

IMIE Angers

Institut de la filière numérique



Projet de fin d'étude : Gestion des activités des services techniques

Lieu de stage : Mairie de Corné

Superviseurs du projet : E. BETIN (D-G services municipaux),

P. BODAN (Chargé des ressources humaines et de la comptabilité),

P.FOUASSIER (Responsable des services techniques)

Exécutant du projet : Mamadou G. DIOP

Dossier de projet de stage de :

DIOP Mamadou Gaye

Développeur logiciel _ Promotion 2013-2014

Avant-propos

Pour le programme « je me qualifie » de la région du Pays de la Loire 2013-2015, j'ai été admis en formation professionnelle pour le titre de Développeur Logiciels au sein de l'IMIE (Ecole de la filaire numérique) situé au centre Soulez LARIVIERE au 12, rue Philippe La PEYROUSSE 49000 Angers. La formation se compose en une période de 7 mois au centre et d'une période 7 semaines en entreprise P.A.E. (Période d'Application en Entreprise).

Dans le cadre de la P.A.E., ce document constitue le dossier de mon projet de stage réalisé à la mairie de Corné située à une quinzaine de kilomètres d'Angers, du 04 juin au 23 juillet 2014. Le projet de ce stage consiste en la réalisation d'un programme informatique client-serveur pour la gestion des activités des services techniques municipaux.

Le secteur des services techniques, jusqu'ici, utilisait un programme sous Acces-95 qui est jugé obsolète par rapport au besoin de gestion actuel.

Ce document traitera des principales étapes de ce projet qui, il faut le signaler, a été réalisé en l'absence d'un maître de stage dû à la non-existence d'un service informatique à la mairie.

SOMMAIRE

I.	Résumé du projet.....	6
II.	Compétences couvertes par le projet.....	8
II.1	Compétences professionnelles.....	8
II.2	Compétences transversales.....	8
III.	Expression des besoins.....	9
IV.	Spécifications fonctionnelles.....	11
IV.1	Contenu des écrans de l'utilisateur final.....	11
IV.2	Contenu de la base de données.....	12
IV.3	Transactions et cheminements entre l'utilisateur et la base de données.....	12
IV.4	Les traitements à effectuer pour chaque transaction.....	13
V.	Spécifications techniques.....	17
V.1	Liaisons du produit avec d'éventuelles applications existantes.....	18
V.2	Interface utilisateur.....	18
V.3	Localisation des terminaux utilisateurs et du serveur.....	19
V.4	Localisation de l'application et de la base de données.....	19
VI.	Réalisation.....	20
VI.1	Etude de l'existant.....	20
VI.2	Etude des besoins.....	20
VI.3	Planning.....	21
VI.4	Conception de la base de données.....	23
VI.5	Création de la base de données.....	25
VI.6	Maquettage.....	26

Dossier de projet de stage

VI.7	Développement.....	32
VI.8	Rédaction du manuel d'utilisation.....	32
VI.9	Déploiement.....	33
VI.10	Formation des utilisateurs.....	33
VII.	Conclusion.....	34
VIII.	Annexes.....	35

Remerciements

Je formule ces sincères remerciements à l'ensemble des individus qui, de près ou de loin, ont contribué à ce que j'aie effectué ce stage fort en expériences avec différentes problématiques qui ont largement enrichi mon savoir et mes compétences.

Ces remerciements s'adressent particulièrement au Directeur des services municipaux de la mairie de Corné, Monsieur Emanuel BETIN, et son Adjoint, Mr Pascal BODAN qui ont eu la confiance et l'amabilité d'accepter ma candidature me permettant de réaliser ce stage.

Ensuite à M. Pascal FOUASSIER, Responsable des Services Techniques de la mairie. Malgré un emploi du temps chargé, il a toujours fait preuve de disponibilité à mon égard. Mais également pour m'avoir fait confiance tout au long de mon stage en me laissant l'opportunité de prendre des initiatives et m'accompagner dans leur réalisation.

Par la même occasion, l'ensemble du personnel de la mairie, du restaurant scolaire et les élus locaux, sans oublier les élèves pour leur accueil chaleureux et leurs encouragements, mention spéciale à Mr Didier ROUGER, Adjoint délégué aux Bâtiments, voirie et espaces publics.

Je tiens également à formuler ma profonde gratitude à l'IMIE d'Angers, principalement nos formateurs et intervenants pour le savoir transmis.

Ces remerciements s'adressent également à l'administration de l'IMIE, particulièrement Mlle Céline LAIRMELIN pour l'efficacité de leur travail depuis le dépôt de ma candidature pour suivre cette formation.

Et pour finir, sans oublier l'ensemble de la promotion DL01 pour la bonne ambiance et la collaboration durant toute la formation.

Résumé du projet

Pour une meilleure gestion, il a été décidé de remplacer le programme de gestion des interventions des services techniques, qui ne présentait pas beaucoup de fonctionnalités, par un logiciel permettant d'avoir une vision plus ample sur les activités de ces services d'une grande importance.

J'avais pour mission de concevoir une simple solution informatique, capable de programmer, d'enregistrer et d'évaluer les interventions des services techniques. Pour ce faire, après des concertations avec les responsables et les utilisateurs, j'ai proposé une application client-serveur en utilisant le langage java et une base de données localisée sur le serveur de la mairie et accessible à partir de (ou depuis) tous les postes du réseau. Par contre, je me suis servi de la base existante pour concevoir une autre base de données qui permet de réunir le maximum d'informations utiles à la gestion des interventions, des temps de travail et des coûts. Après quelques itérations suites aux présentations de mon travail aux clients, le résultat du projet a été salué.

Abstract

For a better management, it was decided to replace the old program which was managing the missions of the technical services at the municipality of Corné by a new updated software. To have a better vision on the activities of that service of great importance.

My task was about creating a simple IT solution, allowing people to create, record and evaluate the activities of the technical services. I began the project by meeting the end users of the future software. I was trying to get the real needs of the users and know what they were expecting from the software. After analyzing the situation, I proposed a client-server application. A program executed on client computers and getting or writing information from or on the database in the server. I was supposed to migrate the existing database and make a new application from it. But that one didn't give many possibilities comparing the expectations of the users. So that, I had to make a new one based on the existing database. The application is made on java language with a graphic user interface by swing. After a few iterations from presentations of my work to the users, the outcome of the project was welcomed.

I made a user manual as the application help and another in PDF so that they can understand how to use the functionalities of the software. After the transfer of the database that I created on the server of the network, I installed the application on few client computers. I did training sitting with the principal users, to show them how to use the application.

In future I want to make a new version of that software, because I feel that I can go deeper with it.

II. Compétences référentielles couvertes par le stage

Durant le stage, j'ai pratiqué six des neuf compétences professionnelles et les deux compétences transversales. Cette situation est imposée par le fait que je n'avais pas un maître de stage dans le domaine.

II.1 Les compétences professionnelles :

- 1_Maquettage
- 2_Concevoir une base de données
- 3_Mettre en place une base de données
- 4_Développer une interface utilisateur
- 5_Développer des composants d'accès aux données
- 6_Utiliser l'anglais dans son activité professionnelle

II.2 Les compétences transversales :

- 1_Actualiser et partager ses compétences en développement informatique
- 2_Organiser son activité en développement informatique

III. Expression des besoins

A défaut d'un cahier des charges, j'ai reçu cette fiche de mission qui exprime les besoins, les attentes et les conditions du stage d'une manière bien concise :

Fiche mission Stage informatique – Mairie de Corné

Poste : développement d'une solution informatique ou d'un logiciel pour l'analyse des missions réalisées par les services techniques communaux mais aussi la création et la gestion des temps et des interventions

Lieu d'accueil : stage réalisé au sein de la mairie de Corné – 52 rue Royale – 49630 Corné – Collaboration avec les services techniques aux ateliers municipaux et sur le terrain.

La commune de Corné souhaite pouvoir se doter d'une solution informatique simple permettant le recueil et la programmation des interventions des services techniques, l'inscription des données remplies par les agents des services, la compilation et l'analyse de ces données pour une utilisation comptable de meilleure qualité.

La base utilisée actuellement a été réalisée il y a presque 15 ans à l'aide du logiciel ACCESS mais devient aujourd'hui obsolète.

Niveau d'études recherché : BTS

Description des fonctions :

Analyse détaillée des besoins de la collectivité

Propositions de mise en œuvre et d'éventuels outils à acquérir

Développement d'une solution logicielle

Rédaction d'un manuel d'utilisation

Tests d'utilisation et de simulation

Formation des utilisateurs

Compétences recherchées :

- Avoir un intérêt pour les missions de service public d'une collectivité locale
- Disposer de capacités rédactionnelles, de capacités relationnelles et de travail en équipe
- Faire preuve d'autonomie et de dynamisme
- Faire preuve de créativité

Durée indicative du stage : deux mois – dates indicatives du stage : dès que possible

Avantages liés au stage :

- Temps complet : 35h

Dossier de projet de stage

- Gratification à hauteur de 12 € par jour travaillé
- Prise en charge du repas par la collectivité au sein du restaurant scolaire
- Prise en charge des frais de déplacement

IV. Spécifications fonctionnelles

Dans ce chapitre, je vais traiter les spécifications fonctionnelles du projet. Dans la fiche de mission on peut lire, en premier, parmi la description des fonctions : Analyse détaillée des besoins de la collectivité.

IV.1 Contenu des écrans de l'utilisateur final

Avant d'analyser quoi que ce soit, il est fortement recommandé de comprendre le sujet. Alors, ma première action a été rencontrer les acteurs et responsables pour mieux cerner les besoins réels qu'ils attendaient du produit. A la sortie des rencontres, j'ai compris que le produit devrait en mesure de :

- _ ajouter des types de tâche dans une catégorie et s'il s'agit d'une catégorie inexistante, la possibilité d'en créer est offerte.
- _ajouter de nouveaux locaux dans un site, sinon créer le site en tant que tel.
- _ajouter des produits ou outils d'intervention.
- _ajouter des équipes et leur affecter des membres.
- _ajouter des agents qui, engagés soit pour une durée temporaire ou bien indéterminée.
- _enregistrer les interventions des agents à partir d'une fiche journalière et les produits et outils utilisés si c'est le cas.
- _ rechercher et afficher des types de tâche selon leur catégorie ou simplement par mot clé.
- _rechercher et afficher les locaux municipaux par site ou par mot clé.
- _ rechercher et afficher des agents par type de contrat ou par mot clé.
- _rechercher et afficher les équipes avec option d'en afficher par domaine d'activité.
- _rechercher et afficher les produits et outils utilisés ou en stock.
- _constituer un outil pour assister le responsable des services techniques à programmer des interventions avec une fiche journalière.

_évaluer des interventions sur une période déterminée et selon une équipe, un lieu donné, un type de tâche ou simplement selon les interventions.

_calculer le nombre d'interventions, leur durée et leur cout, par équipe, par localité ou par type de tâche. Le produit calcule également le temps de travail individuel de chaque agent pour des fins comptables.

_imprimer les états des différentes tables.

IV.2 Contenu de la base de données

C'est enjoué de voir toutes les fonctions que le produit devait offrir, non ? Mais si on y réfléchit bien on se rend compte que la base de données doit contenir une multitude d'informations. Sachant que la base de données existante était loin de répondre à ces requêtes, d'où le besoin de la reprendre de zéro. Pour être franc, je pensais qu'il fallait juste faire une migration, mais ça, c'était avant de voir la base de données existante (...illusion !!). La nouvelle base de données, pour répondre aux besoins, devrait permettre de :

_ recenser toutes les formes d'activité des services techniques qui regroupent un bon nombre de tâches. Ces tâches sont divisées en différentes catégories.

_ recenser tous les lieux dont les services techniques en ont la charge, ce qui fait une liste qui ne finit jamais...j'exagère quand même...je dois dire presque jamais, si on considère que dans une école par exemple chaque pièce est un local avec son propre code, y compris les salles de classe, les toilettes, la cour de l'école et j'en passe. Tous ces locaux sont abrités par des sites.

_recenser tous les intervenants.

_recenser les différentes équipes d'interventions et leurs membres.

_recenser les produits et outils pouvant être utilisés durant une intervention tout en tenant compte que de leur valeur changeante selon le fournisseur ou la période.

_recenser les interventions effectuées et l'utilisation de produits et outils, s'il y en a eu.

IV.3 Transactions et cheminements entre l'utilisateur et la base de données

L'utilisateur devant son écran pourra interagir avec la base de données située sur le serveur de la mairie à travers l'application installée sur son poste de travail. Il peut :

_enregistrer des données.

_rechercher et afficher des informations.

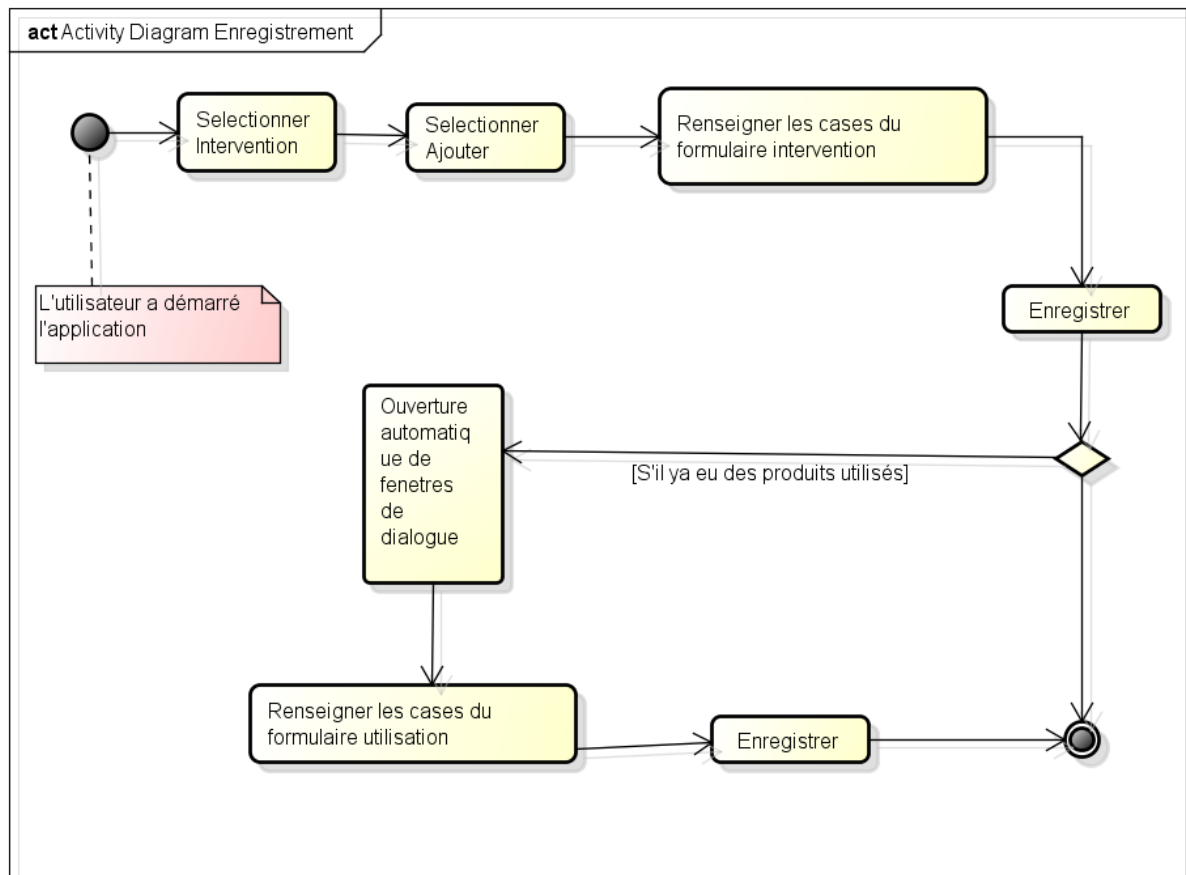
_modifier des données entrées dans les menus où cette fonction est disponible.

_faire des requêtes plus complexes telles que des calculs ou des évaluations par exemple pour connaître les différentes interventions opérées par une équipe donnée, dans un local bien déterminé ou par type de tâche durant une période bien précise.

IV.4 Traitement à effectuer pour chaque transaction

Les transactions entre l'utilisateur et la base de données se feront par le biais de l'interface graphique, donc ses traitements se seront effectués sur des formulaires et des tables.

_Pour enregistrer des données l'application lui présentera des formulaires pour saisir les informations et les valider par un bouton, l'enregistrement lui sera notifié ou pas en cas d'erreur.



powered by Astah

Diagramme d'activité : Enregistrement d'intervention

_Pour modifier une entrée, l'utilisateur devra la rechercher, l'afficher et la sélectionner enfin d'accéder à sa modification dans le formulaire. Il pourra apporter les modifications voulues et l'enregistrer.

