République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abderrahmane Mira de Béjaïa

Faculté des Sciences Exactes
Département d'Informatique

Mémoire de Master Recherche

Option : Système d'information avancé

Thème

Introduction de l'agilité dans le processus de développement de systèmes intelligents

Présenté par :

LAGAB Sara & GALI Kinza

Soutenu devant le jury composé de :

Président	Mme. BOUADEM Nassima	M.C.B	U. A/Mira Béjaïa
Examinateur	M. MOHAMMEDI Mohamed	M.C.A	U. A/Mira Béjaïa
Encadrant	M. ACHROUFENE Achour	M.C.A	U. A/Mira Béjaïa

Promotion 2022 – 2023

Remerciements

Nous tenons tout d'abord à exprimer notre profonde gratitude envers Allah pour nous avoir donné la force, la persévérance et la sagesse nécessaires pour mener à bien ce travail.

Nous voudrions adresser des remerciements spéciaux à notre encadrant, ***M. Achour ACHROUFENE****, dont l'expertise, la patience et les conseils avisés ont été inestimables pour l'aboutissement de ce mémoire. Ses éclairages ont éclairé notre route tout au long de ce processus.*

Nous souhaitons également exprimer notre reconnaissance envers les membres du jury, pour leur précieux temps et le grande honneur d'évaluer ce travail.

Un sincère merci à nos parents, notre famille et nos proches pour leur soutien indéfectible, leurs encouragements et leur confiance en nous. Leur présence a été notre source d'inspiration.

Nous n'oublions pas de remercier chaleureusement nos collègues et amis qui nous ont apporté leur aide et leurs encouragements tout au long de ce parcours.

Enfin, nous tenons à exprimer notre gratitude envers toute personne qui, de près ou de loin, a contribué à la réalisation de ce mémoire.

Dédicaces

Je tiens à dédier humblement ce travail à des personnes très chères à mon cœur.

Mes parents en particulier, qui ont eu la bienveillance de me transmettre une éducation empreinte de sagesse et de clarté de vision.

À mes sœurs Ikram, Lydia, Thanina et Aya.

À ma sœur Hanan et son mari Sid Ahmed.

À mes tantes et oncles.

À mon grand-père et ma grand-mère.

À toute le reste de la famille.

À ma chère binôme Sara.

Et tous mes amis.

Kinza

Dédicaces

Je tiens à dédier humblement ce travail à des personnes très chères à mon cœur

À mes chers parents, sources de lumière et de sagesse dans ma vie. Votre amour et votre soutien inconditionnels ont été le socle de toutes mes réussites. Vous êtes les étoiles qui illuminent mon chemin.

À ma sœur Asma.

À mes frères Amin et Fouad.

À mes tantes et mes oncles.

À ma chère grand mère.

À mon meilleur ami Youcef et tous mes amis et collègues.

Et sans oublier ma binôme Kinza.

Sara

Résumé

Le développement de systèmes intelligents exige une approche méthodologique adaptée à leurs spécificités. Les méthodes classiques se révèlent rigides, tandis que les méthodes agiles, bien qu'adaptatives, ne suivent pas toujours les étapes recommandées. Cette étude propose une amélioration de la méthode agile Scrum en intégrant un cycle d'apprentissage cohérent avec les besoins des systèmes intelligents . Les acteurs du domaine et les cogniticiens peuvent ainsi intervenir efficacement. Cette approche est appliquée au développement d'un système d'aide à la décision pour la prédiction des arrêts de pipelines chez SONATRACH.

Mots clés : Systèmes Intelligents, Méthodes de développement, Méthode Agile Scrum, Apprentissage automatique, Aide à la Décision, SONATRACH, Oléoduc.

Abstract

The development of Intelligent Systems requires a methodological approach tailored to their specificities. Classical methods tend to be rigid, while agile methods, although adaptive, may not always follow recommended steps. This study proposes an enhancement of the agile Scrum method by incorporating a learning cycle aligned with the needs of Intelligent Systems. Domain experts and cognitivists can thus intervene effectively. This approach is applied to the development of a decision support system for predicting pipeline shutdowns at SONATRACH.

Key words : Intelligent Systems, Development Methods, Agile Scrum Method, machine learning, Decision support, SONATRACH, Pipelines.

Table des matières

Table des figures

Liste des tableaux

Liste d'Acronymes

SLI Systèmes logiciels intelligents

IdO Internet des Objets

SIAD système interactif d'aide à la décision

KNN K-Nearest neighbor

RF Random Forest

LR Logistic Regression

XP Extreme programming

SVM support vector machines

SONATRACH Société Nationale pour la Recherche, la Production, le Transport, la Transformation, et la Commercialisation des Hydrocarbures

QSS Qt style sheets

SQL Structured query language

UML Unified Modeling Language

SMTP Simple Mail Transfer Protocol

HTTP Hypertext Transfer Protocol

OOSE Object-Oriented Software Engineering

BOOCH Grady Booch's Object-Oriented Design and Analysis Method

OMT Object Modeling Technique

AXIAL Analyse et Conception de Système d'Information Assistées par Logiciels

MERISE Méthode d'Étude et de Réalisation Informatique pour les Systèmes d'Entreprise

SAGACE Solution Algorithmique Génétique pour l'Anticipation de Comportements Evolutifs

SADT Structured Analysis and Design Technique

RTC Région Transport Centre

SOPEG Société Pétrolière de Gérance

IHM Interface Homme-Machine

DOT Department of Transportation

CV Cross Validation

Introduction générale

De nos jours, les systèmes intelligents sont vraiment importants et jouent un rôle clé, en particulier au sein des entreprises, en aidant les décideurs et les opérateurs dans les tâches qui leur ont été attribuées, telles que la gestion de grandes quantités de données et la prise de décisions[18].

Cependant, pour développer rapidement ces systèmes et assurer leur bon fonctionnement, il est essentiel de disposer d'une méthodologie appropriée qui tienne compte de leurs spécificités. En effet, le génie logiciel offre une multitude de méthodes de développement de logiciels, mais celles-ci ne répondent pas de manière adéquate aux exigences des systèmes logiciels intelligents (SLI)[19]. D'une part, les méthodes classiques peuvent être utilisées, car elles suivent fidèlement les étapes du développement des SLI. En revanche, elles sont trop rigides et n'acceptent pas les changements qui peuvent survenir avec l'arrivée de nouvelles connaissances au cours du développement. D'autre part, les méthodes agiles sont favorables au changement, mais ne suivent pas les étapes de développement recommandées pour développer les SLI. Néanmoins, les méthodes agiles nous semblent plus intéressantes dans le sens où elles sont plus adaptatives et peuvent être améliorées pour répondre aux besoins des SLI ou des systèmes intelligents en général.

Dans ce contexte, nous proposons d'améliorer la méthode agile Scrum en y ajoutant des éléments représentants les spécificités des systèmes intelligents. Le choix de la méthode Scrum est due à sa forte adaptabilité, son efficacité et sa large utilisation dans le développement logiciel.

Nous allons ajouter à la méthode Scrum un cycle d'apprentissage qui soit cohérent avec les besoins des systèmes intelligents à savoir : une phase d'acquisition de connaissances, une phase de conceptualisation et une phase d'implémentation propre à l'apprentissage. De plus, les expert du domaine et le cogniticien, qui sont des acteurs indisponibles pour les systèmes intelligents, pourront intervenir dans ce cycle.

Nous allons appliquer l'approche proposée pour le développement d'un système d'aide à la décision pour la prédiction de l'arrêt des pipelines de l'entreprise SONATRACH.

Ce mémoire est organisé de la manière suivante :

Les deux premiers chapitres sont consacrés à l'état de l'art. Le premier chapitre présente d'une manière détaillée les systèmes intelligents en se focalisant sur les systèmes d'aide à la décision. Le deuxième chapitre est un état de l'art sur les méthodes de développent logiciel. Il met en valeur les méthodes agiles en donnant un aperçu de quelques méthodes les plus utilisées actuellement.

Le troisième chapitre constitue notre contribution. Après l'exposition de la problématique, il s'étale sur l'amélioration apportée à la méthode agile Scrum pour le développement des systèmes intelligents tout en justifiant le choix de Scrum et les éléments de l'amélioration.

Les chapitres restants sont consacrés à l'application de la méthode proposée pour le développement d'une application d'aide à la décision. Le quatrième chapitre est dédié à la présentation de l'organisme d'accueil et au sprint planification. Le chapitre six porte sur la réalisation de la fonctionnalité "prédiction de l'arrêt de pipelines" où sera mis en oeuvre le cycle apprentissage proposé. Les chapitre cinq et sept sont dédiés à la réalisation des autres fonctionnalités du systèmes.

Nous terminons le mémoire par une conclusion qui résume l'essentiel de notre travail et quelques perspectives.

Chapitre 1

Les systèmes intelligents

Introduction

Les systèmes intelligents sont de plus en plus présents dans notre vie quotidienne. Ils ont révolutionné la manière dont les êtres humains interagissent avec les technologies. Grossièrement, un système intelligent sert à résoudre des problèmes complexes de manière automatique et efficace dans des environnements particuliers en faisant collaborer des personnes et des technologies telles que le bigdata, l'internet des objets, les réseaux de communication, la robotique, l'intelligence artificielle, etc.

Dans ce chapitre, nous explorons les systèmes intelligents en présentant leurs définitions, leur fonctionnement, leurs composants, ainsi que les applications les plus utilisées aujourd'hui. Nous examinons ensuite les différentes classifications proposées dans la littérature selon divers critères. Par la suite, nous mettons l'accent sur les systèmes d'aide à la décision en commençant par définir ces systèmes et les concepts relatifs à la prise de décisions. Nous abordons également les approches de décision, les architectures et topologies de ce type de systèmes pour mieux comprendre leur fonctionnement et leur utilité. Enfin, certaines techniques de l'apprentissage automatique sont présentées sommairement.

1.1 Systèmes intelligents

Dans l'objectif de répondre à la croissance technologique accélérée de ces dernières années et aux besoins des personnes et des organisations dans un monde de plus en plus interconnecté sont nés les systèmes intelligents et ont gagné en importance rapidement. D'une manière générale, un système quelconque est un ensemble d'éléments interagissant pour atteindre un objectif commun [130]. Mais qu'est-ce qu'un système intelligent concrètement ?

1.1.1 Définitions

Nous trouvons plusieurs définitions des systèmes intelligents dans la littérature et aucune ne fait l'unanimité jusqu'à présent. Pour [132], un système intelligent peut être défini comme étant « une machine intégrant un ordinateur connecté à Internet capable de collecter et d'analyser des données, ainsi que de communiquer avec d'autres systèmes ». Une autre définition est proposée par l'auteur de [17], un système intelligent est « un système informatique qui utilise des algorithmes d'intelligence artificielle pour collecter et analyser des données, afin de prendre des décisions autonomes ». A partir de ces définitions nous pouvons comprendre que ces systèmes sont capables de communiquer avec d'autres machines ou systèmes en ligne, et peuvent prendre des décisions de manière autonome, en se basant sur les données collectées et analysées.

Ces définitions semblent restrictives puisqu'elles ne font pas référence au coté organisationnel ou environnemental. Une définition récente d'un système intelligent est donnée par Molina dans son article intitulé "Qu'est-ce qu'un système intelligent ? [96] : "Un système intelligent fonctionne dans un environnement avec d'autres agents, possède des capacités cognitives telles que la perception, le contrôle de l'action, le raisonnement délibératif ou le langage, suit des principes de comportement fondés sur la rationalité et les normes sociales, et a la capacité de s'adapter par l'apprentissage." Dans ce contexte, les systèmes intelligents visent à permettre à des organisations spécialisées, en s'appuyant sur diverses parties physiques, numériques et humaines d'atteindre un objectif commun. Ces arrangement, interaction et apprentissage entre tous ces composants vont modifier le modèle de travail des entreprises actuelles afin qu'elles puissent s'adapter et se développer dans ce nouvel environnement collaboratif entre les machines et les humains.

1.1.2 Fonctionnement des systèmes intelligents

Habituellement, les systèmes intelligents utilisent des capteurs pour collecter des informations à partir d'un environnement spécifique et les partager entre ses différents éléments afin d'atteindre un objectif commun. Cette interconnexion entre le monde virtuel et le monde physique est connue sous le de l'internet des objets (IdO) [1]. Un autre élément qui contribue à ce type de système est le BigData, à savoir la possibilité de collecter des informations et des connaissances au sein d'un système à partir de grandes quantités d'informations provenant de différentes bases de données. La technologie BigData bénéficie de retour d'expériences des systèmes intelligents grâce aux technologies de l'intelligence artificielle [2] et de l'apprentissage automatique.

Propriétés des systèmes intelligents

Selon le domaine d'activité, les systèmes intelligents peuvent varier considérablement, cependant ils présentent des propriétés élémentaires communes qui sont :

1. **Capteurs :** c'est la technologie qui permet d'obtenir des données de l'environnement et de les transmettre au noyau d'intelligence à des fins d'identification et d'analyse.
2. **Actionneurs :** c'est l'élément qui exécute les actions que le noyau d'intelligence détermine après avoir analysé l'environnement en temps réel.
3. **Environnement spécifique :** c'est le contexte que le système intelligent analyse et modifie, qui peut être statique ou dynamique, discret ou continu, épisodique, déterministe ou connu.
4. **Noyau de renseignement :** ce noyau utilise principalement l'intelligence artificielle et en particulier l'apprentissage automatique. C'est ce qui permet de générer des connaissances sur la situation et d'apprendre de celle-ci et de prendre des décisions.
5. **Interface utilisateur (IU) :** c'est le moyen par lequel un agent externe communique et modifie la relation entre le système et l'environnement.
6. **Agents externes :** Il s'agit des personnes qui supervisent le processus du système intelligent ou même d'autres intelligences artificielles.

La figure 1.1 représente un schéma simplifié du fonctionnement d'un système intelligent.

1. L'internet des objets (IdO) est le processus de connexion à l'internet d'objets physiques de tous les jours équipés de capteurs, de logiciels et d'autres technologies permettant de recevoir des données d'autres objets [18].
2. L'intelligence artificielle est un domaine de l'informatique qui cherche à donner aux machines la capacité d'accomplir des tâches qui normalement nécessitent l'intelligence humaine [93].

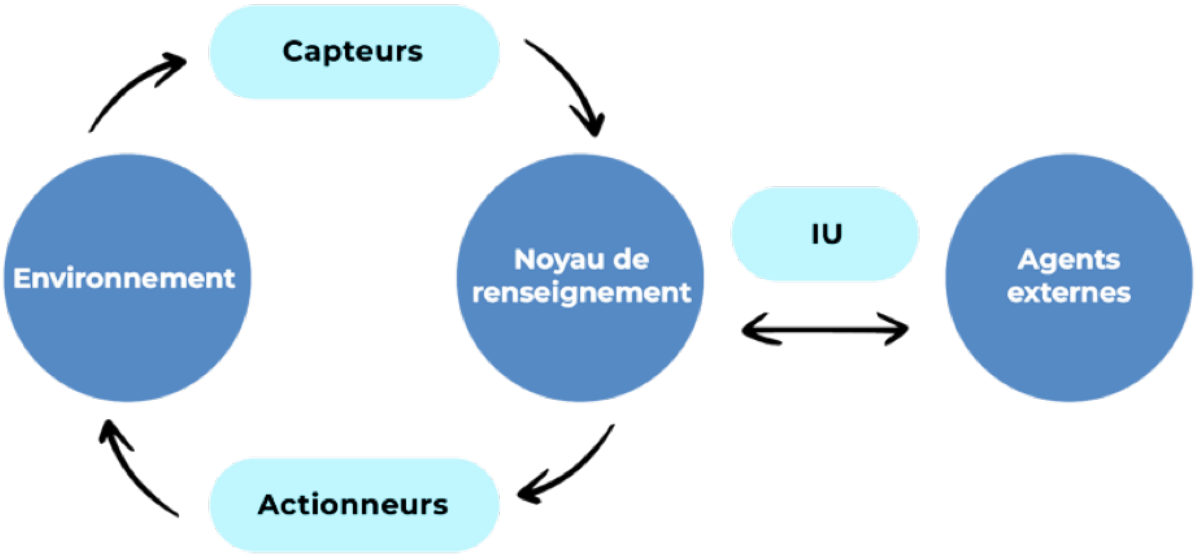

FIGURE 1.1 – Fonctionnement d'un système intelligent [18].

Caractéristiques des systèmes intelligents

Les principales caractéristiques d'un système intelligent [18] sont présentées dans la figure 1.2 et peuvent être résumées comme suit :

1. **Perception :** Un système intelligent construit une représentation du monde afin d'interagir avec un environnement spécifique et d'effectuer des tâches.
2. **Contrôle des actions :** Un système intelligent peut effectuer des actions ou les interrompre pour atteindre un objectif.
3. **Interaction ou connectivité :** Un système intelligent peut mettre ses éléments en communication par le biais d'un langage commun.
4. **Raisonnement délibéré et social :** La machine prend des décisions de son propre chef pour obtenir un résultat spécifique, tout en considérant le contexte humain.
5. **Auto-apprentissage :** Les systèmes intelligents sont capables de réduire les erreurs et d'optimiser leurs performances en tirant les leçons de leurs propres expériences.
6. **Identification :** Il est possible pour les systèmes intelligents de reconnaître automatiquement des informations spécifiques et de les transmettre par le biais de différents canaux.
7. **Protection :** Pour fonctionner correctement, les réseaux et les communications d'un système intelligent doivent être sécurisés.
8. **Gestion à distance :** Un système intelligent permet à des personnes d'interagir avec lui à partir de n'importe quel endroit.
9. **Expérience utilisateur :** Les systèmes intelligents doivent avoir des interfaces accessibles et ajustables pour interagir avec les utilisateurs.
10. **Analyse des données :** Un système intelligent doit être capable de traiter d'immenses quantités de données.

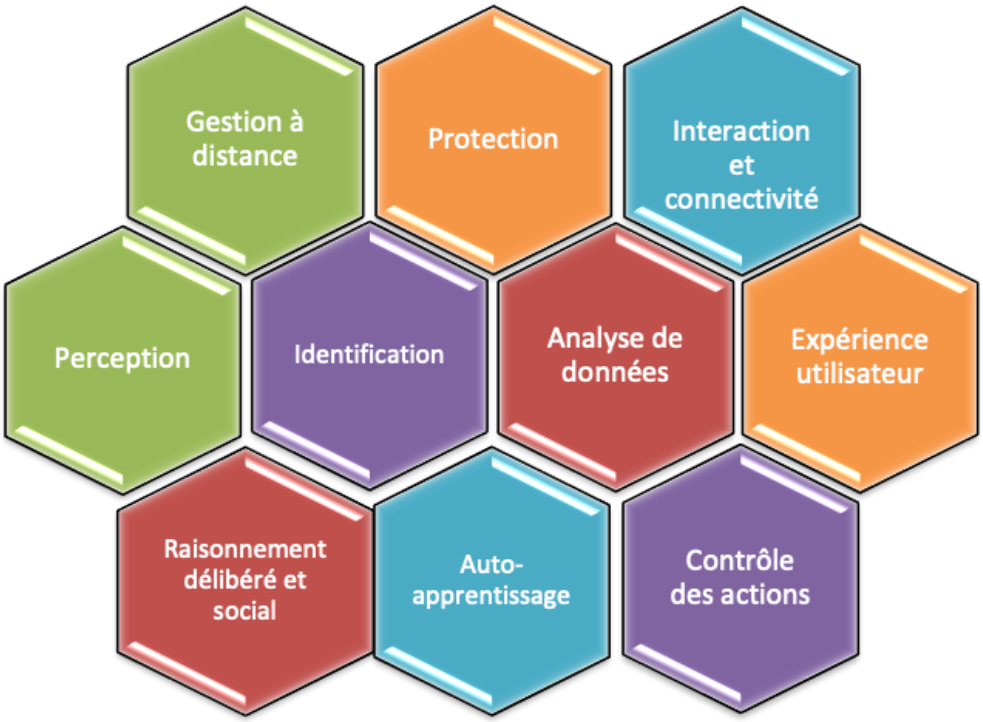

FIGURE 1.2 – Caractéristiques des systèmes intelligents.

1.2 Classification des systèmes intelligents

Les chercheurs ont tenté de regrouper les systèmes intelligents en catégories pour mieux comprendre les différents types existants et identifier les caractéristiques clés de chaque catégorie. Plusieurs classifications ont été proposées basées sur différents critères, tels que la fonction du système, le type d'apprentissage utilisé ou le niveau d'autonomie du système, etc.

1.2.1 Classification basée sur la fonction des systèmes intelligents

L'auteur de [54] a proposé une classification pour les systèmes intelligents selon leurs fonctions. Cette classification consiste à ranger les différents systèmes informatiques qui ont des fonctions similaires dans des groupes distincts :

- ⋆ **Systèmes experts :** Un système expert désigne un logiciel informatique [141] utilisé pour résoudre des problèmes complexes ou prendre des décisions dans un domaine spécifique en utilisant des connaissances expertes. Les connaissances sont souvent stockées sous forme de règles, qui sont utilisées pour prendre des décisions. Les systèmes experts les répondus dans cette catégorie sont les systèmes d'aide au diagnostic médical.
- ⋆ **Systèmes de recommandation :** selon [109] ces systèmes sont des outils conçus pour proposer des produits, des services, des films ou des musiques à des utilisateurs en fonction de leurs préférences et de leur historique. Comme exemple un système qui aide les utilisateurs à acheter des produits en ligne.
- ⋆ **Systèmes de reconnaissance de formes :** ces systèmes sont conçus pour reconnaître des formes, des images, des sons, des caractères ou des objets à partir de données brutes telles que des pixels ou des ondes sonores [118]. Un exemple illustrant ces systèmes est une application de reconnaissance de visages en analysant les pixels de l'image pour identifier les visages qui figurent sur une photo prise avec un téléphone portable.

⋆ **Systèmes de traitement de langage naturel :** d'après [50] ces systèmes sont conçus pour comprendre et générer des textes et des dialogues en langage naturel. Ils peuvent utiliser des techniques telles que la segmentation, l'étiquetage, l'analyse syntaxique, la génération de texte, etc. L'application la plus en vogue cette année est ChatGPT [3].

⋆ **Systèmes d'aide à la décision :** selon[135] les systèmes d'aide à la décision (SAD) sont des solutions logicielles conçues pour aider les individus ou les organisations à prendre des décisions éclairées et stratégiques en utilisant des données, des analyses et des modèles informatiques. Ils combinent souvent des informations provenant de diverses sources pour fournir une vue d'ensemble complète, facilitant ainsi la prise de décision.

⋆ **Les systèmes de traitement de l'information :** Ces systèmes collectent et analysent des données pour en extraire des informations utiles. Ils sont utilises dans des domaines tels que la surveillance de l'environnement, la reconnaissance vocale et la vision par ordinateur. Les auteurs donnent comme exemple de système de traitement de l'information un système de détection de fraude bancaire.

1.2.2 Classification basée sur le niveau d'autonomie des systèmes intelligents

Une deuxième approche de classification des systèmes intelligents repose sur leur niveau d'autonomie [97] :

⋆ **Systèmes à faible autonomie :** d'après les auteurs de [118], sont des systèmes intelligents conçus pour accomplir des tâches spécifiques en utilisant des techniques d'intelligence artificielle telles que la reconnaissance de formes ou la classification de données.

⋆ **Systèmes à haute autonomie :** Ces systèmes sont conçus pour effectuer des tâches plus complexes et sont capables de prendre des décisions indépendantes dans des environnements dynamiques et incertains [141].

1.2.3 Classification basée sur la complexité et le niveau d'autonomie des systèmes intelligents

Une autre classification des systèmes intelligents proposée par l'auteur de [126], en plus de l'autonomie, elle ajoute le critère de la complexité des systèmes :

— **Systèmes intelligents autonomes :** des systèmes capables de fonctionner de manière autonome, sans intervention humaine. Nous pouvons prendre comme exemple les robots autonomes.

— **Systèmes intelligents semi-autonomes :** ce sont des systèmes qui ont besoin d'une certaine intervention humaine pour fonctionner correctement. Les voitures autonomes nécessitants une intervention humaine dans certaines situations est un exemple de ces systèmes.

— **Systèmes intelligents interactifs :** des systèmes conçus pour interagir avec les utilisateurs. Tels systèmes peuvent être les chatbots et les assistants virtuels.

— **Systèmes intelligents de surveillance :** ce sont des systèmes développés pour surveiller et contrôler un environnement. Nous citons comme exemple les systèmes de surveillance de sécurité.

— **Systèmes intelligents de décision (SAD) :** des systèmes qui sont conçus pour aider à prendre des décisions complexes en fournissant des informations et des recommandations. Les systèmes de gestion des stocks est l'exemple communément présenté.

3. https ://openai.com/chatgpt

— **Systèmes intelligents de planification :** ce sont des systèmes mis en œuvre pour planifier et organiser des activités complexes à l'image des systèmes de planification de la production.

1.3 Applications des systèmes intelligents

Les systèmes intelligents apportent des avantages aux organisations en travaillant sur des solutions concrètes. Parmi les applications de systèmes intelligents que les entreprises utilisent pour améliorer leurs fonctionnements, leurs produits et/ou leurs services, nous citons [18] :

— **Robots autonomes :** Il s'agit de l'un des systèmes intelligents les plus complexes, mais aussi de l'un de ceux qui ont connu la plus forte croissance et la plus grande étendue ces dernières années. Ils ont été conçus à l'origine pour des activités industrielles critiques. Toutefois, au fil des années, ils ont été incorporés dans les marchés domestiques et automobiles. C'est une technologie très prometteuse qui ouvre de nouveaux domaines de collaboration dans les situations à haut risque.

— **Systèmes experts :** Ces systèmes utilisent les connaissances accumulées et l'expérience antérieure pour faire des déductions et simuler des stratégies de résolution. Ils sont développés pour maîtriser les connaissances et simuler le fonctionnement du cerveau humain. Ils collectent des informations pour résoudre les problèmes les complexes. Il est nécessaire d'interpréter les données pour parvenir à la bonne solution en fonction des objectifs de l'entreprise, tels que l'analyse, le traitement et le diagnostic pour les patients et les médecins.

— **Traitement du langage naturel :** La recherche dans ce domaine porte sur la façon dont les machines communiquent avec les gens en utilisant les langues humaines, telles que l'espagnol et l'anglais. Toutes les langues peuvent être traitées par des ordinateurs, ce qui en fait un outil plus avancé pour l'analyse des textes, car il est plus facile d'analyser ces informations sous forme électronique. Google est une entreprise utilisant ce type de technologie pour son moteur de recherche ou ses applications de traduction linguistique.

— **Vision artificielle :** La vision par ordinateur (ou artificielle) permet aux ordinateurs de comprendre les informations visuelles qu'ils reçoivent. Pour que cette application puisse acquérir des informations à partir d'une image ou d'une vidéo, des algorithmes basés sur des le machine Learning doivent être mis en œuvre. En plus de la compréhension d'images et de vidéos, cet outil permet également de classifier, de détecter et de suivre des objets.

— **Analyse des sentiments :** Il s'agit d'un domaine de l'intelligence artificielle et de la linguistique appliquée qui étudie les interactions entre les machines et les humains. Cet outil est actuellement mis en œuvre dans divers réseaux sociaux pour reconnaître les concepts et les intentions de la personne qui écrit le message. Il utilise des technologies avancées d'intelligence artificielle, de traitement du langage naturel, d'analyse de texte et de science des données pour identifier, extraire et étudier les informations subjectives. Avec cette technologie, les entreprises ont appris comment les clients réagissent à un produit ou à un service spécifique en classant les textes comme positifs, négatifs ou neutres.

Ces systèmes intelligents offrent aux entreprises la possibilité d'optimiser divers processus et sont essentiels pour générer des produits qui répondent aux besoins de la société afin d'améliorer la qualité des produits et des services.

1.4 Exemples de systèmes intelligents

Plusieurs systèmes intelligents sont utilisés quotidiennement dans notre vie. Cela va des objets de tous les jours, comme les montres intelligentes, à des solutions beaucoup plus étendues, comme les systèmes de fabrication intelligents pour des projets de grande envergure. Quelques exemples de l'utilisation et de la mise en œuvre de ces systèmes intelligents sont présentés ci-dessous [18] :

- **Montres intelligentes :** elles ont connu une grande popularité au cours de la dernière décennie. En se connectant par Bluetooth à un autre appareil (généralement un smartphone) ou par Wi-Fi directement à un réseau, les smartwatches peuvent recevoir et passer des appels téléphoniques, lire des courriels, écouter de la musique, gérer des fichiers ou même accéder à des sites web. Certaines smartwatches peuvent également surveiller des données biométriques, telles que la fréquence cardiaque, la pression artérielle, l'oxygénation ou le nombre de pas et l'activité physique effectués pendant une certaine période. Ces données biométriques sont utiles pour la santé de nombreuses personnes, en particulier celles qui souffrent de maladies nécessitants des contrôlées réguliers.
- **Systèmes de surveillance et d'analyse des réseaux :** c'est des logiciels qui permettent aux utilisateurs de surveiller les réseaux informatiques et leurs composants. Ils connectent les appareils et les logiciels tout en optimisant les informations qui circulent entre eux. Ils se sont avérés essentiels pour les organisations qui dépendent de ces réseaux, car une erreur de connexion peut mettre en péril l'ensemble des opérations.
- **Planification et livraison des produits :** le système de fabrication intelligent implique la collaboration entre les humains, les machines et les processus afin d'optimiser la planification et la livraison des produits dans le cadre d'un processus de fabrication donné. Pour cela, il crée des modèles de planification, d'agencement de l'usine, de conception de la production et d'utilisation des machines, entre autres, pour créer, proposer et mettre en œuvre les meilleurs processus pour un certain produit, en se concentrant non seulement sur la quantité ou la qualité, mais aussi sur la préservation des ressources et la durabilité.
- **Inspection visuelle de l'IA :** Bien entendu, l'inspection visuelle est nécessaire pour de nombreux processus de production afin de détecter des défauts ou des problèmes dans certains produits ou certaines zones. De nos jours, elle est particulièrement importante dans les usines dont les chaînes de montage traitent des milliers de produits par jour. Elle permet donc de minimiser les erreurs humaines et de gagner du temps, raison pour laquelle l'IA a été intégrée à la procédure. Ce processus d'inspection est basé sur la vision par ordinateur et l'apprentissage profond (Deep learning) en utilisant plusieurs éléments technologiques pour surveiller et inspecter une opération afin de s'assurer qu'elle fonctionne correctement et que ses produits répondent aux normes requises. L'inspection visuelle par IA est essentielle dans les chaînes de montage ou d'autres processus automatisés puisqu'elle peut inspecter les produits partiellement ou entièrement, ce qui permet de gagner du temps et de réduire les erreurs humaines.
- **Détection de la criminalité :** la détection et la prévention de la criminalité sont facilitées par la mise en œuvre de systèmes de surveillance intelligents. Ceux-ci relient les humains, les ordinateurs, les logiciels, les caméras, l'internet et d'autres éléments pour créer un réseau de surveillance capable de détecter en temps réel les délits éventuels ou les délits en cours dans un certain rayon d'action. L'IA peut apprendre à reconnaître des modèles spécifiques et des comportements sociaux, ainsi qu'à mettre en œuvre des fonctions de reconnaissance faciale, ce qui signifie qu'elle peut être entraînée à détecter et à signaler des activités illicites, en transmettant rapidement ces informations à un canal particulier tel que les services de police ou les stations de surveillance.

La figure 1.3 est une illustration des exemples des systèmes intelligents présentés.

