Dédicace

Nous dédions ce présent mémoire

A

Nos parents

Pour les sacrifices déployés à nos égards ; pour leur patience leur amour et leur confiance en nous. Ils ont tout fait pour notre bonheur et notre réussite. Qu'ils trouvent dans ce modeste travail, le témoignage de notre Profonde affection et de notre attachement indéfectible. Nulle dédicace ne puisse exprimer ce que nous leur devons.

Que Dieu leur réserve la bonne santé et une longue vie.

A

Nos amis

En témoignage de nos sincères reconnaissances pour les efforts Qu'ils ont consentis pour nous soutenir au cours de nos études. Que Dieu nous garde toujours unis.

A

Toutes les personnes qui ont contribué de près ou de loin à la réalisation de présent projet.

Remerciements

La réalisation de ce mémoire a été possible grâce au concours de plusieurs personnes à qui nous voudrons témoigner toute notre reconnaissance.

En préambule à ce mémoire nous remercions ALLAH qui nous aide et nous donne la santé, la patience ainsi que le courage durant ces longues années d’étude.

Nous souhaitons adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l’élaboration de ce mémoire ainsi qu’à la réussite de cette formidable année universitaire.

Ces remerciements vont également à l’égard du corps professoral et administratif de notre école IAI-Niger, pour la richesse et la qualité de leur enseignement et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.

Nous tenons à remercier sincèrement Monsieur Chaibou Abdou, qui, en tant que Directeur de mémoire, s’est toujours montré à l’écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l’inspiration, l’aide et le temps qu’il a bien voulu nous consacrer et sans qui, ce mémoire n’aurait jamais vu le jour.

On n’oublie pas nos très chers parents qui ont toujours été là pour nous, pour leur contribution, leur soutien et leur patience. Vous nous avez donné un magnifique modèle de labeur et de persévérance. Nous sommes redevables d’une éducation dont on est fier.

Enfin, nous adressons nos plus sincères remerciements à tous nos proches et amis, qui nous ont toujours encouragé.

Merci à tous et à toutes.

# Sigles et Abréviations

TIC : Technologies d’Information et la Communication

NTIC : Nouvelles Technologies d’Information et de la Communication

UML : Unified Modeling Language

UP: Unified Process (Processus Unifié)

T2UP: Two Tracks Unified Process

PHP: Hypertext Preprocessor

HTML : Hyper Text Markup Language

CSS : Cascading Style Sheets (feuilles de style en cascade)

JS : JavaScript

POO : Programmation Orienté Objet

RSM : Responsable Sélection Médical

AS : Assistant Médical

# Table des figures

[Figure 1:Organigramme de SYNETCOM 3](#_Toc526901324)

[Figure 2 : Le système d’information soumis à deux types de contraintes 8](#_Toc526901325)

[Figure 3 : Le processus de développement en Y 9](#_Toc526901326)

[Figure 4 : Planning du projet 11](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901327)

[Figure 5 : Diagramme de contexte statique 17](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901328)

[Figure 6 : Diagramme de packages 28](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901329)

[Figure 7 : Diagramme de cas d’utilisation, package « gestion de la bibliothèque » 29](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901330)

[Figure 8 : Diagramme de cas d’utilisation, package « gestion administrative » 29](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901331)

[Figure 9: Diagramme de cas d’utilisation, package « gestion des enseignements » 30](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901332)

[Figure 10 : Diagramme de cas d’utilisation, package « gestion des promotions » 31](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901333)

[Figure 11 : Diagramme de cas d’utilisation, package « gestion des notes » 31](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901334)

[Figure 12 : Diagramme de cas d’utilisation, package «Gestion des absences/emplois du temps» 32](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901335)

[Figure 13 : Diagramme de cas d’utilisation, package « Gestion des droits d’accès » 33](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901336)

[Figure 14 : Cas d’utilisation gérer les filières, package « Gestion des enseignements » 35](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901337)

[Figure 15:Diagramme d’activité gérer les filières, package « Gestion des enseignements » 38](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901338)

[Figure 16: Diagramme de séquence gérer les filières, package « Gestion des enseignements » 39](#_Toc526901339)

[Figure 17 : cas d’utilisation : Gérer les classes, package « Gestion des enseignements » 40](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901340)

[Figure 18 : Diagramme d’activité gérer les classes, package « Gestion des enseignements » 43](#_Toc526901341)

[Figure 19 : Diagramme de séquence gérer les classes, package « Gestion des enseignements » 44](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901342)

[Figure 20 : Cas d’utilisation : gérer les années scolaires, package « Gestion des promotions » 45](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526901343)

Table des tableaux

[Table 1 : Exemple de calcul de la moyenne d’une unité d’enseignement 13](#_Toc526903045)

[Table 2 : Exemple de calcul de la moyenne d'un élément. 14](#_Toc526903046)

[Table 3 : Description des besoins fonctionnels 20](#_Toc526903047)

[Table 4 :Liste des cas d’utilisation 25](#_Toc526903048)

[Table 5 : Liste des cas d’utilisation par package 27](#_Toc526903049)

[Table 6 : Formalisme de description d’un cas d’utilisation 34](#_Toc526903050)

[Table 7 : Description détaillé, cas d’utilisation gérer les filières, package « Gestion des enseignements » 37](#_Toc526903051)

[Table 8 : Description détaillé, cas d’utilisation gérer les classes, package « Gestion des enseignements » 42](#_Toc526903052)

[Table 9 : Description détaillé, cas d’utilisation gérer les années scolaires, package « Gestion des promotions » 46](#_Toc526903053)

[Table 10 : Description détaillée, cas d’utilisation maintenir les notes, package « Gestion des notes » 49](#_Toc526903054)

[Table 11 : Description détaillée, cas d’utilisation :gérer les étudiants , package « Gestion administrative » 51](#_Toc526903055)

Table des matières

[Dédicaces i](#_Toc526902984)

[Remerciements ii](#_Toc526902985)

[Sigles et Abréviations iii](#_Toc526902986)

[Table des figures iv](#_Toc526902987)

[Table des tableaux vi](#_Toc526902988)

[Introduction 1](#_Toc526902989)

[CHAPITRE I : 2](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526902990)

[CONTEXTE GENERAL DU PROJET 2](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526902991)

[I. Présentation du centre d’accueil 3](#_Toc526902992)

[II. Cadre général du projet 4](#_Toc526902993)

[II.1 Contexte du projet 4](#_Toc526902994)

[II.2 Objectifs attendus 4](#_Toc526902995)

[CHAPITRE II : 5](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526902996)

[DEMARCHE DU PROJET 5](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526902997)

[I. Le choix de la méthode de gestion de projet 6](#_Toc526902998)

[I.1 Définition d’un processus de développement logiciel 6](#_Toc526902999)

[I.2 Le processus unifié 6](#_Toc526903000)

[I.3 Le processus 2TUP 7](#_Toc526903001)

[I.4 Un processus de modélisation avec UML 10](#_Toc526903002)

[II. Le planning prévisionnel 11](#_Toc526903003)

[CHAPITRE III : 12](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903004)

[ÉTUDE PRELIMINAIRE 12](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903005)

[I. Recueil des besoins fonctionnels 13](#_Toc526903006)

[II. Acteurs du système 16](#_Toc526903007)

[III. Identification des messages 17](#_Toc526903008)

[IV. Modélisation du contexte 19](#_Toc526903009)

[CHAPITRE IV : 21](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903010)

[CAPTURE DES BESOINS FONCTIONNELS 21](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903011)

[I. Identification des cas d’utilisation 22](#_Toc526903012)

[II. Structuration des cas d’utilisations en packages 25](#_Toc526903013)

[III. Description détaillée des cas d’utilisation 33](#_Toc526903014)

[IV. Identification des classes candidates 53](#_Toc526903015)

