[Introduction Générale 7](#_Toc511392536)

[Chapitre 1 : Etude de l’existant 9](#_Toc511392537)

[1. Introduction 10](#_Toc511392538)

[2. Critique de l’existant : 12](#_Toc511392539)

[3. Problématique : 13](#_Toc511392540)

[4. Solution proposée : 13](#_Toc511392541)

[5. Méthodologies du travail 14](#_Toc511392542)

[5.1. Etude méthodologique 14](#_Toc511392543)

[5.1.1. Les méthodes agiles 14](#_Toc511392544)

[5.1.2. Le processus unifié 14](#_Toc511392545)

[5.1.3. Etude comparative des sous méthodes du processus unifié 15](#_Toc511392546)

[5.2. Critère de choix de la méthodologie 2TUP 16](#_Toc511392547)

[5.2.1. Principe de la méthodologie 2TUP 16](#_Toc511392548)

[6. Conclusion : 18](#_Toc511392549)

[Chapitre 2 : Analyse et spécification des besoins 19](#_Toc511392550)

[1. Introduction 20](#_Toc511392551)

[2. Branche fonctionnelle 20](#_Toc511392552)

[2.1. Les besoins fonctionnels 20](#_Toc511392553)

[2.2. Modélisation des besoins 21](#_Toc511392554)

[2.2.1. Les acteurs du système 21](#_Toc511392555)

[2.2.2. Spécifications fonctionnelles détaillées 22](#_Toc511392556)

[2.3. Raffinement des cas d’utilisation 23](#_Toc511392557)

[2.3.1. Raffinement de cas d’utilisation « créer un projet » 23](#_Toc511392558)

[2.3.2. Raffinement de cas d’utilisation « Générer un planning » 24](#_Toc511392559)

[2.3.3. Raffinement de cas d’utilisation « Générer un rapport de test » 25](#_Toc511392560)

[2.3.4. Raffinement de cas d’utilisation « Générer les documents financiers » 26](#_Toc511392562)

[2.4. Spécification détaillée des exigences 28](#_Toc511392563)

[2.4.1. Diagramme de séquence système « s’authentifier » 28](#_Toc511392564)

[2.4.2. Diagramme de séquence système « Ajouter utilisateur » 29](#_Toc511392566)

[2.4.3. Diagramme de séquence système « Ajouter un projet » 30](#_Toc511392567)

[2.4.4. Diagramme de séquence système « générer planning » 31](#_Toc511392569)

[2.4.5. Diagramme de séquence système « Générer un rapport de test » 32](#_Toc511392570)

[2.4.6. Diagramme de séquence système « générer document financier » 33](#_Toc511392571)

[2.5. Diagramme de classe d’analyse 34](#_Toc511392572)

[Figure 1 Nuxeo workflow 10](#_Toc511392756)

[Figure 2 Cycle de développement en Y (Processus 2TUP) 14](file:///C:\Users\HLOL\Desktop\Semestre2\RapportV1.docx#_Toc511392757)

[*Figure 3 : Diagramme de cas d'utilisation général* 20](#_Toc511392758)

[Figure 4 : raffinement de cas d'utilisation "créer un projet" 21](#_Toc511392759)

[Figure 5 : Raffinement de cas d'utilisation "Générer un planning" 22](#_Toc511392760)

[Figure 6 : Raffinement de cas d'utilisation "générer un rapport de test" 23](#_Toc511392761)

[Figure 7 : Raffinement de cas d'utilisation "générer les documents financiers" 24](#_Toc511392762)

[Figure 8 Diagramme de séquence système "s'authentifier" 26](#_Toc511392763)

[Figure 9 Diagramme de séquence système "Ajouter utilisateur" 27](#_Toc511392764)

[Figure 10 Diagramme de séquence système "ajouter un projet" 28](#_Toc511392765)

[Figure 11 Diagramme de séquence système « générer planning » 29](#_Toc511392766)

[Figure 12 Diagramme de séquence système "générer un rapport de test" 30](#_Toc511392767)

[Figure 13 Diagramme de séquence système « générer document financier » 31](#_Toc511392768)

[Figure 14 Diagramme de classe d'analyse 32](#_Toc511392769)

# **Introduction Générale**

L'introduction de l'informatique dans la vie professionnelle date de trente ans, celle de la micro-informatique de dix ans. Affaire de spécialistes au départ, orientée principalement vers l'automatisation des tâches de gestion administrative ou de contrôle de processus, elle est devenue plus récemment culture, outil privilégié de la modernisation des services, parfois vecteur du projet d'entreprise. Aujourd'hui, par décantation de l'expérience collective, se dégagent des lignes de force, des leçons générales, une méthodologie du bon usage de l'informatique. A la fois démythification et mise en valeur des acquis, un tel bilan devrait marquer la fin d'une période pionnière, sorte de ruée vers l'or. Maintenant s'ouvre un temps moins romantique, plus exigeant, sans doute encore plus efficace.

Désormais, l’informatique n’est plus seulement l’un des instruments de productivité, elle devient un outil de gestion et de pilotage de l’entreprise, voire un instrument stratégique apportant les moyens d’évolution des métiers de l’entreprise. On ne choisit donc plus les applications logicielles pour leur aptitude à remplir une fonction informatique précise, mais comme élément d’une chaîne de production de valeur.

Depuis maintenant plus d’un siècle, les grandes industries s’intéressent et s’investissent dans la gestion de projet. En effet, alors que l’origine du projet date du 14ème siècle où la phase de conception est séparée de la phase de réalisation lors de la construction d’un édifice, la « gestion » de projet n’apparaît qu’à la fin du 19ème siècle, où les entreprises et leurs affaires deviennent complexes. Elle prendra de l’ampleur au fur et à mesure des années où l’on verra apparaître les diagrammes de Gantt, les organigrammes PERT et la professionnalisation de cette discipline.

C’est dans ce cadre que s’inscrit notre projet de fin d’année afin de mettre en place un workflow de gestion et de suivi des projets informatiques.

Un workflow qui permet d’assurer l’échange des informations entre les différents utilisateurs et d'éliminer les échanges de documents dans des supports éparpillés ce qui facilite le suivi de l’état du processus métier de gestion de projet.

Le présent rapport comprend quatre chapitres dans lesquels nous essayons de synthétiser le travail réalisé tout au long de ce projet :

* Le premier chapitre intitulé « Etude de l’existant » permet d’étudier l’existant, le critiquer et proposer notre solution. Puis nous allons étudier la méthodologie du travail.
* Par la suite, Dans le deuxième chapitre « Analyse et spécification des besoins » nous allons étudier l’approche fonctionnels et non fonctionnels de la solution, ainsi que les choix technique et technologiques de l’application.
* Le troisième chapitre intitulé « Étude conceptuelle » nous allons présenter l’architecture générale. Elle sera réalisée en utilisant le langage de modélisation UML.
* Le quatrième chapitre « Réalisation » nous allons présenter quelques imprimés écrans de notre application.
* Et en fin nous clôturons par une conclusion générale dans laquelle nous décrivons les résultats ainsi que l’ouverture d’une perspective.