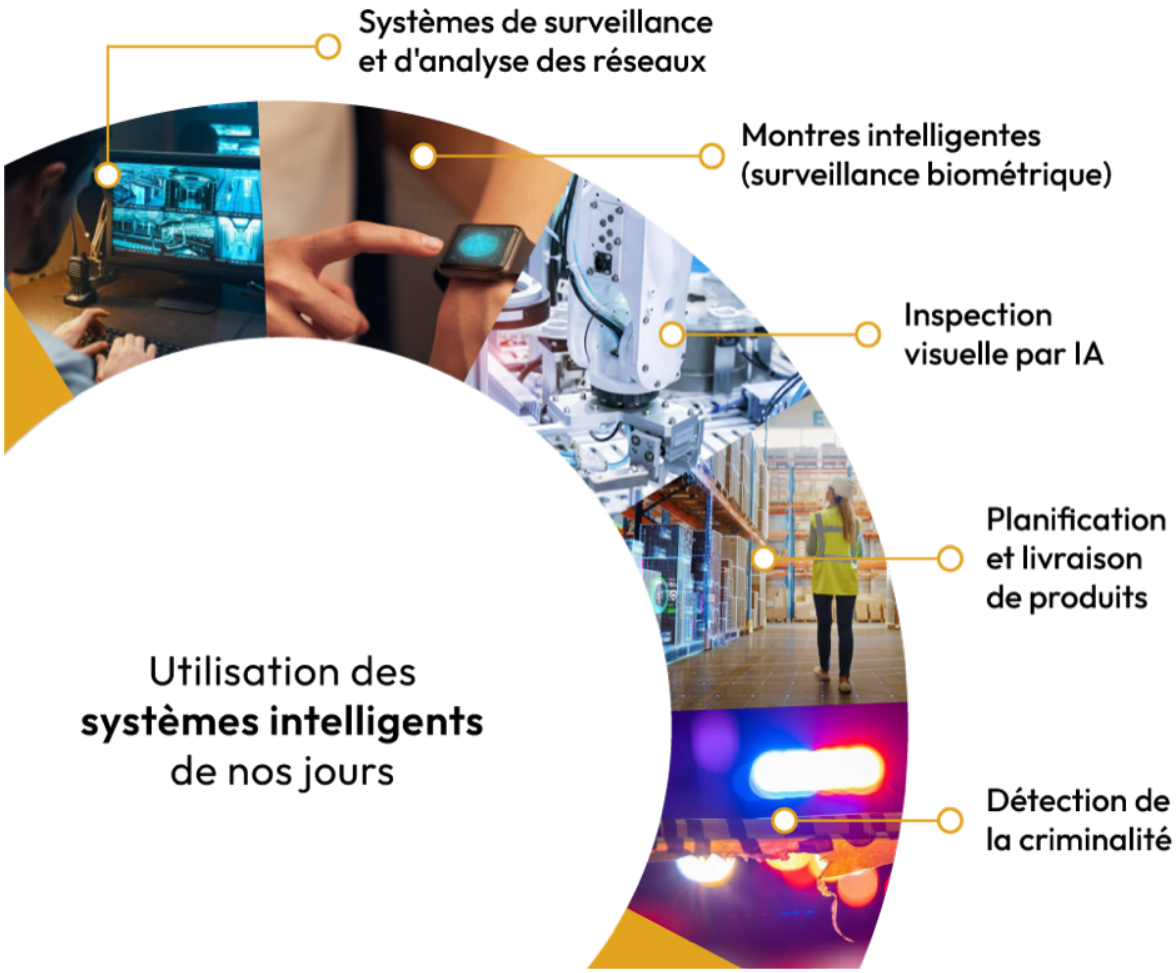

FIGURE 1.3 – Exemples de systèmes intelligents [18].

1.5 Système d'aide à la décision (SAD)

Dans la suite de ce document, nous allons nous intéresser aux systèmes logiciels intelligents [138] et en particulier aux systèmes d'aide à la décision (SIAD). En effet, notre travail s'insère dans le contexte de développement d'une application informatique pour aider les responsables d'une entreprise à prendre des décisions dans le secteur de l'hydrocarbure. Pour cela, nous présentons les SIAD d'une manière détaillée.

1.5.1 Décision et prise de décision

Le terme « décision » a plusieurs significations qui le considèrent comme une action, une activité ou un processus visant à résoudre un problème. Parmi les diverses définitions de ce terme, nous présentons quelques-unes dans le tableau 1.1.

Ce tableau montre que le processus de prise de décision est complexe et que chacune de ces définitions met en avant un aspect différent de la prise de décision. Schneider insiste sur le processus de sélection des alternatives, tandis que Levin souligne la nécessité de prendre en compte les éléments de la situation pour prendre une décision. Roy et Bouysson mettent en avant l'utilisation de modèles pour aider à la prise de décision, tandis que Mintzberg insiste sur

Auteur	Définitions	Caractéristiques
Schneider [15]	Lorsqu'on prend une décision, on sélectionne une option parmi plusieurs possibilités en se basant sur des critères qu'on a déjà définis auparavant.	Le processus global de résolution de problèmes consiste à sélectionner les objectifs à atteindre ainsi que les alternatives possibles pour y parvenir.
Levin [104]	La décision est une action prise pour faire face à une difficulté ou répondre à une modification de l'environnement.	Prise en compte de la situation ou de l'environnement.
Roy et Bouysson [112]	L'aide à la décision consiste en l'expertise qui utilise des modèles pour fournir des réponses aux interrogations d'un décideur.	Il s'agit d'utiliser des modèles précisément définis pour éclairer la prise de décision et promouvoir un comportement en accord avec les objectifs et les valeurs.
Mintzberg [94]	La décision est le choix que l'on fait après avoir réfléchi, analysé et pensé aux conséquences possibles. C'est comme un plan d'action que l'on décide de mettre en place.	Engagement à agir. Volonté claire et déterminée d'agir en conséquence.

TABLE 1.1 – Tableau résumé des définitions de la décision selon plusieurs auteurs et de leurs caractéristiques

la nécessité d'une intention explicite d'agir. En considérant leur contexte et leur portée, il est possible de tirer pleinement profit de ces perspectives pour améliorer le processus de prise de décision.

1.5.2 Processus de décision

Un processus de décision est un ensemble d'étapes à suivre pour prendre la décision. Ce processus peut aider à minimiser les erreurs qui peuvent affecter le jugement du décideur. Simon [125] a proposé un schéma général de prise de décision pour les organisations présenté par la figure 1.4. Ce schéma se compose de quatre phases principales [22] qui sont :

- ⋆ **La phase intelligence :** Dans cette phase, l'organisation identifie le problème qui besoin une décision. Les informations pertinentes sont collectées, analysées et synthétisées pour déterminer la nature du problème.
- ⋆ **La phase de modélisation :** Dans cette phase, différentes options sont envisagées pour résoudre le problème identifiée dans la première phase. Des hypothèses sont formulées et des modèles sont développés pour évaluer les conséquences potentielles de chaque option.
- ⋆ **La phase de choix :** Dans cette phase, l'organisation choisit la meilleure option parmi les autres dans la phase de conception. Les avantages et les inconvénients de chaque option sont évalués en fonction des critères de décision établis.
- ⋆ **La phase d'évaluation :** cette phase, le décideur examine toutes les options qu'elle

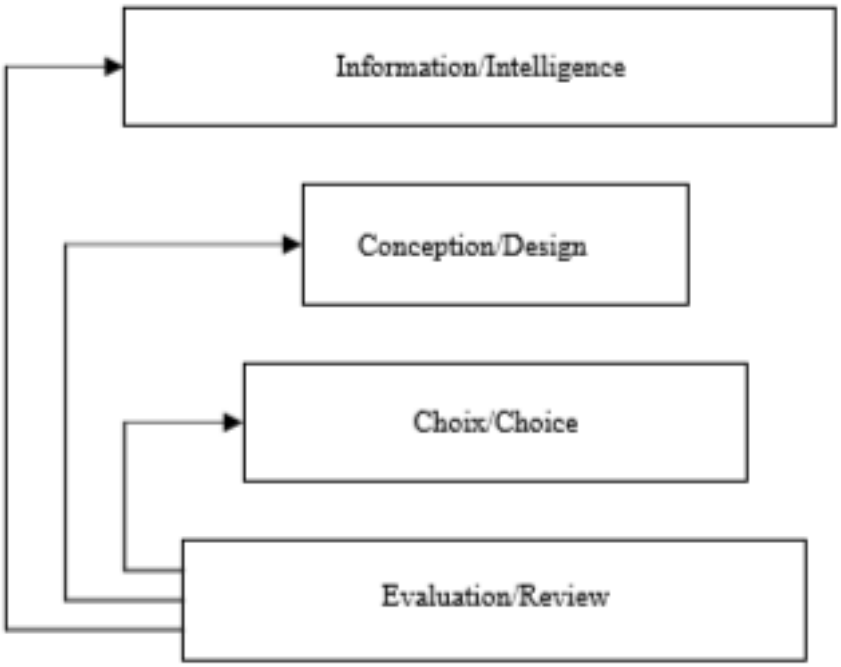

FIGURE 1.4 – Le processus de décision (Simon, 1977)

a trouvées. Ensuite, il va évaluer ces options en utilisant des critères spécifiques tels que : est-ce que c'est possible à faire ? Est-ce que ça rapportera de l'argent ? Est-ce que les ressources nécessaires sont disponibles ? Est-ce que cela correspond aux objectifs de l'organisation ? En utilisant ces critères, il va choisir la meilleure option.

Il convient de noter que ce modèle est quelque peu simpliste et ne reflète pas nécessairement la complexité des décisions prises par les organisations modernes. Le processus de décision peut être amélioré grâce à l'utilisation d'un système interactif d'aide à la décision (SIAD) qui fera l'objet de la section suivante.

1.5.3 Système interactifs d'aide à la décision (SIAD)

Les systèmes interactifs d'aide à la décision sont l'un des types de systèmes d'aide à la décision les plus populaires car ils ont la capacité de simplifier la prise de décision, d'améliorer la collaboration entre les utilisateurs et de fournir des résultats instantanés. Les travaux [59], [125], [133], [66] et [89] définissent un comme un logiciel informatique complexe qui aide les décideurs à résoudre des problèmes peu ou mal structurés en utilisant des techniques avancées de traitement de l'information. Il facilite l'interaction entre l'utilisateur et la machine par l'interface homme/machine et permet une prise de décision de qualité et rapide .

Caractéristiques des SIAD

En se basant sur la définition précédente, nous pouvons identifier plusieurs caractéristiques clés des SIAD parmi lesquelles nous citons :

- ⋆ **Interactivité :** les SIAD permettent une interaction directe entre l'utilisateur et le système pour formuler des requêtes, effectuer des analyses et prendre des décisions [134].
- ⋆ **Convivialité :** d'après l'auteur de [106] les SIAD sont conçus pour être faciles à utiliser et à comprendre par les utilisateurs, même sans connaissances approfondies en informatique.

⋆ **Flexibilité :** l'analyse faite dans [40] révèle que les SIAD sont capables de traiter une grande variété de types de données, de formats et de sources, et peuvent être configurés pour effectuer différents types d'analyses en fonction des besoins des utilisateurs.

⋆ **Support multi-utilisateurs :** les SIAD peuvent être utilisés par plusieurs utilisateurs simultanément [124] pour collaborer sur des tâches complexes de prise de décision.

Évidemment les caractéristiques clés des SIAD peuvent varier en fonction des spécificités de chaque SIAD, ainsi que des besoins et des objectifs de l'utilisateur. En effet, certains SIAD peuvent avoir des fonctionnalités supplémentaires ou différentes pour répondre aux exigences particulières de chaque situation.

Architecture des SIAD

Une des architectures proposée est celle présentée dans [133], [119]. Cette architecture est composée de quatre composantes comme le montre la figure 1.5 : Interface homme machine, base d'information, base de modèle, base de connaissance.

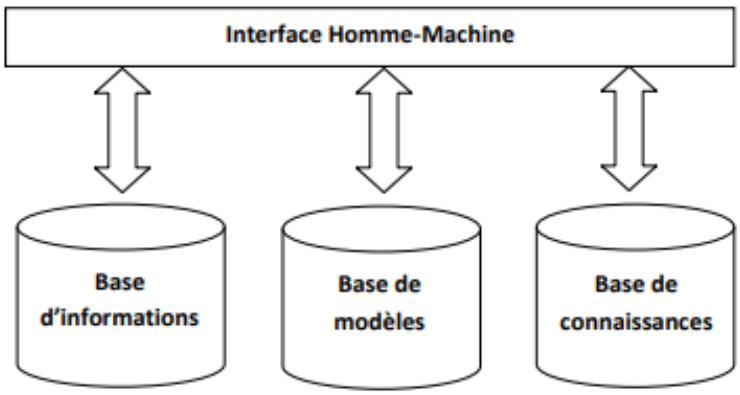

FIGURE 1.5 – Architecture du SIAD [133], [119]

⋆ **Interface homme-machine :** c'est la partie du système qui permet à le décideur de visualiser, analyser et interagir avec les informations pertinentes pour prendre des décisions éclairées[107].

⋆ **Base d'information :** selon [79] elle joue un rôle essentiel en stockant des informations de manière permanente ou temporaire, ainsi qu'en gérant l'enregistrement et l'effacement de données volatiles. Ces données volatiles peuvent inclure les résultats obtenus lors du traitement de données et sont effacées en fonction de la demande de l'utilisateur.

⋆ **Base de connaissance :** contient des connaissances concernant le problème étudié selon [79]

⋆ **Base de modèle :** selon [39]] aide à prendre des décisions est un ensemble de modèles et d'un système qui gère ces modèles. Les modèles peuvent être différents types, comme des outils pour résoudre des problèmes ou des modèles pour analyser des données. Pour être plus flexible, un tel système doit avoir plusieurs modèles différents.

Une autre architecture plus évoluée, mais plus complexe, des SIAD est présentée dans [139]. Elle peut être représenté brièvement en trois couches :

1. **Couche de données (Alimentation par les applications opérationnelles) :** d'après [49] la couche de données stocke les données brutes provenant de différentes sources internes ou externes à l'organisation. Elle est responsable de l'extraction, de la transformation et du chargement de ces données dans un entrepôt de données pour faciliter l'analyse ultérieure.

2. **Stockage historisé, l'agrégation et la constitution des hypercubes :** cette couche implique le stockage de données provenant de différentes sources dans les hypercubes de manière à pouvoir les utiliser pour l'analyse et la prise de décision [139]. Le silo d'échange est une base de données utilisée par chaque application pour stocker les événements ou les photographies au fur et à mesure de leur occurrence. Le moteur d'alimentation du SIAD consulte périodiquement le silo, recopie ses éléments vers une base temporaire nommée « sas ».
3. **Couche de représentation :** elle est responsable de présenter les informations générées sous forme de graphiques, tableaux, etc. pour faciliter la prise de décision [119]. Elle permet également une interaction facile entre les utilisateurs et les informations présentées.

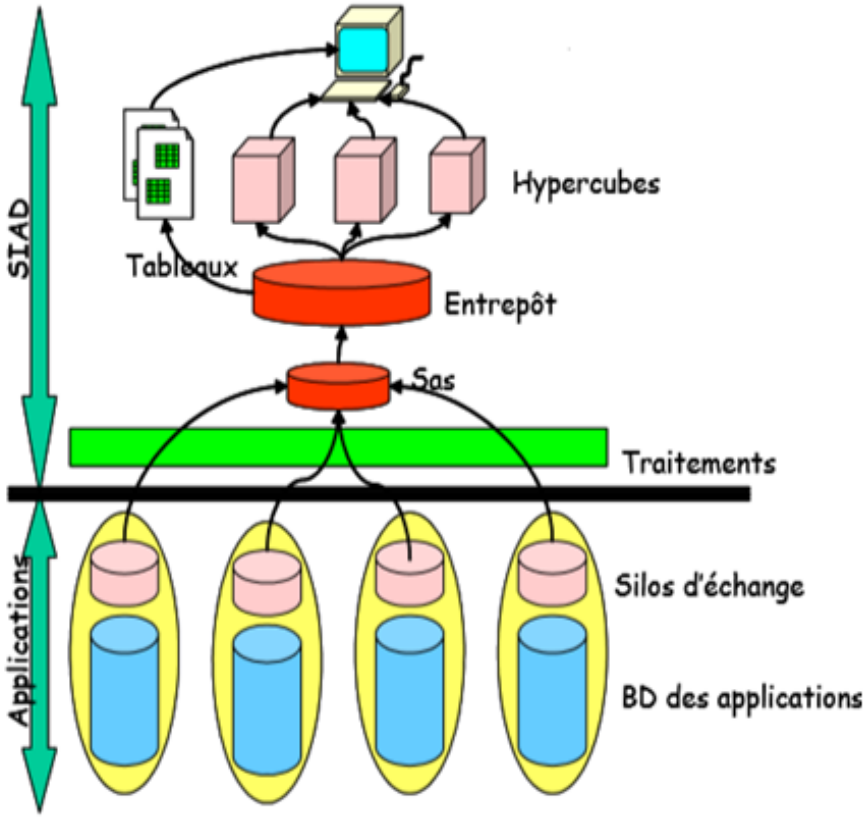

FIGURE 1.6 – Architecture du SIAD [139]

D'après [22], le type de SIAD le plus évolué est appelé SIAD orientés connaissance. Ils sont définis [107] comme des systèmes d'information qui permettent de résoudre des problèmes en utilisant des connaissances stockées sous différentes formes. Ils sont appelés "intelligents" car ils sont capables de prendre des décisions en utilisant ces connaissances. Ces systèmes sont utiles pour des tâches complexes qui nécessitent beaucoup de connaissances.

Afin qu'un système soit qualifié d'intelligent, il doit être capable d'apprendre automatiquement. Pour ce faire, il devra faire appel à des techniques d'intelligence artificielle, en particulier l'apprentissage automatique (ou machine learning). En effet, l'intégration de l'apprentissage automatique dans les SIAD présente des avantages significatifs tels que :

— **Amélioration de la précision des Prédictions :** Les algorithmes d'apprentissage automatique peuvent déceler des schémas et effectuer des anticipations en se basant sur des données historiques, ce qui peut améliorer la précision.

— **Traitement de Données Volumineuses :** De la même manière que les systèmes d'aide à la décision dotés de vastes quantités de données, les algorithmes d'apprentissage automatique peuvent également traiter et analyser de grandes quantités d'informations.

1.6 Techniques de machine learning

L'apprentissage automatique est défini par Samuel [120] comme « un domaine d'étude qui fournit aux ordinateurs la capacité d'apprendre sans être explicitement programmés pour le faire ». Cela offre la capacité aux machines de développer des connaissances et de s'adapter aux données pour résoudre des problèmes complexes. L'apprentissage automatique est vue comme une branche en développement d'algorithmes informatiques conçus pour imiter l'intelligence humaine en apprenant de l'environnement qui les entoure [24]. Il joue un rôle essentiel dans la nouvelle ère du traitement des grandes quantités de données.

Les différentes techniques de machine learning peuvent être classées comme suit :

1.6.1 Apprentissage supervisé

L'apprentissage supervisé, c'est un peu comme quand on apprend quelque chose en suivant des exemples. Le modèle apprend à prédire ou à classer de nouvelles choses en se basant sur des exemples avec des étiquettes. Ces exemples lui servent à comprendre comment fonctionnent les choses, et ensuite, il peut prendre des décisions sur des données qu'il n'a jamais rencontrés auparavant. En résumé, le modèle suit l'exemple des données déjà étiquetées pour apprendre à faire des prédictions sur des données nouvelles et inconnues[95].

Il existe deux types de ce technique d'apprentissage supervisé [62] :

— **Apprentissage supervisé de classification :** Cette méthode vise à former un modèle afin qu'il puisse prédire des étiquettes ou des catégories spécifiques pour de nouvelles données. Par exemple, elle peut être utilisée pour trier des e-mails en distinguant ceux qui sont indésirables de ceux qui ne le sont pas, ou pour anticiper si un patient présente des symptômes spécifiques liés à une maladie donnée.[62].

— **Apprentissage supervisé de régression :** elle vise à prédire une valeur numérique continue en fonction des caractéristiques d'un exemple [75]. Cela peut être utilisé, par exemple, pour prédire le prix d'une voiture en fonction de ses caractéristiques.

1.6.2 Apprentissage non supervisé

L'apprentissage non supervisé permet au modèle d'explorer les données pour trouver des schémas, des similarités ou des anomalies, sans avoir besoin de réponses préétablies ou d'étiquettes [62]. Grossièrement, c'est un peu comme si on donnait à un ordinateur un tas de données et qu'il devait les trier tout seul pour trouver des motifs, des ressemblances ou des trucs bizarres, sans qu'on lui dise ce qu'il doit chercher à l'avance.

1.6.3 Apprentissage par renforcement

Dans ce type d'apprentissage un agent apprend à prendre des décisions en interagissant avec son environnement. L'agent reçoit des renforcements ou des conséquences en fonction de ses actions, ce qui lui permet d'apprendre quelle est la meilleure action à prendre dans différentes situations pour obtenir le maximum de récompenses à long terme [98].

1.6.4 Quelques techniques d'apprentissage

Dans cette sous-section, nous allons donner un bref aperçu des méthodes à utiliser au cours de ce travail : K-Nearest Neighbors, régression logistique, Support Vector Machine et random forest.

K-Nearest Neighbors (KNN)

L'algorithme des k plus proches voisins (KNN) [47] est un algorithme de classification et de régression largement utilisé en apprentissage automatique supervisé. Son principe est simple : pour prédire la classe (dans le cas de la classification) ou la valeur (dans le cas de la régression) d'un nouvel échantillon, KNN recherche les K échantillons les plus proches dans l'ensemble de données d'apprentissage et utilise leur classe ou valeur moyenne comme prédiction.

Voici comment fonctionne l'algorithme KNN selon [68] :

1. **Calcul de la distance :** La première étape consiste à calculer la distance entre le nouvel échantillon à prédire et tous les échantillons de l'ensemble de données d'apprentissage. La distance peut être mesurée en utilisant différentes métriques telles que la distance euclidienne, la distance de Manhattan, la distance de Minkowski, etc.
2. **Sélection des K voisins :** Une fois les distances calculées, les K échantillons les plus proches sont sélectionnés en fonction de la valeur de K que nous avons spécifiée.
3. **Vote majoritaire :** le résultat est déterminé par un vote majoritaire parmi les K voisins. C'est-à-dire que la classe la plus fréquente parmi les K voisins est attribuée au nouvel échantillon (voir figure 1.7). Si K = 1, alors le nouvel échantillon est simplement attribué à la classe de son voisin le plus proche.

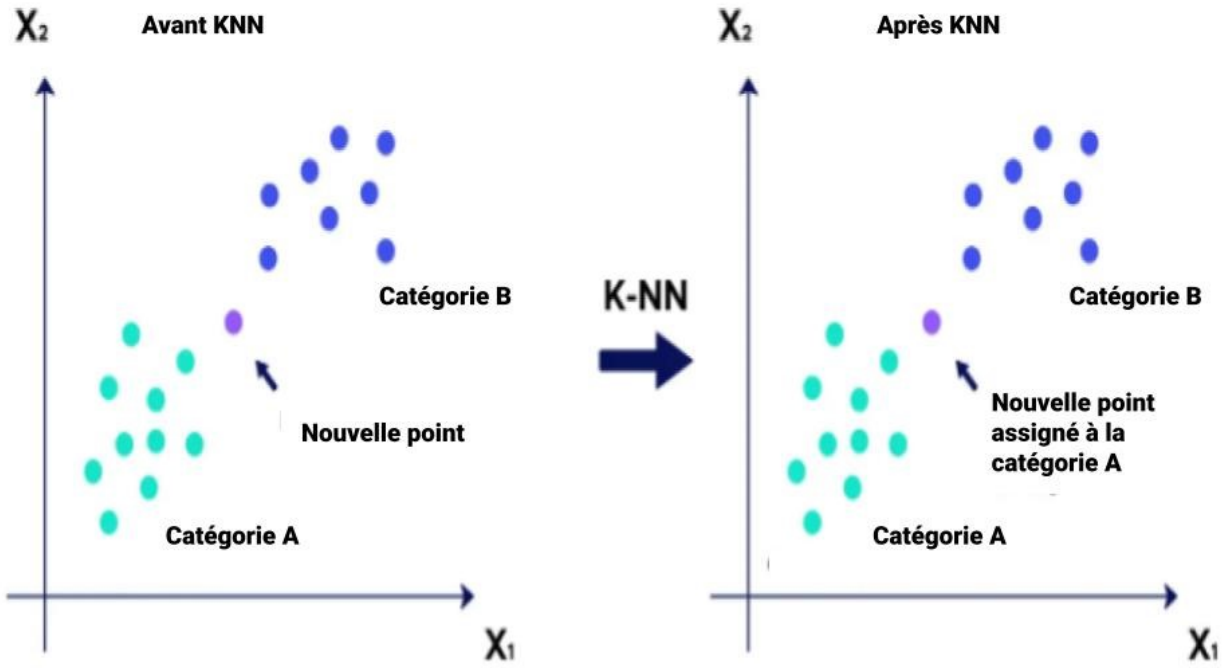

FIGURE 1.7 – Classifcation avec KNN[47] ;

Régression logistique (RL)

La régression logistique est un algorithme utilisé pour la classification binaire. Son objectif est de prédire la probabilité qu'une observation appartienne à une classe particulière [69].

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

où :

- z représente l'entrée de la fonction sigmoïde. Dans la régression logistique, z est la combinaison linéaire des caractéristiques d'entrée et de leurs poids correspondants, ainsi qu'un terme de biais facultatif. Il peut être calculé comme $z = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$, où w_i est le poids associé à la caractéristique x_i.
- $\sigma(z)$ représente la sortie de la fonction sigmoïde, qui est la probabilité estimée que l'entrée appartienne à la classe positive (par exemple, la classe 1).

La fonction sigmoïde est utilisée pour modéliser la relation entre les caractéristiques d'entrée et la probabilité de l'événement étudié. Une fois entraîné sur un ensemble de données, le modèle peut être utilisé pour prédire la classe cible des nouvelles observations en utilisant les coefficients appris. Comme le montre la figure 1.8 si la probabilité prédite est supérieure à un seuil prédéfini (généralement 0,5), l'observation est classée comme appartenant à la classe positive(classe 1), sinon elle est classée comme appartenant à la classe négative(classe0).

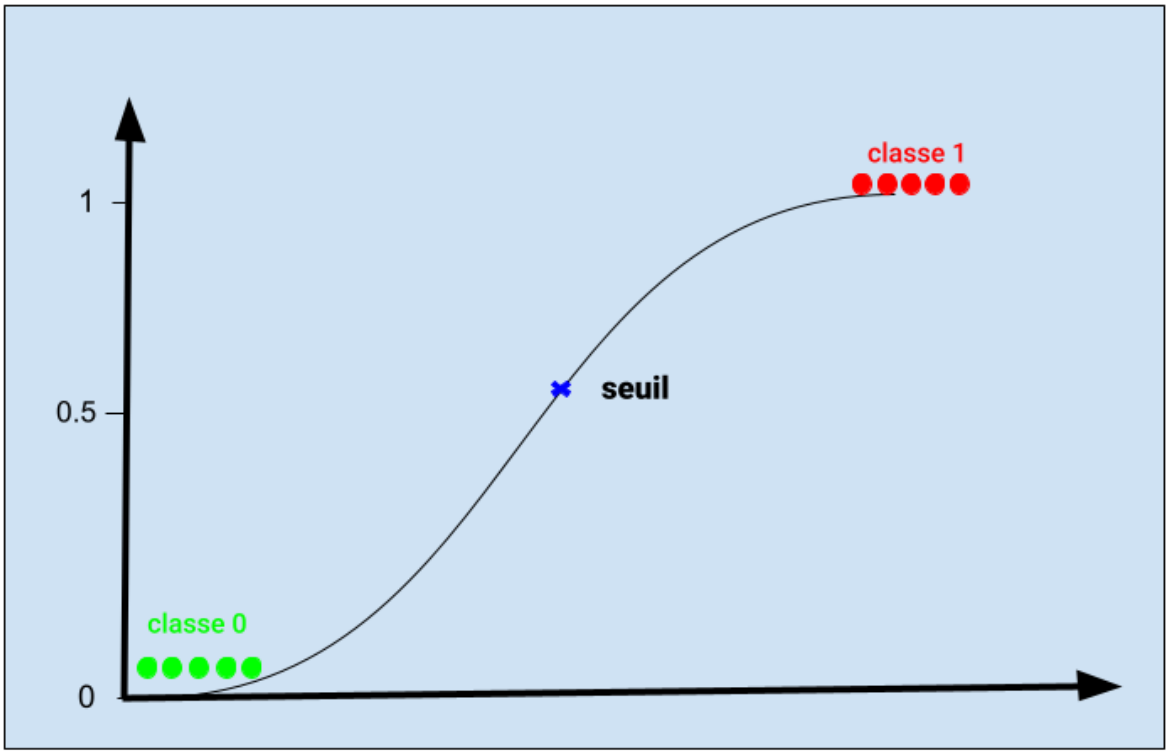

FIGURE 1.8 – Classification avec régression logistique

Support Vector Machine (SVM)

L'algorithme de Support Vector Machine (SVM) est une technique d'apprentissage supervisé utilisée pour la classification et la régression. Son objectif principal est de trouver la meilleure séparation entre deux classes dans un espace multidimensionnel [137]. Les concepts principaux de SVM sont les suivants :

1. **Principe de base :** Dans le contexte de SVM, les points de données sont représentés par des vecteurs de caractéristiques (features), où chaque dimension du vecteur correspond à une coordonnée. Ainsi, les coordonnées des points de données définissent l'espace dans lequel l'algorithme recherche l'hyperplan optimal pour séparer les deux groupes de manière

optimale. Le processus de SVM consiste à trouver le meilleur hyperplan qui maximise la marge entre les deux classes(voir figure1.9), permettant ainsi au modèle d'être plus performant et généraliser efficacement sur de nouvelles données.

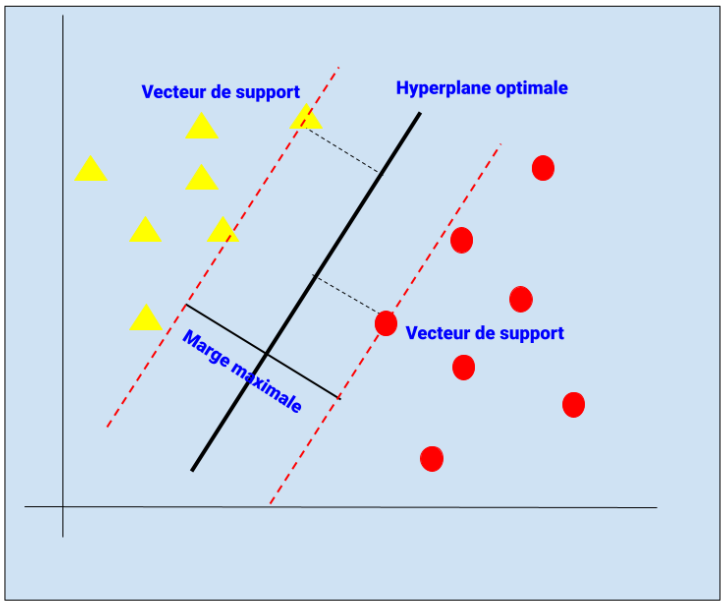

FIGURE 1.9 – Classification avec SVM

2. **Données linéairement séparables :** Dans le cas où les deux classes de données sont linéairement séparables (c'est-à-dire qu'il est possible de tracer une ligne droite ou un plan pour les séparer), SVM trouve directement cet hyperplan optimal.
3. **Données non linéairement séparables :** Dans de nombreux cas réels, les données ne sont pas linéairement séparables, ce qui signifie qu'un simple hyperplan ne suffit pas pour les séparer. Dans ces situations, SVM utilise une technique appelée "noyau" (kernel) pour transformer l'espace d'origine en un espace de dimension supérieure où les données deviennent linéairement séparables. Les noyaux les plus couramment utilisés sont le noyau linéaire, le noyau polynomial et le noyau gaussien (RBF - Radial Basis Function).
4. **Marges souples :** Dans certains cas, il est impossible de trouver un hyperplan qui sépare parfaitement les deux classes. C'est là qu'intervient le concept de "marges souples". SVM permet de tolérer certaines erreurs de classification en autorisant des points à se trouver de part et d'autre de l'hyperplan, mais en imposant une contrainte sur ces erreurs. L'objectif reste de maximiser la marge tout en minimisant ces erreurs.
5. **Régularisation :** En plus de la marge, SVM utilise également une technique de régularisation pour éviter le surajustement (overfitting) du modèle. La régularisation contrôle la complexité du modèle en ajoutant un terme de régularisation à la fonction objectif. Cela permet d'éviter que le modèle ne devienne trop complexe et ne s'adapte trop précisément aux données d'entraînement.