[CHAPITRE V : 56](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903016)

[ANALYSE 56](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903017)

[I. Diagramme de package 57](#_Toc526903018)

[II. Développement du modèle statique 58](#_Toc526903019)

[CHAPITRE VI : 63](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903020)

[CAPTURE DES BESOINS TECHNIQUES 63](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903021)

[I. Architecture Matérielle 64](#_Toc526903022)

[I.1 L’architecture 1 tiers 64](#_Toc526903023)

[I.2 L’architecture 2 tiers 64](#_Toc526903024)

[I.3 L’architecture 3 tiers 65](#_Toc526903025)

[I.4 Le choix de l’architecture 66](#_Toc526903026)

[CHAPITRE VII : 67](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903027)

[REALISATION 67](file:///C:\Users\hp\Downloads\Memoire%203aap%20Final.docx#_Toc526903028)

[I. Technologies et outils de développement 68](#_Toc526903029)

[I.1 Matériel technique 68](#_Toc526903030)

[I.3 Système de gestion de base des données utilisé : MySQL 70](#_Toc526903031)

[I.4 Framework de développement 70](#_Toc526903032)

[1. Pourquoi utiliser un Framework ? 70](#_Toc526903033)

[2. Laravel 70](#_Toc526903034)

[II. Quelques captures d’écrans 71](#_Toc526903035)

[II.1 Authentification 71](#_Toc526903036)

[II.2 Gestion des droits d’accès 72](#_Toc526903037)

[II.3 Profil des utilisateurs 74](#_Toc526903038)

[II.4 Affectation des notes 75](#_Toc526903039)

[II.5 Emplois du temps 76](#_Toc526903040)

[II.6 Absences 77](#_Toc526903041)

[Conclusion générale 78](#_Toc526903042)

[Bibliographie 79](#_Toc526903043)

[Webographie 80](#_Toc526903044)

INTRODUCTION

La révolution des technologies de l’information et de la communication (TIC), a engendré au cours de ces dernières années une progression notable de la création et de l’usage des applications de gestion dans la vie courante et professionnelle. Dans ce contexte de transformation numérique, et afin d’aider le centre national de transfusion sanguine (CNTS) à mettre en place un système de gestion automatique*,* SYNETCOM a initié un projet de développement d’une plateforme web standard qui répondra aux attentes des gestionnaires du centre national de transfusion sanguine objet de la présente étude.

La présente étude qui nous a été soumise dans le cadre de notre projet de fin d’étude pour l’obtention du diplôme d’ingénieurs des travaux informatiques, vient pour répondre aux besoins d’un secteur de base intéressant et très sensible, à savoir la santé.

En effet, la santé fait partie des domaines qui sont à la traine dans l’adoption des outils TIC alors qu’il en a plus besoin compte tenu du volume d’information géré tant au niveau des dons du sang qu’au niveau des donneurs mais aussi au niveau des différents acteurs impliqués.

Pour concevoir et mettre en place un système numérique répondant à notre thème «la gestion d’un centre de transfusion sanguine », et qui réunit les différentes fonctionnalités nécessaires à la gestion du centre national de transfusion sanguine, prenant en compte la reforme d’un centre régional de transfusion sanguine.

Le présent rapport reflète et décrie la démarche suivie pour mettre en place ce système informatisé demandé :

-La première partie décrie de façon globale la présentation du centre d’accueil ainsi que du projet ;

-La deuxième partie présente de manière globale l’analyse et la conception du projet ;

-La troisième partie est consacrée à la réalisation où nous présentons l’environnement de développement et quelques captures d’écrans.



Chapitre I : Présentation du centre d’accueil

## Présentation de SYNETCOM

SYNETCOM est une jeune entreprise spécialisée dans les domaines des nouvelles technologies de l’information et de la communication.

Vecteur d'innovation, levier d'efficacité, accélérateur de développement, grâce à son expérience et à son anticipation permanente sur les besoins des clients, SYNETCOM agit en véritable partenaire technologique pour lever les défis et les difficultés auxquels heurtent ses partenaires et optimiser les performances de leur système de gestion.

Sa vocation est d’apporter les ressources nécessaires pour faire bénéficier au mieux à ses partenaires les atouts des nouvelles technologies

SYNETCOM réfléchit aux problèmes du marché et conçoit les solutions novatrices et les services d'accompagnement adaptés aux besoins des clients.

## Organigramme

**Développeur 1 d’application**

**Développeur 3 d’application**

**Développeur 2 d’application**

**Assistant Administratif et Financier**

**Administrateur Général**

**Coordonnateur de Projet**

Chapitre II : Présentation du projet

## Contexte du projet

**L**a complexité croissante des systèmes informatiques a conduit les concepteurs à s’intéresser aux méthodes de développement. Ainsi, l’utilisation de l’outil informatique est une réalité de plus en plus présente dans les structures.

Le domaine de la santé n’a cessé ces dernières années d’évoluer et implicitement de se  
complexer, il devient d’autant plus difficile de pouvoir répondre aux différentes problématiques rencontrées sans avoir recours aux méthodes de gestion et d’aide à la décision ; en particulier, l’approche de modélisation et simulation a prouvé son efficacité.

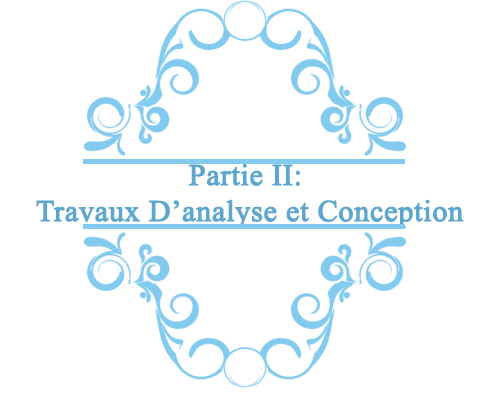
Cette étude se focalise sur la composante de la transfusion sanguine de manière générale et a pour objectif de contribuer plus particulièrement à une meilleure compréhension du fonctionnement organisationnel d’un centre de transfusion sanguine. Ce document ne concerne que ces aspects.

## Travail attendu

Ce projet a pour objectif principal de proposer une solution à un problème concret, et ceci en partant d’une définition des besoins. Le principal objectif du système est d’assurer les dons de sang, ainsi que le suivi de la traçabilité des donneurs entre les différents dons effectués et de conserver ces informations pour consultation ultérieure pour une prise en charge continue, coordonnée et sécurisée des documents.

Le système permettra globalement de :

* Gérer les dossiers des donneurs,
* Gérer les prélèvements,
* Gérer la catégorisation des dons,
* Gérer le stock des poches de sang,
* Gérer le suivi de la traçabilité,
* Planifier les points de collectes (fixes et mobiles),
* Réception des notifications par les donneurs via Mail et/ou SMS sur des mises à jour d’informations (Résultats d’analyse, Urgences, Remerciements…),
* Donner les droits d’accès aux fonctionnalités par profils d’utilisateurs.



# Chapitre III : Analyse

## Méthode d’analyse

Une méthodologie de développement est un cadre utilisé pour structurer, planifier et contrôler le développement d’une application. En effets, nous disposons de nos jours d’un ensemble de méthodologie chacune respectant un certain nombre de principe pour aboutir à la réalisation d’un projet. Ainsi nous allons parler principalement de quatre types de méthodes de conception.

## Pourquoi utiliser une méthode d’analyse

On appelle « gestion de projet » éventuellement « conduite de projet » l’organisation méthodologique mise en œuvre pour faire en sorte que l’ouvrage réalisé par le maitre d’œuvre réponds aux attentes du maitre d’ouvrage et qu’il soit livré dans les conditions de coût et de délais prévus initialement.

