# *EPIGRAPHE*

« Peu importe si vous ne voyez pas tout l’escalier, gravissez juste la première marche avec confiance, puis encore et encore. Car au cœur de toute difficulté, se trouve une belle opportunité à saisir … »

**Martin Luther King.**

# *DEDICACE*

A nos familles respectives, autant de phrases et d’expressions aussi éloquentes soit-elle ne sauraient exprimer notre gratitude et notre reconnaissance. Vous avez su nous inculquer le sens de la responsabilité, de l’optimisme et de la confiance en soi face aux difficultés de la vie. Vos conseils ont toujours guidés nos pas vers la réussite. Votre patience sans fin, votre compréhension et votre encouragement sont pour nous le soutien indispensable que vous avez toujours su nous apporter. Nous vous devons ce que nous sommes aujourd’hui et ce que nous serons demain et nous ferons toujours de notre mieux pour rester votre fierté et ne jamais vous décevoir. Que le Dieu Tout Puissant vous préserve et vous accorde la santé.

***Nous vous dédions ce travail.***



# *REMERCIEMENTS*

Tout d'abord, nous rendons grâce et honneur à notre Seigneur Jésus-Christ pour la santé, la force et le courage de mener à bien ce travail.

*‘’La reconnaissance est la mémoire du cœur’’*. **Hans Christian Andersen**

Par ses mots, nous voudrions manifester notre profonde gratitude et reconnaissance à tous ceux qui, d’une façon quelconque, ont contribué à la réalisation de cette œuvre. Ainsi nos sincères remerciements s’adressent :

* Monsieur le Directeur général de **Techno Méga Partners** pour nous avoir donner l’opportunité de mener ce travail au sein de son entreprise. Nous le remercions pour toute la confiance qu’il a placé en nous.
* A l’ensemble du personnel de **Techno Mega Partners** pour la franche collaboration et le soutien qu’ils nous ont apporté.
* Au corps administratif et enseignant de l’**IAI** pour le dévouement et la qualité de ses enseignements.
* A **l’Ecole du dimanche** (**ECODI**) pour son soutien spirituel et moral.
* A tous nos collègues de promotion pour leur solidarité.
* Aux **Communautés des Etudiants Tchadiens et Congolais** ainsi qu’à toute la **communauté estudiantine de l’IAI** qui ont facilité notre intégration au sein de l’institut et la bonne ambiance entretenue tout au long de notre cycle.

# *AVANT-PROPOS*

Une décennie après les indépendances de certains Etats de l’Afrique, onze(11) chefs d’Etats de l’Afrique francophone, dans le soucis d’intégration et de coopération Sud-Sud décident de créer à fort-Lamy actuel N'Djaména au Tchad en 1971 une institution capable de former des cadres supérieurs dans le domaine de l’informatique : de créer une Ecole inter-Etats d'Enseignement Supérieur spécialisée en informatique dénommée Institut Africain d'Informatique (I.A.I) par la Résolution N°33/AEFT/Fort-Lamy du 29 janvier 1971 : C’est la naissance de l’institut Africain d’Informatique qui regroupe le Gabon (pays hôte), le Cameroun, le bénin, le Tchad, le Togo, la cote d’ivoire, le Congo, le Niger, la république centrafricaine, le Sénégal et le Burkina Faso.

De sa création à 1987, sa vocation était de former des analystes programmeurs. En vue des résultats encourageants et face à la nécessité de mettre à la disposition des entreprises et des administrations dans la sous-région des compétences de haut niveau en informatique, l’IAI a ouvert depuis 1987 un second cycle d’Ingénieur.

Suivant la recommandation du conseil scientifique du 29 juin au 02 juillet 1998, le cycle Licence forme en métier d’Analyste programmeur en Informatique. Deux parcours sont proposés : le parcours Génie logiciel et systèmes Informatique(GLSI), le parcours Administration système et réseau(ASR).

Dans le cadre de l’obtention du diplôme de fin de formation du cycle d’Analyste-Programmeur, une formation théorique devra être complétée par un projet de fin d’étude d’une durée de quatre (4) mois en entreprise ou dans un centre de recherche.

Ce projet a pour objectif l’insertion des étudiants en milieu professionnel, notamment dans le domaine d’ingénierie informatique afin de développer certaines compétences en matière de conception et de développement d’application et s’adapter à l’environnement professionnel.

**Table des matières**

[*EPIGRAPHE* 1](#_Toc35536508)

[*DEDICACE* 2](#_Toc35536509)

[*REMERCIEMENTS* 3](#_Toc35536510)

[*AVANT-PROPOS* 4](#_Toc35536511)

[INTRODUCTION 9](#_Toc35536512)

[*PARTICIPANTS AU PROJET* 10](#_Toc35536513)

[PARTIE A 11](#_Toc35536514)

[PRESENTATION DE LA STRUCTURE D’ACCUEIL ET DU SUJET 11](#_Toc35536515)

[CHAPITRE 1 : PRESENTATION DE LA STRUCTURE D’ACCUEIL 12](#_Toc35536516)

[I. Présentation 12](#_Toc35536517)

[II. Organigramme 12](#_Toc35536518)

[III. Localisation 13](#_Toc35536519)

[1. Situation géographique 13](#_Toc35536520)

[2. Contacts 13](#_Toc35536521)

[CHAPITRE 2 : PRESENTATION DU SUJET 14](#_Toc35536522)

[I. Le sujet et son contexte 14](#_Toc35536523)

[II. Problématique 14](#_Toc35536524)

[III. Objectifs visés et résultat attendus 15](#_Toc35536525)

[CHAPITRE 3 : CONCEPTS LIES AU DOMAINE D’ETUDE 16](#_Toc35536526)

[PARTIE B 21](#_Toc35536527)

[SOLUTION 21](#_Toc35536528)

[CHAPITRE 1 : ETUDE DE L’EXISTANT 22](#_Toc35536529)

[I. Présentation de l’existant 22](#_Toc35536530)

[II. Critique de l’existant 22](#_Toc35536531)

[CHAPITRE 2 : PROPOSITION DES SOLUTIONS 23](#_Toc35536532)

[I. Proposition des solutions 23](#_Toc35536533)

[1. Première solution 23](#_Toc35536534)

[2. Deuxième solution 23](#_Toc35536535)

[II. Choix et justification de la solution 24](#_Toc35536536)

[CHAPITRE 3 : CHOIX DU CONCEPT ET DE L’ARCHITECTURE DE LA SOLUTION RETENUE 25](#_Toc35536537)

[I. Client lourd/client léger 25](#_Toc35536538)

[II. Concept client/serveur 26](#_Toc35536539)

[III. Architecture client/serveur 26](#_Toc35536540)

[1. 1-tier 27](#_Toc35536541)

[2. 2-tier 27](#_Toc35536542)

[3. 3-tier 27](#_Toc35536543)

[4. Multi-tier 28](#_Toc35536544)

[5. Avantages 29](#_Toc35536545)

[6. Inconvénients 29](#_Toc35536546)

[IV. Architecture REST (Respresentational State Transfert) 29](#_Toc35536547)

[1. Avantages 29](#_Toc35536548)

[2. Inconvénients 30](#_Toc35536549)

[V. Choix et justification de l’Architecture logiciel choisie 30](#_Toc35536550)

[VI. Architecture réseau du futur système 30](#_Toc35536551)

[PARTIE C 31](#_Toc35536552)

[ANALYSE ET CONCEPTION 31](#_Toc35536553)

[CHAPITRE 1 : CHOIX DE LA METHODE D’ANALYSE ET DE CONCEPTION 32](#_Toc35536554)

[I. Généralités sur les méthodes d’analyse et de conception 32](#_Toc35536555)

[1- Méthodes cartésiennes 32](#_Toc35536556)

[2- Les méthodes Systémiques 32](#_Toc35536557)

[3- Les méthodes formelles 32](#_Toc35536558)

[4- Les méthodes orientées objet 32](#_Toc35536559)

[II. Langage UML 33](#_Toc35536560)

[1. Présentation du langage de modélisation 33](#_Toc35536561)

[2. Diagrammes UML 33](#_Toc35536562)

[III. Le Processus 35](#_Toc35536563)

[1. Le processus unifié (UP, Unified Process) 35](#_Toc35536564)

[2. Rational Unified Process (RUP) 35](#_Toc35536565)

[3. 2TUP(Two Track Unified Process) 36](#_Toc35536566)

[IV. La méthodologie agile 36](#_Toc35536567)

[1. eXtrem Programming 37](#_Toc35536568)

[2. Scrum 37](#_Toc35536569)

[V. Choix de la méthodologie de conception 40](#_Toc35536570)

[VI. Justification du choix de la méthodologie de conception 40](#_Toc35536571)

[CHAPITRE 2 : SPRINT 0 42](#_Toc35536572)

[I. Spécifications des besoins 42](#_Toc35536573)

[1. Identification des acteurs 42](#_Toc35536574)

[2. Besoins fonctionnels 42](#_Toc35536575)

[3. Besoins techniques 43](#_Toc35536576)

[II. Management du projet avec Scrum 44](#_Toc35536577)

[1. Equipes et rôles 44](#_Toc35536578)

[2. Product Backlog (carnet du produit) 45](#_Toc35536579)

[3. Structure et découpage 47](#_Toc35536580)

[CHAPITRE 3 : SPRINT 1, GESTION DES UTILISATEURS 48](#_Toc35536581)

[I. User Stories 48](#_Toc35536582)

[II. Classification des cas d’utilisation par acteurs 50](#_Toc35536583)

[III. Diagramme de cas d’utilisation 51](#_Toc35536584)

[IV. Description textuelles des cas d’utilisation 51](#_Toc35536585)

[CHAPITRE 4 : SPRINT 2, GESTION, SUIVI ET FINANCEMENT DES PROJETS 57](#_Toc35536586)

[I. User Stories 57](#_Toc35536587)

[II. Classification des cas d’utilisation par acteurs 59](#_Toc35536588)

[III. Diagramme de cas d’utilisation 60](#_Toc35536589)

[IV. Description textuelles des cas d’utilisation 60](#_Toc35536590)

[CHAPITRE 5 : SPRINT 3, MESSAGERIES ET NOTIFICATIONS 67](#_Toc35536591)

[I. User Stories 67](#_Toc35536592)

[II. Classification des cas d’utilisation par acteurs 68](#_Toc35536593)

[III. Diagramme de cas d’utilisation 68](#_Toc35536594)

[IV. Description textuelles des cas d’utilisation 68](#_Toc35536595)

[PARTIE D 70](#_Toc35536596)

[GESTION DE PROJET 70](#_Toc35536597)

[CHAPITRE 1 : CONDUITE DE PROJET 71](#_Toc35536598)

[I. Intervenant au projet 71](#_Toc35536599)

[II. Planification en tâches 72](#_Toc35536600)

[III. Diagramme de Gantt 74](#_Toc35536601)

[CHAPITRE 2 : ESTIMATION DES CHARGES 75](#_Toc35536602)

[I. Charges matérielles 75](#_Toc35536603)

[II. Charges logicielles 75](#_Toc35536604)

[III. Coûts humains 75](#_Toc35536605)

[IV. Coûts totaux 76](#_Toc35536606)

[CONCLUSION 77](#_Toc35536607)

[LISTE DES ABREVIATIONS 78](#_Toc35536608)

[TABLES DES FIGURES 79](#_Toc35536609)

[LISTE DES TABLEAUX 80](#_Toc35536610)

[DOCUMENTATION 81](#_Toc35536611)

[Bibliographie 81](#_Toc35536612)

[Webographie 81](#_Toc35536613)

# INTRODUCTION

Les Technologies de l'Information et de la Communication (TIC) sont de plus en plus utilisées de façon variable par bon nombre de nos entreprises pour une meilleure gestion des structures publiques ou privées à travers le monde. Les systèmes informatiques sont devenus des outils indispensables au fonctionnement des entreprises. Ils sont aujourd’hui déployés dans tous les secteurs professionnels, à s’avoir : la médecine, les banques, les assurances, les ministères etc.

La rentabilité des projets est devenue, au fil du temps, une des préoccupations majeures des organisations. Le caractère critique du respect des délais d’achèvement d’un projet s'est accrue, on s'est alors rendu compte de l’importance d’un outil d’accroissement de la rentabilité, non seulement pour un projet en cours de réalisation, mais aussi futur. C'est ainsi que dans la majorité des organisations, les chercheurs se sont attelés à identifier les méthodes et outils qui pouvaient permettre le suivi d’un projet afin de contribuer efficacement à l'atteinte de résultats.

Conçu durant la phase de planification, en même temps que la conception du projet, le suivi est utilisé de manière continue durant toute la phase de mise en œuvre du projet. On entend par la planification, le processus permettant de définir les objectifs, d’élaborer les stratégies, de tracer les grandes lignes des dispositions de mise en œuvre et d’attribuer les ressources nécessaires à la réalisation de ces objectifs.

Pour les institutions en général et celles en particulier des pays en voie de développement, le suivi et l’évaluation de projets recèlent un potentiel de développement qui commence à s’esquisser et des opportunités qu’il serait judicieux de saisir.

C’est dans cette optique que **Techno Mega Partners**, une entreprise spécialisée dans le conseil et développement informatique nous a accueilli et nous a confié la tâche consistant à mettre en œuvre une plateforme efficiente facilitant le suivi des financements sur emprunts, contreparties et dons de la Direction Générale de la Dette.

Pour mener à bien la présentation du travail, nous l’avons scindée en trois (4) grandes parties :

* La première partie est consacrée à la présentation de la structure d’accueil et du sujet. Nous y présenterons la structure qui nous a accueillis pour quatre mois de stage, le sujet, ses contours et les concepts liés au domaine d’étude.
* La deuxième partie nous parlerons de la solution choisie
* La troisième partie abordera l’analyse et la conception
* La quatrième partie traitera de la conduite, la planification et l’estimation globale de notre projet

# *PARTICIPANTS AU PROJET*

|  |  |  |
| --- | --- | --- |
| **Noms et Prénoms** | **Fonction** | **Rôles** |
| KOUSSOUBE Souleymane | Enseignant permanent à IAI | Superviseur |
| NOUSSI Roger | Enseignant permanent à IAI | 2ème correcteur |
| Rodrigue TCHOKOTHE | Ingénieur informaticien | Maitre de stage |
| DOMINIQUE MAOUDONGONE Damba | Etudiant en 3ème année Analyste-Programmeur | Réalisateur |
| VIDILA BAKABADIO Saint Jelys | Etudiant en 3ème année Analyste-Programmeur | Réalisateur |

# PARTIE A

# PRESENTATION DE LA STRUCTURE D’ACCUEIL ET DU SUJET

# CHAPITRE 1 : PRESENTATION DE LA STRUCTURE D’ACCUEIL

## Présentation

TECHNO MEGA PARTNERS (TMP) est un cabinet de prestation de services informatiques, étude, conseil et infogérance. Le siège social est à Libreville (Gabon), sis au quartier SOGATOL, derrière le lycée COLBERT et tout près de l’Académie Privée FOTSOMOBILE.

Officiellement créé en 2019, TMP est un cabinet d’innovation multi technologique, qui allie créativité et savoir-faire pour accompagner les entreprises dans leur transformation numérique.

Dans le cadre de notre stage ; nous avons été placé sous la direction de **M. Rodrigue TCHOKOTHE**, responsable du Pôle recherche et développement.

Le cabinet TMP offre des services dans divers domaines informatiques, notamment

* L’informatique décisionnelle
* La gestion électronique des documents et l’archivage
* L’infogérance
* La maintenance informatique
* Le développement web et mobile
* Le web design.

S’inscrivant dans l’optique de son slogan « *Nous améliorons votre quotidien* », **TMP** offre à ses clients des solutions génériques ou sur mesure destinées à résoudre des problèmes cruciaux dans leurs activités ou de faciliter des procédures de traitements existants.