Random Forest (RF)

Random Forest (RF) est un algorithme d'apprentissage appartenant à la famille des méthodes d'ensemble. Son fonctionnement repose sur la construction d'un ensemble d'arbres de décision et la combinaison de leurs prédictions pour obtenir une prédiction globale.

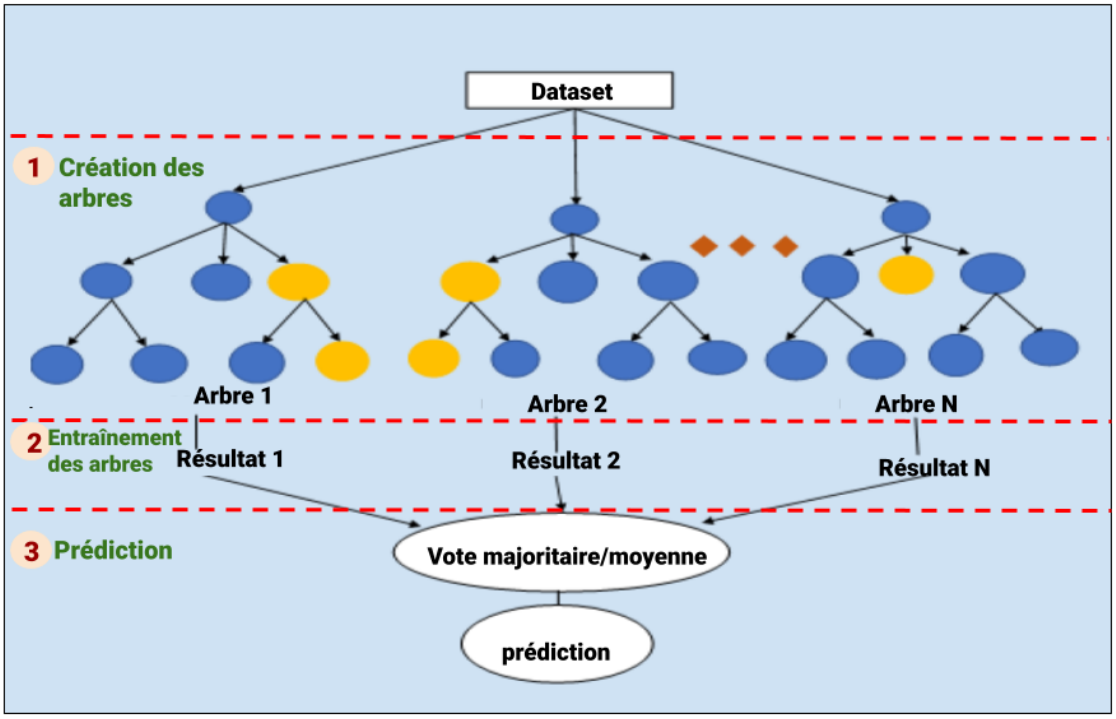

FIGURE 1.10 – Prédiction avec Random Forest

Comme le montre la figure 1.10, l'algorithme Random Forest est constitué de 3 étapes principales [71] qui sont :

1. **Construction des arbres de décision :** RF construit un certain nombre d'arbres de décision en utilisant différentes sous-parties des données d'entraînement et des caractéristiques disponibles. Chaque arbre est formé à partir d'un échantillon aléatoire (bootstrapped) des données d'entraînement, ce qui signifie que certaines données sont répliquées dans l'échantillon, tandis que d'autres sont omises. En outre, pour chaque division dans l'arbre, un sous-ensemble aléatoire de caractéristiques est considéré, ce qui aide à réduire la corrélation entre les arbres et à augmenter la diversité de l'ensemble.
2. **Entraînement des arbres :** Chaque arbre de décision est entraîné de manière récursive en divisant les données en sous-groupes basés sur les caractéristiques et leurs valeurs. À chaque étape de la division, l'algorithme recherche la caractéristique qui permet de séparer au mieux les données selon les classes cibles (dans le cas de la classification) ou de minimiser l'erreur de régression (dans le cas de la régression). Cette procédure se répète jusqu'à ce qu'un critère d'arrêt soit atteint, par exemple, lorsque l'arbre atteint une profondeur maximale prédéfinie ou qu'un nombre minimum d'échantillons est présent dans chaque feuille.
3. **Prédiction :** Une fois que tous les arbres de décision sont construits, RF combine leurs prédictions pour obtenir une prédiction globale. Pour les tâches de classification, le modèle effectue un vote majoritaire sur les prédictions de chaque arbre, tandis que pour les tâches de régression, il calcule la moyenne des prédictions des arbres. Cette étape de combinaison permet de réduire le surajustement et d'améliorer la stabilité et la précision du modèle.

1.6.5 Tableau comparatif des modèles d'apprentissage étudiés

Le tableau 1.2 présente une comparaison des modèles d'apprentissage étudiés en montrant les principes de fonctionnement, les avantages et les inconvénients de chacun.

Modèle	Type	Principes de fonctionnement	Avantages	Inconvénients
KNN	Supervisé	Calcul des distances entre le nouvel échantillon et les échantillons d'apprentissage.	Facile à comprendre et à implémenter.	Sensible à la dimensionnalité (peut être inefficace avec de nombreuses caractéristiques).
RL	Supervisé	Modélise la relation entre les caractéristiques d'entrée et la probabilité de la classe cible en utilisant une fonction sigmoïde.	- Bonne interprétabilité. - Peut être utilisé pour la classification binaire et la régression. - Gestion des caractéristiques continues et catégorielles.	- Ne fonctionne pas bien avec des relations complexes entre les caractéristiques. - Peut sous-performer lorsque les données ne sont pas linéairement séparables.
SVM	Supervisé	Recherche d'un hyperplan optimal qui sépare les données en maximisant la marge entre les classes.	- Peut gérer des données non linéairement séparables en utilisant des noyaux. - Bonne performance en présence de nombreuses caractéristiques.	- Le choix du noyau et des paramètres peut être complexe. - Sensible au surajustement en cas de mauvais réglage des paramètres. - Temps de formation potentiellement long pour de grandes quantités de données.
RF	Ensembliste	- Construction d'un ensemble d'arbres de décision en utilisant des sous-échantillons de données et des caractéristiques aléatoires. - Combinaison des prédictions des arbres pour obtenir une prédiction globale.	- Bonne performance en général. - Robuste aux valeurs aberrantes et aux données bruitées. - Peut gérer des données avec de nombreuses caractéristiques.	- Moins interprétable que certains autres modèles. - Peut être gourmand en calcul et en mémoire avec de nombreux arbres et caractéristiques.

TABLE 1.2 – Tableau comparatif entre les modèles d'apprentissage étudiés

Chacun de ces modèles présente des avantages et des inconvénients, donc il est généralement une bonne idée de les tester sur nos propres données pour voir lequel fonctionne le mieux pour notre problème. En conséquence, le choix d'un modèle d'apprentissage dépend beaucoup plus du type de données à traiter et du but à atteindre.

1.6.6 Métriques d'évaluation

En apprentissage automatique, les métriques d'évaluation sont essentielles pour mesurer la qualité du modèle, identifier les points forts et les points faibles, comparer différents modèles, et prendre des décisions éclairées sur le choix du modèle le mieux adapté à un problème spécifique. Les principales métriques d'évaluation sont :

- **Accuracy (Exactitude) :** Mesure la proportion d'échantillons correctement classés parmi tous les échantillons du jeu de données. Son utilité est de fournir une mesure globale de la performance du modèle.

$$\frac{Vrais\ positifs + Vrais\ ngatifs}{Vrais\ positifs + Faux\ positifs + Vrais\ ngatifs + Faux\ ngatifs} \tag{1.1}$$

- **Precision (Précision) :** Représente la proportion d'échantillons positifs correctement identifiés parmi tous les échantillons identifiés comme positifs par le modèle. Elle est utile lorsque les faux positifs sont coûteux à éviter.

$$\frac{Vrais\ positifs}{Vrais\ positifs + Faux\ positifs} \tag{1.2}$$

- **Recall (Rappel) :** Indique la proportion d'échantillons positifs correctement identifiés parmi tous les échantillons réellement positifs du jeu de données. Il est utile lorsque les faux négatifs sont coûteux à éviter.

$$\frac{Vrais\ positifs}{Vrais\ positifs + Faux\ ngatifs} \tag{1.3}$$

- **F1-Score :** C'est une mesure harmonique entre la précision et le rappel, il combine ces deux métriques pour donner un équilibre entre les faux positifs et les faux négatifs. Il est utile lorsque l'on veut une évaluation globale des performances du modèle, en particulier lorsque les classes sont déséquilibrées.

$$2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{1.4}$$

- **Macro avg :** La moyenne non pondérée des métriques pour chaque classe, donnant la même importance à chaque classe indépendamment de sa taille.

$$\frac{\text{Somme des valeurs pour chaque classe}}{\text{Nombre de classes}} \tag{1.5}$$

- **Weighted avg :** La moyenne pondérée des métriques pour chaque classe, en prenant en compte le nombre d'échantillons de chaque classe.

$$\frac{\text{Somme des valeurs pour chaque classe} \times \text{Nombre d'échantillons pour chaque classe}}{\text{Nombre total d'échantillons}} \tag{1.6}$$

Conclusion

Ce chapitre a présenté les systèmes intelligents qui sont importants dans notre monde connecté et en particulier les systèmes d'aide à la décision qui sont devenus indispensables dans le fonctionnement des entreprises modernes pour traiter et analyser les données et aider à prendre des décisions.

Après avoir défini les systèmes intelligents, nous avons introduit leurs propriétés et caractéristiques et classé ces systèmes selon plusieurs critères. Nous avons également, recensé un certain nombre de domaines de leurs applications et d'exemples actuellement en vogue. Ensuite, nous nous sommes concentré sur le système d'aide à la décision, en définissant la prise de décisions et le processus décisionnel, ainsi que l'architecture des systèmes interactifs d'aide à la décision.

Enfin, nous avons abordé les techniques de l'apprentissage automatique qui sont le noyau des systèmes intelligents. Quelques techniques ont été brièvement présentées avant d'être comparées. Comme tout système informatique, le développement d'un SIAD suit un certains nombre d'étapes. Pour cela, dans le prochaine chapitre, nous allons étudier les méthodes de développement logiciel les plus connues dans la littérature.

Chapitre 2

État de l'art sur les méthodes de développement logiciel

Introduction

En matière de gestion de projets informatiques ou autres, pour être performant et assurer le succès, il est nécessaire de suivre un certain nombre d'étapes allant des besoins aux objectifs recherchés. En effet, plusieurs processus ont été conçus au fil des années pour bien mener les tâches et répondre aux besoins.

Dans le contexte informatique, avant la fin des années 90, les méthodes de développement de logiciels les plus courantes étaient basées sur une approche séquentielle en phases distinctes. Ces méthodes étaient connues sous le nom de méthodes classiques. Cependant, ces méthodes ont été critiquées en raison de leurs limitations, telles que la rigidité et le manque d'implication des clients dans le processus de développement. Cela a conduit à l'émergence de méthodes agiles plus flexibles, adaptatives et répondant mieux aux problèmes rencontrés dans les méthodes classiques.

Ce chapitre porte sur l'analyse de l'état de l'art des méthodes de développement en se focalisant sur les méthodes agiles. Nous allons présenter les méthodes de développement selon les différents points de vue qui existent dans la littérature, notamment les méthodes classiques, leurs types et leurs cycles de vie et comment ces méthodes ont donnée naissance aux méthodes agiles ? Nous allons également expliquer les principes de l'agilité en nous appuyant sur quelques exemples de méthodes les plus utilisées. Le chapitre se termine par une comparaison des méthodes étudiées et les méthodes hybrides.

2.1 Méthodes de développement logiciel

D'une manière générale, une méthode désigne un ensemble de moyens et de techniques ordonnés et systématisés qui sont utilisés pour atteindre un objectif précis. Elle repose sur des principes rigoureux et des règles précises qui sont appliquées avec discipline. Les méthodes sont largement utilisées dans divers domaines, tels que la recherche scientifique, la gestion de projet, la conception de logiciels, l'enseignement, etc[4].

Dans le contexte logiciel, les méthodes de développement logiciel sont un ensemble de processus et de techniques qui sont utilisées pour planifier et exécuter le développement d'applications logicielles. Ces méthodes aident à décomposer les tâches de développement complexes en composants plus petits et gérables, qui peuvent alors être travaillés de manière systématique et organisée. Ils sont conçus pour garantir le bon fonctionnement d'un projet de logiciel et pour veiller à ce que le projet soit terminé dans le budget et dans les délais impartis.

Afin d'enlever toute ambiguïté, nous commençons par clarifier certains concepts utilisés d'une manière indifférencié dans la littérature. En effet, souvent dans diverses références, les termes démarche, processus, cycle de vie et méthode sont utilisés pour designer la même chose. A notre sens les termes démarche, processus et cycle de vie ont des significations très proches et peuvent désigner l'ensemble d'étapes à suivre pour réaliser un logiciel. Mais le terme méthode est plus général et englobe les autres. Il est clairement défini par Rumbaugh [116] comme suit :

— Un ensemble de concepts fondamentaux de modélisation pour capturer la connaissance sémantique d'un problème et de sa solution ;

— Un ensemble de vues et de notations pour présenter la modélisation sous-jacente aux personnes qui les examineront et les modifieront ;

— Un processus itératif pas à pas employé pour la construction des modèles et pour leur implémentation ;

— Une collection de suggestions et de règles qui conduisent à l'exécution du développement ;

— Méta-schémas (« pattern ») est une tentative pour la représentation de l'expérience personnelle des développeurs d'une manière uniforme.

Il existe une gamme de méthodes de développement logiciel qui peuvent être classées selon l'ordre d'apparition dans le temps (chronologiquement), leurs structures ou méthodes classiques (prédictives) [72] et non classiques (agiles). Nous allons adopté cette dernière classe pour présenter les différentes méthodes et l'ordre chronologique pour donner un aperçu des méthodes classiques. A noter qu'il existe une autre classification selon le degré formalisation de la modélisation utilisée dans la méthode qui peut être informel, semi-formel ou formel, mais elle est au-delà du cadre de ce mémoire.

2.2 Méthodes classiques

Les méthodes classiques, d'après [65], sont des méthodes axées plan, c'est-à-dire qu'ils sont basées sur des processus bien définis pour répondre aux besoins de développement logiciel. De plus, l'étude [25] affirme que ces méthodes supposent que les projets évoluent dans un environnement connu avec de variations mineures et gérables d'où vient leur nature prédictive.

Afin de bien comprendre le fonctionnement des ces méthodes nous allons discuter par la suite les types de méthodes classiques en donnant quelques exemples. Mais avant cela, nous allons rappelé les différents cycles de vie les plus répondus proposés en génie logiciel sur lesquels reposent les méthodes prédictives.

2.2.1 Cycles de vie des méthodes classiques

Les méthodes classiques sont caractérisées par le fait qu'elles utilisent des cycles de vie prédictifs. kerzner [77] fait mention de deux modèles de cycle de vie, principalement linéaires, au sein des méthodes de gestion de projet classiques : le modèle en cascade et le modèle en V. Par la suite un troisième modèle a été introduit par Boehm en 1986 [35] qui est le modèle en spirale défini comme : *« Le modèle en spirale, à la fin des années quatre-vingt, propose une nouvelle vision du processus de développement en suivant une démarche itérative et incrémentale basée sur la réalisation de prototypes successifs. »* [64].

Cycle de vie en cascade

Le cycle de développement en cascade a été proposé par Winston Royce [114] pour améliorer la qualité et la fiabilité de développement des projets logiciels complexes. Cependant, il a

rencontré des problèmes tels que des spécifications mal comprises, du code inutilisable et une nécessité de prévoir parfaitement les tâches à réaliser. [29].

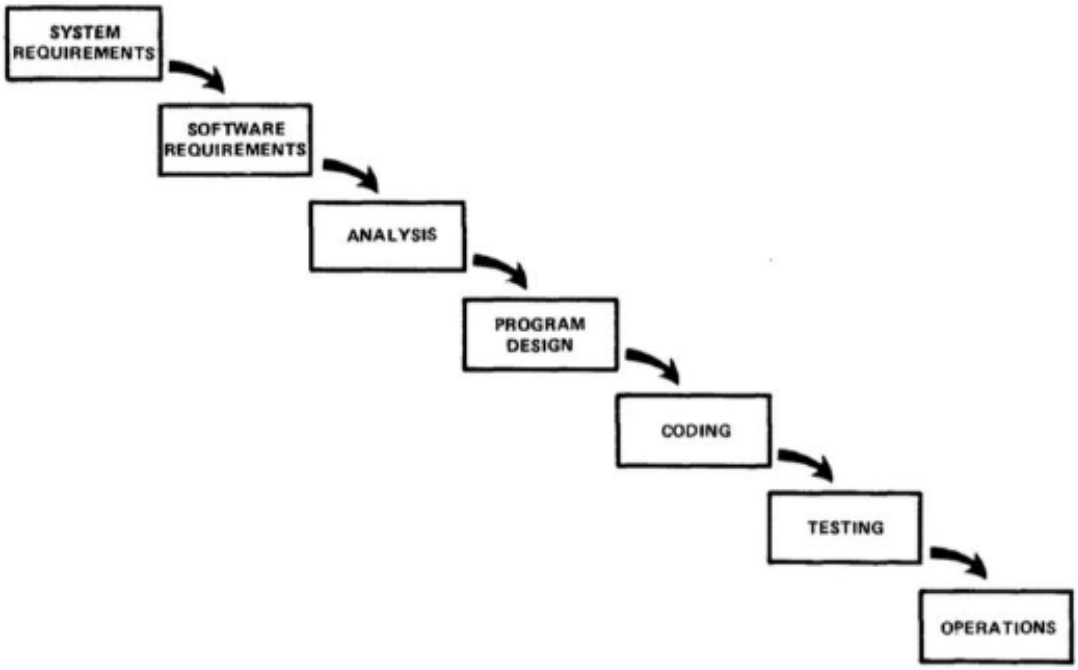

FIGURE 2.1 – cycle de vie en cascade [29].

La figure 2.1 présente les phases de cycle de vie en cascade. D'abord, les deux premières phases **system requirements** et **software requirements** consistent à collecter les besoins, puis se fera l'analyse de ces besoins (**analyses**), après viendra la phase de conception (**program design**) qui est une modélisation détaillée de système et faite pour ensuite établir la phase de programmation (**coding**). Une fois la programmation terminée, l'application subit différents tests (**testing**) pour assurer le bon fonctionnement, le cycle en cascade fini par la mise en ouvre (**operation**) de l'application et une éventuelle livraison.

En se basant sur le travail de [111], nous pouvons mettre en évidence les principales caractéristiques de cycle en cascade qui sont :

- ◇ On ne peut pas passer à la phase suivante que lorsque les livrables de la phase actuelle sont validés.
- ◇ Tous les besoins du projet sont exprimés et recueillis lors de la première phase.
- ◇ Le système est conceptualisé et validé avant la phase de développement.
- ◇ Les tests techniques et fonctionnels ne démarrent qu'après la phase du codage.
- ◇ L'intégration et la mise en production ont lieu après correction des anomalies.

Une variante du cycle en cascade est ainsi apparue afin d'identifier plus rapidement les anomalies en limitant le retour aux étapes précédentes. Il s'agit du cycle de vie en V.

Cycle de vie en "V"

Nous allons présenter le cycle de vie en "V" à l'aide de la figure 2.2 et l'étude faite dans [117].

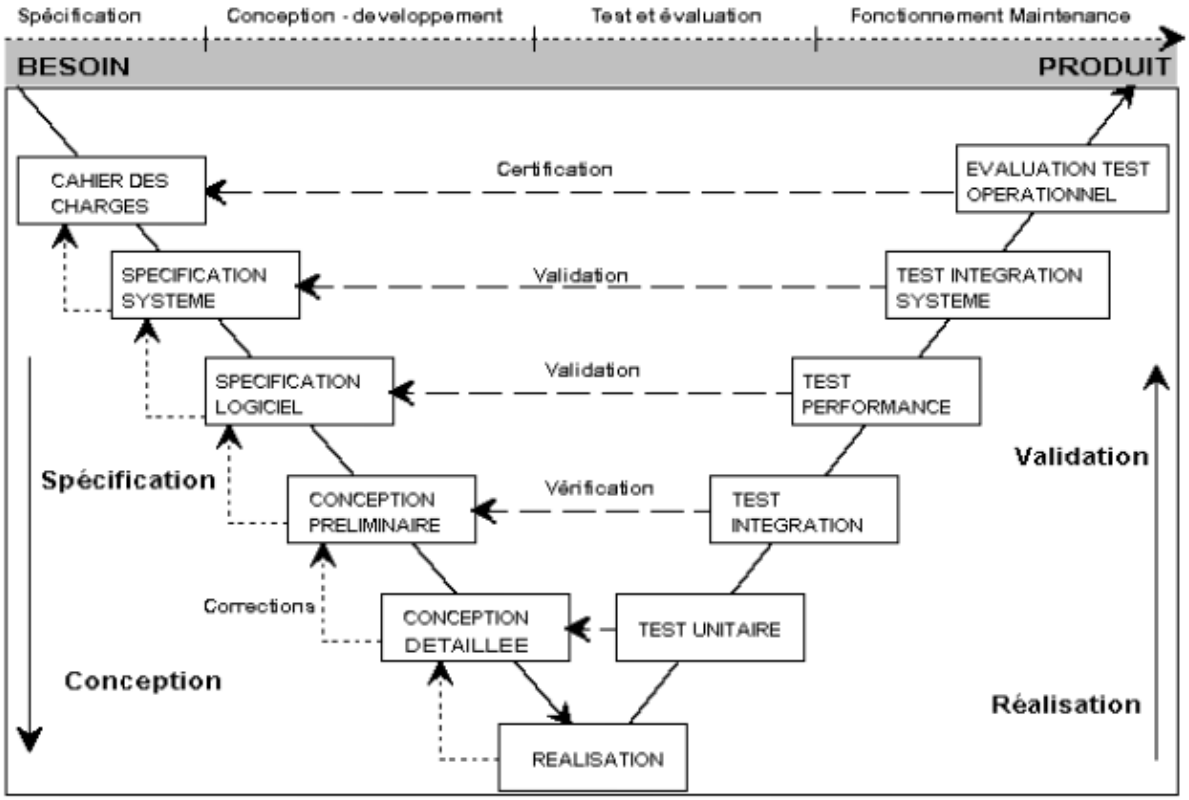

FIGURE 2.2 – Cycle de vie en V [64].

Le modèle du cycle en "V" a été crée pour éviter les problèmes de réactivité du cycle en cascade en cas d'anomalies. Le modèle est constitué d'une branche descendante qui comprend les étapes de conception du projet et d'une branche ascendante qui contient les étapes de tests comme le montre la figure la 2.2. La jonction des deux branches constitue la phase de codage du produit. Chaque étape d'une branche est en vis-à-vis avec l'étape de la branche opposée, les étapes de la branche montante doivent renvoyer de l'information à leur vis-à-vis lorsqu'une anomalie est détectée. Cela permet de déceler plus rapidement les problèmes que dans un cycle en cascade. Par ailleurs, cela offre la possibilité de préparer dans les étapes descendantes les besoins des étapes montantes. Ainsi, les résultats attendus des tests unitaires seront définis dans la conception détaillée.

Le modèle en V, tout comme le modèle en cascade, place l'utilisateur final de la solution à développer principalement dans la phase de définition des exigences. C'est un point qui a souvent été critiqué. Cependant, ce modèle a été largement adopté par les entreprises européennes à l'époque (les années 1990) [80].

Cycle de vie en spiral

Le modèle en spirale est un modèle de développement de logiciels proposé par Barry Boehm en 1986 pour résoudre certaines des faiblesses du modèle en cascade [35].

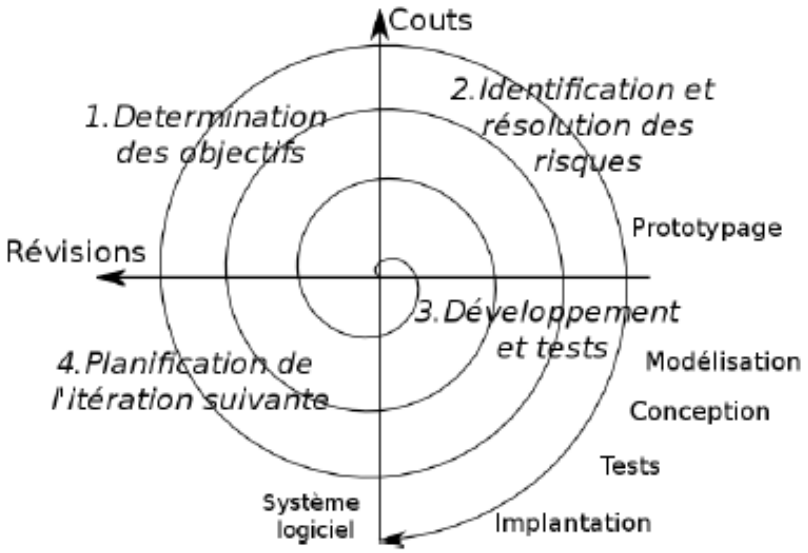

FIGURE 2.3 – cycle de vie en spiral [44].

Le modèle en spirale est un processus itératif et incrémental, où les cycles de développement sont répétés plusieurs fois avec l'ajout de nouvelles fonctionnalités à chaque étape. Chaque cycle de la spirale comprend quatre phases comme montré dans la figure 2.3 :

— **Détermination des objectifs** : elle s'agit de la toute première phase en haute à gauche de la figure dans laquelle les objectifs et les contraints sont définies.

— **Evaluation des risques** : cette phase consiste principalement à analyser les risques.

— **développement et tests** : le développement des fonctionnalités et les tests sont se font à ce niveau.

— **planification** cette phase vient pour planifier la prochaine étape ou itération.

L'accent est mis sur la gestion des risques pour déterminer le temps et les efforts nécessaires pour toutes les activités du cycle. Le modèle en spirale utilise également la révision des prototypes pour inclure les commentaires des utilisateurs et les intégrer dans le développement.

2.2.2 Quelques types de méthodes classiques

Selon [86], nous pouvons distinguer trois types de méthodes ordonnées chronologiquement : méthodes systémiques, méthodes cartésiennes et méthodes orientées objet. Chacun de ces type sera introduit brièvement dans cette sous-section. Néanmoins, nous tenons à souligner que le démarrage de l'approche industrielle de l'informatisation dans les années 60 a été faite avec des méthodes en informatique de gestion à l'image de CORIG [99].

Méthodes cartésiennes

Aussi nommées méthodes analytiques [82] ou méthodes fonctionnelles [55]. Ces méthodes permettent de procéder à un découpage fonctionnel et hiérarchique du système [86], autrement dit elles découpent le système en sous-système hiérarchique pour que le développement de fonctionnalités soit facile. Parmi lesquelles on peut citer : **Jakson**,**SA-SD**,**SADT**. Ces méthodes modélisent les sous-systèmes en utilisant quelques diagrammes à appliquer dans les étapes du cycle de vie en cascade.

Méthodes systémiques

Ces méthodes permettent une modélisation basée sur la séparation entre les données et les traitements, en s'appuyant sur de nombreux modèles complémentaires comme les modèles de données et les modèles de traitement, et en des définissant des différents niveaux d'abstraction (logique, conceptuel, etc.). Parmi ces méthodes on trouve : **AXIAL**, **MERISE**, **SAGACE**, etc. La méthode MERISE, par exemple, propose son propre cycle de vie qui intègre les niveaux d'abstraction et les phases d'un cycle de vie linéaire.

Méthodes orientées objets

Les méthodes orientées objets prennent les objets du monde réel comme base [30]. Tout d'abord, le système à développer est observé, analysé et les exigences sont définies. Ensuite, les objets du système requis sont identifiés et modélisés en fonction de leurs relations. Une fois cela fait, le codage du système est réalisé en suivant une démarche séquentielle d'analyse, de conception système, de conception objet et de mise en œuvre. Donc d'après [30] et [74], nous pouvons distinguer ces 4 phases comme suit :

- **Phase d'analyse système :** l'analyste interagit avec l'utilisateur pour déterminer les exigences de l'utilisateur et prépare un modèle du système souhaité.
- **Phase de conception système :** l'architecture globale du système est décidée et organisée en sous-systèmes interagissant les uns avec les autres.
- **Phase de conception objet :** l'identification des objets avec leurs structures de données et leurs interrelations.
- **Phase de mise en ouvre :** l'implémentation des détails de l'analyse et de la conception système avec les objets identifiés.

Il existe plusieurs méthodes orientées objet dont on peut mentionner :**OMT**,**OOSE**,**BOOCH**.

En résumé, les méthodes classiques sont caractérisées par processus et planifié, dont les exigences du système sont négociées avec les clients au début du projet et documentées dans des contras. Les exigences seront profondément analysées pour ensuite établir la conception et la construction du système. Les modèles appliqué dans le processus sont accompagnés avec une documentation complète et détaillée dans chaque étape. Cependant, il a été constaté que ces méthodes sont inadaptées aux environnements en constante évolution, conduisant souvent à l'échec du projet [142] [32] [33] [34] [53] [87] [58].

Dans la prochaine sous-section, nous allons mettre l'accent sur les limites des méthodes classiques que nous avons étudiés et comment elles nous incitent à aller vers les méthodes agiles.

2.2.3 Limites des méthodes classiques

Sans être exhaustifs, les principales limites des méthodes classiques peuvent être résumées en trois points :

- Les méthodes classiques se caractérisent par un cycle de vie séquentiel sans rétroaction possible, ce qui signifie une planification exhaustive, stable et définie au début du projet avec une lourde documentation. Les équipes sont spécialisées et gérées par un chef de projet, le changement est fortement résisté et la qualité du produit ne peut être vérifiée qu'à la fin du projet sauf pour les méthodes qui suit un cycle de vie en spiral car celle ci offre des prototypes aux clients.
- D'une part, cette rigidité de l'approche a engendré une limitation du succès des projets et une mauvaise adaptation aux nouvelles exigences du client, ainsi qu'une difficulté lors

de la planification due à l'évolution et au changement constant de l'environnement dans lequel se trouvent les projets.

— D'autre part, la séparation des membres d'équipe selon l'expertise et la commandite du chef d'équipe peuvent conduire à des barrières de communication et à des désaccords dans la définition et la collecte des besoins.

Pour cette raison, les méthodes dites classiques ont été remises en question et les méthodes dites agiles ont pris place.

2.3 Méthodes agiles

Les méthodes agiles sont largement adoptées par la communauté de développement logiciel au cours de la dernière décennie. Cependant, la multitude des méthodes a fait que plusieurs définitions ont été proposées et reportées dans diverses références.