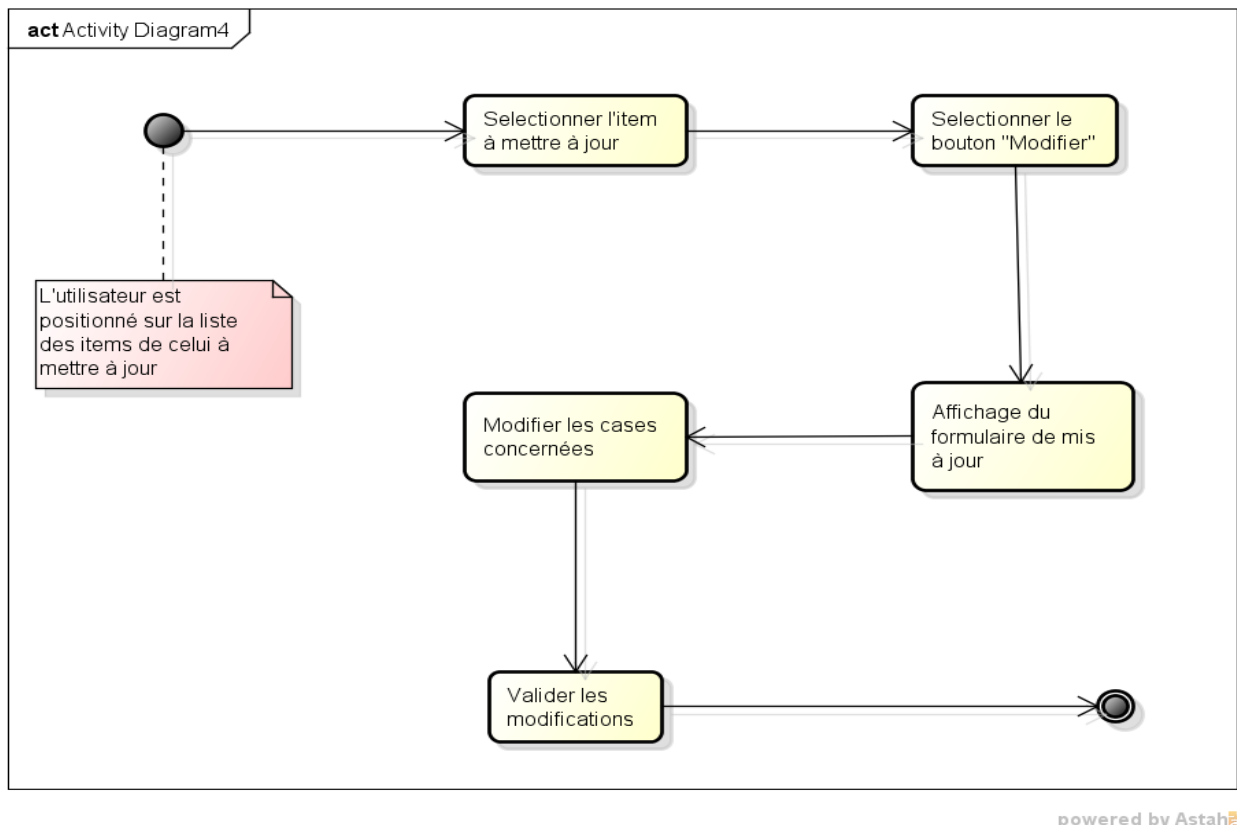
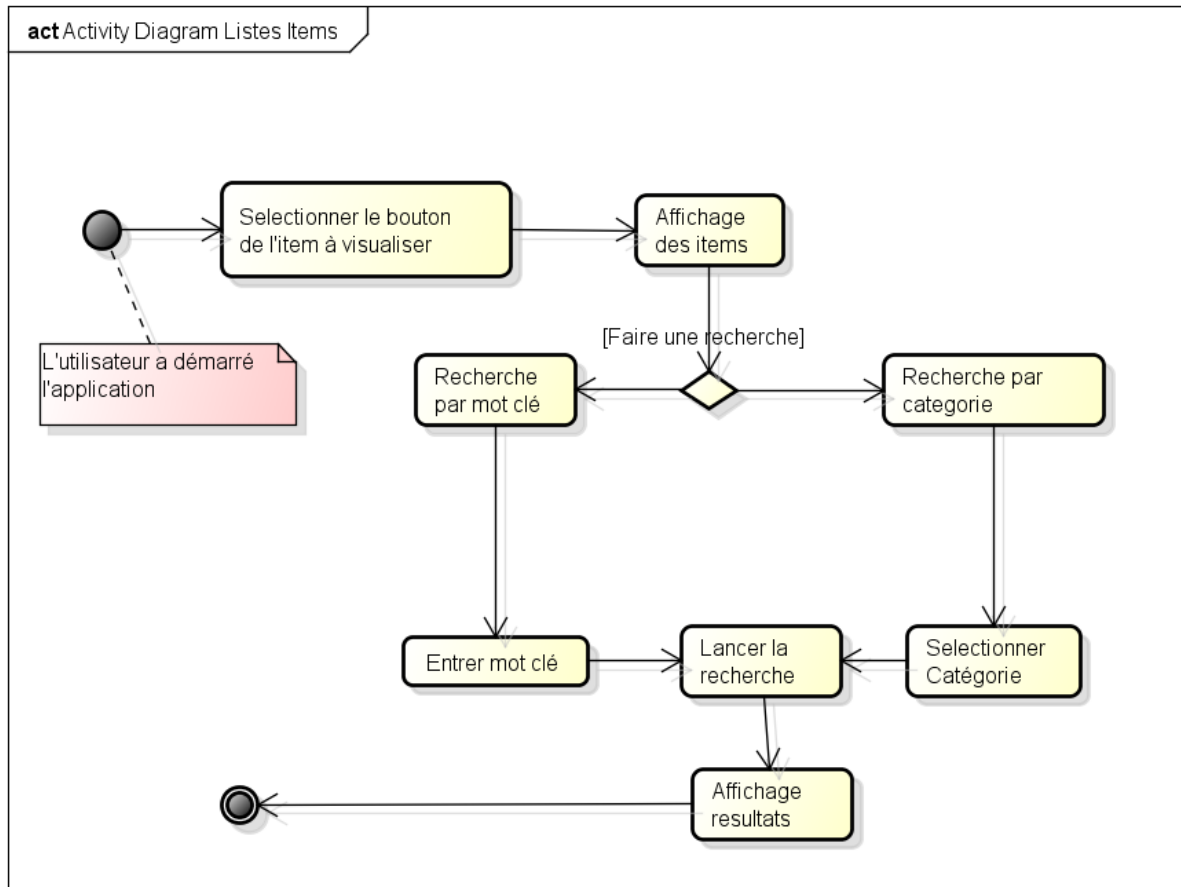


Diagramme d'activité : modifier un item

_Pour rechercher et afficher des données, l'utilisateur choisira le menu de son choix pour faire sa recherche :

- soit par mot clé, en entrant le mot clé et lancer sa recherche,
- soit par catégorie, en sélectionnant la catégorie depuis un menu déroulant, puis, lancer la recherche.



powered by Astah

Diagramme d'activité : Faire une recherche

_Pour faire des évaluations ou des calculs, l'utilisateur devra sélectionner une période comprise entre deux dates, l'item et son genre avant de lancer l'opération.

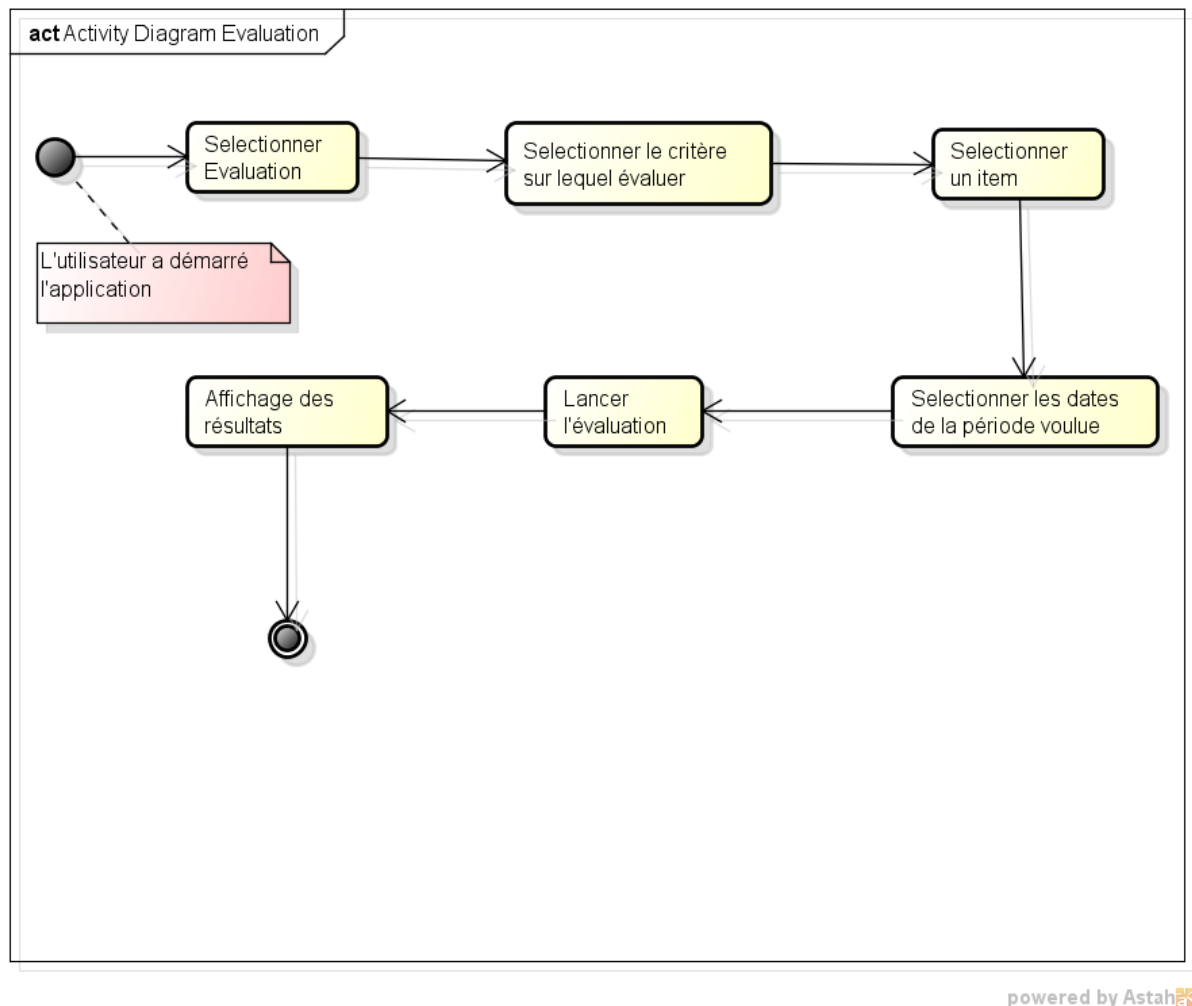


Diagramme d'activité : évaluation

V. Spécifications techniques

Ce chapitre concerne la partie technique de mon analyse du projet :

- des liaisons du produit avec d'éventuelles applications existantes,
- de l'interface utilisateur,
- de la localisation des terminaux utilisateurs et du serveur,
- de la localisation de l'application et de la base de données,
- des volumes etc.

Je définirai mon choix sur certains aspects important du produit, tels que le réseau, l'architecture, la sécurisation, la documentation, et bien sûr des résultats du produit.

V.1 Les liaisons du produit avec d'éventuelles applications existantes

Le produit du projet a deux parties qui vont être exploitées par la comptabilité de la mairie, à savoir les temps de travail des agents et le coût des interventions. Mais il n'existe pas vraiment de liaison directe avec une autre application existante et de l'application que le produit est censé remplacer servira désormais d'archive pour son remplaçant.

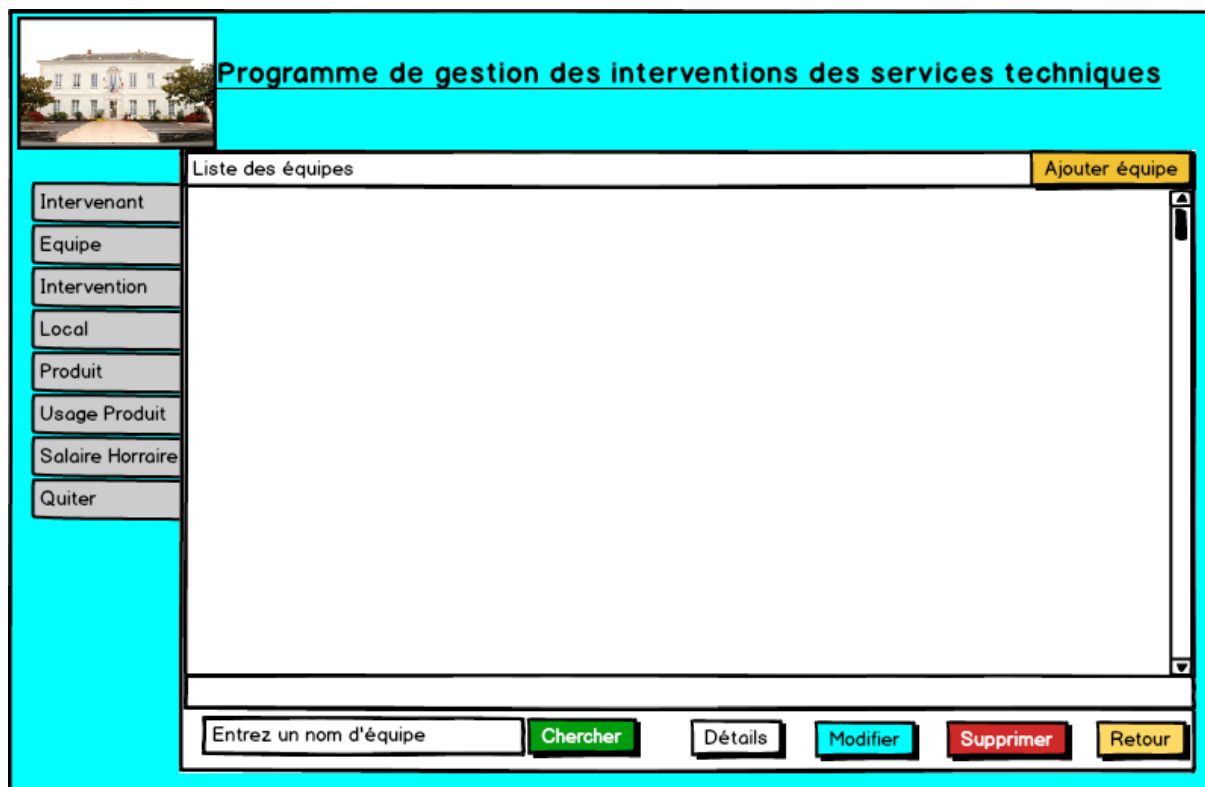
V.2 L'interface utilisateur

Durant notre formation au centre, le langage Java fut le seul langage orienté objet qu'on a vu, donc le choix du langage est automatique (*c'est-à-dire pas vraiment le choix..*). Donc j'ai choisi swing pour concevoir l'interface utilisateur et bénéficier de la bibliothèque d'objets suivant les besoins du produit. L'interface présente une fenêtre (JFrame) divisée en trois parties :

_L'entête du produit contenant le titre et une photo de la mairie.

_La navigation qui contient les boutons du menu de l'application (style PopupMenu).

_La présentation avec les différents composants superposés (CardLayout) avec certaines fenêtres de dialogue (voir figure ci-dessous).



V.3 Localisation des terminaux utilisateurs et du serveur

La mairie possède un réseau local avec un serveur, sous Windows server 2008. Ce dernier contient tous les documents de la mairie et son site web.

Le système a été installé par une entreprise du nom de Point-6, qui m'a donné l'aval d'y installer une machine virtuelle (VMware).

J'ai installé Debian version 6 sur cette machine virtuelle pour y stocker la base de données de l'application.

En résumé, les terminaux utilisateurs et le serveur sont localisés au sein de la mairie.

V.4 Localisation de l'application et de la base de données

L'application sera installée sur trois postes de travail :

_celui de la secrétaire, qui est chargée d'enregistrer les fiches journalières.

_celui du responsable des services techniques, qui programme les interventions, ajoute les nouveaux éléments tels qu'un nouvel agent, un nouveau local etc...

_celui du chargé des ressources humaines et de la comptabilité, qui a besoin d'accéder aux données pour des raisons comptables tels que le temps de travail des agents et les coûts d'intervention.

La base de données se situe sur le serveur Debian de la machine virtuelle hébergée par serveur de la mairie.

A l'aide du JDBC de l'application, la connexion avec la base de données s'ouvre à temps voulu et se referme à la fin du processus déclenché au niveau des postes de travail.

Remarques :

La base de données sera d'abord installée sur un serveur local (WAMP Server) durant le long de l'élaboration du projet. Une fois le développement fini, elle sera transférée au serveur de la mairie.

Pour ce faire, une machine virtuelle (VMware) sera installée sur le serveur pour ne pas interférer avec la configuration de ce dernier. Sur la machine virtuelle il

y'aura Debian qui servira de serveur pour l'application. La base de données est conçue avec MySQL, qui est simple d'utilisation et approprié pour une base de données petite taille.

Pour assurer la disponibilité de la base de données, j'ai configuré la machine virtuelle de manière à ce qu'elle s'exécute au démarrage du serveur.

VI. Réalisation

Dans ce chapitre, se trouvent les différentes étapes de réalisation du projet, de la conception à la formation des utilisateurs.

VI.1 Etude de l'existant

Les services techniques ont pour mission principale de maintenir en état de fonctionnement le patrimoine bâti et non bâti de la commune. Outre l'entretien préventif et curatif du patrimoine, ce service pluri-professionnel assure également la maîtrise d'œuvre de travaux neufs, aussi bien en bâtiment, espaces publics, qu'en voirie. Ce qui nécessite par ailleurs une logistique de gestion des différentes interventions des services.

Dans le cas de la mairie de Corné cette gestion se faisait à l'aide d'un programme conçu sous Access95, il y a de cela plus d'une quinzaine d'années. Le programme existant se trouvait sur un poste et ne permettait que des enregistrements plutôt imprécis sur les activités. Cette configuration ne donnait pas beaucoup de possibilités pour une gestion adéquate. Je peux citer l'exemple des fiches journalières qui ne concernaient qu'un seul agent à la fois alors qu'une même intervention est souvent effectuée par un groupe d'agents.

Pour des soucis économiques, la mairie a décidé de prendre un stagiaire et lui faire faire une solution qui répondrait juste à leurs besoins. Pour éviter de dépenser beaucoup d'argent sur un logiciel (surdimensionné) super sophistiqué dont plus des 90% des fonctions seront inutilisés.

VI.2 Etudes des besoins

La commune de Corné souhaite se doter d'une solution informatique simple, permettant :

- le recueil et la programmation des interventions des services techniques,
- l'inscription des données remplies par les agents des services,

- la compilation et l'analyse de ces données pour une utilisation comptable de meilleure qualité.

Ainsi se présente l'expression des besoins notée sur la fiche de mission.

J'ai organisé une rencontre avec le directeur des services municipaux, le responsable des services techniques et le chargé des ressources humaines et de la comptabilité, dans le but de mieux comprendre le sujet. A la sortie de cette réunion j'ai retenu que :

_plusieurs facteurs étaient impliqués dans l'organisation d'une intervention des services techniques. Chaque opération était effectuée par une personne ou une équipe appartenant à un des trois secteurs que comptaient les services techniques (Bâtiment, Espaces verts et Voirie).L'opération se déroulait dans un lieu précis et concernait une tâche bien déterminée parmi des centaines.

_que les agents avaient leur spécialité mais étaient polyvalent aussi, c'est-à-dire que un agent des espaces verts ou de la voirie pouvait participer à une intervention dans le bâtiment et vice versa.