.

# Chapitre 1 : Etude de l’existant

## Introduction

Dans ce chapitre, nous Dans cette rubrique, nous allons analyser les solutions du marché dans l’objectif de réaliser une application moderne et plus performante critiquons et nous finissons par proposer notre solution.

* **Dans la plupart des entreprises :**

Le processus actuel de gestion de projet suivit par les entreprises est manuel et se compose des phases suivantes :

* **Phase d’étude :** traiter le cahier des charge**s** puis extraire les fonctionnalités et fixerun plan de réalisation à suivre et l’enregistrer dans un support papier.
* **Phase d’Achat et d’approvisionnement :** dégager les produits nécessaires et lesfournisseurs optimaux.
* **Phase de réalisation** : exécuter les tâches du projet et générer un rapport sous formatpapier critiquant le travail en détaillant chaque tâche.
* **Phase de Facturation et paiements**

Notre étude de l’existant ne se limite pas à étudier seulement le processus suivi par ces entreprises, mais aussi à étudier quelques solutions qui existent sur le marché pour ensuite en dégager ses points faibles et montrer en quoi notre solution est sophistiquée.

Il existe des plateformes open source qui permettent d’organiser et d’exécuter le processus de gestion de projet d’une manière meilleure que celle utilisée par la plupart des entreprises.

* **Sur le marché :**

**JIRA workflow**

JIRA workflow est un ensemble d’états et de transitions, un problème se déplace d’un état vers un autre dans le cycle de vie, on peut classer le workflow avec JIRA en deux catégories :

* **Default workflow**



C’est un workflow qui ne peut pas être modifié. Toutefois, vous pouvez copier et utiliser ce workflow pour créer votre propre processus.

* **Workflow personnalisé**

C’est l'une des caractéristiques principales de JIRA, la Création d’un workflow personnalisé permet JIRA de reproduire des processus internes spécifiques, À un niveau très élevé, les workflows peuvent être personnalisés pour différents projets et types d'émission.

**Nuxeo workflow**

Le moteur de workflow Nuxeo fournit les fonctionnalités habituelles que vous attendez d'un moteur de workflow et exploite les principaux modules de la plate-forme : le dépôt, le service d'automatisation, mises en page pour toutes les interactions de l'utilisateur et les mises en œuvre de processus. Il fournit des fonctionnalités de haut niveau en matière de gestion des tâches telles que les listes de tâches filtrables, des rappels, tâche réaffectation, la délégation des tâches, des rappels de tâches. Outre le moteur de workflow fournit des moyens pour mettre en œuvre la plupart des concepts BPMN : Fork, fusion, décision, point exclusivisme….

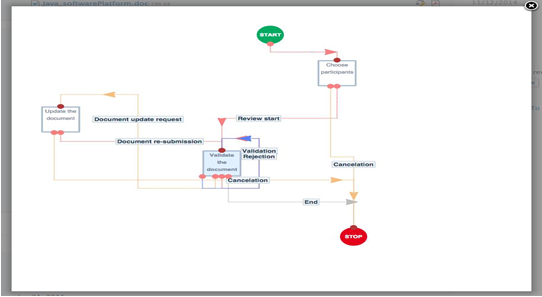
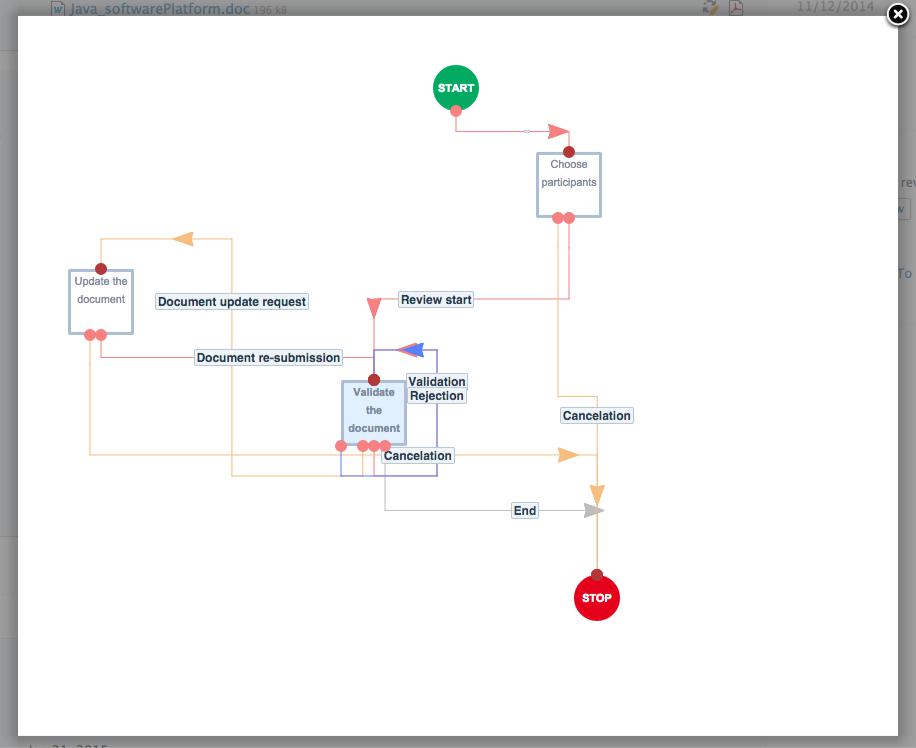
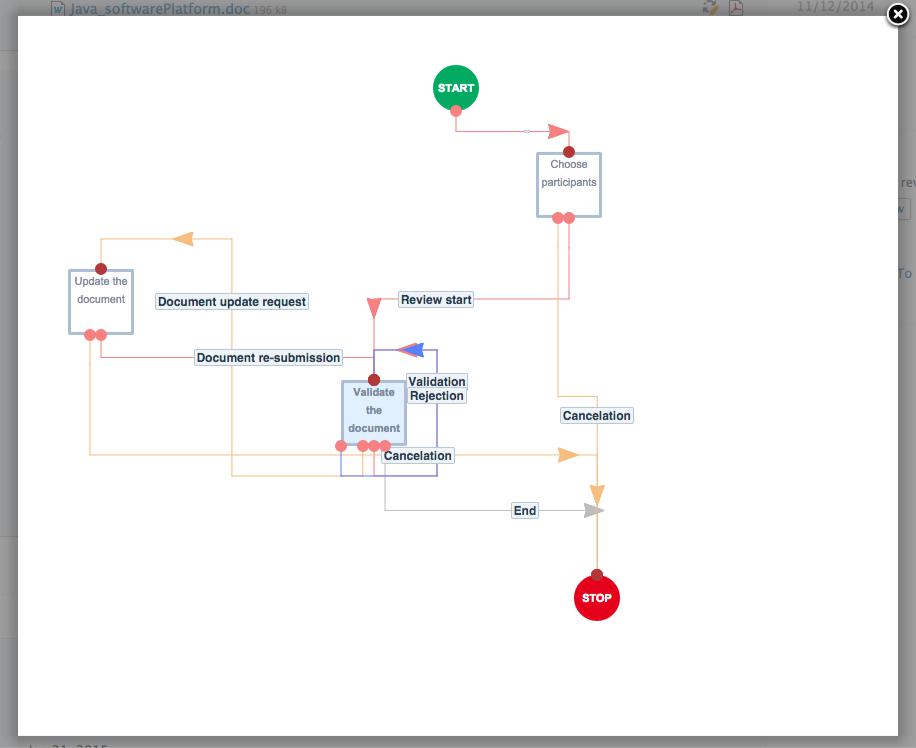