Prenons l'exemple de l'étude [76] qui cite plusieurs définitions dans sa revue de littérature. Entre autres celle de Jin Highsmith [7] qui met l'accent sur l'aspect adaptatif des méthodes agiles à l'aide de la notion d'agilité : *« L'agilité est la capacité de favoriser le changement et d'y répondre en vue de s'adapter au mieux à un environnement turbulent »*. Une autre définition portant sur l'agilité a été donnée par Larman et Vodde [83] : *« une façon de penser et de travailler qui met l'accent sur la livraison continue de valeur, en s'adaptant rapidement aux changements des besoins du client et en impliquant activement les parties prenantes dans le processus de développement »*. Donc, nous pouvons comprendre qu'il faut non seulement travailler en agilité mais y penser également afin d'assurer l'adaptabilité et l'amélioration continue.

Les deux définitions suivantes nous semblent plus complètes :

— Selon [111] : *« 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. »*.

— D'après Scott Ambler [23] : *« Une méthode agile est une approche itérative et incrémentale pour le développement de logiciel, réalisée de manière très collaborative par des équipes responsabilisées, appliquant un cérémonial minimal, qui produisent dans un délai contraint un logiciel de grande qualité répondant aux besoins changeants des utilisateurs »*.

Ces deux définitions décrivent les méthode d'une façon exhaustif puisqu'elles sont basées sur les 4 valeurs principales de Manifeste Agile. Alors que la deuxième définition se limite au domaine de développement logiciel, la première ne exclue pas son application dans d'autres domaines. De plus, la première définition fait mention de l'utilisation de formalismes et la deuxième de l'application de cérémonies.

Enfin, même ces définitions varient légèrement selon les auteurs, la plupart mettent l'accent sur les valeurs fondamentales de l'agilité telles que la communication, la collaboration, la simplicité, la responsabilité, l'amélioration continue, la transparence et l'adaptation. Nous allons maintenant donner quelques exemples de ces méthodes et les détailler pour mieux les comprendre.

2.3.1 Quelques méthodes agiles

Dans ce qui suit, nous allons discuter quelques méthodes agiles à savoir Scrum ,XP , Crystal et Lean. Nous avons choisi de présenter les méthodes Scrum, XP et Lean en nous appuyant sur des statistiques qui prouvent leurs pertinences (Figure 2.4). Ces statistiques sont le résultat d'un sondage mené par Digital.ai en 2022 et mentionné dans leur rapport *"the 16 state of agile"* [1].

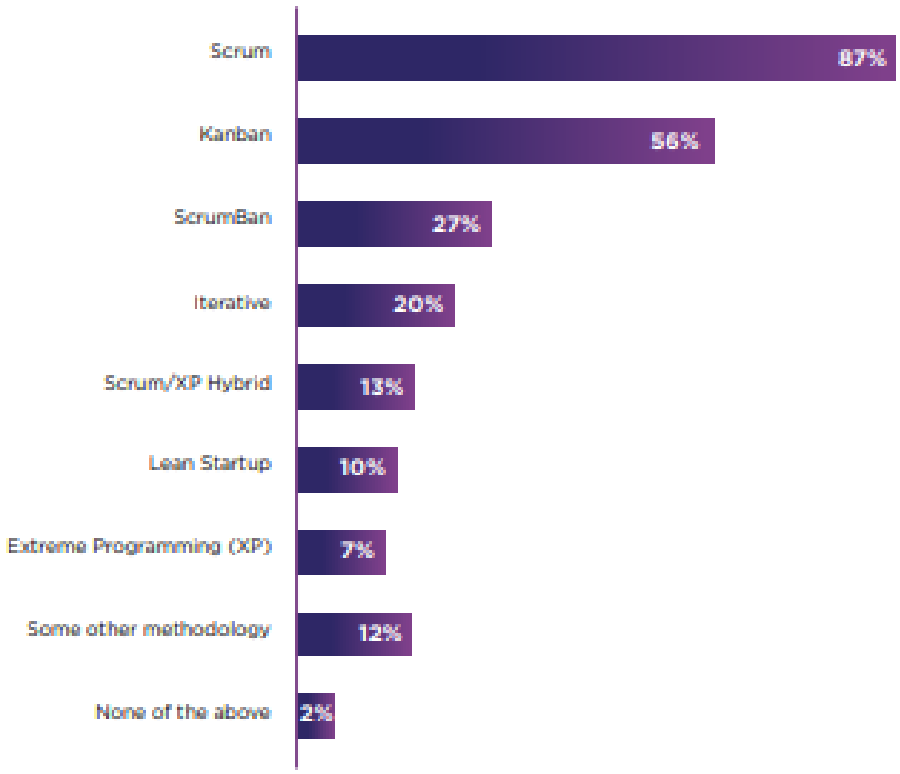

FIGURE 2.4 – Comparaison d'utilisation des méthodes agiles [1].

La figure 2.4 montre que la méthode Scrum est largement utilisée, voire la plus utilisée, tandis que Lean et XP sont également populaires mais moins utilisés que Scrum. Cependant, notre choix d'inclure la méthode Crystal a été basé sur notre expérience personnelle dans la maîtrise et l'application de cette méthode. Nous pensons qu'elle mérite d'être examinée car elle offre des avantages intéressants dans certains contextes de développement logiciel.

Méthode SCRUM

La méthode Scrum a été développé au début des années 1990 par ken schwaber et jeff sutherland et depuis elle est devenu de plus en plus utilisé. Elle a été défini comme étant un cadre pour les éléments qui seront utilisés dans le processus appliqué dans la réalisation[128] [123].

Les rôles dans Scrum sont définis comme suit :

1. **Scrum Master :** son rôle principal est de faciliter la mise en œuvre de la méthode Scrum et de veiller à ce que l'équipe respecte les principes et les pratiques de Scrum. Il est responsable de maintenir un environnement de travail agile, de gérer les obstacles et les problèmes qui peuvent entraver la progression du projet, et de promouvoir la collaboration et la communication efficace au sein de l'équipe. Le Scrum Master veille également à ce que les réunions Scrum, telles que les réunions de planification, les revues de sprint et les rétrospectives, soient organisées et productives.
2. **Équipe de développement :** elle est composée de 2 à 7 membres. Ils collaborent étroitement pour la réalisation du projet dans toutes ses phase pour garantir que la conception et l'implémentation de l'application répondent aux exigences de client et aux normes de qualité.
3. **Product Owner :** il définit la vision du produit, établit les priorités des fonctionnalités à développer et collabore étroitement avec l'équipe de développement pour s'assurer que les besoins et les exigences de client sont pris en compte dans l'application. Le Product

Owner est responsable de la validation des fonctionnalités développées et de la garantie que l'application répond aux objectifs et aux attentes énoncées.

En s'appuyant sur les explications données par les travaux [111] [128] [123] et à l'aide de la figure 2.5, nous allons présenter le déroulement de cycle de vie de la méthode Scrum.

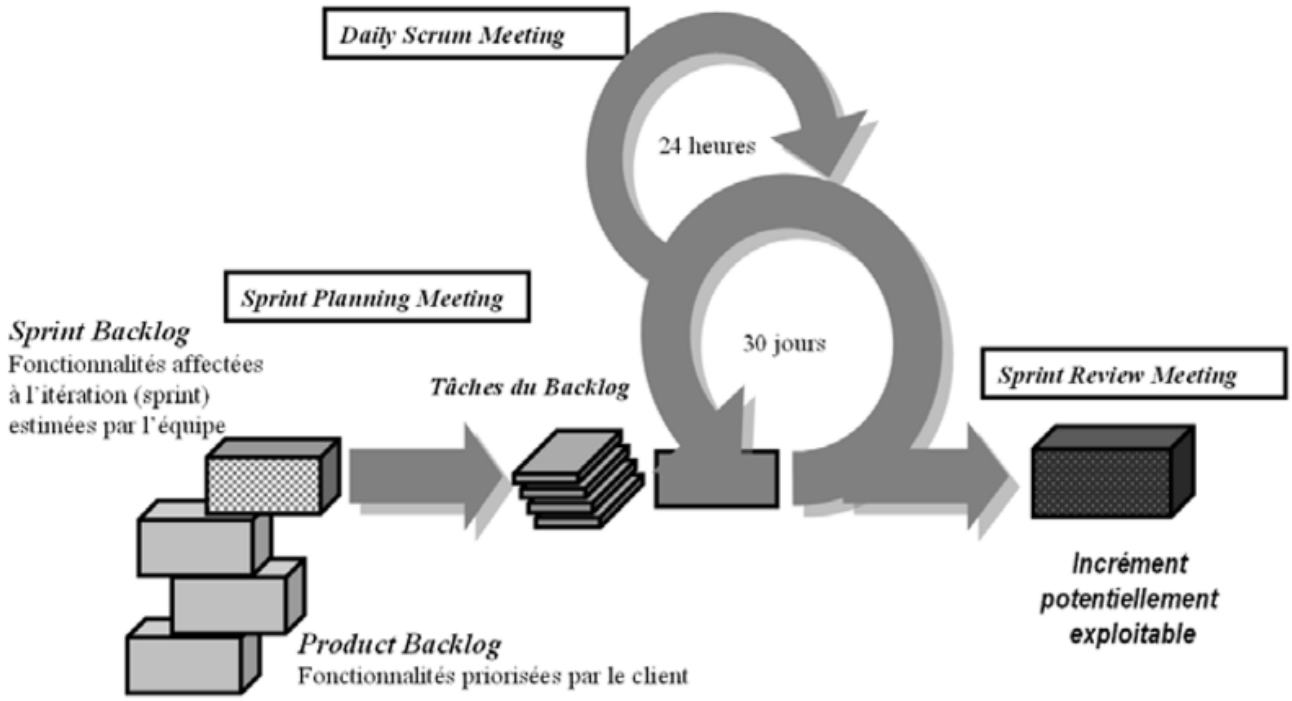

FIGURE 2.5 – Cycle de vie de la méthode Scrum [111].

1. Le **product owner**, qui le propriétaire du produit à réaliser, exprime ces besoins à partir desquels sont définis les fonctionnalités à réaliser et qui seront classées dans le **product backlog** selon la priorité de réalisation comme montré sur la figure 2.5.
2. Au début de chaque **sprint** (ou itération), un **sprint plan meeting**, qui est une réunion de planification, aura lieu dans lequel se fait la sélection des tâches du **sprint backlog** qui est une partie dans le **product backlog**.
3. Au cours d'un sprint qui dure jusqu'à 4 semaines, chaque jour il y aura un **sprint daily meeting** qui dure 15 minutes pour discuter l'avancement du **sprint** en question.
4. A la fin d'un **sprint**, se déroule le **sprint review meeting**, qui dure au maximum 4 heures, dans lequel est faite la démonstration d'un **release** qui est une version fini de produit au prés de client.
5. Pour conclure un **sprint**, un **sprint rétrospective**, qui dure à peu prés 3 heures de temps, aura lieu pour discuter les éventuelles améliorations qui peuvent être effectuées dans l'application de la méthode Scrum.

Méthode XP

La méthode XP "Extreme Programming" a été créee par Kent Beck, Ward Cunningham et Ron Jeffries à la fin des années 1990. Cette méthode a pour but de livrer un logiciel de qualité et de s'adapter aux besoins du client. Elle a comme valeur la simplicité, le feedback, le courage, le respect et la communication [26] [16] [140] [27].

Pratiques de la méthode XP
La méthode XP se caractérise par 13 pratiques qui sont réparties en 3 catégories [78] [111]. Nous allons les définir en s'inspirant de [111] et [9] :

1. **Pratiques de programmation :**
   - **Conception simple :** concevoir des solutions simples pour faciliter l'évolution dans les prochaines itérations.
   - **Remaniement (Refactoring) :** il s'agit de améliorer le code sans affecter le fonctionnement.
   - **Développement piloté par les test unitaires :** des test automatisées seront planifiés avant de commencer la programmation.
   - **Test de recette :** le client participe dans la rédaction des tests de recette, qui seront déclenchés automatiquement à la fin de chaque itération.
2. **Pratiques de collaboration :**
   - **Programmation en binôme (pair programming) :** les développeurs travaillent à deux pour produire un code de qualité et détecter mieux les erreurs.
   - **Règles de code :** l'équipe établit leurs propres règles de code.
   - **Métaphore :** il recommander que les développeurs décrivent le système et les fonctionnalités avec leur peur propre métaphore pour encourager la communication.
   - **Propriété collaboratif :** tous les membres de l'équipe doivent intervenir dans le processus de développement.
   - **Intégration continue :** intégration régulière pour détecter les anomalies le plus tôt possible.
3. **Pratiques de gestion de projet :**
   - **Client sur site :** implication étroite de client dans le processus de développement.
   - **Séance de planification (planning game) :** le client et l'équipe de développement réalise ensemble des planifications tout au long du projet.
   - **Livraison fréquentes :** l'équipe livre fréquemment des versions du logiciel au client pour assurer qu'il correspond à ses attentes.
   - **Rythme soutenable :** l'équipe doit développer un rythme de travail qui est à la fois efficace, rapide et garantit le confort des membres de l'équipe.

Fonctionnement de la méthode XP
La méthode XP est centrée sur la programmation, son fonctionnement est expliqué à l'aide de la figure 2.6 et les détails fourni par [13].

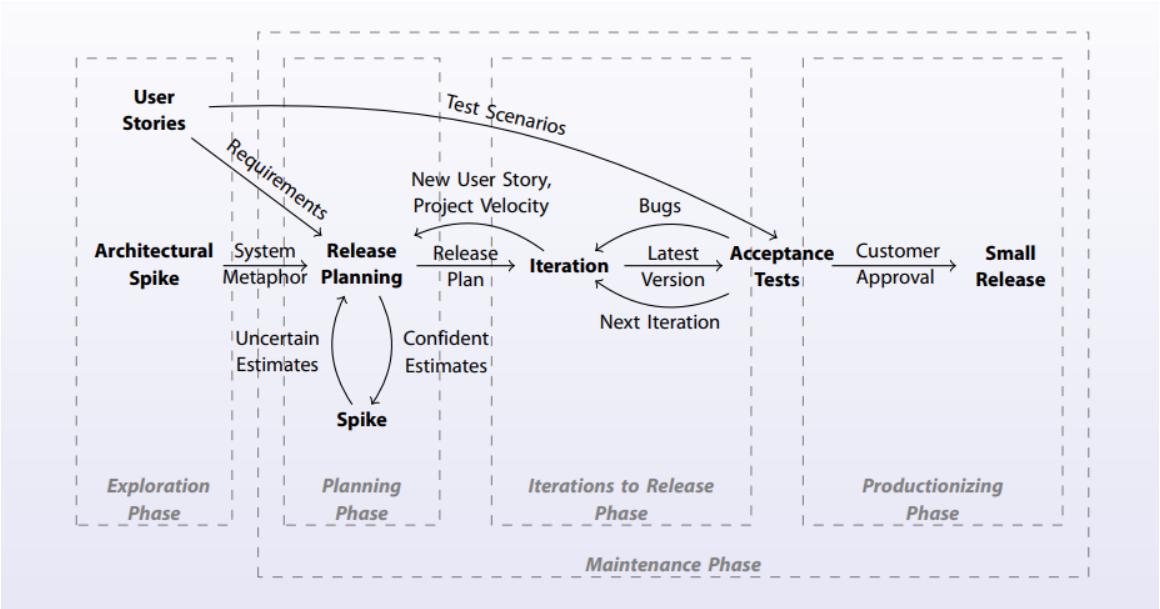

FIGURE 2.6 – Cycle de vie de la méthode XP [103].

1. **Phase d'exploration** (exploration phase) : dans cette phase le client et l'équipe de développement collaborent pour établir les scénarios d'utilisateur (user stories) qui incluent les besoins du client et les scénarios des tests. Ils étudient ainsi la fusibilité de ces scénarios.
2. **Phase de planification** (planning phase) : cette phase consiste à déterminer le plan de livraison, à diviser les fonctionnalités en itérations et à définir les objectifs et les délais de chaque itération.
3. **Phase d'itérations jusqu'à livraison**(iterations to release phase) : les développeurs travaillent en binômes pour coder les tâches qui leurs sont assignées, puis ils effectuent des tests d'acceptation avec les utilisateurs. En cas d'échec, les scénarios sont révisés et la boucle est recommencée. Sinon, le processus se poursuit jusqu'à ce que tous les scénarios retenus soient développés.
4. **Phase de production** (productionizing phase) : dans cette phase une version finie (release) est livrée au client avec sa documentation.
5. **Phase de maintenance** (maintenance phase) : cette phase aura lieux si le client signale des problèmes ou propose des améliorations après avoir utilisé le produit, ce qui engendre une répétition des phases (planification-itérations jusqu'à livraison-production) mais généralement ça sera plus coûteux en durée.

Méthode Lean software development

Le modèle de développement logiciel "Lean" est inspiré du système de production Toyota, qui vise l'élimination radicale du gaspillage et l'amélioration de la satisfaction client. Les principes du Lean management ont été adaptés au développement informatique et sont décrits pour la première fois par Mary et Tom Poppendieck dans leur livre "Implementing Lean Software Development : From Concept to Cash" [105].

Cependant, il est important de noter que le développement Lean ne se limite pas à une méthodologie précise, mais plutôt à une philosophie de management basée sur des principes industriels [78].

Selon [70] la méthode Lean se décline en sept principes que nous allons expliquer en s'inspirant de [102] et [37] :

- ◇ **L'élimination des gaspillages :** c'est le principal fondement de Lean. Selon taiichi ohno [105] le fondateur de Toyota production système, le gaspillage est tous ce qui ne produit pas de valeur au client.
- ◇ **L'amélioration d'apprentissage :** en essayent a chaque fois de satisfaire les attentes de client grâce au feedback et itération.
- ◇ **Le retard de l'engagement :** prendre les décisions en retard (surtout les décisions irréversible, qui ne peuvent pas être modifiées ultérieurement), c'est-à-dire ne pas prendre des décisions jusqu'à ce que les attentes du client sont bien comprises pour éviter les incertitudes.
- ◇ **La livraison aussi vite que possible :** pour ne pas faire patienter le client et prouver l'avancement dans le travail et aussi pour récupérer le retard engendré qui est mentionné dans le point précédent.
- ◇ **L'octroi de pouvoir à l'équipe :** crée un environnement motivant et qui encourage l'autonomie de l'équipe de développement.
- ◇ **L'intégration de la qualité dès la conception :** grader le client informé sur l'état de travail, garder le travail simple et clair, tester, et améliorer continuellement pour assurer la bonne qualité.
- ◇ **La considération du produit dans sa globalité :** le fait que la méthode découpe le travail en sous fonctionnalités ne doit pas restreindre l'équipe de voir le produit globalement, c'est-à-dire ne pas focaliser sur le détail et négliger le tous.

Méthode Crystal

Créée par Alistair Cockburn en 1995 [42], la famille de méthodologie "Crystal" est conçue pour améliorer la communication entre les membres de l'équipe de développement et accélérer la vitesse de livraison des solutions technologiques et elle comprend plusieurs variétés, telles que la méthode "Crystal Clear", "Crystal Orange" et "Crystal Red". Le choix de la méthode dépend du contexte d'application.

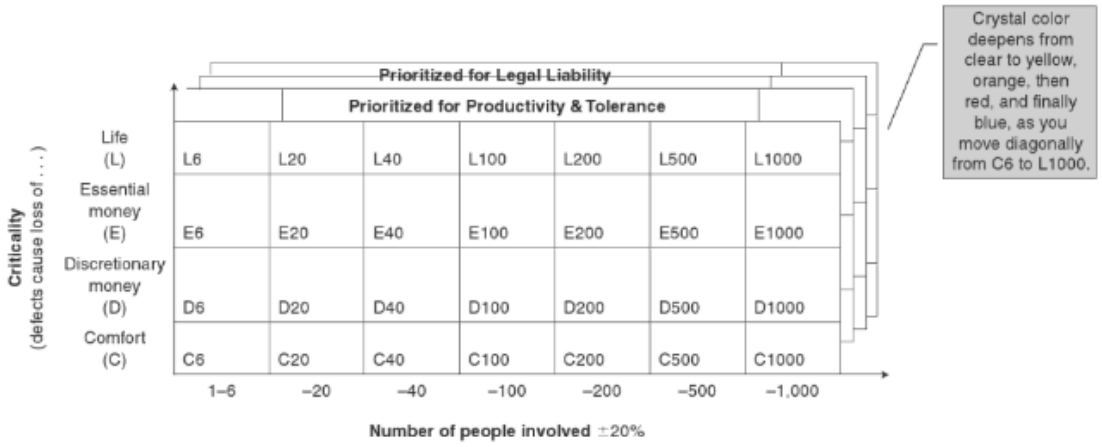

FIGURE 2.7 – Processus d'implémentation de la méthode Crystal [42].

La figure 2.7 montre comment la méthode utilise un continuum de couleur pour sélectionner le nombre de personnes nécessaires pour l'équipe de projet et la "lourdeur" du projet. Plus la couleur est foncée, allant de clair C8 comme la plus légère à la plus profonde L1000 comme la plus lourde, plus il sera nécessaire d'avoir des membres d'équipe et un processus plus lourd pour

le projet. De plus le continuum de dureté sur l'axe verticale de la figure représente la sévérité des dommages si le système ne fonctionne pas correctement.

Au lieu de fournir un ensemble spécifique de lignes directrices pour chaque couleur, Cockburn utilise des études de cas pour illustrer les approches courantes pour une couleur spécifique. Cependant, il existe des valeurs et des règles spécifiques applicables à toutes les méthodes Crystal [21]. Les principales propriétés de Crystal sont [41] :

- Livraison fréquente.
- Communication osmotique (communication étroite et souple).
- Amélioration réfléchie.

2.3.2 Caractéristiques communes aux méthodes agiles

Les principales caractéristiques des méthodes agiles que nous avons constatées sont les suivantes :

- **Itérative et incrémentale :** le projet est découpé en phases (itérations), à la fin de chaque itération, une version fonctionnelle du produit est livrée au client pour sa validation [108][7].
- **Esprit collaboratif :** les méthodes agiles valorisent le travail en équipe et la communication entre les différents acteurs du projet notamment avec le client [2][10][108][45][7].
- **Formalisme léger :** les méthodes agiles ne nécessitent pas autant de documentation que les méthodes classiques. Les outils qu'elles utilisent sont plutôt des outils qui aident à accomplir la tâche en un délai minimal [7].
- **Produit de haute qualité :** grâce à la livraison de valeur en continu et au feedback permanent reçu, tout défaut sera détecté et corrigé immédiatement, ce qui en résulte un livrable de qualité [10][108][2].
- **Acceptation du changement :** contrairement aux méthodes classiques qui restent conformes à un plan initial, les méthodes agiles acceptent les changements demandés par le client tout au long du projet [108][7][45]

.

2.3.3 Tableau comparatif des méthodes étudiées

Le tableau 2.1 présente une comparaison entre les 4 méthodes agiles que nous avons déjà étudiées.

méthode agile	scrum	XP	Crystal	lean
Principes de base	Transparence, Inspection, Adaptation	Communication, Simplicité, Feedback, Courage	Communication, Collaboration, Livraison fréquente, Amélioration continue	Amélioration continue, Collaboration et autonomie, Élimination des gaspillages
Rôles	Product Owner, Scrum Master, Équipe	Coach de l'équipe, Programmeur, Client	Propriétaire du projet, Conseiller technique, Équipe	Le responsable, Le propriétaire du produit, L'équipe de développement, Les parties prenantes
Cérémonies	Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective	Planning du jeu, Suivi, Conception, Tests, Revue	Retrospective du projet, Conférence technique, Livraison	Les marches Gemba, Les événements Kaizen, Le juste-à-temps (JIT), L'amélioration continue
Livrables	Backlog de produit, Sprint Backlog, Incréments	Fonctionnalités fonctionnelles, Tests automatisés	Versions, Codes de bonne qualité, Tests réguliers	MVP, des prototypes et des tableaux de bord.
Avantages	Facilité d'adaptation, Meilleure visibilité, Renforcement de l'équipe	Qualité du code élevée, Meilleure communication, Livraison plus rapide	Flexibilité, Réduction des coûts, Communication améliorée	Amélioration de la qualité, Flexibilité, Collaboration
Inconvénients	Peut être difficile à comprendre, Peut nécessiter un changement de culture	Les nouvelles pratiques peuvent être difficiles à adopter	Dépendant de la compétence de l'équipe, Risque de dépassement de délais	Manque de structure, Difficulté à prédire les délais, Nécessité d'une expertise technique

TABLE 2.1 – Tableau comparatif des méthode Scrum, XP, Crystal et Lean

A travers cette comparaison, nous constatons que Scrum met l'accent sur la communication et le développement itératif, XP se concentre sur l'excellence technique, Crystal est adaptée à différents projets et équipes, et Lean met l'accent sur la valeur client et la réduction des gaspillages. Et nous remarquons aussi que certain méthode sont plus strictes que les autres en termes de structure par exemple Scrum respecte des rituels et des rôles définis tandis que Lean est beaucoup plus moins structurée.

2.3.4 Avantages et Inconvénients des méthodes agiles

Comme toutes les méthodes, des avantages et des inconvénients peuvent être mis en évidences pour les méthodes agiles.

Les avantages :

Les avantages des méthodes agiles dans le domaine du développement sont les suivants :

- Comme les méthodes agiles sont itératives et impliquent le client, le changement au cours du projet est possible, ce qu'on appelle l'adaptabilité.
- Meilleure détection des risques.
- Livraison rapide de logiciels utiles et de qualité.
- Communication efficace de l'équipe.

Les inconvénients :

Malgré que ces méthodes sont les plus utilisées actuellement et offre plusieurs avantages, elles présentent également certains inconvénients. Voici quelques uns :

- Chaque groupe impliqué doit avoir une bonne communication ou bien une communication transparente entre ses membres, sinon le projet échouera.
- La difficulté d'appliquer le principe de réunion d'équipe lors de la réalisation en raison de la proximité géographique des membres.
- Très peu de projets sont développés intégralement avec l'agile [31].
- Difficulté de garantir la transparence dans des grands projets.
- La mise en cause du rôle des managers et des chefs de projet, car ces derniers deviennent plutôt des coachs ou ambassadeurs.

Maintenant que nous avons étudié les méthodes classiques et les méthodes agiles, nous constatons que les deux types ont des avantages et des inconvénients. Cela fait penser à une autre solution qui est le juste milieu, celle-ci est les méthodes hybrides et que nous allons détailler dans la prochaine section.

2.4 Méthodes hybrides

D'après [36], la méthode hybride est celle qui combine deux méthodes deux ou plus pour en tirer le meilleur de ces derniers. L'hybridation des méthodes agiles est le concept le plus courant comme par exemple la méthode Scrumban [81]. Mais il reste possible d'avoir des méthodes qui combine les types agile et classique pour bénéficier de la souplesse d'agile et de la rigueur de planification des méthodes classiques, un exemple de ces méthodes est Agile Waterfall [61].

Avantages des méthodes hybrides :

La compositions de deux ou plusieurs méthodes de développement a permis au méthodes hybrides d'avoir beaucoup d'avantages qui sont [144] :

- Économiser le budget.
- Un court délai de planification.
- Améliorer la collaboration.
- Augmenter la qualité de produit.
- Une meilleure adaptation aux besoins du client.

Inconvénients des méthodes hybrides :

Malgré que ces méthodes ont beaucoup d'avantages, elles ont aussi quelques inconvénients, entre autres :

- Complexité d'implémentation.
- Nécessite de l'expertise dans plusieurs méthodes.
- Risque de conflit entre les méthodes combinées.

Conclusion

En conclusion, l'analyse de l'état de l'art des méthodes de développement nous a permis de distinguer différents types de méthodes, plus spécialement les méthodes classiques et les méthodes agiles. Nous avons introduit quelques méthodes agiles existantes et leurs caractéristiques distinctes, notamment Scrum, XP, Lean et Crysatl pour comprendre leurs fonctionnements. En outre, nous avons examiné les méthodes hybrides qui combinent des éléments des méthodes classiques et agiles. En gros, cette étude de l'état de l'art a montré que les méthodes agiles ont évolué en réponse aux limitations des méthodes de développement classiques et continuent d'être utilisées de manière croissante dans les projets de développement de logiciels en raison de leur succès due leur flexibilité et adaptabilité.

Le chapitre suivant va porter sur l'adaptation de la méthode agile le plus utilisée actuellement Scrum pour le développement des systèmes intelligents.

Chapitre 3

Adaptation de la méthode Scrum au développement des systèmes intelligents

Introduction

L'automatisation des systèmes d'information dans les entreprises est une évidence mais l'extension de ces derniers aux systèmes logiciels intelligents l'est encore plus. D'une part, l'étude de l'état de l'art des méthodes de développement, faite dans le chapitre précédent, révèle la disponibilité d'une multitude de méthodes et le choix de l'un d'elles dépend fortement des besoins et de la solution attendue. D'autre part, aucune méthode ne prend en compte les caractéristiques et les propriétés des systèmes intelligents identifiées dans le premier chapitre. Comme conséquence, le développement de systèmes intelligents est confronté aux défis spécifiques tels l'acquisition de connaissances, la formalisation des données, l'apprentissage et l'amélioration continuelle des modèles.

Pour relever ces défis, dans ce chapitre nous allons proposé une amélioration de la méthode Scrum pour l'adapter au développement des systèmes intelligents. Nous allons tout d'abord poser la problématique, ensuite introduire notre proposition, puis justifier le choix des éléments de l'adaptation, enfin nous allons expliquer d'une manière détaillée la méthode proposée.

3.1 Problématique

L'évolution des méthodes de développement logiciel a permis d'améliorer le taux de réussite de projets informatiques [1]. Spécialement, l'utilisation des méthodes agiles, telle que Scrum, a démontré leur efficacité en favorisant la flexibilité, la collaboration et l'adaptabilité aux changements. Cependant, l'état de l'art révèle que les méthodes agiles, les plus évoluées des méthodes existantes, ne sont pas directement adaptées aux défis particuliers posés par les systèmes intelligents.

En effet, les systèmes intelligents, comme ceux basés sur l'apprentissage automatique, présentent des caractéristiques spécifiques qui nécessitent une approche différente pour leur développement. Ces caractéristiques incluent l'acquisition de connaissances et leur formalisation, l'apprentissage et l'amélioration continuelle des modèles, l'adaptation aux ensembles de données massifs en constante évolution, la gestion des algorithmes complexes, ainsi que l'intervention des expertes de domaines et des cogniticiens.

Par conséquent, certains questions se posent sur la procédure à suivre pour développer un systèmes intelligent qui sont les suivantes : comment adapter les méthodes agiles, puisqu'elles ont prouvé leur fiabilité, pour répondre aux spécificités du développement des systèmes intelligents ? Comment intégrer la phase d'apprentissage, gérer les ensembles de données évolutifs

et s'adapter aux évolutions constantes des modèles pour obtenir des résultats de haute qualité dans un contexte agile ? Comment les experts du domaine et les cogniticiens travailleront-ils avec les membres de l'équipe agile ? A quelle phase du cycle de vie vont-ils intervenir ?

Dans les sections suivantes, nous présentons notre proposition d'adaptation et d'amélioration de la méthode Scrum pour répondre aux questions posées en justifiant le choix de la méthode Scrum et de la méthodologie proposée, ainsi que les avantages attendus de cette amélioration.

3.2 Proposition d'adaptation de la méthode Scrum

Notre contribution consiste à améliorer le cycle de vie de la méthode Scrum en ajoutant des cycles d'apprentissage. Notre proposition vise à répondre aux besoins spécifiques du développement des systèmes intelligents basés sur l'apprentissage automatique en intégrant des itérations d'apprentissage, les experts du domaines et les cogniticiens dans le processus comme le montre la figure 3.1.