_une mission pouvait se faire, par exemple, dans une seule pièce d'un bâtiment, dans une partie d'espace vert d'un parc ou dans un tronçon de route.

_une équipe pouvait faire plusieurs interventions par jour comme une intervention pouvait s'étaler sur plusieurs jours.

_intervention pouvait nécessiter des matériaux et/ou des engins (utilisation de béton, goudron ou la location d'une pelleteuse).

_la comptabilité avait besoin d'une certaine précision sur les coûts des interventions et les temps de travail des agents.

Tous ces aspects font que les analyses devaient être bien poussées pour concevoir un produit qui réponde aux besoins. C'est ce qu'on a fait tout au long du projet, en se renseignant et en faisant des propositions.

VI.3 Planning

Après analyses et le peu d'expériences que j'avais, j'ai établi un planning du projet tout en me donnant des marges de temps me permettant de rendre mes livrables à échéance. J'ai utilisé la technique de PERT pour aboutir à un diagramme de GANTT illustrés ci-dessous :

Diagramme de PERT

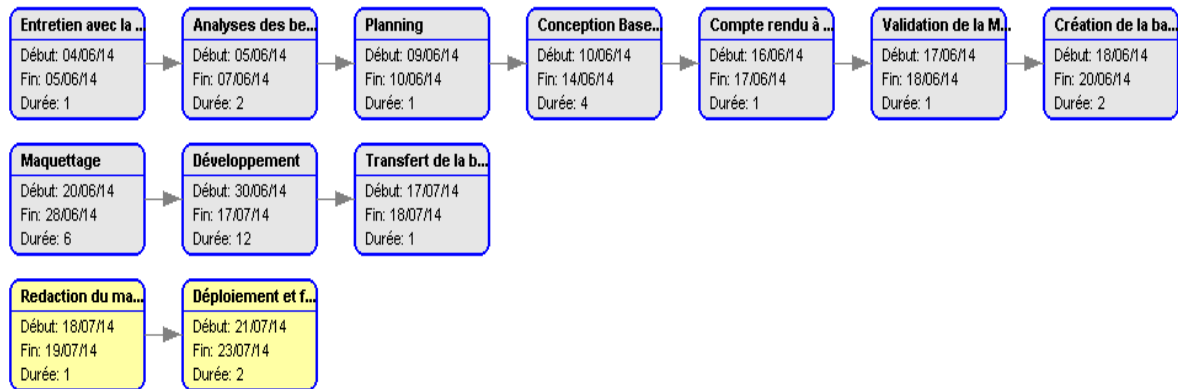
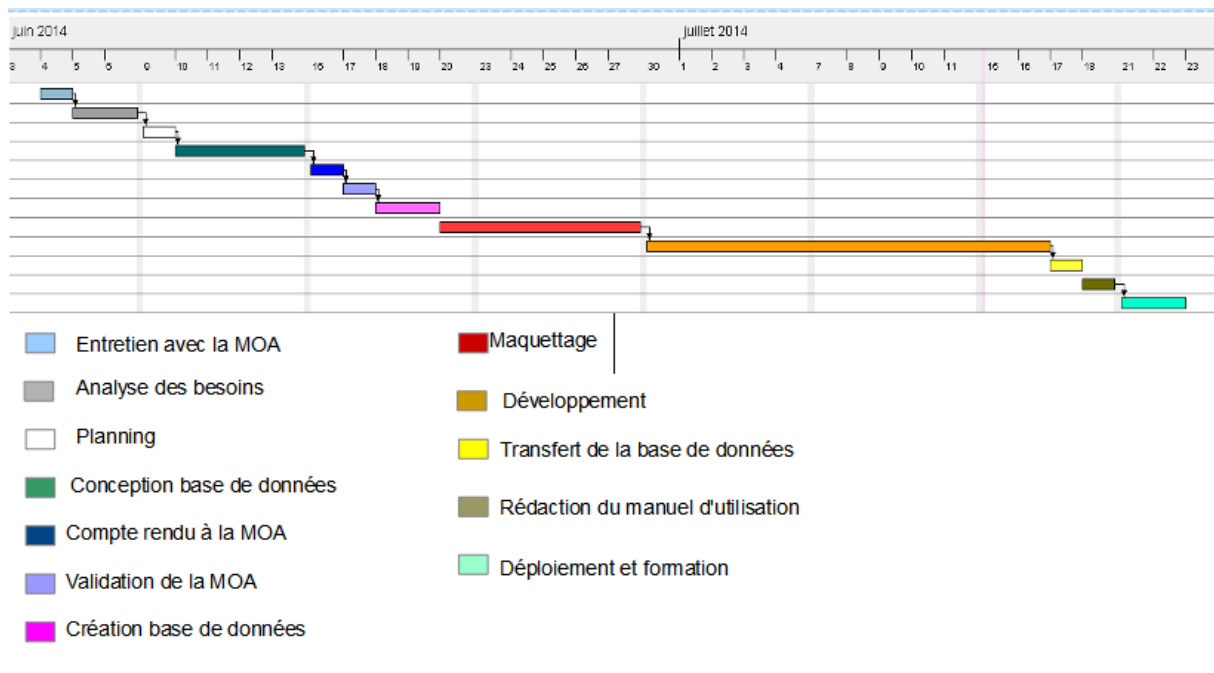


Diagramme de Gantt



VI.4 Conception de la base de données

La conception de la base de données n'a pas été une chose facile, vu le nombre de facteurs qui étaient impliqués dans une intervention. Cependant, j'ai pu atteindre mon objectif en prenant certaines initiatives sur le fonctionnement telles que :

_reprendre une nouvelle fiche journalière quotidiennement pour les interventions qui duraient plus d'une journée, et de considérer chaque jour comme une intervention à part entière.

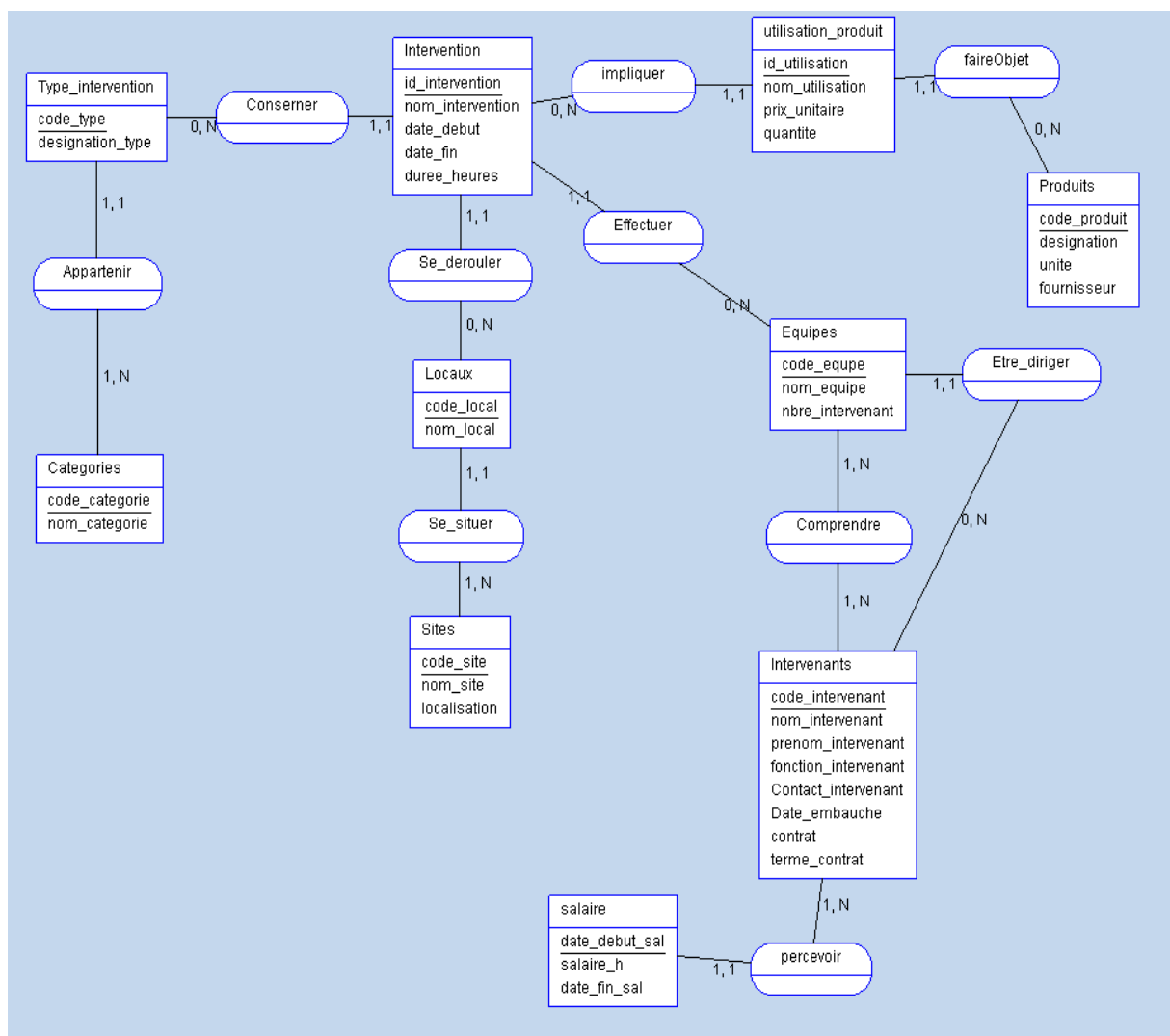
_considérer en tant qu'équipe tout agent qui serait amené à travailler individuellement, pour éviter d'avoir deux entités d'intervenants.

_mettre les prix de chaque matériau lors de son utilisation et non dans la table des produits, pour prévoir les changements de prix qui pouvaient intervenir dans le temps.

J'ai utilisé la méthode MERISE pour concevoir la base de données.

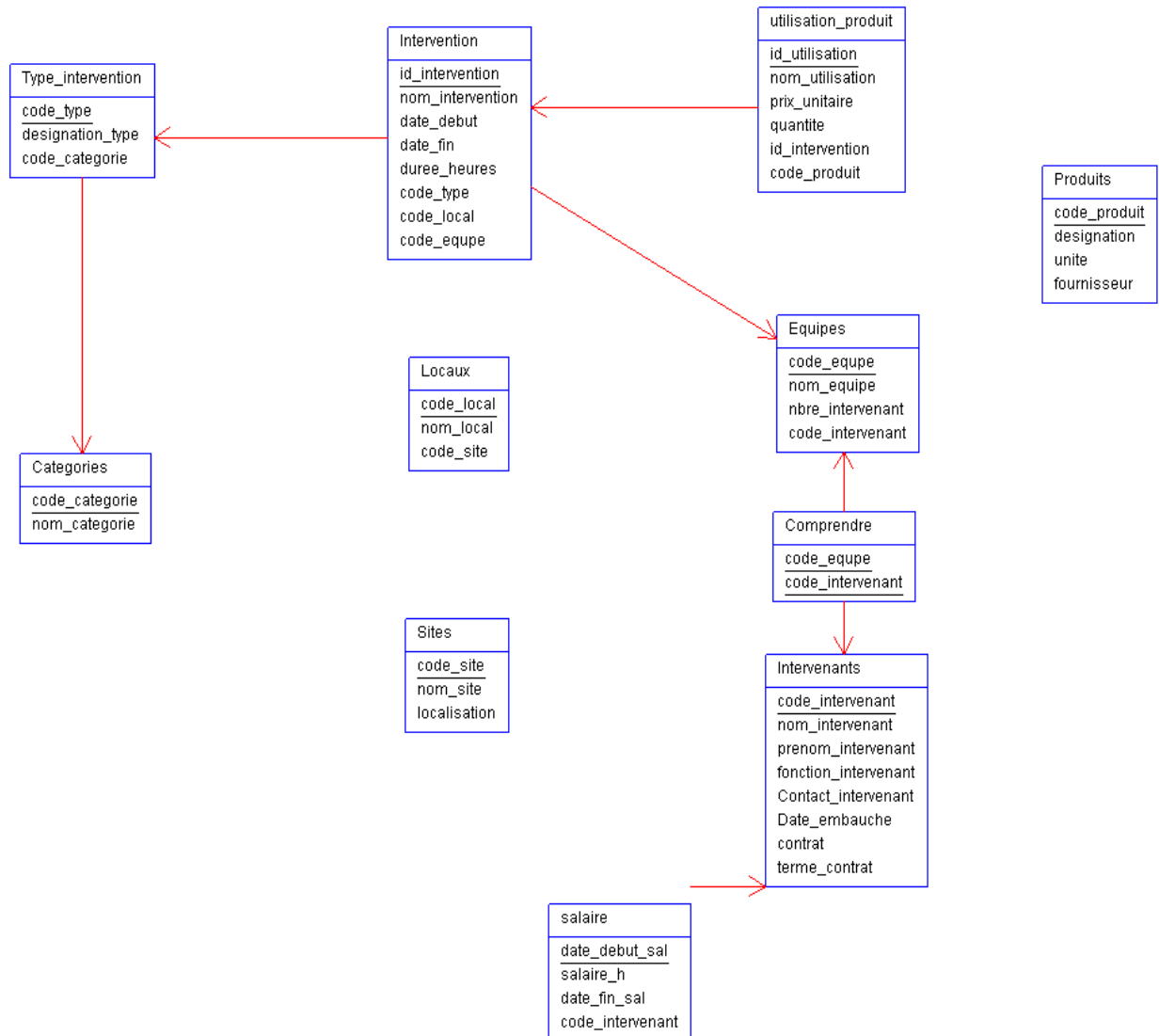
Le M.C.D (Modèle de Conceptuel de Données) est représenté par la figure ci-dessous.

MCD (Modèle Conceptuel de Données)



Ce MCD est fait avec le logiciel AnalyseSI. Après transformation du MCD en MPD (Modèle Physique de Données) le résultat est schématisé dans la figure ci-dessous.

MPD (Modèle Physique de Données)



VI.5 Création de la base de données

La création de la base de données s'est fait en deux temps :

_sur un serveur local (WAMP Server) en attendant de rencontrer les techniciens de l'entreprise Point-6 qui ont configuré le réseau de la mairie. Le logiciel

AnalyseSI avec lequel j'ai conçu la base de données, génère un code SQL. Après la génération du code on se connecte directement au serveur pour qu'il crée la base de données avec les tables du MPD.

_sur le serveur de la mairie avec à l'aide d'une machine virtuelle (VMware). Le système d'exploitation utilisé est Debian 6, sur lequel j'ai installé Apache 2, PHP 5 et MySQL server. Après configuration, j'ai transféré la base de données depuis mon serveur WAMP.

Les codes de création de la base de données sont reportés en annexe avec les requêtes concernées.

J'ai aussi créé des vues pour répondre à certaines fonctionnalités de l'application, elles sont visible dans les scripts de sauvegarde de la base de données.

VI.6 Maquettage

Dans cette étape du projet, j'ai utilisé :

_la méthode UML (Unified Modeling Language) pour les diagrammes avec Astah community

_des maquette avec l'outil Balsamiq Mockups. Après conception, je me faisais valider par le responsable des services technique qui était le principal utilisateur de l'application. C'est ce qui explique certaines nuances entre les maquettes et l'aspect final de l'interface utilisateur.

Les principales fonctionnalités de l'application sont définies comme suit :

_ajouter des items comme un site, un agent, une tache, un produit etc... ces ajouts se font à l'aide de formulaires présentés à l'utilisateur.

_afficher des items en faisant des recherches dont les résultats seront affichés sous forme de tableaux (JTable).

_enregistrer des interventions en se basant sur les fiches journalières retournées par les agents après chaque journée de travail.

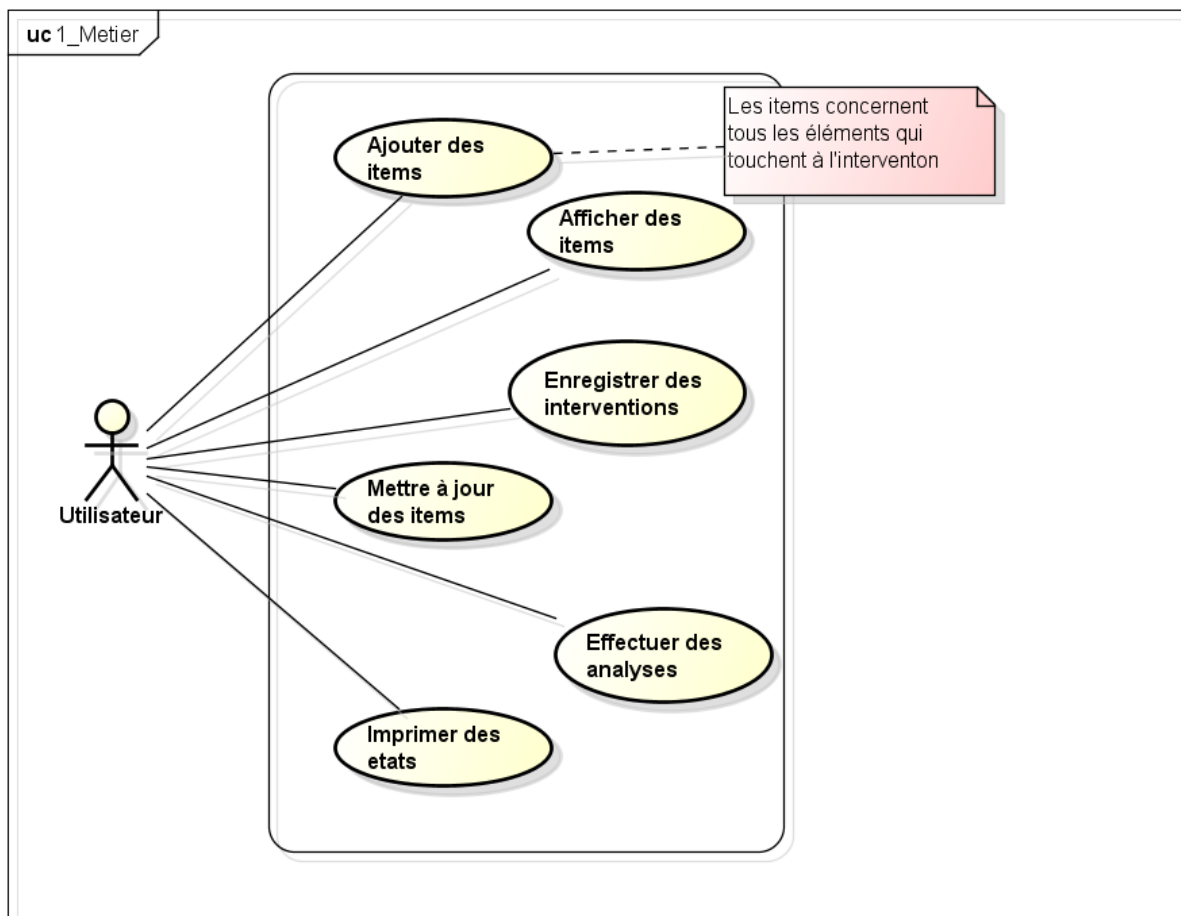
_mettre à jour des items. Pour des soucis d'exactitude dans les analyses, seuls deux composants de l'application possèdent la fonction « modifier », à savoir les interventions et les agents.

