Table des matières

[Portail de rapport Big Data 3](#_Toc518046894)

[Introduction 3](#_Toc518046895)

[CHAPITRE I : Présentation du sujet 4](#_Toc518046896)

[Contexte 4](#_Toc518046897)

[Problématique 5](#_Toc518046898)

[Objectifs 6](#_Toc518046899)

[Objectif général 6](#_Toc518046900)

[Objectifs spécifiques 6](#_Toc518046901)

[Conclusion 6](#_Toc518046902)

[CHAPITRE II : Étude de l’existant 7](#_Toc518046903)

[Définitions des concepts du domaine 7](#_Toc518046904)

[Fonctionnalités existantes 7](#_Toc518046905)

[Les acteurs principaux 7](#_Toc518046906)

[Les modules existants 7](#_Toc518046907)

[Architecture et Technologies utilisées 7](#_Toc518046908)

[Architecture technique 7](#_Toc518046909)

[Architecture applicative 7](#_Toc518046910)

[Technologies utilisées 7](#_Toc518046911)

[Les évolutions souhaitées 7](#_Toc518046912)

[CHAPITRE III : Choix d’une méthode d’analyse et de conception 7](#_Toc518046913)

[Processus de développement 7](#_Toc518046914)

[Les rôles 14](#_Toc518046915)

[SCRUM dans notre projet 14](#_Toc518046916)

[Présentation et justification du choix d'UML 15](#_Toc518046917)

[Conclusion 17](#_Toc518046918)

[CHAPITRE IV : Spécification fonctionnelle détaillées 17](#_Toc518046919)

[Généralités 17](#_Toc518046920)

[Règle générale 17](#_Toc518046921)

[Règles liées à la validation des formulaires 17](#_Toc518046922)

[Règles liées aux actions sur les objets 18](#_Toc518046923)

[Fonctionnalités de l’application 19](#_Toc518046924)

[Fonctionnalité « Gestion des visualisations » 19](#_Toc518046925)

[Fonctionnalité « Gestion des tableaux de bord » 20](#_Toc518046926)

[Fonctionnalité « Exploration de données » 22](#_Toc518046927)

[Fonctionnalité « Gestion requête personnalisée» 22](#_Toc518046928)

[Fonctionnalité « Gestion de monitoring » 22](#_Toc518046929)

[Fonctionnalité « Gestion Utilisateur » 23](#_Toc518046930)

[Fonctionnalité « Gestion des Index » 24](#_Toc518046931)

[Conclusion 25](#_Toc518046932)

[CHAPITRE V : Conception du système 26](#_Toc518046933)

[Spécification fonctionnelle détaillée 26](#_Toc518046934)

[Catalogue 26](#_Toc518046935)

[Définition de l’acteur 26](#_Toc518046936)

[Étude préliminaire 26](#_Toc518046937)

[Diagramme de cas d’utilisation et leur descriptions 27](#_Toc518046938)

[Schéma de la base de données 29](#_Toc518046939)

[Conception détaillée 29](#_Toc518046940)

[Architecture générale du système 29](#_Toc518046941)

[Architecture technico-fonctionnelles 29](#_Toc518046942)

[Conclusion 29](#_Toc518046943)

[CHAPITRE VI : Implémentation de la solution 29](#_Toc518046944)

[Choix des outils et des technologies 29](#_Toc518046945)

[Aperçu des résultats 29](#_Toc518046946)

[Conclusion 29](#_Toc518046947)

[Annexes 29](#_Toc518046948)

# Portail de rapport Big Data

# Introduction

Chaque jour, 2,5 de téraoctets de données sont générées dans le monde. D’ici 2020, il est prévu que la taille des données se multiplie de 50 fois. Google reçoit 40 000 requêtes toutes les secondes, 72 vidéos sont mises en lignes chaque minute sur YouTube et 217 nouveaux utilisateurs de Smartphone sont comptés toutes les minutes (Miranda, 2013). Aujourd’hui l’information nous provient de toute part : des capteurs de géolocalisation, des données émanant des smartphones (logs de connexion, appels…), des données postées sur les réseaux sociaux, des vidéos et des images digitales, des transactions des clients, des capteurs de formes ou de mouvement des objets connectés… Le développement et l’accès à ces données représentent ce qui est appelé le phénomène « Big Data ». Ce phénomène impacte en particulier les entreprises qui sont amenées à manipuler des téraoctets voire des penta octets de données nécessitant une infrastructure spécifique pour leur création, leur stockage, leur traitement, leur analyse et leur récupération. En d’autres termes, il s’agit du développement en temps réel d’une masse de données volumineuse qui dépasse la capacité des outils de traitement et d’analyse traditionnels (bases de données relationnelles, requêtes SQL …). Le Big Data est une évolution de la Business Intelligence (BI). Alors que la BI traditionnelle repose sur des entrepôts de données (datawarehouses) limités en taille (quelques téraoctets), gérant difficilement des données non structurées et les analyses en temps réel, l’ère du Big Data ouvre une nouvelle période technologique offrant des architectures et infrastructures évoluées permettant des analyses sophistiquées en prenant en compte ces nouvelles données intégrées à l’écosystème de l’entreprise (Chen et al., 2012).

En effet, le « phénomène Big Data » pour les entreprises recouvre ainsi deux réalités : d’une part cette explosion des données de façon continue, d’autre part la capacité technologique de traiter et d’analyser cette grande masse de données pour en tirer un profit. Grâce au Big data, les entreprises peuvent désormais gérer et traiter des données massives pour en extraire de la valeur, décider et agir en temps réel. L’enjeu central pour elle est bien « d’améliorer l’efficacité des prises de décision par l’exploitation d’informations protéiformes » (Cigref, 2013). Elles deviennent ainsi plus réactives et plus compétitives. De fait la capacité à traiter de grande masse de données, à tisser des liens et corrélations entre des informations hétérogènes et à faire communiquer entre elles des données structurées ou non, ouvre la voie à des traitements probabilistes qui permettent l’amélioration des opérations managériales et la conception de produits et services innovants pour les clients.