Les itération d'apprentissage sont essentielles pour le développement des systèmes intelligents, car ils permettent d'améliorer continuellement les modèles et les algorithmes utilisés. Dans le contexte de l'apprentissage automatique, il est souvent nécessaire de s'adapter aux données, de les analyser, de former des modèles et d'évaluer leurs performances. Les itération d'apprentissage permettent de répéter ce processus itératif afin d'obtenir des résultats de plus en plus précis et pertinents.

En ajoutant des itérations d'apprentissage à la méthode Scrum, nous offrons plusieurs avantages. Tout d'abord, cela permet une adaptation plus rapide aux changements et aux évolutions des données grâce à l'intervention de l'expert du domaine. Souvent les systèmes intelligents sont confrontés à des données en évolution constante et les itérations d'apprentissage permettront de mettre à jour les modèles en fonction de ces nouvelles données.

De plus, dans les itérations d'apprentissage, le cogniticien en concertation avec l'équipe développement procède à la modélisation de la connaissance ce qui permet d'améliorer l'efficacité du développement en évitant les erreurs coûteuses à long terme. En itérant régulièrement sur les modèles conçus les erreurs peuvent être détectées plus tôt et corrigées rapidement, ce qui permet d'économiser du temps et des ressources.

Enfin, l'ajout des itérations d'apprentissage contribue à une meilleure qualité des résultats obtenus. Itérer avec un modèle d'apprentissage en changeant ses paramètres permet au modèle d'apprendre et de s'adapter aux données. En outre, si un modèle ne donne pas satisfaction après évaluation, d'autres modèles peuvent être intégrés dans une nouvelle itération. Ainsi, il est possible d'obtenir des prédictions plus précises et de meilleures performances globales du système intelligent.

La figure 3.1 montre le cycle de vie correspondant à la méthode proposée.

3.3 Justification de notre adaptation

Dans cette section, nous fournissons des justifications solides de notre choix d'adapter la méthode Scrum pour le développement des systèmes intelligents basés sur l'apprentissage automatique. Nous mettons en évidence les spécificités de l'apprentissage automatique et les défis auxquels les développeurs de systèmes intelligents sont confrontés. En outre, nous soulignons les avantages significatifs et les bénéfices potentiels que l'adaptation de Scrum peut apporter dans ce contexte spécifique.

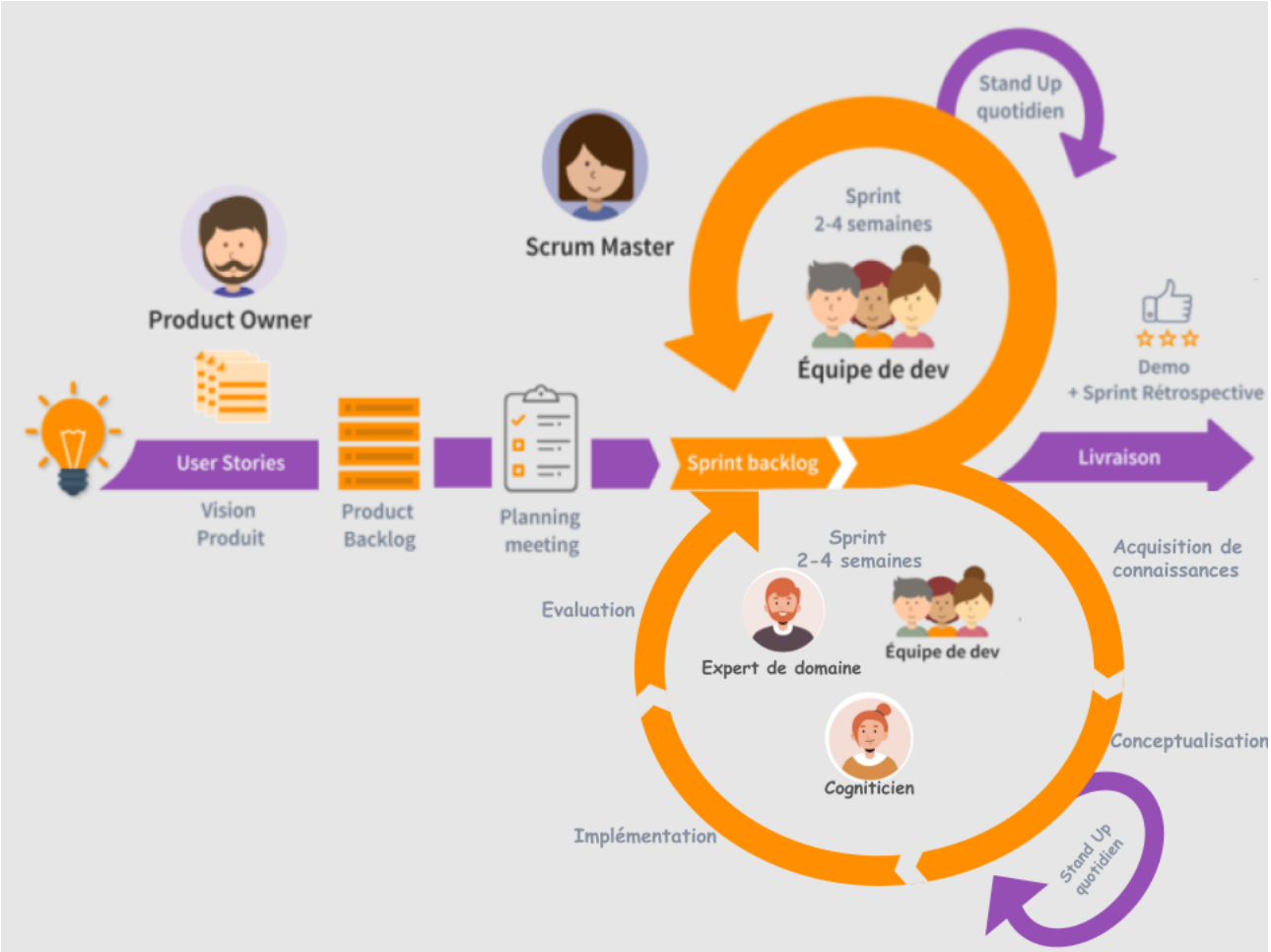

FIGURE 3.1 – Méthode Scrum adapté au développement de systèmes intelligents basés sur l'apprentissage automatique

3.3.1 Spécificités des systèmes logiciels intelligents

L'apprentissage automatique présente des caractéristiques distinctes qui nécessitent une approche de développement agile et itérative telle que Scrum. Dans cette section, nous allons examiner les spécificités de l'apprentissage automatique et expliquer pourquoi une adaptation de Scrum est nécessaire pour le développement des systèmes intelligents.

- **Données non structurées :** Les systèmes intelligents basés sur l'apprentissage automatique traitent souvent des données non structurées telles que des images, des textes ou des signaux audio. Contrairement aux données structurées, ces données nécessitent une approche différente pour leur préparation, leur traitement et leur analyse. L'adaptation de Scrum doit prendre en compte cette spécificité en incorporant des tâches de prétraitement des données non structurées dans la phase d'acquisition de connaissances des itérations du processus [143][84].
- **Données évolutives :** Les données utilisées dans les systèmes intelligents peuvent évoluer au fil du temps. De nouveaux exemples peuvent être ajoutés, des données obsolètes peuvent être supprimées et des modifications peuvent survenir dans la distribution des données. Par conséquent, le modèle d'apprentissage automatique doit être capable de s'adapter à ces changements. L'adaptation de Scrum doit permettre une flexibilité et une adaptabilité accrues pour tenir compte de ces évolutions des données [38][51].
- **Expérimentation et itérations fréquentes :** L'apprentissage automatique implique souvent une phase d'expérimentation et de réitération fréquente. Les modèles et les al-

gorithmes doivent être testés, évalués et ajustés en fonction des performances obtenues. L'adaptation de Scrum doit intégrer des itérations spécifiques pour l'expérimentation et l'amélioration continue du modèle d'apprentissage [28][115].

- **Besoin d'expertise spécialisée :** Le développement de systèmes intelligents nécessite une expertise spécialisée dans le domaine de l'apprentissage automatique et de l'intelligence artificielle en général. Les équipes doivent inclure des experts compétents pour guider le processus de développement et prendre des décisions éclairées. L'adaptation de Scrum doit faciliter la collaboration entre les membres de l'équipe et favoriser une communication étroite entre les experts et les autres membres de l'équipe [63][52].

3.3.2 Défis du développement de systèmes intelligents

Les développeurs de systèmes intelligents sont confrontés à des défis complexes, tels que le choix et le raffinement des modèles d'apprentissage, la gestion des ensembles de données massifs, la sélection des caractéristiques pertinentes et la recherche d'une performance optimale. Ces défis nécessitent une approche itérative où les améliorations incrémentielles peuvent être apportées au fur et à mesure de l'apprentissage et de la compréhension accrue du problème.

3.3.3 Avantages de l'adaptation de Scrum

Le choix de le méthode agile n'est pas fortuit. Elle est la méthode la plus utilisée dans le marché. Selon le rapport établi par Digital.ai en 2022, 78 % des interviewés utilisent cette méthode dans le développement de leurs projets. En adaptant Scrum, nous pouvons bénéficier de nombreux avantages dans le développement de systèmes intelligents qui sont :

— La flexibilité de Scrum permet d'explorer différentes approches et modèles d'apprentissage de manière itérative, ce qui facilite l'ajustement et l'optimisation des modèles en fonction des performances et des retours d'information obtenus.

— De plus, l'approche itérative de Scrum favorise la collaboration étroite entre les membres de l'équipe, y compris les experts du domaines et les cogniticiens. Cette collaboration favorise l'échange d'idées, la résolution de problèmes et la prise de décisions éclairées pour améliorer les résultats obtenus.

— Un autre avantage clé de Scrum est l'amélioration continue. Grâce à des cycles courts, les équipes peuvent rapidement intégrer les nouvelles connaissances acquises et les leçons apprises, ce qui permet d'ajuster et d'améliorer constamment les modèles et les approches utilisés.

3.4 Fonctionnement de la méthode proposée

Dans cette section, nous détaillons le fonctionnement de la méthode proposée pour adapter la méthode Scrum au développement des systèmes intelligents basés utilisant l'apprentissage automatique. Notre approche vise à intégrer de manière cohérente et efficace les itérations d'apprentissage tout au long du processus de développement. Voici les principaux éléments de notre méthode :

1. **Intégration des itérations d'apprentissage :** Nous proposons d'ajouter des itérations spécifiques pour l'apprentissage dans le cadre du cycle de vie de Scrum. Cela permettra de consacrer du temps et des ressources pour l'entraînement, l'évaluation et l'amélioration des modèles d'apprentissage automatique utilisés dans le système intelligent.

2. **Planification des tâches de prétraitement des données :** Le prétraitement des données est une étape cruciale dans le développement des systèmes intelligents. Nous recommandons de planifier ces tâches de prétraitement dans les sprints afin de garantir que les données sont préparées de manière adéquate pour l'apprentissage. Cela peut inclure des activités telles que le nettoyage des données, la normalisation, la sélection des caractéristiques, etc.
3. **Gestion des itérations d'apprentissage :** Nous proposons d'organiser les itérations d'apprentissage en fonction des objectifs spécifiques de chaque sprint. Cela peut impliquer l'entraînement de modèles sur des ensembles de données spécifiques, l'expérimentation avec différents algorithmes ou paramètres, et l'évaluation des performances obtenues. Il est important de définir des critères d'évaluation clairs pour mesurer le progrès et l'efficacité de l'apprentissage.
4. **Communication avec les parties prenantes :** La communication est essentielle dans le développement des systèmes intelligents. Nous recommandons de maintenir une communication régulière avec les parties prenantes tout au long du processus, en partageant les résultats des itérations d'apprentissage, en recueillant leurs feedbacks et en ajustant les objectifs en conséquence. Cela favorise la transparence, la collaboration et l'alignement des attentes.

3.4.1 Phases du sprint d'apprentissage

Comme présenté dans la figure 3.1, la démarche des itérations d'apprentissage se déroule de la manière suivante :

1. **Acquisition de Connaissance**
   - **Objectif :** Dans cette phase, nous nous concentrons sur la collecte de données pertinentes pour notre projet. Cela peut inclure la recherche de sources de données, la collecte de données brutes et la préparation des données pour l'analyse.
   - **Activités :** Nous identifions les sources de données et sur lesquelles se fera la collecte. Nous évaluons également la qualité des données à ce stade. Les experts du domaine interviennent intensivement dans cette phase.
2. **Conceptualisation**
   - **Objectif :** Cette phase vise à préparer les données collectées pour l'entraînement des modèles d'apprentissage automatique. Il s'agit de les rendre exploitables et de les structurer.
   - **Activités :** Nous effectuons des opérations telles que la normalisation des données, la gestion des valeurs manquantes, la transformation des caractéristiques, et la création de jeux de données d'entraînement et de test. Nous pourrions également effectuer une analyse exploratoire des données pour mieux comprendre les données. En outre, il peut s'agir d'utiliser un formalise de représentation de connaissances pour représenter les données et raisonner sur elles. Le cogniticien joue un rôle prépondérant dans cette phase.
3. **Implémentation**
   - **Objectif :** Dans cette phase, nous développons et entraînons nos modèles d'apprentissage automatique en utilisant les données prétraitées.
   - **Activités :** Nous choisissons les algorithmes d'apprentissage appropriés, divisons les données en ensembles d'entraînement et de test, réglons les hyperparamètres des modèles, et les entraînons. Cette phase peut nécessiter plusieurs itérations pour optimiser les modèles.

4. **Évaluation**
    - **Objectif :** L'objectif principal de cette phase est d'évaluer les performances des modèles que nous avons entraînés. Nous devons déterminer si les modèles répondent aux critères de qualité et d'efficacité souhaités.
    - **Activités :** Nous évaluons les modèles en utilisant des métriques appropriées, comparons les modèles entre eux. Si les performances ne sont pas satisfaisantes, nous envisageons de revenir aux étapes précédentes pour apporter des ajustements.

Le cycle de ces quatre phases constitue une itération spécifique à l'apprentissage au sein de notre processus Scrum global. Après chaque itération, nous organisons une réunion de revue pour examiner les résultats, discuter des problèmes et décider des étapes suivantes. Cette approche itérative nous permet d'adapter notre développement à l'apprentissage continu à partir des données et d'ajuster nos modèles en conséquence, avec la participation active de notre équipe de développement, de l'expert de domaine et du cogniticien.

3.4.2 Rôles des intervenants

Les rôles du Scrum Master et du Product Owner restent inchangés dans notre proposition par rapport à Scrum.

1. **Équipe de Développement :** L'équipe de développement est chargée de la mise en œuvre concrète du projet. Elle traduit les concepts, les données et les modèles en code informatique fonctionnel.
2. **Cogniticien :** Le cogniticien apporte une expertise dans la compréhension des processus cognitifs et dans la conception adaptées à la cognition humaine.
3. **Expert de Domaine :** L'expert de domaine apporte une connaissance approfondie du sujet sur lequel porte le projet. Il comprend les enjeux, les problématiques et les besoins spécifiques du domaine concerné.

Conclusion

Dans ce chapitre, nous avons fait une proposition qui vise à améliorer la méthode Scrum en ajoutant des itérations d'apprentissage dans son cycle de vie pour le développement des systèmes intelligents basés sur l'apprentissage automatique. Cette adaptation est essentielle pour répondre aux besoins spécifiques de ces systèmes et pour obtenir des résultats de meilleure qualité. En intégrant des itérations d'apprentissage, nous pouvons adapter Scrum aux données en constante évolution, améliorer l'efficacité du développement et obtenir des prédictions plus précises. Cette approche offre de nombreux avantages, tels que la flexibilité, l'adaptabilité aux changements et l'amélioration de la qualité des résultats. En mettant en œuvre la méthode proposée, les développeurs de systèmes intelligents pourront relever les défis spécifiques de ce domaine et exploiter pleinement le potentiel de l'apprentissage automatique.

Dans le prochain chapitre, nous mettrons en œuvre cette méthode dans une étude de cas concrète, en détaillant les étapes, les outils et les techniques utilisés.

Chapitre 4

Réalisation d'une application de prédiction d'arrêt de pipelines

Introduction

Ce chapitre marque le début de la phase pratique de notre projet, où nous mettons en œuvre notre méthode pour développer une application de prédiction des arrêts de pipelines. Nous commençons par aborder la problématique des arrêts de pipelines et proposons une solution basée sur l'apprentissage automatique. Ensuite, nous décrivons en détail le sprint 0, qui constitue le point de départ du développement de l'application. Nous identifions les acteurs Scrum impliqués dans le projet, estimons les sprints nécessaires pour atteindre nos objectifs et détaillons les outils, bibliothèques et environnements requis pour le développement de l'application.

4.1 Présentation l'organisme d'accueil

L'entreprise "SONATRACH" (société nationale pour le transport et la commercialisation des hydrocarbures) a été créée le 31 décembre 1963 par le décret n° 63/491. Les statuts ont été modifiés par le décret n° 66/292 du 22 septembre 1966, et SONATRACH est devenue "société nationale pour la recherche, la production, le transport, la transformation et la commercialisation des hydrocarbures". Cette évolution a conduit à une restructuration de l'entreprise dans le cadre d'un schéma directeur approuvé au début de l'année 1981 pour une meilleure efficacité organisationnelle et économique. Parmi ses objectifs visés, on peut citer [127] :

— Le renforcement de ses capacités technologiques.

— Le développement international et le partenariat

— La diversification de son portefeuille d'activité.

— La maîtrise continue de ses métiers de base.

Nous nous intéressons dans ce travail à la direction Région Transport Centre (RTC) de SONATRACH, plus précisément au département "Entretien lignes & Bacs de stock", qui gère les oléoducs.

4.1.1 Présentation de la RTC (Région Transport Centre)

La RTC gère l'oléoduc de Haoud el-Hamra vers Béjaia et l'oléoduc vers Béni-Mansour raffinerie d'Alger, elle s'occupe aussi d'entretien des installations et sites de stockage. La coordination des moyens humains et matériels par coupages permanents du brut, gaz et condensat. De plus, elle alimente en gaz la SONALGAZ pour les besoins énergétiques en gaz naturel et

assure le chargement en brut de la raffinerie d'Alger par l'oléoduc Béni-Mansour/Alger. La RTC gère le chargement des navires pétroliers à partir du port pétrolier de Béjaia. L'oléoduc Haoud el Hamra/Béjaia est le premier pipeline installé en Algérie par la société SOPEG (société pétrolière de gérance). Il est d'une longueur de 660 km, d'un diamètre de 24 pouces et d'une capacité de transport de 17 millions de tonnes par an (MAT) de pétrole brut et de condensat vers le terminal marin à la raffinerie d'Alger. La figure 4.1 représente l'organigramme de la RTC qui est constituée des différents départements regroupés en sous-directions [127].

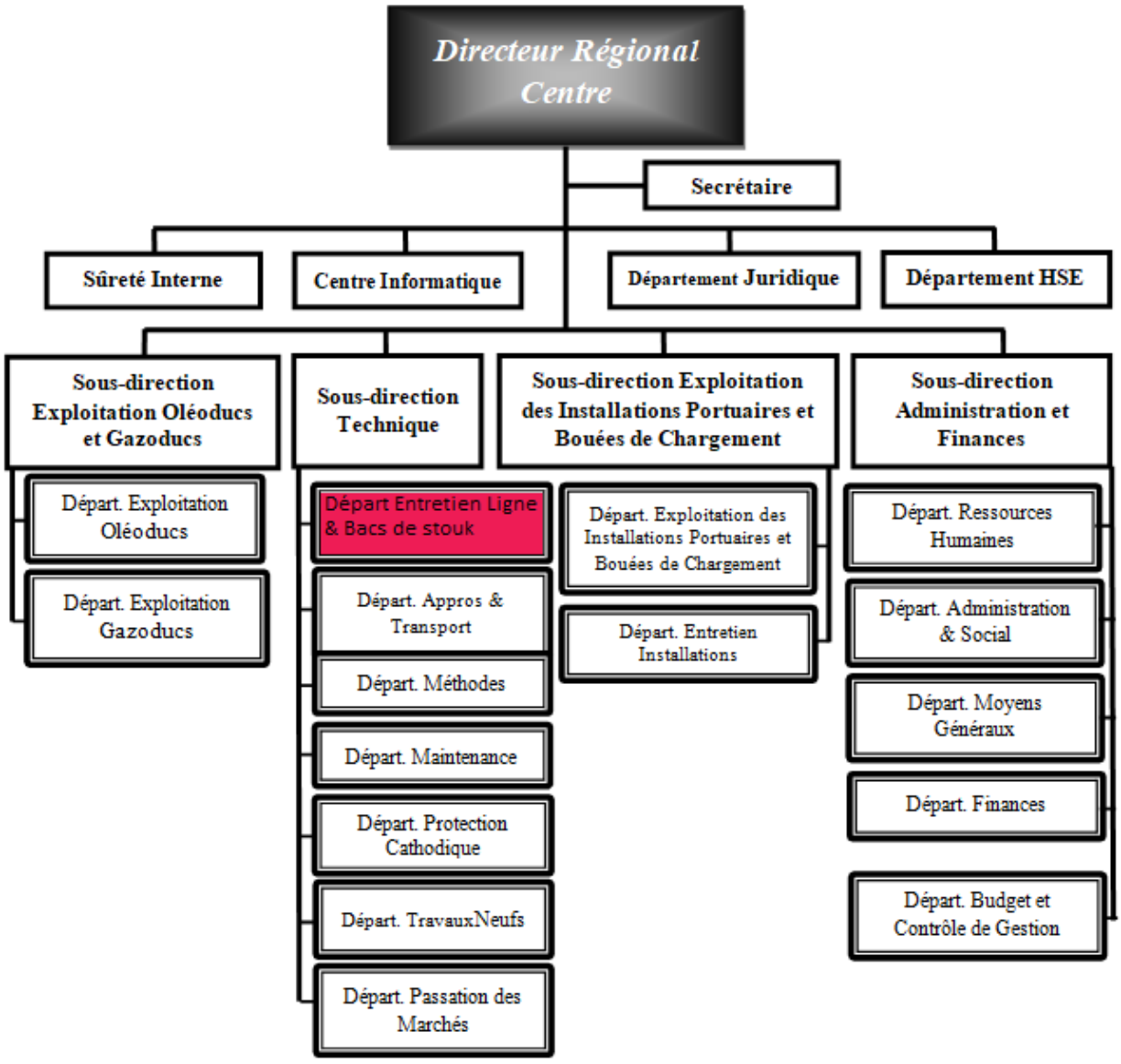

FIGURE 4.1 – Organigramme de la RTC [127]

4.1.2 Présentation du département entretien lignes & bacs de stockage

Il se compose de deux services [127] :

— service entretien lignes.

— service entretien bacs et auxiliaires.

Notre étude a été faite au sein du département "entretien lignes" que nous allons détailler ci-dessous.

Présentation du service entretien lignes

Le service entretien lignes a une mission principale comme l'indique son nom est d'entretenir les lignes de transport des hydrocarbures OB1, DOG1, GG1. Et pour réaliser sa mission il programme une inspection régulière des lignes, l'assistance des travaux exécuter dans les limites de sécurité afin d'assurer le respect des règles de sécurité et la procédure de travail aussi assurer le bon fonctionnement des engins et les équipements d'intervention en cas d'urgence, assurer la formation du personnel de service dans les différents domaines [127].

Composition du service entretien lignes

Ce service est composé de [127] :

— Chef de service.

— Des ingénieurs travaux.

— Sept secteurs travaux.

— Une section d'intervention.

— Une section de maintenance engins

La structure du service d'entretien lignes est représentée dans la figure 4.2 :

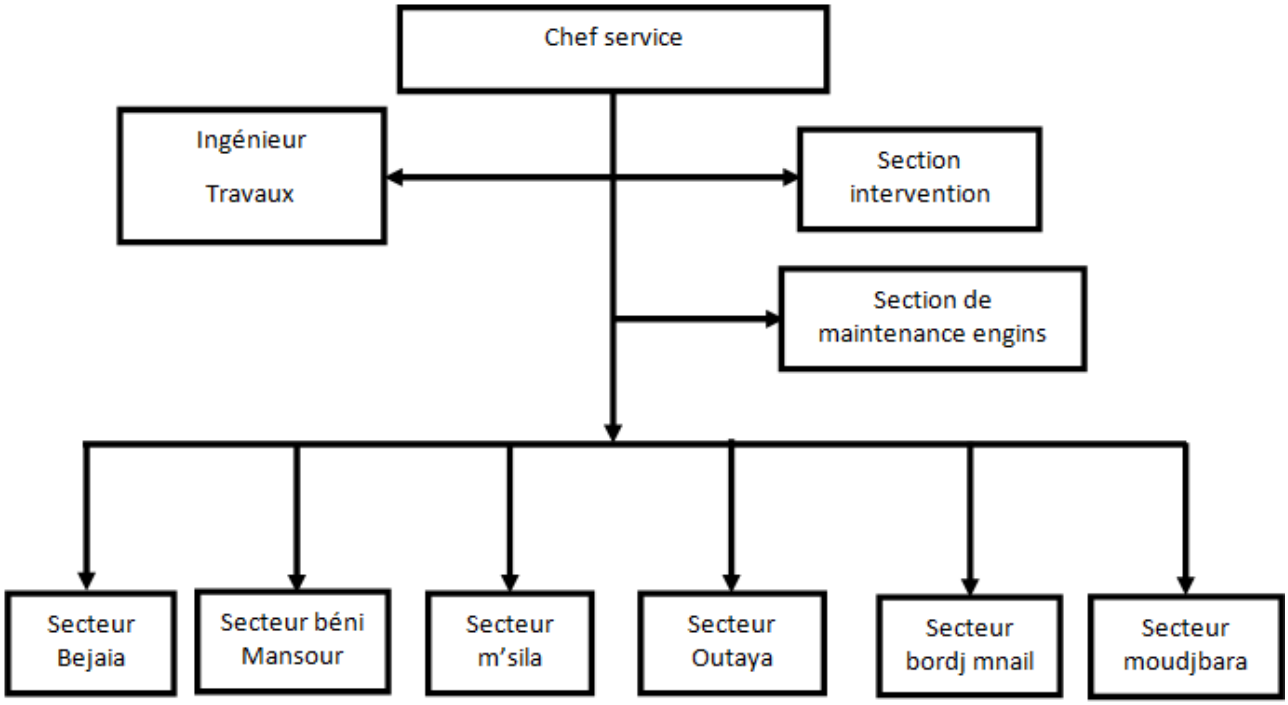

FIGURE 4.2 – Organigramme de département entretien ligne [127]

4.2 Problématique

Les arrêts imprévus des pipelines représentent un défi majeur dans l'industrie du transport de fluides, entraînant des conséquences graves sur l'environnement, la sécurité et les opérations. Ils peuvent entraîner des pertes financières importantes et des perturbations dans l'approvisionnement en énergie.

Dans le cadre de notre stage au sein du département d'entretien lignes & bacs de stockage de Sonatrach, nous avons identifié plusieurs problèmes liés à la gestion des arrêts de pipelines :

- L'incapacité de prévoir les arrêts des pipelines de manière proactive, ce qui entraîne des interruptions non planifiées et des pertes de productivité.
- L'utilisation de méthodes traditionnelles pour le suivi et l'enregistrement des incidents, qui peuvent être inefficaces et manquent de précision.
- L'absence d'un système de gestion centralisé des missions et des interventions, ce qui peut entraîner des retards et une coordination inefficace.

Ces problèmes ont suscité notre recherche d'une solution efficace pour la gestion des arrêts imprévus des pipelines.

4.3 Solution proposée

Face aux problèmes précédemment mentionnés, nous proposons de développer un système d'aide à la décision qui utilise des techniques d'apprentissage automatique et l'analyse des données historiques pour prédire les arrêts potentiels des pipelines. Pour entraîner notre modèle, nous utiliserons un jeu de données provenant de Kaggle[8], en raison de la confidentialité des données de l'entreprise. Cette approche permettra une gestion proactive des arrêts, une amélioration de la planification des interventions et des missions, ainsi qu'une réduction des perturbations dans l'approvisionnement en énergie. En mettant en œuvre cette solution, notre objectif est d'accroître la fiabilité des pipelines et d'optimiser les opérations de transport de fluides.

4.4 Sprint Planification

Le sprint 0 est une phase de préparation essentielle pour démarrer le projet Scrum de manière efficace. Elle permet de poser les bases nécessaires, de définir les objectifs et de préparer l'équipe pour les sprints itératifs suivants.

4.4.1 Spécification des besoins

Afin d'établir le product backlog nous allons recueillir les besoins de notre application.

Besoin fonctionnels

Les besoins fonctionnels désignent comment l'application doit fonctionner. Donc voici ceux de notre application :

— **Ajouter un incident :** permet à l'utilisateur d'ajouter un nouvel incident sur le pipeline.

— **Visualiser les lieux des incidents :** l'utilisateur peut visualiser les lieux des incidents sur le pipeline. Il peut accéder à une représentation des lieux des incidents, mettant en évidence leur localisation géographique.

— **consultation l'historique des incidents :** permet à l'utilisateur de visualiser l'historique des incidents sur le pipeline. L'utilisateur peut accéder à une liste détaillée des incidents passés.

— **Prédiction :** fournir à l'utilisateur des prédictions sur l'arrêt des pipelines dans les incidents futurs.

— **Planification de missions :** permet au chef de service de planifier des missions liées à la maintenance et à l'entretien des pipelines. L'utilisateur peut définir les détails de la mission, y compris la date, l'heure et les assigner au ingénieur.

— **Gestion des utilisateurs :** fournit à l'administrateur de l'application la capacité de gérer les utilisateurs. Cela inclut l'ajout et la suppression de comptes utilisateurs.

— **Authentification :** pour que l'utilisateur puisse accéder à l'application avec son mot de passe et son nom.

Besoin non fonctionnels

les besoins non fonctionnels décrivent les contraintes et les exigences non liées à la fonctionnalité du système, parmi lesquelles nous allons principalement respecter celles-ci :

— **Sécurité :** l'application doit garantir la confidentialité, l'intégrité et la disponibilité des données. Elle doit également permettre la gestion des accès aux données en fonction des rôles et des privilèges des utilisateurs.

— **Performance :** l'application doit être capable de traiter rapidement de grandes quantités de données pour répondre aux besoins opérationnels de l'entreprise. Elle doit également être résistante aux pannes pour assurer une disponibilité maximale.

— **Ergonomie :** l'interface utilisateur doit être intuitive et facile à utiliser pour permettre aux utilisateurs de naviguer facilement dans l'application, de trouver les informations dont ils ont besoin et de réaliser les tâches de manière efficace.

— **Extensibilité :** l'application doit être capable de s'adapter à l'évolution des besoins de l'entreprise , de nouvelles fonctionnalités et de nouveaux flux de travail.

— **Maintenance :** l'application doit être facile à maintenir et à mettre à jour, avec un code clair et bien structuré, des outils de diagnostic intégrés et une documentation complète.

4.4.2 Répartition des rôles Scrum

Dans cette section, nous allons présenter les acteurs Scrum qui participent à notre projet de développement de l'application de prédiction d'arrêt de pipelines. Nous décrirons le rôle du Scrum Master, de l'équipe de développement et du Product Owner, ainsi que leurs responsabilités et leurs contributions dans le cadre de notre projet.

1. **Scrum Master :** Le Scrum Master de notre équipe est Mr.Achour ACHROUFENE. Son rôle principal est de faciliter la mise en œuvre de la méthode Scrum et de veiller à ce que l'équipe respecte les principes et les pratiques de Scrum.
2. **Équipe de développement :** Nous sommes une équipe de développement composée de deux membres, GALI Kinza et LAGAB Sara. GALI Kinza se concentre sur les tâches de conception, notamment la création de diagrammes et la modélisation du système. Quant à LAGAB Sara est responsable des tâches de développement, y compris la programmation et l'implémentation des fonctionnalités de l'application. Elle se charge également de la création des interfaces visuelles, en veillant à ce qu'elles soient conviviales et attrayantes pour les utilisateurs.
3. **Product Owner :** Dans notre contexte, le Product Owner est le département d'entretien des lignes de l'entreprise Sonatrach. Il définit les fonctionnalités à développer et établit leurs priorités. Il veille à ce que l'application répond aux objectifs et aux attentes de Sonatrach en termes de prévention des arrêts de pipelines.