_effectuer des analyses sur l'item de son choix en sélectionnant la période concernée. Cette partie permet de programmer les interventions et de calculer les coûts des interventions et les temps de travail des agents.

_imprimer des résultats de recherche sous forme de tableaux.

Ces fonctionnalités sont schématisées à l'aide d'un diagramme de cas d'utilisation sur la figure suivante.

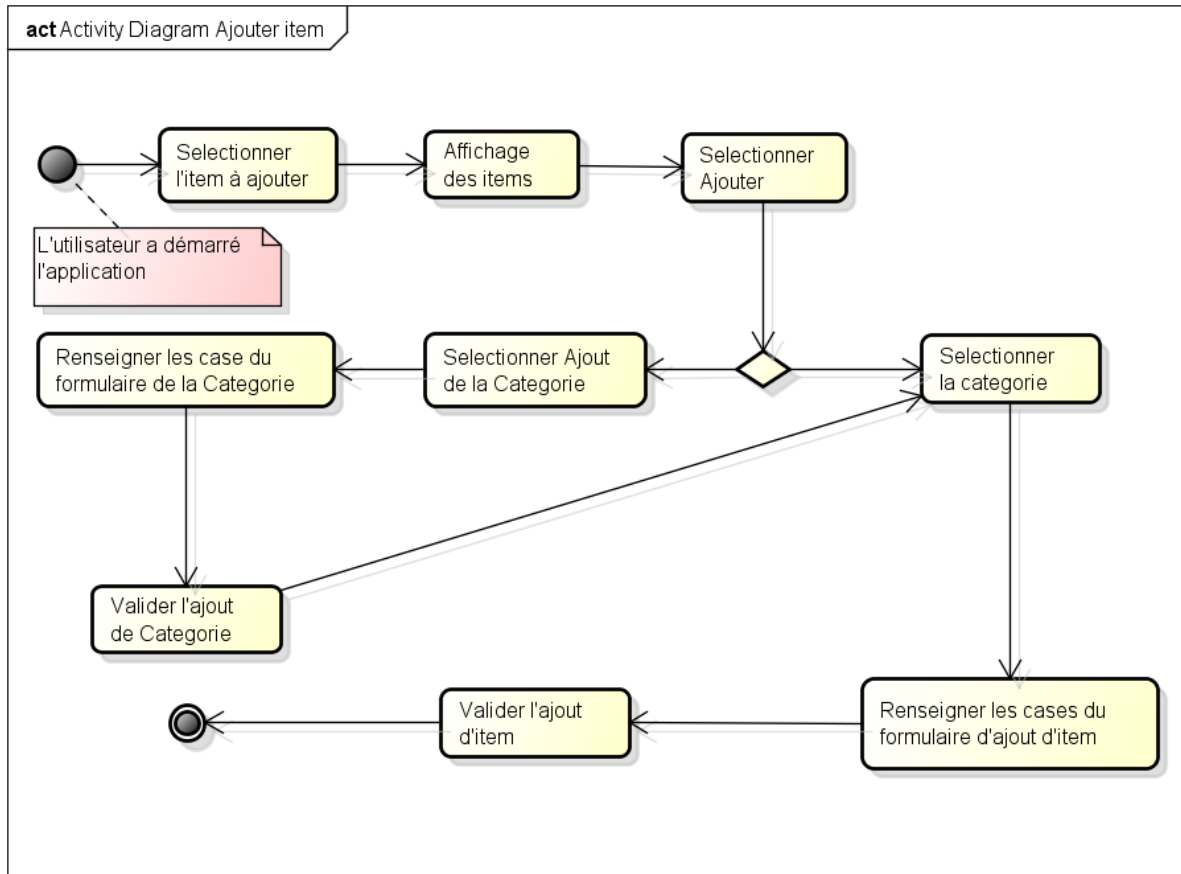
Diagramme de cas d'utilisation



powered by Astah

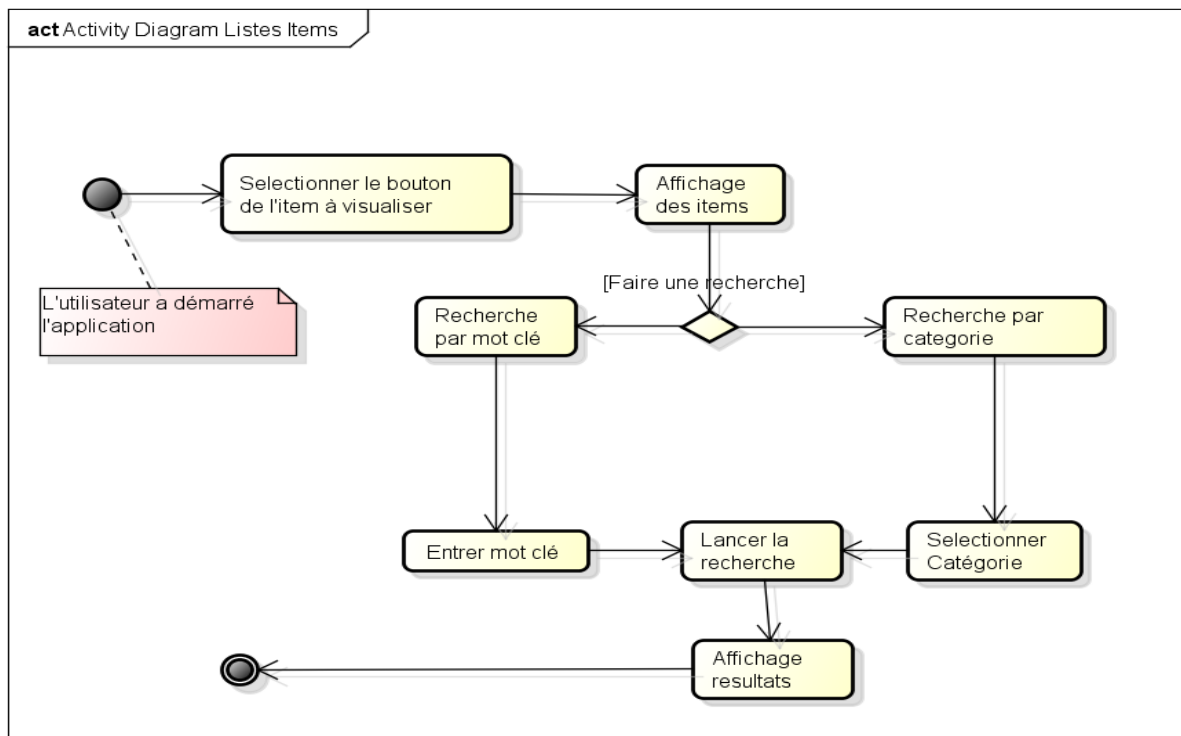
Le diagramme ci-dessus est décomposable en autant de diagrammes qu'il comporte de fonctionnalités. Dans mon cas j'ai opté pour des diagrammes d'activité pour chaque fonctionnalité.

_Ajouter des items :



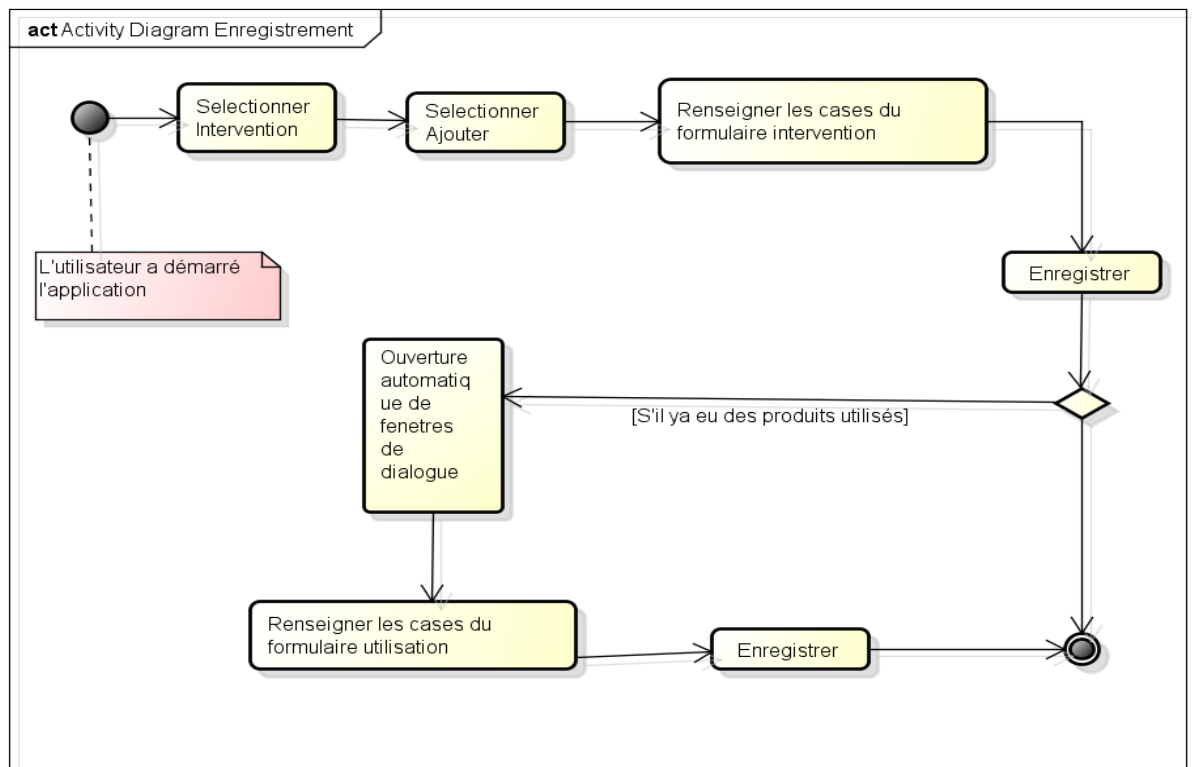
powered by Astah

_Afficher des items



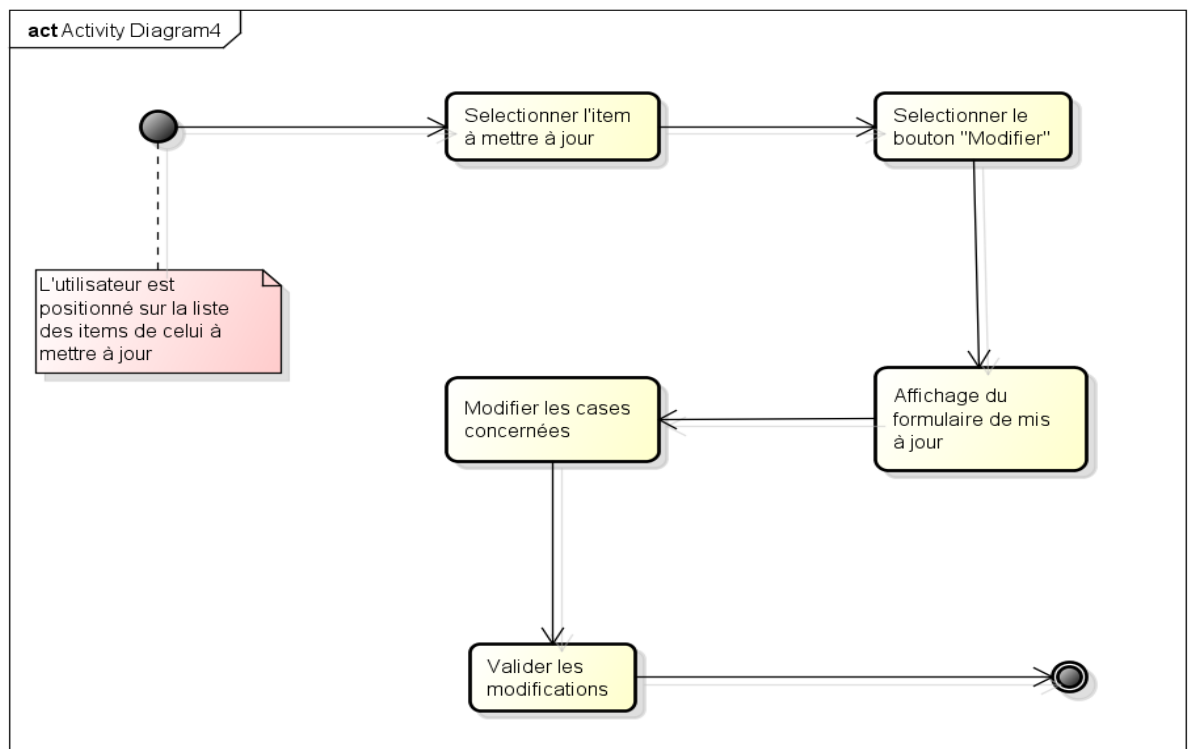
powered by Astah

_Enregistrer des interventions



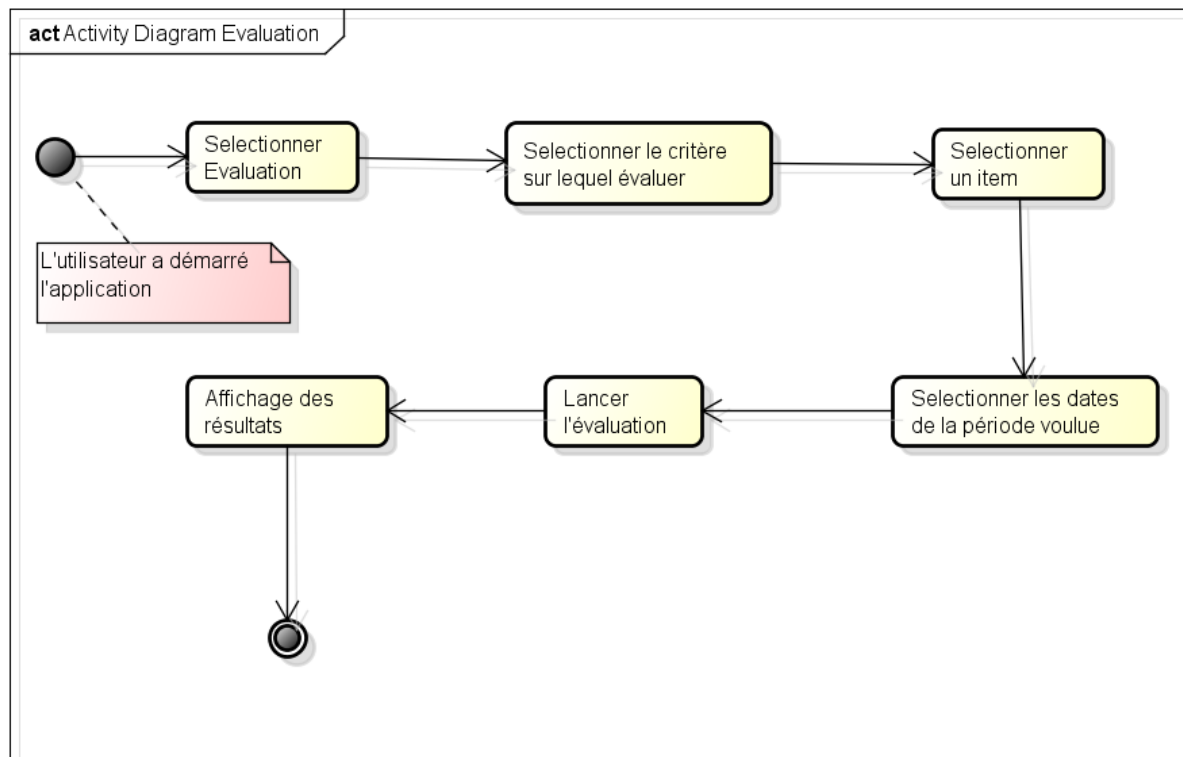
powered by Astah

_Mettre à jour des items



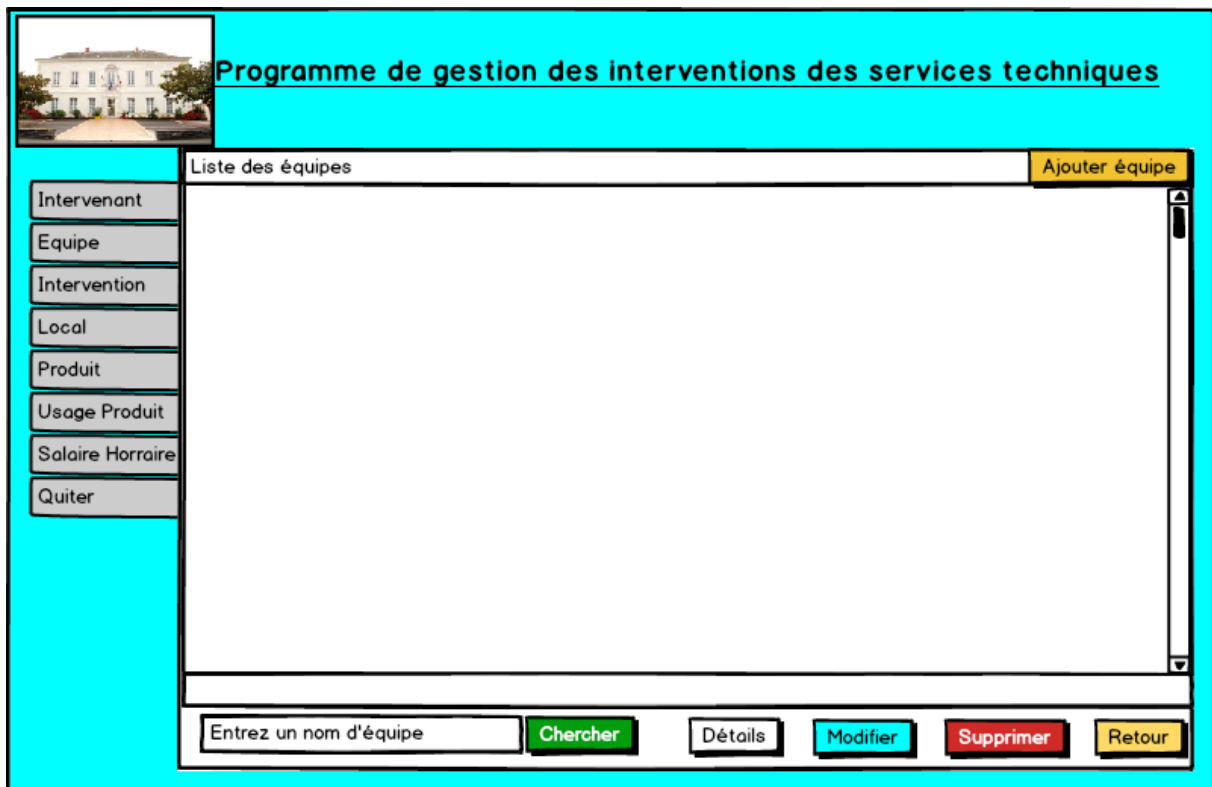
powered by Astah

_Effectuer des analyses



Ces diagrammes d'activité m'ont permis de passer aux maquettes de l'interface utilisateur et de pouvoir déterminer ses principaux composants (fenêtres). Les maquettes de ces composants ont subi des améliorations au moment du développement pour des soucis ergonomiques. En voici quelques exemples :

