur le plan technique c'est sous la direction de M. Denis MALLET, responsable de l'application AUDITELcom sur laquelle j'ai travaillé.

5. L'état financier

Memobox - 2G Technologies voit son chiffre d'affaire rester pratiquement stable depuis trois ans, compte tenu du rachat en 2007 de 2G Technologies. Avec une évolution majeure pour le secteur de 2G Technologies en 2008

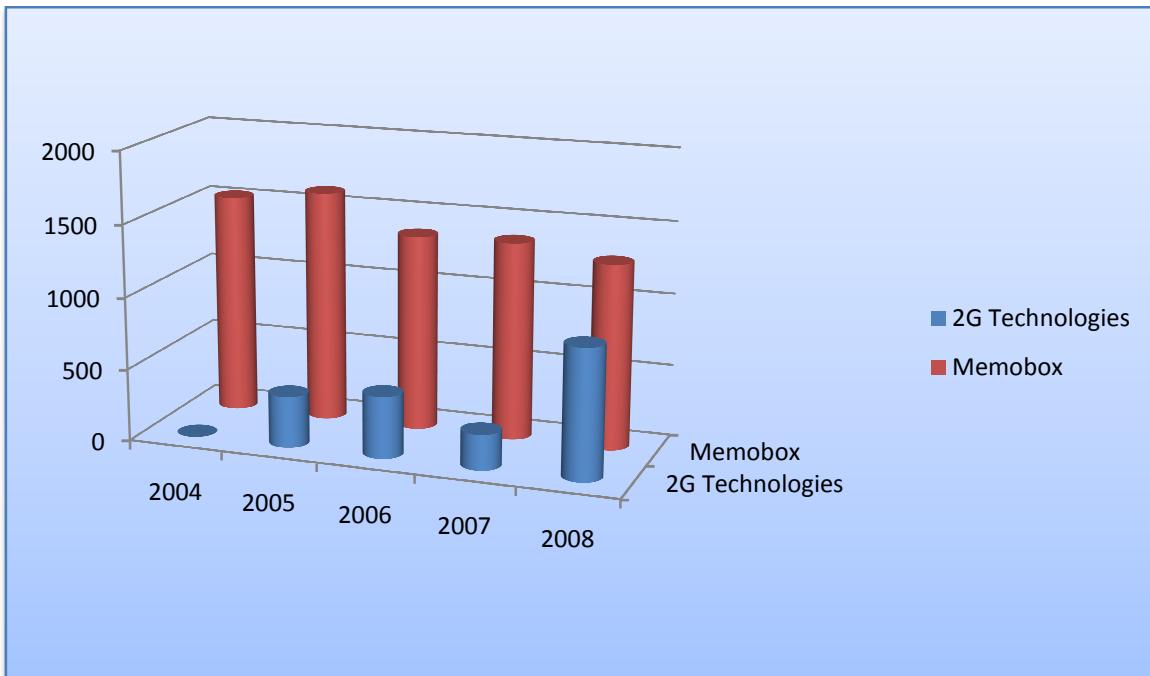


Figure 2 Graphique du Chiffre d'affaire (en K€)

| | 2004 | 2005 | 2006 | 2007 | 2008 |
|------------------------|------|------|------|------|------|
| 2G Technologies | - | 357 | 430 | 245 | 900 |
| Memobox | 1530 | 1610 | 1360 | 1370 | 1280 |

Figure 3 Tableau du Chiffre d'affaire (en K€)

Avec une part de marché d'environ 13% par an, le groupe se positionne en leader sur le marché français.

6. Les produits

2G TECHNOLOGIES a développé les gammes de logiciel:

- GeoTaxe ES (analyse de trafic pour les entreprises). Progiciel destiné aux grandes sociétés, comportant une machine hébergé par les clients, collectant leurs données pour les mettre à leurs dispositions ; via leurs infrastructures internet. Il est développé en ASP – XSL (présentation), VB – DELPHI (cœur applicatif), avec une base de données SQL Serveur. Il en est à sa version 7.0.8
- GEOTEL (gestion des appels téléphoniques dans les chambres d'hôtel).
- GEOPITAL (gestion des appels téléphoniques en milieu médicalisé).
- GEOVOX (serveurs vocaux d'entreprise).

MEMOBOX, a aussi développé les gammes de logiciels :

- AUDITELcom®.Application sous forme de service, destinée aux grands comptes.
- AUDITELbox : Appliance à installer chez le client de petite taille. Il permet la réalisation d'un audit des télécoms en temps réel

II. Organisation et Environnement de travail



Cette partie est consacrée à la description des outils utilisés lors de mon apprentissage, aussi bien du niveau technique que physique. Cela va vous permettre d'avoir une idée plus précise sur mon environnement de travail.

1. Matériel

Durant toutes les périodes d'entreprise j'ai travaillé sous environnement Windows XP SP3. Ce système était installé sur un ordinateur Dell de la série « OPTIPLEX » avec un processeur Pentium 4 cadencé à 2.80GHz, avec 1.5 Go de mémoire vive.

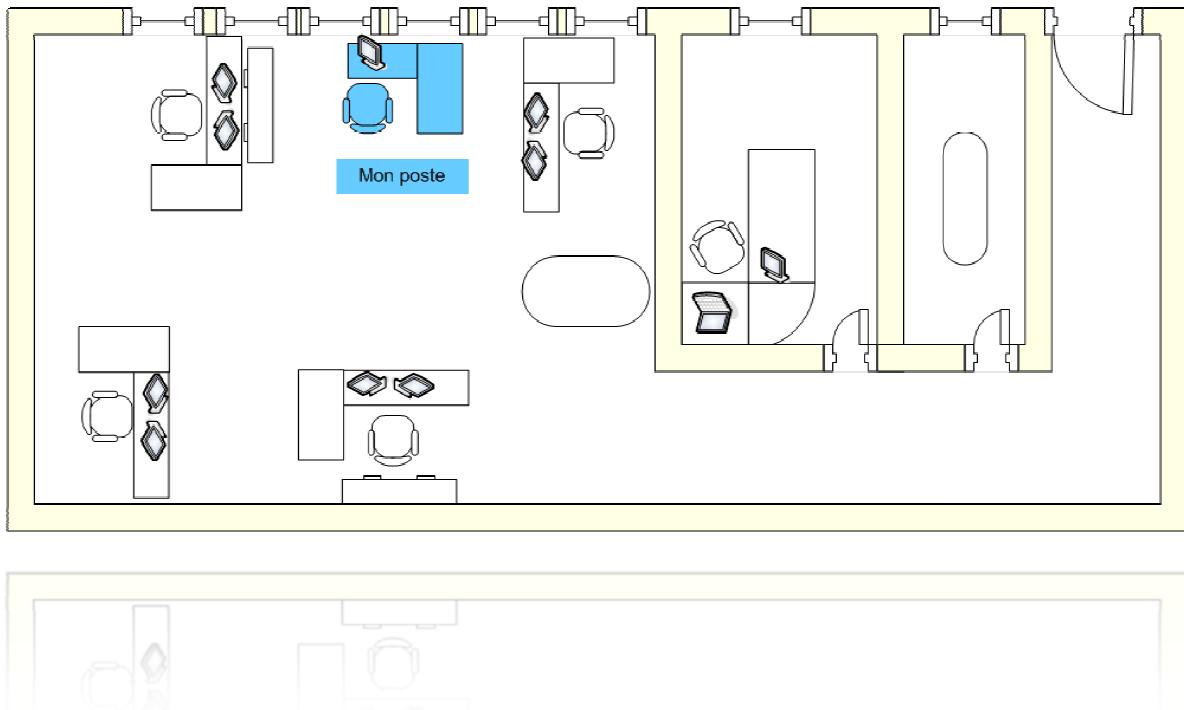


Figure 4 Plan du site R&D de Toulouse

Travaillant sur des applications orientées web, j'ai effectué tous mes tests sur un serveur web distant. Ce serveur est installé dans les locaux à Paris. On y accède via un réseau privé virtuel (VPN). Ce serveur est un ordinateur Dell sous système FreeBSD. Ce système gratuit dérivé d'Unix BSD est par sa nature optimisée pour les plateformes de type serveur ou système embarqué.

2. Pour le développement



J'ai utilisé l'environnement de développement Eclipse. Il a l'avantage d'être gratuit et très modulable; grâce aux nombreux projets qui ont été réalisés pour cette plateforme. En ce qui me concerne, pour pouvoir avoir un environnement de développement en PHP, j'ai rajouté le projet « PHP Development Tools » (PDT) sous sa version 1.0.3. Il permet de faire de l'analyse syntaxique ainsi que de naviguer entre les classes et les fonctions.

3. Pour le travail collaboratif

N'étant pas seul sur le projet, il a fallu mettre en place des outils permettant le travail collaboratif. Pour cela tous les projets de la société sont regroupés sous un système de gestion de versions (SVN). Ce système permet d'enregistrer toutes les modifications effectuées sur un fichier. Ainsi chaque collaborateur peut mettre à jour, comparer ou récupérer une ancienne version d'un fichier. Pour avoir un environnement Eclipse complet, il y a été rajouté le projet « subclipse » proposé par Tigris.org. Il contient svnClientAdapter et SVNKit.



Pour pouvoir se partager les tâches, la société utilise le système FlySpray. Cet outil permet de concentrer toutes les actions à réaliser pour faire une version de produit. L'avantage de ce produit est de permettre d'affecter des ressources et de suivre l'avancement de chaque tâche. Ainsi chaque personne à un tableau de bord lui permettant d'organiser son temps de travail en fonction de la granularité des tâches qui lui sont affectées.



4. Pour la synchronisation serveur

Pour travailler sur le serveur web à distance j'utilise l'outil WinSCP. Il permet de naviguer dans les répertoires et les fichiers d'un serveur UNIX à la manière d'un explorer Windows. Il possède un éditeur de texte simpliste qui permet de visionner des documents.



5. Période Type

Pour permettre de comprendre mon mode de fonctionnement, je vous présente un planning type d'une période de travail.

Au début de chaque période je fais un point avec le responsable du produit, pour avoir un retour sur l'état d'avancement du produit. Cette réunion permet de fixer les objectifs à atteindre pour la période en cours.

Suite à cette réunion, j'examine les tâches FlySpray qu'ils m'ont été affectées ce qui permet d'avoir une vision globale des tâches à réaliser.

Une fois mon travail organisé je me lance dans la réalisation des tâches. Durant cette phase je réalise de nombreux tests unitaires pour évaluer mon travail. Dès lors que la tâche me semble terminée, je passe en phase d'intégration. Pour cela je me connecte à une base de données qui contient un nombre de n-uplets beaucoup plus important.

Une fois que les tests d'intégration se sont bien passés, je fais évoluer l'avancement de la tâche FlySpray. Cela permet d'informer le responsable produit de mon avancement.

6. Organisation d'une évolution produit

Le fait que l'on soit intégré dans le monde du travail doit nous permettre d'avoir une vision plus claire sur la manière de gérer un produit logiciel. En effet, étant au cœur d'un processus de gestion de projet, il me paraît important de connaître les éléments qui le composent. Ainsi je vais vous parler des étapes qui permettent de sortir une nouvelle version d'un produit dans la société 2G TECHNOLOGIES.

Tout processus a un début et une fin en ce qui concerne celui-ci le début est représenté par des demandes d'évolution clients et la fin par la mise à disposition d'une nouvelle version aux clients.