En effet, les projets sont gérés avec la méthode dite « classique » qui se caractérise pour recueillir les besoins, définir le produit, le développer et le tester avant de le livrer. On parle alors ici d’une approche prédictive. Comme son nom l’indique, il s’agit ici de prévoir des phases séquentielles où il faut valider l’étape précédente pour passer à la suivante.

Le chef de projet doit alors  
s’engager sur un planning précis de réalisation du projet en prévoyant des  
jalons de débuts et fins de phases ainsi que les tâches à effectuer. Il faut tout  
faire bien du premier coup car elle ne peut pas permettre de retours en arrière.  
Une décision ou un problème rencontré dans une phase peuvent remettre en  
cause partiellement ou totalement les phases précédentes validées. Dans un  
cycle « en cascade » les risques sont détectés tardivement puisqu’il faut  
attendre la fin du développement pour effectuer la phase de test. Plus le projet  
avance, plus l’impact des risques augmente : il sera toujours plus difficile et  
coûteux de revenir en arrière lorsqu’on découvre une anomalie tardivement.  
Par conséquent, comment peut-on augmenter la satisfaction du client en  
facilitant la gestion de projet et améliorant la qualité de développement ? Pour  
se faire, nous allons voir d’autres méthodes modernes plus adéquates.

## Classification des méthodologies

## Méthodes fonctionnelles

Les méthodes fonctionnelles ou cartésiennes sont des démarches consistant à rechercher et à caractériser les fonctions offertes par un produit pour faire les besoins de son utilisateur. Il est aussi défini comme étant une méthode permettant à décomposer hiérarchiquement une application en un ensemble de sous applications. Parmi les fonctions de chacune de celles-ci sont affinées alternativement en sous fonctions simples à coder dans un langage de programmation donné. L’une de ces méthodes, on peut noter FAST (Fonction Analyse System Technique), SADT (Structurel-Analyse-Design-Technique) Warnier …

Forces de ces méthodes :

* Simplicité du processus de conception préconisée ;
* Adéquations à capturer les besoins de l’utilisateur ;
* Capacités de produire des solutions à plusieurs niveaux d’abstraction.

Faiblesses de ces méthodes

* Effort d’analyse concentré sur les fonctions (négligeant la cohérence des données) ;
* Règles de décomposition non explicites ;
* Difficulté à tenir compte des interactions non hiérarchiques dans les systèmes complexes.

## Méthodes classiques

Souvent spécialisées pour la conception d’un certain type de systèmes, les méthodes systémiques définissent différents niveaux de préoccupation ou d’abstraction et proposent de nombreux modèles complémentaires. Autrement dit ces méthodes proposent une double démarche de modélisation, celles des données et des traitements. Elles sont influencées par les systèmes de gestion de bases de données SGBD. On a comme exemple MERISE, AXIAL RACINE...

Forces de ces méthodes :

* Les concepts sont peu nombreux et simples
* Elle est assez indépendante vis-à-vis de la technologie,
* Bonne adaptation à la modélisation des données et à la conception des bases de données,

Faiblesses de ces méthodes

* Double démarche de conception celles des données et des traitements,
* Impossible de fusionner les deux aspects (données et traitements)

## Méthodes unifiées :

Les méthodes unifiées sont des méthodes de développement pour les logiciels orientés objets. Plus exactement, ce sont les meilleures pratiques du  
développement objet suivies pour la réalisation d’un système. Donc le Processus Unifié (UP) est un processus de développement logiciel « Itératif et incrémental, centré sur l’architecture, conduit par les cas d’utilisation et piloté par les risques » :

* **Itératif et incrémental** : le projet est découpé en itérations de courte durée (environ 1 mois) Qui aident à mieux suivre l’avancement global. A la fin de chaque itération, une partie exécutable du système final est produite, de façon Incrémentale.
* **Centré sur l’architecture** : tout système complexe doit être décomposé en parties Modulaires afin de garantir une maintenance et une évolution facilitées. Cette architecture (Fonctionnelle, logique, matérielle, etc.) doit être modélisée en UML et pas seulement Documentée en texte.
* **Piloté par les risques** : les risques majeurs du projet doivent être identifiés au plus tôt, mais surtout levés le plus rapidement possible. Les mesures à prendre dans ce cadre déterminent l’ordre des itérations.
* **Conduit par les cas d’utilisation** : le projet est mené en tenant compte des besoins et des exigences des utilisateurs. Les cas d’utilisation du futur système sont identifiés, décrits avec précision et priorisés.

## 2.3.1. Méthodes UP :

Le Processus Unifié (PU ou UP en anglais pour Unified Process) est une méthode de développement logiciel construite sur UML ; elle est itérative et incrémentale, centrée sur l’architecture, conduite par les cas d’utilisation et pilotée par les risques.

Itérative et incrémentale : la méthode est itérative dans le sens où elle propose de faire des itérations lors de ses différentes phases, ceci garantit que le modèle construit à chaque phase ou étape soit

* Affiné et amélioré. Chaque itération peut servir aussi à ajouter de nouveaux incréments.
* Conduite par les cas d’utilisation : elle est orientée « utilisateur » pour répondre aux besoins de celui-ci.
* Centrée sur l’architecture : les modèles définit tout au long du processus de développement vont contribuer à établir une architecture cohérente et solide.

### Pilotée par les risques : en définissant des priorités pour chaque fonctionnalité, on peut minimiser les risques d’échec du projet.

## Méthode 2TUP :

**2TUP** est un processus de développement logiciel qui implémente le Processus Unifié. Le **2TUP** propose un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels. Il commence par une étude préliminaire qui consiste essentiellement à :

* Identifier les acteurs qui vont interagir avec le système à construire ;
* Identifier les messages qu'échangent les acteurs et le système ;
* Produire le cahier des charges ;
* Modéliser le contexte (le système est une boîte noire, les acteurs l'entourent et sont reliés à lui, sur l'axe qui lie un acteur au système on met les messages que les deux s'échangent avec le sens).

Le processus s'articule ensuite autour de 3 phases essentielles :

**

Figure 1 : Le système d’information soumis à deux types de contraintes

**La branche gauche** (fonctionnelle) comporte :

* La capture des besoins fonctionnels, qui produit un modèle des besoins focalisé sur le métier des utilisateurs. Elle qualifie au plus tôt le risque de produire un système inadapté aux utilisateurs. De son côté, la maîtrise d’œuvre consolide les spécifications et en vérifie la cohérence et l'exhaustivité.
* L'analyse, qui consiste à étudier précisément la spécification fonctionnelle de manière à obtenir une idée de ce que va réaliser le système en termes de métier. Les résultats de l'analyse ne dépendent d'aucune technologie particulière.

### **La branche droite** (architecture technique} comporte :

* La capture des besoins techniques, qui recense toutes les contraintes et les choix dimensionnant la conception du système. Les outils et les matériels sélectionnés ainsi que la prise en compte de contraintes d'intégration avec l'existant conditionne généralement des prés requis d'architecture technique.
* La conception générique, qui définit ensuite les composants nécessaires à la construction de l'architecture technique. Cette conception est complètement indépendante des aspects fonctionnels. Elle a pour objectif d'uniformiser et de réutiliser les mêmes mécanismes pour tout un système. L'architecture technique

Elle construit le squelette du système informatique et écarte la plupart des risques de niveau technique. L'importance de sa réussite est telle qu'il est conseillé de réaliser un prototype pour assurer sa validité.

**La branche du milieu** comporte :

* **La conception préliminaire**, qui représente une étape délicate, car elle intègre le modèle d'analyse dans l'architecture technique de manière à tracer la cartographie des composants du système à développer, la conception détaillée, qui étudie ensuite comment réaliser chaque composant ; l'étape de codage, qui produit ces composants et teste au fur et à mesure les unités de code réalisées
* **La conception détaillée**, qui étudie ensuite comment réaliser chaque composant ;
* **Le codage et tests**, qui produisent ces composants et teste au fur et à mesure les unités de code réalisées ;
* **L’étape de recette**, qui consiste enfin à valider les fonctions du système développé.