_Maquette de la fenêtre des équipes d'intervention



Programme de gestion des interventions des services techniques

Intervenant
Equipe
Intervention
Local
Produit
Usage Produit
Salaire Horaire
Quiter

Liste des équipes

Ajouter équipe

Entrez un nom d'équipe

Chercher

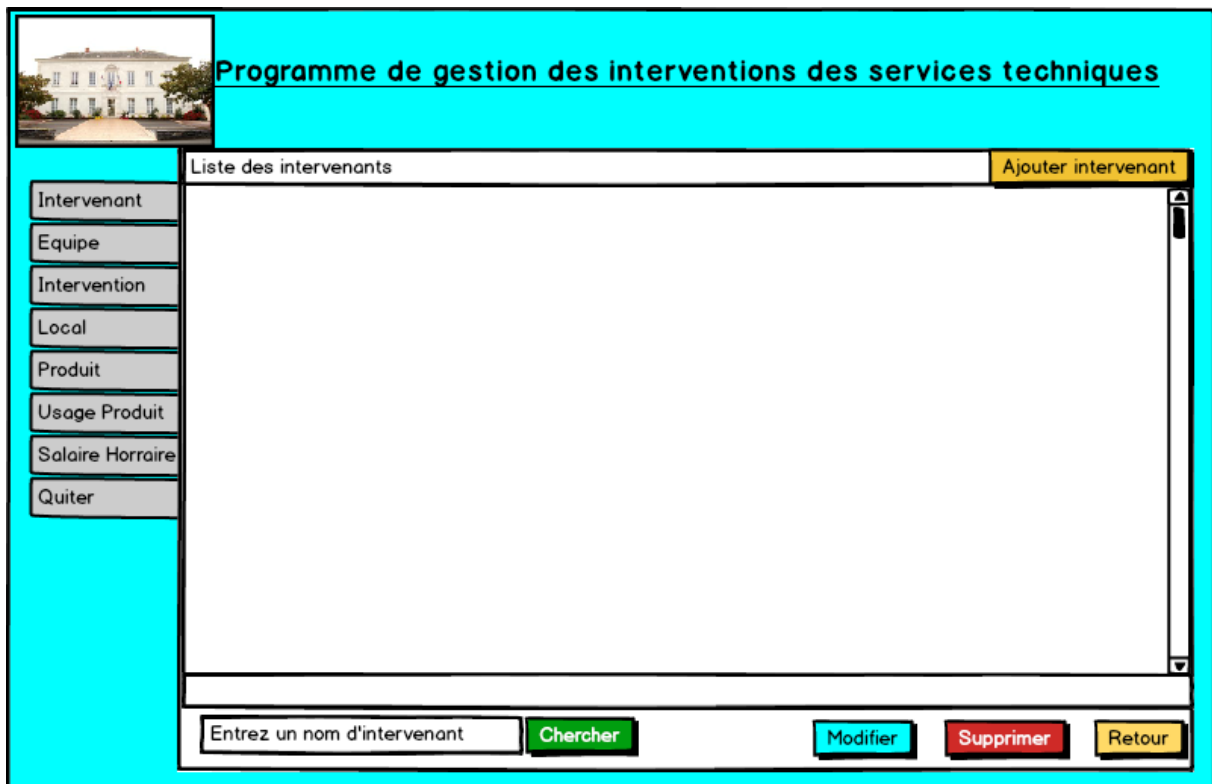
Détails

Modifier

Supprimer

Retour

_Maquette de la fenêtre des agents d'intervention



Programme de gestion des interventions des services techniques

Intervenant
Equipe
Intervention
Local
Produit
Usage Produit
Salaire Horaire
Quiter

Liste des intervenants

Ajouter intervenant

Entrez un nom d'intervenant

Chercher

Modifier

Supprimer

Retour

VI.7 Développement

Le langage de programmation orienté objet java, est le langage que j'ai utilisé pour coder le produit. Je me suis servi de Netbeans comme environnement de développement intégré pour le nombre important de fonctionnalités qu'il présente.

J'ai développé l'application avec l'architecture DAO (Data Access Object) car cet architecture permet de

_préserver la base de données en la séparant de l'implémentation.

_faire évoluer facilement le produit quand le besoin se fait sentir.

En ce qui concerne l'interface utilisateur j'ai utilisé swing et sa bibliothèque. J'ai opté pour un CardLayout en superposant les principales fenêtres, accessibles à partir des boutons du menu (PopupMenu). Des fenêtres de dialogue permettent l'enregistrement de certains items sans quitter la page.

Les JTables ont comme modèle AbstractTableModel.

La partie ORM (Mapping-Objet-relationnel) s'est fait manuellement en respectant le concept orienté objet, c'est-à-dire les types de relation entre les tables de la base de données.

Pour la partie persistance, j'ai implémenté la plupart des méthodes à travers des interfaces. Certaines des méthodes ont été implémentées directement dans les classes IHM, comme les JComboBoxes qui affichent des listes depuis une colonne des tables de la base de données. Les composants d'accès à la base donnée, avec JDBC (Jave DataBase Connectivity), passent par des PreparedStatement en se connectant grâce à la classe Singleton de connexion pour éviter des répétitions.

VI.8 Rédaction du manuel d'utilisation

Le manuel d'utilisation, annexé au document, est rédigé en deux formats :

_en format HTML, attaché à l'application et accessible depuis le bouton « Aide »

_en format PDF, livré avec le produit.

J'ai conçu le manuel avec un outil en version démo du nom de « Dr Explain ».

VI.9 Déploiement

A l'aide d'une machine virtuelle hébergée sur le serveur de la mairie, j'ai installé et configuré:

_Debian 6 comme système d'exploitation

_Apache 2

_PHP 5

_MySQL Server

_PhpMyAdmin

La base de données qui, jusqu'ici résidait sur le serveur WAMP, est transférée sur le serveur de la mairie, afin d'être accessible depuis le réseau local.

Etant donné que dans le parc informatique de la mairie et les postes de travail utilisent différents systèmes d'exploitation, l'application est livrée dans un format JAR (Java Archive). Sous ce format l'application est exécutable sur toutes les plateformes munies de java runtime. Je l'ai installé sur trois postes de travail :

_pour la secrétaire

_le responsable des services techniques

_le chargé des ressources humaines et de la comptabilité

Une copie du dossier regroupant l'application, les codes d'accès du serveur et le manuel d'utilisation est sauvegardée sur le serveur physique. Une autre copie remise au directeur des services municipaux.

VI.10 Formation des utilisateurs

Trois personnes ont reçu la formation à savoir :

_La secrétaire qui devait juste connaître la procédure pour enregistrer des interventions car son rôle se limite à cela, sur décision du Directeur des services municipaux.

_Le Responsable des services techniques, qui est le principal utilisateur de l'application. Il analyse les opérations effectuées pour programmer des

interventions à faire. L'ajout de nouveaux items fait également partie de son rôle. Avec lui il était question de faire le tour complet de l'application.

_Le Chargé des ressources humaines et de la comptabilité étant en vacances, j'ai formé son supérieur hiérarchique, le Directeur des services municipaux à sa place.

Grace à la simplicité du logiciel et à sa conception ergonomique, la formation s'est faite sans difficulté.

VII. Conclusion

Ce stage de fin d'étude a été une occasion pour moi de côtoyer la gestion d'un projet, le langage java et la vie en entreprise, mais avec plus de responsabilités.

Dans mon stage de fin d'année le travail a été fait au profit de la mairie de Corné. Dans ce cadre j'ai mis en place un système de gestion des interventions des services techniques.

Pour ce faire, j'ai commencé la description des besoins des futurs utilisateurs du produit. Pour la suite je me suis basé sur l'application existante afin de décider de la solution à proposer.

Donc pour ce développement spécifique, j'ai débuté par une analyse conceptuelle des éléments impliqués et par la suite j'ai développé l'ensemble des vues et des objets. Finalement j'ai implémenté les composants afin de répondre exactement aux besoins de la mairie.

Coté compétence, j'ai découvert beaucoup de choses et je sais aussi qu'il y a encore du chemin à faire sur java. En tout cas j'ai appris un tas de nouvelles possibilités qu'offrent les environnements de développement intégrés tels que Netbeans ou Eclipse et les frameworks comme Hibernate qui peuvent nous faire gagner beaucoup de temps. Ce fut également une occasion de découvrir la grandeur de la communauté de développeurs depuis tous les coins du globe (un tuto espagnol m'a sorti d'une impasse).

Comme perspective, je me suis engagé à améliorer les fonctionnalités de l'application auprès de la mairie, si on regarde la base de données il y a deux tables inutilisées (salaire et utilisateur). Je crois pouvoir, avec l'application,

calculer le coût global des interventions en tenant compte des revenus de chaque agent.

VIII Annexe

- Scripts de la base de données
- Rapport sur l'état du projet destiné à la MOA
- Manuel d'utilisation(en pièces jointes)
- Codes (en pièces jointes)

Script de création de la base de données généré par AnalyseSI

```
# use VOTRE_BASE_DE_DONNEE ;
```

```
DROP TABLE IF EXISTS Type_intervention ;
```

```
CREATE TABLE Type_intervention (code_type VARCHAR(10) NOT NULL,  
designation_type VARCHAR(80),  
code_categorie VARCHAR(10),  
PRIMARY KEY (code_type) ) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS Intervention ;
```

```
CREATE TABLE Intervention (id_intervention int AUTO_INCREMENT NOT NULL,  
nom_intervention VARCHAR(40),  
date_debut DATE,  
date_fin DATE,  
duree_heures TIME,  
code_type VARCHAR(10) NOT NULL,  
code_local VARCHAR(10) NOT NULL,  
code_equipe VARCHAR(10) NOT NULL,  
PRIMARY KEY (id_intervention) ) ENGINE=InnoDB;
```

DROP TABLE IF EXISTS Locaux ;

CREATE TABLE Locaux (code_local VARCHAR(10) NOT NULL,
nom_local VARCHAR(80),
code_site VARCHAR(10),
PRIMARY KEY (code_local)) ENGINE=InnoDB;

DROP TABLE IF EXISTS Produits ;

CREATE TABLE Produits (code_produit VARCHAR NOT NULL,
designation VARCHAR(40),
unite INT(20),
Fournisseur VARCHAR(80),
PRIMARY KEY (code_produit)) ENGINE=InnoDB;

DROP TABLE IF EXISTS Intervenants ;

CREATE TABLE Intervenants (code_intervenant VARCHAR(10) NOT NULL,
nom_intervenant VARCHAR(40),
prenom_intervenant VARCHAR(40),
fonction_intervenant VARCHAR(80),
Contact_intervenant VARCHAR(40),
Date_embauche DATE,
contrat VARCHAR(5),
terme_contrat DATE,
PRIMARY KEY (code_intervenant)) ENGINE=InnoDB;

DROP TABLE IF EXISTS Categories ;

CREATE TABLE Categories (code_categorie VARCHAR(10) NOT NULL,

```
nom_categorie VARCHAR(40),  
PRIMARY KEY (code_categorie) ) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS Sites ;  
  
CREATE TABLE Sites (code_site VARCHAR(10) NOT NULL,  
nom_site VARCHAR(80),  
localisation INT(80),  
PRIMARY KEY (code_site) ) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS Equipes ;  
  
CREATE TABLE Equipes (code_equipe VARCHAR(10) NOT NULL,  
nom_equipe VARCHAR(40),  
nbre_intervenant INTEGER,  
code_intervenant VARCHAR(10) NOT NULL,  
PRIMARY KEY (code_equipe) ) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS Salaire ;  
  
CREATE TABLE salaire (date_debut_sal DATE NOT NULL,  
salaire_h FLOAT,  
date_fin_sal DATE,  
code_intervenant VARCHAR(10) NOT NULL,  
PRIMARY KEY (date_debut_sal) ) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS Utilisation_produit ;  
  
CREATE TABLE utilisation_produit (id_utilisation int AUTO_INCREMENT NOT NULL,  
nom_utilisation VARCHAR(40),  
prix_unitaire FLOAT,
```

```
quantite FLOAT,  
id_intervention INT NOT NULL,  
code_produit VARCHAR NOT NULL,  
PRIMARY KEY (id_utilisation) ) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS Comprendre ;
```

```
CREATE TABLE Comprendre (code_equipe VARCHAR(10) NOT NULL,  
code_intervenant VARCHAR(10) NOT NULL,  
PRIMARY KEY (code_equipe,  
code_intervenant) ) ENGINE=InnoDB;
```

```
ALTER TABLE Type_intervention ADD CONSTRAINT  
FK_Type_intervention_code_categorie FOREIGN KEY (code_categorie) REFERENCES  
Categories (code_categorie);
```

```
ALTER TABLE Intervention ADD CONSTRAINT FK_Intervention_code_type FOREIGN  
KEY (code_type) REFERENCES Type_intervention (code_type);
```

```
ALTER TABLE Intervention ADD CONSTRAINT FK_Intervention_code_local FOREIGN  
KEY (code_local) REFERENCES Locaux (code_local);
```

```
ALTER TABLE Intervention ADD CONSTRAINT FK_Intervention_code_equipe FOREIGN  
KEY (code_equipe) REFERENCES Equipes (code_equipe);
```

```
ALTER TABLE Locaux ADD CONSTRAINT FK_Locaux_code_site FOREIGN KEY  
(code_site) REFERENCES Sites (code_site);
```

```
ALTER TABLE Equipes ADD CONSTRAINT FK_Equipes_code_intervenant FOREIGN  
KEY (code_intervenant) REFERENCES Intervenants (code_intervenant);
```

```
ALTER TABLE salaire ADD CONSTRAINT FK_salaire_code_intervenant FOREIGN KEY  
(code_intervenant) REFERENCES Intervenants (code_intervenant);
```

```
ALTER TABLE utilisation_produit ADD CONSTRAINT  
FK_utilisation_produit_id_intervention FOREIGN KEY (id_intervention) REFERENCES  
Intervention (id_intervention);
```

```
ALTER TABLE utilisation_produit ADD CONSTRAINT  
FK_utilisation_produit_code_produit FOREIGN KEY (code_produit) REFERENCES  
Produits (code_produit);
```

```
ALTER TABLE Comprendre ADD CONSTRAINT FK_Comprendre_code_equipe  
FOREIGN KEY (code_equipe) REFERENCES Equipes (code_equipe);
```

```
ALTER TABLE Comprendre ADD CONSTRAINT FK_Comprendre_code_intervenant  
FOREIGN KEY (code_intervenant) REFERENCES Intervenants (code_intervenant);
```

Script de sauvegarde de la base de données généré par PhpMyAdmin

```
-- phpMyAdmin SQL Dump
```

```
-- version 3.5.1
```

```
-- http://www.phpmyadmin.net
```

```
--
```

```
-- Client: localhost
```

```
-- Généré le: Mar 19 Août 2014 à 14:48
```

```
-- Version du serveur: 5.5.24-log
```

```
-- Version de PHP: 5.3.13
```

```
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
```

```
SET time_zone = "+00:00";
```

```
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT  
*/;
```

```
/*!40101 SET  
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
```

```
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION  
*/;
```

```
/*!40101 SET NAMES utf8 */;
```

--

-- Base de données: `smtc`

--

--

-- Structure de la table `categorie`

--

```
CREATE TABLE IF NOT EXISTS `categorie` (  
  `code_categorie` varchar(10) NOT NULL,  
  `nom_categorie` varchar(40) DEFAULT NULL,  
  PRIMARY KEY (`code_categorie`),  
  UNIQUE KEY `code_categorie` (`code_categorie`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

--

-- Contenu de la table `categorie`

--

```
INSERT INTO `categorie` (`code_categorie`, `nom_categorie`) VALUES  
( 'CAD', 'Catégorie tâches Administratives'),  
( 'CBT', 'Catégorie tâches Batiment'),  
( 'CDV', 'Catégorie tâches Diverses'),  
( 'CEV', 'Catégorie tâches Espace Vert'),  
( 'CVR', 'Catégorie tâches Voirie');
```

--

-- Structure de la table `comprendre`

--

```
CREATE TABLE IF NOT EXISTS `comprendre` (  
  `code_equipe` varchar(10) NOT NULL,  
  `code_intervenant` varchar(10) NOT NULL,  
  PRIMARY KEY (`code_equipe`,`code_intervenant`),  
  KEY `FK_Comprendre_code_intervenant` (`code_intervenant`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