4.4.3 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation est un outil de modélisation qui permet de décomposer le fonctionnement d'un système en fonctionnalités. Il permet également de représenter la manière

dont les acteurs externes interagissent avec ces fonctionnalités qui sont offertes sous forme de services [113].

Alors pour construire ce diagramme, il est nécessaire de déterminer les acteurs qui interagissent avec le système. Dans le cadre de notre étude, nous avons identifié les trois acteurs suivants comme acteurs clés :

- **Ingénieur :** son rôle consiste a signaler de nouveaux incidents, effectuer des missions et consulter les statistiques et les différentes données.
- **Chef de service :** son principal rôle est la planification des missions.
- **Administrateur :** chargé de gérer les utilisateurs du système en effectuant des actions telles que l'ajout, la suppression et la gestion des comptes utilisateurs afin de garantir un accès approprié et sécurisé au système.

Après avoir identifié les acteurs impliqués dans le système, le diagramme de cas d'utilisation peut être présenté par la figure 4.3 :

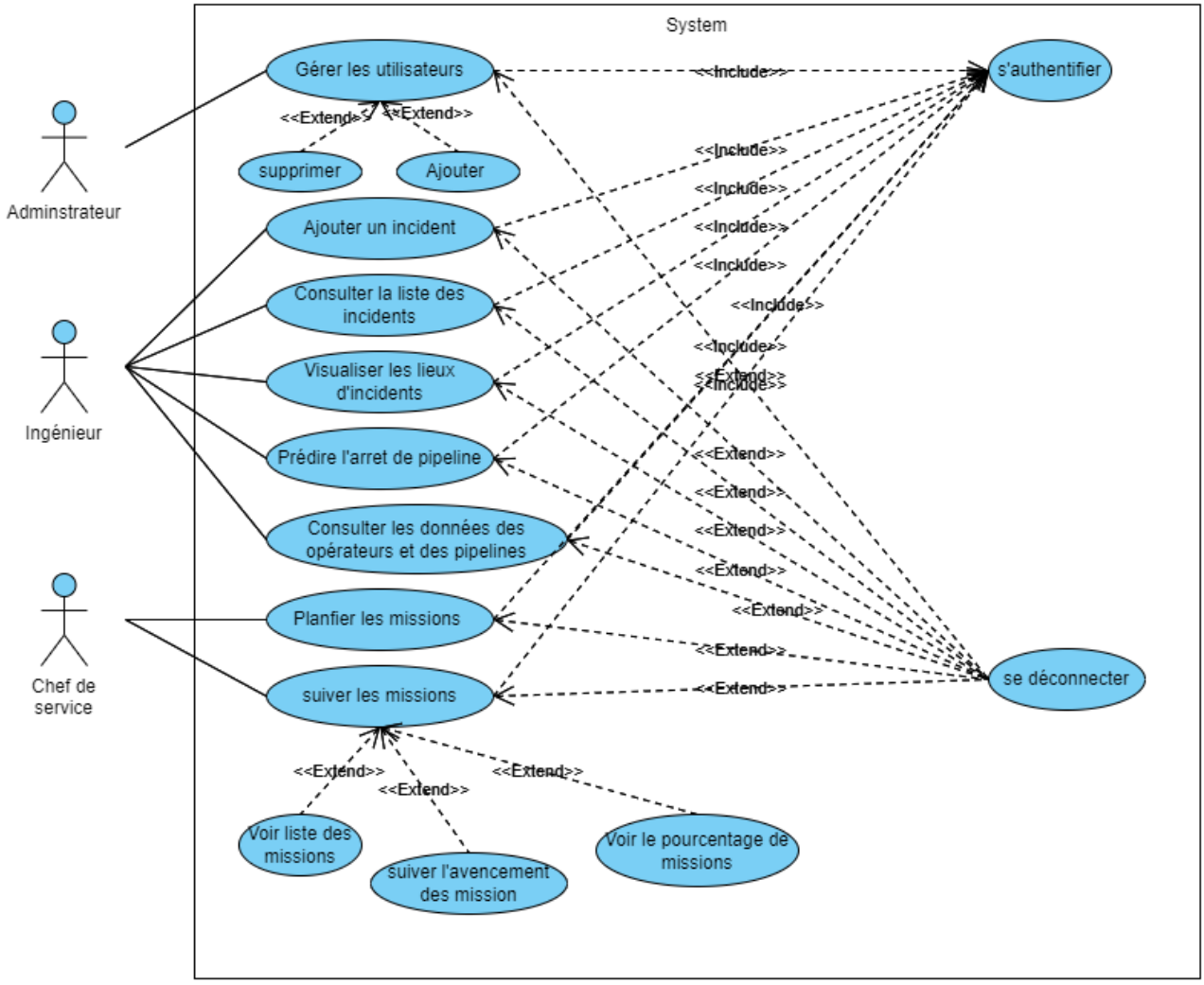

FIGURE 4.3 – Diagramme de cas d'utilisation général

4.4.4 Product backlog et estimation des sprints

Le Product Backlog est un élément important de la méthodologie Agile Scrum. Il consiste en une liste hiérarchisée des exigences fonctionnelles du produit, qui est constamment réévaluée

tout au long du projet pour répondre aux besoins changeants des utilisateurs et de l'entreprise [122].

Dans le tableau 4.1 nous allons définir l'ensemble des fonctionnalités (le product backlog) trié par ordre de priorité et les divisé en sprints :

Numéro de sprint	Fonctionnalités	Description	Durée
1	1- Authentification.	Ce sprint a pour objectif de permettre la consultation des incidents passées, l'ajout des nouveaux incidents, la consultation des données sur les pipelines et la visualisation des lieux des incidents. Il inclut également la mise en place d'un système d'authentification pour sécuriser l'accès aux données.	3 semaines
	2- Consultation des incidents passés.		
	3- Ajouter les nouveaux incidents.		
	4-Consultation des données sur les pipelines.		
	5- Visualisation des lieux des incidents.		
2	6- Prédiction.	Ce sprint a pour objectif de réaliser la prédiction des arrêts de pipelines en utilisant les données historiques.	2 semaines
3	7- Gestion des utilisateurs.	Ce sprint se concentre sur trois fonctionnalités principales la gestion des utilisateurs, la planification des missions et le suivit des missions. L'équipe se chargera d'intégrer les fonctionnalités permettant à l'administrateur de gérer les comptes utilisateurs et d'implémenter un système permettant au chef de service de planifier et d'assigner des missions aux ingénieurs et de suivre l'avancement des missions dans le temps.	2 semaines
	8- Planification des missions.		
	9- Suivi des missions.		

TABLE 4.1 – Backlog de produit et estimation de sprints.

4.4.5 Outils et bibliothèques

Dans le cadre de la planification de notre projet, nous avons choisi les outils suivants pour le développement de notre système d'aide à la décision :

Environnement matériel

Processeur	Intel(R) Core(TM) i3-6006U CPU @ 2.00GHz 2.00 GHz
Mémoire RAM	4 Go
Type du système	64 bits, processeur x64
Système d'exploitation	Windows 10 Professionnel

TABLE 4.2 – Spécification de matériel

Langages

Python : c'est un langage de programmation open source le plus utilisé par les programmeurs. Ce langage est utilisé dans plusieurs domaines tels que la gestion d'infrastructure, l'analyse de données et le développement de logiciels [20].

FIGURE 4.4 – Logo de python

SQL : est un langage de requête utilisé pour communiquer avec des bases de données relationnelles. Il permet de manipuler et d'interroger les données stockées dans ces bases de données, d'effectuer des opérations de création, modification et suppression des données, ainsi que des opérations de gestion de la structure de la base de données. SQL est un langage standardisé largement utilisé dans l'industrie et les applications de gestion de bases de données [48].

JavaScript : JavaScript est un langage de programmation de scripts utilisé principalement pour le développement web. Il permet d'ajouter des fonctionnalités interactives et dynamiques aux sites web, de manipuler le contenu HTML, de gérer les événements, de créer des animations et d'interagir avec les utilisateurs. JavaScript est un langage interprété, multiplateforme et orienté objet. Il est largement pris en charge par les navigateurs web modernes et est également utilisé côté serveur grâce à des plateformes telles que Node.js [91].

FIGURE 4.5 – Logo de javascript

QSS : QSS (Qt Style Sheets) est un langage de feuilles de style utilisé dans le framework Qt pour la définition de l'apparence et de la présentation des interfaces utilisateur. Il permet de définir les couleurs, les polices, les espacements, les effets visuels et autres propriétés de l'interface graphique de manière déclarative. QSS est largement utilisé pour personnaliser l'apparence des applications Qt et offrir une expérience utilisateur cohérente [46].

FIGURE 4.6 – Logo de QSS

Outils de conception et de modélisation

Qt Designer : Qt Designer est un outil de conception graphique utilisé pour créer des interfaces utilisateur pour des applications basées sur le framework Qt. Il permet aux développeurs de concevoir des interfaces utilisateur en utilisant une approche de glisser-déposer, en plaçant et en personnalisant les éléments graphiques de l'interface. Qt Designer génère ensuite le code source correspondant à l'interface conçue, qui peut ensuite être intégré dans le code de l'application [131].

FIGURE 4.7 – Logo de QT Designer

Figma : est un outil de conception d'interfaces utilisateur basé sur le cloud. Il permet aux designers et aux équipes de collaborer en temps réel sur la création, la modification et le partage de maquettes, de prototypes et de designs interactifs. Figma offre une interface intuitive, des fonctionnalités de conception avancées et une intégration transparente avec d'autres outils de conception. Il est largement utilisé dans l'industrie du design pour faciliter le processus de conception et la collaboration entre les membres de l'équipe [56].

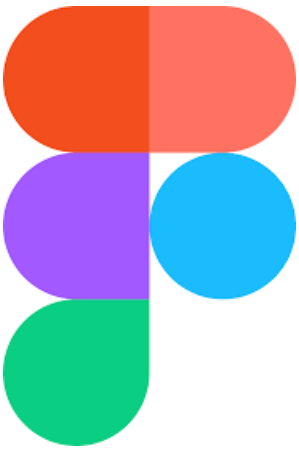

FIGURE 4.8 – Logo de Figma

UML (Langage de Modélisation Unifié) : est un langage de modélisation graphique. Il se compose d'un ensemble de symboles graphiques basés sur des diagrammes, qui permettent de spécifier, visualiser et documenter les systèmes logiciels orientés objet [136].
UML propose plusieurs diagrammes entre autres :

1. **Diagrammes de cas d'utilisations :** Les diagrammes de cas d'utilisation constituent un moyen de recueillir et de décrire les besoins des utilisateurs (ou acteurs) du système.
2. **Diagrammes d'interaction :** dans ce type de diagramme, les objets communiquent entre eux en s'envoyant des messages qui déclenchent l'exécution d'opérations sur les objets destinataires. Ces objets sont des instances des classes d'analyse : les interfaces, les contrôles et les entités. Cette représentation permet de visualiser les interactions dynamiques entre les objets et de comprendre le flux des messages dans le système [92].
3. **Diagramme de classe :** ce diagramme décrit la structure d'un système en modélisant ses classes, ses attributs, ses opérations et les relations entre ses objets [6].

FIGURE 4.9 – logo de UML

Visual paradigm : est un outil de modélisation et de développement agile de classe entreprise qui prend en charge l'UML, BPMN, ERD, DFD, SysML, ArchiMate, etc. Visual Paradigm est également un outil de collaboration qui prend en charge l'analyse de codes, la gestion de versions, la modélisation de bases de données, etc. [12].

FIGURE 4.10 – Logo visual paradigm

Environnement de développement

Spyder : Spyder est un environnement de développement intégré (IDE) spécialement conçu pour la programmation en Python. Il offre une interface conviviale et fonctionnelle avec des fonctionnalités avancées telles que l'éditeur de code, la console interactive, le débogueur et l'explorateur de variables. Spyder facilite le développement, le débogage et la gestion de projets Python, ce qui en fait un outil populaire parmi les développeurs Python [129].

FIGURE 4.11 – Logo de Spyder

SQLite Browser : est un outil open-source qui permet de visualiser, de modifier et de gérer des bases de données SQLite. Il offre une interface conviviale pour naviguer à travers les tables, exécuter des requêtes SQL, créer de nouvelles tables, importer et exporter des données, et bien plus encore. SQLite Browser est largement utilisé par les développeurs et les administrateurs de bases de données pour interagir avec des bases de données SQLite de manière facile et intuitive [11].

FIGURE 4.12 – Logo de sqlite browser

Jupyter : Jupyter est un environnement de développement interactif utilisé pour la création et l'exécution de documents contenant du code, des visualisations et des explications textuelles. Il prend en charge plusieurs langages de programmation, notamment Python, R et Julia, et permet de combiner du code, des textes, des images et des graphiques dans un même document appelé "notebook". Les notebooks Jupyter sont populaires dans le domaine de la science des données et de l'apprentissage automatique en raison de leur flexibilité et de leur capacité à faciliter l'exploration des données, la création de modèles et la communication des résultats [101].

FIGURE 4.13 – Logo de sqlite browser

API's et bibliothèques

Folium : est une bibliothèque Python utilisée pour la visualisation de données géospatiales. Elle permet de créer des cartes interactives et d'ajouter des marqueurs, des polygones, des couches de tuiles et d'autres éléments géographiques à ces cartes. Folium s'appuie sur les bibliothèques Leaflet.js et Matplotlib pour générer des cartes personnalisées et est particulièrement populaire pour la visualisation de données sur des cartes basées sur des feuilles de calcul ou des bases de données [110].

FIGURE 4.14 – Logo de Folium

Mapbox : Mapbox est une plateforme de cartographie en ligne qui fournit des services et des outils pour créer, personnaliser et intégrer des cartes interactives dans des applications. Elle offre des API et des SDK pour accéder aux données cartographiques, aux services de géocodage, de routage, de visualisation et d'autres fonctionnalités liées à la cartographie. Mapbox propose également des tuiles de cartes prêtes à l'emploi avec différentes couches, y compris des images satellites, des cartes routières et des cartes thématiques [88].

FIGURE 4.15 – Logo de mapbox

OpenStreetMap : OpenStreetMap (OSM) est un projet collaboratif de cartographie en ligne qui vise à créer une carte du monde libre et modifiable par tous. Les données cartographiques d'OpenStreetMap sont collectées par des contributeurs du monde entier et sont disponibles sous une licence ouverte. OSM fournit des données détaillées sur les routes, les bâtiments, les points d'intérêt et d'autres informations géographiques. Les utilisateurs peuvent accéder aux données OSM et les utiliser pour créer des cartes personnalisées et des applications basées sur la géolocalisation [60].

FIGURE 4.16 – Logo de openstreetmap

Matplotlib : Matplotlib est une bibliothèque de visualisation de données en Python largement utilisée pour créer des graphiques de haute qualité. Elle permet de créer des graphiques statiques, des diagrammes, des histogrammes, des graphiques en barres, des graphiques en nuages de points, des graphiques en boîte et bien plus encore. Matplotlib offre une grande flexibilité et un contrôle précis sur l'apparence des graphiques. Elle est souvent utilisée en conjonction avec d'autres bibliothèques Python telles que NumPy et Pandas pour l'analyse de données et la visualisation [67].

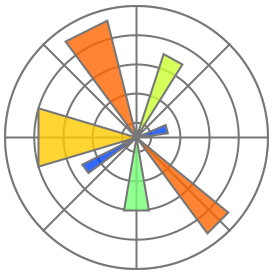

FIGURE 4.17 – Logo de Matplotlib

Pandas : est une bibliothèque open-source en Python spécialisée dans la manipulation et l'analyse de données. Elle offre des structures de données flexibles, telles que les DataFrames, qui permettent de gérer facilement des données tabulaires. Pandas fournit des fonctionnalités puissantes pour le nettoyage, la transformation, la fusion et l'analyse de données. Elle est largement utilisée dans les domaines de l'analyse de données, de l'apprentissage automatique et de la science des données [90].

FIGURE 4.18 – Logo de Pandas

Scikit-learn : Scikit-learn est une bibliothèque de machine learning en Python qui offre une large gamme d'algorithmes et d'outils pour l'apprentissage automatique. Elle permet de réaliser des tâches telles que la classification, la régression, le regroupement, la réduction de dimensionnalité et la sélection de modèles. Scikit-learn est appréciée pour sa simplicité d'utilisation, sa documentation détaillée et sa compatibilité avec d'autres bibliothèques Python comme NumPy et Pandas [100].

FIGURE 4.19 – Logo de Scikit-learn

SMTP : Le SMTP (Simple Mail Transfer Protocol) est un protocole de communication utilisé pour le transfert d'e-mails entre les serveurs de messagerie. Il définit les règles et les conventions pour l'acheminement et la livraison des messages électroniques. SMTP fonctionne selon un modèle client-serveur, où le client SMTP envoie un message au serveur SMTP qui se charge de la livraison du message à son destinataire. Le protocole SMTP utilise des commandes spécifiques pour l'envoi, la réception et le transfert des messages. Il utilise le port 25 par défaut, mais peut également utiliser d'autres ports sécurisés. Le SMTP repose sur le modèle de livraison "store-and-forward", où les serveurs SMTP temporisent et transfèrent les messages jusqu'à ce qu'ils puissent être livrés au destinataire final [73].

FIGURE 4.20 – Logo de SMTP

Elastic Email : Elastic Email est un service d'envoi d'e-mails en ligne qui offre des solutions d'automatisation de l'envoi de courriers électroniques. Il fournit une infrastructure cloud pour gérer les campagnes d'e-mails, les bulletins d'information et les notifications par e-mail. Elastic Email propose une interface conviviale permettant de créer et de personnaliser des modèles d'e-mails, de gérer les listes de contacts, de planifier l'envoi des messages et de suivre les performances des campagnes grâce à des statistiques détaillées. Le service offre également des fonctionnalités avancées telles que le suivi des ouvertures et des clics, la gestion des désabonnements et la gestion des rebonds. Elastic Email utilise des protocoles standard tels que SMTP et HTTP pour l'envoi et la réception des e-mails [14].

FIGURE 4.21 – Logo de Elastic Email

Conclusion

Dans ce chapitre, nous avons abordé la problématique des arrêts de pipelines et proposé une solution basée sur l'apprentissage automatique. Nous avons ensuite spécifié les besoins de l'application et détaillé le sprint 0, qui marque le démarrage du développement. Nous avons identifié les acteurs Scrum impliqués dans le projet, estimé les sprints nécessaires et défini les outils, bibliothèques et environnements requis. Ce chapitre constitue une étape cruciale dans notre projet, en établissant les bases solides de notre application de prédiction des arrêts de pipelines. Nous sommes maintenant prêts à avancer vers les prochains sprints de développement, à présenter dans les chapitres suivants, afin de concrétiser notre solution.

Chapitre 5

Sprint 1

Introduction

Dans ce chapitre, nous entamons le Sprint 1, qui représente la première itération de développement. Nous allons sélectionner les fonctionnalités les plus prioritaires du backlog de produit. En collaboration avec le client, nous identifions les tâches réalisables nécessaires pour réaliser ces fonctionnalités et nous les assignons aux membres de l'équipe. Nous planifions également la durée d'exécution de ces tâches afin de respecter les délais du sprint. Notre objectif ultime est de fournir des fonctionnalités qui satisferont pleinement le client.

5.1 Réunion de planification (planning meeting)

Nous avons tenu une réunion de planification pour le premier sprint, au cours de laquelle nous avons défini le backlog de sprint et la définition de done. Nous avons identifié les fonctionnalités prioritaires, telles que l'authentification, la consultation de l'historique des incidents, la consultation des données sur les pipelines et l'ajout de nouveaux incidents.

L'objectif principal de cette réunion était de déterminer les tâches spécifiques à réaliser dans le cadre du sprint, ainsi que les critères de finition pour chaque fonctionnalité. En tant qu'équipe de développement, nous avons réparti les responsabilités et estimé la durée des tâches afin de garantir une exécution efficace du sprint.

5.2 Diagramme de cas d'utilisation

Dans cette section nous allons présenté le diagramme de cas d'utilisation correspondant à ce sprint (Figure 5.1).

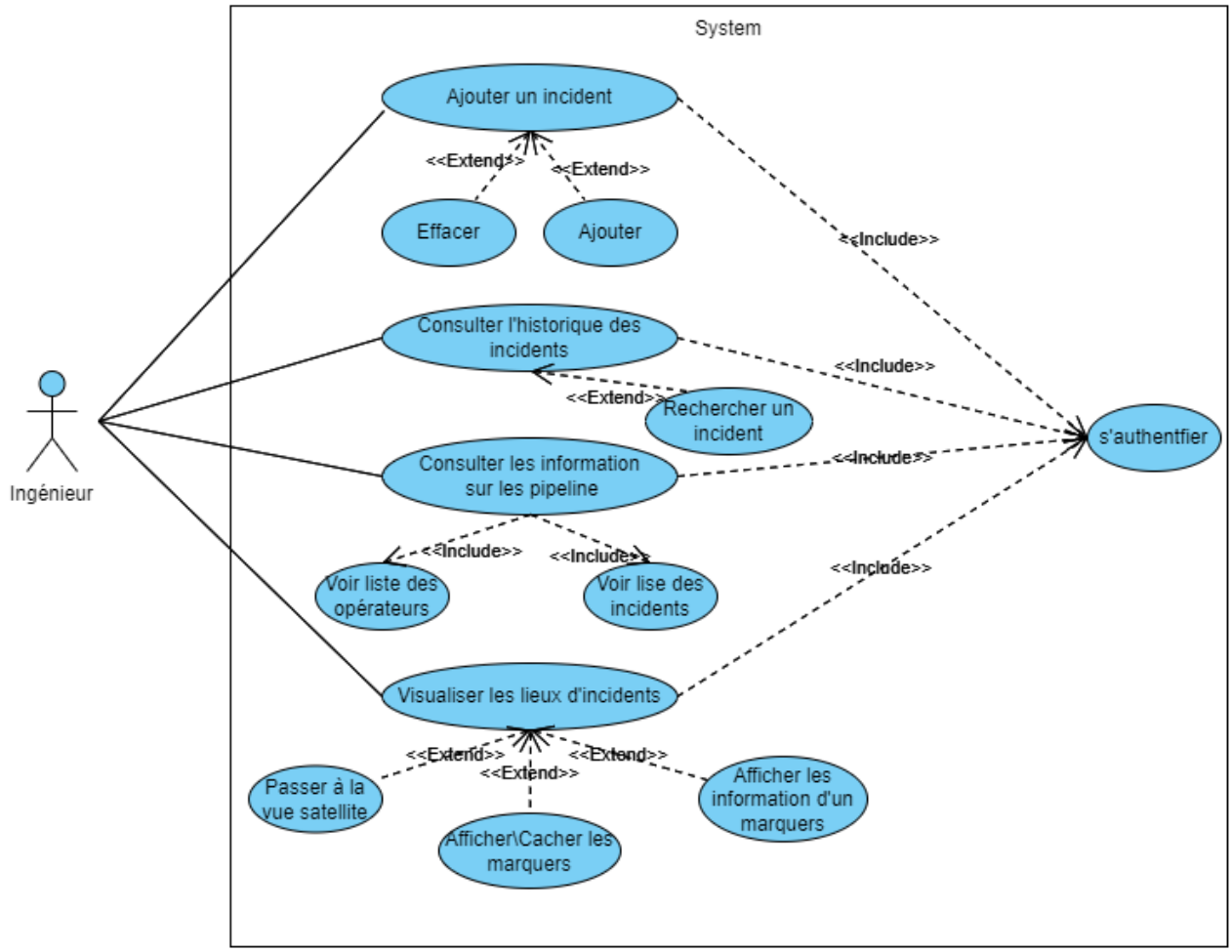

FIGURE 5.1 – Diagramme de cas d'utilisation de sprint 1

5.3 Backlog de sprint

Le tableau 5.1 représente le Backlog de Sprint que nous avons défini lors de la réunion de planification. Il offre une vue claire des fonctionnalités et des tâches prioritaires qui seront développées pendant le sprint.

Numéro de tâche	Description des tâches avec des User stories	Durée estimée	Qui réalise
1	En tant que membre de l'équipe de développement, je veux construire le diagramme de cas d'utilisation, pour expliquer les fonctionnalités de ce sprint.	1h	Gali Kinza, Lagab Sara
2	En tant que membre de l'équipe de développement, je veux rédiger la description des cas d'utilisation , pour les expliquer.	3h	Gali Kinza
3	En tant que membre de l'équipe de développement, je veux construire les diagrammes d'interaction pour les cas d'utilisation de ce sprint, pour expliquer en détail leurs scénarios.	3h	Gali Kinza

Numéro de tâche	User stories	Durée estimée	Qui Réalise
4	- En tant que membre d'équipe de développement, je veux modéliser les interface homme/machine pour les fonctionnalités de ce sprint, pour satisfaire le client	4h	Lagab Sara
5	Programmer la fonctionnalité "s'authentifier"	2h	Lagab Sara
6	Effectuer un test sur la fonctionnalité "s'authentifier"	30 min	Le client, Lagab Sara et Gali Kinza
8	Programmer la fonctionnalité "consulter l'historique des incidents"	4h	Lagab Sara
9	Effectuer un test sur la fonctionnalité "consulter l'historique des incidents"	30 min	
11	- En tant que membre d'équipe de développement, je veux programmer la fonctionnalité"visualiser les lieux des incidents", pour satisfaire le besoin de client.	2h	Lagab Sara
12	- En tant que membre d'équipe de développement, je veux tester la fonctionnalité "visualiser les lieux des incidents", pour garantir son bon fonctionnement.	2h	Lagab Sara
14	Programmer la fonctionnalité "Ajouter une incident"	3h	Lagab Sara
15	Effectuer un test sur la fonctionnalité "Ajouter une incident"	30 min	Le client, Gali Kinza et Lagab Sara
17	Programmer la fonctionnalité "consultation des données sur les pipelines"	3h	Lagab Sara
18	Effectuer un test sur la fonctionnalité "consultation des données sur les pipelines"	30 min	Le client, Lagab Sara et Gali Kinza
19	En tant que membre de l'équipe de développement, je veux construire construire le diagramme de classe, pour m'aider à construire le modèle logique de données	2h	Lagab Sara, Gali Kinza
20	En tant que membre de l'équipe de développement, je veux construire le modèle logique de données, pour savoir les tables de ma base de données (MLD)	2h	Gali Kinza
21	Réunion de revue de sprint	2h	Le client, Lagab Sara et Gali Kinza
22	Rédiger la documentation du Sprint 1	2 jours	Lagab Sara, Gali Kinza

TABLE 5.1 – Backlog de sprint 1 (Tâches)

5.3.1 Definition of Done (Définition de Fini)

La Definition of Done (DoD) est un ensemble d'objectifs et de critères convenus par l'équipe de développement pour déterminer quand une fonctionnalité ou un incrément de travail est considéré comme terminé et prêt à être livré. La DoD est utilisée pour s'assurer que chaque élément de backlog achevé répond aux normes de qualité et de finition attendues par le client et l'organisation[121].

Voici les définitions de done pour chaque fonctionnalité :

1. **Authentification :**
   - L'utilisateur peut s'inscrire et se connecter avec succès.
   - Les informations d'identification sont vérifiées et validées.
   - L'accès aux fonctionnalités restreintes est contrôlé par l'authentification.
2. **Consultation de l'historique des incidents :**
   - L'utilisateur peut afficher la liste complète des incidents précédents.
   - Les détails de chaque incident sont visibles.
3. **Consultation de données sur les pipelines :**
   - Les informations sur les pipelines sont accessibles.
   - Les informations sur les opérateur sont accessibles.
   - Les données sont présentées de manière claire et facilement compréhensible.
4. **Ajout de nouveaux incidents :**
   - Les utilisateurs peuvent saisir les détails d'un nouvel incident.
   - Les informations saisies sont validées et enregistrées dans la base de données.
   - Un numéro de rapport unique est attribué à chaque nouvel incident.
5. **Visualisation Géographique des Incidents :**
   - Les incidents sont correctement récupérés depuis la base de données.
   - Les incidents sont affichés sur la carte géographique en utilisant des marqueurs.
   - Les informations des incidents sont affichées lorsque l'utilisateur clique sur un marqueur.
   - L'utilisateur peut basculer entre les couches de la carte (carte standard, satellite) pour une meilleure visualisation.
   - La carte est réactive et se centre automatiquement sur les incidents affichés.
   - La visualisation des incidents sur la carte est fluide et sans erreurs d'affichage.

Ces définitions de done spécifiques assurent que chaque aspect de la fonctionnalité est pris en compte et fonctionne conformément aux attentes.

5.4 Description textuelle des cas d'utilisation

Par manque d'espace, nous allons présenter les cas d'utilisation les plus pertinents à savoir : "Ajouter un incident", "Consulter les information sur un pipeline" et "Visualiser les lieux d'incidents".

Cas d'utilisation "Ajouter un incident" :

Acteurs	Ingénieur
Description	Ajouter un nouvel incident
Date	15/05/2023
pré-condition	l'ingénieur doit être authentifié
Description des scénarios	
Scénario nominal	**Ajouter :** 1. L'ingénieur doit d'abord accéder à l'interface d'ajout d'incidents. 2. Après il doit remplir le formulaire. 3. Puis il clique sur le bouton ajouter. 4. Après avoir vérifié les informations fournies, le système enregistre les information dans la base de données. 5. Une boite de dialogue s'affiche avec le message "l'incident a été enregistré avec succès". **Effacer :** 1. L'ingénieur clique sur le bouton effacer. 2. Les champs sont remis a leur état initial.
Scénario alternatif	**Cas" nom d'établissement du pipeline invalide** 3.1 Une boite de dialogue avec le message "nom d'établissement du pipeline non valide veuillez entrer un nom d'établissement valide" s'affiche. **Cas "Emplacement invalide"** 3.1 Une boite de dialogue avec le message "Latitude et longitude invalides. Veuillez fournir un emplacement situé dans les États-Unis continentaux." s'affiche. **Cas "champs vides"** 3.1 Une boite de dialogue avec le message "champs vides. Veuillez remplir tous les champs" s'affiche.

TABLE 5.2 – Description textuelle du cas d'utilisation "Ajouter un incident"

Cas d'utilisation "Consulter les informations sur les pipelines"

Acteurs	Ingénieur
Description	Permet a l'ingénieur de voir les information sur les pipelines
Date	25/05/2023
pré-condition	L'ingénieur doit être authentifie.
Description des scénarios	
Scénario nominal	**Consultation du la liste des pipelines :** 1. L'ingénieur accéder à la page des informations des pipelines. 2. Il parcoure la liste des pipelines . **Consultation du la liste des opérateurs :** 1. L'ingénieur accéder à la page des informations des pipelines. 2. Il parcoure la liste des opérateurs .
Scénario alternatif	Aucun.