Figure 1 Nuxeo workflow

## Critique de l’existant :

Le tableau ci-dessous va nous permettre de bien extraire les points forts et les points faibles des solutions du marché :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Solution** | | **Points forts** | |  | **Points faibles** | |
|  | |  |  |  |  | |
| **JIRA workflow** | | -Rapidité d’exécution | |  | - fonctionnalités limitées | |
|  | | -facilité de l’utilisation | |  | -pas d’interfaces spécifiques à chaque tache. | |
|  | | - multiplateforme | |  | - pas de modifications tout au long du processus | |
|  | |  |  |  |  | |
| **Nuxeo workflow** | | -Rapidité d’exécution | |  | -Cout trop cher | |
|  | | - facilité de l’utilisation | | | - fonctionnalités limitées | |
|  | |  |  |  | -pas d’interfaces spécifiques à chaque tache. | |
|  | |  |  |  | - pas de modifications tout au long du processus | |
|  | |  |  | |  | |
|  |
|  | **Tableau 1 : Critique de l’existant** | | | | |

## Problématique :

Le processus actuel de gestion de projet de la plupart des entreprises se fait principalement à travers des supports semi-papier ou papier, ces moyens empêchent souvent le bon déroulement de tout le processus dans ses différentes phases, ainsi, la communication et la collaboration inter-membres n’est plus organisée de la manière nécessaire à la réussite d’un projet.

Cette méthode limitée peut être suffisante pour une société de petite taille mais avec la croissance de la valeur sur le marché, l’augmentation du nombre de ses projets et sa brillance dans le domaine d’informatique, l’excellence et la perfection deviennent des exigences et ne pas des options.

Pour remédier à ces défaillances, et afin de renforcer ses positions dans le marché et d’évoluer ses processus de gestion des projets et rendre ses productivités plus efficace et plus rentable (rendre le processus métier flexible) ainsi qu'améliorer ses qualités de service, nous avons proposé une solution permettant la réalisation d’un système informatique qui intervient dans toutes les phases de réalisation d’un tel projet, depuis l’identification des besoins des clients jusqu’à la livraison.

## Solution proposée :

Dans le but de remédier aux problèmes énumérés précédemment, ne nécessité d’avoir un outil de gestion de projet s’impose, permettant la centralisation des données et facilitant le travail collaboratif en temps réel entre les différents intervenants sur un même projet. C'est dans ce sens que l'idée de la plateforme *PROJECTMANAGER.FLOW* prend naissance afin de répondre à plusieurs besoins et de pallier plusieurs contraintes spatiales (éviter les supports de stockages traditionnels), temporel, disponibilité et les limites des solutions existantes. Les technologies utilisées ont rendu notre solution plus performante et avantageuse par rapport aux autres solutions.

Cette solution doit assurer les objectifs suivants :

* Le développement et l’exécution du processus de gestion des projets qui commence par l’importation du cahier de charge, passant par l’extraction des taches à faire, génération du document financière et du contrat, détermination des ressources en faveur du projet, arrivant au planning des taches et estimation du cout et du temps finissant par la réalisation, l’extraction du rapport de test et la gestion de facturation.
  + L’intégration au sein du système grâce à un système de notification

## Méthodologies du travail

Afin d’aboutir à un logiciel à la hauteur répondant aux besoins, le génie logiciel offre des méthodologies et des normes pour remédier à ces problèmes. Bien qu’il existe une multitude de méthodologies de développement logiciel, le choix d’une méthode pour un projet donné devient une décision de plus en plus draconienne.

### Etude méthodologique

Plusieurs études ont montré l’efficacité des méthodes itératives et incrémentales pour produire un logiciel de qualité répondant aux besoins du client, donc nous avons opté à ces méthodes qu'on vient maintenant d'en parler plus en détails.

#### Les méthodes agiles

Elles reposent sur des cycles de développement itératifs et adaptatifs en fonction des besoins évolutifs du client. Elles permettent d'impliquer le client dans la réalisation de l’application. Ces méthodes consacrés de mieux répondre aux besoins du client en un temps limité (Scrum, XP, RUP…).

#### Le processus unifié

Le processus unifié est une méthodologie itérative et incrémentale, guidée par les cas d’utilisation, pilotée par l’architecture et orientée risque.

Le tableau ci-dessous montre un récapitulatif des différences entre les deux méthodes :

|  |  |
| --- | --- |
| **Agile** | **Processus unifié** |
| Faible formalisme | Extrêmement formaliste |
| Concentré sur le code exécutable | Production de nombreux artefacts |
| Choix idéal pour les petits projets | Pour les projets complexes |

**Tableau 2 : Tableau comparatif entre les méthodes agiles et le processus unifié**

Vu le caractère conceptuel et technique de ce travail, nous allons utiliser les méthodes descriptives et analytiques basées sur les principes de modélisation UML 2.0 issu d’UP (Unified Process).

#### Etude comparative des sous méthodes du processus unifié

Ci-dessous un tableau qui résume cette comparaison :

|  |  |  |  |
| --- | --- | --- | --- |
| **Sous méthode** | **Description** | **Avantages** | **Limites** |
| **Cascade** | -Les phases du projet se déroulent séquentiellement | -Phases de projet claires. | -Non itératif  -Pas de modèles de documents |
| **2TUP** | -S'articule autour de l’architecture.  -Cycle en V. | -Itératif  -La technologie et la gestion des risques sont les plus importants. | -Plutôt superficiel sur les phases  -Pas de documents types. |
| XP (eXtreme Programming) | -Ensemble de «  Meilleurs pratiques » de développement  -Efficace pour les projets de moins de 10 personnes | -Itératif  -Mise en œuvre simple.  -Fait une grande importance aux aspects techniques  Innovante | -Pas de phase en amont et en aval au développement  -très floue dans sa mise en œuvre :quels intervenants, quels livrables ? |
| **RUP** | -Outil prêt à l’emploi  -Destiné pour les grands projets | -Itératif et incrémental.  -Piloté par les cas d’utilisation.  -Réduction des risques  -Qualité acceptable | -Axé processus  -Cout élevé. |