--

-- Contenu de la table `comprendre`

--

```
INSERT INTO `comprendre` (`code_equipe`,`code_intervenant`) VALUES  
( 'GIB00', 'ABT01-PS'),  
( 'GIB01', 'ABT01-PS'),  
( 'GIB02', 'ABT01-PS'),  
( 'GIB02', 'ACE01-GO'),  
( 'GIV02', 'ACE01-GO'),  
( 'GIV03', 'ACE01-GO'),  
( 'GIEV00', 'AEV01-LFJ'),  
( 'GIEV01', 'AEV01-LFJ'),
```

```
('GIEV02', 'AEV01-LFJ'),
('GIEV03', 'AEV01-LFJ'),
('GIEV01', 'AEV02-HD'),
('GIEV02', 'AEV02-HD'),
('GIEV03', 'AEV02-HD'),
('GIEV02', 'AEV03-DM'),
('GIEV03', 'AEV03-DM'),
('GIB02', 'AEV04-MG'),
('GIEV03', 'AEV04-MG'),
('GIV03', 'AEV04-MG'),
('GIV00', 'AVR01-MG'),
('GIV01', 'AVR01-MG'),
('GIV02', 'AVR01-MG'),
('GIV03', 'AVR01-MG'),
('GIB01', 'AVR02-TC'),
('GIV01', 'AVR02-TC'),
('GIV02', 'AVR02-TC'),
('GIV03', 'AVR02-TC'),
('GIAD-00 ', 'STG00-MD');
```

--

-- Structure de la table `equipe`

--

```
CREATE TABLE IF NOT EXISTS `equipe` (
```

```
`code_equipe` varchar(10) NOT NULL,
`nom_equipe` varchar(40) DEFAULT NULL,
`nbre_intervenant` int(11) DEFAULT NULL,
`code_intervenant` varchar(10) NOT NULL,
PRIMARY KEY (`code_equipe`),
KEY `FK_Equipe_code_intervenant` (`code_intervenant`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Contenu de la table `equipe`
--

INSERT INTO `equipe` (`code_equipe`, `nom_equipe`, `nbre_intervenant`,
`code_intervenant`) VALUES
('GIAD-00 ', 'Groupe d"intervention administraif 0', 1, 'STG00-MD'),
('GIB00', 'Groupe d"intervention Bâtiment 0 ', 1, 'ABT01-PS'),
('GIB01', 'groupe intervention batiment 1 ', 2, 'ABT01-PS'),
('GIB02', 'groupe intervention batiment 2', 3, 'ABT01-PS'),
('GIEV00', 'Groupe d"intervention espaces verts 0', 1, 'AEV01-LFJ'),
('GIEV01', 'groupe intervention espaces verts 1', 2, 'AEV02-HD'),
('GIEV02', 'groupe intervention espaces verts 2', 3, 'AEV01-LFJ'),
('GIEV03', 'groupe intervention espaces verts 3', 4, 'AEV01-LFJ'),
('GIV00', 'Groupe d"intervention Voirie 0', 1, 'AVR01-MG'),
('GIV01', 'groupe intervention voirie 1', 2, 'AVR02-TC'),
('GIV02', 'groupe intervention voirie 2', 3, 'AVR01-MG'),
('GIV03', 'groupe intervention voirie 1', 4, 'AVR01-MG');
```

--

-- Doublure de structure pour la vue `fullteam`

--

CREATE TABLE IF NOT EXISTS `fullteam` (

`code_eqq` varchar(10)

, `nom` varchar(40)

, `nbre_mbres` int(11)

, `code_agent` varchar(10)

, `nom_agent` varchar(40)

, `fonction` varchar(80)

, `code_chef` varchar(10)

);

--

-- Structure de la table `intervenant`

--

CREATE TABLE IF NOT EXISTS `intervenant` (

`code_intervenant` varchar(10) NOT NULL,

`nom_intervenant` varchar(40) DEFAULT NULL,

`prenom_intervenant` varchar(40) DEFAULT NULL,

`fonction_intervenant` varchar(80) DEFAULT NULL,

`contact_intervenant` varchar(40) DEFAULT NULL,

`date_embauche` date DEFAULT NULL,

Dossier de projet de stage

```
`contrat` varchar(60) DEFAULT NULL,
`terme_contrat` date DEFAULT NULL,
PRIMARY KEY (`code_intervenant`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Contenu de la table `intervenant`
--

INSERT INTO `intervenant` (`code_intervenant`, `nom_intervenant`, `prenom_intervenant`,
`fonction_intervenant`, `contact_intervenant`, `date_embauche`, `contrat`, `terme_contrat`)
VALUES

('ABT01-PS', 'PECHERON', 'Serge', 'Gestion et maintenance Batiment', '0608037366', '2000-
10-01', 'CDI', '0000-00-00'),

('ACE01-GO', 'GRELET', 'Olivier', 'conducteur d"engins', '0608037344', '1992-01-09', 'CDI',
'0000-00-00'),

('AEV01-LFJ', 'LE FUR', 'Joël', 'Agent espaces verts', '0608037344', '1988-08-01', 'CDI',
'0000-00-00'),

('AEV02-HD', 'HAMMARD', 'Didier', 'Agent espaces verts', '0608037344', '2007-11-12',
'CDI', '0000-00-00'),

('AEV03-DM', 'DEBELLY', 'Michel', 'CDD emploi avenir espaces verts', '0608037344',
'2014-03-03', 'CDD', '0000-00-00'),

('AEV04-MG', 'MUNUERA', 'Geoffrey', 'CDD agent espaces verts', '0608037344', '2012-10-
22', 'CDD', '0000-00-00'),

('AEV05-BND', 'NDIAYE', 'Baye', 'ingenieur', ' ', '2014-07-01', '_', '2014-07-31'),

('AVR01-MG', 'MICHEL', 'Guy', 'Agent voirie', '0608037344', '2000-09-01', 'CDI', '0000-00-
00'),

('AVR02-TC', 'TRIGANNE', 'Cyril', 'Agent voirie', '0608037344', '2007-01-01', 'CDI', '0000-
00-00'),

('AVR05-PN ', 'NOIRET', 'Pierre', 'voirie', '0758236946', '2014-07-02', 'CDD', '2014-07-31'),
```

('RST-FP', 'FOUASSIER', 'Pascal', 'responsable services techniques', '0608037344', '2012-05-01', 'CDI', '0000-00-00'),

('STG00-MD', 'DIOP', 'Mamadou', 'Stagiaire', '0658719618', '2014-06-04', 'C.Stage', '2014-07-23');

--

-- Structure de la table `intervention`

--

```
CREATE TABLE IF NOT EXISTS `intervention` (  
  `id_intervention` int(11) NOT NULL AUTO_INCREMENT,  
  `nom_intervention` varchar(40) DEFAULT NULL,  
  `date` date DEFAULT NULL,  
  `heure_debut` time DEFAULT NULL,  
  `heure_fin` time DEFAULT NULL,  
  `code_type` varchar(10) NOT NULL,  
  `code_local` varchar(10) NOT NULL,  
  `code_equipe` varchar(10) NOT NULL,  
  PRIMARY KEY (`id_intervention`),  
  KEY `FK_Intervention_code_type` (`code_type`),  
  KEY `FK_Intervention_code_local` (`code_local`),  
  KEY `FK_Intervention_code_equipe` (`code_equipe`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=24 ;
```

--

-- Contenu de la table `intervention`

--

```
INSERT INTO `intervention` (`id_intervention`, `nom_intervention`, `date`, `heure_debut`,  
`heure_fin`, `code_type`, `code_local`, `code_equipe`) VALUES
```

```
(1, 'tonte', '2014-03-11', '08:00:00', '13:00:00', 'CEV-G00', 'DBM-CS00', 'GIEV02'),  
(2, 'curage', '2014-05-14', '08:00:00', '15:00:00', 'CVR-F04', 'DVR-VR11', 'GIV01'),  
(3, 'rebouchage fissures', '2013-09-10', '08:00:00', '16:00:00', 'CBT-T05', 'DBM-EG00',  
'GIB02'),  
(4, 'Revêtement routier', '2014-04-09', '08:00:00', '22:00:00', 'CVR-T04', 'DEV-ET32',  
'GIV03'),  
(5, 'Refection', '2014-06-02', '08:00:00', '16:00:00', 'CBT-T06', 'DBM-AH04', 'GIB02'),  
(10, 'Plantation', '2014-03-12', '10:00:00', '18:00:00', 'CEV-D00', 'DEV-PP01', 'GIEV03'),  
(12, 'intervention', '2013-06-14', '08:00:00', '12:00:00', 'CEV-D02', 'DEV-ET17', 'GIEV03'),  
(13, 'intervention1', '2014-03-18', '08:00:00', '17:00:00', 'CBT-T05', 'DBM-AH20', 'GIB02'),  
(14, 'intervention2', '2014-05-12', '08:00:00', '12:00:00', 'CBT-T08', 'DBM-AH12', 'GIB01'),  
(15, 'intervention3', '2014-05-29', '09:00:00', '16:00:00', 'CVR-S02', 'DVR-VR97', 'GIV01'),  
(16, 'intervention9', '2014-06-24', '10:00:00', '12:00:00', 'CVR-S00', 'DVR-VR97', 'GIV01'),  
(17, 'intervention4', '2014-06-02', '08:00:00', '16:00:00', 'CEV-D05', 'DEV-MA08', 'GIEV01'),  
(18, 'intervention5', '2014-06-04', '08:00:00', '12:00:00', 'CBT-T19', 'DBM-CS02', 'GIB01'),  
(19, 'intervention6', '2014-06-06', '08:00:00', '15:00:00', 'CEV-D00', 'DEV-MA03', 'GIEV01'),  
(20, 'intervention7', '2014-06-12', '10:00:00', '18:00:00', 'CBT-T00', 'DBM-AH00', 'GIB00'),  
(21, 'Intervention8', '2014-06-20', '08:00:00', '12:00:00', 'CDV-T02', 'DEV-ET25', 'GIEV02'),  
(23, 'Intervention11', '2014-07-03', '08:00:00', '16:00:00', 'CBT-T06', 'DBM-AH01', 'GIB00');
```

-- -----

--

-- Structure de la table `local`

--

```
CREATE TABLE IF NOT EXISTS `local` (  
  `code_local` varchar(10) NOT NULL,  
  `nom_local` varchar(80) DEFAULT NULL,  
  `code_site` varchar(10) DEFAULT NULL,  
  PRIMARY KEY (`code_local`),  
  KEY `FK_Local_code_site` (`code_site`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

--

-- Contenu de la table `local`

--

```
INSERT INTO `local` (`code_local`, `nom_local`, `code_site`) VALUES  
( 'DBM-AH00', 'site ALSH', 'DBM-AH'),  
( 'DBM-AH01', 'hall d"entrée', 'DBM-AH'),  
( 'DBM-AH02', 'escaliers', 'DBM-AH'),  
( 'DBM-AH03', 'grande salle', 'DBM-AH'),  
( 'DEV-ET01', 'aire de pique nique le long de l"Authion', 'DEV-ET'),  
( 'DEV-ET02', 'CSC les Moulins', 'DEV-ET'),  
( 'DVR-VR99', 'rue des Poètes', 'DVR-VR');
```

-- -----

--

-- Doublure de structure pour la vue `membre`

--

CREATE TABLE IF NOT EXISTS `membre` (

`Equipe` varchar(10)

, `Code` varchar(10)

, `Nom` varchar(40)

);

-- -----

--

-- Doublure de structure pour la vue `pretri`

--

CREATE TABLE IF NOT EXISTS `pretri` (

`Id` int(11)

, `Intervention` varchar(40)

, `Date` date

, `H_debut` time

, `H_fin` time

, `Tache` varchar(80)

, `Local` varchar(80)

, `Equipe` varchar(10)

, `NbreAgent` int(11)

);

-- -----

--

-- Structure de la table `produit`

--

```
CREATE TABLE IF NOT EXISTS `produit` (  
  `code_produit` varchar(10) NOT NULL,  
  `designation` varchar(60) DEFAULT NULL,  
  `unite` varchar(20) DEFAULT NULL,  
  `fournisseur` varchar(80) NOT NULL,  
  PRIMARY KEY (`code_produit`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
--  
-- Contenu de la table `produit`  
--  
  
INSERT INTO `produit` (`code_produit`, `designation`, `unite`, `fournisseur`) VALUES  
( 'ACD-CEL', 'Acide cellulosique', 'Litre', 'ZOLPAN'),  
( 'CIM-B', 'Ciment blanc', 'Kg', 'TERRENA'),  
( 'GRVR', 'Gravier', 'm3', 'POINT P'),  
( 'GZOIL', 'Gaz-oil', 'Litre', 'SUPER-U'),  
( 'PAP-P', 'Papier peint', 'm2', 'LE ROY MERLIN'),  
( 'PEINT-H', 'Peinture à huile', 'Kg', 'ZOLPAN'),  
( 'PELTSE', 'Pelleteuse', 'Heure', 'HEULIN ROUSSEAU'),  
( 'PLSE-SYN', 'Pelouse synthétique', 'm2', 'ECHO VERT'),  
( 'RLR', 'Roller ', 'Heure', 'AXIMUM'),  
( 'SBLE-M', 'Sable à maçonner', 'm3', 'TERRENA'),  
( 'UEV-AQX', 'Aquamix', 'RIPPERT', 'Litre'),  
( 'UEV-LSTE', 'La sente', 'RIPPERT', 'Kg'),  
( 'UEV-P10T', 'Puissance 10T', 'RIPPERT', 'Litre');
```

--

-- Structure de la table `salaire`

--

```
CREATE TABLE IF NOT EXISTS `salaire` (  
  `date_debut_sal` date NOT NULL,  
  `salaire_h` float DEFAULT NULL,  
  `date_fin_sal` date DEFAULT NULL,  
  `code_intervenant` varchar(10) NOT NULL,  
  PRIMARY KEY (`date_debut_sal`),  
  KEY `FK_Salaire_code_intervenant` (`code_intervenant`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

--

-- Contenu de la table `salaire`

--

```
INSERT INTO `salaire` (`date_debut_sal`, `salaire_h`, `date_fin_sal`, `code_intervenant`)  
VALUES  
  
('1988-08-01', 16.2, '2014-06-20', 'AEV01-LFJ'),  
('1992-01-09', 16.2, '2014-06-20', 'ACE01-GO'),  
('2000-09-01', 16.2, '2014-06-20', 'AVR01-MG'),  
('2000-10-01', 16.2, '2014-06-20', 'ABT01-PS'),  
('2007-01-01', 16.2, '2014-06-20', 'AVR02-TC'),
```

```
('2007-11-12', 16.2, '2014-06-20', 'AEV02-HD'),  
('2012-05-01', 20, '2014-06-20', 'RST-FP'),  
('2012-10-22', 16.2, '2014-06-20', 'AEV04-MG'),  
('2014-03-03', 16.2, '2014-06-20', 'AEV03-DM');
```

```
-- -----
```

```
--
```

```
-- Structure de la table `site`
```

```
--
```

```
CREATE TABLE IF NOT EXISTS `site` (  
  `code_site` varchar(10) NOT NULL,  
  `nom_site` varchar(80) DEFAULT NULL,  
  `localisation` varchar(80) DEFAULT NULL,  
  PRIMARY KEY (`code_site`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
--
```

```
-- Contenu de la table `site`
```

```
--
```

```
INSERT INTO `site` (`code_site`, `nom_site`, `localisation`) VALUES  
('DBM-31', 'batiment 31', '31, Rue Royale'),  
('DBM-37', 'batiment 37', '37,Rue Royale'),  
('DBM-39', 'batiment 39', '39,Rue Royale'),  
('DBM-54', 'batiment 54', '54,Rue Royale'),  
('DVR-VR', 'les voiries', '');
```

--

-- Doublure de structure pour la vue `team`

--

CREATE TABLE IF NOT EXISTS `team` (

`equipe` varchar(40)

, `nbre_mbres` int(11)

, `code_agent` varchar(10)

, `nom_agent` varchar(40)

, `fonction` varchar(80)

, `code_chef` varchar(10)

);

--

-- Doublure de structure pour la vue `timing`

--

CREATE TABLE IF NOT EXISTS `timing` (

`Intervention` varchar(40)

, `Date` date

, `Equipe` varchar(10)

, `Code` varchar(10)

, `Nom` varchar(40)

, `Duree` varchar(10)

);

--

-- Doublure de structure pour la vue `tri`

--

CREATE TABLE IF NOT EXISTS `tri` (

`Id` int(11)

,`Intervention` varchar(40)

,`Date` date

,`Duree` varchar(10)

,`Tache` varchar(80)

,`Local` varchar(80)

,`Equipe` varchar(10)

,`NbreAgt` int(11)

,`Produit` varchar(60)

,`Cout` double(19,2)

);

--

-- Structure de la table `type_intervention`

--

CREATE TABLE IF NOT EXISTS `type_intervention` (

`code_type` varchar(10) NOT NULL,

`designation_type` varchar(80) DEFAULT NULL,

`code_categorie` varchar(10) DEFAULT NULL,

```
PRIMARY KEY (`code_type`),
KEY `FK_Type_intervention_code_categorie` (`code_categorie`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Contenu de la table `type_intervention`
--

INSERT INTO `type_intervention` (`code_type`, `designation_type`, `code_categorie`)
VALUES

('CAD-E01', 'absences', 'CAD'),
('CAD-E02', 'formations', 'CAD'),
('CAD-E03', 'réunions', 'CAD'),
('CAD-E04', 'syndicat', 'CAD'),
('CBT-T00', 'batiment-divers', 'CBT'),
('CBT-T01', 'controle/surveillance batiment municipaux', 'CBT'),
('CBT-T02', 'controle travaux entreprises', 'CBT'),
('CBT-T03', 'maintenance plomberie', 'CBT'),
('CBT-T04', 'maintenance électricité', 'CBT'),
('CBT-T05', 'maintenance maçonnerie', 'CBT'),
('CBT-T06', 'maintenance peinture', 'CBT'),
('CBT-T07', 'maintenance plâtrerie', 'CBT'),
('CBT-T08', 'maintenance serrurerie', 'CBT'),
('CBT-T09', 'maintenance sécurité', 'CBT'),
('CBT-T10', 'affichage', 'CBT'),
('CBT-T11', 'appro/fourniture ST', 'CBT'),
('CBT-T12', 'appro/fourniture mairie', 'CBT'),
```