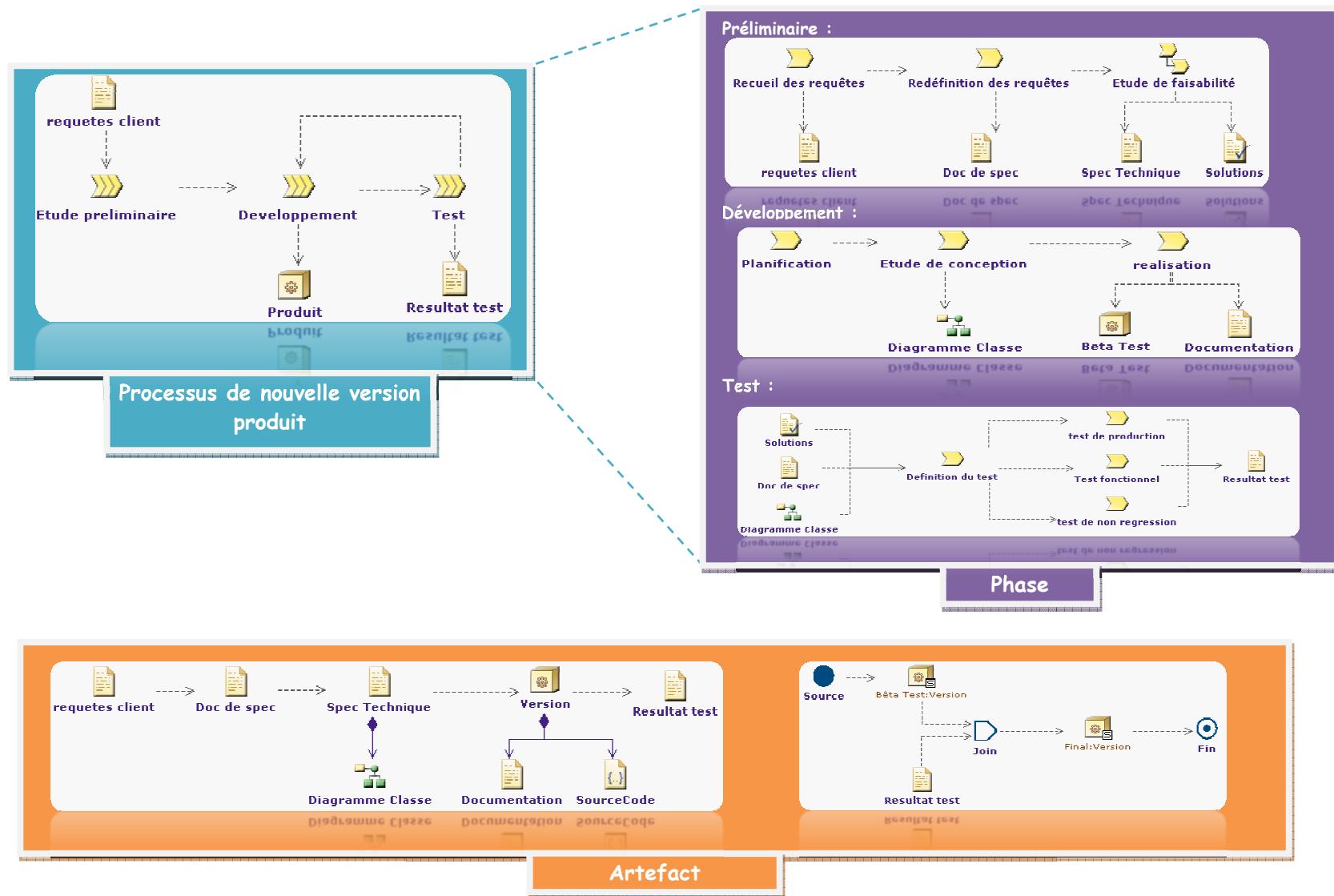


Figure 5 Modélisation du processus d'évolution de produit

Les requêtes clients sont recueillies par les commerciaux ou par le service clientèle. Elles sont ensuite examinées par la direction du service clientèle afin de permettre d'en ressortir les principales informations. Cet examen va déboucher sur la production d'une première documentation de spécification. Elle regroupe les demandes clients de manière formalisées.

Cette documentation va être d'une part communiquée aux clients ayant fait les demandes d'amélioration et d'autre part au service technique. Ainsi celui-ci va pouvoir commencer à identifier les modules et les parties de code concernés. Cette estimation va permettre de faire un premier chiffrage, avec une estimation de la durée de réalisation et son coût.

Une fois que celle-ci est rédigée, une 2^e phase de rencontre est organisée, elle regroupe les trois acteurs : clients, commerciales, service clientèle et service technique. Elle doit permettre de trouver un point de convergence entre chaque partie, concernant les fonctionnalités à inclure dans la nouvelle version. Elle se conclut en faisant une liste des jalons à tenir.

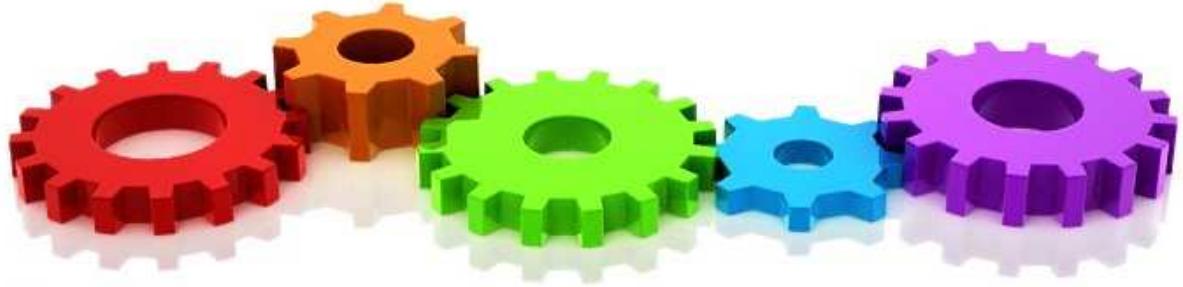
Ces indications sont ensuite traitées par le service R&D qui les transforme en tâches sur l'outil FlySpray. Durant cette étape une réunion est organisée, entre les responsables techniques et les ingénieurs, pour planifier les actions à mener ainsi que sur les délais à tenir.

Une fois que la partie planification est réalisée, on passe à la phase développement. Il se peut que des versions intermédiaires soit mises à la disposition du client pour éviter l'effet tunnel.

Dès lors que toutes les fonctionnalités ont été développées, le service technique met une version dite Bêta test, à la disposition de son service clientèle. De cette manière une grosse phase de test est lancée au niveau de service clientèle. Cela permet de récupérer toute une partie des bugs de la version.

La fin du processus intervient une fois la phase de test validée. Ainsi une version finale est proposée à tous les clients.

III. Partie Technique



1. Planification annuelle

Cette partie va permettre de mettre en avant la manière dont j'ai effectué le travail durant cette année.

Lors de mon arrivée, il a fallu que je m'imprègne du secteur activité de la société en l'occurrence ici la GFT. Pour ce faire, j'ai commencé par explorer les interfaces graphiques de l'application sur laquelle j'ai été affecté.

Pour pouvoir aborder l'infrastructure d'une application, rien ne vaut une mise en situation. Pour cela j'ai été affecté à la réalisation de correctifs applicatifs. Cela m'a permis de prendre connaissance du code source et de m'y familiariser.

Après avoir passé cette période d'adaptation, nous avons pris la décision de mettre en place une évolution majeure sur le produit: « L'intégration d'un objet arbre dynamique ». Cette activité sera longuement détaillée dans la suite de ce document.

Pour continuer dans la partie développement, j'ai travaillé sur l'uniformisation des formulaires des WorkFlows.

Après avoir réalisé ces différents jalons de développement nous avons figé ces modifications pour mettre en place une nouvelle version. Ainsi j'ai pu faire les correctifs de bug, en fonction des retours que j'ai pu avoir de la part des personnes qui ont effectuées les tests de production.

Durant ce temps j'ai commencé à réfléchir à la conception d'objets métiers pour la version V7 du produit.

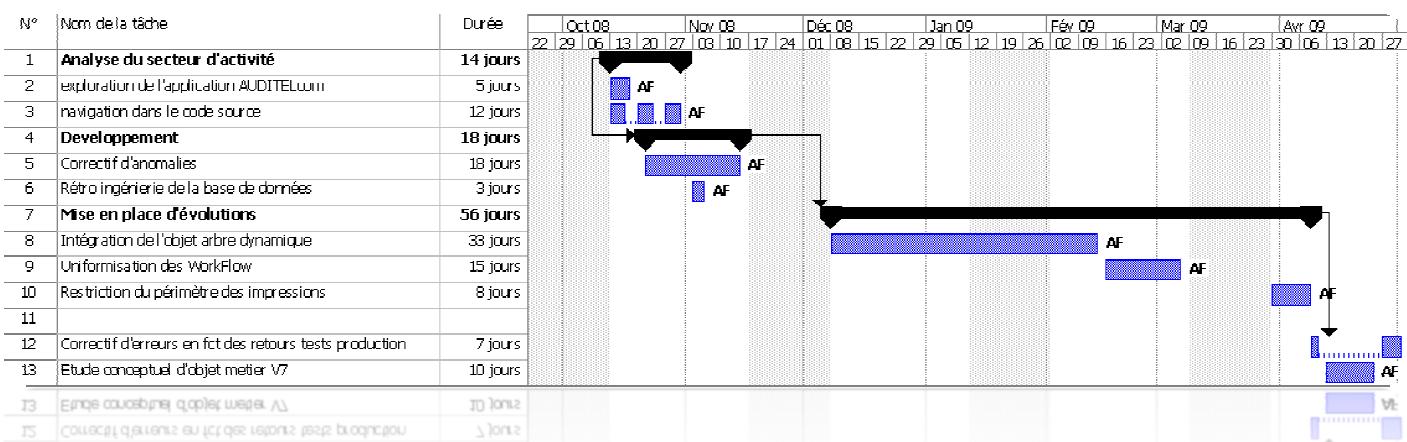


Figure 6 Diagramme de Gant de la planification annuelle

2. Etude de AUDITELcom V6

a. Descriptif

AUDITELcom est une application qui couvre le domaine de la gestion financière des télécommunications. Elle permet de faire du *reporting* sur tout ce qui touche aux ressources communicantes.

Grâce à son architecture, service à la demande ou Software as a Service (SaaS), est adaptable à la plupart des configurations téléphonique utilisées dans les sociétés actuelles. Cette technologie permet de proposer des services par le biais du Web, tous en s'appuyant sur les ressources des clients.

L'application prend en compte les factures des opérateurs, les coûts des communications transitant sur les équipements des sociétés. En s'appuyant sur ces caractéristiques, elle permet de mieux gérer les actifs mobilisés pour la communication (appareil mobile, téléphone fixe,...) ainsi que leurs prestations associées (abonnement, locations, réparations,...).

Enfin il est très facile de mettre en place des indicateurs de performance, tant d'un point de vue technique (qualité de réception, dimensionnement, optimisation), que de ressources actives (prévision financière par centre de frais, suivie,...).

b. Détail technique

Pour bien comprendre le travail qu'il m'a été demandé de faire il est nécessaire de voir un peu plus en détail la structure technique de cette application.

Dans un premier temps il faut prendre en compte que cette application est proposée sous forme de service applicatif hébergé (ASP) à tous ses clients. A contrario des progiciels, qui sont installées chez le client et qui nécessitent des environnements fixes et spécifiques, cette application est accessible directement depuis n'importe qu'elle navigateur Web.

Elle s'appuie sur du code source codé en PHP 5 fortement orienté Objet et du système de base de données MySQL.

Les données communes à tous nos clients tels que les indicateurs de facturations des appels téléphoniques sont regroupées sur des serveurs sous forme de bases mutualisées. Cela permet une très grande réactivité au niveau des évolutions des standards.

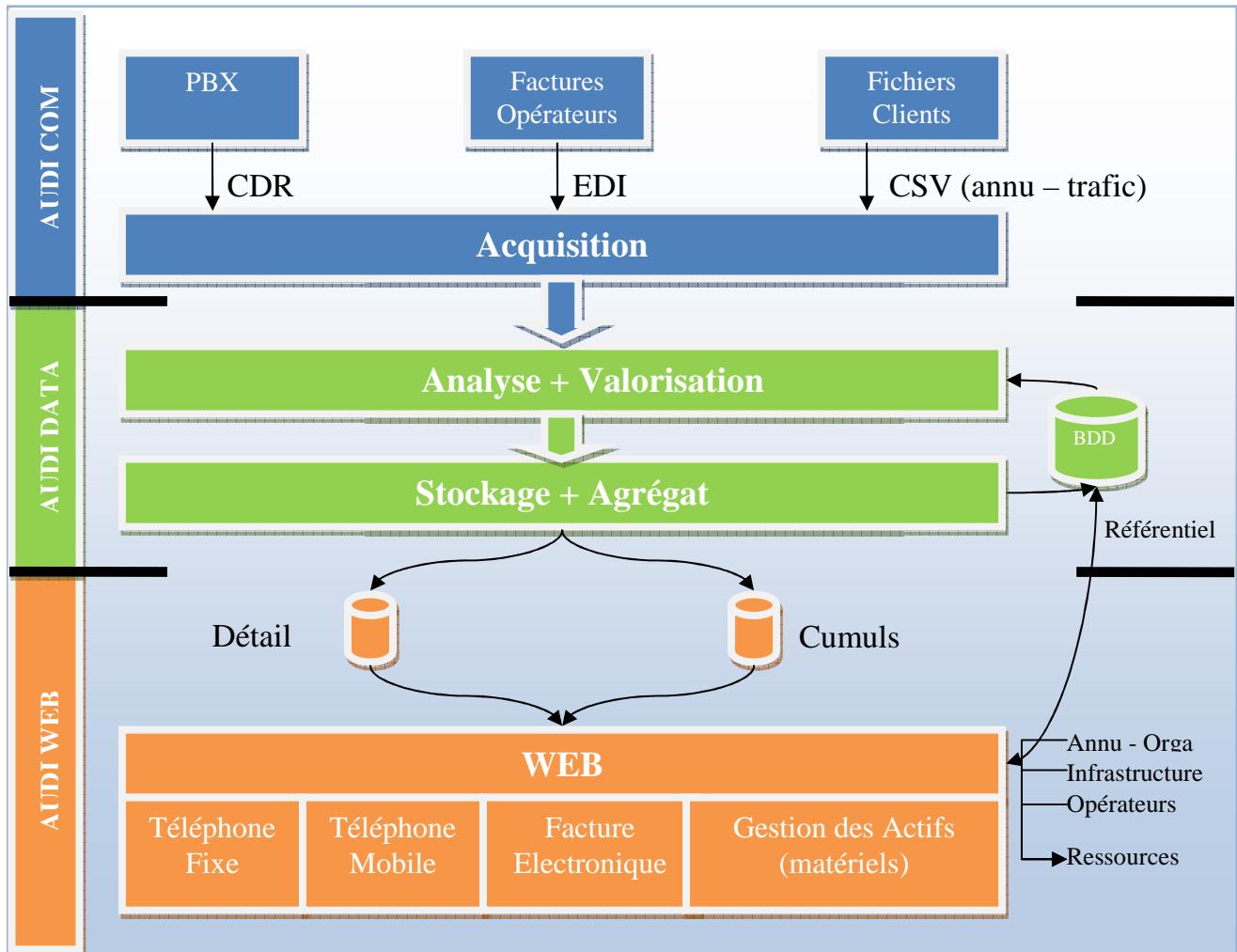


Figure 7 Architecture globale de l'application AUDITELcom V6

Cette application est formée de trois parties physiques distinctes.