Ainsi, ce présent mémoire qui est l’illustration du travail effectué dans ce stage, relate les différentes phases ainsi que les moyens auxquels nous avons eu recours pour atteindre les objectifs fixés. Ce rapport est articulé autour de cinq chapitres

* Un premier chapitre « PRESENTATION GENERALE » dans lequel nous présentons la structure d’accueil, le contexte, la problématique ainsi que les objectifs du sujet.
* Un deuxième chapitre « GENERALITES » définissant les différents concepts généraux apparaissant dans notre architecture ainsi que les outils utilisés dans la réalisation du sujet.
* Un troisième chapitre « SPECIFICATION FONCTIONNELLE » dans lequel nous présentons les fonctionnalités du système, leur cas d’utilisation ainsi que les règles métiers qui sont associés. Ce chapitre constitue la colonne vertébrale de notre sujet.
* Un quatrième chapitre « CONCEPTION DU SYSTEME » qui présente notre travail d’analyse et de conception à travers des spécifications des modèles nécessaires à chaque étape. Il présente également les différentes architectures du système.
* Un cinquième chapitre « REALISATION DE LA SOLUTION » qui présente les outils que nous avons utilisés pour la réalisation de notre **portail de rapport big data.**

# CHAPITRE I : Présentation du sujet

Dans cette section, nous présentons explicitement le sujet de notre mémoire s’intitulant « **Portail de rapport Big Data** ». Il sera question de présenter d’abord le contexte du projet ensuite énumérer la problématique et enfin exposer les différents objectifs fixés.

## Contexte

Chaque jour, nous générons 2,5 trillions d’octets de données. À tel point que 90% des données dans le monde ont été créées au cours des deux dernières années seulement. Ces données proviennent de partout : de capteurs utilisés pour collecter les informations climatiques, de messages sur les sites de médias sociaux, d'images numériques et de vidéos publiées en ligne, d'enregistrements transactionnels d'achats en ligne et de signaux GPS de téléphones mobiles, pour ne citer que quelques sources. Ces données sont appelées **Big Data ou volumes massifs de données**. Le Big Data contraint à de nouvelles manières de voir et analyser le monde. De nouveaux ordres de grandeur concernent la capture, le stockage, la recherche, le partage, l'analyse et la visualisation des données.

Pour caractériser le Big Data, on parle de la règle des 3V devenue par extension 5V qui fait référence à **Volume**, **Variété**, **Vitesse**, **Véracité** et **Valeur**. S’agissant de la notion de **volume**, on parle de Big Data au-delà de 100 To (téraoctets) de données structurées, semi-structurées ou non structurées. La **variété** nous rappelle que la diversité des données est fondamentale pour le Big Data. La **vitesse** de traitement est enfin cruciale et propre à la notion de Big Data. La véracité qui suppose à vérifier systématiquement la validité des données. Ces dimensions supposent que l’on soit capable de stocker et d’exploiter des volumes énormes de données, d’accéder à de multiples sources par un réseau performant, de traiter dans des temps toujours plus courts des volumes considérables de données.

Face à cette situation, le groupe Atos leader international dans le secteur informatique, veut mettre en place un **portail de rapport Big Data** au niveau de ses puissants serveurs afin d’améliorer de façon significative le processus de prise de décision et de bien exploiter ses données qui représentent une source précieuse d’informations. Par ailleurs avec toute cette masse de données on évoque de nouvelles possibilités d’exploration pour pouvoir en apprendre un peu plus sur ces données. C’est dans ce cadre que le groupe Atos a proposé un projet de mettre en place un **portail de rapport Big Data**.

## Problématique

Aujourd’hui, ce sont plus de 2,5 quintillions octets, soit 1030 données qui sont créés par jour. L’hyper-connectivité des individus, la multiplication des terminaux (ordinateurs, téléphones, tablettes), et des objets connectés, ont fait se multiplier les données transitant sur le réseau. Ainsi, on estime que le nombre de données produites par les internautes doublerait tous les 18 à 24 mois. Et la tendance ne risque pas de s’atténuer car avec les objets connectés, c’est encore plus de données qui vont être disponibles. Demain, nous posséderons en moyenne 8 objets connectés à titre personnel.

Face à cette croissance exponentielle du volume des données et la diversité de leurs formats, les organisations sont confrontées à des problèmes tels que la collecte, le stockage, l’analyse et l’exploitation de grands volumes de données pour créer de la valeur ajoutée. D’ailleurs, les bases de données traditionnelles ne suffisent plus pour gérer les données. Il faut imaginer de nouvelles architectures, de nouvelles technologies et des nouveaux algorithmes pour stocker, traiter et transporter ces masses considérables de données.

Mais comment gérer des milliers ou des millions de données ? D'abord il faut centraliser toutes ces données dans un ou plusieurs serveurs.

Donc, tout l’enjeu pour les entreprises et les administrations du XXIème siècle consiste à stocker à long terme ces volumes de données ainsi de les traiter et de les analyser pour ne pas passer à côté d’informations précieuses cachées dans la masse.

C’est dans ce cadre que le Groupe Atos leader international dans le secteur informatique, veut étendre son expertise en service Big Data, en fournissant les bénéfices d’un portail de rapport Big Data. Les besoins exprimés et les contraintes à respecter en vue de mettre en place un tel système conduit à répondre à la problématique suivante :

Comment centraliser et fournir un accès facile à ces données pour en tirer profit ?

## Objectifs

### Objectif général

L'objectif principal de cet étude est de mettre en place un outil permettant de pouvoir collecter ces données, les traitées et les observées via une plateforme web de visualisation. Cet objectif permettra la simplification d’étude et aidera à prendre des décisions.

### Objectifs spécifiques

Pour atteindre cet objectif général décrit plus haut, nous allons de façon plus concrète en définir de plus spécifiques qui nous permettront d'atteindre le premier nommé.

IL s’agit de:

* Centraliser la collecte de toutes les données dans un Data Lake;
* Fournir un accès facile aux données en utilisant le graphique
* Exécuter des recherches rapides et avancées
* Visualiser les données d’un data Lake (Lac de données) en utilisant des modules de visualisation
* Définir une utilisation interactive pour la manipulation des données et la visualisation des données
* Créer des Tableaux de bords.