**Tableau 3: Tableau Comparatif entre les différentes méthodologies**

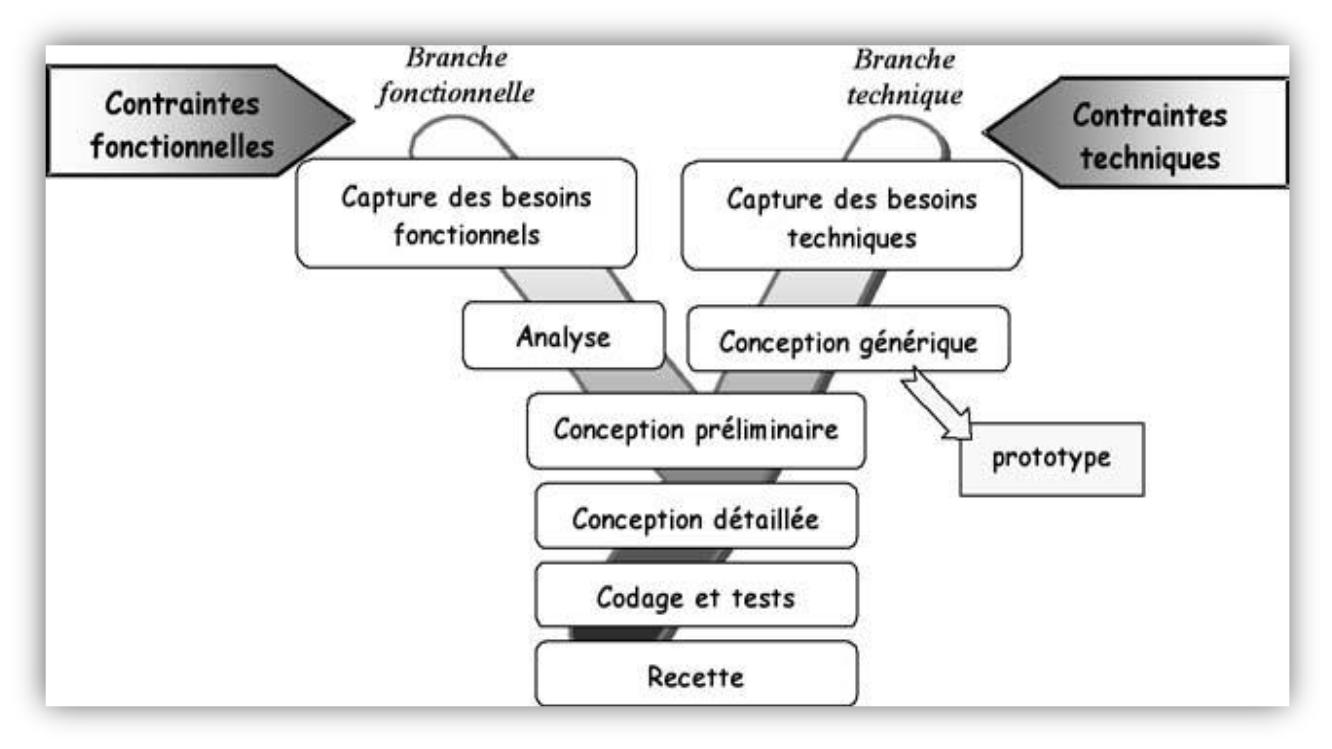
### Critère de choix de la méthodologie 2TUP

D’après le tableau comparatif, nous avons opté pour le processus 2TUP pour des raisons multiples. D’une part, 2TUP donne une grande importance à la technologie ce qui est important pour notre projet, d’autre part, 2TUP est un processus en Y composé par une branche technique et branche fonctionnelle.

#### Principe de la méthodologie 2TUP

Ce processus commence par une étude préliminaire qui permet d’identifier les acteurs du système à mettre en œuvre qui est considéré comme une boite noir tout en présentant les différents messages entre les utilisateurs et ce système et d’élaborer le cahier des charges. La figure 4, montre les différentes étapes de processus 2TUP :

**Figure 2 Cycle de développement en Y (Processus 2TUP)**



D’après la figure ci-dessus, on remarque que 2TUP est composé essentiellement de trois branches:

* + **Une branche fonctionnelle (à gauche) :**
* Capture des besoins fonctionnels, qui produit les modèles des besoins en se basant sur le métier des utilisateurs. Cette étape élimine le risque d’avoir un système inadapté aux besoins des utilisateurs.
* L’analyse qui consiste à étudier les besoins fonctionnels pour avoir une idée sur le traitement métier du système.
* **Une branche technique (à droite) :**

Capture des besoins techniques, cette étape permet de recenser les contraintes de choix de dimensionnement et de conception du système. Les outils sélectionnés ainsi que les contraintes d’intégration (pré requis d’architecture technique).

* L’étape conception générique définit ensuite les composants nécessaires à la construction de l’architecture technique. Cette conception est complètement indépendante des aspects fonctionnels.
* **Une branche de réalisation (au milieu) :**
* La conception préliminaire, elle intègre le modèle d’analyse fonctionnel dans l’architecture technique de comprendre les composants du système à développer.
* La conception détaillée, permettant d’analyser chaque composant.
* L’étape de codage, c’est l’étape de réalisation de ces composants ainsi que les tests unitaires sur chaque composant.
* L’étape de recette, qui consiste enfin à la validation des différentes fonctionnalisées du système.

## Conclusion :

L'étude de l'existant nous permet de bien analyser les procédures et la démarche du processus de Gestion de projet à implémenter et nous présente clairement la problématique ainsi que la solution proposée qui sera élaborée conformément aux principes de la méthodologie 2TUP.

# Chapitre 2 : Analyse et spécification des besoins

## Introduction

Pour extraire l’ensemble des fonctionnalités que doit répondre notre application, ainsi que les différentes contraintes, nous allons consacrer ce chapitre pour l’analyse ainsi que la spécification des besoins

## Branche fonctionnelle

L’analyse de ce sujet nous a permis d’identifier les différents besoins auxquels doit répondre notre application. Les besoins de la branche fonctionnelle sont seulement les besoins fonctionnels.

### Les besoins fonctionnels

L’analyse fonctionnelle est indispensable pour une bonne réalisation d’un projet, pour cela, nous commençons par la présentation des différents besoins fonctionnels qui doivent répondre aux exigences du futur système. Les besoins recensés sont comme suit :

* La création d’un nouvel utilisateur et ajouter leur rôle et leur groupe.
* La gestion des utilisateurs.
* La création d’un nouveau projet et l’ajout de ses informations et clients.
* L’ajout du cahier de charge et des besoins du client.
* La gestion des tâches du projet, l’estimation du temps ainsi que l’ordre d’exécution.
* La création du devis.
* La création de contrat.
* La planification de la partie réalisation (les tâches mentionnées) par la génération d’un diagramme de Gantt.
* L’affectation des acteurs nécessaires pour le projet (la liste des utilisateurs, la liste des fournisseurs) et la liste des produits qu’on a besoin.
* L’automatisation des tâches du projet (un workflow appart).
* La génération d’un rapport de qualité avant de valider la partie réalisation.
* La création d’une facture et livraison du projet.