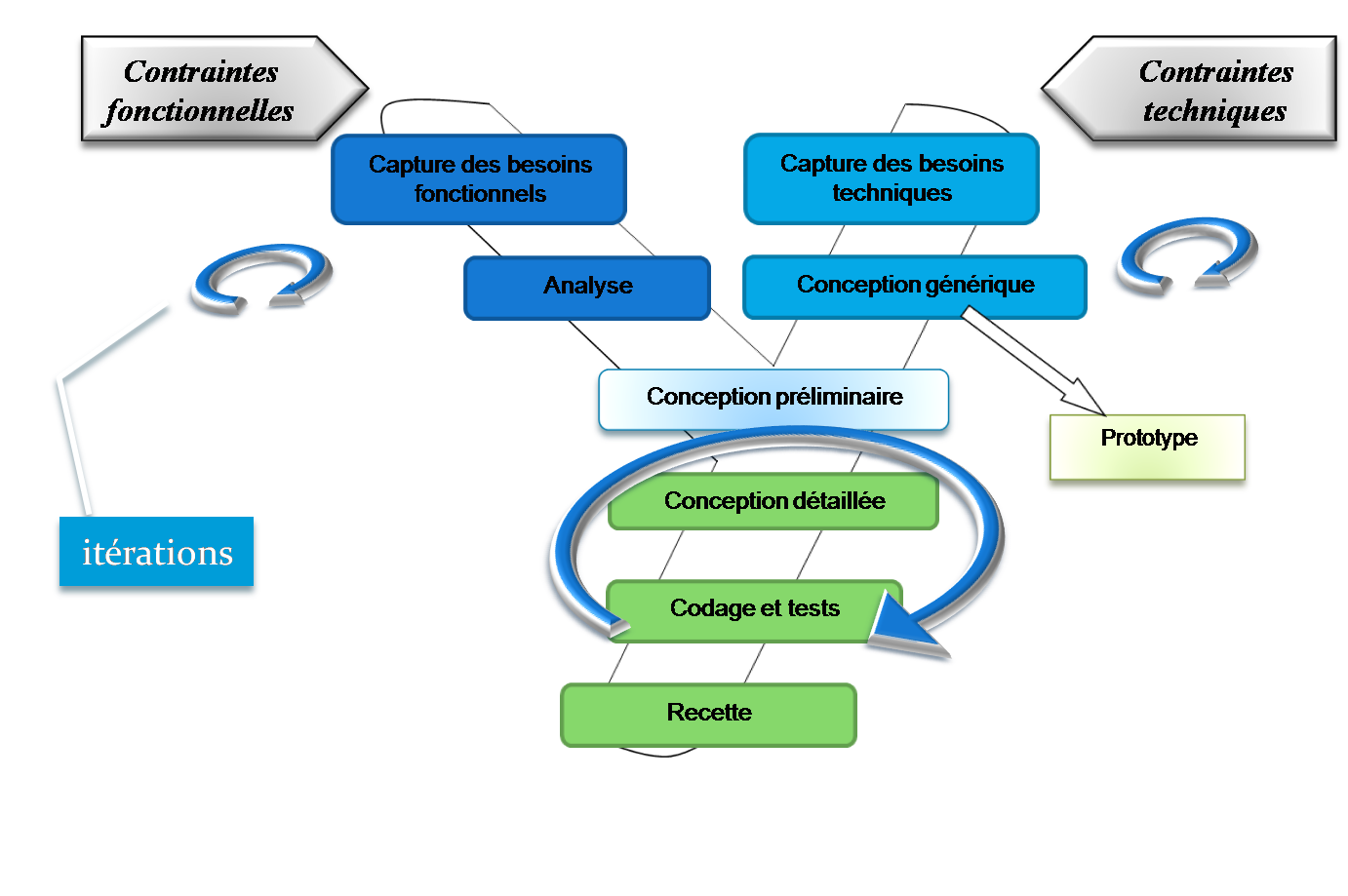


Figure 2 : Le processus de développement en Y

## éthodes agiles :

Ces méthodes constituent un gain en productivité ainsi qu'un avantage  
compétitif tant du côté client que du côté du fournisseur. En effet, ces méthodes de développement dites « méthodes agiles » visent à réduire le cycle de vie du logiciel (donc accélérer son développement) en développant une version minimale, puis en intégrant les fonctionnalités par un processus itératif basé sur une écoute client et des tests tout au long du cycle de développement. Parmi ces méthodes nous pouvons citer : Scrum, XP (eXtreme Programming),   
RAD (Rapid Application Development), etc.…

## 2.4.1. XP :

Extrême Programming, ou XP, est une méthode agile de gestion de projet particulièrement bien adaptée aux projets de développement informatique. Elle a été conçue par Kent Beck pour accélérer les développements alors qu’il travaillait pour la société Chrysler. L’idée lui est venue alors qu’il devrait intervenir sur un logiciel de paie écrit en langage « Small talk » ayant accumulé une dette technique considérable, le rendant particulièrement complexe à maintenir et à faire évoluer.

Le principe fondamental de la méthode XP est de faire collaborer étroitement tous les acteurs du projet et d’opter pour des itérations de développement très courtes. La planification des tâches reste très souple et l’estimation des charges simplifiée par des projections à très court terme. Ainsi la correspondance entre ce qu’attendent le client et les réalisations est garantie. Les fonctionnalités sont livrées régulièrement, afin d’être testées et validée au travers de prototypes opérationnels. L’Extrême Programming préconise également le travail en binôme des développeurs, facilitant ainsi la production d’un code simple, facilement lisible et maintenable.

## 2.4.2. Scrum :

Scrum est considéré comme un cadre ou « Framework » de gestion de projet.  
Ce cadre est constitué d'une définition des rôles, de réunions et d'artefacts. Le mot Scrum est un terme emprunté au rugby qui signifie « mêlée » encore appelée « Morning **»**. Dans cette méthode, on focalise l'équipe de façon itérative sur un ensemble de fonctionnalités. En effet, Scrum définit 3 rôles :

* Le propriétaire du produit « Product Owner » qui porte la vision du produit à réaliser (représentant généralement le client) ;
* Le directeur du produit « Scrum Master » garant de l'application de la  
  méthodologie Scrum ;
* L'équipe de développement qui réalise le produit.

La vie d'un projet Scrum est rythmée par un ensemble de réunions clairement  
définies et strictement limitées dans le temps (timeboxing):Planification du Sprint (Sprint = itération) : au cours de cette réunion, l'équipe de développement sélectionne les éléments prioritaires du « Product Backlog » (liste ordonnancée des exigences fonctionnelles et non fonctionnelles du projet) qu'elle pense pouvoir réaliser au cours du sprint (en accord avec le « Product Owner »).Revue de Sprint : au cours de cette réunion qui a lieu à la fin du sprint, l'équipe de développement présente les fonctionnalités terminées au cours du sprint et recueille les feedbacks du Product Owner et des utilisateurs finaux. C'est également le moment d'anticiper le périmètre des prochains sprints et d'ajuster au besoin la planification de release (nombre de sprints restants).  
Rétrospective de Sprint : la rétrospective qui a généralement lieu après la revue de sprint est l'occasion de s'améliorer (productivité, qualité, efficacité, conditions de travail, etc.) à la lueur du "vécu" sur le sprint écoulé (principe d'amélioration continue).Mêlée quotidienne : il s'agit d'une réunion de synchronisation de l'équipe de développement qui se fait debout (elle est aussi appelée "stand up meeting") en 15 minutes maximum au cours de laquelle chacun répond principalement à 3 questions :« Qu'est-ce que j'ai terminé depuis la dernière mêlée ? Qu'est-ce que j'aurai terminé d'ici la prochaine mêlée ? Quels obstacles me retardent ? »

C:\Users\MINA\Desktop\momo IGS\Untitled-Diagram.png

Figure 3 : La méthode scrum

## 3. Choix de la méthodologie :

## 3.1. Méthodologie de conception :

Après l’étude des différentes méthodologies, nous avons opté pour Scrum et UP pour les raisons qui suivent :

* Scrum convient aux équipes ayant un nombre de développeurs réduits. Ceci est le cas de notre projet ;
* La progression des tâches s’effectue pendant une durée de développement courte ;
* UP s’applique à tous les types de projet informatique pour les suggestions et remarques ;
* UP est une méthode itérative et incrémental ;
* UP est essentiellement basée sur la satisfaction des besoin set des exigences du clients et des utilisateurs ;
* UP se base sur le diagramme des cas d’utilisation qui illustre au mieux les besoins fonctionnels et opérationnels auxquels doivent répondre le produit final.