## Conclusion

Ce premier chapitre, nous a permis de présenter la structure d’accueil, de contextualiser le sujet et de dégager la problématique ainsi que les objectifs à atteindre.

# CHAPITRE II : Étude de l’existant

## Définitions des concepts du domaine

## Fonctionnalités existantes

### Les acteurs principaux

### Les modules existants

## Architecture et Technologies utilisées

### Architecture technique

### Architecture applicative

### Technologies utilisées

### Les évolutions souhaitées

# CHAPITRE III : Choix d’une méthode d’analyse et de conception

La réalisation d'un projet informatique doit être impérativement précédée d'une méthodologie d'analyse et de conception qui a pour objectif de permettre de formaliser les étapes préliminaires du développement d'un site afin de rendre ce développement plus fidèle aux besoins du client. La phase d'analyse permet de lister les résultats attendus, en termes de fonctionnalité et la phase de conception permet de décrire de manière non ambiguë, le plus souvent en utilisant un langage de modélisation, le fonctionnement futur du système, afin d'en faciliter la réalisation.

Dans cette partie il sera question de parler des processus de développement et présenter le processus que nous avons adopté ainsi que son application dans notre projet. La deuxième partie sera consacrée à la présentation du langage de modélisation UML.

### Processus de développement

Un processus de développement définit un ensemble d’activités et leur enchaînement. Une activité comprend des tâches, des contraintes, des ressources et une démarche. Pour satisfaire le client, ces trois paramètres doivent être pris en compte.

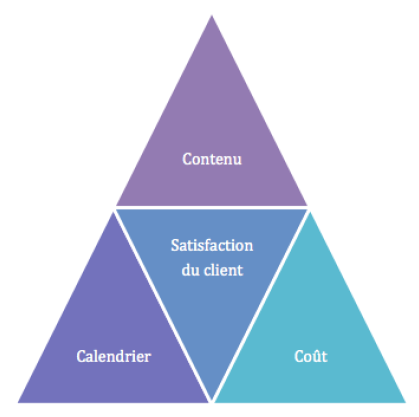
****

Figure X

Il existe deux grandes familles de méthodes de gestions de projet que sont les méthodes classiques et les méthodes agiles. Le principal défaut des méthodes classiques est effet tunnel qui peut être très néfaste et conflictuel, on constate souvent un déphasage entre le besoin initial et l’application réalisée. On se rapporte alors aux spécifications validées et au contrat. Les méthodes agiles quant à elles, se présentent comme une alternative qui se propose de réduire considérablement voir complètement cet effet tunnel en impliquant le client dans le projet.

Avant de passer à la présentation de quelques-uns de ces méthodes, nous allons d’abord faire une description de quelques concepts dans un projet.

#### Définition des concepts

* **Projet**

Il existe de nombreuses tentatives de normalisation de la notion de projet, donnant lieu à beaucoup de définitions relativement proches. Parmi celles-ci, nous avons celle proposée par la norme ISO 10006 selon laquelle: 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 telles que des contraintes de délais, de coûts et de ressources.

* **Cycle de vies d’un projet**

On appelle « cycle de vie du projet » l'enchaînement dans le temps des étapes entre l'émergence du besoin et la livraison du produit.

* **Gestion d’un projet**

La gestion de projet est la mise en œuvre de connaissances, de compétences, d’outils et de techniques appliqués au projet afin d’en respecter les exigences, vis-à-vis du client et de sa propre hiérarchie.

* **Analyse**

Dans le cadre de projets informatiques, la phase d’analyse est une étape clé qui consiste à caractériser les besoins du futur produit, de lister les résultats attendus en terme de fonctionnalités, de performance, de maintenance, de robustesse, de sécurité etc. L’analyse permet donc d’avoir une vision claire et précise du problème posé et du système à réaliser. Il est donc important, durant cette phase de ne pas perdre de vue les besoins des utilisateurs ainsi que les frontières du système à bâtir.

* **Conception**

La conception, quant à elle, consiste à décrire de manière très claire, grâce à un langage de modélisation, le futur système, afin d’en faciliter la réalisation. Etant la phase qui suit l’étape de l’analyse, la conception doit répondre à la question du « Comment ? » en mettant l’accent sur une solution conceptuelle qui répond aux exigences plutôt que sur une implémentation. Ainsi, dans la phase de conception, nous allons procéder aux choix de l’architecture du système et des outils à utiliser. De ce fait, avec les modèles obtenus au sortir de l’étape de conception, seront assez détaillés pour passer à l’étape d’implémentation.

#### Principales méthodologies

* **Les méthodes classiques**

Ces méthodes sont basées sur des activités séquentielles : recueil des besoins, définition du produit, développement puis test avant livraison au client. Elles sont utilisées pour gérer les projets depuis plusieurs années. Elles se basent sur le principe que tout doit être planifié au début et que tout doit être prévisible. Nous présentons ci-dessous quelques-unes des méthodes classiques.

1. Modèle en cascade

Modèle apparu en 1966, il est issu de la construction du bâtiment où la fondation doit être faite avant le reste de la maison. Les tâches s’exécutent d’une manière totalement successive. Dans son modèle de base, il se basait sur le principe du non-retour c’est-à-dire qu’il n’y a pas de remise en cause d’une étape précédente. Mais dans sa version actuelle, la fin de chaque phase a lieu à une date prédéterminée et s’achève par la production d’un certain nombre de documents ou logiciels. Ces livrables sont produits pour réaliser la Vérification/Validation qui permet d’approuver le passage à la phase suivante ou la correction de l’étape précédente. Il est très adapté aux petits projets simples avec peu d’incertitude et où les besoins sont clairement identifiés et stables.

**Figure : Modèle en cascade**

1. Cycle en V

Issu du monde de l'industrie, le cycle en V est devenu un standard de l'industrie logicielle depuis les années 1980. C’est une méthode qui découle de l’amélioration du modèle en cascade où chaque phase du projet a une phase de test qui lui est associée. Cette association découle de l’idée selon laquelle chaque phase en amont du développement doit préparer une phase correspondante de vérification en aval de la production du logiciel. Cette méthode permet de limiter un retour aux étapes précédentes.