1. AUDIcom: serveur de communication permettant de se baser sur les éléments physiques utilisés pour les communications. Ainsi il récupère les informations brutes et les garde à sa disposition.
2. AUDIdata: comme son nom l'indique il permet de récupérer les données envoyées par AUDIcom et de les traiter, à l'aide de table d'agrégation. Une fois qu'elles sont traitées il est nécessaire de les sauvegarder pour pouvoir les réutiliser.
3. AUDIweb: serveur Web, qui permet de mettre en forme les données. Cette partie s'appuie sur la base de données créée précédemment ainsi que des référentiels clients : annuaire, ressources, opérateurs.

c. Analyse de la base de données

Pour pouvoir réaliser des tâches de maintenance ainsi que d'évolution il a été important de prendre en compte la structure de la base de données. Ainsi je vais vous relater ce que j'ai pu remonter lors du travail de rétro ingénierie. Pour cela j'ai utilisé l'outil MySQL Workbench 5.0 OSS, qui réalise des diagrammes entité relationnel étendu.

L'application utilise plusieurs bases de données. Cela permet d'avoir une séparation physique entre les types de données traités. On y retrouve 4 familles de bases de données :

- LAC : contenant toutes les informations du client (Lignes, Annuaire, CUMULS, Ressources, ...),
- WWW : sauvegardant l'environnement des sessions Web,
- OR : regroupant toutes les informations communes à tous les clients. C'est la base Opérateur.
- Et la base ALL: étant la base centralisée de la partie Web. Elle possède les images et le dictionnaire de traduction.

Les bases OR et ALL sont des bases générales. Elles sont communes à tous les clients.

Les tables n'ont pas de clefs étrangères pour une raison historique par rapport à la maintenabilité. Ainsi le travail de rétro ingénierie n'a pas été seulement de positionner les entités mais de voir les différentes dépendances entre les tables.

L'étude s'est concentré sur la partie centrale de l'application, c'est-à-dire la base LAC, celle qui touche les informations propres à un client. Cette étude a permis de faire ressortir des groupes de données distincts, ce qui devrait permettre, par la suite, de réaliser une optimisation en terme d'organisation de données. Ainsi j'ai pu ressortir 6 groupes :

- L'annuaire dans lequel y sont regroupées les informations relatives aux ressources humaines avec leurs structures organisationnelles et infra structurales
- Consommation: tous ce qui est lié au trafic télécom. C'est ici que l'on trouve les informations liées aux appels (facture, durée, coûts,...)
- PX: pour tous ce qui est relatif à l'infrastructure physique réelle.
- Catalogue: comme nous l'avons vu précédemment l'application gère tous ce qui est terminaux mobile. Aussi il est possible de constituer un catalogue contenant des choix de prestations et d'appareils.
- WorkFlow (ou flux de données) : ces données permettent d'enregistrer toutes les actions (modification, insertion) qui ont été réalisées sur l'application.
- Lignes: Ce sont ces informations qui permettent de lier des ressources (utilisateurs) à des terminaux (équipement). Ils font le lien avec les prestations associées.

3. Evolution majeur de l'objet arbre

L'interface de l'application AUDITELcom est formée à partir de nombreux composants génériques. La notion de composant regroupe aussi bien les listes, les tableaux détails, les onglets que les arbres de navigations.

Les arbres sont utilisés pour représenter des données hiérarchiques. Pour l'instant l'application les utilise pour représenter l'organisation et l'infrastructure d'un client.

L'organisation est l'image de la structure organisationnelle d'un client. Elle est composée de plusieurs niveaux que l'on appellera des nœuds de l'arbre. On y trouve :

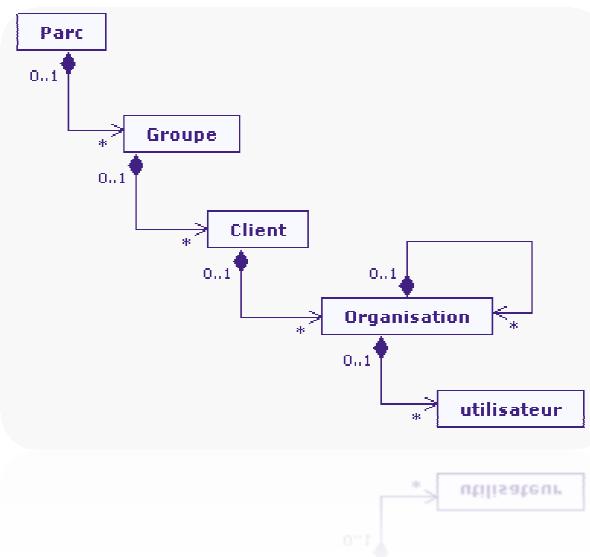


Figure 8 Hiérarchie de l'arbre organisationnel

Comme nous pouvons le voir dans le diagramme ci-dessus, l'arbre peut être infinie car il existe un lien de récursivité entre les nœuds organisations.

L'infrastructure permet d'avoir une idée sur la localisation physique des sites. Elle est divisée en :

- Tous: qui sont le noeud commun à tous les sites.
- Pays: regroupe les sites par pays.
- Sites: emplacement physique caractérisé par une adresse postale.

Avant de se plonger un peu plus dans la structure même de cet objet, il faut savoir que l'application tourne autour de 2 grandes classes que sont:

- TAccessManager qui permet de charger le périmètre d'un utilisateur ainsi que d'initialiser la session. Cette classe est très importante d'un point de vue fonctionnel.
- TDataLink qui regroupe toutes les fonctions en relation avec la base de données

a. Etude de l'existant

L'application utilise la session pour fonctionner. Une session est un objet qui permet de garder des informations tout au long du cycle de vie d'une connexion. Elle enregistre toutes les informations contextuelles : interactions avec le programme et les paramètres du profil de l'utilisateur. Une session est l'exécution du programme pour un utilisateur donnée.

Il existe des outils tels que les sessions PHP ou Apache. Mais ils ne permettent pas de garantir la pérennité des informations lorsque l'on effectue une répartition de charge sur plusieurs serveurs Web. C'est pour cela que nous avons décidé de gérer ces informations en interne et ainsi de les enregistrer en base.

La session contient tous les objets actifs de l'application. On y trouve des éléments Web pour l'affichage, ou métier pour les données. Ainsi lors de la prise en main de l'application, la session contenait toutes les informations relatives à l'annuaire avec son affichage.

L'impression d'écran suivant, permet de positionner l'objet arbre dans son contexte.

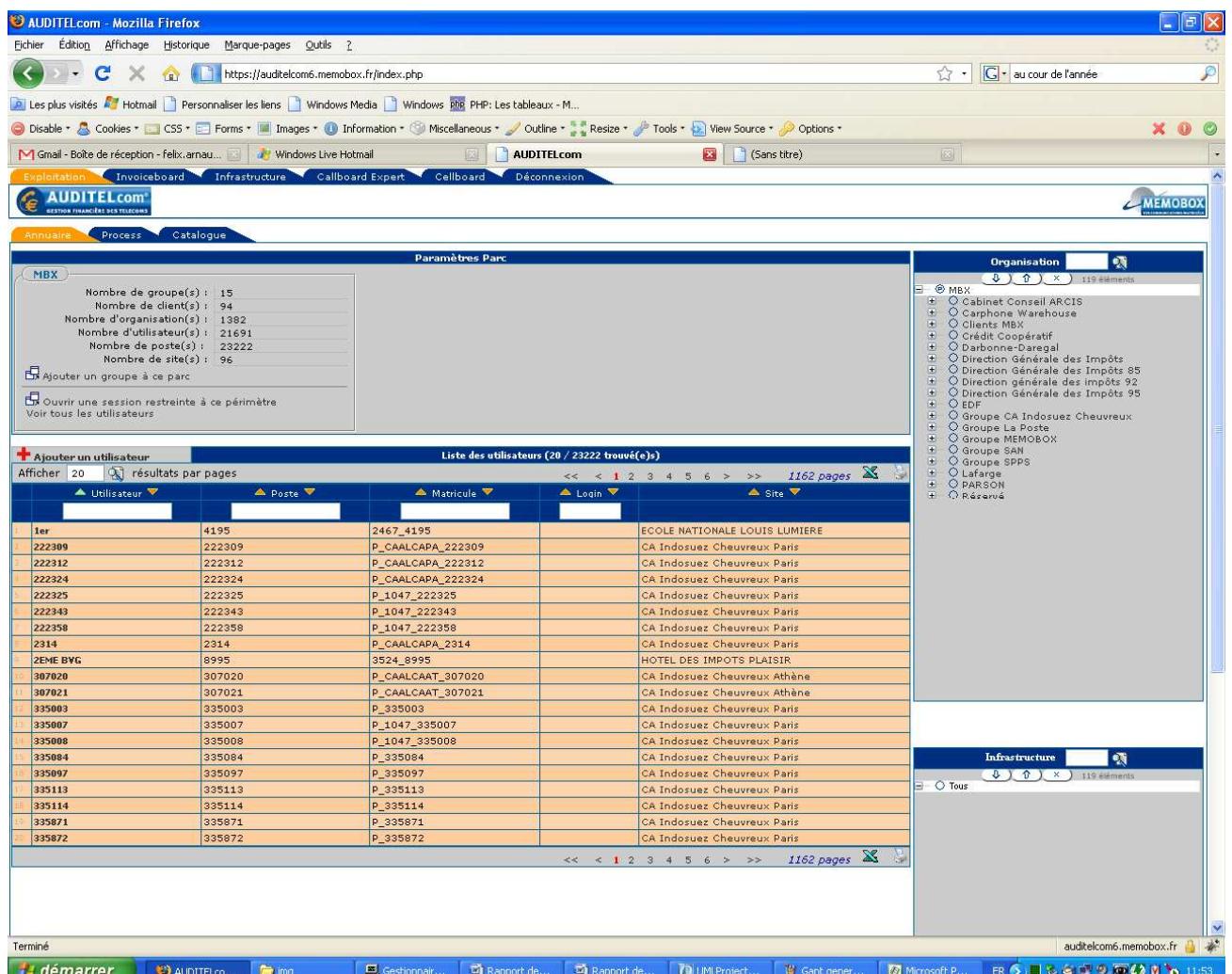


Figure 9 Impression écran de l'ancien arbre dans l'application

L'initialisation de l'arbre s'effectue durant la création de la session de l'utilisateur. On y retrouve des grandes phases tel que:

- le chargement du profil de l'utilisateur : pour cela l'application va lire le profil XML stocké en base et initialise un tableau PHP ('user_access_table') qui sera enregistré en session.
- La création du périmètre de l'arbre. C'est dans cette phase que les objets arbre sont créés.
- La construction de tous les objets Web pour l'affichage

Dans ce chapitre nous allons descendre dans l'ancien objet arbre matérialisé par la classe TWebTree.

Niveau statique

Nous allons voir comment était conçu l'objet arbre. Il a été conçu de la même manière que tous les objets graphiques de l'application. C'est-à-dire qu'ils héritent tous de la classe abstraite **TWebObjet**.