### Modélisation des besoins

#### Les acteurs du système

Un acteur est un élément qui communique directement avec l’application. Après avoir étudier les différentes interactions internes et externes du système, nous avons pu dégager ces acteurs ainsi que leurs interactions:

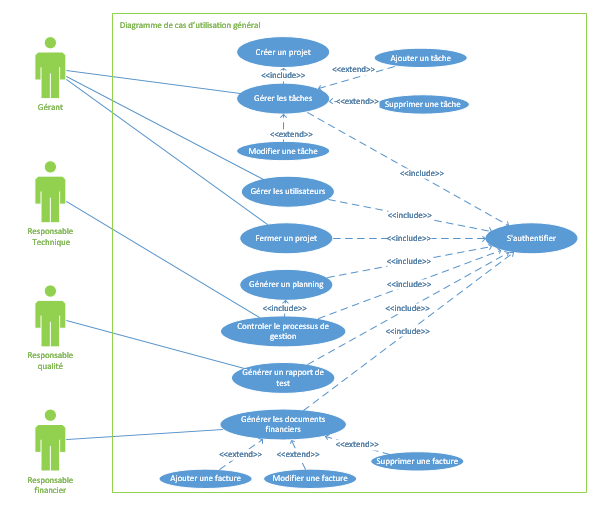
|  |  |
| --- | --- |
| Acteurs | Besoins fonctionnels |
| Le gérant | -s’authentifier  -créer un nouveau projet  -ajouter le cahier de charge et les besoins de client.  -ajouter les taches du projet et affecter un contrat de travail.  -affecter les acteurs nécessaires pour le nouveau projet.  -gérer la liste des utilisateurs. |
| Le directeur Technique | -s’authentifier  -générer un planning (diagramme de Gant).  -contrôler l’avancement de réalisation avec un moteur de workflow qui automatise la gestion des taches du projet. |
| Le responsable de qualité | -s’authentifier  -générer un rapport sur la phase de réalisation et validation du travail. |
| Le responsable financier | -s’authentifier  -générer les opérations de financement (devis, facture). |

**Tableau 4 : Liste des acteurs de notre système**

#### Spécifications fonctionnelles détaillées

Les cas d’utilisation permettent de comprendre les relations fonctionnelles entre les acteurs et le système. UML propose des concepts issus de bonnes pratiques pour la représentation d’un cas d’utilisation.

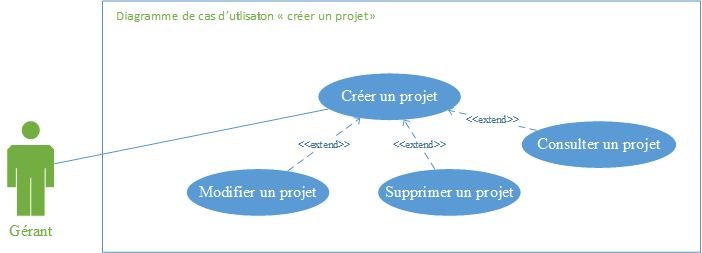
Les figures ci-dessous présentent les diagrammes de cas d’utilisation, ils décrivent la majorité des fonctionnalités du système de point de vue des acteurs, mais ne décrivent pas de façon détaillée l’interaction entre les acteurs.

***Figure 3 : Diagramme de cas d'utilisation général***

### Raffinement des cas d’utilisation

Au cours de cette activité nous allons détailler les cas d’utilisations dégagés grâce auxbesoins fonctionnels vus dans le premier chapitre. Pour chaque cas, nous allons parlertextuellement des préconditions et des post-conditions liées au cas courant ainsi qu’on vadécrire le scénario de base, éventuellement les exceptions.

#### Raffinement de cas d’utilisation « créer un projet »

****

**Figure 4 : raffinement de cas d'utilisation "créer un projet"**

La figure ci-dessus représente le cas d’utilisation « Créer un projet » raffiné.

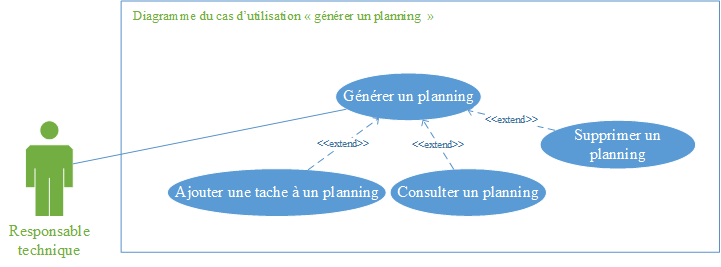
En effet, l’accès ne se fait qu’après authentification. Le gérant, qui est dans notre cas le chef de projet, ajoute un nouveau projet, le consulte, le modifie ou le supprime.

* **Description textuelle de cas d’utilisation « Créer un projet » :**

|  |  |  |
| --- | --- | --- |
| **Titre** | **Créer un projet** | |
| **Résumé** | Détaille les étapes nécessaires pour ajouter un projet | |
| **Acteur** | Gérant | |
| ***DESCRIPTION D’ENCHAINEMENT*** | | |
| **Préconditions** | | **Post-conditions** |
| Le gérant s’authentifie | | Un nouveau projet ajouté |
| ***SCENARIO PRINCIPAL*** | | |
| 1. L’utilisateur demande d’ajouter un nouvel projet 2. L’utilisateur remplit le formulaire concernant le projet 3. Consulter la liste des clients 4. Sélectionner un client de la liste des clients 5. L’utilisateur confirme les informations ajoutées | | |
| ***ENCHAINEMENT ALTERNATIF*** | | |
| Enchainement alternatif A1 :  4) Choisir d’ajouter un nouveau client.   1. L’utilisateur remplit le formulaire pour ajouter un nouveau client 2. Il confirme l’ajout | | |

**Tableau 5 : « créer un projet »**

#### Raffinement de cas d’utilisation « Générer un planning »



**Figure 5 : Raffinement de cas d'utilisation "Générer un planning"**

La figure ci-dessus, concerne le cas d’utilisation « générer un planning », maintenant c’est au  
responsable technique d’orchestrer les tâches et y préparer un planning garantissant le bon  
déroulement du travail dans les délais prédéfinis, et ce par le fait d’ajouter, consulter ou  
modifier un planning.