## 3.2. Présentation du langage UML :

Les auteurs des méthodes (OOSE de Jacobson, OMT de Rumbaugh, BOOCH de Grady Booch) se sont fixé des objectifs dont entre autres la représentation des systèmes entiers par des concepts objet ; l’établissement d’un couplage explicite entre les concepts et les interfaces exécutables et la création d’un langage de modélisation utilisable à la fois par les humains et les machines et adaptés aux systèmes simples et complexes. D’où la naissance d’UML qui est donc une norme du langage de modélisation objet qui a été publiée, dans sa première version en septembre 1997 par l’OMG (Object Management Group), instance de normalisation internationale du domaine de l’objet. UML est un langage de modélisation graphique et textuel destiné à comprendre et à décrire des besoins, spécifier et documenter des systèmes, esquisser architectures logicielles, concevoir des solutions et communiquer des points des vues.  De plus UML comble une lacune importante des technologies objet, il permet d’exprimer, d’élaborer et de modéliser au sens de la théorie des langages. De ce fait il contient les éléments constitutifs de ces derniers : concepts, une syntaxe et une sémantique.

## 3.2.1. Les diagramme UML :

UML définit 13 types de diagrammes divisés en deux catégories : les diagrammes structurels ou statiques rassemblent :

* Diagramme de classe (Class Diagram) ;
* Diagramme d’Objet (Object Diagram) ;
* Diagramme de composant (Composent Diagram) ;
* Diagramme de déploiement (Deployment Diagram) ;
* Diagramme de paquetage (Package Diagram)
* Diagramme de structure composite (Composite Structure Diagram).

Les diagrammes comportementaux qui rassemblent :

* Diagramme des cas d’utilisation (Use Case Diagram).
* Diagramme d’état-transitions (State Machine Diagram) ;
* Diagramme d’activité (Activity Diagram) ;
* Diagramme de séquence (Sequence Diagram) ;
* Diagramme de communication (Communication Diagram) ;
* Diagramme global d’interaction (Interaction Overview Diagram) ;
* Diagramme de temps (Timing Diagram) ;

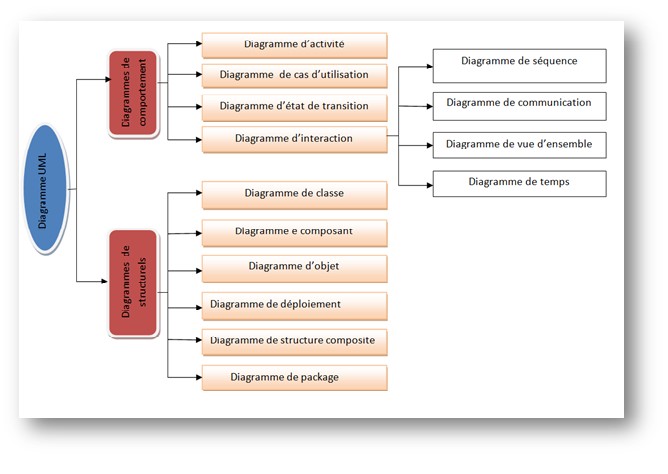


Figure 4 : Les diagrammes UML

## 4. Planning prévisionnel

La clé principale de la réussite d’un projet est un bon planning. En effet, le planning aide à bien subdiviser le travail et séparer les taches à réaliser, il offre une meilleure estimation et gestion de temps nécessaire pour chaque tâche. De plus, il donne assez de visibilité permettant d’estimer approximativement la date d’achèvement de chaque tâche.

Nous avons choisi comme outil pour la planification de notre projet Gantt Project qui est un logiciel de gestion de projet.

Gantt Project permet de planifier les projets et les ressources, et d’assurer le suivi des projets durant leur réalisation. Il permet de réaliser une représentation graphique du déroulement d'un projet et de rendre compte de son avancement.

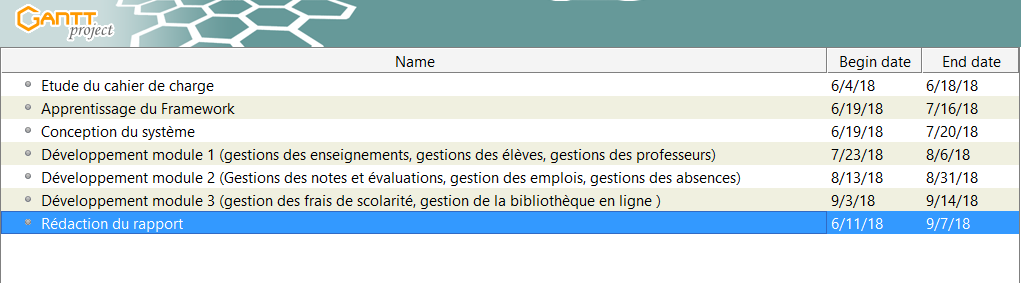


Figure 4 : Planning du projet

## Recueil des besoins fonctionnels

## Cette phase correspond à une recherche sur le terrain pour bien définir le cadre de notre système. Nous avons effectué plusieurs recherches pour identifier au mieux les besoins auxquels l’application doit répondre, et ceci afin de répondre aux attentes des potentiels utilisateurs.