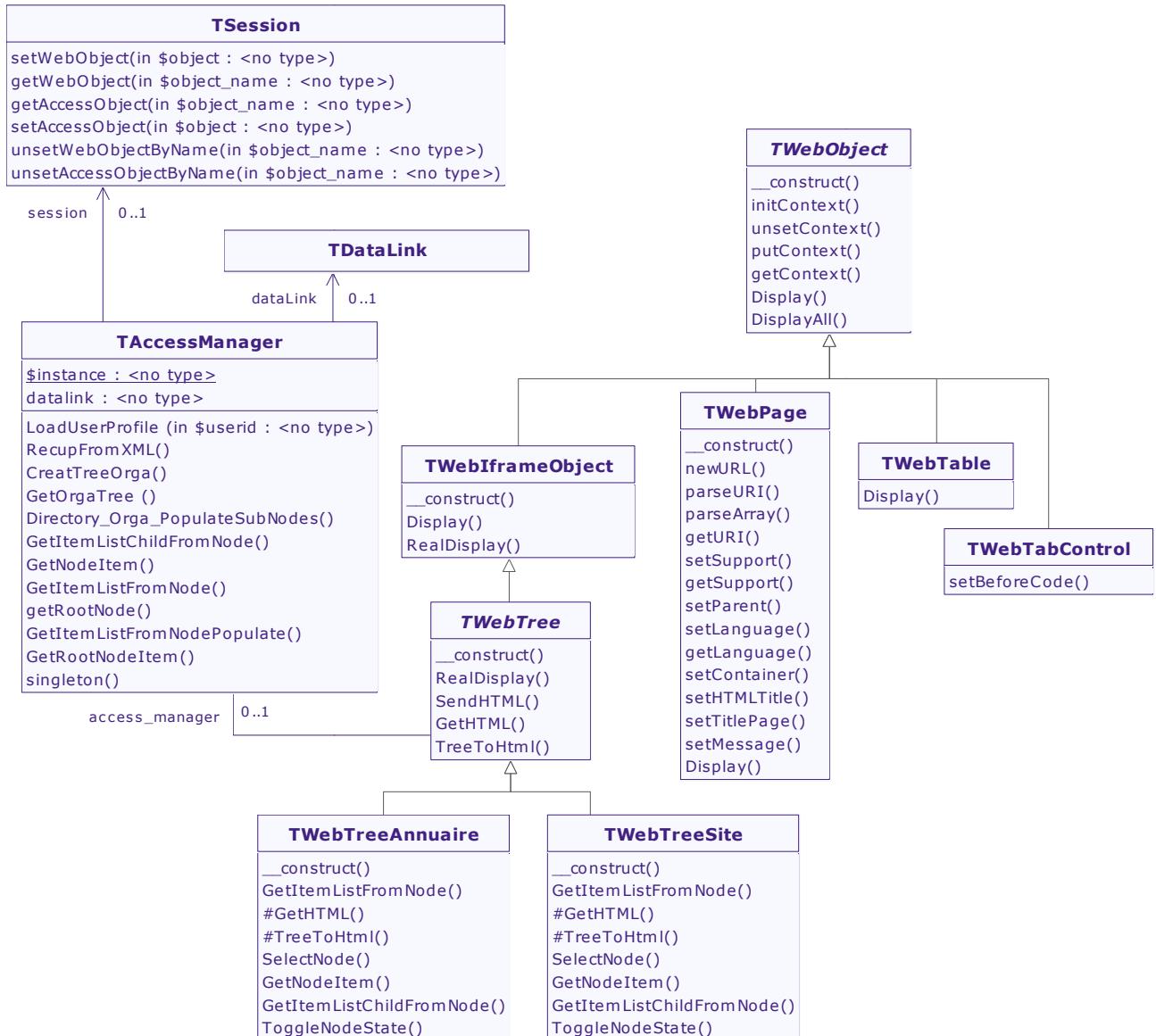


Figure 10 Diagramme de classe de l'ancien arbre

Comme nous pouvons le voir sur le diagramme ci-dessus il est possible d'encapsuler des classes de type Web. Cela permet de paramétrer l'aspect d'affichage que l'on souhaite donnée à notre objet. Auparavant l'objet arbre était encapsulé dans une iframe par le biais de la classe **TWebIFrameObjet**.

Une iframe est en quelque sorte une fenêtre dans une fenêtre. C'est une zone indépendante d'une page Web. Pour l'application cela permet d'interagir avec l'objet arbre situé à l'intérieur de cette iframe sans recharger le contenu global de la page.

Pour construire les arbres il faut aller voir la classe TAccessManager qui contient toutes les fonctions gérant le contexte. C'est lors du processus de création de la session du début de connexion à l'application par un utilisateur, que la fonction « CreatTree » est appelée pour chaque arbre (organisationnel et infrastructurel). Cette fonction charge la totalité du contenu des arbres dans des tableaux PHP.

Après avoir chargé le contexte, on récupère les objets graphiques qui le composent. Une fois qu'ils sont en session, il suffit de faire appel à la fonction « display » de l'objet TWebPage pour lancer la création des pages HTML côté serveur. Cette fonction permet de faire un lancement en cascade des fonctions Display des objets graphiques qui ont été mis en session. Cette étape permet de créer la totalité des pages Web du côté du serveur pour le client.

❖ Avantages et inconvénients

Grâce à cette étude de l'existant, on peut voir que l'ancien arbre est assez lourd d'un point de vue conceptuel. Le fait de charger tous les éléments d'un annuaire en session pose des problèmes de performances quand au démarrage de l'application. Surtout que les utilisateurs ont rarement besoin de voir toutes les branches de l'arbre.

Cette architecture pose la question de la légitimité de la classe TWebTree qui ne fait que prendre des données pour les retranscrire sous forme HTML. Dans ce cas, nous retrouvons le cœur du traitement des arbres dans la classe TAccessManager, qui je le rappel est dédié à cadrer l'accès à certains objets en fonction du contexte utilisateur.

En revanche cette méthode, de mise en session d'informations, permettait d'avoir accès à la totalité des éléments de l'arbre, à partir d'une navigation dans un tableau PHP.

b. Travail réalisé

Suite au constat réalisé dans le paragraphe précédent, nous avons décidé de s'occuper de cet objet arbre.

Un arbre se définit par la notion d'un ensemble de nœuds eux-mêmes composés d'items.

Nous avons créé un cahier des charges, ce qui nous a permis de convenir des modalités générales que le nouvel objet devrait posséder.

Il a fallu s'intéresser à l'utilisation qu'en font les clients. Il est nécessaire de mettre en place un système qui permet de mieux répondre au mode opératoire des utilisateurs. Nous nous sommes penchés sur les questions de performance. En effet, le but est de diminuer considérablement le temps de réponse de l'application lors de son ouverture. Dans ce même ordre d'idées, il faut réduire massivement la taille prise par l'objet dans la session.

Et comme dans toute évolution, il faut pouvoir garantir la non régression des fonctionnalités existantes.

En dehors de l'arbre lui-même on retrouve, des outils de manipulations regroupés dans une barre de menu ainsi que de la traduction des champs textes.

Pour la barre de menu, il est composé :

- D'une zone de saisie.
- D'un bouton de validation de recherche.
- Et de deux autres boutons pour réduire tous les nœuds déroulés de l'arbre et pour désélectionner le nœud courant.

Dans la suite de ce document je vais décrire les choix et les méthodes utilisées pour pouvoir effectuer ce projet.

▣ Technologies Utilisées

Pour parvenir à ce résultat nous sommes partis sur la technologie AJAX (*XML et Javascript asynchrones*). Celle-ci permet au client de garder la main, tout en effectuant des traitements côté serveur. Ce qui est approprié dans notre cas, car nous souhaitons mettre à jour le contenu de l'arbre, en réponse d'une demande d'ouverture d'un nœud. Tout en ne rafraîchissant pas toute la page. Cette technologie est dérivée des balises Iframe avec l'avantage qu'il est possible de lancer des requêtes du côté du serveur et d'en récupérer les résultats.

Pour l'échange d'informations entre le serveur et le client nous décidons d'utiliser, non pas l'XML mais le JSON (*JavaScript Object Notation*). C'est un format de données générique utilisant la notation d'objet Javascript. Il est donc plus rapide à implémenter et à utiliser avec ce langage.

Pour simplifier le développement côté client, en Javascript, nous avons utilisé la librairie Jjavascript Mootools. Elle contient de nombreuses fonctions de manipulations et de création d'objet d'API Dom. Elle permet de rajouter facilement des effets visuels. Elle a l'avantage d'être une des librairies les plus rapides et des plus légères de sa catégorie.

▣ Partie Technique

Dans ce chapitre je vais aborder la partie technique de l'arbre. On y retrouvera les choix de conception que j'ai adopté et l'explication du processus de déroulement d'un nœud.

Le nouvel objet arbre sera donc composé de deux parties totalement complémentaires. L'objet possède deux centres de traitement.

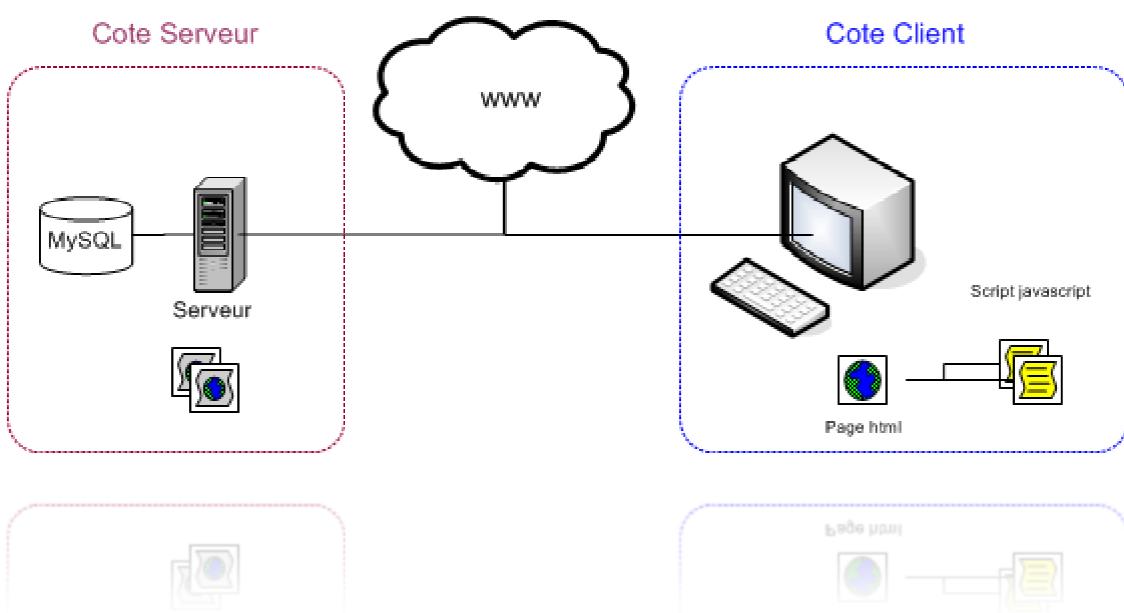


Figure 11 Architecture physique de l'application AUDITELcom

La première du côté client, qui est représenté par du code Javascript. Cela permet de manipuler des parties de pages HTML directement sur le navigateur Web. C'est un fichier script qui est lancé sur la machine hôte de l'utilisateur.

La seconde du côté du serveur de données sur lequel on effectue du traitement en lançant des requêtes sur le serveur de base de données. Ce sont des scripts codés en PHP.

Côté client

Comme nous venons de le voir la partie cliente est représentée par du code Javascript. Concernant notre application nous l'avons scindé en deux classes:

- AjaxManager qui s'occupe de toute la partie qui concerne l'envoie de requêtes au serveur.

- TWebTree qui comprend le cœur du traitement associé à l'objet arbre d'un point de vue mis en forme, qui regroupe toutes les fonctions pour fabriquer et faire interagir l'arbre avec l'utilisateur.