**Figure : Cycle en V**

1. Cycle en spirale

C’est une méthode qui reprend les différentes étapes du cycle en V en proposant un produit de plus en plus complet et de plus en plus robuste grâce à des implémentations de versions successives. Toutefois, Cette méthode met plus l’accent sur la gestion des risques que le cycle en V. Ainsi, le début de chaque itération comprend une phase d’analyse des risques qui est rendue nécessaire par le fait que, lors d’un développement cyclique, il y’a plus de risques de défaire, au cours de l’itération, ce qui a été fait au cours de l’itération précédente.

**Figure : Cycle en spirale**

1. Limites des méthodes classiques

Conçues pour piloter un projet par l’élaboration de plans, les méthodes classiques poussent les acteurs d’un projet à refuser systématiquement tout changement, que ce soit des changements dans le contenu ou le périmètre du projet, dans le processus de développement, au sein de l’équipe mais également à toute modification des plans initiaux, auxquels on doit rester conforme. Par conséquent ces méthodes, dites classiques, présentent les inconvénients suivants :

* **La rigidité de l’approche** On déplore que la nouveauté, la marge de manœuvre laissée, à juste titre, au client pour préciser ou faire évoluer ses attentes, la non-prévisibilité de tous les événements soient difficilement compatibles avec une approche prédictive comme celle du cycle en cascade
* **L’effet tunnel** qui est dû au fait que le client n’intervient que lors de l’expression des besoins et voit le résultat quelque temps après. Ce qui ne favorise pas la collaboration entre les utilisateurs et les informaticiens surtout si le résultat ne correspond pas aux attentes.
* **Une mauvaise communication** qui se manifeste par la prohibition par l’absence de jalons intermédiaires de la validation de ce que sera la version finale du projet. Ainsi les surprises en fin de cycle et le refus de changement par les équipes de développement pénalisent la qualité des relations avec les utilisateurs.
* **La levée tardive des facteurs à risques** due au fait que les tests de performance et d’intégration sont reportés après le développement.
* **Une documentation pléthorique** qui permet de repousser la phase de codage qui est, toutefois, irréversible. Elle permet également de s’opposer au changement en brandissant un document contractuel validé précédemment.
* **Les méthodes agiles**

Une méthode agile est une approche itérative et incrémentale qui est menée dans un esprit collaboratif, avec juste ce qu’il faut de formalisme. Elle génère un produit de haute qualité tout en prenant en compte l’évolution des besoins des clients.

L’approche itérative consiste à découper le projet en plusieurs étapes d’une durée de quelques semaines ; ce sont les itérations. Au cours d’une itération, une version minimale du produit attendu est développée puis soumise, dans sa version intermédiaire, au client pour validation. Les fonctionnalités sont ainsi intégrées au fur et à mesure du cycle de vie sur un mode incrémental, le système s’enrichissant progressivement pour atteindre les niveaux de satisfaction et de qualité requis.

Chaque itération est un mini-projet en soi qui comporte toutes les activités de développement, menées en parallèle: analyse, conception, codage et test, sans oublier les activités de gestion de projet. L’objectif est d’obtenir, au terme de chaque itération, un sous ensemble opérationnel du système cible et, au terme de la dernière itération, la version finale du produit.

Nous allons présenter ci-dessous trois méthodologies agiles. La première, **Scrum** qui est la méthodologie la plus populaire. La méthodologie **eXtreme Programming (XP)** qui est souvent pratiquée conjointement à **Scrum**. Et enfin la méthodologie **RUP (Rational Unified**

**Process)**. Il existe un grand nombre d’autres méthodologies agiles qui ne seront pas traitées ici car l’objectif n’est pas de présenter de façon exhaustive les méthodologies agiles mais plutôt de mettre en évidence celles qui sont les mieux adaptées ou les plus utilisées dans le développement logiciel. Toutefois, les méthodes suivantes méritent d’être citées : **Rapid Application Development (RAD)** qui est un précurseur des méthodologies agiles**, Dynamic Software Development Method (DSDM)**, **Feature-Driven Development (FDD)**, **Crystal** ou encore **Adaptive Software Development (ASD).**

1. eXtrem Programming(XP)

XP ou eXtreme Programming est une méthode de développement agile, orientée projet informatique dont les ressources sont régulièrement actualisées. C’est une méthode de management de projet destinée à accélérer drastiquement la réalisation des projets de type flexible. Une règle fondamentale de la méthode est le fait que le client ou un représentant avisé participe au développement. Le but principal de XP est de réduire les coûts du changement. XP s'attache à rendre le projet plus flexible et ouvert au changement en introduisant des valeurs de base, des principes et des pratiques.

Les principes de cette méthode ne sont pas nouveaux : ils existent dans l'industrie du logiciel depuis des dizaines d'années et dans les méthodes de management depuis encore plus longtemps. L'originalité de la méthode est de les pousser à l'extrême :

* la revue de code sera faite en permanence (par un binôme),
* les tests seront faits systématiquement avant chaque mise en oeuvre,
* la conception sera faite tout au long du projet (refactoring),
* la solution la plus simple sera toujours choisie,
* les métaphores seront définies,
* l’intégration des modifications se fera plusieurs fois par jour,
* des cycles de développement très rapides se feront pour s’adapter au changement.

L'Extreme Programming repose sur des cycles rapides de développement (des itérations de quelques semaines) dont les étapes sont les suivantes :

* Une phase d'exploration détermine les scénarios qui seront fournis par le client pendant cette itération,
* l'équipe transforme les scénarios en tâches à réaliser et en tests fonctionnels,
* chaque développeur s'attribue des tâches et les réalise avec un binôme,
* lorsque tous les tests fonctionnels passent, le produit est livré

Le cycle se répète tant que le client peut fournir des scénarios à livrer. Généralement le cycle de la première livraison se caractérise par sa durée et le volume important de fonctionnalités embarquées. Après la première mise en production, les itérations peuvent devenir plus courtes.