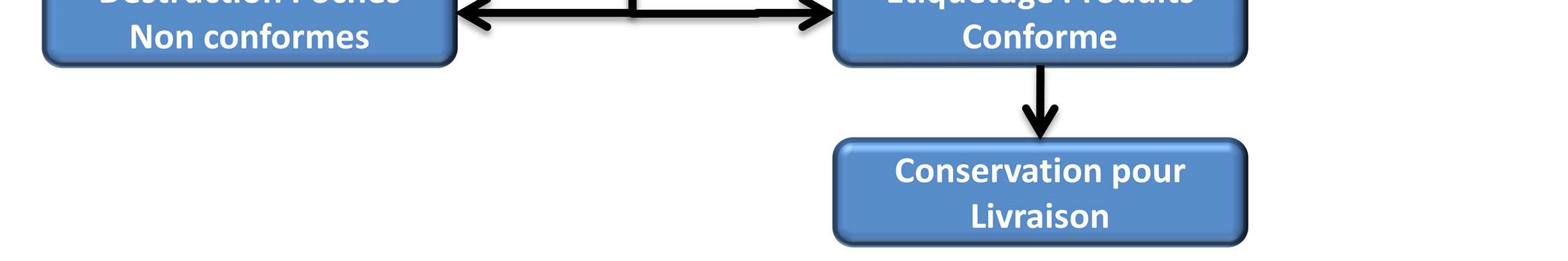
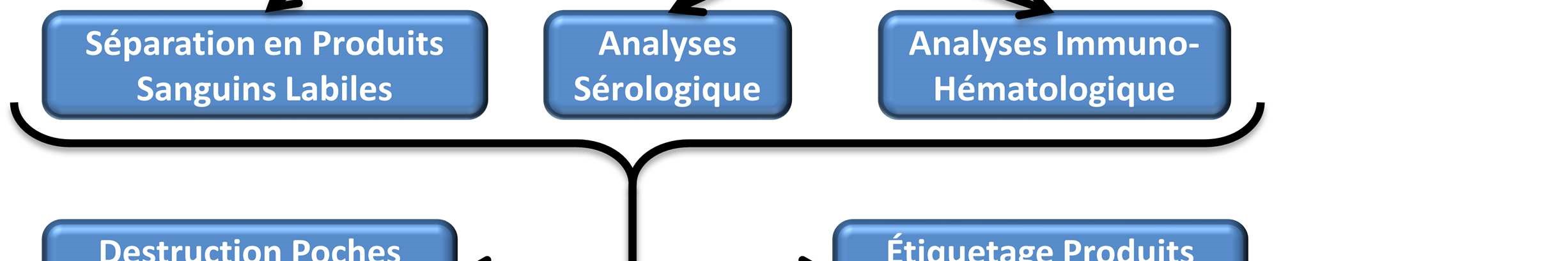
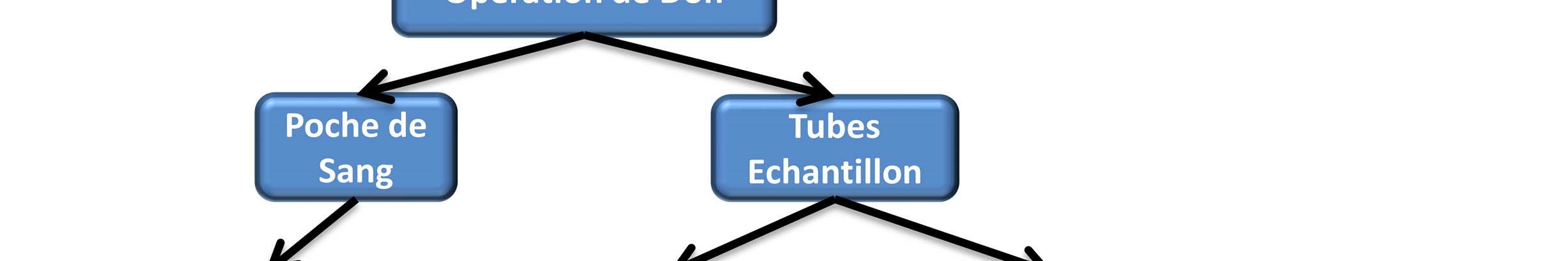
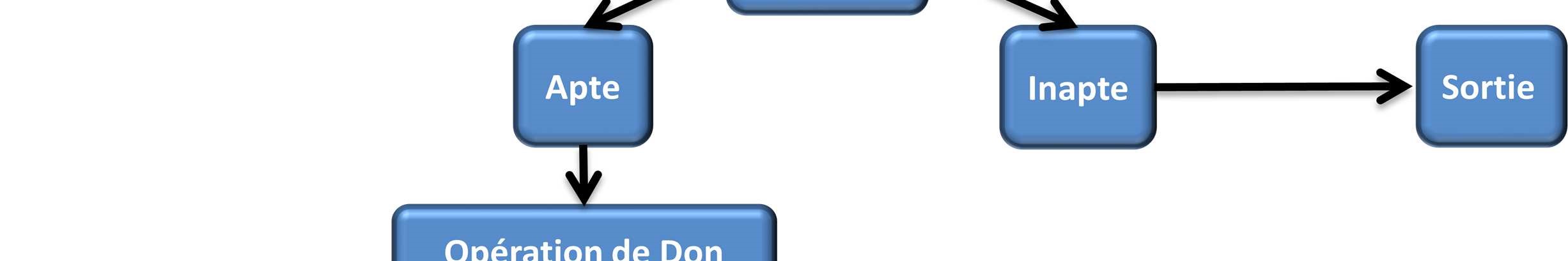
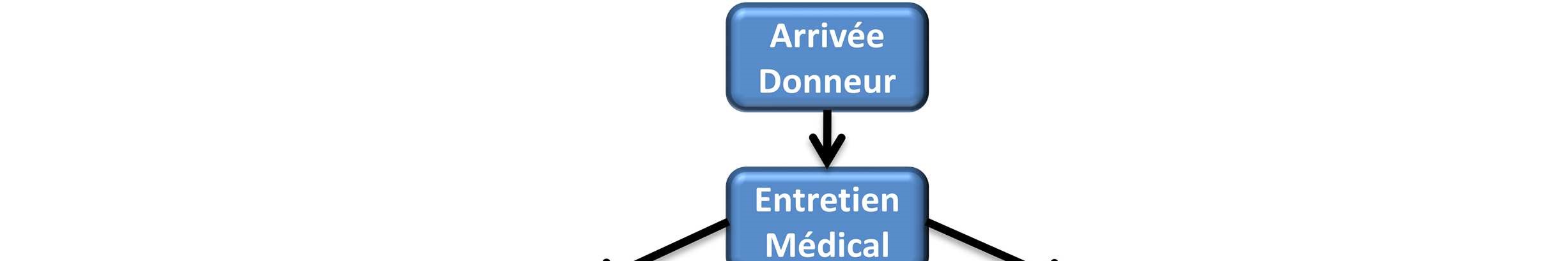
A cet effet nous sommes allés chercher les informations à travers des interviews menées avec la Directrice du centre national de transfusion sanguine (CNTS).

Tout ceci dans le but de comprendre le fonctionnement de don du sang, et avoir une idée de comment s’effectue la gestion d’un centre de transfusion sanguine :

Le centre national de transfusion sanguine est responsable d’assurer la collecte, le traitement et la distribution des produits sanguins labiles.

Il est situé au sein de la capitale, la plus importante en matière de population ce qui induit forcément une demande très importante en produits sanguins labiles (PSL).  
Les donneurs qui se présentent au centre régional de transfusion sanguine (CRTS) passent d’abord par l’accueil pour s’enregistrer et ensuite passent à l’entretien médical qui détermine si le donneur est apte à donner son sang, dans ce cas-là la prochaine étape et la salle de prélèvement ou l’opération de don se déroule.

À la fin le donneur profite d’une collation avant de quitter le centre pendant que le sang collecté est transféré pour être analysé et séparé en produits sanguins labiles (PSL) qui va être stocké en attendant d’être livré aux malades.



### Figure I. Différents processus du CRTS de Niamey

## C. Analyse des besoins

## 1. Les acteurs du système

## a. Définition

## Un acteur représente l'abstraction d'un rôle joué par des entités externes (utilisateur, dispositif matériel ou autre système) qui interagissent directement avec le système étudié. Un acteur peut consulter et/ou modifier directement l’état du système, en remettant et/ou en recevant des messages éventuellement porteurs de données.

## a. Identification des acteurs

Les acteurs qui interagissent avec le système dans un premier temps sont :

* Le préleveur :
* Le laborantin :
* L’assistant médical :
* Le responsable de la sélection médical :
* L’administrateur : l’administrateur a accès à toutes les fonctionnalités de bases et avancées de l’application. Il effectue les paramétrages de base. Il crée les profils utilisateurs et attribue les droits d’accès.