## Organigramme

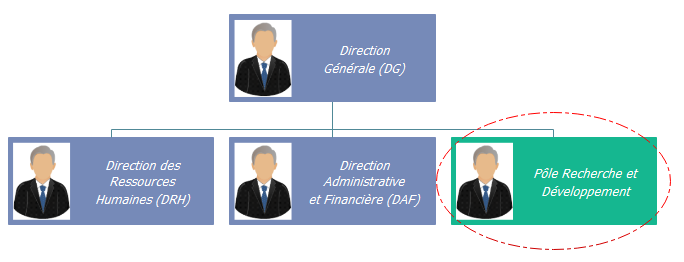
L’Organigramme ci-après, basée sur une gouvernance tournée vers la performance donne une idée de l’organisation de TMP.

Figure : Organigramme de TMP

## Localisation

Le siège social de TMP est à Libreville (Gabon), sis au quartier SOGATOL, derrière le lycée COLBERT et tout près de l’Académie privée FOTSOMOBILE.

### Situation géographique

Figure :Situation géographique de TMP



**TMP**

### Contacts

Tableau :Contacts TMP

|  |  |
| --- | --- |
| BP | 15004 Libreville |
| **Téléphone** | (+241) 66 50 71 65 |
| **E-mail** | [tmpparteners@gmail.com](mailto:tmpparteners@gmail.com), contact@technomegapartners.com |
| **Site web** | [www.technomegapartners.com](http://www.technomegapartners.com/) |

# CHAPITRE 2 : PRESENTATION DU SUJET

Dans ce chapitre, nous essayerons de faire un tour sur le sujet, la problématique de ce thème, l’objectif visé et l’apport de la solution apporté.

## Le sujet et son contexte

La direction générale de la dette (DGD) a été créé en 2011, c’est le département du gouvernement gabonais qui s’occupe de la dette.

Cette institution vient remplacer la Banque Gabonaise de développement et la Caisse Nationale autonome qui ont gérer la dette publique de 1960 à 2011. Pour mener à bien sa mission d’administration, d’analyse et d’élaboration des prévisions budgétaires de la dette publique, la DGD ambitionne de se doter d’un système d’information automatisé qui lui permettra d’assurer la traçabilité d’emprunt, des financements sur contrepartie et des dons, leur mise en place jusqu’à leur exécution; c’est-à-dire permettre de suivre en temps réel, la vie de l’exécution d’un projet cofinancé.

La gestion et la suivie d’un projet cofinancé sont des taches très complexe, pouvant faire intervenir plusieurs acteurs.

Cette complexité fait intervenir la notion de groupe de travail (groupware). La notion de workflow qui est un système de gestion de flux.

Dans l’optique de trouver une solution fiable, souple et sécurisée, la DGD a missionné Techno Mega Partners Gabon, qui est une entreprise d’ingénierie logicielle , de fournir une solution qui s’intitule : « ***Implémentation d’un système de Workflow et Groupware : cas d’application de suivi de financements sur emprunts, contreparties et dons*** »

## Problématique

* Les projets en raisons de leurs caractéristiques, leurs unicités
* Le manque d’archives numériques consultable à distance,
* Le manque de coordination causé par un manque de communication entre les différentes parties prenantes
* Le non-respect de la date butoir
* Les conflits internes suite aux pilotage des différents projets
* Les modifications fréquentes des spécificités initiales des projets entraînant la modification des calendriers des coûts et le surpassement du budget initial
* Certaines taches ne sont pas effectuées

## Objectifs visés et résultat attendus

L’objectif principal poursuivit par ce projet est celui de mettre à la disposition de la DGD, un outil logiciel qui lui permettra d’assurer la traçabilité d’un emprunt de sa mise en place jusqu’à son exécution; c’est-à-dire permettre de suivre en temps réel, la vie de l’exécution d’un projet cofinancé. Cet outil logiciel se veut être une solution d'automatisation sur mesure et adaptée au suivi et au contrôle de l'exécution des marchés publics financés sur emprunts, et dons. Il doit être simple, fiable et sécurisée. Pour ce qui est de son module tableau de bord, il doit être accessible à long terme à partir d’un Smartphone.

Ce logiciel permettra à la Direction Générale de la Dette et ses différents services et entités provinciales associés, de poursuivre certains objectifs bien précis à savoir (entre autres) :

Garantir un suivi transparent du cycle de vie des projets cofinancés. Il permettra d’avoir les informations exactes en temps réel sur un financement global (extérieur et contrepartie). Il s’agira entre autres de:

* Identifier de manière fiable des gestionnaires des différents projets (identité, adresse complète de l’unité de gestion du projet, de son responsable, les acteurs clés et les bénéficiaires ciblés et son comptable);
* Identifier; pour les FINEX, les correspondants budgétaires des différents projets cofinancés (identité, adresse complète), et pour les contreparties, leurs prévisions.
* Apporter des informations sur la manière dont chaque projet prévoit l’exécution de montant emprunté et celui de la contrepartie;
* Informer sur l’état des décaissements ou des ordonnancements (une distinction devant être faite entre les états de sortie concernant les décaissements et ceux concernant la contrepartie car les informations sont obtenus par des administrations différentes) ;
* Apporter des informations sur la manière dont chaque emprunt est prévu pour être exécuté ;
* Voir si un tirage a fait l’objet d’un décaissement
* Voir si la demande de décaissement a déjà été exécutée et à quelle hauteur et à quelle date;
* Suivre l’état d’avancement des projets à un instant donné
* Aider le décideur à prendre d’une façon éclairée et rapide une décision à travers certains indicateurs tels que : (% de décaissement, % de contrôle, d’exécution des travaux, etc.)
* Au besoin, tenir compte du tableau d’amortissement de la dette pour lancer des alertes.
* Garantir l’unicité et la fiabilité des informations collectées et traitées.

# CHAPITRE 3 : CONCEPTS LIES AU DOMAINE D’ETUDE

Pour mener à bien notre travail et atteindre les objectifs fixés, il nous faudra bien comprendre les notions liés à notre domaine d’étude. Dans ce chapitre, nous allons présenter quelques concepts relatifs à notre travail.

* **Projet**

Un projet est un ensemble d'activités organisées en phases ou étapes et formant l'unité de gestion permettant la réalisation d'un objectif défini et précis. Projet = opération ponctuelle ayant un début et une fin, nécessitant la mise en œuvre de ressources humaines et matérielles pour sa réalisation.

Dans notre système les porteurs de projet sont les institutions gabonaises telle que les ministères. Le terme technique pour les designer est maître d’ouvrage.

* **Maîtrise d'ouvrage**

La maîtrise d'ouvrage (MOA), aussi dénommée maître d'ouvrage est la personne physique ou morale, pour qui est réalisé le projet. Elle est l'entité porteuse d'un besoin, définissant l'objectif d'un projet, son calendrier et le budget consacré à ce projet. Le résultat attendu du projet est la réalisation d'un produit, appelé ouvrage. Le maître d'ouvrage a l'initiative et le soin de l'opération. Il doit, avant la réalisation, exercer certaines compétences en vue de définir ses besoins, les caractéristiques et le budget de l'opération.

Décideur, il prend en charge les risques relatifs à l'exercice de ses compétences. Compte tenu de la professionnalisation des missions de maîtrise d'ouvrage, nombre de maîtres d'ouvrage choisissent de désigner un mandataire pour les représenter dans l'exercice de ces missions.

* **Maître d’œuvre**

La maîtrise d'œuvre ou maître d'œuvre (souvent abrégée MOE ou MŒ ou Moe ou moe) est la personne physique ou morale choisie par le maître d'ouvrage pour la conduite opérationnelle des travaux en matière de coûts, de délais et de choix techniques, le tout conformément à un contrat et un cahier des charges. Un maître d'œuvre ne peut pas effectuer de travaux.

Pour la partie relative au marché de travaux, la maîtrise d’œuvre désigne une personne physique ou morale qui, pour sa compétence, peut être chargée par le maître de l'ouvrage :

* de l'assister pour la consultation des entreprises et pour la conclusion du ou des marchés avec le ou les entrepreneurs ;
* de diriger l'exécution du ou des marchés de travaux ;
* d'assister le maître de l'ouvrage pour la réception des ouvrages et le règlement des comptes avec les entrepreneurs.

Il est le plus souvent utilisé en conjonction avec le terme maître d'ouvrage qui désigne le propriétaire de l'ouvrage ou commanditaire des travaux (État, collectivités, entreprises publiques, maître d'ouvrage privé) qui exécute la passation des marchés.

Initialement employé dans le secteur économique du bâtiment et des travaux publics, le diptyque maître d'ouvrage/maître d'œuvre est très utilisé dans le domaine de l'informatique, aussi bien lors de la passation de marchés publics que dans le secteur privé.

* **La maîtrise d’ouvrage déléguée**

La maîtrise d’ouvrage déléguée est une mission complète d’assistance à la réalisation d’un projet, elle est à ce titre un mandataire qui exécute « pour le compte » du maître d'ouvrage et s'engage à mettre en œuvre tous les moyens nécessaires pour réaliser l'ouvrage

* **La dette publique**

La dette publique correspond à l’ensemble des engagements financiers pris sous forme d’emprunts par l’État, les collectivités publiques et les organismes qui en dépendent directement

La dette publique évolue constamment au rythme des remboursements d’emprunts effectués par l’État et les administrations publiques et des nouveaux emprunts qu’ils contractent pour financer leurs déficits.

* **Le financement sur contrepartie ou financement participatif**

Le financement participatif avec contribution implique que les contributeurs à un projet s’attendront à recevoir une récompense en échange de leur participation. C’est un des moteurs de collecte de fond, un moyen d’attirer des contributeurs, et un puissant facteur d’engagement pour les visiteurs.

Ce type de financement intervient suite à un appel d’offre lancé par le gouvernement et les participants peuvent être des personnes physiques (particuliers), des entreprises, des banques, etc. C’est un moyen pour le gouvernement d’impliquer plus des personnes concernant un projet.

Dans un cas pratique , il pourrait aussi s’agir d’un fond engagé par le gouvernement gabonais.

* **Les dons**

Les dons représentent ici les ressources ou bien et services cédées par une personne physique ou morale à l’État

* **Notion de workflow**

Le Workflow se traduit par le « flux de travaux ». Il s’agit d’un processus qui permet d’automatiser la circulation des flux d’informations dans une entreprise. Divers acteurs sont touchés par le processus et notamment les salariés mais aussi les personnes en charge de tâches extérieures à l’entreprise mais concernées par elles.

Ainsi, dans le cadre de notre projet l’outil workflow peut être utilisé pour gérer les projets , les demandes de décaissement , le contrôle fait sur chaque projets , l’attribution et la validation des taches, les financements et aussi la gestions des documents lié aux projets et financements

**Le rôle de l’outil Workflow**

L’outil Workflow permet d’améliorer le circuit de validation. Il partage à chaque intervenant l’information nécessaire à l’exécution de ses tâches. Il signale les délais de livraison ou les délais d’accomplissement de la tâche et rappelle les modes de validation. Ses différentes fonctions lui permettent d’organiser les différentes tâches que les salariés doivent accomplir et de s’assurer que chaque tâche est bien réalisée. Grâce à lui, le manager peut contrôler la bonne marche, la performance et déterminer les points de blocage.

**Type de workflow**

Une première manière de regrouper les workflows est la suivante :

**Le workflow documentaire** : Il s’agit dans ce cas d’un circuit de routage des documents permettant à ces derniers de suivre un trajet prédéfini, pouvant être modifié par un des participants selon les cas.

**Le workflow métier** : Il représente un processus transverse à l'entreprise (concerne plusieurs entités organisationnelles). Il s'agit le plus souvent de procédures rattachées à une division opérationnelle de l'entreprise. Cela désigne un ensemble d'activités qui s'enchaînent de manière chronologique pour atteindre un objectif, généralement délivrer un produit ou un service, dans le contexte d'une organisation de travail (ex : une entreprise, administration, etc.).

On peut aussi regrouper les workflows selon les catégories suivantes :

**Le Workflow de production**, qui correspond à la gestion des processus de base de l'entreprise. Les procédures supportent peu de changements dans le temps, et les transactions sont répétitives (ex : contrats d'assurance, gestion de litiges, de réclamations clients, etc.) ;

**Le Workflow Administratif**, qui correspond à tout ce qui est routage de formulaires ;

**Le Workflow Ad-Hoc** pour la gestion des procédures non déterminées, ou mouvantes

**Le Workflow Coopératif**, gérant des procédures évoluant assez fréquemment, et liées à un groupe de travail restreint dans l'entreprise.

* **Notion de groupware**

Le groupware est l’ensemble des technologies et des méthodes de travail associées qui, par l’intermédiaire de la communication électronique, permettent le partage de l’information sur un support numérique à un groupe engagé dans un travail collaboratif et/ou coopératif.

Pour notre cas d’étude ,Il est indispensable de disposer d'un mécanisme de groupware permettant de partager un calendrier, des listes de tâches à accomplir(avec possibilité de les déléguer) assorties de dates butoirs, et d'organiser de façon automatique des réunions par consultation automatique des plages libres de chacun (avec réservation automatique de la salle de réunion ou des salles de téléconférence également).

**La réplication**

Un aspect essentiel du groupware est la question de la réplication: un membre du groupe peut être amené à voyager avec son ordinateur portable. Si pendant son voyage il est isolé du groupe, l'ensemble du travail du groupe au moment de son départ doit être à sa disposition sur son disque dur, il doit avoir la possibilité d'apporter des contributions au travail existant, qui sera transmis en différé dès son retour, dans le même temps que son travail additionnel est envoyé au groupe, l'ensemble de ses données doit être remis au niveau du groupe, qui a continué à travailler en son absence sur le réseau.

**Les agendas partagés**

Ces logiciels permettent de trouver un créneau horaire pour une réunion - physique ou virtuelle en quelques clics, et sans déranger les personnes concernées. Chaque employé entre en effet son agenda dans une grande base partagée : les informations sont accessibles à l'ensemble du groupe de travail.

Pour bien fonctionner, un tel outil requiert la mise à jour détaillée de tous les agendas, mais cet objectif est rarement atteint en entreprise. En effet, les utilisateurs ne peuvent noter sur cet agenda que leurs rendez-vous officiels.

**Les types de groupwares**

Il faut distinguer deux familles de groupwares : les groupwares de messagerie collaborative, des groupwares orientés projet (projectware)

Dans ce chapitre on a eu à présenter les divers concepts liés au projet qui nous a été confié. Comprendre ces notions, est important car la réussite du projet en dépend.

Nous pouvons dès à présent appréhender et analyser le problème qui nous a été posé, afin de proposer une solution qui puisse répondre parfaitement aux besoins des différents institutions du Gabon porteur de projet , financé ou cofinancé par l’État gabonais

# PARTIE B

# SOLUTION

# CHAPITRE 1 : ETUDE DE L’EXISTANT

Dans le processus de création d’un logiciel , l’étude éventuelle des solutions existantes est fondamentale car cette phase nous permettra d’identifier les manques afin de fournir une solution qui réponde réellement aux besoins exprimés.

Dans cette partie nous allons nous pencher sur les solutions utilisé au seins de la Direction Générale de la Dette en matière de Suivie des projets, financements, dettes ,contreparties et dons.

## Présentation de l’existant

**Le système Sigal (Société d'Informatique de Gestion d'Assistance et de Logiciel)**

A ce jours il existe une seule solution utilisé au sein de la DGD en matière de gestion et suivi des projets, cette solution est un logiciel propriétaire appelé Sigal. Cette solution est proposé par une entreprise française du même nom, le progiciel Sigal permet de gérer la relation clientèle, mais aussi un progiciel de gestion intégrée, le système Sigal est payant et est déployé sur demande.