**Figure : Cycle de vie d’un projet XP**

1. Scrum

Scrum tient son origine du terme sportif de rugby signifiant : mêlée. La méthodologie à ses acteurs d'être soudés dans l'accomplissement d'un projet, dans l'atteinte d'un but. Elle utilise une procédure dite itérative (une itération est appelée sprint). Chaque itération ou sprint fournit une partie fonctionnelle du produit. C’est une méthodologie qui est principalement basée sur la gestion des ressources humaines et peut être adaptée à d’autres contextes où il existe un besoin de travailler en équipe. Scrum peut donc être vu comme le cadre de développement venant par exemple appuyer des pratiques XP.

Scrum suppose donc une intense collaboration entre les différentes personnes impliquées. Le directeur de produit **(product owner**) est le représentant du client, il définit les priorités pour la réalisation. Le gestionnaire de projet est nommé **ScrumMaster** : il a pour rôle de faciliter l'application de Scrum par l'équipe.

Le processus Scrum repose sur deux journaux ou "backlog" :

* **Product Backlog:** liste regroupant les exigences du client. Ce référentiel évolue au cours du développement pour prendre en charge au mieux les besoins du client.
* **Sprint Backlog**: recense les tâches du Sprint en cours.

Un projet utilisant Scrum a son cycle de vie composé de Sprints successifs. Un Sprint dure au plus quatre semaines. Pendant un Sprint, des réunions quotidiennes de moins de 15 minutes (appelées Scrum) permettent à toute l'équipe de faire le point sur le travail accompli par chacun depuis la dernière réunion Scrum, les obstacles rencontrés, et le travail prévu d'ici la prochaine réunion.

Pendant un Sprint l'équipe développe un produit partiel. Elle déroule toutes les activités nécessaires pour cela : analyser, concevoir, développer, tester, documenter, intégrer. Chaque Sprint se termine par une revue de Sprint, pour que le directeur de produit évalue, au cours d'une démonstration, le produit partiel obtenu et modifie au besoin le backlog de produit.

En définitive, Scrum introduit des règles pour suivre un processus itératif empirique permettant d'obtenir un produit très proche des besoins qui évoluent et ainsi de maximiser la valeur pour les clients.

**Figure 10: Fonctionnement de Scrum**

1. Rational Unified Process (RUP)

Cette méthode qui peut être considérée comme la moins agile des méthodes présentées ici, est un mélange des pratiques issues des méthodes traditionnelles et des méthodes agiles. Le principe est de parcourir un cycle de vie (inspection, élaboration, construction, transition) durant une itération. Chaque phase du cycle de vie est très précisément détaillée. Son approche assez lourde et le coût d’investissement de cette méthode la réserve à des projets de grande ou moyenne taille.

### Les rôles

Dans *SCRUM,* on distingue les différents rôles suivant :

* **Le Product Owner**

Il est responsable de la définition du contenu du produit et de la gestion des priorités pour son développement. Il représente le client final et communique la vision globale du produit à l’équipe.

* **Le Scrum Master**

Il agit en tant que facilitateur entre le Product Owner et les membres de l’équipe. Il doit identifier les obstacles mis en évidence par la méthode et les éliminer.

* **L’Equipe**

Elle est responsable de la réalisation opérationnelle en développant un incrément à chaque sprint. Elle est auto-organisée et doit avoir les compétences nécessaires pour accomplir le travail.

### SCRUM dans notre projet

Dans le cadre de notre projet, on a adopté la méthode *SCRUM* dans la mesure où l’on devait rendre régulièrement des livrables afin que les maîtres les valident pour pouvoir passer à la tâche suivante.

* Répartition de rôle

Les rôles définis dans *SCRUM* ont été attribués aux personnes engagées dans le projet en tenant compte de leur spécialisation. Par conséquent, le Rôle de « *PRODUCT OWNER* » a été attribué à ABDOU KHADIR GAYE.

Les rôles « *SCRUM MASTER* » et « EQUIPE » ont été attribués comme suit :

* *SCRUM MASTER* : Abdou Khadir GAYE
* L’EQUIPE DE DEVELLOPEMENT: Abdoulaye Ndigue SENE

Des réunions ont été organisées toutes les deux ou trois semaines pour voir l’état d’avancement du projet et pour redéfinir les sprints (durée, tâches, etc.)

### Présentation et justification du choix d'UML

Dans cette section, nous présentons l’historique d’UML ainsi que ces différents types de diagramme.

#### Définition

#### Historique

#### À quoi sert UML ?

UML donne une définition sur une approche objet plus formelle et apporte la dimension logique à l'approche objet.

Pour concevoir en UML, il faut commencer par prendre de la hauteur par rapport au problème qui est posé et utiliser des concepts abstraits complètement indépendants des langages de programmation. L'utilisation d'un langage de programmation comme support de conception revient à faire une analyse peu précise et réductrice par rapport à une modélisation objet. A l'inverse de la plupart des technologies objet, UML permet de s'affranchir totalement de tout langage de programmation (permettant ainsi l'écueil de la limitation de vue du langage de programmation) pour élaborer et exprimer des modèles objet. Il a été pensé comme support d'analyse objet.

De plus, UML est un méta-modèle : il décrit très précisément tous les éléments de modélisation (permettant ainsi de limiter les ambiguïtés) et normalise les concepts objet. Étant un méta modèle, UML est valable pour tous les langages de programmation.

Il présente comme qualités d'être un langage sans ambigüité, un langage universel, un langage offrant le moyen de définir les structures de programmation, un langage ayant une représentation universelle (une communication performante et une notation graphique simple.

UML propose aussi une notation pour représenter graphiquement les éléments de modélisation du méta modèle ; cette notation graphique est le support du langage UML. Ceci lui permet donc d'être visuellement plus compréhensible (pour comparer ou évaluer) et limite ainsi les ambiguïtés. Il offre un cadre avec différentes vues complémentaires du système sur plusieurs niveaux d'abstraction, les diagrammes, et contrôle ainsi que la complexité dans l'expression des solutions objets.

#### Que ce qu'un diagramme UML ?

#### Avantages et inconvénients d'UML

* Points forts
* Points faible

#### Les diagrammes

UML 2 s’articule autour de treize types de diagrammes, chacun d’eux étant dédié à la représentation des concepts particuliers d’un système logiciel. Ces types de diagrammes sont répartis en deux grands groupes :

* Six diagrammes structurels que sont : Diagramme de Diagramme d’objets, Diagramme de paquetages, Diagramme de structure composite, Diagramme de composants, Diagramme de déploiement.
* Sept diagrammes comportementaux : Diagramme de cas d’utilisation, Diagramme de vue d’ensemble des interactions, Diagramme de séquence, Diagramme de communication, Diagramme de temps, Diagramme d’activité et les Diagrammes d’états.