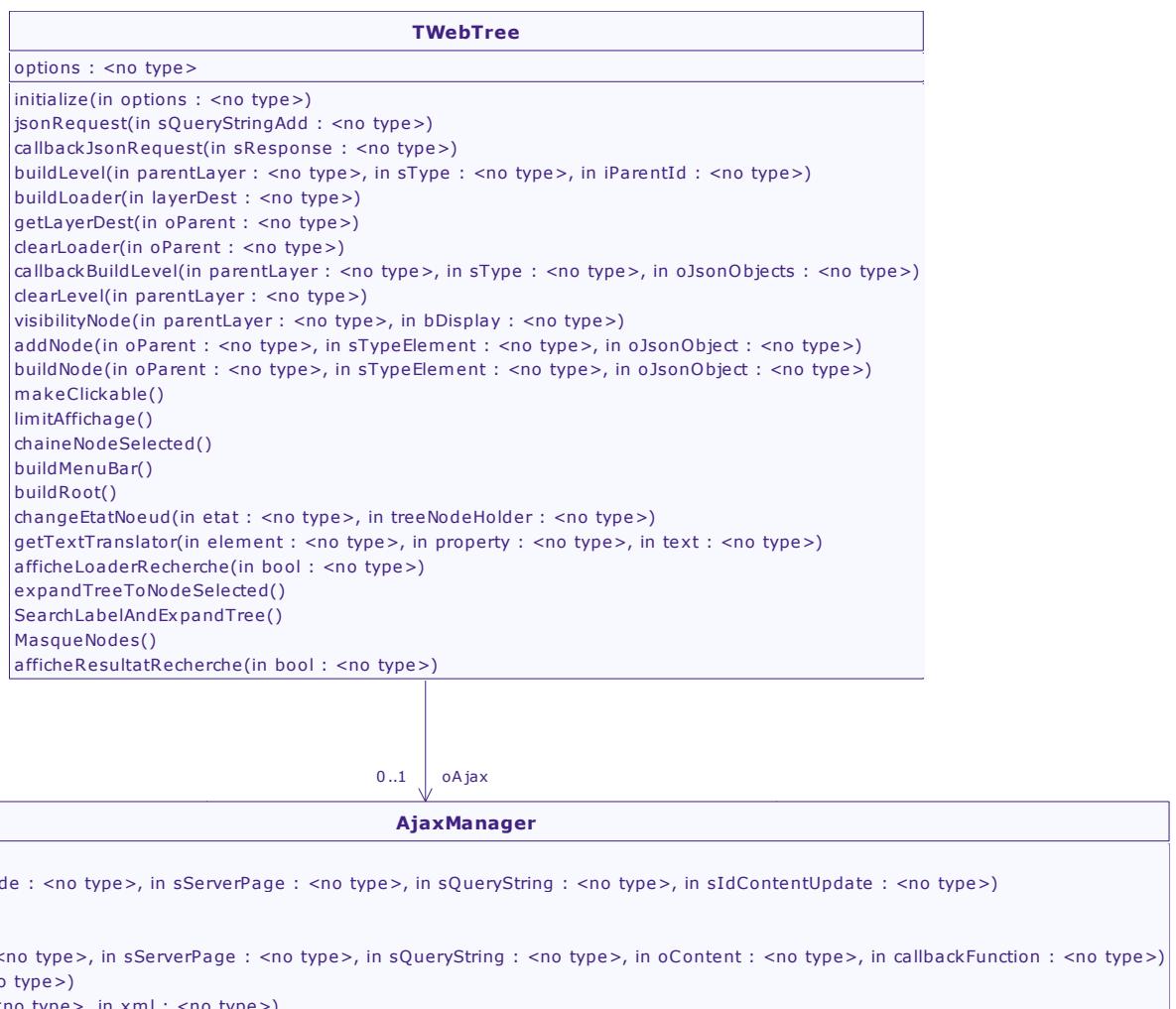


Figure 12 Diagramme de classe script Javascript

On peut remarquer que les deux classes possèdent la fonction initialize. Elle représente le constructeur de la classe en Javascript. C'est à partir de celle-ci que sont initialisées les variables de la classe à partir des paramètres passés au constructeur.

AjaxManager

Cette classe a été mise en place pour permettre de formater les requêtes à envoyer au serveur. Comme nous pouvons le voir sur le diagramme de classe précédent, elle est séparée en 2 groupes.

Le premier qui permet de prendre les requêtes et de les encapsuler dans une pile First In First Out (FIFO). Il regroupe les fonctions setRequestQueue, popRequest et cancelRequest.

Le second qui s'occupe de tout ce qui concerne la partie dialogue avec le serveur. Avec la fonction ajaxRequest qui permet d'envoyer les requêtes, et les fonctions ajaxFailure et ajaxComplete qui sont appelées en fonction de la réponse retournée par le serveur.

TWebTree

Le déroulement de ce script se réalise suivant deux phases:

L'initialisation de la Racine: elle s'effectue lors de l'appel du constructeur. En effet après avoir initialisé les variables globales de la classe, on fait appel à la fonction « chaineNodeSelected » qui permet de récupérer le nœud sélectionné ainsi qu'une chaîne contenant le chemin à parcourir pour y accéder.

La construction de l'arbre: on fait appel à la fonction « buildRoot » qui permet de créer le nœud racine, et de lancer le processus de construction. Nous allons détailler cette phase dans la partie qui suit.

Processus de construction d'un nœud

La fonction de lancement est ‘buildLevel’. Cette fonction prend en paramètre le nœud courant à développer, le type du nœud et son identifiant. Elle fait appel à la fonction ‘jsonRequest’ avec ces paramètres.

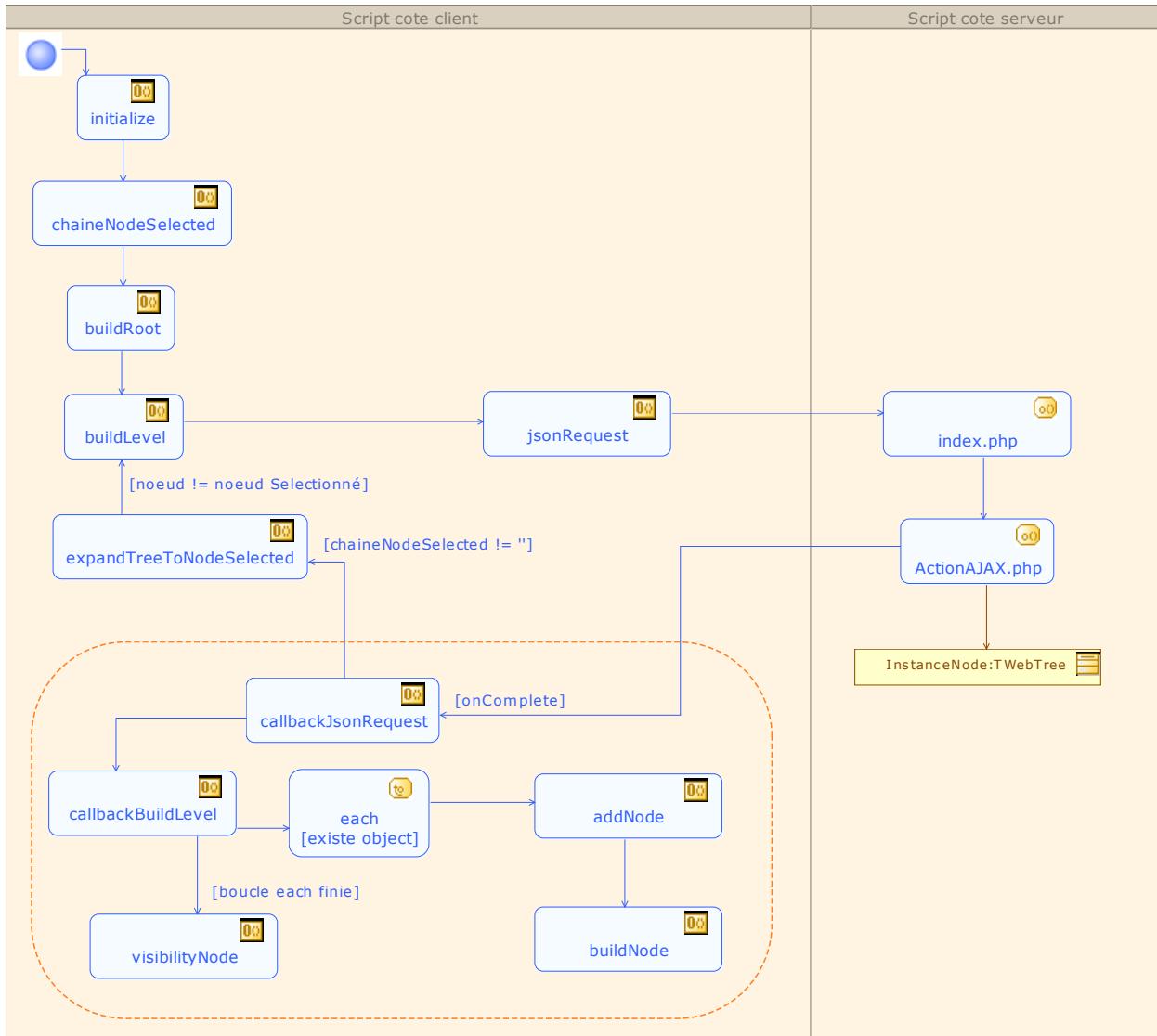


Figure 13 Diagramme d'activité de l'instanciation de TWebTree Javascript

Cette fonction permet de lancer la requête côté serveur et de spécifier que la fonction ‘callbackJsonRequest’ sera la fonction à appeler lorsque la réponse sera disponible; on l'appelle fonction de retour (ou callback).

‘callbackJsonRequest’ fait quant à elle appel à la fonction ‘callbackBuildLevel’. Puis elle regarde si la chaîne contenant le chemin du nœud sélectionné est vide ce qui signifiera qu'il n'est pas nécessaire de construire d'autre nœud. En revanche, si elle comporte des éléments il faut faire appel à la fonction ‘expandTreeToNodeSelected’, ce qui permet de faire une boucle récursif sur tous les nœuds de cette chaîne.

callbackBuildLevel’ récupère tous les objets JSON retournés par le serveur et appelle la fonction ‘addNode’ pour construire chaque nœud.
Une fois tous les nœuds fils créés il rend le nœud courant visible.

addNode fait appel à la fonction buildNode et récupère le nœud créé et l'insère dans le tableau des nœuds de l'arbre.

buildNode construit les « div » représentatifs d'un nœud.

Côté Serveur

Cette partie regroupe les traitements effectués au niveau du serveur.



Figure 14 Diagramme de Classe de l'arbre AJAX

On retrouve bien les classes TAccessManager et TDataLink. En revanche on peut voir qu'elles ont été recentrées sur leurs domaines de compétences. TAccessManager ne fait que construire les bornes des arbres (nœud Racine, limite d'affichage), en fonction du périmètre de l'utilisateur. Et TDataLink permet d'aller chercher les informations dans la base en les transformant en nœud via la classe TTreenode.

Le cœur de la gestion des arbres est redescendu au niveau de l'objet TWebTree. Cette classe regroupe toutes les fonctions nécessaires à la navigation de l'arbre.

La classe TWebTree permet:

- la gestion du périmètre en récupérant le nœud Root et les limites d'affichage.
- la sélection d'un nœud, via getPathselectedNode qui permet de récupérer tous les nœuds à étendre pour arriver jusqu'au nœud sélectionné elle fait appel à la fonction revenirDunNiveau.
- son affichage au format HTML, en appelant GetHTML fait une insertion dans la page de l'appel de la partie Javascript.
- sa navigation matérialisé par les fonctions getNode et getNodeListChildFromNode permettant d'avoir la liste des nœuds fils
- Et enfin la recherche de nœud dans l'annuaire.

Comme nous pouvons le voir les classes de spécifications TWebTreeAnnuaire et TWebTreeSite ne sont composées que de 3 variables locales:

- typesNodes : tableau permettant de faire la transition entre les types des nœuds définis dans la DTD des données XML (parc, groupe, société,...) contenues dans le profil de l'utilisateur avec les constantes du programme.
- fct_sql: tableau regroupant le nom de toutes les fonctions SQL qui est possible d'appeler via la classe TDataLink. Ce tableau est ordonné en fonction des actions que l'on souhaite réaliser sur l'arbre : récupérer les nœuds fils, parents, et faire une recherche d'éléments en correspondance à un libellé.
- Img_picto: tableau contenant la liste des chemins d'accès aux images miniatures à associer aux nœuds lors de l'affichage.

On remarque que nous utilisons de manière poussée le formalisme de l'héritage. Le cœur du traitement est réalisé dans la classe généraliste et le paramétrage est effectué dans les classes de spécifications.

De telle manière il serait très simple de rajouter un nouvel arbre qui mettrait en forme des données hiérarchique. Il suffirait de coder les fonctions SQL associées.

c. Les Tests

Pour permettre de dire que le nouvel arbre est stable et qu'il répond bien aux attentes, il a fallu faire un listing de celles-ci. Ainsi nous avons élaboré une liste d'un certain nombre de fonctionnalités nominales que l'arbre doit posséder.

Le Tableau suivant donne les règles à respecter.

| | |
|--|---|
|   | prendre en compte le périmètre du profil connecté : <ul style="list-style-type: none"> • doit se positionner sur le nœud racine du profil • il ne doit pas dérouler les nœuds qui sont en dessous de la limite d'affichage |
|  | expansé des nœuds |
|  | Masquer tous les nœuds fils de la racine |
|  | Faire une recherche sur le contenu de l'arbre (avec des requêtes appropriées) |
|  | Afficher le résultat de la recherche |
|   | Sélectionner un nœud de l'arbre : <ul style="list-style-type: none"> • modification du contexte • déroulement de l'arbre jusqu'à l'élément sélectionné |
|  | Désélectionné un nœud, rendre le nœud racine sélectionné |
|   | Faire fonctionner 2 arbres en parallèle <ul style="list-style-type: none"> • La navigation • La recherche |
|  | Sélectionné un nœud, et ouvrir une sous session de ce nœud : <ul style="list-style-type: none"> • Le nœud racine doit être l'élément sélectionné • Récupérer la nouvelle limite d'affichage |

Figure 15 Liste des fonctionnalités testées

Cette liste m'a permis de faire mes tests unitaires.