## 1. Le diagramme de contexte

Le diagramme de contexte statique délimite le domaine d’étude en précisant ce qui est à la charge du système et en identifiant l’environnement extérieur au système étudié avec lequel ce dernier communique. Ses composants sont :

* Les acteurs externes. Un acteur externe est une entité externe au système étudié qui interagit avec le système.
* Un processus unique symbolisant le Système Information étudié :
* Echange entre le système étudié et son environnement

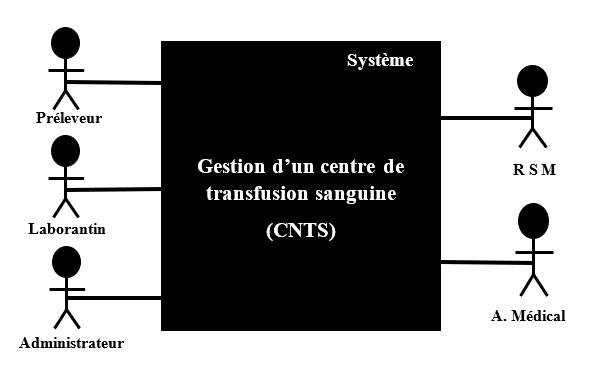


Figure 5 : Diagramme de contexte statique

2. Le diagramme de packages

Figure 6 : Diagramme de packages

Chapitre IV : Conception

I. Diagramme des cas d’utilisations



# Chapitre V : Architecture et description des outils et technologies utilisés

## 5.1. Architecture interne de l’application :

Une architecture est un modèle générique et conceptuel qui se rapporte à un sujet et qui représente la fonctionnalité, la structure, le positionnement, l’interrelation des différents types d’élément qui la composent.

En règle générale, une application est découpée 3 couches d’abstraction :

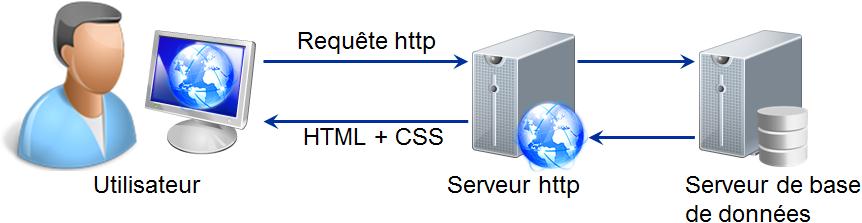
* La couche présentation ; c’est la partie de l’application visible par les utilisateurs (nous parlons de l’interface utilisateur). Dans notre cas cette couche est un navigateur web, qui se présente sous forme de page HTML, composé de formulaire et de bouton ;
* La couche métier : correspond à la partie fonctionnelle de l’application, celle qui implémente la logique, et qui décrit les opérations que l’application opère sur les données en fonction des requêtes d’un utilisateur effectuées à travers la couche présentation :
* La couche accès aux données : c’est la partie gérant l’accès à la base des données du système.

Il existe différentes architectures dont entre autres ;

* L’architecture 1\_tiers : La conception de l’application est élaborée de manière à fonctionner sur un ordinateur unique. En fait, tous les services fournis par l'application résident sur la même machine et sont inclus dans l'application. Toutes les fonctionnalités sont donc comprises dans une seule couche logicielle.
* L’architecture 2\_tiers : Appelée aussi architecture client lourd/serveur, elle est assez simple dans sa mise en œuvre. Ce type d'architecture est constitué uniquement de deux parties : le « client lourd » demandeur de service d’une part et le « serveur de données » qui fournit le service d'autre part. Nous aurons donc la base de données qui sera délocalisée sur un serveur dédié appelé le serveur de données qui fournira les données à exploiter.
* L’architecture 3\_tiers : Cette architecture physique est assez semblable à l’architecture client/serveur, mais en plus des « clients » et du serveur de données évoquées plus haut, un serveur d'application intervient comme un troisième tiers. En effet, les machines clientes également appelées « clients légers » ne contiennent que l'interface de l'application de manière qu’elles soient déchargées de tout traitement. En effet, le traitement est ainsi assuré par le serveur d'application, qui sert de liaison entre l'interface applicative et les données localisées au niveau du serveur de données.

## 5.2. Choix de l’architecture :

Pour nous, utiliser une architecture à 3 niveaux est beaucoup plus propre. Cela permet de diviser les tâches et par conséquent d’avoir des développeurs spécialisés sur un des trois niveaux. Cependant dans notre application nous avons utilisé l’architecture 3\_tiers qui est illustrée dans la figure qui suit :

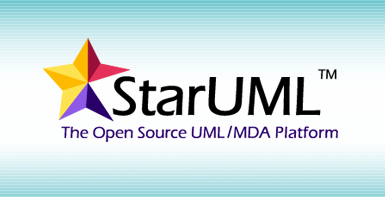


## 5.3. Outils utilisés :

* PowerAMC :

PowerAMC fait partie des AGL (Atelier de Génie Logiciel), programme qui assiste le développeur au moment de la conception de certaines tâches, les AGL sont très pratique compte tenu de leur souplesse, leur efficacité et permettent aux développeurs de gagner plus de temps dans la réalisation de certain nombre de travaux ;

* StarUML :

 C’est un logiciel de modélisation UML open source sous une licence modifiée de GNU GPL.L'objectif de ce projet est de se substituer à des solutions commerciales comme IBM Rational Rose ou Borland Together. StartUml gère la plupart des diagrammes spécifiés dans la norme Uml 2.0 ;

## 5.4. Outils de développements

* **Apache**

 Le logiciel **Apache** est un **serveur** http en Open Source Utilisé principalement sur les hébergements Internet en Linux, bien qu’il soit également utilisable en Windows (concurrent d’Internet et Information Service-IIS), Unix ou OSX. C’est actuellement le plus utilisé sur le WEB ;

* **MySql**

MySQL est un serveur de bases de données relationnelles SQL développé dans un souci de performances élevées en lecture, ce qui signifie qu'il est davantage orienté vers le service de données déjà en place que vers celui de mises à jour fréquentes et fortement sécurisées. La première version de MySQL est apparue le 23 mai 1995.

* **Xamp**

Xamp est un ensemble de logiciels permettant de mettre en place facilement un serveur Web local, un serveur FTP et un serveur de messagerie électronique. Il s’agit d’une distribution de logiciels libres offrent une bonne souplesse d’utilisation, réputée pour son installation simple et rapide. Ainsi il est à la portée d’un grand nombre de personnes puisqu’il ne requiert pas de connaissance particulières et fonctionne, de plus, sur les systèmes d’exploitation les plus répandus ;

## 5.4. Outils de conceptions

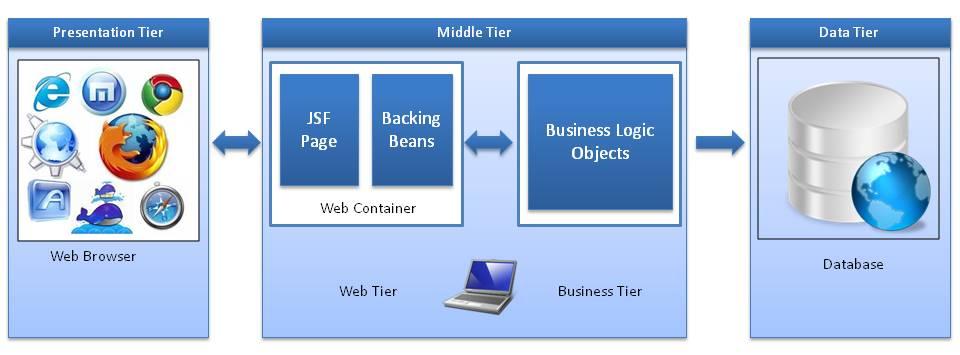
* **Google Chrome**

 Google Chrome est un navigateur web créé par Google et basé sur le projet open sourceChromium. L'objectif initial de Google en développant Chrome était de fournir auxinternautes un nouveau navigateur plus rapide et proposant plusieurs innovations par rapportà Mozilla Firefox ou à Microsoft Internet Explorer. En plus des questions de performances,Google Chrome se différencie notamment de sa concurrence en dédiant un espace mémoire àl'utilisateur et un processus unique pour chaque fenêtre ou nouvel onglet, afin d'assurer unemeilleure stabilité et sécurité.