#### Planning

Ayant adopté *SCRUM* dans le cadre de ce projet, nous avons utilisé le logiciel GantProject pour établir notre diagramme deGant. Le diagramme de Gant, est un outil qui permet de représenter visuellement l’état d’avancement des différentes activités (tâches) qui constituent un projet.

La *Figure X* ci-dessous montre la planification des différentes tâches du projet *portail de rapport Big Data*  a effectuées ainsi qu’une estimation de leur durée de réalisation.

## Conclusion

Dans ce chapitre, nous avons abordé les différentes notions importantes utilisées tout au long du document plus précisément dans l’architecture. On a présenté le ***portail*** ainsi que les diverses technologies en rapport avec ce dernier pour sa mise en œuvre. Le chapitre suivant sera consacré à la spécification fonctionnelle du système à développer.

# CHAPITRE IV : Spécification fonctionnelle détaillées

## Généralités

Dans cette partie, nous présentons les fonctionnalités du système, leur cas d’utilisation ainsi que les règles métiers liées à ces derniers.

### Règle générale

Cette section présente l’ensemble des règles valables pour toutes les fonctionnalités. Cela évite de répéter chacune de ces règles au niveau de chaque cas d’utilisation. Les règles métiers propres à chaque cas (lorsqu’il y en a) seront précisées lors de la description de ce cas.

### Règles liées à la validation des formulaires

#### Champs obligatoire

Lorsque l’utilisateur valide un formulaire sans renseigner un ou plusieurs champs obligatoires :

* Un message d’erreur apparait : « Veuillez renseigner les champs obligatoires »,
* Les champs obligatoires non renseignés sont mis en évidence,
* Les champs saisis ne sont pas réinitialisés (sauf mention contraire).

#### Champs uniques

Lorsque l’utilisateur valide un formulaire alors qu’une règle d‘unicité est violée :

* Un message d’erreur apparait : « Le champ « X » doit être unique ! »,« X » étant le nom du champ en erreur,
* Le champ en erreur est mis en évidence,
* Les champs saisis ne sont pas réinitialisés (sauf mention contraire).

#### Format des champs

Les zones de saisie doivent empêcher la saisie de valeurs au format incorrect (contrôles en Front-end).

#### Violation des règles métiers

Lorsque l’utilisateur valide un formulaire alors qu’une règle métier est violée (contrôles en Back-end) :

* Un message d’erreur apparait indiquant la règle métier violée,
* Les champs saisis ne sont pas réinitialisés (sauf mention contraire).

### Règles liées aux actions sur les objets

#### Règles sur l’ajout

Lorsque l’opération d’ajout réussit :

* Un message de succès apparait afin d’en informer l’utilisateur
* L’écran d’ajout est réinitialisé

De même, lorsque l’opération d’ajout échoue :

* Un message d’erreur apparait afin d’en informer l’utilisateur et contient une description précise de la (es) cause(s) de cet échec
* L’écran d’ajout n’est pas réinitialisé

#### Règles sur la recherche

Sauf mention contraire, les critères de recherche peuvent ne pas être renseignés et dans ce cas la liste de tous les objets est retournée, De même, lorsque le(s) critère(s) est (sont) renseigné(s), alors la recherche doit retourner uniquement les objets correspondant à ce(s) critère(s).

Lorsqu’aucun résultat n’est retourné :

* Le tableau résultat de la recherche est tout de même affiché
* Le message suivant apparait dans ce tableau : « Aucun résultat trouvé »

#### Règles sur la modification

Lorsque l’utilisateur tente de modifier un objet qui n’existe pas (notamment si la liste affichée des objets n’est pas actualisée), le message d’erreur suivant doit être affiché :

« L’opération ne peut être effectuée, car cet élément n’existe pas ! ».

#### Règles sur la suppression

Toute opération de suppression doit être précédée de l’affichage d’un message de confirmation avant application. Ce message sera libellé ainsi : « Êtes-vous sûr de vouloir effectuer cette suppression ? ». Lorsque l’utilisateur tente de supprimer un objet qui n’existe pas (notamment si la liste affichée des objets n’est pas actualisée), le message d’erreur suivant doit être affiché : « L’opération ne peut être effectuée, car cet élément n’existe pas ! »,

La suppression en cascade n’est pas autorisée, ainsi lorsqu’un utilisateur tente de supprimer un objet lié à un ou plusieurs objets, le message d’erreur suivant doit être affiché : « L’opération ne peut être effectuée, car cet élément est lié à d’autres objets ! ».

**NB** : Mais s’il s’agit d’un « index » contenant un ensemble de données, celui-ci sera supprimé.

## Fonctionnalités de l’application

Cette section présente l’ensemble des fonctionnalités du portail « Rapport Big Data» sur la base du cahier des charges. Pour chacune des fonctionnalités, il sera listé :

* Les cas d’utilisations liées à cette fonctionnalité
  + Les cas d’utilisation permettent d’exprimer le besoin des utilisateurs d’un système
* La description des objets (attributs et leurs contraintes)
* Les règles métiers lorsqu’il y en a

### Fonctionnalité « Gestion des visualisations »

* Description

Cette fonctionnalité permet à un utilisateur habilité de pouvoir interagir avec la base de faire différents types de visualisations, il pourra ainsi : ajouter une visualisation, rechercher une visualisation, modifier une visualisation, supprimer une visualisation, exporter visualisation (exporter en format PDF ou image) et obtenir les détails sur une visualisation. Une visualisation est définie par les champs suivants :

|  |  |  |  |
| --- | --- | --- | --- |
| Champ | Type | Obligatoire | Contrainte |
| description | text | Non |  |
| title | text | Oui |  |
| visState | text | Oui |  |
| uiStateJSON | text | Non |  |

#### Cas d’utilisation « ajouter visualisation »

Ce cas d’utilisation permet à un utilisateur habilité d’ajouter une visualisation dans le système. Les règles métiers à respecter sont :

* L’identifiant doit être unique

#### Cas d’utilisation « supprimer visualisation »

Ce cas d’utilisation permet à un utilisateur habilité de supprimer une visualisation dans le système.