## Critique de l’existant

Malgré sa robustesse, cette solution n’est pas tout à fait adapté à la réalité du Gabon et ne cadre pas avec le contexte actuelle de gestion des projets, des suivies des financements, voici en quelque points , les critiques du système Sigal:

* Le système ne permet pas d’effectuer un travail à distance,
* Chaque modification éventuelle sur un projet ou un financement doit être fait sur un post sur lequel Sigal est installé
* Impossible de suivre en temps réelle l’évolution d’un projet,
* Ne permet pas le travail en groupe ,
* Ne permet d’organiser des taches qui suivent l’évolution d’un flux de travail,
* Les contextes de gestion de projet sous Sigal ne concordent pas forcement avec les normes du journal officiel de la République Gabonaise ,

Nous avons, dans ce chapitre, présenté la solutions utilisé au sein de la Direction Générale de la Dette en matière de gestion des projets, suivi des financements, dettes, contreparties et dons , tout en énumérant les fonctionnalités et un certain nombre de difficultés de ce système d’information. Il a été vite constaté que cette solution ne s’adapte pas parfaitement aux besoins de cette institution. Ce qui nous amène à opter pour une conception et la réalisation d’un système pour répondre aux besoins exprimés.

# CHAPITRE 2 : PROPOSITION DES SOLUTIONS

## Proposition des solutions

Dans ce chapitre, nous ferons les différentes propositions des solutions du futur système.

### Première solution

Il s’agit de la réalisation d’une application web basée sur la technologie PHP et ayant le HTML5 comme langage de présentation des données. Le système de gestion de bases de données sera MySQL.

Les applications PHP couplé au système de gestion de bases de données MySQL sont très facile à implémenter et surtout facile à maintenir à cause de la communauté au tour de ces technologies mais le fait que le langage de programmation PHP ne gère pas d’une manière native les web sockets nous limite quand on veut avoir une application interactive, d’autre part la sécurité sous PHP est un vrai défi et pour des entités qui ne disposent pas assez de ressources humaines et matériels à allouer à la sécurité, PHP semble être un mauvais choix.

### Deuxième solution

La seconde solution est une application web basée sur la technologie Java avec l’utilisation du Framework Spring MVC pour le backend et Angular pour le frontend, il s’agit ici d’une architecture REST(REpresentational State Transfer) en backend, et l’utilisation du paradigme de programmation MVC coté client avec Angular et comme base de données nous allons opter pour PostgreSQL

Representational State Transfer (REST) évoque l'image du fonctionnement d'une application Web bien construite : un réseau de pages Web (une machine à états finis virtuelle) où l'utilisateur progresse dans l'application en cliquant sur des liens (transition entre états) ce qui provoque l'affichage de la page suivante (représentant le nouvel état de l'application) à l'utilisateur qui peut alors l'exploiter.

Côté positif, parmi les contraintes d'interfaçage uniforme de l'architecture REST, toutes les ressources doivent disposer d'un URI (Uniform Resource Identifier) et répondre aux opérations HTTP (HyperText Transfert Protocol) GET, PUT, POST (Power-On Self-Test; autotest à la mise sous tension) et DELETE. Ces opérations correspondent globalement à la doctrine de manipulation des données CRUD (Create, Read, Update, Delete : créer, lire, mettre à jour, supprimer).

### Choix et justification de la solution

Vue les spécifications de l’application à concevoir et surtout dans le but de respecter les orientations de l’utilisateur, notre choix s’est porté sur la deuxième solution.

Cette solution est sûre et offre beaucoup de perceptive coté robustesse car Java offre une possibilité de faire du web socket d’une manière native et Angular permet de créer des applications web interactive, c’est d’ailleurs l’architecture utilisé dans la plupart des entreprises web qui ont le plus de succès

Les API REST imitent la façon dont le web lui-même marche dans les échanges entre un client et un serveur. Une API REST est :

* Sans état
* Cacheable (avec cache = mémoire)
* Orienté client-serveur
* Avec une interface uniforme
* Avec un système de couche
* Un code à la demande (optionnel)

Le principe du client-serveur définit les deux entités qui interagissent dans une API REST : un client et un serveur, les mêmes entités qui communiquent sur le web. Un client envoie une requête, et le serveur renvoie une réponse. Ce dernier doit avoir le plus d’informations possible sur le client, car il est important qu’ils soient capables de travailler indépendamment l’un de l’autre.

Bien que cette solution soit coûteux en temps et en ressources, coté sécurité on a une garantie, car le Framework Spring MVC est réputé pour la sécurité car il possède tout un module dédié à ce thématique intitulé **Spring Security**

En c’est qui concerne la base des données PostgreSQL, on peut citer ces quelques avantages :

* PostgreSQL est plus fiable et l’intégrité des données y est plus performante;
* Tout ce qui est lié aux requêtes est meilleur avec PostgreSQL.,
* PostgreSQL dispose d’un planificateur de requêtes sophistiqué et d’un optimiseur de requêtes ;
* Dans un contexte de datamining/data Warehouse, PostgreSQL est très performant par rapport à MySQL ;
* La documentation de PostgreSQL est plus dense et complète et de façon général le support est meilleur ;
* PostgreSQL traîne moins de casseroles (astuces, limitations…) que MySQL ;
* PostgreSQL est meilleur pour les requêtes avec sous-requêtes ;
* Argument non technique : PostgreSQL ne «dépend» pas d’un contributeur principal privé comme c’est le cas pour MySQL avec Oracle ;

# CHAPITRE 3 : CHOIX DU CONCEPT ET DE L’ARCHITECTURE DE LA SOLUTION RETENUE

## Client lourd/client léger

Le terme « client lourd » désigne une application cliente graphique exécutée sur le système d'exploitation de l'utilisateur. Un client lourd possède généralement des capacités de traitement évoluées et peut posséder une interface graphique sophistiquée. Cela permet aux systèmes des utilisateurs finaux d'être gérées de manière centralisée et avec des logiciels déployés sur un seul serveur central, le client lourd nécessite l’installation du logiciel sur chaque poste utilisateur.

Les principaux avantages du client lourd sont entre autre :

* **L’ergonomie** : étant exécutée sur le poste client le développeur a accès à l’ensemble des fonctionnalités de l’OS ainsi que toutes les ressources est évènements matériels.
* **La répartition** : en mode client-serveur, le code d’affichage côté IHM et de gestion de l’interaction utilisateur est exécuté au niveau du poste client ce qui offre l’avantage de solliciter les ressources serveurs uniquement pour la restitution des données et/ou traitements centralisés.
* **L’interconnectivité** : ayant accès aux ressources matérielles les solutions de type client lourd permettent le pilotage et/ou la réception d’informations de périphériques matériels connectés au poste client.

Les principaux inconvénients des solutions client lourd sont :

* **Le déploiement et la montée de version** : étant installée sur chacun des postes clients, le déploiement d’une solution de type client lourd devient un autre projet. Par la suite, à chaque montée de version il est nécessaire de redéployer la nouvelle version sur chacun des postes.
* **La compatibilité** : étant compilé pour une architecture processeur et faisant appel à des fonctionnalités au niveau OS, une solution de type client lourd offre une compatibilité restreinte à une configuration matérielle et un système d’exploitation donné.

Les clients légers sont vraiment mieux adaptés aux environnements dans lesquels les mêmes informations vont être accessibles par les clients, ce qui présente en fait une meilleure solution pour les environnements publics.

Ce mode de fonctionnement permet de résoudre les inconvénients vus précédemment sur les clients lourds. En effet, le passage par un navigateur Web évite la phase de déploiement de l’application sur l’ensemble des postes clients. Ainsi lors des montées de version seul l’applicatif côté serveur a besoin d’être mise à jour. Quant à la compatibilité, une application en client léger est compatible avec tous les OS et processeurs permettant l’exécution d’un navigateur Web supportant l’application. Cela facilite le déploiement et la maintenance.

En revanche, un client lourd fournit aux utilisateurs finaux plus de fonctionnalités, des graphiques et des choix qui rendent les applications plus personnalisables. Contrairement à des clients légers, les clients lourds ne reposent pas sur un serveur central de traitement parce que le traitement est effectué localement sur le système de l'utilisateur et le serveur est accessible principalement à des fins de stockage. Pour cette raison, les clients lourds ne sont souvent pas bien adaptés pour les environnements publics.

## Concept client/serveur

De nombreuses applications fonctionnent selon un environnement client/serveur, cela signifie que des machines clientes (des machines faisant partie du réseau) contactent un serveur, une machine généralement très puissante en termes de capacités d’entrée-sortie, qui leur fournit des services.

Les services sont exploités par des programmes, appelés programmes clients, s’exécutant sur les machines clientes, on parle ainsi de client FTP, client de messagerie, etc. Lorsque l’on désigne un programme, tournant sur une machine cliente, capable de traiter des informations qu’il récupère auprès du serveur (dans le cas du client FTP il s’agit de fichiers, tandis que pour le client messagerie il s’agit de courrier électronique).

Dans un environnement purement client/serveur, les ordinateurs du réseau (les clients) ne peuvent voir que le serveur, c’est un des principaux atouts de ce modèle.

## Architecture client/serveur

Une architecture client/serveur fonctionne selon le schéma suivant :

* Le client émet une requête vers le serveur grâce à son adresse et le port qui désigne un service particulier du serveur
* Le serveur reçoit la demande et répond à l’aide de l’adresse de la machine client et son port

Plusieurs architectures client/serveur peuvent être rencontrées. Nous commencerons par décrire l’architecture d’égal à égal, qui n’est pas à proprement parler de type client/serveur.

### 1-tier

L'architecture à **un niveau est une architecture sur** lequel le client, le serveur et la base de données résident tous sur le même ordinateur. Chaque fois que vous installez une base de données sur votre système et y accédez pour vous entraîner aux requêtes SQL, il s'agit d'une architecture à 1 niveau. Mais une telle architecture est rarement utilisée en production.

### 2-tier

L’architecture à deux niveaux, aussi appelée *architecture 2-tier*, caractérise les systèmes clients/serveurs dans lesquels le client demande une ressource et le serveur la lui fournit directement. Cela signifie que le serveur ne fait pas appel à une autre application afin de fournir le service.

Le client échange directement avec le serveur sans intermédiaire. C’est ce

que l’on retrouve le plus souvent dans les environnements de moins de cinquante utilisateurs. Elle peut être représentée par le schéma suivant :

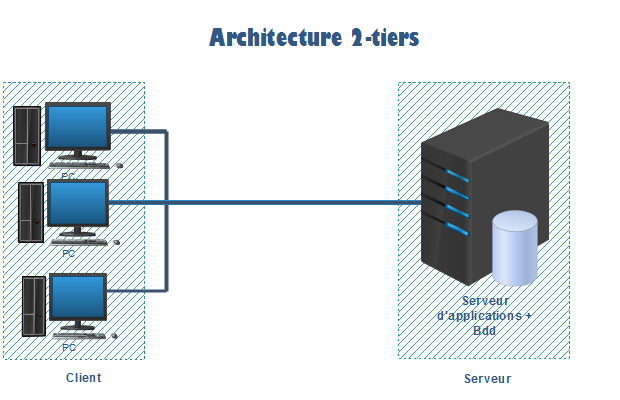
L’architecture à deux niveaux est donc une architecture client/serveur dans laquelle le serveur est polyvalent, c’est-à-dire qu’il est capable de fournir directement l’ensemble des ressources demandées par le client.

Figure :Architecture 2-tier

### 3-tier

Dans l’architecture à 3 niveaux (appelées *architecture 3-tier*), il existe un niveau intermédiaire, c’est-à-dire que l’on a généralement une architecture partagée entre :

* Le client : le demandeur de ressources,
* Le serveur d’application (appelé aussi middleware) : le serveur chargé de fournir la ressource mais faisant appel à un autre serveur,
* Le serveur secondaire (généralement un serveur de base de données), fournissant un service au premier serveur.

Dans l’architecture à trois niveaux par contre, les applications au niveau serveur sont délocalisées, c’est-à-dire que chaque serveur est spécialisé dans une tâche (serveur web ou serveur de base de données par exemple). Ainsi, l’architecture à trois niveaux permet :

* Une plus grande flexibilité/souplesse
* Une plus grande sécurité (la sécurité peut être définie par exemple pour chaque service)
* De meilleures performances (les tâches sont partagées)

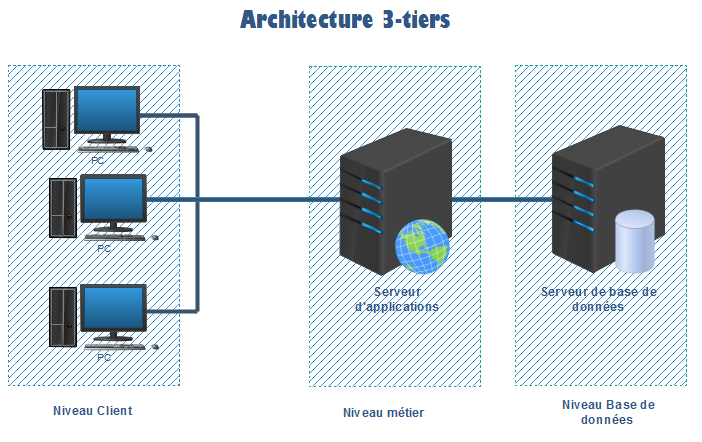


Figure :Architecture 3-tier

### Multi-tier

L’architecture multi tiers constitue une évolution de l’architecture trois tiers. Elle se retrouve le plus souvent dans le contexte de l’Internet et des applications web. L’éventuelle source de données est aussi bien accessible à partir de l’extérieur que de l’intérieur de l’entreprise. La figure ci-dessous illustre cette situation :

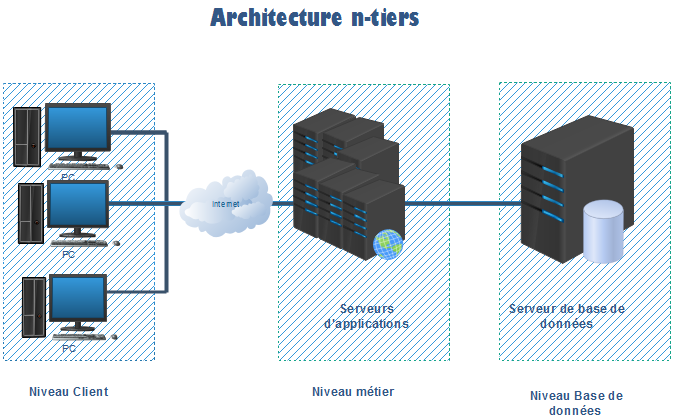


Figure : Architecture multi-tier

### Avantages

Le modèle client/serveur est particulièrement recommandé pour des réseaux nécessitant un grand niveau de fiabilité, ses principaux atouts sont :

* Des ressources centralisées : étant donné que le serveur est au centre du réseau, il peut gérer des ressources communes à tous les utilisateurs, comme par exemple une base de données centralisée, afin d'éviter les problèmes de redondance et de contradiction ;
* Une meilleure sécurité car le nombre de points d'entrée permettant l'accès aux données est moins important ;
* Une administration au niveau serveur : les clients ayant peu d'importance dans ce modèle, ils ont moins besoin d'être administrés ;
* Un réseau évolutif : grâce à cette architecture il est possible de supprimer ou de rajouter des clients sans perturber le fonctionnement du réseau et sans modifications majeures.

### Inconvénients

L'architecture client/serveur a tout de même quelques lacunes parmi lesquelles :

* Un coût élevé dû à la technicité du serveur ;
* Un maillon faible : le serveur est le seul maillon faible du réseau client/serveur, étant donné que toute les autres machines connectées au réseau dépendent du serveur.

## Architecture REST (Respresentational State Transfert)

**REST** est un moyen d'accéder aux documents et aux ressources distants selon une architecture logicielle simple, représentée par une API. On dit "RESTful" les sites utilisant ce modèle de communication.