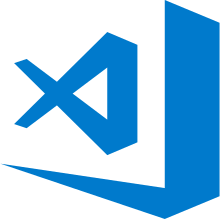
* **Mozilla Firefox**

 Mozilla Firefox est un navigateur Web libre et gratuit, développé et distribué par la MozillaFoundation avec l'aide de centaines de bénévoles grâce aux méthodes de développementdu logiciel libre/open source et à la liberté du code source.Firefox est à l'origine un programme dérivé du logiciel Mozilla (aujourd'hui connu sous lenom de Sea Monkey),mais reprenant uniquement les fonctions de navigation de ce dernier.Ce logiciel multiplateforme est compatible avec diverses versions de Windows, Mac OSX etGNU/Linux ;

* **MVC**



Le paradigne MVC est un schéma de programmation qui propose de sépare une application en trois parties :

* Le modèle : qui contient la logique et l’état de l’application ;
* La vue : qui représente l’interface utilisateur ;
* Le contrôleur qui gère la synchronisation entre la vue et le modèle.
*  **Visual Code Studio**

Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et OS X. Visual Studio Code est un nouvel outil qui combine la simplicité d'un éditeur de code avec ce dont les développeurs ont besoin pour le cycle edit-build-debug. Aller au-delà de la coloration syntaxique et de la saisie semi-automatique avec IntelliSense, qui fournit des complétions intelligentes basées sur des types de variables, des définitions de fonctions et des modules importés. Code de débogage à partir de l'éditeur. Lancez ou attachez à vos applications en cours d'exécution et déboguez avec des points d'arrêt, des piles d'appels et une console interactive.

## 5.5. Langage de programmation utilisé

* **PHP**

PHP (abréviation de Personal Home Page Hypertext Préprocesseur )à ce jour est un langage se scripts que s’intègre aux pages Html et qui permet sz réaliser des pages dynamiques .La première version date 1994 et s’appelait PHP FI. Elle n’avait pour ambition que de pouvoir insérer quelques traitement simples dans une page Html, comptage des visites.PHP s’exécute coté serveur, c’est-à-dire que le serveur interprète, analyse et exécute le script PHP avant d’en envoyer le résultat au client (navigateur).De ce fait, le client n’interagit pas directement avec PHP ce qui renforce la sécurité de l’application. De plus PHP est un produit « Open Source » c’est-à-dire que le code est accessible à tout développeur. En plus, il propose un accès facile aux bases de données. Malgré son manque de puissance face à d’autre langage de programmation (Perl, C, Java, JEE, etc …).PHP n’en demeure pas moins un langage solide et accessible à tous.

* **HTML**

 HTML (HyperText Markup Language) est un langage de balisage (un langage qui utilise des balises ‘<>’) servant à l’écriture de page du Word Wide Web (WWW). Il permet :

* de structurer sémantiquement et de faire la mise en page du contenu des pages ;
* d’inclure des ressources multimédias dont des images ;
* de réaliser des formulaires de saisie, etc.

Il est souvent utilisé conjointement avec langages de programmation comme PHP, JavaScript et les CSS. HTML5 est l’évolution la plus récente d’HTML. Cette évolution consiste en une multitude des nouvelles fonctionnalités qui ont été apportées au langage HTML ainsi qu’au JavaScript. Si vous savez faire de l’HTML « classique » vous devriez donc apprendre à manipuler ces nouvelles fonctionnalités.

* **CSS**

CSS (Cascading Style Sheets : feuille de style en cascade) est un langage informatique qui sert à décrire la présentation des pages Web CSS s’applique à une ou plusieurs pages du site. Le terme « en cascade » indique que la mise en forme d’une page peut faire appel) plusieurs feuilles de style. La nouvelle version de CSS est le CSS3 qui a apporté au langage de nombreuses nouveautés. Il s’agit par exemple d’un ensemble de nouveaux effets à appliquer sur nos éléments HTML, d’un ensemble d’un ensemble de nouveaux sélecteurs, de nouvelles manières de spécifier les couleurs, une détection des caractéristiques de l'appareil de l'utilisateur, des calculs dans les feuilles de style, des SVG en arrière-plan... Bref, le CSS3 est un généreux enrichissement des feuilles de style qui, en plus d'être profitable à l'utilisateur, l'est également pour le développeur.

* **JQuery**

 JQuery est une bibliothèque (c'est-à-dire un ensemble de codes prêts à  
l'emploi) conçue pour simplifier l'écriture de codes JavaScript et AJAX. Créée en 2006 par John Resig, cette bibliothèque est la plus célèbre et la plus utilisée à ce jour.

JQuery est une bibliothèque (c'est-à-dire un ensemble de codes prêts à  
l'emploi) conçue pour simplifier l'écriture de codes JavaScript et AJAX. Créée en 2006 par John Resig, cette bibliothèque est la plus célèbre et la plus utilisée à ce jour.

* **Ajax**

AJAX (acronyme d'Asynchronous JavaScript And XML.) est apparu en 1995. Son utilisation est très intéressante, car elle permet de mettre à jour une partie (et une partie seulement) d'une page Web en demandant les données nécessaires à un serveur. Les échanges client/serveur sont donc limités et les pages Web sont affichées plus rapidement, pour le plus grand plaisir des utilisateurs.

* **JavaScript**

Le JavaScript est un langage de script incorporé dans un document HTML. Historiquement il s’agit même du premier langage de script pour le Web. Ce langage est un lange de programmation qui permet d’apporter des améliorations au langage HTML, en permettant d’exécuter des commandes du coté client, c’est-à-dire au niveau du navigateur et non du coté serveur web. Ainsi le langage JavaScript est fortement dépendant du navigateur appelant la page web dans laquelle le script est incorporé, mais en contrepartie il ne nécessite pas de compilateur, contrairement au langage Java, avec lequel il a longtemps été confondu .Il peut également être utilisé coté serveur comme PHP.

* **Bootstrap**



Bootstrap est une collection d'outils utiles à la création du design (graphisme, animation et interactions avec la page dans le navigateur ... etc. ...) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option. C'est l'un des projets les plus populaires sur la plate-forme de gestion de développement GitHub.

## 5.6. Framework de développement

## 5.6.1. Pourquoi utiliser un Framework ?

Un Framework est un ensemble d'outils et de composants logiciels organisés conformément à un plan d'architecture et des patterns, l'ensemble formant ou promouvant un « squelette » de programme. Il est souvent fourni sous la forme d'une bibliothèque logicielle, et accompagné du plan de l'architecture cible du Framework.

Les avantages des Frameworks sont nombreux. En effet, un Framework est portable, de la part de son abstraction de la base de données et de la gestion générique du cache. Un Framework permet le développement des applications sécurisées. Grâce aux systèmes d'authentification, à la gestion des injections SQL ainsi qu'à la protection CSRF (Cross-Site Request Forgery) qui est gérée par la plupart des Framework. Les Framework sont des outils communautaires et ont, par conséquent, des forums, des listes de diffusion et des canaux IRC pour les soutenir. De plus vu que les Framework sont largement déployés, la chance de trouver les correctifs des problèmes rencontrés est plus grande.

5.6.2. Laravel

Comme Framework de développement nous avons utilisé Laravel. Laravel est un Framework web open-source écrit en PHP respectant le principe modèle-vue-contrôleur et entièrement développé en programmation orientée objet. Laravel est distribué sous licence MIT, avec ses sources hébergées sur GitHub. Laravel, créé par Taylor Otwel, initie une nouvelle façon de concevoir un Framework en utilisant ce qui existe de mieux pour chaque fonctionnalité.

En Mars 2015, Laravel est considéré comme l’un des plus populaires Framework PHP.

## 5.7. Quelques captures d’écrans