#### Cas d’utilisation «modifier visualisation »

Ce cas d’utilisation permet à un utilisateur habilité de modifier une visualisation dans le système. Lors de la modification, les contraintes suivantes sont à respecter :

#### Cas d’utilisation «rechercher visualisation »

Ce cas d’utilisation permet à un utilisateur habilité de rechercher une visualisation dans le système. Les critères de recherche sont les suivants : titre de la visualisation. Le résultat de la recherche doit présenter un tableau avec les colonnes suivantes : titre, type de l’index, un lien détails pour accéder directement à la visualisation.

#### Cas d’utilisation «exporter visualisation »

Ce cas permet à un utilisateur habilité d’exporter une visualisation en un format PDF ou image.

### Fonctionnalité « Gestion des tableaux de bord »

* Description

Cette fonctionnalité permet à un utilisateur habilité de pouvoir interagir avec la base pour pouvoir créer des tableaux de bord à partir des visualisations créées (Un ensemble de visualisations regroupées forme un tableau de bord), il pourra ainsi : ajouter un tableau de bord, rechercher un tableau de bord, modifier un tableau de bord, supprimer un tableau de bord, exporter tableau de bord en un format PDF ou image et obtenir les détails sur un tableau de bord. Un tableau de bord est défini par les champs suivants :

|  |  |  |  |
| --- | --- | --- | --- |
| Champ | Type | Obligatoire | Contrainte |
| description | text | Non |  |
| panelsJSON | text | Oui |  |
| title | text | Oui |  |
| optionsJSON | text | Non |  |

#### Cas d’utilisation « ajouter tableau de bord »

Ce cas d’utilisation permet à un utilisateur habilité d’ajouter un tableau de bord dans le système à partir des visualisations. Les règles métiers à respecter sont :

* L’identifiant doit être unique

#### Cas d’utilisation « supprimer tableau de bord »

Ce cas d’utilisation permet à un utilisateur habilité de supprimer une visualisation dans le système.

NB : La suppression d’un tableau de bord n’affecte pas la suppression d’une visualisation mais la suppression d’une visualisation affectera le tableau de bord.

#### Cas d’utilisation «modifier tableau de bord »

Ce cas d’utilisation permet à un utilisateur habilité de modifier un tableau de bord dans le système.

NB : La modification d’un tableau de bord affecte juste la disposition des différentes visualisations pas la modification des visualisations.

#### Cas d’utilisation «rechercher tableau de bord  »

Ce cas d’utilisation permet à un utilisateur habilité de rechercher un tableau de bord dans le système. Les critères de recherche sont les suivants : titre du tableau de bord. Le résultat de la recherche doit présenter un tableau avec les colonnes suivantes : titre, description, un lien détails pour accéder directement au tableau de bord.

#### Cas d’utilisation «exporter tableau de bord  »

Ce cas permet à un utilisateur habilité d’exporter un tableau de bord en un format PDF ou image.

### Fonctionnalité « Exploration de données »

* Description

Cette fonctionnalité permet à un utilisateur de partir à la découverte de nos données, de réaliser des recherches, de filtrer et de visualiser les documents résultats. C'est à partir de cette fonctionnalité que vous allez réaliser vos requêtes préliminaires pour développer une intuition de phénomènes intéressants à observer.

Ces résultats ainsi fournies pourront être sauvegardé dans la base puis après les utiliser dans une visualisation.

#### Cas d’utilisation « Choisir type d'exploration »

Ce cas d’utilisation permet à un utilisateur habilité de choisir un type d’exploration des données du système. On a deux types d’explorations de données :

#### Cas d’utilisation « exploration par metrics »

#### Cas d’utilisation « exploration par buckets »

#### Cas d’utilisation « ajouter filtre »

#### Cas d’utilisation « supprimer filtre»

Ce cas nous permet de filtrer nos recherches en fonction d’un critère bien défini.

### Fonctionnalité « Gestion requête personnalisée»

* Description

Personnalisé sa requête s'avère souvent utile lorsque vous savez exactement de quelles informations vous avez besoin et comment formuler des requêtes.

Cette fonctionnalité permet à un utilisateur de pouvoir personnaliser ses requêtes afin de visualiser le résultat fournis.

#### Cas d’utilisation « Faire une requête personnalisée »

Ce cas d’utilisation permet d’interagir directement avec notre base à travers des requête personnalisées

### Fonctionnalité « Gestion de monitoring »

* Description

Cette fonctionnalité permet à un utilisateur habilité d’effectuer la surveillance et la mesure de performance du système.

En effet, plusieurs processus liés au système (dont le temps d’exécution des requêtes, état de santé du cluster, etc.) seront lancés et il faudra alors surveiller leur état d’exécution et leur temps de réponse.

#### Cas d’utilisation « Consulter état cluster »

### Fonctionnalité « Gestion Utilisateur »

* Description

Cette fonctionnalité permet à un utilisateur habilité de paramétrer les utilisateurs au sein du système, il pourra ainsi : ajouter un utilisateur, rechercher un utilisateur, modifier un utilisateur et supprimer un utilisateur. Un utilisateur est défini par les attributs suivants. Le *Tableau X* suivant donne les caractéristiques d’un utilisateur :

|  |  |  |  |
| --- | --- | --- | --- |
| Attributs | Type | Obligatoire | Contraintes |
| Email | text | Oui | Alphanumérique |
| enabled | boolean | Oui | Correspond à savoir si l’utilisateur est actif ou non |
| mot\_de\_passe | text | Oui | Alphanumérique |
| role | text | Oui | Correspond à savoir quel type d’utilisateur on a |
| otherOptions | text | Non | D’autres informations non importantes relatives à l’utilisateur |
| nom | text | Oui | Alphanumérique |
| prenom | text | Oui | Alphanumérique |
| date\_creation | date | Oui | Correspond à la date de la création de l’utilisateur |

#### Cas d’utilisation « ajouter utilisateur»

Ce cas d’utilisation permet à un utilisateur habilité d’ajouter un compte dans le système. Lors de l’ajout de ce compte :

* L’email fournit est unique
* Le mot de passe sera généré et ainsi les informations seront envoyées par mails
* Un compte est associé à un et un seul utilisateur
* Le système doit prévoir un utilisateur pré configuré « Super Admin »

#### Cas d’utilisation « modifier utilisateur»

Ce cas d’utilisation permet à un utilisateur habilité de modifier un utilisateur dans le système. Les règles métiers pour ce cas sont :

* L’utilisateur « Super Admin » ne peut pas être modifié.
* Un seul utilisateur tout au plus peut posséder le profil « Profil Super Admin ».