Cette application à l'avantage d'être centralisé. Cela signifie que l'environnement de développement est presque identique à l'environnement de production, à l'exception de la puissance du serveur. Ainsi pour faire des tests de production il m'a suffit de me connecter sur la base de données de notre plus gros client. Cela s'effectue en modifiant la chaîne de connexion du fichier de configuration.

d. Les Résultats

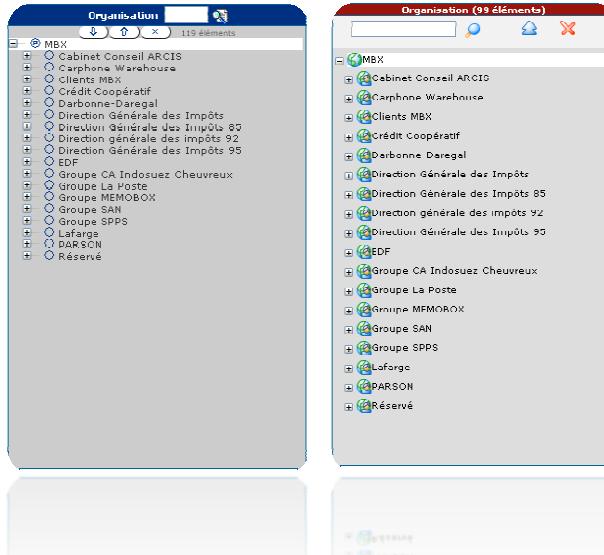


Figure 16 Evolution graphique de l'arbre

Pour que vous puissiez avoir une idée du travail que j'ai réalisé, je vais illustrer les résultats obtenus à l'aide d'impressions écran.

Grâce à l'impression écran ci-dessus, on peut remarquer que l'arbre n'a pas seulement été modifié au niveau de sa conception, mais aussi au niveau de son aspect graphique. Nous y avons ajouté des images sous forme de pictogramme en fonction du niveau du nœud auquel on se situe.

De plus la barre de menu associée a été remise au goût du jour. On y retrouve bien les éléments qui permettent d'actionner l'arbre tel que la réduction de tous les nœuds de l'arbre ainsi que la désélection du nœud courant.



Figure 17 Déroulement d'un nœud

Dorénavant le déroulement d'un nœud se fait par requête asynchrone, ainsi durant le processus de chargement, c'est-à-dire le lancement de la requête et mise en forme côte client, un message indique à l'utilisateur que le nœud est en train de se charger.

**Figure 18 Recherche dans l'arbre**

La recherche d'information dans l'arbre a été modifiée. Auparavant les résultats de la recherche étaient identifiés à l'aide d'une mise en couleur des nœuds résultants. Or dans le nouvel arbre, une zone de résultats apparaît avec la liste des éléments retournés par le serveur.

Dès que l'on clique sur un des éléments de cette liste l'application est obligée d'actualiser toute la page car on modifie le contexte. Ainsi l'arbre se déroule automatiquement jusqu'au nœud sélectionné.

4. Modification des WorkFlow

Un WorkFlow est un ensemble de flux d'information pour réaliser une action. Dans l'application on utilise ce procédé pour effectuer deux types de requête:

- La demande d'actif mobile et d'abonnement associé. Cette requête est formée d'une suite chronologie d'événements passant par des acteurs différents.
- La modification des informations de la base à partir du web.

Le problème était que ces 2 types furent développés à des intervalles de temps différents ce qui a engendré une conception disjointe.

a. Etude de l'existant

Les WorkFlows représentent une partie importante de la partie interface d'une application. Ils permettent de créer des points d'entrer sur les informations stockées en base de données. Il est ainsi important de stocker l'historique des modifications réalisées pour pouvoir en cas de problème, remonter un journal des interventions des utilisateurs.

Le processus de sauvegarde de toutes modifications est accentué, du fait que l'application est de type prestation de service et donc par définition accessible par un grand nombre d'utilisateurs.

En ce qui concerne l'application AUDITELcom, la gestion des WorkFlow se retrouve à deux niveaux.

Il y a toute la partie concernant le paramétrage et la définition des éléments qui interagissent avec les WorkFlows

Et il y a l'utilisation qu'en fait le programme. Pour cela il faut s'appuyer sur les diagrammes de classes de l'application.

Ainsi dans le chapitre qui suit je vais vous expliquer un peu plus en détail ces deux parties.

■ Les WorkFlows en table

La définition de tous les workFlow qu'il est possible de réaliser sur l'application sont regroupés dans une base qui est commune pour tous les clients: base ALL_WWW. Elle regroupe aussi les traductions et les images.

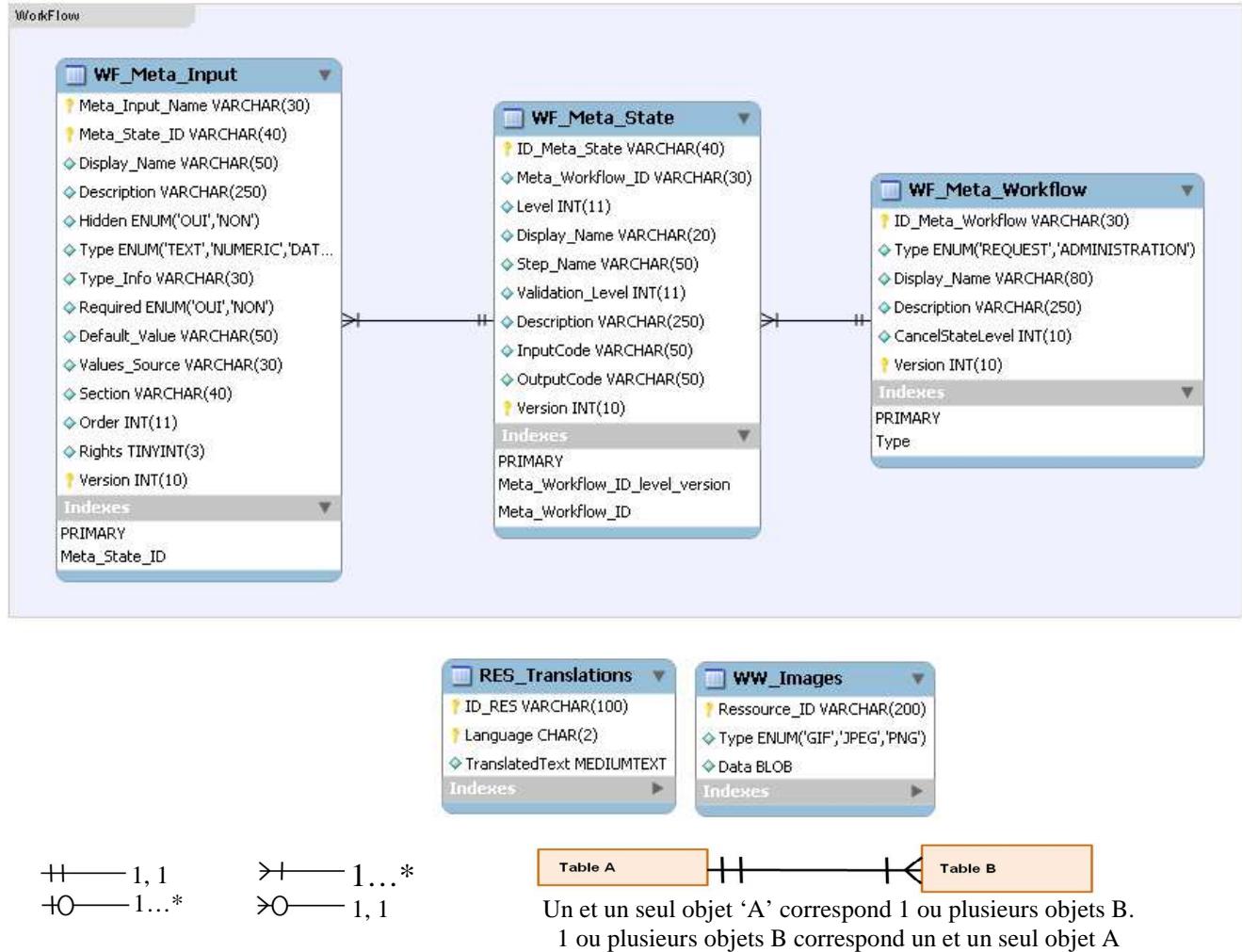


Figure 19 Base ALL_WWW

Un WorkFlow est découpé selon trois tables : Meta_WorkFlow, Meta_State, Meta_Input.

1. La table Meta_WorkFlow permet de définir un workFlow dans son ensemble. Il précise son nom, son type (Request ou d'administration) et le niveau pour lequel il sera considéré dans un état échappé.

2. La table Meta_State permet d'avoir la liste des étapes qui composent un workFlow. Chaque étape est matérialisé par un niveau, un nom (d'affichage, de niveau) et d'une description. En plus ces champs nous avons deux colonnes particulières: InputCode et OutputCode. Elles permettent de faire le lien entre les informations en base et les champs de l'interface. InputCode sera appelée pour initialiser les champs de l'interface à partir des valeurs de la base. Et la fonction sera appelée lors de la validation du formulaire pour pouvoir enregistrer les informations saisies au niveau de l'interface, du côté de la base de données.
3. La table Meta_Input, qui permet de regrouper tous les champs qui seront renseignés pour une étape d'un workFlow. Il est possible de déterminer son nom, son type, si c'est un champ caché, s'il est obligatoire lors de la saisie, son ordre d'affichage, sa zone (qui permet de regrouper des champs entre eux), sa valeur par défaut, et la source de la valeur. Ce dernier permet d'aller chercher une liste de valeurs dans la base. Cela permet de construire des listes déroulantes à partir d'informations extraites de la base de données.

Pour pouvoir contrôler l'accessibilité des informations, au niveau de leurs modifications, il a été mis en place des droits d'accès. Ils permettent de définir les différentes actions qu'un utilisateur a le droit de réaliser. Ces droits sont définis dans la table des profils des utilisateurs.

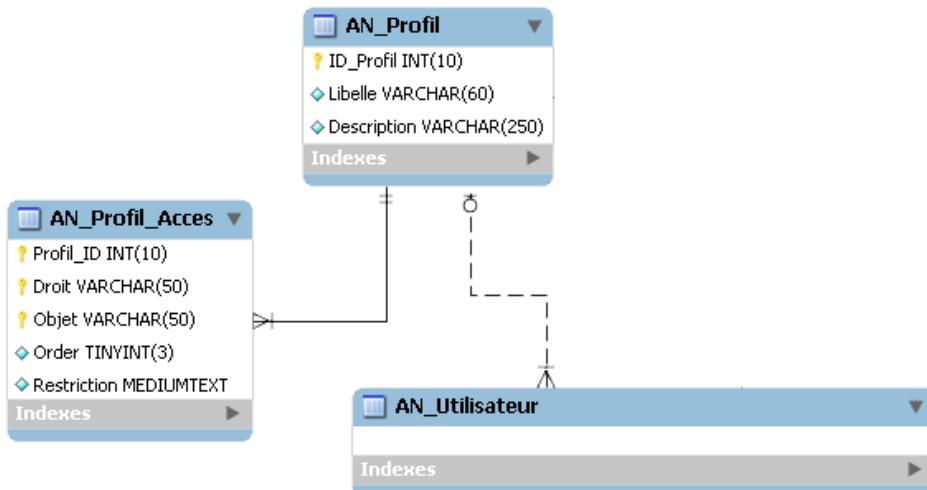


Figure 20 Table pour la gestion des profils

Un profil permet de définir:

- les accès aux différents modules de l'interface graphique
- les accès aux différents workFlow. On met le nom des meta_state dans la colonne objet.

La création d'un workFlow se réalise en créant un ticket.

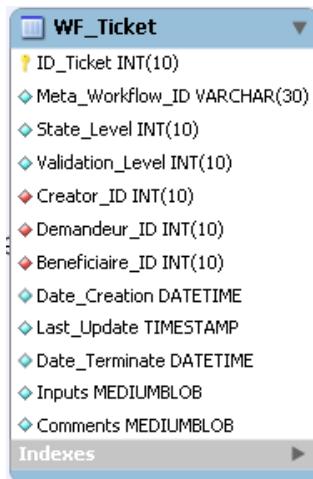


Figure 21 Table des WF_Ticket

Un ticket permet de stocker toutes les informations qui découlent des actions réalisées. Il regroupe les informations essentielles telles que le nom du workflow qui permet de savoir de quelle action il s'agit, l'identifiant du créateur, demandeur et bénéficiaire, ce qui aide à remonter vers les responsables de ce ticket. Il possède les informations sur les dates de création, modification, de clôture.

Puis on y trouve les inputs. Ce champ contient toute la définition du workflow sous forme d'objet : les définitions des états (States) avec leurs données associés (inputs).

b. Travaille réalisé

Le travail qu'il m'a été demandé concernant les WorkFlow est de n'avoir qu'une seul méthode de gestion de WorkFlow. Car comme nous l'avons vu dans l'introduction de ce chapitre, il existe 2 types de gestion :

- Les WF de Requête qui permettent de gérer des demandes de plus de cinq étapes
- Et les WF d'administration, qui sont composés d'une seule étape

Pour réaliser cette action je me suis appuyé sur les mécanismes existants. Je suis parti du moteur le plus évolué qui est la gestion des WF de Requête.