('CBT-T13', 'déplacements administratifs', 'CBT'),
('CBT-T14', 'préparation manifestation', 'CBT'),
('CBT-T15', 'préparation OMA', 'CBT'),
('CBT-T16', 'états des lieux', 'CBT'),
('CBT-T17', 'conteneur OM', 'CBT'),
('CBT-T18', 'récupération animeaux divaguant', 'CBT'),
('CBT-T19', 'visite batiments communaux futur occupants', 'CBT'),
('CBT-T20', 'préparation salles', 'CBT'),
('CBT-T21', 'formalités location/occupation salles', 'CBT'),
('CBT-T22', 'controles techniques matériel/installations', 'CBT'),
('CDV-T01', 'entretien locaux', 'CDV'),
('CDV-T02', 'entretien matériel', 'CDV'),
('CDV-T03', 'préparation matériel', 'CDV'),
('CEV--G01', 'gazon tontes', 'CEV'),
('CEV--G06', 'gazon réparation', 'CEV'),
('CEV--T02', 'création espaces/massifs', 'CEV'),
('CEV-D00', 'désherbage-divers', 'CEV'),
('CEV-D01', 'déshserbage manuel', 'CEV'),
('CEV-D02', 'désherbage mécanique', 'CEV'),
('CEV-D03', 'désherbage thermique', 'CEV'),
('CEV-D04', 'désherbage selectif gazon', 'CEV'),
('CEV-D05', 'désherbage selectif espaces verts', 'CEV'),
('CEV-D06', 'désherbage phyto', 'CEV'),
('CEV-D07', 'broyage prairie', 'CEV'),
('CEV-G00', 'gazon-divers', 'CEV'),
('CEV-G02', 'gazon fertilisation', 'CEV'),
('CEV-G03', 'gazon regarnissage', 'CEV'),

('CEV-G04', 'gazon aération', 'CEV'),
('CEV-G05', 'gazon scarification', 'CEV'),
('CEV-G07', 'gazon création', 'CEV'),
('CEV-G08', 'gazon arrosage', 'CEV'),
('CEV-T00', 'espaces verts-divers', 'CEV'),
('CEV-T01', 'plantation', 'CEV'),
('CEV-T03', 'entretien massifs floraux', 'CEV'),
('CEV-T04', 'entretien massifs d"arbustes', 'CEV'),
('CEV-T05', 'préparation/entretien jardin', 'CEV'),
('CEV-T06', 'taille arbres', 'CEV'),
('CEV-T07', 'taille arbustes', 'CEV'),
('CEV-T08', 'taille haies', 'CEV'),
('CEV-T09', 'arrosage citerne', 'CEV'),
('CEV-T10', 'arrosage intégré', 'CEV'),
('CVR--S04', 'panneaux info', 'CVR'),
('CVR--TI2', 'autres intempéries', 'CVR'),
('CVR-F00', 'fossés-divers', 'CVR'),
('CVR-F01', 'épareuse', 'CVR'),
('CVR-F02', 'broyage bermes', 'CVR'),
('CVR-F03', 'curage mécanique', 'CVR'),
('CVR-F04', 'curage manuel', 'CVR'),
('CVR-F05', 'busage', 'CVR'),
('CVR-RP00', 'réseau pluvial-divers', 'CVR'),
('CVR-RP01', 'resceller grilles/tampons', 'CVR'),
('CVR-RP02', 'nettoyage avaloires/grilles', 'CVR'),
('CVR-RP03', 'gargouilles', 'CVR'),
('CVR-RP04', 'entretien réseau pluvial', 'CVR'),

('CVR-RP05', 'controle conformité', 'CVR'),
('CVR-S00', 'signalisation-divers', 'CVR'),
('CVR-S01', 'marquage résine', 'CVR'),
('CVR-S02', 'marquage peinture', 'CVR'),
('CVR-S03', 'panneaux police', 'CVR'),
('CVR-S05', 'poteaux', 'CVR'),
('CVR-S06', 'signalisations temporaires', 'CVR'),
('CVR-S07', 'controle arrêté municipal', 'CVR'),
('CVR-T00', 'voirie divers', 'CVR'),
('CVR-T01', 'enrobés à chaud', 'CVR'),
('CVR-T02', 'enrobés à froid', 'CVR'),
('CVR-T03', 'bouchage nids de poule', 'CVR'),
('CVR-T04', 'reprise pavage/dallage', 'CVR'),
('CVR-T05', 'sable de carrière', 'CVR'),
('CVR-T06', 'travaux bordures', 'CVR'),
('CVR-T07', 'meublier urbain', 'CVR'),
('CVR-T08', 'ramassage feuilles mortes', 'CVR'),
('CVR-T09', 'balayage manuel', 'CVR'),
('CVR-T10', 'balayage mécanique', 'CVR'),
('CVR-T11', 'lavage mécanique', 'CVR'),
('CVR-T12', 'ramassage ordures dépôts sauvages', 'CVR'),
('CVR-T13', 'vidage poubelle', 'CVR'),
('CVR-TH1', 'dénivellement', 'CVR'),
('CVR-TH2', 'salage', 'CVR'),
('CVR-TI1', 'inondations', 'CVR');

--

-- Structure de la table `utilisateur`

--

```
CREATE TABLE IF NOT EXISTS `utilisateur` (  
  `username` varchar(25) NOT NULL,  
  `password` varchar(20) NOT NULL,  
  PRIMARY KEY (`username`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

--

-- Doublure de structure pour la vue `utilisation`

--

```
CREATE TABLE IF NOT EXISTS `utilisation` (  
  `Produit` varchar(60)  
, `Code` varchar(10)  
, `Fournisseur` varchar(80)  
, `Utilisation` varchar(80)  
, `Intervention` varchar(40)  
, `Id` int(11)  
, `Date` varchar(8)  
, `Duree` varchar(10)  
, `Qte` float  
, `Prix_U` double(19,2)
```

```
,`Prix_T` double(19,2)
```

```
);
```

```
-- -----
```

```
--
```

```
-- Structure de la table `utilisation_produit`
```

```
--
```

```
CREATE TABLE IF NOT EXISTS `utilisation_produit` (
```

```
`id_utilisation` int(11) NOT NULL AUTO_INCREMENT,
```

```
`nom_utilisation` varchar(80) NOT NULL,
```

```
`prix_unitaire` float DEFAULT NULL,
```

```
`quantite` float DEFAULT NULL,
```

```
`id_intervention` int(11) NOT NULL,
```

```
`code_produit` varchar(10) NOT NULL,
```

```
PRIMARY KEY (`id_utilisation`),
```

```
KEY `FK_Utilisation_produit_id_intervention` (`id_intervention`),
```

```
KEY `code_produit` (`code_produit`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=13 ;
```

```
--
```

```
-- Contenu de la table `utilisation_produit`
```

```
--
```

```
INSERT INTO `utilisation_produit` (`id_utilisation`, `nom_utilisation`, `prix_unitaire`,  
`quantite`, `id_intervention`, `code_produit`) VALUES
```

```
(1, 'Diluant peinture', 2.5, 1, 3, 'ACD-CEL'),
```

(2, 'Réparation murs', 1.8, 5, 3, 'CIM-B'),
(3, 'Maçonnerie', 22, 7, 4, 'GRVR'),
(4, 'Pelleteuse', 1.67, 6, 4, 'GZOIL'),
(5, 'Revêtement', 15, 5, 4, 'PELTSE'),
(6, 'Espace vert', 8.99, 50, 1, 'PLSE-SYN'),
(8, 'maçonnerie', 2.5, 2, 14, 'CIM-B'),
(9, 'taille', 23, 1, 19, 'PLSE-SYN'),
(10, 'ballayage', 2.63, 1, 20, 'ACD-CEL'),
(11, 'debroussaille', 2.5, 1, 21, 'SBLE-M'),
(12, 'Detergent', 0.9, 2, 11, 'ACD-CEL');

--

-- Structure de la vue `fullteam`

--

DROP TABLE IF EXISTS `fullteam`;

```
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY
DEFINER VIEW `fullteam` AS select `c`.`code_equipe` AS `code_eqq`,`eq`.`nom_equipe`
AS `nom`,`eq`.`nbre_intervenant` AS `nbre_mbres`,`c`.`code_intervenant` AS
`code_agent`,`intt`.`nom_intervenant` AS `nom_agent`,`intt`.`fonction_intervenant` AS
`fonction`,`eq`.`code_intervenant` AS `code_chef` from ((`comprendre` `c` join `equipe` `eq`)
join `intervenant` `intt`) where ((`c`.`code_equipe` = `eq`.`code_equipe`) and
(`c`.`code_intervenant` = `intt`.`code_intervenant`)) order by `c`.`code_equipe`;
```

--

-- Structure de la vue `membre`

--

DROP TABLE IF EXISTS `membre`;

```
CREATE ALGORITHM=MERGE DEFINER=`admin`@`%` SQL SECURITY DEFINER
VIEW `membre` AS select `c`.`code_equipe` AS `Equipe`,`c`.`code_intervenant` AS
`Code`,`i`.`nom_intervenant` AS `Nom` from (`comprendre` `c` join `intervenant` `i`
on((`c`.`code_intervenant` = `i`.`code_intervenant`))) where `c`.`code_intervenant` in (select
`intervenant`.`code_intervenant` from `intervenant`) WITH CASCADED CHECK OPTION;
```

--

-- Structure de la vue `pretri`

--

DROP TABLE IF EXISTS `pretri`;

```
CREATE ALGORITHM=MERGE DEFINER=`root`@`localhost` SQL SECURITY
INVOKER VIEW `pretri` AS select `i`.`id_intervention` AS `Id`,`i`.`nom_intervention` AS
`Intervention`,`i`.`date` AS `Date`,`i`.`heure_debut` AS `H_debut`,`i`.`heure_fin` AS
`H_fin`,`t`.`designation_type` AS `Tache`,`l`.`nom_local` AS `Local`,`i`.`code_equipe` AS
`Equipe`,`e`.`nbre_intervenant` AS `NbreAgent` from (((`intervention` `i` join
`type_intervention` `t`) join `local` `l`) join `equipe` `e`) where ((`i`.`code_type` =
`t`.`code_type`) and (`i`.`code_local` = `l`.`code_local`) and (`i`.`code_equipe` =
`e`.`code_equipe`)) order by `i`.`date` WITH CASCADED CHECK OPTION;
```

--

-- Structure de la vue `team`

--

Dossier de projet de stage

```
DROP TABLE IF EXISTS `team`;
```

```
CREATE ALGORITHM=MERGE DEFINER=`admin`@`%` SQL SECURITY DEFINER
VIEW `team` AS select `eq`.`nom_equipe` AS `equipe`,`eq`.`nbre_intervenant` AS
`nbre_mbres`,`c`.`code_intervenant` AS `code_agent`,`intt`.`nom_intervenant` AS
`nom_agent`,`intt`.`fonction_intervenant` AS `fonction`,`eq`.`code_intervenant` AS
`code_chef` from ((`comprendre` `c` join `equipe` `eq`) join `intervenant` `intt`) where
((`c`.`code_equipe` = `eq`.`code_equipe`) and (`c`.`code_intervenant` =
`intt`.`code_intervenant`)) WITH CASCADED CHECK OPTION;
```

```
-- -----
```

```
--
```

```
-- Structure de la vue `timing`
```

```
--
```

```
DROP TABLE IF EXISTS `timing`;
```

```
CREATE ALGORITHM=UNDEFINED DEFINER=`admin`@`%` SQL SECURITY
DEFINER VIEW `timing` AS select distinct `t`.`Intervention` AS `Intervention`,`t`.`Date` AS
`Date`,`m`.`Equipe` AS `Equipe`,`m`.`Code` AS `Code`,`m`.`Nom` AS `Nom`,`t`.`Duree` AS
`Duree` from (`tri` `t` join `membre` `m` on((convert(`t`.`Equipe` using utf8mb4) =
convert(`m`.`Equipe` using utf8mb4))));
```

```
-- -----
```

```
--
```

```
-- Structure de la vue `tri`
```

```
--
```

```
DROP TABLE IF EXISTS `tri`;
```

Dossier de projet de stage

```
CREATE ALGORITHM=MERGE DEFINER=`admin`@`%` SQL SECURITY DEFINER
VIEW `tri` AS select `p`.`Id` AS `Id`,`p`.`Intervention` AS `Intervention`,`p`.`Date` AS
`Date`,time_format((`p`.`H_fin` - `p`.`H_debut`),'%H:%i') AS `Duree`,`p`.`Tache` AS
`Tache`,`p`.`Local` AS `Local`,`p`.`Equipe` AS `Equipe`,`p`.`NbreAgent` AS
`NbreAgt`,`u`.`Produit` AS `Produit`,`u`.`Prix_T` AS `Cout` from (`pretri` `p` left join
`utilisation` `u` on((`p`.`Id` = `u`.`Id`))) order by `p`.`Date`;
```

```
-- -----
```

```
--
```

```
-- Structure de la vue `utilisation`
```

```
--
```

```
DROP TABLE IF EXISTS `utilisation`;
```

```
CREATE ALGORITHM=MERGE DEFINER=`root`@`localhost` SQL SECURITY
DEFINER VIEW `utilisation` AS select `p`.`designation` AS `Produit`,`p`.`code_produit` AS
`Code`,`p`.`fournisseur` AS `Fournisseur`,`u`.`nom_utilisation` AS
`Utilisation`,`i`.`nom_intervention` AS `Intervention`,`i`.`id_intervention` AS
`Id`,date_format(`i`.`date`,`%d-%m-%y`) AS `Date`,time_format((`i`.`heure_fin` -
`i`.`heure_debut`),'%H:%i') AS `Duree`,`u`.`quantite` AS `Qte`,round(`u`.`prix_unitaire`,2)
AS `Prix_U`,round((`u`.`quantite` * `u`.`prix_unitaire`),2) AS `Prix_T` from ((`produit` `p`
join `utilisation_produit` `u`) join `intervention` `i`) where ((`p`.`code_produit` =
`u`.`code_produit`) and (`i`.`id_intervention` = `u`.`id_intervention`)) order by `i`.`date`
WITH CASCADED CHECK OPTION;
```

```
--
```

```
-- Contraintes pour les tables exportées
```

```
--
```

```
--
```

```
-- Contraintes pour la table `comprendre`
```

```
--
```



```
ALTER TABLE `comprendre`
```

```
    ADD CONSTRAINT `FK_Comprendre_code_equipe` FOREIGN KEY (`code_equipe`)
REFERENCES `equipe` (`code_equipe`),
```

```
    ADD CONSTRAINT `FK_Comprendre_code_intervenant` FOREIGN KEY
(`code_intervenant`) REFERENCES `intervenant` (`code_intervenant`);
```

```
--
```

```
-- Contraintes pour la table `equipe`
```

```
--
```

```
ALTER TABLE `equipe`
```

```
    ADD CONSTRAINT `FK_Equipe_code_intervenant` FOREIGN KEY (`code_intervenant`)
REFERENCES `intervenant` (`code_intervenant`);
```

```
--
```

```
-- Contraintes pour la table `intervention`
```

```
--
```

```
ALTER TABLE `intervention`
```

```
    ADD CONSTRAINT `FK_Intervention_code_equipe` FOREIGN KEY (`code_equipe`)
REFERENCES `equipe` (`code_equipe`),
```

```
    ADD CONSTRAINT `FK_Intervention_code_local` FOREIGN KEY (`code_local`)
REFERENCES `local` (`code_local`),
```

```
    ADD CONSTRAINT `FK_Intervention_code_type` FOREIGN KEY (`code_type`)
REFERENCES `type_intervention` (`code_type`);
```

```
--
```

```
-- Contraintes pour la table `local`
```

```
--
```

```
ALTER TABLE `local`
```

```
    ADD CONSTRAINT `FK_Local_code_site` FOREIGN KEY (`code_site`) REFERENCES
`site` (`code_site`);
```

--

-- Contraintes pour la table `salaire`

--

ALTER TABLE `salaire`

ADD CONSTRAINT `FK_Salaire_code_intervenant` FOREIGN KEY (`code_intervenant`)
REFERENCES `intervenant` (`code_intervenant`);

--

-- Contraintes pour la table `utilisation_produit`

--

ALTER TABLE `utilisation_produit`

ADD CONSTRAINT `FK_Utilisation_produit_id_intervention` FOREIGN KEY
(`id_intervention`) REFERENCES `intervention` (`id_intervention`),

ADD CONSTRAINT `utilisation_produit_ibfk_1` FOREIGN KEY (`code_produit`)
REFERENCES `produit` (`code_produit`);

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;

/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;

/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

Script des jeux d'essai

SELECT i.id_intervention, i.nom_intervention, i.date, i.heure_debut, i.heure_fin,
t.designation_type, l.nom_local, i.code_equipe, e.nb_intervenant

FROM intervention i, type_intervention t, local l, equipe e

WHERE i.code_type = t.code_type and i.code_local = l.code_local and i.code_equipe =
e.code_equipe

ORDER BY i.date ;

Requette utilisation: Produit, Code, Fournisseur, Utilisation, Intervention, Id, Date, Duree, Quantite, Prix_U, Prix_T

```
SELECT p.designation, p.code_produit, p.fournisseur, u.nom_utilisation, i.nom_intervention,
i.id_intervention, DATE_FORMAT(i.date, '%d-%m-%y')AS Date,
TIME_FORMAT(i.heure_fin - i.heure_debut, '%H:%i') As Duree, u.quantite,
round(u.prix_unitaire,2)AS Prix_U, round((u.quantite * u.prix_unitaire),2)AS Prix_T

FROM produit p, utilisation_produit u, intervention i

WHERE p.code_produit = u.code_produit and i.id_intervention = u.id_intervention

ORDER BY i.date
```

```
SELECT Intervention, Debut, Fin, Duree, Tache, Local, Equipe, NbreAgent, Prix_T

FROM pretri p, utilisation u

WHERE p.Id= u.Id and Debut between ?and ?