* **Description textuelle de cas d’utilisation « Générer un planning » :**

|  |  |  |
| --- | --- | --- |
| **Titre** | **Générer un planning** | |
| **Résumé** | Détaille les étapes nécessaires pour ajouter un planning (diagramme deGantt) | |
| **Acteur** | Responsable technique | |
| ***DESCRIPTION D’ENCHAINEMENT*** | | |
| **Préconditions** | | **Post-conditions** |
| La liste des tâches est ajoutée | | Un diagramme de Gantt est généré |
| ***SCENARIO PRINCIPAL*** | | |
| 1. Consulter la liste des tâches 2. Ajouter un diagramme de Gant | | |

Tableau 8 : « Générer un planning »

#### Raffinement de cas d’utilisation « Générer un rapport de test »

#### C:\Users\HLOL\Conception\ucrapport.png

Figure 6 : Raffinement de cas d'utilisation "générer un rapport de test"

#### La figure ci-dessus, concerne le cas d’utilisation « générer un rapport de test », qui est affecté au responsable de qualité, il s’implique à la fin de chaque projet pour relever un rapport de test au gérant lui permettant d’avoir une vision microscopique sur le pourcentage d’accomplissement des tâches.

• **Description textuelle de cas d’utilisation « Générer un rapport de test »:**

|  |  |  |
| --- | --- | --- |
| **Titre** | **Générer un rapport de test** | |
| **Résumé** | Détaille les étapes nécessaires pour générer un rapport de test | |
| **Acteur** | Responsable de qualité | |
| ***DESCRIPTION D’ENCHAINEMENT*** | | |
| **Préconditions** | | **Post-conditions** |
| La phase de réalisation est terminée | | Un rapport de test est ajouté |
| ***SCENARIO PRINCIPAL*** | | |
| 1. Consulter la liste des projets. 2. Choisir le projet à tester. 3. Consulter la liste des tâches. 4. Noter chaque tâche. 5. Générer le rapport de test (des charts). | | |

Tableau 9 : générer un rapport de test

#### Raffinement de cas d’utilisation « Générer les documents financiers »

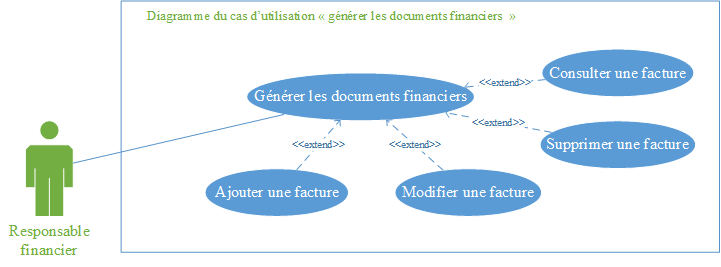


Figure 7 : Raffinement de cas d'utilisation "générer les documents financiers"

La figure ci-dessus, concerne le cas d’utilisation « générer les documents financiers », ce dernier volet est accordé au responsable financier, qui génère un devis puis le passer en facture pour le fournir au client.

* **Description textuelle des cas d’utilisations « générer les documents financiers »**

|  |  |  |
| --- | --- | --- |
| **Titre** | **Générer les documents financiers** | |
| **Résumé** | Détaille les étapes nécessaires pour générer les factures | |
| **Acteur** | Responsable financier | |
| ***DESCRIPTION D’ENCHAINEMENT*** | | |
| **Préconditions** | | **Post-conditions** |
| Un projet crée | | Un document financier ajouté |
| ***SCENARIO PRINCIPAL*** | | |
| 1. Consulter la liste des projets. 2. Choisir un projet. 3. Choisir le document financier à ajouter. 4. Remplir le formulaire. 5. Confirmer le remplissage et générer le document. | | |

**Tableau 10 : générer un document financier**

### Spécification détaillée des exigences

#### Diagramme de séquence système « s’authentifier »

#### C:\Users\HLOL\Conception\auth.png

Figure 8 Diagramme de séquence système "s'authentifier"

Pour passer à exploiter les fonctionnalités du système, il faut tout d’abord se connecter comme étant utilisateur. En cas de mauvais renseignement de paramètres d’authentification, le système va notifier l’utilisateur, et l’interface de connexion s’auto initialise de nouveau.

#### Diagramme de séquence système « Ajouter utilisateur »

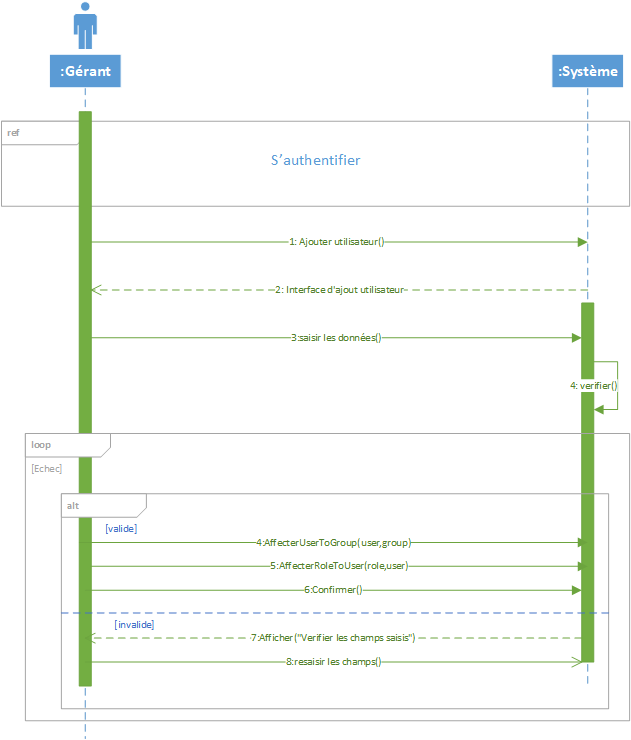


Figure 9 Diagramme de séquence système "Ajouter utilisateur"

Ce diagramme de séquence système présente le scénario à suivre pour ajouter un nouvel utilisateur à la base de données.

#### Diagramme de séquence système « Créer un projet »

#### C:\Users\HLOL\Conception\addprj.png

Figure 10 Diagramme de séquence système "créer un projet"

L’ajout des projets est accordé au gérant, donc le système va réagir aux demandes de création de projet comme suit : Un utilisateur connecté demande d’ajouter un projet, le système vérifie s’il possède les droits de création, si non, il sera bloqué par une alerte, sinon, il sera invité à saisir les données et les ressources nécessaires, en cas de mal remplissage, le système ne cesse de signaler à l’utilisateur.