**REST** (***representational state transfer***) est un style d'architecture définissant un ensemble de contraintes à utiliser pour créer des services web. Les services web conformes au style d'architecture REST, aussi appelés services web ***RESTful***, établissent une interopérabilité entre les ordinateurs sur [Internet](https://fr.wikipedia.org/wiki/Internet). Les services web REST permettent aux systèmes effectuant des requêtes de manipuler des ressources web via leurs représentations textuelles à travers un ensemble d'opérations uniformes et prédéfinies [sans état](https://fr.wikipedia.org/wiki/Protocole_sans_%C3%A9tat).

### Avantages

* Plus facile à mettre en œuvre que les alternatives classiques (on ne parle pas de WebSockets).
* Il suffit d'un navigateur pour accéder à un service.
* Mise en cache des ressources, donc accélération des opérations.
* Moins de consommation mémoire.
* Possibilité de répartir les requêtes sur plusieurs serveurs. Cela grâce à l'absence d'états.
* L'utilisation de formats standards comme HTML ou XML assure la compatibilité dans le temps.
* On peut échanger des requêtes entre diverses applications ou média car elle sont représentées par des URI.

### Inconvénients

* Les données nécessaires à l'utilisation du service web doivent être conservées localement.

## Choix et justification de l’Architecture logiciel choisie

Pour la conception de notre système, nous utiliserons l’architecture client/Serveur de type **REST** qui est un style d’architecture permettant de construire des applications (Web, Intranet, Web Service).

## Architecture réseau du futur système

La figure ci-dessous montre les différents niveaux de services de l’architecture utilisée.

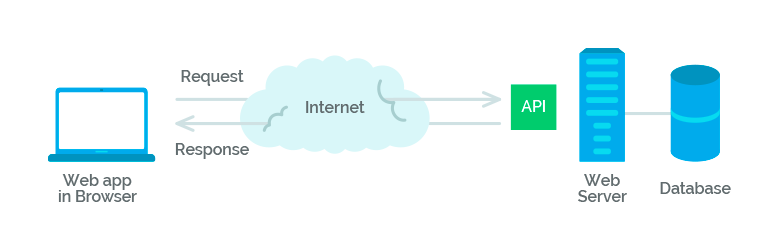


Figure : Architecture de la solution choisie

# PARTIE C

# ANALYSE ET CONCEPTION

# CHAPITRE 1 : CHOIX DE LA METHODE D’ANALYSE ET DE CONCEPTION

## Généralités sur les méthodes d’analyse et de conception

En informatique, on définit une méthode d’analyse et de conception comme étant un procédé qui, à partir des besoins exprimés permet de concevoir un système (informatique) adéquat. Il existe deux grandes approches informatiques, les plus utilisées sur lesquelles reposent plusieurs méthodes : l’approche systémique ou fonctionnelle et l’approche objet.

## Méthodes cartésiennes

Les méthodes cartésiennes qui sont les toutes premières, créées suite aux problèmes de développement du logiciel appelé crise du logiciel. Ces méthodes préconisent qu’un problème soit décomposé en sous problèmes en vue d’en maîtriser la complexité.

## Les méthodes Systémiques

Les méthodes systémiques dont l’un des objectifs est de corriger les failles des méthodes cartésiennes en incorporant les données. Elles définissent différents niveaux de préoccupation ou d'abstraction et proposent de nombreux modèles complémentaires. Les méthodes systémiques sont souvent spécialisées pour la conception d'un certain type de systèmes tels que les systèmes de conception de bases de données relationnelles (MERISE, AXIAL …). Toutefois, ces méthodes juxtaposent les données et les traitements sans véritablement les intégrer.

## Les méthodes formelles

Les méthodes formelles, quant à elles, sont des techniques permettant de raisonner rigoureusement, à l'aide de logique mathématique, sur des programmes informatiques ou du matériel électronique, afin de démontrer leur validité par rapport à une certaine spécification. Elles sont basées sur les sémantiques des programmes, c'est-à-dire sur des descriptions mathématiques formelles du sens d'un programme donné par son code source. Comme exemple nous avons les méthodes formelles RAISE ( Rigorous Approach for Industrial Software Engineering)

## Les méthodes orientées objet

Ce sont des méthodes consistant à créer une représentation informatique des éléments du monde réel auxquels on s'intéresse, sans se préoccuper de l'implémentation, ce qui signifie indépendamment d'un langage de programmation. Il s'agit donc de déterminer les objets présents et d'isoler leurs données et les fonctions qui les utilisent.

Exemples : BOOCH, OMT, OSE.

## Langage UML

UML, c’est l’acronyme anglais pour « Unified Modeling Language ». On le traduit par « Langage de modélisation unifié ». La notation UML est un langage visuel constitué d’un ensemble de schémas, appelés des diagrammes, qui donnent chacun une vision différente du projet à traiter. UML nous fournit donc des diagrammes pour représenter le logiciel à développer : son fonctionnement, sa mise en route, les actions susceptibles d’être effectuées par le logiciel, etc.

UML unifie à la fois les notations et les concepts orientés objet. Il ne s’agit pas d’une simple notation graphique, car les concepts transmis par un diagramme ont une sémantique précise et sont porteurs de sens au même titre que les mots d’un langage.

UML est un langage de modélisation né de la fusion des trois méthodes qui ont le plus influencé la modélisation objet au milieu des années 90 : OMT, Booch et OOSE.

### Présentation du langage de modélisation

C’est le langage graphique le plus utilisé pour la modélisation des différents aspects du Système d’Information. Car, il unifie les notations nécessaires aux différentes activités d’un processus de développement et offre, par ce biais, le moyen d’établir le suivi des décisions prises, depuis la définition des besoins jusqu’au codage. Il comporte ainsi treize types de diagrammes distincts servant à représenter des concepts particuliers du Système d’ Information.

Par contre, dans le contexte de l’ingénierie système, son pouvoir d’expression est très limité. En effet, certains concepts spécifiques à ce domaine ne peuvent être spécifiés simplement avec UML. Par exemple, le fait qu’il peut exister des variantes entraînant un fonctionnement particulier du système ne peut être modélisé avec UML. De plus, le lien fort entre le logiciel et le matériel ne trouve pas en UML de moyen satisfaisant d’expression.

### Diagrammes UML

UML 2 intègre 13 diagrammes permettant de décrire et étudier le système selon différents points de vue.

Ainsi, UML fait la distinction entre la vue statique et la vue dynamique du système à concevoir.

#### Diagrammes statiques

UML comporte sept (07) diagrammes pour décrire la structure statique du

logiciel. Ainsi nous avons :

* Le diagramme des cas d’utilisation : Il décrit les utilisateurs du système en termes de rôle et des fonctionnalités qui leur sont nécessaires.
* Le diagramme de classe : Il décrit les objets du système en termes de classe et les relations qui existent entre ces dernières.
* Le diagramme des packages : Il permet d’Organiser les éléments de modélisation en groupes avec pour objectif de rendre les diagrammes plus simples et plus faciles à comprendre. Il permet de visualiser la hiérarchie des modules d’un projet.
* Le diagramme d’objets : Il sert à illustrer les structures de classes compliquées en montrant des exemples d’instances. Il permet de vérifier l’adéquation d’un diagramme de classe dans diverses situations.
* Le diagramme de structure composite : Il permet d’explorer les instances des classes collaborant à travers des liens de communications, la description d’un objet complexe en phase d’exécution.
* Le diagramme des composants : Il permet de décrire l’architecture d’exploitation constituée entre autres par les librairies dynamiques, les instances de bases de données, les applications.
* Le diagramme de déploiement : Il décrit la structure du réseau informatique qui prend en charge le système logiciel.

#### Diagrammes dynamiques

UML comporte six (06) diagrammes dynamiques :

* Le diagramme d’états : Il représente le cycle de vie commun aux objets d’une même classe.
* Le diagramme d’activité : Il représente les règles d’enchaînement des activités et actions du système.
* Le diagramme de séquence : Ce diagramme montre des exemples d’historique de communication entre les objets ou les utilisateurs. Il décrit les interactions entre objets d’un point de vue temporel.
* Le diagramme de collaboration : Il présente un ensemble des rôles joués par des objets dans un contexte particulier ainsi que les liens entre les objets décrits au travers d’envoi de message en se focalisant sur un point de vue spatial.
* Le diagramme global d’interaction : Il permet notamment de décrire une méthode complexe en associant les notations du diagramme de séquence et du diagramme d’activité.
* Le diagramme de temps : Il explore le comportement d’un ou plusieurs objets pendant une période de temps donnée. Il est très utilisé dans les systèmes temps réelle.

## Le Processus

UML n’est qu’un langage, un ensemble d’outils permettant de décrire un modèle ; afin d’avoir une méthode qui nous permettra de passer des besoins au code, il nous faut un processus de développement.

Le processus de développement décrit une approche du développement logiciel. Il définit une séquence d'étapes, en partie ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant. Deux points de vue sont considérés dans un processus : la production de logiciels de qualité qui répondent aux besoins des utilisateurs (c'est le point de vue technique. Le point de vue de la planification et de l'estimation de coûts consiste à imposer des délais et des coûts raisonnables.

### Le processus unifié (UP, Unified Process)

Le processus unifié (PU), ou « unified process (UP) » en anglais, est une famille de méthodes de développement de logiciels orientés objets. Elle se caractérise par une démarche itérative et incrémentale, pilotée par les cas d'utilisation, et centrée sur l'architecture et les modèles UML. Elle définit un processus intégrant toutes les activités de conception et de réalisation au sein de cycles de développement composés d'une phases de création, d'une phase d'élaboration, d'une phase de construction et d'une phase de transition, comprenant chacune plusieurs itérations.

### Rational Unified Process (RUP)

Contrairement au processus en cascade (souvent appelé cycle en V, en France), le Processus Unifié ne considère pas que les disciplines sont purement séquentielles. En fait, une itération comporte une certaine quantité de travail dans la plupart des disciplines. Cependant, la répartition de l’effort relatif entre celles-ci change avec le temps. Les premières itérations ont tendance à mettre plus l’accent sur les exigences et la conception, les autres moins, à mesure que les besoins et l’architecture se stabilisent grâce au processus de feedback et d’adaptation.

RUP, quant à lui, est caractérisé par un découpage temporel en phases et itérations, l’utilisation d’un support entièrement informatique et un aspect générique et paramétrable. Il repose sur six principes :

* développement itératif ;
* traitement et formalisation des exigences ;
* architecture à base de composants ;
* modélisation visuelle ;
* vérification de la qualité du logiciel ;
* gestion du changement.

### 2TUP(Two Track Unified Process)

Le processus 2TUP (Two Track Unified Process) est un processus unifié. Il gère la complexité technologique en donnant part à la technologie dans son processus de développement. Il est construit sur UML et a aussi les mêmes caractéristiques que UP c’est-à-dire itératif, centré sur l’architecture et conduit par les cas d’utilisation. Il apporte une réponse aux contraintes de changement continuel imposées aux systèmes d’informations. 2TUP peut se décomposer et se traiter parallèlement, suivant un axe fonctionnel et un axe technique. À l’issue des évolutions du modèle fonctionnel et de l’architecture technique, la réalisation du système consiste à fusionner les résultats des deux branches.

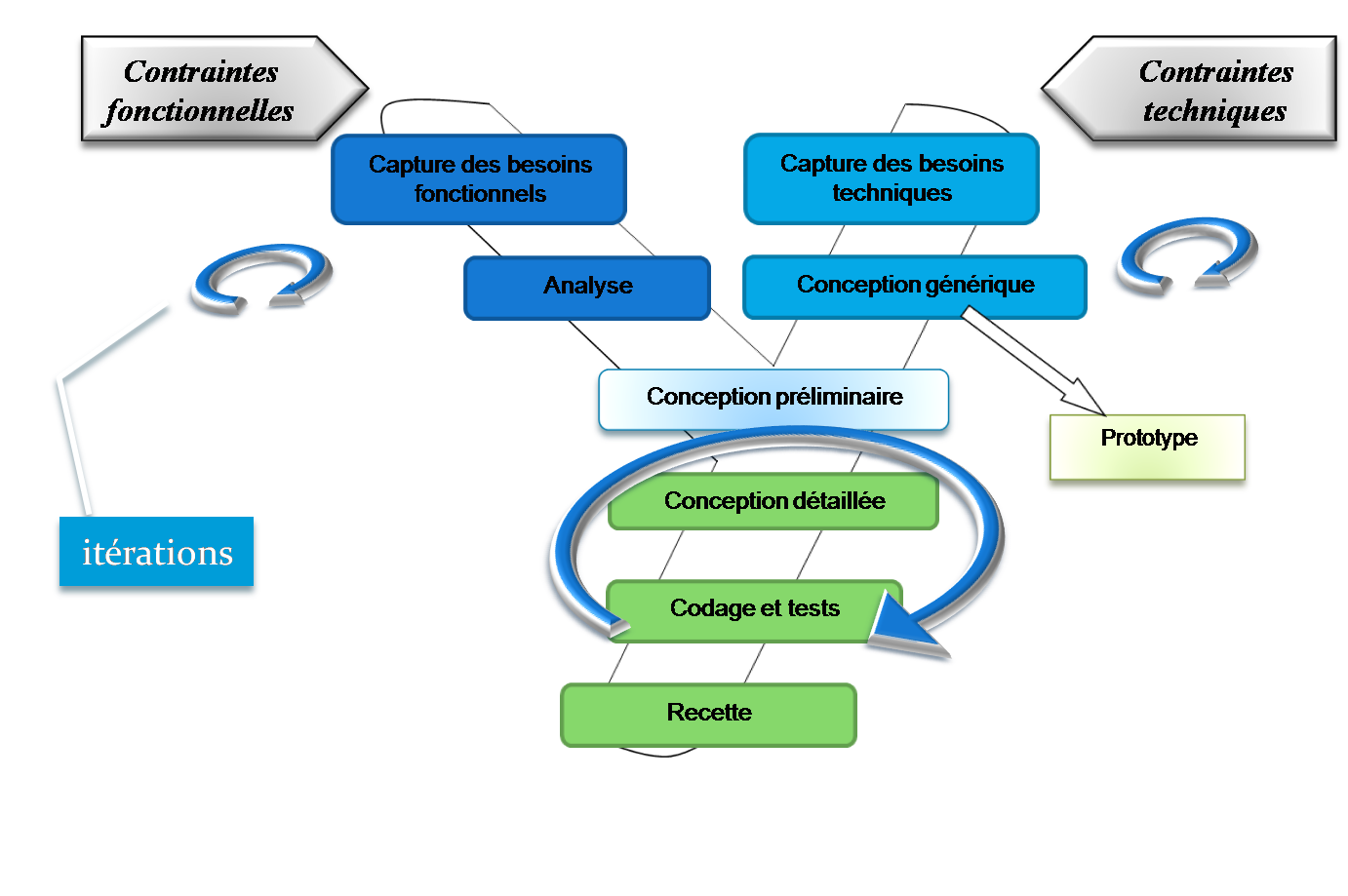


Figure : Processus de développement en Y

## La méthodologie agile

Les méthodes agiles caractérisent un mode de gestion des projets informatiques privilégiant le dialogue entre toutes les parties prenantes, clients, utilisateurs, développeurs et autres professionnels du projet, la souplesse en cours de réalisation, la capacité à modifier les plans et la rapidité de livraison. Il s'agit de rompre avec les pratiques plus traditionnelles bien trop rigides et trop exigeantes en matière de spécifications (contractuelles). Pour cela il est important d'accorder la priorité au relationnel et à la communication étendue sur les processus de développement. Les méthodes Agiles les plus populaires en usage aujourd’hui sont :

* l’eXtrême Programming (XP),
* Scrum,
* Feature Driven Development (FDD),
* Lean Software Development,
* Crystal
* Dynamic Systems Development Method (DSDM)

### eXtrem Programming

eXtrem 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 devait intervenir sur un logiciel de paie écrit en langage Smalltalk 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’attend le client et les réalisations est garantie. Les fonctionnalités sont livrées régulièrement, afin d’être testées et validée au travers de prototypes opérationnels.