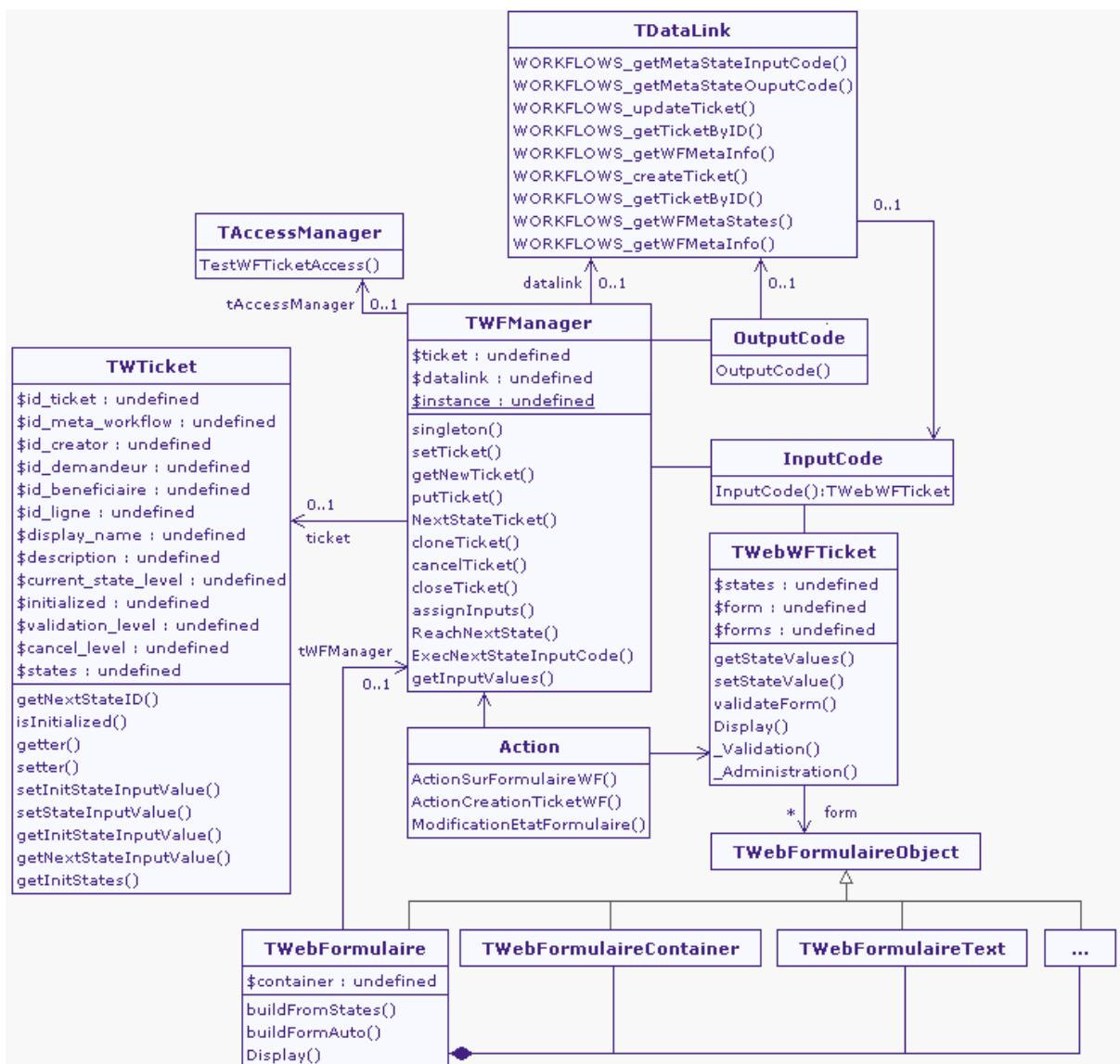


Figure 22 Diagramme de classe des WorkFlows

Auparavant les WF d'administration étaient totalement gérés dans les actions. Dorénavant on utilise la classe TWebWFTicket comme base pour gérer les formulaires des interfaces des WorkFlows.

Un workflow est contrôlé via le moteur TWFManager. Cette classe regroupe toutes les fonctions de manipulation d'un ticket.

Ainsi elle permet de créer, cloner et enregistrer un ticket.

C'est à partir de celle-ci qu'il est possible d'enregistrer les informations de l'interface dans les champs de différentes étapes du workFlow.

On y retrouve les fonctions permettant d'exécuter les fonctions de liaison entre la base et les champs à afficher.

Elle est donc en forte relation avec la classe InputCode et OutputCode. Les fonctions qui composent ces classes sont structurées suivant le même moule. Ils prennent en paramètre les identifiants des informations que l'on souhaite afficher, et assigne chaque champs de l'étape (State) du workFlow avec les champs contenue en base.

Une fois que les champs sont renseignés, il ne reste plus qu'à les mettre en forme pour cela on fait appel à la classe TWebFormulaire. Cette classe permet de construire des formulaires. Elle a juste besoin d'avoir les caractéristiques des champs à afficher que nous possédons au niveau des Meta_Inputs.

Enfin on récupère les champs des formulaires qui ont été remplis dans la classe Action. Cette classe permet de repérer le type d'action qui a été demandé et de lancer les différentes actions en correspondance. On y regroupe les types d'actions suivantes :

- on crée un ticket,
- on enregistre le workFlow: pour cela on récupère les valeurs rentrées et on fait appel à la classe TWFManager pour enregistrer le ticket,
- on passe à l'étape suivante. On valide les informations saisies, on les récupère pour les enregistrer. Et on lance la fonction de création du formulaire de la nouvelle étape.
- On annule le workFlow: on enregistre le ticket courant avec l'état « cancel ».

5. Analyse d'objets métiers V7

AUDITELcom V7 est un projet qui implique des modifications majeures du produit. Cette version a pour objectif d'être orienté composants. Le souhait est de renforcer l'imperméabilité des objets. Le but est d'avoir des composants spécifiques pour chaque domaine technique auxquels ils sont affectés. Ainsi il serait facile de répartir la charge de travail en fonction des qualités de chaque membre de l'équipe de R&D.

Ce projet a été lancé dans le but de trouver un point de convergence entre les différents produits de la société.

Pour arriver à ces fins la V7 va demander un certain de nombre de réunions de conception entre les différentes équipes de développement.

Le but est de construire des composants qui pourront être réutilisés pour les autres produits de la société. Il est facile à comprendre qu'un composant commun à différents produits est plus facile à maintenir et à faire évoluer. La grande difficulté qui réside ici est le fait que les produits sur lesquelles les composants vont être implantés peuvent être hétérogènes.

Ce critère représente une contrainte forte au niveau de la conception. Pour pouvoir satisfaire cette contrainte il est nécessaire de trouver le juste milieu entre le totalement générique et le facilement utilisable. En cela qu'il ne faut pas tomber dans l'excès pour ne pas avoir au final des composants tellement opaques que personne ne puisse les utiliser.

De plus il est important de bien marquer la frontière entre chaque couche de l'application. Ainsi l'interface serait totalement dissociée de la partie base de données et elle-même dissociée de la partie métier. On se retrouverait avec une architecture trois tiers.

Grâce à ce projet il n'est pas exclut de produire une API à partir des composants métier propre à la GFT, pour la vendre aux clients souhaitant construire leurs propres interfaces graphiques.

a. Etude de l'existant

Comme nous avons pu le voir lors de l'analyse de l'existant de l'objet arbre, la classe TAccesManager était partiellement dérivée de son centre de compétence.

Ainsi les règles entre les objets n'étaient pas très claires ce qui impliquait que les développeurs passaient outre et ils créaient des liens transversaux.

De plus la classe TDataLink regroupe toutes les fonctions d'interrogations sur la base de données. Le problème de cette classe, elle est submergée de fonctions et il est difficile de s'y retrouver. De plus toutes les fonctions sont spécialisées pour le système de base de données MySQL. Ce qui rend la migration difficile à ce jour.

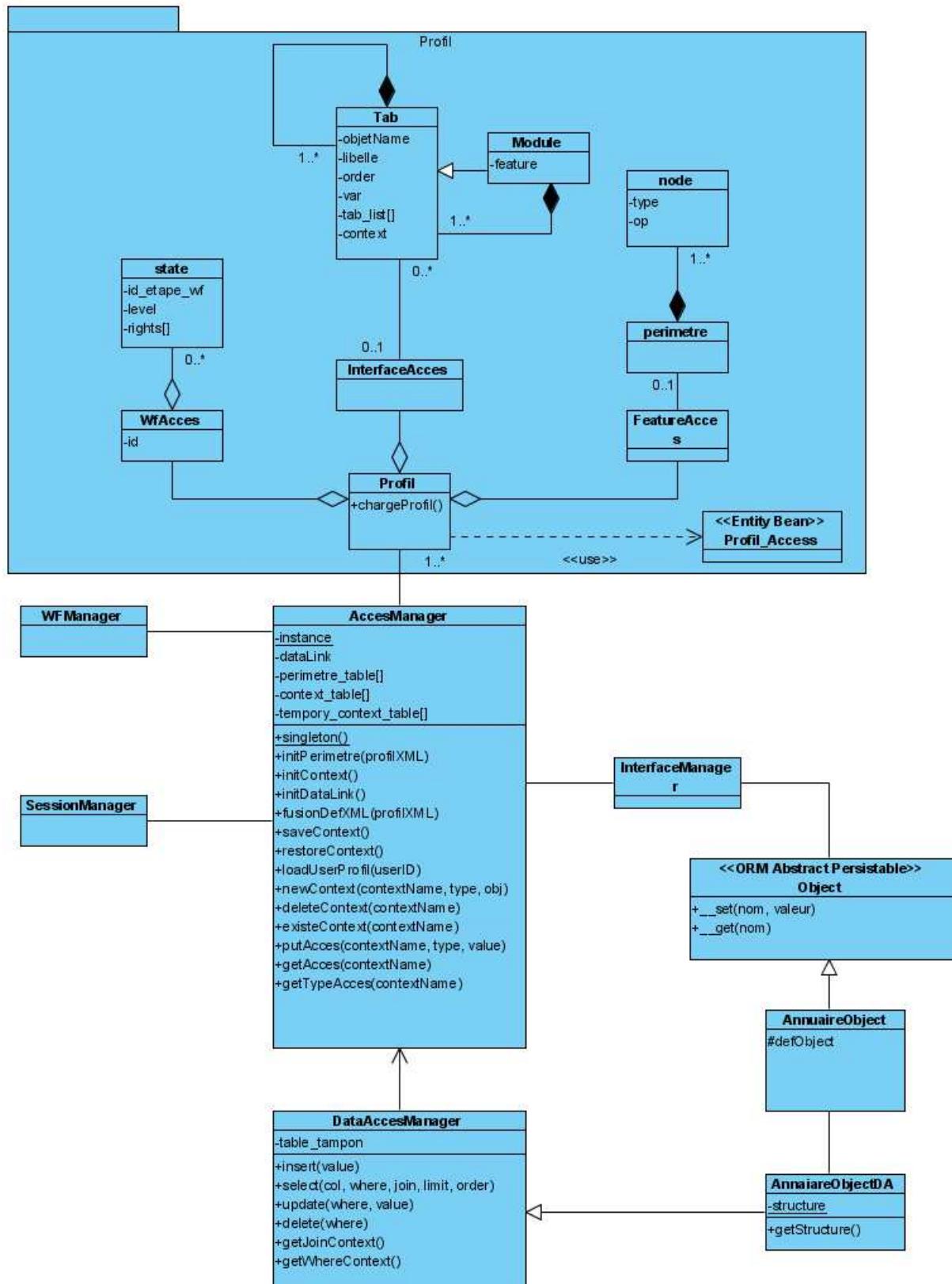
b. Travail réalisé

Ce projet étant en phase de démarrage mon travail a consisté à réaliser une étude préliminaire. Cette étude doit permettre de fixer les bases du projet. Pour cela il a fallu avoir un regard critique sur le système actuel tout en proposant des solutions.

Cette étude repose sur la conceptualisation d'objet. Elle a pour but, dans un premier temps, de fixer le périmètre d'action de chaque objet. Ce qui permettrait à terme d'avoir des objets indépendant tout en élevant leur niveau de compétence.

Pour cela il a fallu m'ouvrir aux concepts de base utilisé dans des applications sous architecture trois tiers.

Dans la suite de ce chapitre je vais décrire les concepts que j'ai pu extraire de cette première réflexion.

Figure 23 1^{er} approche conceptuelle des concepts V7

Classe AccessManager

Le but de cette classe est de gérer l'accès aux informations en prenant en compte des restrictions (profil ou contexte).

L'accès aux informations doit être totalement indépendant de leurs structures physique, ainsi que leurs modes de stockage (MySQL, LDAP...).