#### Diagramme de séquence système « générer planning »

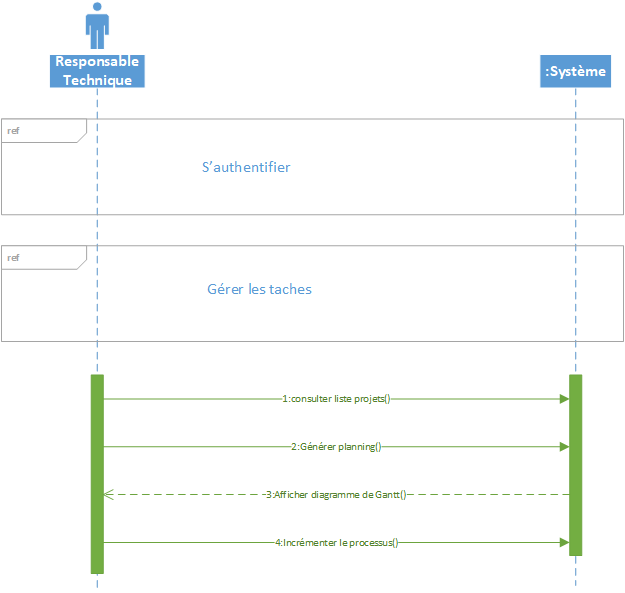


Figure 11 Diagramme de séquence système « générer planning »

#### Diagramme de séquence système « Générer un rapport de test »

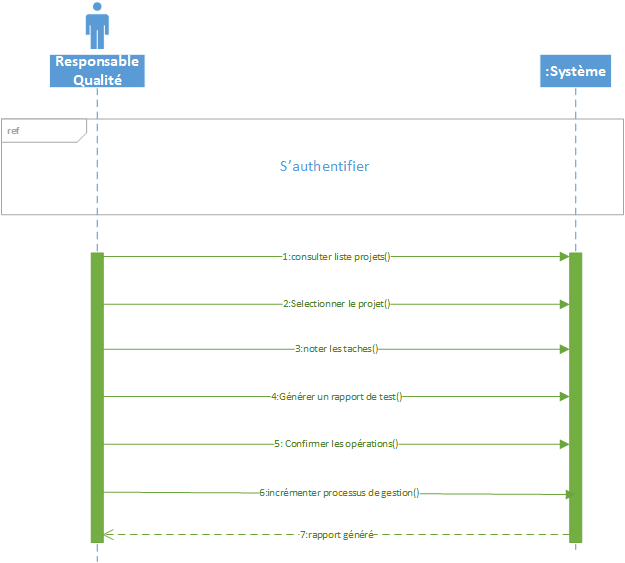


Figure 12 Diagramme de séquence système "générer un rapport de test"

#### Diagramme de séquence système « générer document financier »

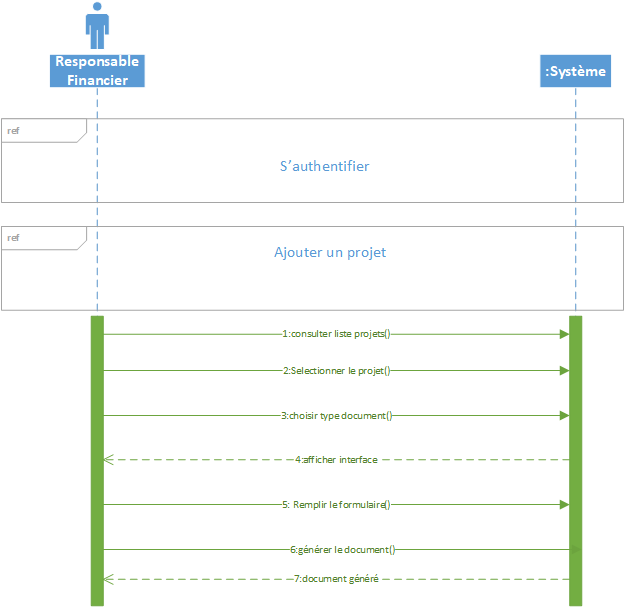


Figure 13 Diagramme de séquence système « générer document financier »

L’utilisateur consulte la liste des projets, sélectionne un projet, puis créé un devis pour finalement le générer.

### 5. Branche techniques

Pour réaliser notre projet nous avons appliqué des ressources techniques que nous allons les présenter par la suite.

#### 5.1. Les besoins techniques

Etant donné qu’une application fonctionnelle et opérationnelle ne garantit pas la satisfaction et la fidélité des utilisateurs, nous devons prendre en considération des critères non fonctionnels lors de la conception et de l’implémentation de notre solution. Parmi ces exigences, certains sont basiques, dont tout produit informatique doit doter. Nous citons :

* **Architecture en couches** :

Nous allons classer l’application en des couches, nous utilisons la technologie EJB (Entreprise Java Bean) pour assurer la communication avec la base de données, qui est composée de deux modules les entités qui représentent les tables de la base de données, le parton DAO qui assure l’accès aux données, ainsi nous avons utilisé pour les traitements métiers le Framework JSF avec Spring, pour la couche présentation nous avons utilisé le framework interfaces utilisateur de JSF PrimeFaces

* **La Sécurité** : nous avons utilisé le Spring Security comme un outil de protectioncontre les accès non autorisés.
* **La Compréhension** : les interfaces graphiques se sont organisés d’une façon pluslisible et clair pour faciliter la compréhension et l’utilisation et pour nous avons utilisé la Framework Primefaces 6 de JSF.
* **Apprentissage** : La facilité pour un utilisateur à utiliser le produit (la tâche de gestiondes projets (métier) est clair et ses instructions sont mentionnées ainsi ne complique pas les interfaces d’interactions avec l’utilisateur de l’application).

### Choix technologique

* **JEE (Java Entreprise Edition)** : un ensemble de spécifications pour plate-forme javad’oracle destinée au application d’entreprise
* **Maven :** conçu pour supprimer les tâches difficiles du processus de buildpar l’utilisationd’une approche déclarative
* **Java Server Faces:** Ou JSF est un Framework java utilisé pour développer des applicationsweb. JSF est un Framework MVC basé sur la notion des composants.
* **Hibernate :** Framework open-source Java ayant pour gérer la persistance des données debase ainsi que la notion d’ORM.
* **Spring Security 3.2.5 :** Framework de sécurité et contrôle d’accès aux applications web.

**Conclusion**

Durant ce chapitre, nous avons analysé les besoins non fonctionnels et fonctionnels de notre application. Ces analyses sont devenues formelles par la suite à l’aide des diagrammes des cas d’utilisation. Les interactions utilisateurs système sont décrites dans des tableaux spécifiques Le chapitre qui suit va être consacré pour l’étude conceptuelle de notre application.

# Chapitre 3 : Etude conceptuelle

## Introduction

La conception permet de décrire le fonctionnement désiré du système afin d'en faciliter la réalisation et la maintenance. Pour ce faire, la première partie dédiée à l'exposition du choix de méthode de conception ainsi que l'architecture du système à développer. La deuxième partie présente la conception détaillée de notre système et la base de données utilisée.

## Conception détaillée

### Les patrons de conception

En génie logiciel, un patron de conception vise à résoudre les problèmes selon l’approche orienté objet. Un modèle de conception peut être vu comme un document contenant la description de la structure d’un ensemble de classes permettant de répondre à une situation particulaire. Le but principal des patrons de conception est de minimiser le temps de développement d’un logiciel tout en augmentant la qualité de résultat.