L’eXtreme Programming préconise également le travail en binôme des développeurs, facilitant ainsi la production d’un code simple, facilement lisible et maintenable.

### Scrum

Scrum est un processus agile issu des travaux de deux des signataires du Manifeste Agile, **Ken Schwaber** et **Jeff Sutherland**, au début des années 1990. Il a été conçu pour améliorer la productivité en évitant de paralyser les équipes par l’emploi de méthodologies trop lourdes.

Scrum est la méthodologie la plus utilisée parmi les méthodes Agiles existantes. Le terme Scrum (qui signifie mêlée) apparaît pour la première fois en 1986 dans une publication de Hirotaka Takeuchi et Ikujiro Nonaka qui décrit une nouvelle approche plus rapide et flexible pour le développement de nouveaux produits. Ils comparent alors cette nouvelle méthode au rugby à XV, le principe de base étant que l'équipe avance ensemble et soit toujours prête à réorienter le projet au fur-et-à-mesure de sa progression, tel un ballon de rugby qui doit passer de main en main jusqu'à marquer un essai.

Scrum est une méthode agile dédiée à la « gestion de projet ». Cette méthode de gestion, ou plutôt ce Framework de management de projet, a pour objectif d’améliorer la productivité de son équipe.

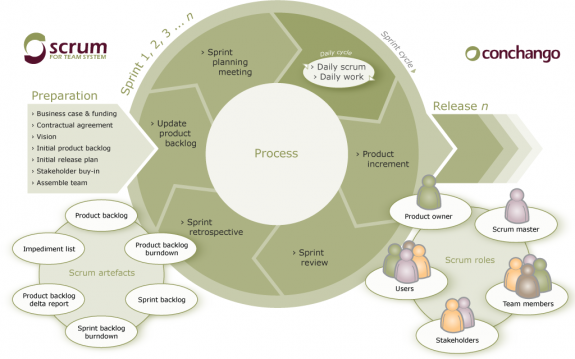


Figure : Différents concepts de Scrum

#### Répartitions des rôles dans Scrum

**Le Scrum Master**

* S’assure que les principes et les valeurs de Scrum sont respectés
* Facilite la communication au sein de l’équipe
* Cherche à améliorer la productivité et le savoir-faire de son équipe

**Le Product Owner**

* Expert métier, définit les spécifications fonctionnelles
* Etablit la priorité des fonctionnalités à développer ou corriger
* Valide les fonctionnalités développées
* Joue le rôle du client

**L’équipe**

* Pas de rôle bien déterminé : architecte, développeur, testeur
* Tous les membres de l’équipe apportent leur savoir-faire pour accomplir les tâches

#### Méthode de gestion de projet Scrum

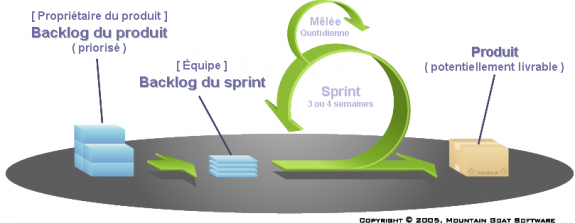


Figure : Cycle d'un sprint

**Les sprints**

Le cycle de vie Scrum est rythmé par des itérations de quelques semaines, les sprints. Le sprint est une période d'un mois au maximum, au bout de laquelle l'équipe délivre un incrément du produit, potentiellement livrable. Une fois la durée choisie, elle reste constante pendant toute la durée du développement. Un nouveau sprint démarre dès la fin du précédent.

Chaque sprint possède un but et on lui associe une liste d'éléments du carnet du produit (fonctionnalités) à réaliser.

**Le product backlog**

Le référentiel des exigences initiales est dressé et hiérarchisé avec le client. Il constitue ce que l’on nomme le product backlog. Il ne doit pas nécessairement contenir toutes les fonctionnalités attendues dès le début du projet, il va évoluer durant le projet en parallèle des besoins du client.

**User Story**

Les fonctionnalités décrites portent le nom de User Stories et sont décrites en employant la terminologie utilisée par le client.

Une User Story ou Story contient généralement les informations suivantes :

* **ID** : un identifiant unique
* **Nom** : un nom court (entre 2 et 10 mots), descriptif de la fonctionnalité attendue par le client (ex. Export / Import Standard Sales Item). Le nom doit être suffisamment clair pour que les membres de l’équipe et le Product Owner comprennent de quelle fonction il s’agit. Le nom ne doit pas introduire d’ambigüités.
* **Importance** : un entier qui fixe la priorité des Stories. La priorité d’une story peut être changée en cours de réalisation du projet.
* **Estimation** : La quantité de travail nécessaire pour développer, tester, et valider cette fonctionnalité. L’unité de mesure peut être un nombre de jours idéaux (jours à 100% dédiés à la fonctionnalité) ou un nombre de points. Les estimations se font en relatif en comparant les estimations des stories terminées avec la story à estimer.

## Choix de la méthodologie de conception

Dans le cadre du développement de notre application, nous allons utiliser une méthode à la fois agile et unifiée. Il s’agit d’une instanciation hybride intégrant Unified Process (UP) avec Scrum.

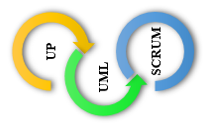
Le langage de modélisation couplé à ce processus hybride est UML que nous utilisé.

Figure : Notre choix de la méthode

## Justification du choix de la méthodologie de conception

Notre choix de démarche s’est fait en prenant en compte des spécifications essentielles de la plateforme à concevoir et des caractéristiques principales des autres méthodes de conception.

Nous synthétisons dans le tableau ci-dessous les critères retenus pour évaluer ces méthodes :

Tableau : Tableau comparatif des méthodes de conception

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Critères\Méthodes | Méthodes cartésiennes | Méthodes  systémiques | Méthodes  agiles (Scrum) | Méthodes  Objet | Méthodes  formelles |
| Itératif |  |  |  |  |  |
| Orienté utilisateur |  |  |  |  |  |
| Prise en charge de projets d’envergure |  |  |  |  |  |
| Incrémental |  |  |  |  |  |
| Axé sur la documentation |  |  |  |  |  |
| Gestion des risques |  |  |  |  |  |
| Simplicité de mise en œuvre |  |  |  |  |  |
| Langage de modélisation |  |  |  |  |  |
| Dialogue entre intervenants |  |  |  |  |  |
| Pilotage par les cas d’utilisation |  |  |  |  |  |
| Axé sur le développement |  |  |  |  |  |

Le Processus Unifié s’est imposé comme processus de développement itératif pour la construction des systèmes orientés objet. Ce processus est très souple et très ouvert, il incite à inclure des pratiques judicieuses issues d’autres méthodes itératives, telles Scrum.

Pour des raisons d'efficacité, de rapidité et d'analyse complète, et compte tenu des critères retenus, nous opterons donc pour un processus situé à mi-chemin entre UP (Unified Process), un cadre général très complet de processus de développement, et Scrum une approche centrée sur le code. Ce processus hybride est adapté aux projets robustes.

# CHAPITRE 2 : SPRINT 0

Après avoir choisie une méthodologie d’analyse et de conception, il est question de l’appliquer. Le sprint 0 représente la première phase de la méthode Scrum ; c’est une étape très importante car elle nous permet d’établir quoi faire, qui fait quoi, quand faire.

Dans ce sprint sont définis les besoins fonctionnels et techniques de l’application, les acteurs du systèmes, le product backlog ainsi que les planifications des sprints.

## Spécifications des besoins

Cette phase consiste à comprendre le contexte du système. Il s'agit de déterminer les fonctionnalités et les acteurs les plus pertinents, de préciser les risques les plus critiques et d'identifier les cas d'utilisation initiaux.

### Identification des acteurs

Un acteur est une personne, un matériel ou un logiciel qui interagit avec le système dans le but de réaliser une plus-value. Il représente un type d’utilisateur. Les acteurs de notre système sont au nombre de :

* **Administrateur** : Il est chargé de la gestion des profils utilisateurs du système, de l’attribution des droits d’accès au système de gestion de base de données et la sauvegarde des informations.
* **Gestionnaire de projet** : Il est chargé de la création du projet, de la définition de ses paramètres et de celle des tâches, ainsi que de la constitution du groupe de travail. C’est lui qui s’occupe du suivi du projet.
* **Participant** : Il participe à la gestion d’une tâche.

### Besoins fonctionnels

Pour mener à bien ce travail, nous avons effectué plusieurs recherches pour identifier au mieux les besoins de l’application, ceci afin de répondre aux attentes des utilisateurs et définir le cadre de notre système ; ce qui nous a permis de recenser les besoins préliminaires suivants:

* **Faciliter la gestion des projets** en mettant en place un système de contrôle de cohérence de données pour diminuer les risques d’erreur ;
* **Faciliter la gestion des tâches** du projet ;
* Vérifier l’accomplissement de toutes les tâches du projet avant la clôture du projet ;
* **si un participant ne termine pas une tâche**, lui faire parvenir des notifications jusqu’à ce que la tâche soit totalement accomplie ;
* **Etablir un rapport de performance pour chaque participant au projet** ;
* **Elaborer un tableau de bord** pour le suivi des activités du projet.

A cet stade le système doit permettre aux utilisateurs qui sont les employés membre de l’équipe de projet appartenant à chaque institution de :

* Soumettre un projet,
* Créer des groupes de travail ,
* Ajouter des membres dans les groupes de travail,
* Définir les droits d’accès aux documents,
* Faire une demande de décaissement ,
* Voir les derniers financement disponibles,
* Voir le bilan financier de chaque projet,
* Voir le taux de réalisation de chaque projet,
* Définir les différentes phases ou étapes du projet,
* Définir les taches et les allouer aux membres du groupe de travail qui sont des employés,
* Accéder aux liste des taches
* Valider les taches,
* Ajouter les différentes opérations financières sur chaque financement,

### Besoins techniques

Les besoins techniques représentent les exigences implicites auxquelles le système doit répondre. Ces besoins sont entre autre :

* Le système doit être une application web;
* Faciliter la gestion des utilisateurs et le paramétrage du système ;
* La recherche d’une information en temps réel ;
* Rapidité des mises à jour des indicateurs grâce à une interaction avec le logiciel
* Traiter des volumes de données de plus en plus grand ;
* Centralisation des données dans une base de données;
* Fiabilité des données
* L'ergonomie et la convivialité : L'application fournira une interface conviviale et simple à utiliser et qui ne requiert aucun pré requis, donc elle pourra être exploitable par tout type d'utilisateurs (même les non informaticiens).
* La sécurité : L'accès aux informations n'est possible qu'après vérification des privilèges et des droits d'accès. Ainsi tout utilisateur passera par une phase d'authentification pour pouvoir consulter les services ouverts par l'application.
* L'extensibilité : L'architecture de l'application permettra l'évolution et la maintenance (ajout ou suppression ou mise à jour) au niveau de ses différents modules d'une manière flexible.
* Contraintes humaines : Notre solution doit tenir compte des besoins de chaque acteur participant dans la gestion des changements. Elle doit leur faciliter la tâche et
* Assurer leur activité dans les bonnes conditions. Donc, notre application doit être

## Management du projet avec Scrum

Le management de la méthode agile Scrum est divisé entre trois pôles qui ne changent pas d’identité lors du déroulé de projet et qui ne cessent d’interchanger entre eux.

### Equipes et rôles

* **Le** **ScrumMaster** est, comme son nom l’indique, la personne en charge du management du projet. Il doit faire appliquer les pratiques et les valeurs du Scrum.

Il s’assure que l’équipe est totalement opérationnelle et productive. En cas de problème, c’est lui qui doit éliminer les obstacles pour que l’équipe puisse continuer le projet. Il agit comme facilitateur au sein de l’équipe.

* **Le Product Owner** est le responsable du produit à développer. C’est lui qui a pour mission de définir les fonctionnalités du produit final. C’est aussi lui qui choisit les dates et contenus des différentes versions. Il définit les priorités dans les fonctionnalités et valide ou non les travaux à chaque fin de Sprint. C’est à lui que revient aussi la responsabilité du retour sur investissement ainsi que l’acceptation ou le rejet des résultats (représentant généralement le client).
* **Development Team**est composée de tous les profils nécessaires à la réussite du projet. Elle est responsable d’implémenter les fonctionnalités du produit, de résoudre les problèmes techniques et de s’organiser

### Product Backlog (carnet du produit)

Tableau : Product backolg

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **N°** | **Feature** | **ID story** | **User story** | **Importance** | **Risque** |
| 1 | Gestion des utilisateurs | 1.1 | En tant que administrateur, je veux pouvoir enregistrer un utilisateur | Elevée | Elevée |
| 1.2 | En tant que administrateur, je veux pouvoir modifier un utilisateur | Elevée | Elevée |
| 1.3 | En tant que administrateur, je veux pouvoir supprimer un utilisateur | Elevée | Elevée |
| 1.4 | En tant que administrateur, je veux pouvoir consulter la liste des utilisateurs | Elevée | Elevée |
| 1.5 | En tant que administrateur, je veux pouvoir attribuer les droits d’accès | Elevée | Elevée |
| 1.6 | En tant que administrateur, je veux pouvoir révoquer les droits d’accès | Elevée | Elevée |
| 2 | Gestion et suivi des projets | 2.1 | En tant que gestionnaire de projet, je veux pouvoir enregistrer un projet | Elevée | Elevée |
| 2.2 | En tant que gestionnaire de projet, je veux pouvoir modifier un projet | Elevée | Elevée |
| 2.3 | En tant que gestionnaire de projet, je veux pouvoir financer un projet | Elevée | Elevée |
| 2.4 | En tant que gestionnaire de projet, je veux pouvoir faire une demande de décaissement pour un projet | Elevée | Elevée |
| 2.5 | En tant que a gestionnaire de projet, je veux pouvoir consulter les projets | Elevée | Elevée |
| 2.6 | En tant que a gestionnaire de projet, je veux pouvoir consulter la liste des financements pour un projet | Elevée | Elevée |
| 2.7 | En tant que a gestionnaire de projet, je veux pouvoir consulter la liste des demandes de décaissements pour un projet | Elevée | Elevée |
| 3 | Gestion étapes | 3.1 | En tant que gestionnaire de projet, je veux pouvoir enregistrer une étape d’un projet | Moyen | Elevée |
| 3.2 | En tant que gestionnaire de projet, je veux pouvoir modifier une étape d’un projet | Moyen | Elevée |
| 3.3 | En tant que gestionnaire de projet, je veux pouvoir consulter la liste des étapes d’un projet afin | Moyen | Elevée |
| 4 | Gestion des tâches | 4.1 | En tant que gestionnaire de projet, je veux pouvoir enregistrer une tâche d’un projet | Moyen | Elevée |
| 4.2 | En tant que gestionnaire de projet, je veux pouvoir modifier une tâche d’un projet | Moyen | Elevée |
| 4.3 | En tant que gestionnaire de projet, je veux pouvoir consulter la liste des tâches d’un projet | Moyen | Elevée |
| 4.4 | En tant que participant, je veux pouvoir modifier une tâche d’un projet | Moyen | Faible |
| 5 | Gestion groupe de travail | 5 | En tant que gestionnaire de projet, je veux pouvoir enregistrer un participant | Elevé | Moyen |
| 6 | Gestion paramétrage | 6 | En tant que administrateur, je veux pouvoir paramétrer la plateforme | Elevé | Elevé |
| 7 | Gestion de l’authentification | 7.1 | En tant que administrateur, je veux pouvoir m’authentifier | Elevé | Elevé |
| 7.2 | En tant que gestionnaire de projet, je veux pouvoir m’authentifier | Elevé | Elevé |
| 7.3 | En tant que participant, je veux pouvoir m’authentifier | Elevé | Elevé |
| 8 | Messagerie et Notifications | 8.1 | En tant que gestionnaire de projet, je veux pouvoir envoyer des messages aux participants | Moyen | Elevé |
| 8.2 | En tant que gestionnaire de projet, je veux pouvoir envoyer des messages a un groupe de travail | Moyen | Elevé |
| 8.3 | En tant que gestionnaire de projet, je veux pouvoir recevoir des alertes sur le projet | Elevé | Elevé |
| 8.4 | En tant que participant, je veux pouvoir recevoir des alertes sur les tâches | Moyen | Elevé |

### Structure et découpage

Tableau : Structure et découpage

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SPRINT** | **Module** | **Durée** | **Date début** | **Date fin** |
| Sprint 0 | Product Backlog | 2 semaines | 08/10/2019 | 22/10/2019 |
| Sprint 1 | Gestion des utilisateurs | 1 semaines | 23/10/2019 | 30/10/2019 |
| Sprint 2 | Gestion, suivi et financement des projets | 8 semaines | 01/11/2019 | 28/12/2019 |
| Sprint 3 | Messagerie et Notification | 2 semaines | 29/12/2019 | 13/01/2020 |

# CHAPITRE 3 : SPRINT 1, GESTION DES UTILISATEURS

Dans le chapitre précèdent, nous avons effectué à l’analyse des besoins, puis établie le découpage de notre projet.