ORDER BY Debut ;
```

```
WHERE Debut BETWEEN '20130101' AND '20140101 23:59:59';

Id, Intervention, Date, Duree, Tache, Local, Equipe, NbreAgt, Produit, Cout
```

```
SELECT p.Id, p.Intervention, Date, TIME_FORMAT(p.H_fin - p.H_debut, '%H:%i') As
Duree, p.Tache, p.Local, p.Equipe, p.NbreAgent, u.Produit, u.Prix_T

FROM pretri p

JOIN utilisation u ON p.Id = u.Id

WHERE p.Id = u.Id

ORDER BY p.Date;
```

TRI SUM:

Dossier de projet de stage

```
SELECT Id, Intervention, Debut, Fin, Duree, Tache, Local, Equipe, NbreAgent, sum(Cout)
FROM tri
WHERE Debut BETWEEN '20130101' AND '20140707'
GROUP BY Id
ORDER BY Debut
```

```
SELECT Id, Intervention, Debut, Fin, Duree, Tache, Local, Equipe, NbreAgent, sum(Cout)
14632506
```

```
FROM tri
WHERE Equipe = 'GIV03' AND Debut BETWEEN '20130101' AND '20140707'
```

```
GROUP BY Id
ORDER BY Debut
```

<I20:11/721566788+°ili52305:23 15/07/2014+-- - - - - ///

```
SELECT l.code_categorie, count(Id), sum(Duree), sum(Cout)
FROM tri t JOIN local l ON t.Local = l.nom-local
WHERE l.code_categorie = ? AND Debut BETWEEN ?AND ?;
```

```
SELECT ty.code_categorie, count(Id), sum(Duree), sum(Cout)
FROM tri t JOIN type_intervention ty ON t.Tache = ty.designation_type
WHERE ty.code_categorie = 'CBT' AND Debut BETWEEN '20130101' AND '20140707';
```

```
SELECT ty.code_categorie, count( DISTINCT Id) AS Id, Duree, sum(Cout) AS Cout FROM
tri t
JOIN type_intervention ty ON t.Tache = ty.designation_type
WHERE ty.code_categorie = 'CBT' AND Debut BETWEEN '20130101' AND '20140707'
GROUP BY Id
```

```
SELECT ty.code_categorie, count( DISTINCT Id) AS Id, Duree, sum(Cout) AS Cout FROM
tri t

JOIN type_intervention ty ON t.Tache = ty.designation_type

WHERE Debut BETWEEN '20130101' AND '20140707'

GROUP BY Id
```

FINAL EVALUATION CAT:

```
SELECT Code, SUM(Id) AS NbreInt, SUM(Duree) AS Duree, Cout
FROM (SELECT DISTINCT Intervention, Debut, ty.code_categorie AS Code, count(
DISTINCT Id) AS Id,
Duree, sum(Cout) AS Cout FROM tri t JOIN type_intervention ty ON t.Tache =
ty.designation_type GROUP BY Id) tempo
WHERE Code = 'CBT' AND Debut BETWEEN '20130101' AND '20140707'
```

EVAL SITE:

```
SELECT Code, SUM(Id) AS NbreInt, SUM(Duree) AS Duree, Cout FROM (SELECT
DISTINCT Intervention, Debut, l.code_site AS Code, count( DISTINCT Id) AS Id, Duree,
sum(Cout)
AS Cout FROM tri t JOIN local l ON t.Local = l.nom_local GROUP BY Id) tempo
WHERE Code = 'DBM-EG' AND Debut BETWEEN '20130101' AND '20140707'
```

EVAL EQUIPE:

```
SELECT Code, Nom, SUM(Id) AS NbreInt, SUM(Duree) AS Duree, SUM(Cout) AS Cout
FROM(SELECT DISTINCT Intervention, Debut, e.code_equipe AS Code, i.nom_intervenant
AS Nom, count( DISTINCT Id) AS Id, Duree, sum(Cout)
AS Cout FROM tri t JOIN equipe e ON t.Equipe = e.code_equipe
JOIN intervenant i ON i.code_intervenant = e.code_intervenant
GROUP BY Id) tempo
```

WHERE Code = 'GIV03' AND Debut BETWEEN '20130101' AND '20140707'

SELECT Code, Nom, Team

FROM (SELECT co.code_intervenant AS Code, nom_intervenant AS Nom, code_equipe AS Team FROM comprendre co

JOIN intervenant i ON i.code_intervenant = co.code_intervenant

WHERE co.code_intervenant = 'ABT01-PS')

SELECT * from comprendre

WHERE code_intervenant IN (SELECT code_intervenant from intervenant);

SELECT DISTINCT Intervention, Date, m.Equipe, m.Code, m.Nom, Duree

FROM tri t

JOIN membre m ON t.Equipe = m.Equipe

where Nom = 'PECHERON';

SELECT Nom, SUM(Duree) AS Duree FROM timing WHERE Nom = 'MUNUERA' AND Date BETWEEN '20130101' AND '20140707';

Rapport de l'état du projet pour la MOA

Projet de Gestion des Activités des Services Techniques

Sommaire :

Introduction

A_Le projet

B_La planification

B1_Structure de découpage du projet
B2_Organigramme des tâches
B3_Le diagramme de Gantt
C_État des réalisations
C1_Présentation MCD
C2_Présentation du MPD
C3_Création de de la SGBDR
Récupération des données
Conclusion partielle

● Introduction

Dans le souci d'un travail clair et bien structuré, voici le dossier des analyses fonctionnelles et techniques du projet. Ce dossier a en outre comme objectif de spécifier aux responsables concernés et ceci dans un langage moins informatique ou du moins expliqué en conséquences, ce qu'on a fait, ce qu'on est en train de faire et ce qu'on compte faire durant la période du stage. En bref ce document contient les différentes analyses du projet, le découpage des tâches, le calendrier de déroulement et des jalons des livrables, entre autres la nouvelle structuration de la base de données des services techniques.

La rédaction de ce document a été facilitée par l'aide remarquable et la disponibilité du personnel de l'administration notamment le responsables des services techniques, le chargé des ressources humaines et le directeur général des services.

A) Le projet

Le projet consiste à développer une application informatique pour l'analyse et le suivi de la gestion des interventions réalisées par les services techniques communaux de Corné.

Cette application doit permettre :

- La création de nouvelles entrées dans la structure
- L'enregistrement des différentes interventions
- L'évaluation des temps et des coûts des interventions

Ces points sont convoités dans l'optique d'une utilisation comptable de meilleure qualité, en remplacement du système de gestion actuel jugé obsolète.

B) La planification

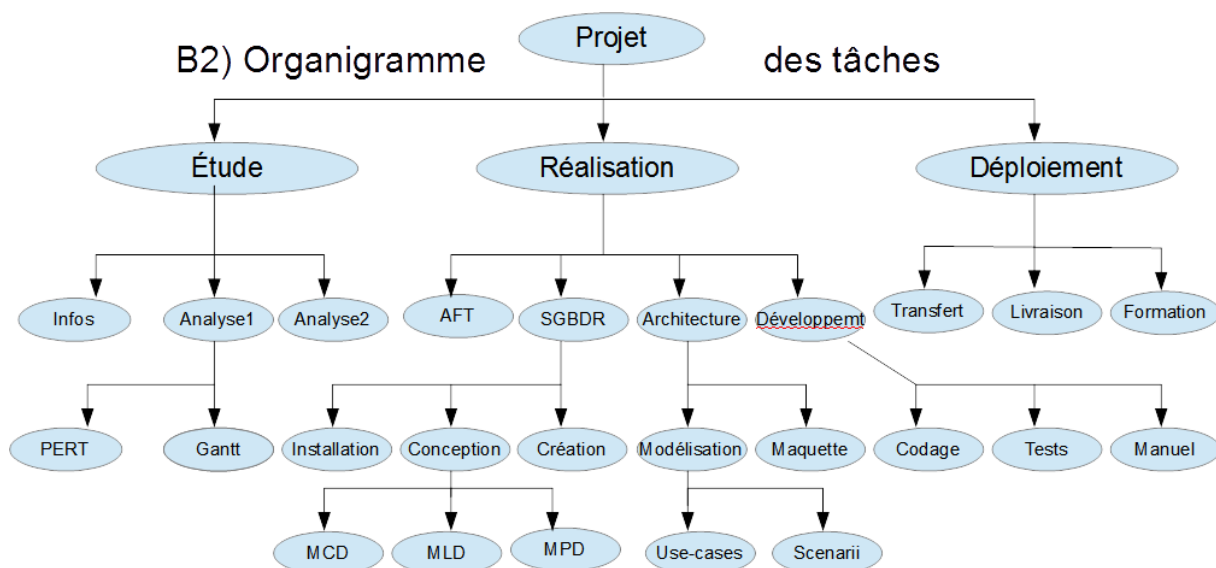
Une bonne gestion de projet se doit d'être bien planifiée et pour une bonne planification, il faut bien déterminer les besoins de la maîtrise d'ouvrage.

On s'est rapproché des responsables concernés pour recueillir des informations, poser des questions sur leurs attentes sur le produit et sur le fonctionnement effectif des services techniques et bien sûr proposer des solutions fonctionnelles.

Après avoir recueillis les informations nécessaires nous avons procédé à une planification avec une SDP (Structure de Découpage de Projet) que voici :

B1) SDP (Structure de Découpage du Projet) :

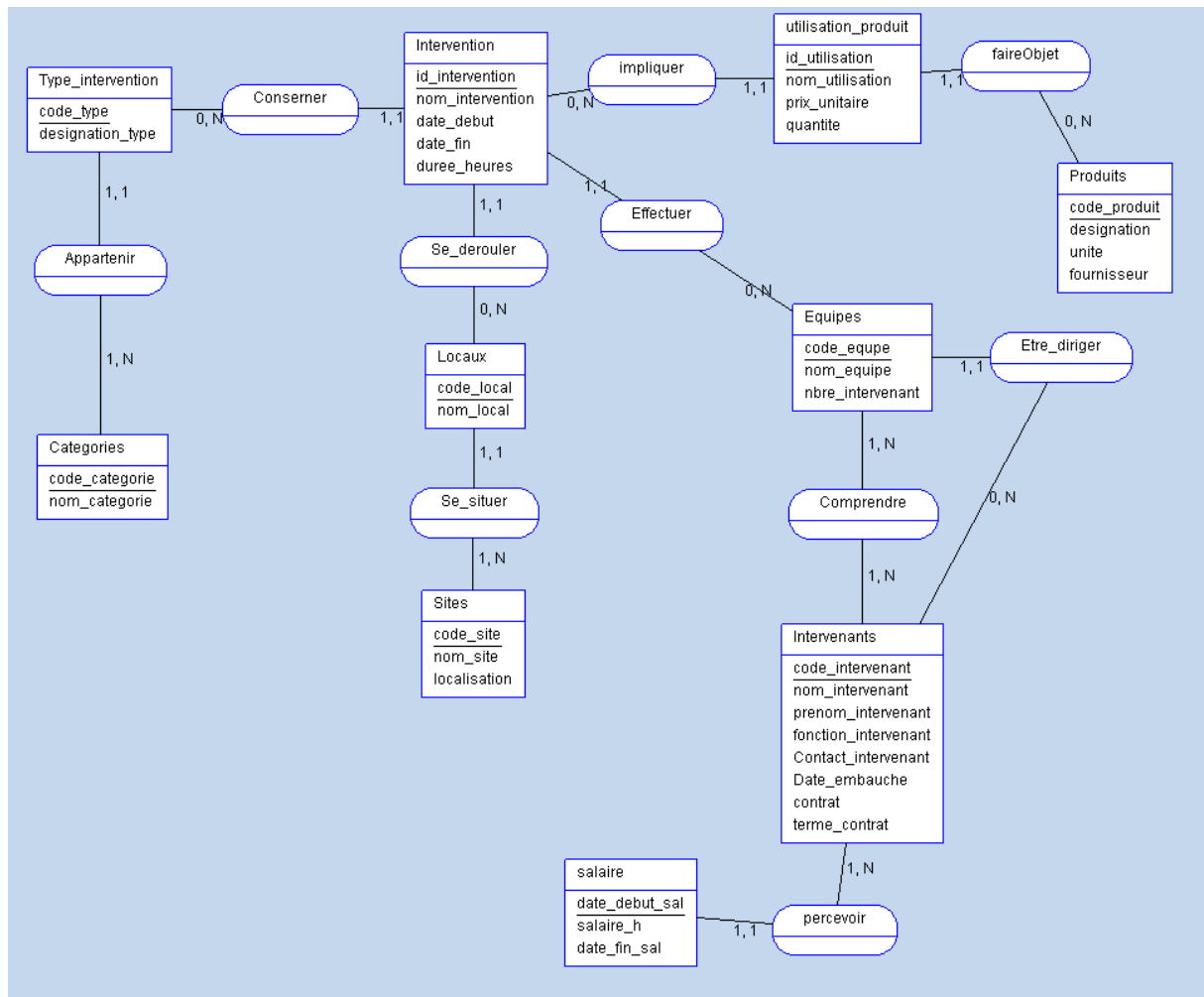
- Rencontre avec les responsables chargés de suivre le projet dans le but de définir les différents besoins du projet.
- Analyse des besoins dans le cadre de la planification du projet.
- Analyses fonctionnelles et techniques, proposées aux responsables et qu'ils ont validé.
- Installation des outils de conception de la base de données (AnalyseSI et MySQL Workbench) sur le poste de travail.
- Conception de la base de données : MCD (Modèle Conceptuel de Données), MLD (Modèle Logique de Données), MPD (Modèle Physique de Données).
- Installation d'un serveur virtuel et création de la base de données
- Entrée des données existantes dans la nouvelle base.
- Architecture de l'application sous forme de cas d'utilisations et de maquette.
- Installation d'un environnement de développement Java SE.
- Développement des Interfaces Homme Machine (IHM) sous Swing.
- Présentation des IHM à valider ou à corriger par les responsables
- Connexion à la base de données sous Wamp-server avec design pattern DAO.
- Implémentation des fonctionnalités et les requêtes SQL.
- Tests unitaires, rédaction du manuel d'utilisation.
- Déploiement de l'application et transfert de la base sur le serveur.
- Formation des utilisateurs.

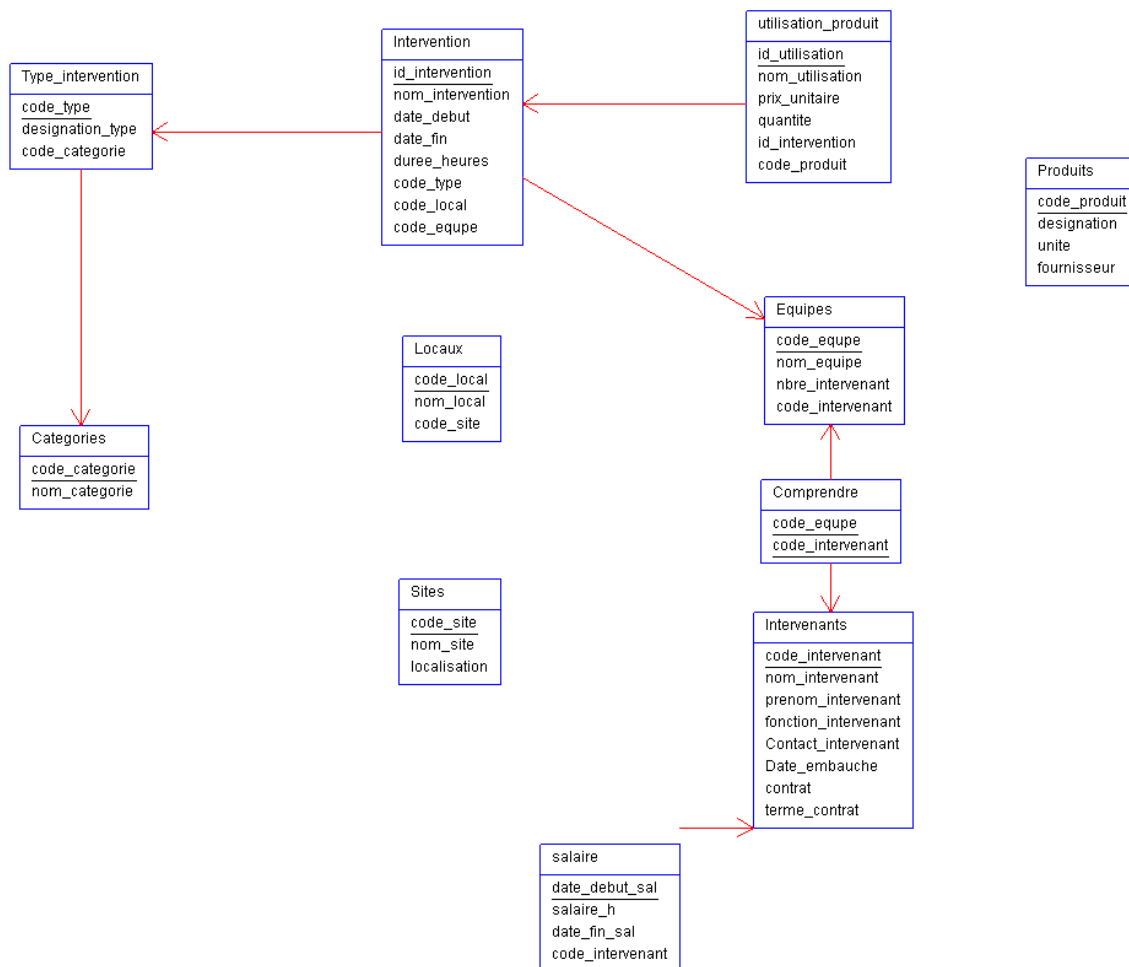


● C) État des réalisations

En ce quatrième jour de stage, on peut dire que qu'on est en avance sur les temps, mais c'est surtout un moyen de revoir et d'améliorer notre travail. On a fait le recueil des besoins, les analyses fonctionnelles et techniques, une réflexion sur la base de données, on a même déjà créé une base mais cette dernière pourra subir des modifications avec l'avancement du projet, car de nouveaux besoins peuvent voir le jour à tout moment.

Voici ce dessous une représentation de la structuration de la nouvelle base de données :





● C3) Création SGBDR

Pour la création de la base de données, on a eu besoin d'installer un serveur local sur le poste de travail. On a utilisé Wamp-server à cause de son interface phpMyAdmin où on peut créer et présenter des vues compréhensibles de tous.

● D) Récupération des données de la base existantes

La base est créée et on est en train de la peupler avec les données reçues du responsable des services techniques. A l'instar de la demande en matière de perfection on a entré de nouvelles informations dont on ira recueillir dans les services respectifs, comme les coûts des ressources pour la partie comptabilité.

● Conclusion partielle

Dans l'espoir que ce document ait été utile, on a cherché d'appliquer certains outils de gestion de projets en relation à notre formation en centre.

Normalement avec notre modeste niveau on n'est pas affronté directement à la gestion d'un projet, on a juste une taches parmi tant d'autres à mener à bien sous la supervision d'un chef de projet expérimenté, heureusement que l'IMIE n'a ménagé aucun effort pour nous éviter des surprises dans la vie professionnelle

Manuel d'utilisation

En pièce jointe.