Pour cela cet objet doit :

- Mémoriser les différentes restrictions contextuelles dépendant de l'interface Web.
- Répondre aux demandes des objets métiers à travers le Data Access Manager afin d'assurer l'accès aux données en garantissant la bonne utilisation du contexte et du périmètre (profil).

Cette définition soulève un problème majeur. En effet, on souhaite accéder à des informations dont on ne connaît ni la forme ni le type. Cela pose un problème au niveau de son imperméabilité par rapport aux classes qui l'entourent.

Pour ce fait nous sommes parties sur des restrictions contextuelles formées sous:

- Un nom, il pourra être régit par une règle de nommage
- Un type (valeur alpha, valeur numérique, date, liste, arbre,...)
- Et une valeur.

Construction de l'objet

Cet objet doit être unique pour une session. Il doit donc être appelé par une méthode « singleton » qui gère l'unicité de la construction de l'objet.

Lors de la construction d'un nouvel objet il doit initialiser:

- les connexions vers le DataAccessManager,
- la table de contexte à vide,
- la table de périmètre à partir du profil par défaut.

Interface de l'objet

L'interface de l'objet est formée à partir :

- la gestion du périmètre : initialisation, accès, définition, type de règle de périmètre.
- La gestion du contexte : création, sauvegarde, restauration du contexte.

Structure interne

L'objet gère en interne 2 tables :

- table du périmètre
- table du contexte

Ces tables sont structurées :

| | |
|--------------------|---|
| Array[ContextName] | [TYPE] : valeur du type [ALL] : tableau des valeurs possibles [SELECTED] : tableau des items sélectionnés |
|--------------------|---|

Paquetage Profil

Un profil est une entité complexe d'une application. C'est à partir de celui-ci qu'il est possible de paramétriser l'apparence et le contenu de l'application.
C'est pour cela que l'on retrouve trois grands groupes.

La partie InterfaceAccess: elle définit la structure applicative de l'utilisateur. Celle-ci est définie comme un ensemble d'onglets nommés « module » ou « tab » dans lesquels peuvent se retrouver d'autres onglets encapsulés, de type « tab ».

Les « modules » sont conventionnellement de premier niveau de « tab » correspondant aux différents modules applicatifs, appareil fixe, appareil mobile, facture ...

Un « tab » est défini par:

- le nom de l'objet métier de référence,
- le libellé sous lequel il apparaîtra.

La partie FeatureAccess: elle définit le périmètre restrictif d'accès aux données. Elle est composée de périmètres qui décrivent les restrictions à apporter à l'accès aux données. On y retrouve une liste de « node ».

La partie WFAccess: elle définit les workflows (accès à la modification des données) autorisés. Elle permet d'associer à certaines étapes du WF des critères spécifiques (niveau de validation, droit d'affichage...)

Classe DataAccessManager

Cette classe permet de gérer l'accès aux données. Elle doit permettre de définir des fonctions génériques de manipulations de données : tel que l'insert, select, update.

A cela il faudrait y rajouter des classes d'accès aux données propre à chaque objet métier. Ainsi ces classes permettent de définir la structure physique des données.

CONCLUSION

Cette année d'alternance a été en premier lieu l'occasion de montrer mes capacités à m'intégrer efficacement dans une entreprise de développement d'applications orientées Web. Les résultats produits, prise en main de l'application, modélisation, évolution produit, sont le fruit d'un travail personnel.

Ces phases ont été réalisées en collaboration avec les membres du service Recherche & Développement de la société 2G TECHNOLOGIES.

L'entrée en matière dans l'analyse de l'application AUDITELcom fortement orientée objet ne s'est pas faite sans un investissement personnel très important. Ainsi, l'apprentissage du mode de son fonctionnement, du langage Javascript avec les concepts de l'AJAX, l'implantation de l'objet arbre, l'étude préliminaire des futurs concepts pour la version V7 représentent 80% du temps consacré durant cette année. Le reste étant utilisé pour apprivoiser le domaine métier de la GFT

Mon travail s'inscrit dans un processus d'évolution des produits de la société, tel que l'application AUDITELcom, ce qui permet de me positionner en temps que force de proposition, sur le plan technique.

À un niveau beaucoup plus personnel, cette année m'a permis de mettre en avant ma capacité d'analyse, tout en l'orientant vers son côté professionnel. Elle m'a montré ma faculté d'adaptation, grâce un bagage technique performant.

Finalement, je considère cette année comme une très bonne expérience, tant du point de vue humain que professionnel. La confiance que m'a exprimé mon responsable technique m'a permis de manifester pleinement mon potentiel.

Dans un avenir proche, j'ai le souhait d'être employé dans cette société pour continuer à faire évoluer les produits et de mener à bien les projets en cours de réalisation.

TABLE DES ILLUSTRATIONS

| | |
|---|----|
| <i>Figure 1 Organigramme du service R&D</i> | 10 |
| <i>Figure 2 Graphique du Chiffre d'affaire (en K€)</i> | 11 |
| <i>Figure 3 Tableau du Chiffre d'affaire (en K€)</i> | 11 |
| <i>Figure 4 Plan du site R&D de Toulouse</i> | 14 |
| <i>Figure 5 Modélisation du processus d'évolution de produit</i> | 17 |
| <i>Figure 6 Diagramme de Gant de la planification annuelle</i> | 20 |
| <i>Figure 7 Architecture globale de l'application AUDITELcom V6</i> | 22 |
| <i>Figure 8 Hiérarchie de l'arbre organisationnel</i> | 24 |
| <i>Figure 9 Impression écran de l'ancien arbre dans l'application</i> | 25 |
| <i>Figure 10 Diagramme de classe de l'ancien arbre</i> | 27 |
| <i>Figure 11 Architecture physique de l'application AUDITELcom</i> | 30 |
| <i>Figure 12 Diagramme de classe script Javascript</i> | 31 |
| <i>Figure 13 Diagramme d'activité de l'instanciation de TWebTree Javascript</i> | 33 |
| <i>Figure 14 Diagramme de Classe de l'arbre AJAX</i> | 35 |
| <i>Figure 15 Liste des fonctionnalités testées</i> | 37 |
| <i>Figure 17 Déroulement d'un nœud</i> | 38 |
| <i>Figure 16 Evolution graphique de l'arbre</i> | 38 |
| <i>Figure 18 Recherche dans l'arbre</i> | 39 |
| <i>Figure 19 Base ALL_WWW</i> | 41 |
| <i>Figure 20 Table pour la gestion des profils</i> | 42 |
| <i>Figure 21 Table des WF_Ticket</i> | 43 |
| <i>Figure 22 Diagramme de classe des WorkFlows</i> | 44 |
| <i>Figure 23 1^{er} approche conceptuelle des concepts V7</i> | 48 |

GLOSSAIRE

| | |
|---------------------|---|
| AJAX : | (Asynchronous JavaScript and XML) terme désignant l'association de plusieurs technologies Web. Principalement HTML et Javascript |
| Arbre : | Objet composé d'item (nœud) pour représenter des informations sous forme hiérarchique |
| ASP : | (application service provider ou fournisseur de service d'application) désigne une application sous forme de service informatique via un réseau |
| BD : | Base de données |
| CDR | Call Detail Record, ticket de taxation consignant les informations concernant un appel téléphonique |
| CSV : | (Comma-separated values) est un format informatique ouvert représentant des données tabulaires sous forme de « valeurs séparées par des virgules ». |
| EDI | L'Échange de Données Informatisées |
| Flyspray : | Logiciel qui permet d'organiser les tâches d'un projet, pour permettre le travail collaboratif |
| GFT : | Gestion Financière des Télécoms |
| HTML : | Hypertext Markup Language, est le format de données conçu pour représenter les pages web |
| JavaScript : | est un langage de programmation de scripts principalement utilisé pour les pages web interactives |
| JSON : | (JavaScript Object Notation) est un format de données textuel, générique. Il permet de représenter de l'information structurée |
| LDAP : | (Lightweight Directory Access Protocol) est à l'origine un protocole permettant l'interrogation et la modification des services d'annuaire |
| Mootools : | Mootools est un framework Javascript compact, modulaire, orienté objet |
| MySQL : | MySQL est un système de gestion de base de données |
| PABX | Un autocommutateur téléphonique privé |
| PHP : | (Hypertext Preprocessor), est un langage de scripts libre principalement utilisé pour produire des pages web dynamiques via un serveur HTTP |
| R&D : | Recherche et développement |
| SaaS : | (Software as a Service) est une technologie consistant à fournir des services ou des logiciels informatiques par le biais du Web et non plus dans le cadre d'une application de bureau ou client-serveur |
| SGBD : | Système de gestion de base de données |
| SQL : | (Structured Query Language), ou langage structuré de requêtes, est un pseudo-langage informatique (de type requête) standard et normalisé, destiné à interroger ou à manipuler une base de données relationnelle |
| UNIX : | est le nom d'un système d'exploitation multitâche et multiutilisateur |
| VPN | Dans les réseaux informatiques et les télécommunications, le réseau privé virtuel (Virtual Private Network en anglais, abrégé en VPN) est vu comme une extension des réseaux locaux et préserve la sécurité logique que l'on peut avoir à l'intérieur d'un réseau local |
| W3C : | World Wide Web Consortium est un organisme de normalisation |
| WF : | WorkFlow, est un flux d'informations au sein d'une organisation |
| XML : | (Extensible Markup Language (en) « langage extensible de balisage ») est un langage informatique de balisage générique |

SOURCES DOCUMENTAIRES

Site de documentation PHP:

<http://www.manuelphp.com/>

ManuelPHP.com propose les manuels officiels de trois langages de programmation :
PHP, MySQL et HTML.

Site Officiel de la librairie Mootools :

<http://mootools.net/>

Répertorie la documentation technique et des démonstrations d'utilisation

Site de collaborations informatique Developpez :

<http://www.developpez.com/>

Comprend de nombreux tutoriels et forums de discussion, sur différents langages.

INDEX

| | |
|--|-----------|
| REMERCIEMENTS | 3 |
| SOMMAIRE | 4 |
| INTRODUCTION..... | 5 |
| I. PRESENTATION DE L'ENTREPRISE | 6 |
| 1. L'HISTORIQUE | 7 |
| 2. LE SECTEUR D'ACTIVITE | 8 |
| 3. LES CONCURRENTS..... | 9 |
| 4. L'ORGANISATION..... | 10 |
| 5. L'ETAT FINANCIER..... | 11 |
| 6. LES PRODUITS | 12 |
| II. ORGANISATION ET ENVIRONNEMENT DU TRAVAIL..... | 13 |
| 1. MATERIEL | 14 |
| 2. POUR LE DEVELOPPEMENT..... | 15 |
| 3. POUR LE TRAVAIL COLLABORATIF..... | 15 |
| 4. POUR LA SYNCHRONISATION SERVEUR..... | 15 |
| 5. PERIODE TYPE..... | 16 |
| 6. ORGANISATION D'UNE EVOLUTION PRODUIT | 16 |
| III. PARTIE TECHNIQUE | 19 |
| 1. PLANIFICATION ANNUELLE | 20 |
| 2. ETUDE DE AUDITELCOM V6..... | 21 |
| a. <i>Descriptif</i> | 21 |
| b. <i>Détail technique</i> | 21 |
| c. <i>Analyse de la base de données</i> | 23 |
| 3. EVOLUTION MAJEUR DE L'OBJET ARBRE | 24 |
| a. <i>Etude de l'existant</i> | 25 |
| • Niveau statique..... | 27 |
| • Avantages et inconvénients..... | 28 |
| b. <i>Travail réalisé</i> | 29 |
| • Technologies Utilisées | 30 |
| • Partie Technique | 30 |
| Côté client | 31 |
| AjaxManager..... | 32 |
| TWebTree | 32 |
| Processus de construction d'un nœud | 33 |
| Côté Serveur | 35 |
| c. <i>Les Tests.....</i> | 37 |
| d. <i>Les Résultats</i> | 38 |
| 4. MODIFICATION DES WORKFLOW | 40 |
| a. <i>Etude de l'existant</i> | 40 |
| • Les WorkFlows en table | 41 |
| b. <i>Travail réalisé.....</i> | 44 |
| 5. ANALYSE D'OBJETS METIERS V7 | 46 |
| a. <i>Etude de l'existant</i> | 47 |
| b. <i>Travail réalisé</i> | 47 |
| c. <i>Classe AccessManager</i> | 49 |
| • Construction de l'objet | 49 |
| • Interface de l'objet | 49 |
| • Structure interne | 50 |
| d. <i>Paquetage Profil</i> | 50 |
| e. <i>Classe DataAccessManager.....</i> | 50 |
| CONCLUSION..... | 51 |

| | |
|-------------------------------------|-----------|
| TABLE DES ILLUSTRATIONS..... | 52 |
| GLOSSAIRE | 53 |
| SOURCES DOCUMENTAIRES | 54 |
| INDEX..... | 55 |