L’étape la plus cruciale dans le processus Scrum est le sprint. Celui-ci va être réalisé pendant une certaine période de temps, à l’issue de laquelle, l’équipe doit présenter un produit fonctionnel et livrable. Nous avons dressé une liste de toutes les users stories(histoires d’utilisateurs) avec leurs affiliations à des features.

Le tableau ci-dessous regroupe toutes les fonctionnalités qui seront développées au cours de ce sprint. Ces users stories vont passer par quatre étapes du cycle Scrum à savoir : la spécification fonctionnelle, la conception, le développement et les tests.

## User Stories

Tableau : User Stories sprint 1

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **User stories** | **ID Task** | **Task** |
| 1.1 | En tant que administrateur, je veux pouvoir enregistrer un utilisateur | 1.1.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Enregistrer » |
| 1.1.b | Développer le cas « Enregistrer » |
| 1.1.c | Tester le cas « Enregistrer » |
| 1.2 | En tant que administrateur, je veux pouvoir modifier un utilisateur | 1.2.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Modifier » |
| 1.2.b | Développer le cas « Enregistrer » |
| 1.2.c | Tester le cas « Modifier » |
| 1.3 | En tant que administrateur, je veux pouvoir supprimer un utilisateur | 1.3.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Supprimer » |
| 1.3.b | Développer le cas « Supprimer » |
| 1.3.c | Tester le cas « Supprimer » |
| 1.4 | En tant que administrateur, je veux pouvoir consulter la liste des utilisateurs | 1.4.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Consulter » |
| 1.4.b | Développer le cas « Consulter » |
| 1.4.c | Tester le cas « Consulter » |
| 1.5 | En tant que administrateur, je veux pouvoir faire le paramétrage | 1.5.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Paramétrage » |
| 1.5.b | Développer le cas « Paramétrage » |
| 1.5.c | Tester le cas « Paramétrage » |
| 1.6 | En tant qu’utilisateur, je veux pouvoir m’authentifier | 1.6.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « S’authentifier » |
| 1.6.b | Développer le cas « S’authentifier » |
| 1.6.c | Tester le cas « S’authentifier » |
| 1.7 | En tant que administrateur, je veux pouvoir ajouter les droits d’accès | 1.7.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « attribuer droit d’accès » |
| 1.7.b | Développer le cas « attribuer droit d’accès  » |
| 1.7.c | Tester le cas « attribuer droit d’accès  » |
| 1.8 | En tant que administrateur, je veux pouvoir révoquer les droits d’accès | 1.8.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « révoquer droit d’accès » |
| 1.8.b | Développer le cas « révoquer droit d’accès » |
| 1.8.c | Tester le cas « révoquer droit d’accès » |
| 1.9 | En tant que administrateur, je veux pouvoir paramétrer la plateforme | 1.9.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Paramétrage » |
| 1.9.b | Développer le cas « Paramétrage » |
| 1.9.c | Tester le cas « Paramétrage » |

Lors de la première étape de chaque itération, la spécification fonctionnelle se traduit par un diagramme de cas d’utilisation. Celui-ci donne une vue extérieure du système et définit les liens entre les utilisateurs et les fonctionnalités que propose celui-ci.

## Classification des cas d’utilisation par acteurs

Tableau : Classification des cas d'utilisations par acteurs sprint 1

|  |  |
| --- | --- |
| Acteurs | Cas d’utilisations |
|  | S’authentifier |
|  | S’authentifier  Gestion des utilisateurs  Gestion paramétrage |

## Diagramme de cas d’utilisation



Figure : Diagramme de cas d'utilisation sprint 1

## Description textuelles des cas d’utilisation

Tableau : Cas d'utilisation <S'authentifier>

|  |
| --- |
| **Cas d’utilisation « S’authentifier »** |
| **Titre :** s’authentifier  **Objectif :** Permettre à l’utilisateur d’accéder à son espace personnel.  **Résumé :** L’utilisateur s’identifie et le système le renvoie dans son espace personnel.  **Acteurs :** Utilisateur, propriétaire  **Date création :** 24/09/19  **Date mise à jour :** 24/09/19 |
| **Précondition :**   * Le système est opérationnel   **Scénario nominal**  Ce cas d’utilisation commence lorsque l’utilisateur arrive sur la page d’authentification.   1. Le système affiche la page d’authentification ; 2. L’utilisateur saisit les informations d’identification puis valide ; 3. Le système vérifie l’authenticité des informations fournies par l’utilisateur ; 4. Le système affiche l’espace personnel de l’utilisateur.   **Scénario alternatif :**  A1 : Informations d’identification erronées  Ceci intervient au point 3 du scénario nominal  A1.1 : Le système invite l’utilisateur à recommencer en lui affichant un message.  Le scénario reprend au point 2. |

Tableau : Cas d'utilisation <Gérer les utilisateurs>

|  |
| --- |
| **Cas d’utilisation « Gérer les utilisateurs »** |
| **Titre :** Gérer les utilisateurs  **Objectif :** Permettre de gérer les utilisateurs du système.  **Résumé :** Ce cas d’utilisations permet au propriétaire de créer, consulter, modifier, supprimer un utilisateur.  **Acteurs :** Propriétaire  **Date création :** 25/09/19  **Date mise à jour :** 28/09/19 |
| **Précondition :**   * Le système est opérationnel * Le propriétaire s’est authentifié   **Scénario nominal**  Ce cas d’utilisation commence quand le propriétaire ouvre la page de gestion des utilisateurs   1. Le système affiche la page de gestion des utilisateurs. 2. Le propriétaire choisit une option. 3. Il ferme la page de gestion des utilisateurs.   **Scénario alternatif :**  Ces enchainements interviennent au point 2 du scénario nominal.  A1 : Le propriétaire choisit de créer un utilisateur.  A1.1 : Le système affiche la page de création d’un utilisateur.  A1.2 : Le propriétaire remplit le formulaire avec les informations de l’utilisateur à créer puis valide.  A1.3 : Le système vérifie la validité des informations saisies puis crée l’utilisateur.  A1.4 : Le système retourne au point 2 du scénario nominal.  **Post-condition**   * Le nombre des utilisateurs du système a augmenté de 1.   A2 : Le propriétaire choisit de modifier un utilisateur.  A2.1 : Le système renvoie les données à modifier ;  A2.2 : Le propriétaire modifie les informations puis valide la modification ;  A2.3 : Le système vérifie les informations saisies.  A2.4 : Le système effectue la modification et retourne au point 2 du scénario nominal.  A3 : Le propriétaire choisit de consulter les informations d’un utilisateur.  A3.1 : Le système affiche les informations de l’utilisateur.  A3.2 : Le propriétaire consulte les informations et ferme la page.  A3.3 : Le système retourne au point 2 du scénario nominal.  A4 : Le propriétaire choisit de supprimer un utilisateur.  A4.1 : Le système affiche un message de confirmation.  A4.2 : Le propriétaire confirme la suppression.  A4.3 : Le système supprime l’utilisateur puis retourne au point 2 du scénario nominal.  **Post-condition**   * Le nombre d’utilisateur du système a diminué de 1.   **Scénario d’exception :**  E1 : L’utilisateur est connecté.  Cette exception intervient aux points A2.4 et A4.3.  E1.1 : Le système renvoie un message d’erreur.  E1.2 : Le système arrête le cas d’utilisation et revient à l’espace de l’utilisateur.  **Scénario alternatif :**  E1 : saisie non valide ;  Ceci intervient aux points A1.3 et A2.3.  E1.1 : Le système renvoie un message signalant l’invalidité de la saisie et invite le propriétaire à recommencer.  Le scénario reprend au point A1.2. |

Tableau : Cas d'utilisation <Attribuer ou révoquer les droits d'accès>

|  |
| --- |
| **Cas d’utilisation « Attribuer ou révoquer des droits »** |
| **Titre :** Attribuer ou révoquer des droits  **Objectif :** Attribuer des droits à l’utilisateur  **Résumé :** Le propriétaire décide d’attribuer ou révoquer certains droits à l’utilisateur.  **Acteurs :** Propriétaire  **Date création :** 26/09/19  **Date mise à jour :** 27/09/19 |
| **Précondition :**   * Le système est opérationnel * Le propriétaire s’est authentifié   **Scénario nominal**  Ce cas d’utilisation débute quand le propriétaire ouvre la page d’attribution ou révocation des droits d’utilisateurs.   1. Le système affiche la liste des utilisateurs déjà enregistrés. 2. Le propriétaire choisit l’utilisateur concerné dans la liste. 3. Le système affiche une liste des droits à attribuer. 4. Le propriétaire sélectionne les droits à attribuer ou révoquer à l’utilisateur puis valide. 5. Le système enregistre les informations sélectionnées et renvoie un message de confirmation au propriétaire. |

Tableau : Cas d'utilisation <Gérer paramétrage>

|  |
| --- |
| **Cas d’utilisation « Gérer paramétrage »** |
| **Titre :** Gérer paramétrage  **Objectif :** Permettre de gérer le paramétrage du système.  **Résumé :** Ce cas d’utilisations permet au propriétaire de créer, consulter, modifier, supprimer un paramétrage.  **Acteurs :** Propriétaire  **Date création :** 26/09/19  **Date mise à jour :** 01/10/19 |
| **Précondition :**   * Le système est opérationnel * Le propriétaire s’est authentifié   **Scénario nominal**  Ce cas d’utilisation commence quand le propriétaire ouvre la page de gestion du paramétrage   1. Le système affiche la page de gestion du paramétrage. 2. Le propriétaire choisit une option. 3. Il ferme la page de gestion du paramétrage.   **Scénario alternatif :**  Ces enchainements interviennent au point 2 du scénario nominal.  A1 : Le propriétaire choisit de créer un paramétrage.  A1.1 : Le système affiche la page de création d’un paramétrage.  A1.2 : Le propriétaire remplit le formulaire avec les informations du paramétrage puis valide.  A1.3 : Le système vérifie la validité des informations saisies puis crée un paramétrage.  A1.4 : Le système retourne au point 2 du scénario nominal.  **Post-condition**   * Le nombre des paramétrages du système a augmenté de 1.   A2 : Le propriétaire choisit de modifier un paramétrage.  A2.1 : Le système renvoie les données à modifier ;  A2.2 : Le propriétaire modifie les informations puis valide la modification ;  A2.3 : Le système vérifie les informations saisies.  A2.4 : Le système effectue la modification et retourne au point 2 du scénario nominal.  A3 : Le propriétaire choisit de consulter les informations d’un paramétrage.  A3.1 : Le système affiche les informations du paramétrage.  A3.2 : Le propriétaire consulte les informations et ferme la page.  A3.3 : Le système retourne au point 2 du scénario nominal.  A4 : Le propriétaire choisit de supprimer un paramétrage.  A4.1 : Le système affiche un message de confirmation.  A4.2 : Le propriétaire confirme la suppression.  A4.3 : Le système supprime l’utilisateur puis retourne au point 2 du scénario nominal.  **Post-condition**   * Le nombre de paramétrage du système a diminué de 1.   **Scénario d’exception :**  E1 : L’utilisateur est connecté.  Cette exception intervient aux points A2.4 et A4.3.  E1.1 : Le système renvoie un message d’erreur.  E1.2 : Le système arrête le cas d’utilisation et revient à l’espace de l’utilisateur.  **Scénario alternatif :**  E1 : saisie non valide ;  Ceci intervient aux points A1.3 et A2.3.  E1.1 : Le système renvoie un message signalant l’invalidité de la saisie et invite le propriétaire à recommencer.  Le scénario reprend au point A1.2. |

# CHAPITRE 4 : SPRINT 2, GESTION, SUIVI ET FINANCEMENT DES PROJETS

## User Stories

Tableau : User Stories sprint 2

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **User stories** | **ID Task** | **Task** |
| 1.1 | En tant que gestionnaire de projet, je veux pouvoir enregistrer un projet | 1.1.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Enregistrer projet» |
| 1.1.b | Développer le cas « Enregistrer projet» |
| 1.1.c | Tester le cas « Enregistrer projet » |
| 1.2 | En tant que gestionnaire de projet, je veux pouvoir modifier un projet | 1.2.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Modifier projet » |
| 1.2.b | Développer le cas « Modifier projet » |
| 1.2.c | Tester le cas « Modifier projet » |
| 1.3 | En tant que gestionnaire de projet, je veux pouvoir financer un projet | 1.3.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Financer projet » |
| 1.3.b | Développer le cas « Financer projet » |
| 1.3.c | Tester le cas « Financer projet » |
| 1.4 | En tant que gestionnaire de projet, je veux pouvoir faire une demande de décaissement pour un projet | 1.4.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Décaissement » |
| 1.4.b | Développer le cas « Décaissement » |
| 1.4.c | Tester le cas « Décaissement » |
| 1.5 | En tant que a gestionnaire de projet, je veux pouvoir consulter les projets | 1.5.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Consulter projet » |
| 1.5.b | Développer le cas « Consulter  projet » |
| 1.5.c | Tester le cas « Consulter projet  » |
| 1.6 | En tant que a gestionnaire de projet, je veux pouvoir consulter la liste des demandes de décaissements pour un projet | 1.6.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Consulter liste décaissement » |
| 1.6.b | Développer le cas « Consulter liste décaissement » |
| 1.6.c | Tester le cas « Consulter liste décaissement » |
| 1.7 | En tant que gestionnaire de projet, je veux pouvoir enregistrer une étape d’un projet | 1.7.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Enregistrer étape  » |
| 1.7.b | Développer le cas « Enregistrer étape» |
| 1.7.c | Tester le cas « Enregistrer étape  » |
| 1.8 | En tant que gestionnaire de projet, je veux pouvoir modifier une étape | 1.8.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Modifier étape » |
| 1.8.b | Développer le cas « Modifier étape » |
| 1.8.c | Tester le cas « Modifier étape » |
| 1.9 | En tant que gestionnaire de projet, je veux pouvoir consulter la liste des étapes d’un projet | 1.9.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Consulter étapes » |
| 1.9.b | Développer le cas « Consulter étapes » |
| 1.9.c | Tester le cas « Consulter étapes » |
| 1.10 | En tant que gestionnaire de projet, je veux pouvoir enregistrer une tâche d’un projet | 1.10.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Enregistrer tâche » |
| 1.10.b | Développer le cas « Enregistrer tâche  » |
| 1.10.c | Tester le cas « Enregistrer tâche  » |
| 1.11 | En tant que gestionnaire de projet, je veux pouvoir Modifier une tâche d’un projet | 1.11.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Modifier une tâche » |
| 1.11.b | Développer le cas « Modifier une tâche» |
| 1.11.c | Tester le cas « Modifier une tâche  » |
| 1.12 | En tant que gestionnaire de projet, je veux pouvoir affecter une ou plusieurs tâches a un participant | 1.12.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « affecter une ou plusieurs tâches » |
| 1.12.b | Développer le cas « affecter une ou plusieurs tâches» |
| 1.12.c | Tester le cas « affecter une ou plusieurs tâches » |