TABLE 5.3 – Description textuelle du cas d'utilisation "consulter les informations sur les pipelines"

Cas d'utilisation "Visualiser les lieux d'incidents"

Acteurs	Ingénieur
Description	Visualisation géographique des lieux d'incident sur une carte interactif.
Date	30/05/2023
pré-condition	l'ingénieur doit être authentifié
Description des scénarios	

Scénario nominal	**Afficher la vue satellite :** 1. L'utilisateur accède à l'interface de visualisation des lieux d'incident. 2. Par défaut, la vue carte est affichée avec tous les marqueurs d'incident positionnés sur la carte. 3. L'utilisateur a la possibilité de basculer vers la vue satellite en sélectionnant la couche satellite dans la boîte de dialogue de contrôle de couches. 4. La carte est mise à jour pour afficher les images satellite. 5. L'utilisateur peut interagir avec la carte pour explorer différents lieux d'incident en utilisant les fonctions de zoom et de déplacement. **Afficher/cacher la liste des marqueurs :** 1. L'utilisateur accède à l'interface de visualisation des lieux d'incident. 2. Par défaut, la liste des marqueurs est affichée sur la carte. 3. L'utilisateur a la possibilité de cacher la liste des marqueurs en désélectionnant la couches de marqueurs dans la boîte de dialogue de contrôle de couches. 4. La liste des marqueurs est masquée de la carte. 5. L'utilisateur peut toujours interagir avec la carte en utilisant les fonctions de zoom et de déplacement. **Afficher les informations d'un marqueur :** 1. L'utilisateur accède à l'interface de visualisation des lieux d'incident. 2. Par défaut, la vue carte est affichée avec tous les marqueurs d'incident positionnés sur la carte. 3. L'utilisateur peut cliquer sur un marqueur spécifique sur la carte pour afficher les informations détaillées de l'incident correspondant. 4. Une fenêtre contextuelle s'ouvre, affichant les détails de l'incident, tels que la date, l'emplacement, la catégorie de cause, etc. 5. L'utilisateur peut fermer la fenêtre contextuelle en cliquant sur le bouton de fermeture ou en cliquant en dehors de la fenêtre. 6. L'utilisateur peut interagir avec la carte pour explorer différents lieux d'incident en utilisant les fonctions de zoom et de déplacement.
Scénario alternatif	Aucun

TABLE 5.4 – Description textuelle du cas d'utilisation "visualiser les lieux d'incidents"

5.5 Diagrammes d'interaction

Dans cette section nous allons présenter le diagramme d'interaction pour correspondants aux cas d'utilisation décrits dans sections précédente.

Diagramme d'interaction pour "Ajouter un incident"

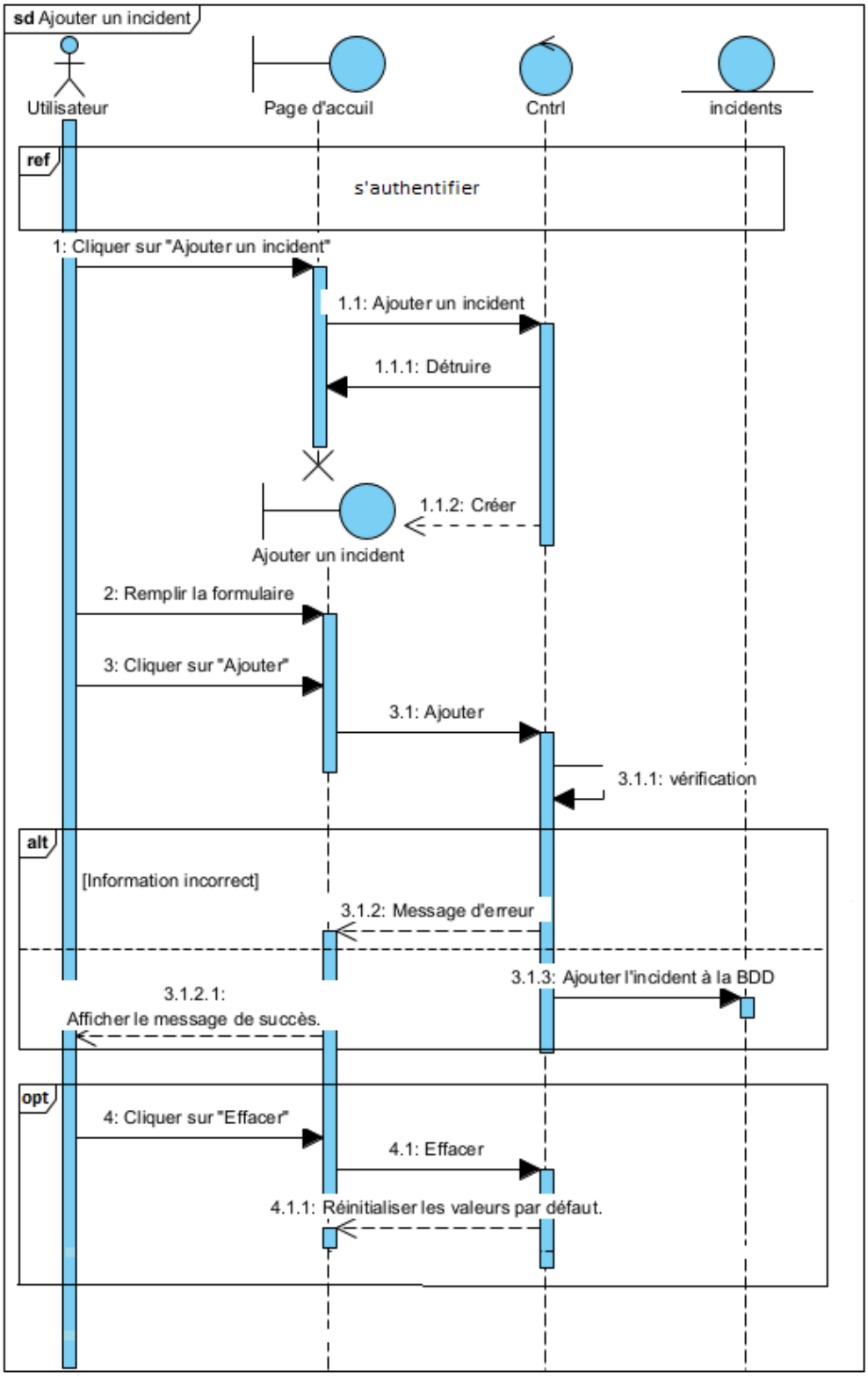

FIGURE 5.2 – Diagramme d'interaction pour "Ajouter un incident"

Diagramme d'interaction pour "Consulter les informations sur les pipelines"

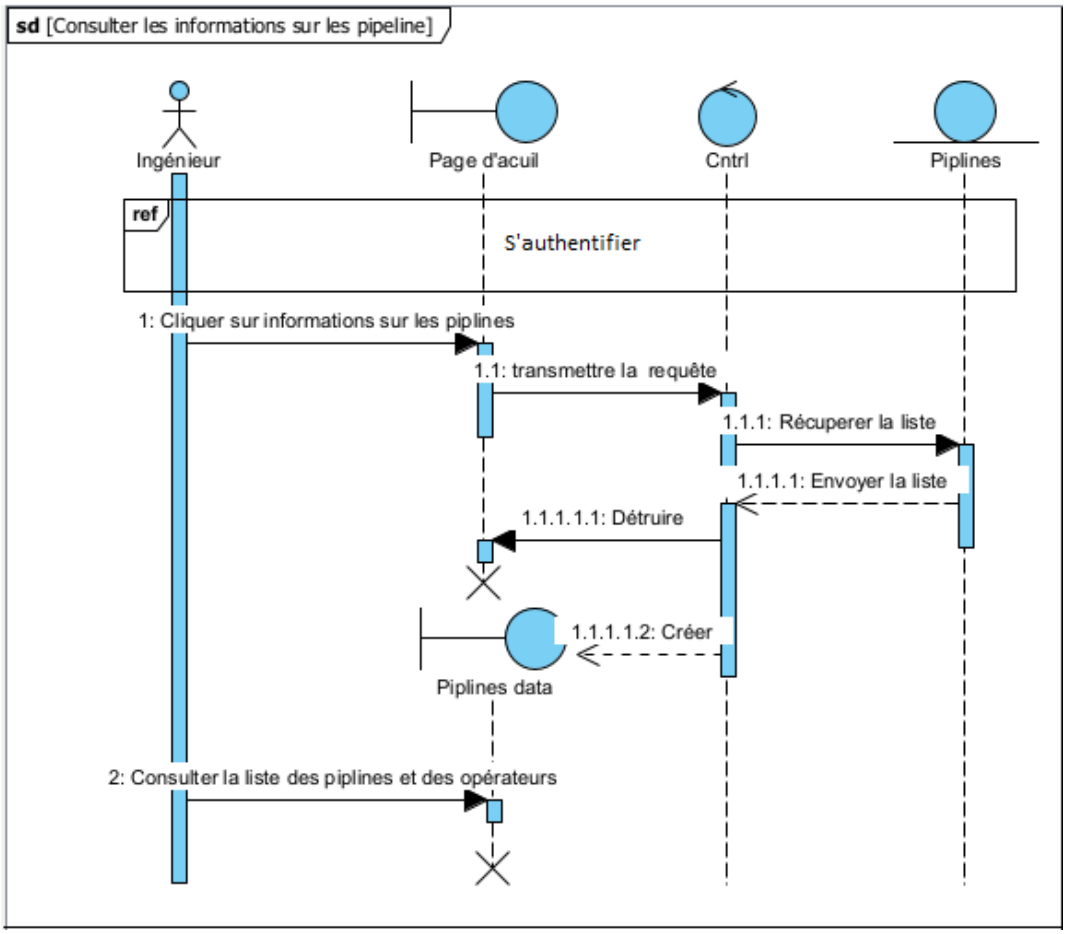

FIGURE 5.3 – Diagramme d'interaction pour "Consulter les informations sur les pipelines"

Diagramme d'interaction pour "Visualiser les lieux des incidents"

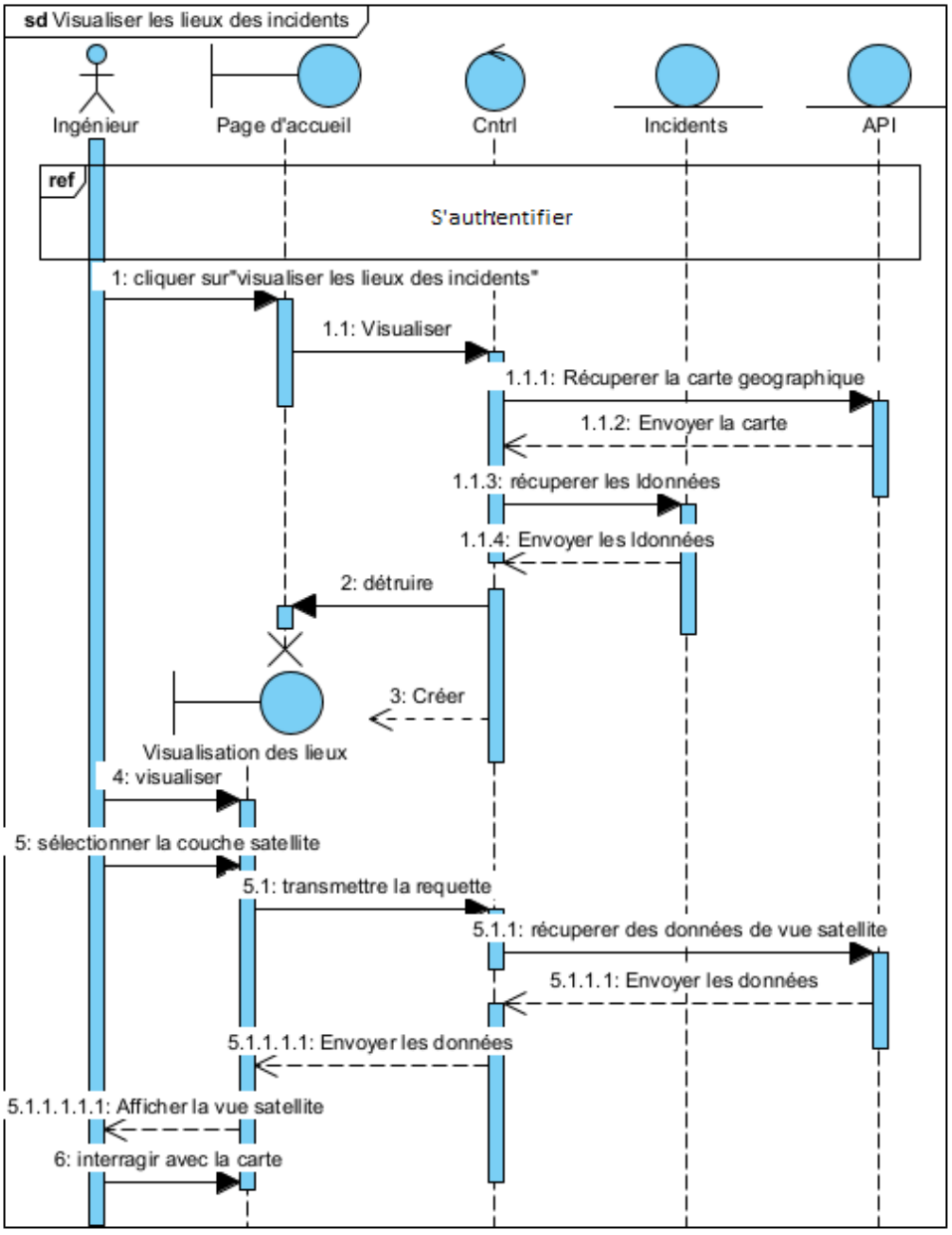

FIGURE 5.4 – Diagramme d'interaction pour "Visualiser les lieux des incidents"

5.6 Diagramme de classe

Le diagramme de classes des fonctionnalités du sprint 1 est donné par ma figure 5.5.

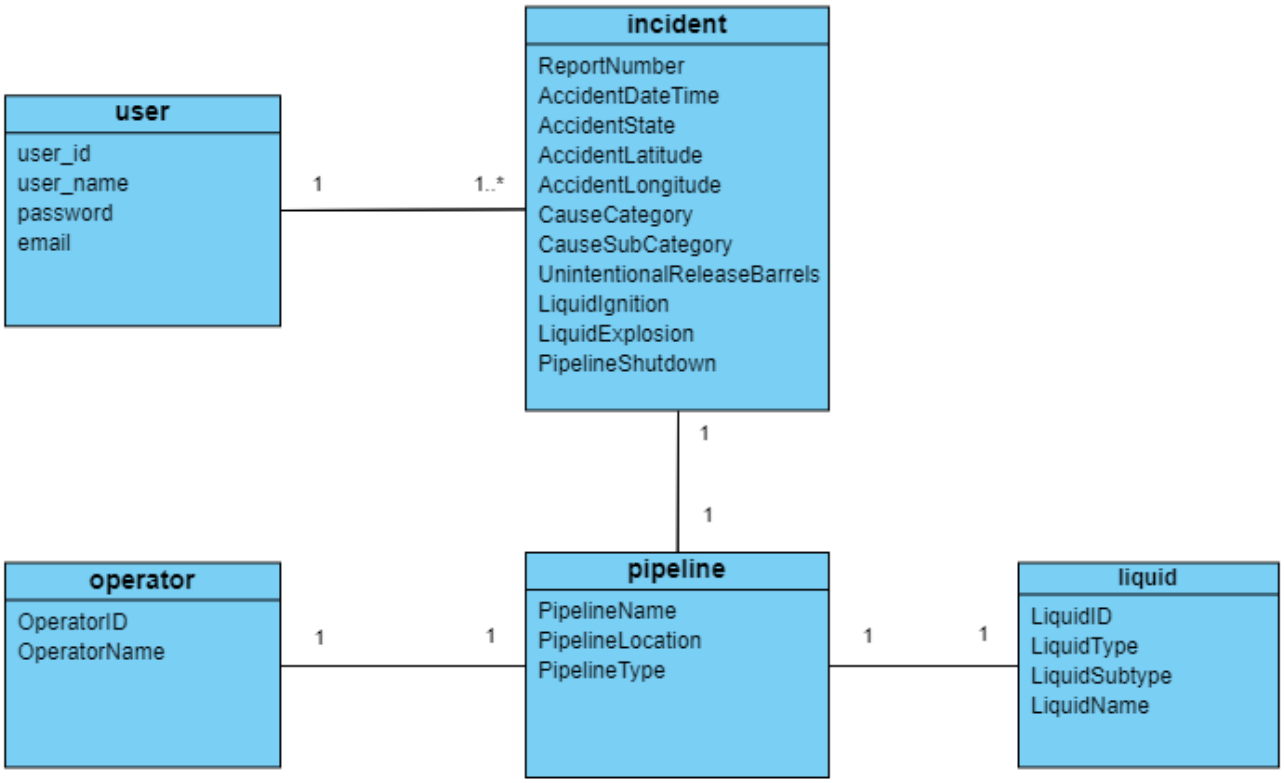

FIGURE 5.5 – Diagramme de classe de sprint 1

5.7 Passage au modèle relationnel

Dans cette section, nous allons passer du diagramme de classe précédemment présenté au modèle relationnel de données en appliquant les règles de transformation appropriées. Ensuite, nous fournirons le dictionnaire de données correspondant au modèle relationnel obtenu.

5.7.1 Règle de passage au modèle relationnel

1. Chaque classe du diagramme de classes devient une table dans le modèle relationnel. Les attributs de chaque classe deviennent des colonnes dans la table correspondante.
2. Les associations entre les classes sont représentées par des clés étrangères dans les tables correspondantes. Une clé étrangère fait référence à la clé primaire d'une autre table pour établir une relation entre les enregistrements.
3. Les attributs multivalués ou les attributs de type tableau dans une classe sont représentés par une nouvelle table avec une clé étrangère vers la table d'origine.
4. Les attributs d'agrégation dans une classe sont également représentés par une nouvelle table avec une clé étrangère vers la table d'origine.
5. Les cardinalités des associations sont traduites en clés étrangères. Par exemple, une association "un à plusieurs" est représentée par une clé étrangère dans la table du côté

"plusieurs".

6. Les attributs avec des contraintes d'unicité dans une classe sont représentés par des contraintes de clé primaire ou d'index dans la table correspondante.
7. Les opérations et méthodes dans les classes ne sont généralement pas directement traduites dans le modèle relationnel, car elles sont implémentées au niveau de l'application.

Ces règles constituent une approche générale pour convertir un diagramme de classes en un modèle relationnel, qui sera présenté dans la prochaine sous-section.

5.7.2 Modèle relationnel

Voici le modèle relationnel corréspondent à ce sprint :

Pipelines (PipelineID, PipelineName, PipelineLocation, PipelineType, #OperatorID, #LiquidID)

Liquids (LiquidID, LiquidType, LiquidSubtype, LiquidName)

Operators (OperatorID, OperatorName)

Incidents (ReportNumber, SupplementalNumber, AccidentDateTime, #PipelineID, AccidentState, AccidentLatitude, AccidentLongitude, CauseCategory, CauseSubcategory, UnintentionalReleaseBarrels, LiquidIgnition, LiquidExplosion, PipelineShutdown)

Users (ID,username,password,email)

5.7.3 Dictionnaire de données

Le dictionnaire de données, également connu sous le nom de dictionnaire de données relationnel, est une représentation structurée des éléments de données utilisés dans un système informatique. Il fournit une vue détaillée des entités, attributs, types de données et contraintes associés à une base de données[43].
Dans cette sous section, nous présenterons le dictionnaire de données du sprint sous forme de tables pour une meilleure organisation et visualisation des informations. **Table incident**

ReportNumber	INTEGER	"ReportNumber" INTEGER
SupplementalNumber	INTEGER	"SupplementalNumber" INTEGER
AccidentDateTime	TEXT	"AccidentDateTime" TEXT
PipelineID	INTEGER	"PipelineID" INTEGER
AccidentState	TEXT	"AccidentState" TEXT
AccidentLatitude	REAL	"AccidentLatitude" REAL
AccidentLongitude	REAL	"AccidentLongitude" REAL
CauseCategory	TEXT	"CauseCategory" TEXT
CauseSubcategory	TEXT	"CauseSubcategory" TEXT
UnintentionalReleaseBarrels	INTEGER	"UnintentionalReleaseBarrels" INTEGER
LiquidIgnition	TEXT	"LiquidIgnition" TEXT
LiquidExplosion	TEXT	"LiquidExplosion" TEXT
PipelineShutdown	TEXT	"PipelineShutdown" TEXT

FIGURE 5.6 – Table incident

Table pipeline

PipelineID	INTEGER	"PipelineID" INTEGER
PipelineName	TEXT	"PipelineName" TEXT
PipelineLocation	TEXT	"PipelineLocation" TEXT
PipelineType	TEXT	"PipelineType" TEXT
OperatorID	INTEGER	"OperatorID" INTEGER
LiquidID	INTEGER	"LiquidID" INTEGER

FIGURE 5.7 – Table pipeline

Table liquide

LiquidID	INTEGER	"LiquidID" INTEGER
LiquidType	TEXT	"LiquidType" TEXT
LiquidSubtype	TEXT	"LiquidSubtype" TEXT
LiquidName	TEXT	"LiquidName" TEXT

FIGURE 5.8 – Table liquide

Table opérateur

OperatorID	INTEGER	"OperatorID" INTEGER
OperatorName	TEXT	"OperatorName" TEXT

FIGURE 5.9 – Tables opérateur

Table utilisateur

id	INTEGER	"id" INTEGER NOT NULL UNIQUE
username	TEXT	"username" TEXT NOT NULL
password	TEXT	"password" TEXT NOT NULL
image	BLOB	"image" BLOB NOT NULL
email	TEXT	"email" TEXT NOT NULL

FIGURE 5.10 – Table utilisateur

5.8 Interfaces Hommes/Machines

L'importance des IHM réside dans leur rôle crucial dans l'expérience utilisateur. Une IHM bien conçue peut améliorer l'efficacité, la convivialité et la satisfaction de l'utilisateur lors de l'utilisation d'un logiciel ou d'un appareil [57]. Par conséquent, nous avons crée des interfaces qui répondent aux besoins de l'utilisateur, nous présentons quelques-unes.

Interface ajout des incidents

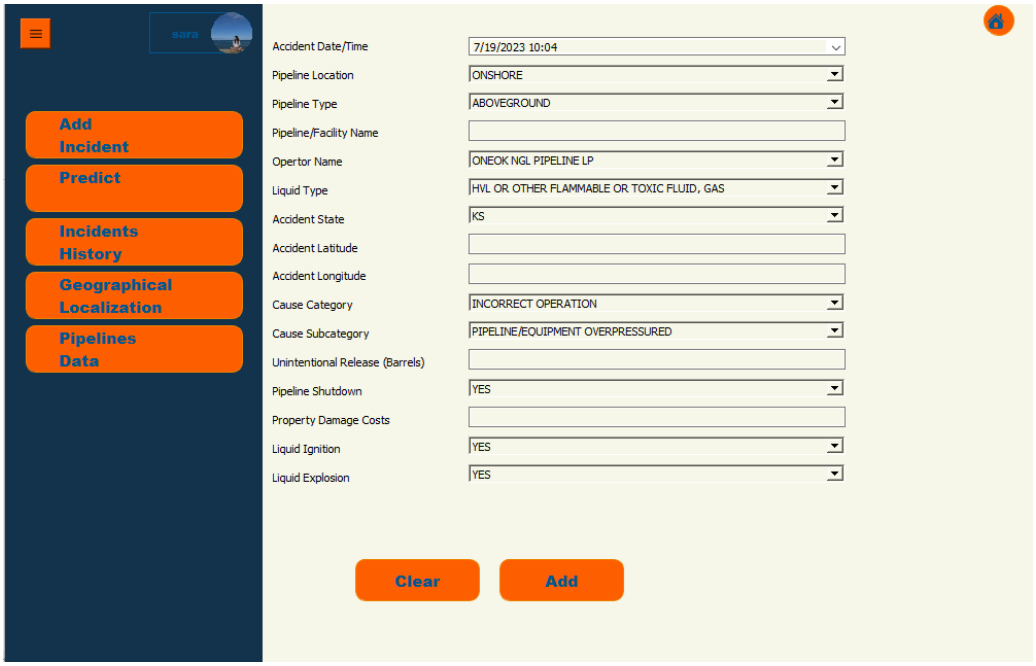

FIGURE 5.11 – Interface d'ajout des incidents

Interface information sur les pipelines

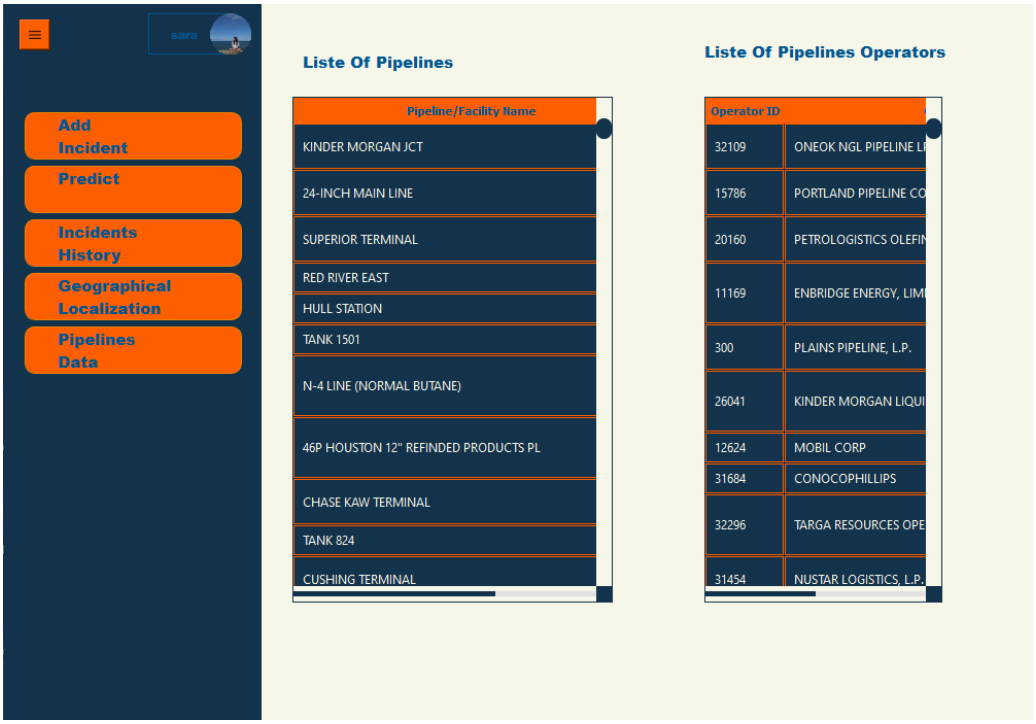

FIGURE 5.12 – Interface information sur les pipelines.

Interface visualisation des lieux d'incidents

Nous pouvons visualiser les lieux d'incidents sous de vue plan ou de vue satellite comme pour la figure 5.13.

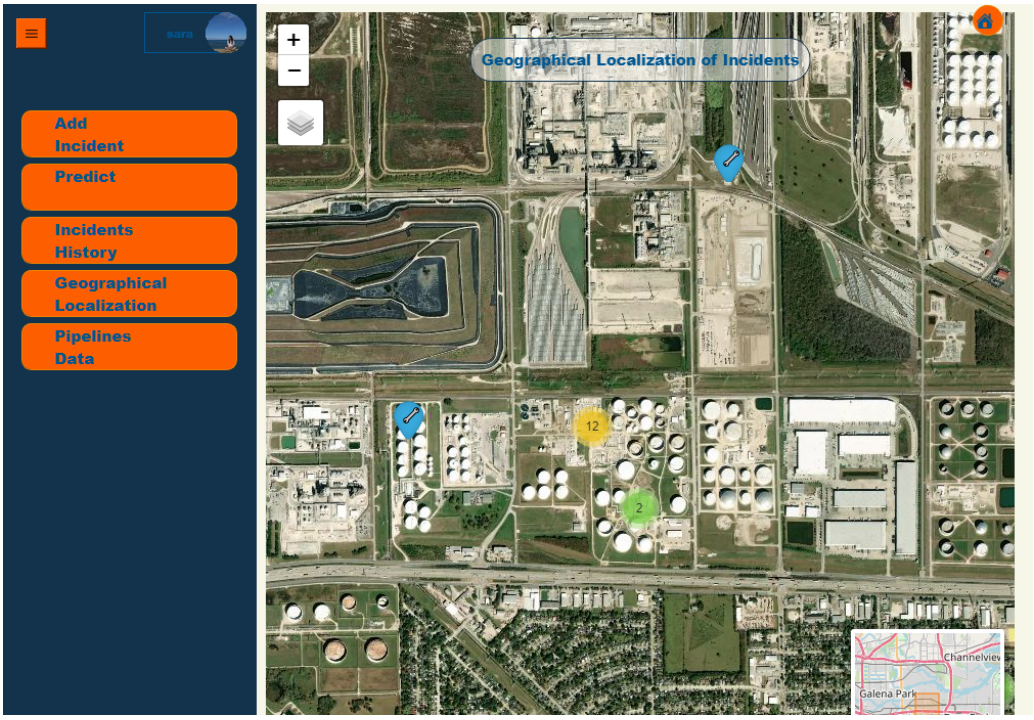

FIGURE 5.13 – Interface visualisation des lieux d'incidents "Carte vue satellite"

5.9 Évaluation de sprint

Dans le cadre de l'évaluation du sprint, nous avons organisé deux réunions importantes : une réunion de revue et une réunion de rétrospective.
La réunion de revue a été l'occasion de présenter les fonctionnalités réalisées lors du sprint au client. Nous avons partagé les avancées du projet et démontré les fonctionnalités développées. Le client a pu donner son avis sur les fonctionnalités livrées et exprimer sa satisfaction quant à la qualité du travail réalisé.
La réunion de rétrospective a permis à l'équipe de réfléchir sur le déroulement du sprint. Nous avons évalué notre méthode de travail et discuté des points forts et des points à améliorer. Nous avons analysé les succès et les difficultés rencontrés durant le sprint, identifié les bonnes pratiques à maintenir et proposé des actions correctives pour améliorer notre efficacité dans les prochains sprints.

Conclusion

Ce chapitre a présenté les étapes suivi avec pour mettre en oeuvre les fonctionnalités du Sprint 1. Nous avons réussi à réaliser un premier livrable qui permet au client de s'authentifier, d'ajouter de nouveaux incidents, de consulter l'historique des incidents et d'accéder aux informations sur les pipelines. Cela a été possible grâce à une planification adéquate et une collaboration étroite avec le client. Nous sommes confiants dans notre capacité à continuer sur cette lancée et à répondre pleinement aux besoins du client tout au long du développement du projet.
La fonctionnalité à réaliser dans le sprint 2 est la prédiction, nous allons donc appliquer le sprint apprentissage de la méthode proposée dans le chapitre suivant.

Chapitre 6

Sprint 2

Introduction

Dans ce chapitre, nous abordons le Sprint 2 de notre projet axé sur le développement de la fonctionnalité prédiction qui fait appelle à l'apprentissage automatique. Au cours de ce sprint, nous mettrons en œuvre la méthode proposée, en intégrant les 4 phases du sprint apprentissage à savoir : l'acquisition de connaissances, la conceptualisation, l'implémention et l'évaluation.

6.1 Réunion de planification

Lors de la réunion de planification du Sprint 2, il a été décidé que Mr. ACHROUFENE joue également le rôle du cogniticien et LAGAB et GALI jouent aussi le rôle de l'expert du domaine. En collaboration, nous avons établi le sprint backlog de tâches détaillées et défini la definition of Done (DoD) de la fonctionnalité prédiction. Cette réunion a permis de clarifier les objectifs du sprint, de fixer les priorités et de s'assurer que toute l'équipe était alignée sur les attentes du Sprint 2.

6.2 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation correspondant au sprint 2 est donnée par la figure 6.1.