#### Cas d’utilisation « supprimer utilisateur»

Ce cas d’utilisation permet à un utilisateur habilité de supprimer un utilisateur dans le système. Les contraintes de suppression sont :

* Tous les utilisateurs peuvent être supprimés à l’exception de l’utilisateur de base « Super Admin ».

#### Cas d’utilisation «rechercher utilisateur»

Ce cas d’utilisation permet à un utilisateur habilité de rechercher un utilisateur dans le système. Les critères de recherche sont les suivants : prénom(s), nom, email.

### Fonctionnalité « Gestion des Index »

* Description

#### Cas d’utilisation « consulter infos index »

Ce cas d’utilisation permet de pouvoir consulter les informations relatives à cet index à savoir le nombre de document, sa taille en stockage, etc.

#### Cas d’utilisation « définir index par défaut »

Ce cas d’utilisation permet de définir un index par défaut pour l’exploitation des données de celui-ci, on n’aura pas besoin tout le temps de choisir l’index pour exploiter ses données.

#### Cas d’utilisation «supprimer index »

Ce cas d’utilisation permet supprimer un index de la base, sa suppression entrainera :

* La suppression de toutes les données qui s’y trouve
* La suppression de toutes les visualisations dérivées de cet index

## Conclusion

Dans ce chapitre, nous avons décrit le processus de développement adopté dans le cadre de ce projet ainsi que la manière dont on l’a appliqué à notre projet. Ensuite on a vu les différentes fonctionnalités de notre portail, leur cas d’utilisation et les contraintes à respecter. Dans le chapitre suivant, nous présentons la conception du système *portail de rapport big data* en détaille tout en proposant différentes types d’architecture

# CHAPITRE V : Conception du système

## Spécification fonctionnelle détaillée

### Catalogue

Les acteurs suivants sont les utilisateurs du système *« Portail de rapport Big Data »*.

|  |  |  |
| --- | --- | --- |
| Id | Nom | Use case (Acteur Principal) |
| A\_01 | Admin. Du système | Tous |
| A\_02 | Utilisateur | Use case assignés |

### Définition de l’acteur

Ce tableau ci-dessus décrit les acteurs du système et leurs rôles.

|  |  |  |
| --- | --- | --- |
| Acteur | Description | Rôles |
| A\_01 | L’admin du système est l’utilisateur qui a tous les droits du système | Tous les niveaux du système |
| A\_02 | Un utilisateur habilité est un utilisateur ayant les droits nécessaires pour effectuer une tâche particulière. | Son rôle principal est d’accomplir les tâches qui lui sont assignées. |

### Étude préliminaire

Les diagrammes de cas d’utilisation permettent d’exprimer les besoins des utilisateurs du système, ils sont orientés utilisateur de ce besoin. Nous allons scinder notre système en diagramme de packages et chaque diagramme de package est vu comme un « sous-système » de notre plateforme.

Les différents diagrammes de packages représentent les fonctionnalités définies dans l’étude conceptuelle. Dans ce chapitre, vu que nous utilisons la méthodologie *SCRUM*, nous allons réaliser les diagrammes de packages des fonctionnalités suivantes: Gérer des utilisateurs, Gérer Index, Gérer des visualisations, Gérer des tableaux de bord, Exploration de données, Gérer de monitoring, Gérer requête personnalisée.

Le *Figure 13* ci-dessous montre le diagramme de package du système.

Ici on met le diagramme de package du système

Vu que nous avons identifié les différents acteurs du système et définit les différents packages, dans les sections suivantes nous allons établir les diagrammes de cas d’utilisation de chaque sous-système (package), la description et le diagramme de séquence de certains cas.

### Diagramme de cas d’utilisation et leur descriptions

La représentation du diagramme de cas d’utilisation de chaque package sera suivie d’une description textuelle des différents cas d’utilisation de cette dernière.

#### Le sous-système « gestion des utilisateurs »

Diagramme de use case de gestion des utilisateurs

* Cas d’utilisation « nom du cas a définir »
  + Description textuelle
  + Diagramme de séquence

#### Le sous-système « gestion des index »

Diagramme de use case de gestion des index

* Cas d’utilisation « nom du cas à définir »
  + Description textuelle
  + Diagramme de séquence

#### Le sous-système « gestion des visualisations »

Diagramme de use de gestion des visualisations

* Cas d’utilisation « nom du cas à définir »
  + Description textuelle
  + Diagramme de séquence

#### Le sous-système « gestion des tableaux de bords »

Diagramme de use case de gestion des index

* Cas d’utilisation « nom du cas a définir »
  + Description textuelle
  + Diagramme de séquence

#### Le sous-système « exploration de données »

Diagramme de use case de gestion des index

* Cas d’utilisation « nom du cas a définir »
  + Description textuelle
  + Diagramme de séquence

#### Le sous-système « Gérer requête personnalisée »

Diagramme de use case de Gérer requête personnalisée

* Cas d’utilisation « nom du cas a définir »
  + Description textuelle
  + Diagramme de séquence

#### Le sous-système « Gérer le monitoring »

Diagramme de use case de Gérer le monitoring

* Cas d’utilisation « nom du cas a définir »
  + Description textuelle
  + Diagramme de séquence

### Schéma de la base de données

## Conception détaillée

## Architecture générale du système

## Architecture technico-fonctionnelles

## Conclusion

# CHAPITRE VI : Implémentation de la solution

## Choix des outils et des technologies

## Aperçu des résultats

## Conclusion

# Annexes