## Classification des cas d’utilisation par acteurs

Tableau : Classification des cas d'utilisations par acteurs sprint 2

|  |  |
| --- | --- |
| Acteurs | Cas d’utilisations |
|  | Gestion et suivi des projets  Gestion des tâches  Gestion des étapes  S’authentifier |

## Diagramme de cas d’utilisation

Figure : Diagramme des cas d'utilisations sprint 2



## Description textuelles des cas d’utilisation

Tableau : Cas d'tilisation <Gérer les projets>

|  |
| --- |
| **Cas d’utilisation « Gérer les projets »** |
| **Titre :** Gérer les projets  **Objectif :** Permettre de gérer les projets du système.  **Résumé :** Ce cas d’utilisations permet au propriétaire de créer, consulter et modifier un projet.  **Acteurs :** Propriétaire  **Date création :** 05/11/2019  **Date mise à jour :** 12/11/19 |
| **Précondition :**   * Le système est opérationnel * Le propriétaire s’est authentifié   **Scénario nominal**  Ce cas d’utilisation commence quand le propriétaire ouvre la page de gestion des projets   1. Le système affiche la page de gestion des projets. 2. Le propriétaire choisit une option. 3. Il ferme la page de gestion des projets.   **Scénario alternatif :**  Ces enchainements interviennent au point 2 du scénario nominal.  A1 : Le propriétaire choisit de créer un projet.  A1.1 : Le système affiche la page de création d’un projet.  A1.2 : Le propriétaire remplit le formulaire avec les informations du projet à créer puis valide.  A1.3 : Le système vérifie la validité des informations saisies puis crée le projet.  A1.4 : Le système retourne au point 2 du scénario nominal.  **Post-condition**   * Le nombre des projets du système a augmenté de 1.   A2 : Le propriétaire choisit de modifier un projet.  A2.1 : Le système renvoie les données à modifier ;  A2.2 : Le propriétaire modifie les informations puis valide la modification ;  A2.3 : Le système vérifie les informations saisies.  A2.4 : Le système effectue la modification et retourne au point 2 du scénario nominal.  A3 : Le propriétaire choisit de consulter les informations d’un projet.  A3.1 : Le système affiche les informations du projet.  A3.2 : Le propriétaire consulte les informations et ferme la page.  A3.3 : Le système retourne au point 2 du scénario nominal.  **Scénario d’exception :**  E1 : L’utilisateur est connecté.  Cette exception intervient aux points A2.4.  E1.1 : Le système renvoie un message d’erreur.  E1.2 : Le système arrête le cas d’utilisation et revient à l’espace de l’utilisateur.  **Scénario alternatif :**  E1 : saisie non valide ;  Ceci intervient aux points A1.3 et A2.3.  E1.1 : Le système renvoie un message signalant l’invalidité de la saisie et invite le propriétaire à recommencer.  Le scénario reprend au point A1.2. |

Tableau : Cas d'utilisation <Gérer financements>

|  |
| --- |
| **Cas d’utilisation « Gérer les financements »** |
| **Titre :** Gérer les projets  **Objectif :** Permettre de gérer les projets du système.  **Résumé :** Ce cas d’utilisations permet au propriétaire de financer, consulter et modifier un financement.  **Acteurs :** Propriétaire  **Date création :** 16/11/19  **Date mise à jour :** 16/11/19 |
| **Précondition :**   * Le système est opérationnel * Le propriétaire s’est authentifié   **Scénario nominal**  Ce cas d’utilisation commence quand le propriétaire ouvre la page de gestion des projets   1. Le système affiche la page de gestion des financements. 2. Le propriétaire choisit une option. 3. Il ferme la page de gestion des financements.   **Scénario alternatif :**  Ces enchainements interviennent au point 2 du scénario nominal.  A1 : Le propriétaire choisit de financer un projet.  A1.1 : Le système affiche la page de financement d’un projet.  A1.2 : Le propriétaire remplit le formulaire avec les informations du financement d’un projet puis valide.  A1.3 : Le système vérifie la validité des informations saisies puis crée le financement d’un projet.  A1.4 : Le système retourne au point 2 du scénario nominal.  **Post-condition**   * Le nombre des financements d’un projet du système a augmenté de 1.   A2 : Le propriétaire choisit de modifier un financement.  A2.1 : Le système renvoie les données à modifier ;  A2.2 : Le propriétaire modifie les informations puis valide la modification ;  A2.3 : Le système vérifie les informations saisies.  A2.4 : Le système effectue la modification et retourne au point 2 du scénario nominal.  A3 : Le propriétaire choisit de consulter les informations d’un financement.  A3.1 : Le système affiche les informations du financement.  A3.2 : Le propriétaire consulte les informations et ferme la page.  A3.3 : Le système retourne au point 2 du scénario nominal.  **Scénario d’exception :**  E1 : L’utilisateur est connecté.  Cette exception intervient aux points A2.4.  E1.1 : Le système renvoie un message d’erreur.  E1.2 : Le système arrête le cas d’utilisation et revient à l’espace de l’utilisateur.  **Scénario alternatif :**  E1 : saisie non valide ;  Ceci intervient aux points A1.3 et A2.3.  E1.1 : Le système renvoie un message signalant l’invalidité de la saisie et invite le propriétaire à recommencer.  Le scénario reprend au point A1.2. |

Tableau : Cas d'utilisation <Gérer étapes>

|  |
| --- |
| **Cas d’utilisation « Gérer les étapes »** |
| **Titre :** Gérer les étapes  **Objectif :** Permettre de gérer les étapes du projet.  **Résumé :** Ce cas d’utilisations permet au propriétaire de créer et modifier une étape.  **Acteurs :** Propriétaire  **Date création :** 17/11/19  **Date mise à jour :** 20/11/19 |
| **Précondition :**   * Le système est opérationnel * Le propriétaire s’est authentifié   **Scénario nominal**  Ce cas d’utilisation commence quand le propriétaire ouvre la page de gestion des étapes   1. Le système affiche la page de gestion des étapes. 2. Le propriétaire choisit une option. 3. Il ferme la page de gestion des étapes.   **Scénario alternatif :**  Ces enchainements interviennent au point 2 du scénario nominal.  A1 : Le propriétaire choisit de créer une étape.  A1.1 : Le système affiche la page de création d’une étape.  A1.2 : Le propriétaire remplit le formulaire avec les informations d’une étape à créer puis valide.  A1.3 : Le système vérifie la validité des informations saisies puis crée l’étape.  A1.4 : Le système retourne au point 2 du scénario nominal.  **Post-condition**   * Le nombre d’étape du projet a augmenté de 1.   A2 : Le propriétaire choisit de modifier une étape.  A2.1 : Le système renvoie les données à modifier ;  A2.2 : Le propriétaire modifie les informations puis valide la modification ;  A2.3 : Le système vérifie les informations saisies.  A2.4 : Le système effectue la modification et retourne au point 2 du scénario nominal.  **Scénario d’exception :**  E1 : L’utilisateur est connecté.  Cette exception intervient aux points A2.4.  E1.1 : Le système renvoie un message d’erreur.  E1.2 : Le système arrête le cas d’utilisation et revient à l’espace de l’utilisateur.  **Scénario alternatif :**  E1 : saisie non valide ;  Ceci intervient aux points A1.3 et A2.3.  E1.1 : Le système renvoie un message signalant l’invalidité de la saisie et invite le propriétaire à recommencer.  Le scénario reprend au point A1.2. |

Tableau : Cas d'utilisation <Demande décaissement>

|  |
| --- |
| **Cas d’utilisation «Demande de décaissement »** |
| **Titre :** demande de décaissement  **Objectif :** Permettre au gestionnaire de projet de faire une demande de décaissement.  **Résumé :** L’utilisateur s’identifie et le système le renvoie dans son espace personnel, qui est organisé par rapport à l’institution auquel il appartient  **Acteurs :** Gestionnaire  **Date création :** 08/12/19  **Date mise à jour :** |
| **Précondition :**   * Le système est opérationnel   **Scénario nominal**  Ce cas d’utilisation commence lorsque l’utilisateur est authentifié.     1. L’utilisateur arrive à la page des décaissements réalisé sur tous les projets, 2. L’utilisateur click sur le bouton faire une demande, 3. L’utilisateur accède à la page de de demande de décaissement et saisi les   Informations nécessaires,   1. L’utilisateur voit le récapitulatif des informations saies avant de valider   la demande,   1. Le système renvoie un message si les informations saisies sont correctes   **Scénario alternatif :**  A1 : Les projets n’ont pas de financement  A1.1 : Le système renvoi un message et masque le bouton envoyer.  Le scénario reprend au point 1. |

Tableau : Cas d'utilisation <Ajouter participant au groupe de travail>

|  |
| --- |
| **Cas d’utilisation «Ajouter un participant au groupe de travail »** |
| **Titre :** **Ajouter un participant au groupe de travail**  **Objectif :** Permettre au gestionnaire de projet Ajouter un participant au groupe de travail**.**  **Résumé :** L’utilisateur s’identifie et le système le renvoie dans son espace personnel, qui est organisé par rapport à l’institution auquel il appartient  **Acteurs :** Gestionnaire de projet  **Date création :** 18/12/19  **Date mise à jour :** |
| **Précondition :**   * Le système est opérationnel   **Scénario nominal**  Ce cas d’utilisation commence lorsque l’utilisateur est authentifié.     1. L’utilisateur arrive à la page de liste des groupes de travail, 2. L’utilisateur sélectionne le groupe de travail du projet pour lequel il veut   Ajouter un participant,   1. L’utilisateur accède à l’espace de travail qui est aussi le groupe de travail, 2. L’utilisateur action le bouton ajouter un membre, 3. L’utilisateur sélectionne parmi les employé, un employé à ajouter   **Scénario alternatif :**  A1 : L’institution du gestionnaire n’a pas d’employé  A1.1 : Le système propose à l’utilisateur d’inviter un employé via adresse email |

Tableau : Cas d'utilisation <Ajouter tâche>

|  |
| --- |
| **Cas d’utilisation «Ajouter une tâche »** |
| **Titre :** Ajouter une tache  **Objectif :** Permettre au gestionnaire de projet d’ajouter une tache et de l’affecter à un membre du projet  **Résumé :** L’utilisateur s’identifie et le système le renvoie dans son espace personnel, qui est organisé par rapport à l’institution auquel il appartient  **Acteurs :** Gestionnaire  **Date création :** 18/09/16  **Date mise à jour :** 18/09/16 |
| **Précondition :**   * Le système est opérationnel   **Scénario nominal**  Ce cas d’utilisation commence lorsque l’utilisateur est authentifié.     1. L’utilisateur arrive à la page de liste des groupes de travail, 2. L’utilisateur sélectionne le groupe de travail du projet pour lequel il veut   Ajouter une tache,   1. L’utilisateur accède à l’espace de travail qui est aussi le groupe de travail, 2. L’utilisateur action le bouton ajouter une tache et sélectionne les employés qui doivent   Exécuter la tache   1. Le système renvoie un message si les informations saisies sont correctes   **Scénario alternatif :**  A1 : Le projet n’a pas des étapes ou des phases  A1.1 : Le système propose à l’utilisateur de créer une étape.  Le scénario reprend au point 3. |

# CHAPITRE 5 : SPRINT 3, MESSAGERIES ET NOTIFICATIONS

## User Stories

Tableau : User Stories sprint 3

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **User stories** | **ID Task** | **Task** |
| 1.1 | En tant que gestionnaire de projet, je veux pouvoir enregistrer un projet | 1.1.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Enregistrer projet» |
| 1.1.b | Développer le cas « Enregistrer projet» |
| 1.1.c | Tester le cas « Enregistrer projet » |
| 1.2 | En tant que gestionnaire de projet, je veux pouvoir modifier un projet | 1.2.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Modifier projet » |
| 1.2.b | Développer le cas « Modifier projet » |
| 1.2.c | Tester le cas « Modifier projet » |
| 1.3 | En tant que gestionnaire de projet, je veux pouvoir financer un projet | 1.3.a | Réaliser le diagramme de cas d’utilisation, séquence, de classe de la fonctionnalité « Financer projet » |
| 1.3.b | Développer le cas « Financer projet » |
| 1.3.c | Tester le cas « Financer projet » |

A cet stade le système doit permettre aux utilisateurs qui sont les employés membre de l’équipe de projet appartenant à chaque institution de :

* Envoyer et de recevoir des messages d’une manière instantané,
* Voir les messages de groupe et interagir,
* Recevoir des notifications sur les projets,
* Recevoir des notifications sur les financements,

## Classification des cas d’utilisation par acteurs

Tableau : Classification des cas d'utilisations par acteur sprint 3

|  |  |
| --- | --- |
| Acteurs | Cas d’utilisations |
|  | Message  Notifications ou alertes |

## Diagramme de cas d’utilisation



Figure : Diagramme de cas d'utilisation sprint 3

## Description textuelles des cas d’utilisation

Tableau Cas d'utilisation <Notification ou alertes:

|  |
| --- |
| **Cas d’utilisation « Notifications ou alertes »** |
| **Titre :** Notifications ou alertes  **Objectif :** Notifications ou alertes  **Résumé :** Le système envoie des notifications a des utilisateurs.  **Acteurs :** Propriétaire  **Date création :** 11/12/19  **Date mise à jour :** 26/12/19 |
| **Précondition :**   * Le système est opérationnel * Le propriétaire s’est authentifié   **Scénario nominal**   1. Le système envoie des notifications aux utilisateurs tenant compte des spécificités bien définies. |

# PARTIE D

# GESTION DE PROJET

# CHAPITRE 1 : CONDUITE DE PROJET

Dans cette partie, nous présentons dans un premier temps l’organisation que nous avons adoptée pour mener à terme notre projet, notamment son découpage en phase et en tâches, les différents intervenants au projet, sa planification et le rendu des charges globales du projet.

Un projet est un processus unique qui consiste en un ensemble d'activités coordonnées et maîtrisées, comportant des dates de début et de fin, entrepris dans le but d'atteindre un objectif conforme à des exigences spécifiques, incluant des contraintes de délai, de coût et de ressource ; il fait donc l'objet d'une budgétisation de moyens et d'un bilan indépendant de celui de l'entreprise. La gestion d’un projet devient alors la démarche visant à organiser de bout en bout le bon déroulement du projet, coordonner, diriger et surtout contrôler les ressources utilisées et ce, de façon structurée et optimale.

Nous présenterons dans cette dernière partie du document l’organisation de notre projet notamment, sa planification en tâches et en phases, les estimations des ressources humaines, matérielles et financières. Nous terminerons par le bilan et les perspectives du travail effectué.