* **MVC (Model-View-Controller) :**

Le modèle vue contrôleur est un modèle de conception responsable de l’organisation de l’interface homme machine (IHM) d’un logiciel. En résume, lorsqu’un utilisateur envoie une requête, elle va être envoyée par la « vue » et analysé par le « contrôleur ». En suite le contrôleur demande au model approprié d’effectuer l’ensemble des traitements nécessaires et notifie la vue que la requête est effectuée. Enfin une notification est effectuée vers le modèle pour se mettre à jour.

### Modélisation du Processus métier



Figure 14 Processus métier

Ce processus représente les différentes tâches pour gérer les phases du projet. Ce processus métier va permettre à l’utilisateur de contrôler l’avancement de ses projets, en commençant par la création du projet et il va ajouter les différents documents nécessaires à la réalisation du projet et aussi il assure la gestion des tâches. Et Enfin, il fournit un rapport de test à l’utilisateur.

## Diagramme de classe

Le diagramme de classe exprime l’aspect statique du système en termes de classes et de relations entre ces classes. Il nous permet de présenter la structure interne du système et d’organiser le travail de développeur. Nous avons utilisé dans notre diagramme de classe les concepts suivant Classe : une classe est une description abstraite (condensée) d’un ensemble d’objets similaires du point de vue de leur structure (attributs), leur comportement (opérations) et leurs relations à l’extérieur (association). L’association : une association exprime une connexion sémantique bidirectionnelle entre classes. C’est une abstraction des liens qui existent entre les objets instances des classes associées.

### C:\Users\HLOL\Conception\dc.png

Figure 15 Diagramme de classe

### 

## Diagramme de séquence détaillé

#### Diagramme de séquence « Ajouter utilisateur »

#### Diagramme de séquence « Ajouter projet »

#### Diagramme de séquence « Ajouter cahier des charges »

## Diagramme d’activité

A ce niveau, Il est intéressant de comprendre la logique comportementale de la solution à proposer. Le diagramme d’activité est un procédé qui permet de d’écrire cet aspect.

#### Diagramme d’activité du cas d’utilisation « S’authentifier »

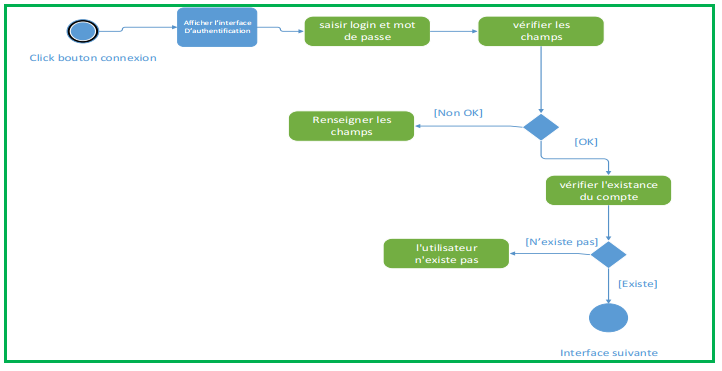


Figure 16 Diagramme d’activité du cas d’utilisation « S’authentifier »

#### Diagramme d’activité du cas d’utilisation « gestion projet »

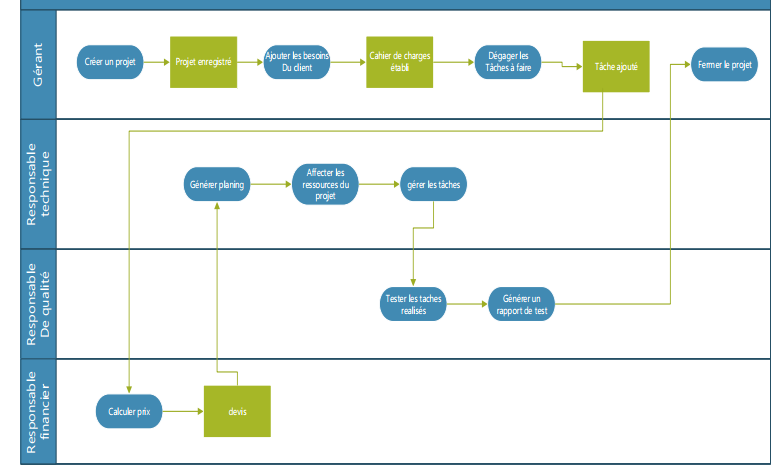


Figure 17 Diagramme d’activité du cas d’utilisation « gestion projet »

Le diagramme d’activités du système montre l’enchainement et le processus de gestion des projets il comporte quatre acteurs chacun a ses activités. Ceci est entamé et clôturé par le gérant avec ses deux activités (créer un projet, fermer un projet).

Conclusion**:**

Ce chapitre nous a permis de détailler la conception statique de notre futur système, une conception globale à travers un diagramme des composants et une conception détaillée à l’aide des diagrammes des classes, ainsi que la conception dynamique, par le biais des diagrammes des séquences et d’activités.

Cette décomposition conceptuelle va être très utile lors de la réalisation de notre système qu’on va l’aborder dans le chapitre suivant.

# Chapitre 4 : Réalisation

## Introduction

Nous entamons cette partie par la présentation de l’environnement de travail logiciel et matériel. Puis, nous exposons la démarche suivie pour la réalisation.

## Environnement de Travail

En premier lieu, nous allons présenter l’environnement de développement (que ce soit matériel ou logiciel) avec lequel nous avons travaillé.

### Environnement matériel

Les différentes phases de notre projet à savoir la documentation, la spécification des besoins, la conception et le développement, sont réalisés à l’aide d’un ordinateur disposant des caractéristiques suivantes:

* Modèle: ASUS X541U
* Processeur: intel core i7
* Mémoire vive : 8 GO
* Disque dur : 100 GO.
* Système d'exploitation: Windows 10 pro.

### Environnement logiciel

**Kit de développement JDK 1.8 :** JDK est l’environnement dans lequel le code Java estcompilé.

**Netbeans 8.2 :** Netbeans est un environnement de développement (IDE) destinéaux plusieurs langages y compris le langage java.

**GlassFish Server 4.1.1** est serveur d’application Java EE libre écrit en Java, c'est un support complet de Java EE 6 qui est proposé : EJB3, JPA2, JSF2…

**Microsoft Visio 13** :Microsoft Visio (officiellement Microsoft Office Visio) estun [logiciel](https://fr.wikipedia.org/wiki/Logiciel) de [diagrammes](https://fr.wikipedia.org/wiki/Diagramme) et de [synoptiques](https://fr.wikipedia.org/wiki/Synoptique) pour [Windows](https://fr.wikipedia.org/wiki/Microsoft_Windows) qui fait partie de la suite bureautique [Microsoft Office](https://fr.wikipedia.org/wiki/Microsoft_Office) mais se vend séparément.

## Présentation des interfaces