## Intervenant au projet

Au cours de notre stage, nous avons travaillé avec plusieurs personnes dont le responsable du projet, notre encadreur et notre superviseur académique :

Tableau : Intervenant au projet

|  |  |  |
| --- | --- | --- |
| **Intervenants** | **Titre** | **Rôle Scrum** |
| **Rodrigue TCHOKOTHE** | Encadreur | Scrum Master, Product owner |
| **DOMINIQUE MAOUNDONGONE Damba** | Etudiant en 3A AP | Scrum Team |
| **VIDILA BAKABADIO Saint Jelys** | Etudiant en 3A AP | Scrum Team |

## Planification en tâches

Tableau : Planification en tâches

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N° | Nom de la tâche | Durée | Début | Fin | Prédécesseurs |
| 1 | **ETUDE PRELIMINAIRE** | **18 jrs** | **Mar 01/10/19** | **Mar 29/10/19** |  |
| 2 | Début projet | 0 jr | Mar 01/10/19 | Mar 01/10/19 |  |
| 3 | Prise de contact et exploration | 2 jrs | Mar 01/10/19 | Jeu 03/10/19 | 2 |
| 4 | Interviews, recensement des attentes du projet | 3 jrs | Jeu 03/10/19 | Mar 08/10/19 | 3 |
| 5 | Etude préliminaire des outils de développements | 7 jrs | Mar 08/10/19 | Ven 18/10/19 | 4 |
| 6 | Etude des concepts liés au domaine | 3 jrs | Ven 18/10/19 | Jeu 24/10/19 | 5 |
| 7 | Lecture portant sur les méthodologies agiles | 3 jrs | Jeu 24/10/19 | Mar 29/10/19 | 6 |
| 8 | **SPRINT 0** | **5 jrs** | **Mar 29/10/19** | **Mer 06/11/19** |  |
| 9 | Définition du product backlog | 2 jrs | Mar 29/10/19 | Jeu 31/10/19 | 7 |
| 10 | Définition de l'architecture de la base de données | 2 jrs | Jeu 31/10/19 | Mar 05/11/19 | 9 |
| 11 | Choix de l'architecture technique | 1 jr | Mar 05/11/19 | Mer 06/11/19 | 10 |
| 12 | Mise en place du design pattern MVC | 1 jr | Mar 05/11/19 | Mer 06/11/19 | 10 |
| 13 | Planification du sprint 1 | 1 jr | Mar 05/11/19 | Mer 06/11/19 | 10 |
| 14 | **SPRINT 1 : Gestion des utilisateurs** | **13 jrs** | **Mer 06/11/19** | **Mer 27/11/19** |  |
| 15 | Spécifications fonctionnelles | 1 jr | Mer 06/11/19 | Jeu 07/11/19 | 13 |
| 16 | Conception | 3 jrs | Jeu 07/11/19 | Mar 12/11/19 | 15 |
| 17 | Développement et test | 7 jrs | Mar 12/11/19 | Ven 22/11/19 | 16 |
| 18 | Revue du sprint | 2 jrs | Ven 22/11/19 | Mer 27/11/19 | 17 |
| 19 | Rétrospective du sprint | 2 jrs | Ven 22/11/19 | Mer 27/11/19 | 17 |
| 20 | Planification du sprint 2 | 1 jr | Ven 22/11/19 | Lun 25/11/19 | 17 |
| 21 | **SPRINT 2 : Gestion et suivi des projets** | **30 jrs** | **Mar 26/11/19** | **Lun 13/01/20** |  |
| 22 | Spécifications fonctionnelles | 2 jrs | Mar 26/11/19 | Jeu 28/11/19 | 20 |
| 23 | Conception | 3 jrs | Jeu 28/11/19 | Mar 03/12/19 | 22 |
| 24 | Développement et test | 21 jrs | Mar 03/12/19 | Lun 06/01/20 | 23 |
| 25 | Revue du sprint | 4 jrs | Lun 06/01/20 | Lun 13/01/20 | 24 |
| 26 | Rétrospective du sprint | 1 jr | Lun 06/01/20 | Mar 07/01/20 | 24 |
| 27 | Planification du sprint 3 | 1 jr | Lun 06/01/20 | Mar 07/01/20 | 24 |
| 28 | **SPRINT 3 : Messagerie et Notifications** | **10 jrs** | **Mar 07/01/20** | **Jeu 23/01/20** |  |
| 29 | Spécifications fonctionnelles | 1 jr | Mar 07/01/20 | Mer 08/01/20 | 27 |
| 30 | Conception | 2 jrs | Jeu 09/01/20 | Lun 13/01/20 | 29 |
| 31 | Développement et test | 5 jrs | Lun 13/01/20 | Lun 20/01/20 | 30 |
| 32 | Revue du sprint | 2 jrs | Mar 21/01/20 | Jeu 23/01/20 | 31 |
| 33 | **FINALISATION** | **65 jrs** | **Ven 15/11/19** | **Jeu 27/02/20** |  |
| 34 | Corrections finales | 10 jrs | Jeu 23/01/20 | Ven 07/02/20 | 32 |
| 35 | Déploiement | 1 jr | Ven 07/02/20 | Lun 10/02/20 | 34 |
| 36 | Test global | 7 jrs | Lun 10/02/20 | Jeu 20/02/20 | 35 |
| 37 | Démo prototype | 1 jr | Jeu 20/02/20 | Ven 21/02/20 | 36 |
| 38 | **Rédactions Cahier des charges, conception, réalisation et programmation** | **64 jrs** | **Ven 15/11/19** | **Mer 26/02/20** | **4** |
| 39 | Préparation soutenance | 1 jr | Mer 26/02/20 | Jeu 27/02/20 | 37;38 |
| 40 | Fin | 0 jr | Jeu 27/02/20 | Jeu 27/02/20 |  |

## Diagramme de Gantt

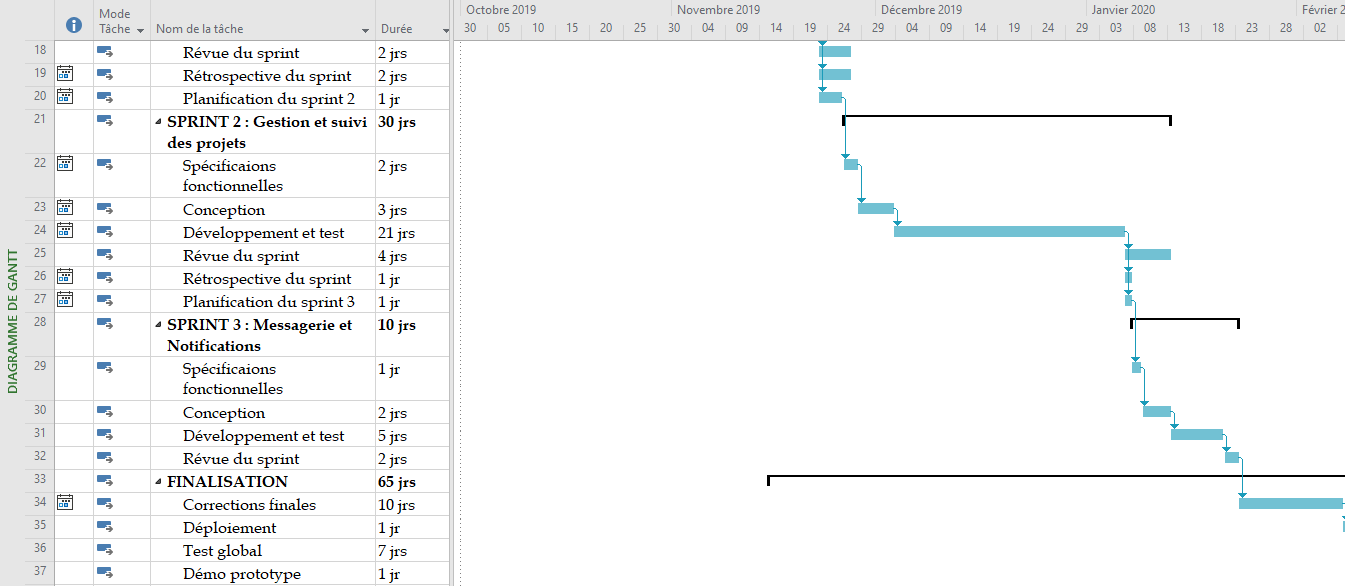


Figure : Diagramme de Gantt

# CHAPITRE 2 : ESTIMATION DES CHARGES

Nous allons effectuer une estimation du cout réel de la mise en place d’une telle solution sur le marché de l’emploi gabonais. Ensuite, nous allons procéder à un état des dépenses que nous en tant que stagiaire avons effectué dans le cadre de ce même projet.

## Charges matérielles

Tableau : Charges matérielles

|  |  |  |  |
| --- | --- | --- | --- |
| **Coût matériel** | | | |
| **Matériel** | **Quantité** | **Coût unitaire** | **Coût total** |
| Ordinateur Lenovo ThinkPad Corei5 | 1 | 250 000 Frs | 250 000 Frs |
| Ordinateur Dell | 1 | 200 000 Frs | 200 000 Frs |
| Prix Total | **450 000 Frs** | | |

## Charges logicielles

Tableau : Charges logicielles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Coût logiciel** | | | | |
| **Logiciel** | **Quantité** | **Coût unitaire** | **Coût total** | **Type Licence** |
| PowerAMC | 1 | 1 861 493 Frs | 1 861 493 Frs | A vie |
| Edraw Max | 1 | 117 700 Frs | 117 700 Frs | A vie |
| Apache Tomcat | 1 | - | - | Gratuite |
| Ms Project | 1 | 989 000 Frs | 989 000 Frs | A vie |
| **Prix Total** | **2 968 193 Frs** | | | |

## Coûts humains

Tableau : Coûts humains

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Coût humains** | | | | |
| **Participant** | **Nombre** | **Prime stage** | **Durée(Mois)** | **Coût total** |
| DOMINIQUE MAOUNDONGONE Damba | 2 | 150 000 Frs | 04 | 1 200 000 Frs |
| VIDILA BAKABADIO Saint Jelys |
| **Prix Total** | **1 200 000 Frs** | | | |

## Coûts totaux

Tableau : Coûts totaux

|  |  |
| --- | --- |
| Cout matériels | 450 000 Frs |
| Cout logiciels | 2 968 193 Frs |
| Cout humains | 1 200 000 Frs |
| **Cout total** | **4 618 193**  **Frs** |

**Sources**

<https://www.microsoft.com/>

<https://www.amazon.fr/>

<https://www.edrawsoft.com/>

# CONCLUSION

Nous avons travaillé sur ***«****Implémentation d’un système de Workflow et Groupware  : cas d’application de suivi de financements sur emprunts, contreparties et dons****»*** dans le but d’améliorer la qualité de gestion, suivi et financements des projets de la Direction Générale de la Dette.

Nous avons utilisé, pour ce faire, une méthode hybride très efficace pour la modélisation des systèmes d’informations. Cette dernière nous a permis de cerner notre sujet dans son entièreté à travers une étude du système existant sur la base de laquelle nous avons proposé la nouvelle solution. Cette solution est considérée comme étant le remède à toutes les insuffisances que nous avions pu identifier.

Ainsi la base de données que nous mettons en place est la réponse à toutes les interrogations

que l’on peut se poser. Par ailleurs, ce projet nous a permet d’approfondir nos connaissances dans la conception des systèmes d’informations avec l’utilisation de la méthode agile.

# LISTE DES ABREVIATIONS

Tableau : Liste des abréviations

|  |  |
| --- | --- |
| **Abréviations** | **Significations** |
| API | Application Programming Interface |
| HTML | HyperText Markup Language |
| PHP | Hypertext PreProcessor |
| URL | Uniform Resource Location |
| DGD | Directeur Générale de la Dette |
| IAI | Institut Africain d’Informatique |
| JPA | Java Persistence API |
| MVC | Modèle-Vue-Contrôleur |
| SGBD | Système de Gestion de Bases de Données |
| SGBDR | Système de Gestion de Bases de Données Relationnel |
| UML | Unified Modeling Language |
| UP | Unified Process |
| XML | eXtensible Markup Language |
| XP | eXtreme Programming |
| CRUD | Create Read Update Delete |
| TMP | Techno Mega Partners |
| URI | Uniform Resource Identifier |

# TABLES DES FIGURES

[Figure 1: Organigramme de TMP 12](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536369)

[Figure 2:Situation géographique de TMP 13](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536370)

[Figure 3:Architecture 2-tier 27](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536371)

[Figure 4:Architecture 3-tier 28](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536372)

[Figure 5: Architecture multi-tier 28](#_Toc35536373)

[Figure 6: Architecture de la solution choisie 30](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536374)

[Figure 7: Processus de développement en Y 36](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536375)

[Figure 8: Différents concepts de Scrum 38](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536376)

[Figure 9: Cycle d'un sprint 39](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536377)

[Figure 10: Notre choix de la méthode 40](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536378)

[Figure 11: Diagramme de cas d'utilisation sprint 1 51](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536379)

[Figure 12: Diagramme des cas d'utilisations sprint 2 60](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536380)

[Figure 13: Diagramme de cas d'utilisation sprint 3 68](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536381)

[Figure 14: Diagramme de Gantt 74](file:///C:\Users\VIDILA%20Saint\Documents\Mes%20cahiers%20de%20troisième%20année\Cahier%20des%20charges.docx#_Toc35536382)

# LISTE DES TABLEAUX

[Tableau 1:Contacts TMP 13](#_Toc35905313)

[Tableau 2: Tableau comparatif des méthodes de conception 40](#_Toc35905314)

[Tableau 3: Product backolg 45](#_Toc35905315)

[Tableau 4: Structure et découpage 47](#_Toc35905316)

[Tableau 5: User Stories sprint 1 48](#_Toc35905317)

[Tableau 6: Classification des cas d'utilisations par acteurs sprint 1 50](#_Toc35905318)

[Tableau 7: Cas d'utilisation <S'authentifier> 51](#_Toc35905319)

[Tableau 8: Cas d'utilisation <Gérer les utilisateurs> 52](#_Toc35905320)

[Tableau 9: Cas d'utilisation <Attribuer ou révoquer les droits d'accès> 54](#_Toc35905321)

[Tableau 10: Cas d'utilisation <Gérer paramétrage> 54](#_Toc35905322)

[Tableau 11: User Stories sprint 2 57](#_Toc35905323)

[Tableau 12: Classification des cas d'utilisations par acteurs sprint 2 59](#_Toc35905324)

[Tableau 13: Cas d'tilisation <Gérer les projets> 60](#_Toc35905325)

[Tableau 14: Cas d'utilisation <Gérer financements> 62](#_Toc35905326)

[Tableau 15: Cas d'utilisation <Gérer étapes> 63](#_Toc35905327)

[Tableau 16: Cas d'utilisation <Demande décaissement> 64](#_Toc35905328)

[Tableau 17: Cas d'utilisation <Ajouter participant au groupe de travail> 65](#_Toc35905329)

[Tableau 18: Cas d'utilisation <Ajouter tâche> 66](#_Toc35905330)

[Tableau 19: User Stories sprint 3 67](#_Toc35905331)

[Tableau 20: Classification des cas d'utilisations par acteur sprint 3 68](#_Toc35905332)

[Tableau 21 Cas d'utilisation <Notification ou alertes: 68](#_Toc35905333)

[Tableau 22: Intervenant au projet 71](#_Toc35905334)

[Tableau 23: Planification en tâches 72](#_Toc35905335)

[Tableau 24: Charges matérielles 75](#_Toc35905336)

[Tableau 25: Charges logicielles 75](#_Toc35905337)

[Tableau 26: Coûts humains 75](#_Toc35905338)

[Tableau 27: Coûts totaux 76](#_Toc35905339)

[Tableau 28 : Liste des abréviations 78](#_Toc35905340)

# DOCUMENTATION

## Bibliographie

* Formation sur SCRUM, Timwi Consulting,2012
* Pascal Roques & Franck Vallée. (2007). *UML 2 en action, De l’analyse des besoins à la conception 4e édition.* Eyrolles.
* Pascal Roques, *UM2 - Modéliser une application web*, 4e édition, Eyrolles
* **Banque mondiale**. Séminaire de formation en suivi évaluation. Niger : 2008.
* « **Cours de Technique de communication** » de M. Emmanuel KOUMBA, Année Académique 2018-2019
* Scrum - 4e éd.- Le guide pratique de la méthode agile la plus populaire
* Eyrolles, Scrum-de la théorie à la pratique

## Webographie

* <https://openclassrooms.com/courses/rediger-correctement-un-cahier-des-charges>

* <https://agiliste.fr/guide-de-demarrage-scrum/>
* https://www.memoireonline.com/sommaires/informatique-telecommunications.html