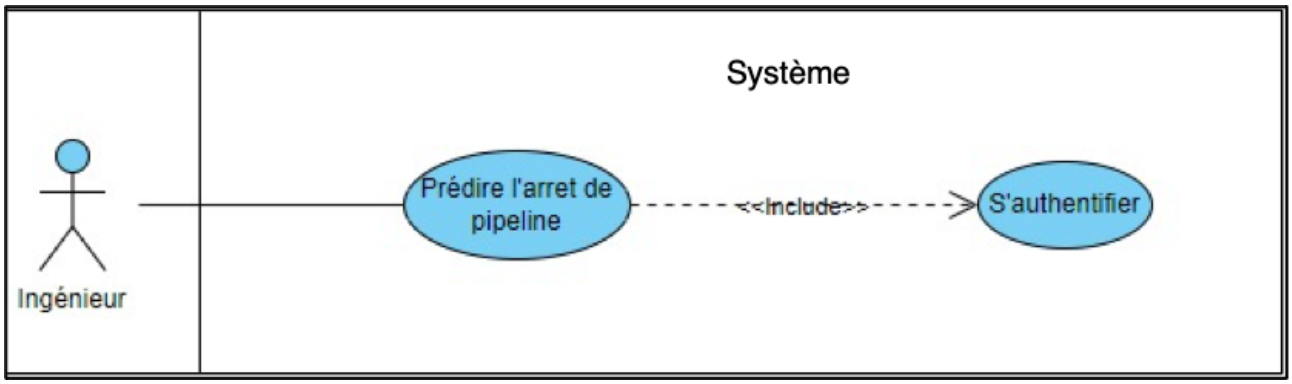

FIGURE 6.1 – Diagramme de cas d'utilisation sprint 2

6.3 Backlog de sprint

Le backlog du sprint 2 est donnée par le tableau 6.1. Les tâches de 2 à 5 peuvent être répétées autant de fois que nécessaire, nous pouvons essayer différents modèles ou bien ajuster les paramètres du même modèle jusqu'à l'obtention d'un modèle qui prédit des valeurs proches de la réalité. Les autres tâches de développement peuvent être réalisées en parallèle.

Numéro de tâche	Description des tâches avec des User stories	Durée estimée	Qui réalise
1	En tant que membre de l'équipe de développement, je veux faire le prétraitement de le jeu de données, pour pouvoir l'utiliser dans l'entraînement.	1 h	LAGAB Sara
2	En tant que membre de l'équipe de développement, je veux entraîner un modèle de classification, pour prédire l'arrêt de pipelines.	1h	LAGAB Sara
3	En tant que membre de l'équipe de développement, je veux tester le modèle de prédiction entraîné , pour comparer les valeurs prédit et les valeurs d'origine.	1h	LAGAB Sara
4	En tant que membre de l'équipe de développement, je veux évaluer le modèle de prédiction entraîné , pour mesurer ses performances.	1h	LAGAB Sara
5	En tant que membre de l'équipe de développement, je veux implémenter le modèle de prédiction dans l'application, pour pouvoir l'utiliser .	1h	LAGAB Sara
6	En tant que membre de l'équipe de développement, je veux construire le diagramme de cas d'utilisation, pour modéliser la fonctionnalité "prédire"	1h	GALI Kinza
7	En tant que membre de l'équipe de développement, je veux rédiger la description de cas d'utilisation, pour l'expliquer	1h	GALI Kinza
8	En tant que membre de l'équipe de développement, je veux construire le diagramme d'interaction, pour expliquer le scénario de fonctionnalité	1h	GALI Kinza
9	En tant que membre de l'équipe de développement, je veux construire le diagramme de classe, pour pouvoir réaliser le modèle logique de données	1h	GALI Kinza
10	En tant que membre de l'équipe de développement, je veux construire le modèle logique de données, pour pouvoir implémenter la base de données	1h	GALI Kinza
11	En tant que membre d'équipe de développement, je veux modéliser l'interface homme/machine de prédiction, pour satisfaire le client	2h	LAGAB Sara
12	En tant que membre d'équipe de développement, je veux programmer la fonctionnalité prédire, pour satisfaire le client	2h	LAGAB Sara
13	En tant que membre d'équipe de développement, je veux tester la fonctionnalité prédire, pour assurer qu'elle correspond au besoin exprémi par le client.	2h	LAGAB Sara et GALI Kinza

Numéro de tâche	User stories	Durée estimée	Qui réalise
14	Réunion de revue de sprint	2h	Le client, LAGAB Sara et GALI Kinza
15	Rétrospective de sprint	2h	Le client, LAGAB Sara et GALI Kinza
16	Rédiger la documentation du sprint	2 h	LAGAB Sara, GALI Kinza
17	Livré le produit de sprint ou bien itérer	1h	LAGAB Sara, GALI Kinza

TABLE 6.1 – Backlog de sprint

Definition of Done (DoD) pour la fonctionnalité "Prédiction" est comme suit :

- Le modèle de prédiction est correctement entraîné sur les données historiques.
- Les performances du modèle ont été évaluées et répondent aux critères définis pour l'exactitude et la précision.
- L'interface utilisateur permet aux utilisateurs autorisés d'accéder à la fonctionnalité de prédiction.
- Les résultats de la prédiction sont affichés de manière claire et compréhensible pour les utilisateurs.

6.4 Acquisition de connaissance

Dans ce travail, l'acquisition de connaissances se résume à la collecte de données sur internet, vue que l'inaccessibilité aux connaissance de l'organisme d'accueil.
Comme indiqué précédemment, nous avons récupéré un jeu de données à partir de Kaggle. Ce jeu de données contient un enregistrement pour chaque fuite ou déversement de pipelines pétroliers signalé à l'administration de la sécurité des pipelines et des matières dangereuses depuis 2010. Les informations incluses dans ces enregistrements comprennent la date et l'heure de l'incident, les détails sur l'opérateur et le pipeline concernés, la cause de l'incident, le type de liquide dangereux et la quantité perdue, les informations sur les blessures et les décès, ainsi que les coûts associés à chaque incident.
Ces rapports sur les accidents de pipelines pétroliers ont été collectés et publiés par l'administration de la sécurité des pipelines et des matières dangereuses du département des transports (DOT).
Le tableau 6.2 explique les 48 colonnes correspondantes à ce jeu de données.

Nom de la colonne	**Signification**	**Type de données**	**Nombre de lignes**
Report Number	Numéro de rapport	int64	2795
Supplemental Number	Numéro supplémentaire	int64	2795
Accident Year	Année de l'accident	int64	2795
Accident Date/Time	Date/heure de l'accident	object	2795

Suite à la page suivante

Nom de la colonne	Signification	Type de données	Nombre de lignes
Operator ID	Identifiant de l'opérateur	int64	2795
Operator Name	Nom de l'opérateur	object	2795
Pipeline/Facility Name	Nom du pipeline/installation	object	2674
Pipeline Location	Emplacement du pipeline	object	2795
Pipeline Type	Type du pipeline	object	2777
Liquid Type	Type de liquide	object	2795
Liquid Subtype	Sous-type de liquide	object	1349
Liquid Name	Nom du liquide	object	222
Accident City	Ville de l'accident	object	2480
Accident County	Comté de l'accident	object	2720
Accident State	État de l'accident	object	2783
Accident Latitude	Latitude de l'accident	float64	2795
Accident Longitude	Longitude de l'accident	float64	2795
Cause Category	Catégorie de la cause	object	2795
Cause Subcategory	Sous-catégorie de la cause	object	2795
Unintentional Release (Barrels)	Libération involontaire (Barils)	float64	2795
Intentional Release (Barrels)	Libération intentionnelle (Barils)	float64	1209
Liquid Recovery (Barrels)	Récupération de liquide (Barils)	float64	2795
Net Loss (Barrels)	Perte nette (Barils)	float64	2795
Liquid Ignition	Ignition du liquide	object	2795
Liquid Explosion	Explosion du liquide	object	2795
Pipeline Shutdown	Fermeture du pipeline	object	2583
Shutdown Date/Time	Date/heure de fermeture	object	1390
Restart Date/Time	Date/heure de redémarrage	object	1341
Public Evacuations	Évacuations publiques	float64	2338
Operator Employee Injuries	Blessures d'employés de l'opérateur	float64	12
Operator Contractor Injuries	Blessures d'entrepreneurs de l'opérateur	float64	12
Emergency Responder Injuries	Blessures de secouristes	float64	12
Other Injuries	Autres blessures	float64	12
Public Injuries	Blessures publiques	float64	12
All Injuries	Total des blessures	float64	12
Operator Employee Fatalities	Décès d'employés de l'opérateur	float64	8
Operator Contractor Fatalities	Décès d'entrepreneurs de l'opérateur	float64	8
Emergency Responder Fatalities	Décès de secouristes	float64	8
Other Fatalities	Autres décès	float64	8
Public Fatalities	Décès publics	float64	8
All Fatalities	Total des décès	float64	8
Property Damage Costs	Coûts des dommages matériels	float64	2788
Lost Commodity Costs	Coûts des marchandises perdues	float64	2791

Suite à la page suivante

Nom de la colonne	Signification	Type de données	Nombre de lignes
Public/Private Property Damage Costs	Coûts des dommages aux biens publics/privés	float64	2785
Emergency Response Costs	Coûts d'intervention d'urgence	float64	2789
Environmental Remediation Costs	Coûts de remédiation environnementale	float64	2787
Other Costs	Autres coûts	float64	2779
All Costs	Coûts totaux	int64	2795

TABLE 6.2 – Informations sur les colonnes du jeu de données des accidents de pipelines pétroliers

Notre prochaine étape va être le nettoyage de données et la préparation des données.

6.5 Conceptualisation

Cette phase englobe tous ce qui est prétraitement et préparation de données.

6.5.1 Nettoyage de données

Le nettoyage des données est une étape essentielle dans la préparation des données pour l'apprentissage automatique. Il vise à garantir que les données utilisées dans le modèle sont de haute qualité, cohérentes, complètes et adaptées à l'objectif de l'apprentissage automatique.
Pour cela, nous avons procédé au nettoyage de notre jeu de données. Tout d'abord, nous avons éliminé les colonnes redondantes ainsi que celles qui contiennent plus de 50% de valeurs manquantes, car leur présence pourrait induire en erreur et entraîner des résultats incorrects lors de l'analyse ou de l'entraînement de modèles. Cette étape permet d'améliorer la qualité et la fiabilité des données, ce qui contribue à obtenir des résultats plus précis dans nos analyses et modèles d'apprentissage automatique.

6.5.2 Imputation des données manquants

L'imputation est une méthode de prétraitement de données utilisée pour remplacer les valeurs manquantes dans un jeu de données par des valeurs estimées. Cela permet de conserver la cohérence des données et de faciliter l'analyse statistique et l'entraînement de modèles d'apprentissage automatique [85].
Dans notre étude, nous avons choisi l'imputation par moyenne pour remplacer les valeurs manquantes dans les variables numériques. L'imputation par moyenne consiste à calculer la moyenne des valeurs non manquantes de la variable et à remplacer les valeurs manquantes par cette moyenne.
Notre choix s'est basé sur les considérations suivantes :

1. **Petite proportion de valeurs manquantes :** Notre jeu de données présentait une petite proportion de valeurs manquantes par rapport à la taille totale du jeu de données. L'imputation par moyenne est généralement appropriée lorsque les valeurs manquantes sont peu nombreuses.
2. **Variables numériques avec distribution approximativement normale :** Les variables pour lesquelles nous avons appliqué l'imputation par moyenne avaient une distribution approximativement normale. Dans ce contexte, la moyenne est une estimation raisonnable des valeurs manquantes.

3. **Absence de relations complexes avec d'autres variables :** Nous avons vérifié qu'il n'y avait pas de relations complexes entre les variables avec des valeurs manquantes et d'autres variables du jeu de données. Dans ce cas, l'imputation par moyenne peut fournir des estimations raisonnables sans introduire de biais significatif.
4. **Gain de temps :** L'imputation par moyenne est une méthode simple, rapide et facile à implémenter. Étant donné que nous devions nettoyer et prétraiter un grand volume de données, cette méthode nous a permis de gagner du temps sans sacrifier la qualité des résultats.

6.5.3 Encodage de données

L'encodage fait référence au processus de transformation de variables catégorielles en une forme appropriée pour les modèles d'apprentissage automatique. Les algorithmes d'apprentissage automatique ne peuvent généralement pas traiter directement les données catégorielles, il est donc nécessaire de les convertir en valeurs numériques. Dans notre cas nous avons utilisé deux techniques d'encodage qui sont :

1. **Encodage One-Hot :** L'encodage one-hot est utilisé pour les variables catégorielles non ordinales. Chaque catégorie est représentée par une variable binaire distincte, où 1 indique la présence de la catégorie et 0 son absence. Cela permet de traiter chaque catégorie comme une entité indépendante, évitant ainsi les biais liés à l'ordre ou à l'échelle des catégories [3].
2. **Encodage Cyclique :** L'encodage cyclique est utilisé pour les variables périodiques telles que les mois, les jours de la semaine ou les heures dans une journée. Plutôt que de les représenter par des nombres continus, l'encodage cyclique les transforme en deux dimensions (souvent appelées sin-cos encodage) sur le cercle unité. Cela permet de conserver les informations de périodicité sans introduire de biais artificiel dans les données[5].

Nous avons utilisé l'encodage cyclique pour traiter les variables de date et d'heure (comme le mois et l'heure de l'accident), car ces variables ont une nature périodique. En utilisant l'encodage cyclique, nous conservons les relations de périodicité entre les valeurs temporelles.
Nous avons également utilisé l'encodage one-hot pour transformer les variables catégorielles non ordinales (telles que le type de pipeline, le type de liquide impliqué dans l'accident, etc.). Cela nous permet de traiter chaque catégorie comme une caractéristique indépendante, ce qui est essentiel pour les modèles d'apprentissage automatique.

6.5.4 Mise à l'échelle des données

La mise à l'échelle (ou "scaling" en anglais) est une étape essentielle du prétraitement des données dans le domaine de l'apprentissage automatique. Elle consiste à transformer les valeurs numériques des différentes caractéristiques (ou variables) de notre jeu de données pour les ramener à une même échelle. L'objectif principal de cette opération est de garantir que toutes les caractéristiques aient une influence égale sur les modèles d'apprentissage.
La technique de mise à l'échelle que nous avons utilisée est la "Standardisation" ("StandardScaler"). Cette méthode transforme les valeurs de chaque caractéristique en soustrayant la moyenne de la caractéristique, puis en divisant par l'écart type. Ainsi, après la standardisation, chaque caractéristique aura une moyenne de zéro et un écart type de un. En mettant toutes les caractéristiques à la même échelle, nous évitons qu'une caractéristique avec une plus grande échelle domine les autres lors de la construction du modèle.

6.6 Implémentation

Dans la méthode proposée, l'étape d'implémentation où se fait l'apprentissage joue un rôle essentiel. Dans cette étape, nous expérimentons différents algorithmes ou techniques d'optimisation pour trouver le modèle qui offre les meilleures performances.
Dans notre cas d'étude, nous sommes confrontés à un problème de classification, et nous allons essayer plusieurs modèles de classification. Cela nous permettra de comparer les performances de ces modèles et de choisir celui qui s'adapte le mieux à notre jeu de données et à nos objectifs d'analyse. D'abord, nous avons divisé notre jeu de données en deux parties : un ensemble d'apprentissage et un ensemble de test. L'ensemble d'apprentissage est utilisé pour entraîner notre modèle, tandis que l'ensemble de test est utilisé pour évaluer ses performances. Pour assurer la reproductibilité des résultats, nous avons choisi aléatoirement 20% des données pour l'ensemble de test et fixé une graine (random state) à 42.

1. **Entraînement d'un modèle KNN** : Nous avons utilisé l'algorithme K-Nearest Neighbors (KNN) pour créer notre modèle de classification. Comme nous n'avons pas spécifié le paramètre k, il a été automatiquement fixé à 5, la valeur par défaut dans scikit-learn. Après avoir entraîné le modèle avec les données d'apprentissage, nous l'avons utilisé pour faire des prédictions sur l'ensemble de test, ce qui nous permet de voir comment le modèle se comporte sur de nouvelles données qu'il n'a pas encore vues.
2. **Optimisation du modèle KNN avec GridSearchCV** : Ensuite, nous avons procédé à l'optimisation du modèle KNN en utilisant la technique de recherche de grille avec validation croisée (GridSearchCV). Cette méthode nous a permis d'explorer différentes valeurs pour le paramètre k du modèle KNN, en l'évaluant avec des valeurs allant de 3 à 49. Pour chaque valeur de k, GridSearchCV a effectué une validation croisée pour évaluer les performances du modèle sur l'ensemble d'apprentissage. Après avoir parcouru toutes les valeurs de k, nous avons trouvé que le k optimal qui maximise l'exactitude sur l'ensemble d'apprentissage était k=45. Cela signifie que notre modèle KNN atteint la meilleure performance lorsqu'il considère les 45 voisins les plus proches pour effectuer ses prédictions.

   Une fois que nous avons identifié le meilleur paramètre k, nous avons instancié un nouveau modèle KNN avec cette valeur optimale et l'avons entraîné sur l'ensemble d'apprentissage.
3. **Entraînement d'un modèle de régression logistique (RL)** : Dans cette étape, nous avons utilisé la régression logistique en utilisant la classe LogisticRegression de scikit-learn pour former notre modèle sur l'ensemble d'entraînement et faire des prédictions sur l'ensemble de test.
4. **Entraînement d'un modèle SVM** : Nous avons entraîné SVM en utilisant la classe SVC (Support Vector Classification) de la bibliothèque scikit-learn. Ce modèle est utilisé pour résoudre des problèmes de classification. Nous avons ajusté le modèle aux données d'entraînement, puis nous l'avons utilisé pour faire des prédictions sur l'ensemble de test.
5. **Optimisation des hyperparamètres du modèle SVM** : Pour améliorer les performances du modèle SVM, nous avons procédé à une optimisation des hyperparamètres. Nous avons défini une grille de paramètres en spécifiant différentes valeurs pour 'C', 'gamma' et 'kernel'. Ensuite, nous avons utilisé GridSearchCV, une méthode de recherche exhaustive, pour explorer toutes les combinaisons spécifiées dans la grille des hyperparamètres. Une fois la recherche terminée, GridSearchCV nous a fourni les meilleurs hyperparamètres trouvés, qui sont C : 0.1, gamma : 0.1 et kernel : linear. Ces hyperparamètres ont été utilisés pour entraîner le modèle SVM final et faire des prédictions sur l'ensemble de test.

6. **Entraînement de modèle Random Forest (RF)** Dans cette partie, nous avons utilisé l'algorithme Random Forest pour créer un modèle de classification. Après entraînement du modèle sur l'ensemble d'entraînement, nous avons utilisé ce modèle pour faire des prédictions sur l'ensemble de test afin d'évaluer ses performances.

6.7 Évaluation

Les résultats des évaluations des différents modèles sont présentés dans le tableau 6.3. Chaque modèle a été évalué en termes de précision, de rappel, de F1-Score pour les deux classes : "Non-Shutdown" (C0) et "Shutdown" (C1).

Modèle	**Précision (C0)**	**Précision (C1)**	**Rappel (C0)**	**Rappel (C1)**	**F1-Score (C0)**	**F1-Score (C1)**	**Exactitude**
KNN	0.72	0.69	0.67	0.74	0.70	0.72	0.71
KNN (Optimisé)	0.72	0.69	0.67	0.74	0.70	0.72	0.74
RL	1.00	0.85	0.83	1.00	0.91	0.92	0.91
SVM	0.95	0.86	0.84	0.95	0.89	0.91	0.90
SVM (Optimisé)	1.00	0.88	0.87	1.00	0.93	0.94	0.93
RF	0.98	0.85	0.83	0.99	0.90	0.91	0.91

TABLE 6.3 – Résultats des évaluations des modèles

Le modèle SVM (Optimisé) obtient la meilleure précision avec un score de 1.00 pour la C0 et 0.88 pour la C1, montrant un excellent équilibre entre précision et rappel pour les deux classes. Il est suivi de près par le modèle de régression logistique qui atteint une précision de 1.00 pour la C0 et 0.85 pour la C1.
Le modèle Random Forest présente également de bonnes performances avec une précision globale de 91% et un F1-Score équilibré entre les deux classes.
En revanche, le modèle KNN (non optimisé) affiche la précision la moins élevée avec un score de 0.72 pour la C0 et 0.69 pour la C1.
Ces résultats confirment que le modèle SVM (Optimisé) est le plus performant pour notre tâche de classification binaire visant à prédire si une panne de pipeline se produira ou non. Il présente le meilleur équilibre entre précision et rappel pour les deux classes, ce qui en fait le choix le plus approprié pour notre application. De ce fait, nous passons à l'étape de la sérialisation du modèle SVM entraîné.

Sérialisation : La sérialisation est le processus de sauvegarde d'un modèle entraîné dans un format persistant, généralement sous la forme d'un fichier. L'objectif de la sérialisation est de permettre de recharger le modèle ultérieurement pour effectuer des prédictions sur de nouvelles données sans avoir besoin de ré-entraîner le modèle à partir de zéro. Nous avons utilisé la bibliothèque pickle pour sérialiser notre modèle SVM entraîné. La sérialisation nous permet de sauvegarder le modèle dans un fichier ".pkl" afin de pouvoir le recharger et l'utiliser dans notre application ultérieurement.

6.8 Réalisation de la fonctionnalité prédiction pour l'acteur Ingénieur

6.8.1 Description textuelle du cas d'utilisation

Le tableau 6.4 présente le cas d'utilisation "Prédire l'arrêt de pipelines".

Acteurs	Ingénieur
Description	Prédire l'arrêt de pipelines à partir des données fournis
Date	30/05/2023
pré-condition	l'ingénieur doit être authentifié
Description des scénarios	
Scénario nominal	**Prédire :** 1. L'ingénieur doit d'abord accéder à l'interface de prédiction d'arrêt de pipelines. 2. Après il doit remplir le formulaire. 3. Puis il clique sur le bouton prédire. 4. Après avoir vérifié les informations fournies, le système utilise le modèle SVM chargé pour effectuer une prédiction sur les données fournies.. 5. Le résultat de la prédiction est affiché à l'ingénieur sous forme de message, indiquant si l'arrêt du pipeline est prévu ou non. 6. les information de la prédiction sont sauvegarder à la base de données.
Scénario alternatif	**Cas "Emplacement invalide"** 3.1 Une boite de dialogue avec le message "Latitude et longitude invalides. Veuillez fournir un emplacement situé dans les États-Unis continentaux." s'affiche. **Cas "champs vides"** 3.1 Une boite de dialogue avec le message "champs vides. Veuillez remplir tous les champs" s'affiche.

TABLE 6.4 – Description textuelle du cas d'utilisation "Prédire l'arrêt de pipelines"

6.8.2 Diagramme d'interaction

La figure 6.2 représente les interactions entre l'Ingénieur et le système dans le cas de prédiction.

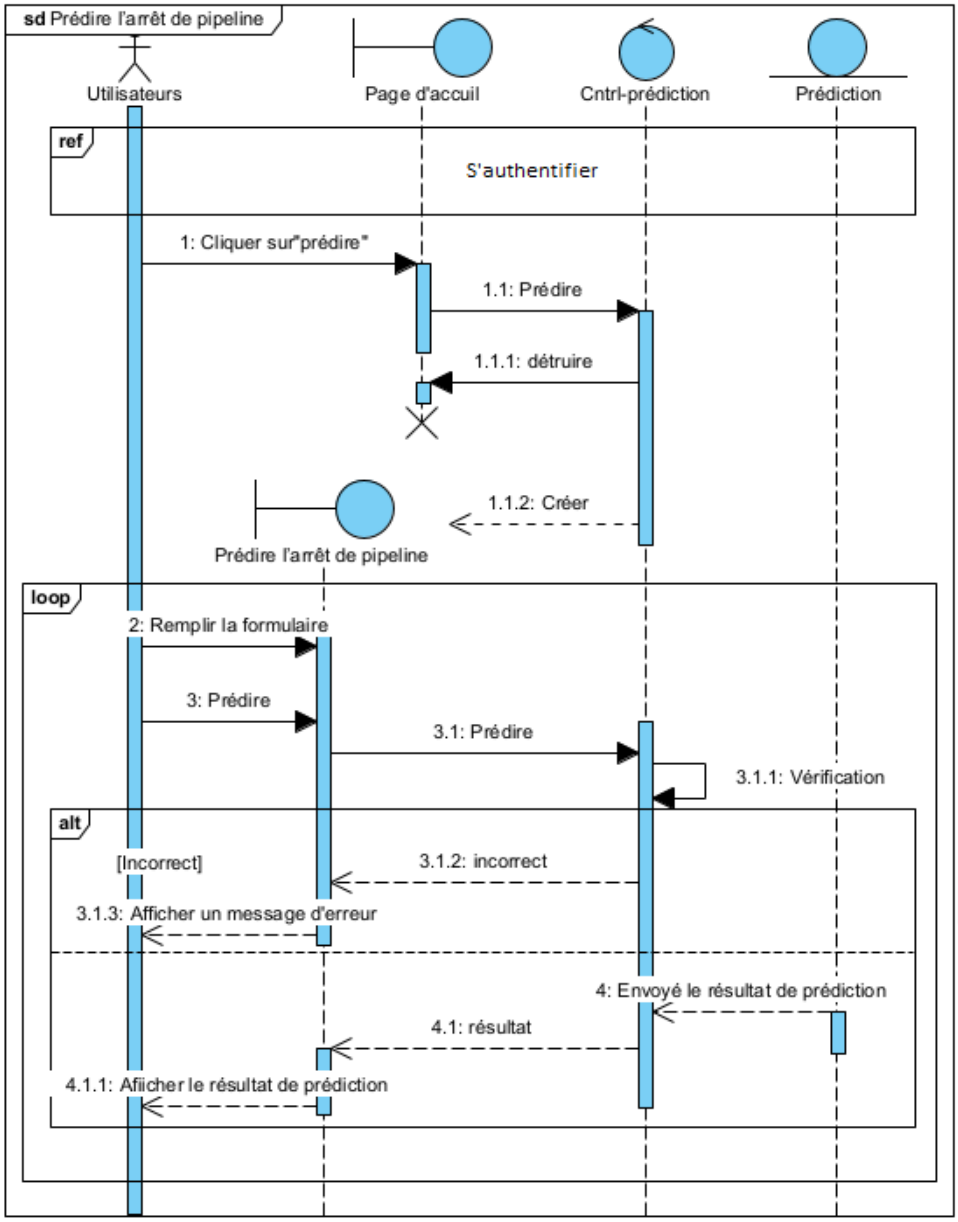

FIGURE 6.2 – Diagramme d'interaction pour "Prédire l'arrêt de pipelines"

6.8.3 Diagramme de classe

Le diagramme de classes de ce sprint est pratiquement le même que celui du sprint précédent. Nous avons simplement ajouté la classe "Prédiction" qui est reliée à la classe "Pipeline".

6.8.4 Passage au modèle relationnel

Dans cette section, nous allons construire le modèle relationnel ainsi que le dictionnaire de données à partir du diagramme de classe.

Modèle relationnel

D'après le diagramme de classe constitué précédemment voici le modèle relationnel correspondant :

Prediction (PredictionNumber, AccidentDateTime, PipelineType, LiquidType, AccidentState, AccidentLatitude, AccidentLongitude, CauseCategory, CauseSubcategory,

UnintentionalReleaseBarrels, LiquidIgnition, LiquidExplosion, PredictedShutdown, PredictionDateTime)

Users (ID,username,password,email)

Dictionnaire de données

Table utilisateur

id	INTEGER	"id" INTEGER NOT NULL UNIQUE
username	TEXT	"username" TEXT NOT NULL
password	TEXT	"password" TEXT NOT NULL
image	BLOB	"image" BLOB NOT NULL
email	TEXT	"email" TEXT NOT NULL

FIGURE 6.3 – Table utilisateur

Table prédiction

Prediction Number	INTEGER	"Prediction Number" INTEGER NOT NULL
Accident Date/Time	TEXT	"Accident Date/Time" TEXT
Pipeline Type	TEXT	"Pipeline Type" TEXT
Liquid Type	TEXT	"Liquid Type" TEXT
Accident State	TEXT	"Accident State" TEXT
Accident Latitude	REAL	"Accident Latitude" REAL
Accident Longitude	REAL	"Accident Longitude" REAL
Cause Category	TEXT	"Cause Category" TEXT
Cause Subcategory	TEXT	"Cause Subcategory" TEXT
Liquid Ignition	TEXT	"Liquid Ignition" TEXT
Liquid Explosion	TEXT	"Liquid Explosion" TEXT
Predicted Shutdown	TEXT	"Predicted Shutdown" TEXT
Prediction Date/Time	TEXT	"Prediction Date/Time" TEXT

FIGURE 6.4 – Table prédiction

6.8.5 Interface prédire l'arrêt de pipelines

La fonctionnalité de prédiction de l'arrêt de pipelines est représentée par l'interface de la figure 6.5 :

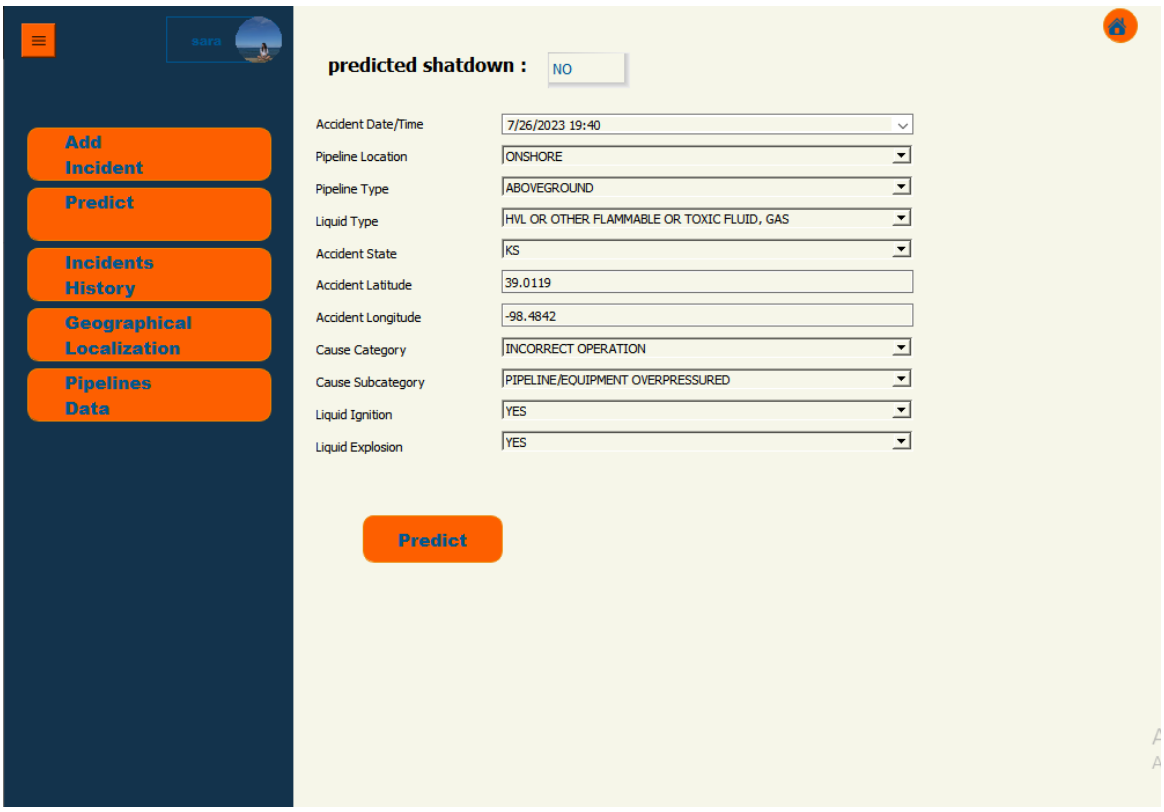

FIGURE 6.5 – Interface prédiction d'arrêt de pipelines

6.9 Évaluation de sprint

Lors de la réunion de revue, nous avons présenté les fonctionnalités développées pendant le sprint à notre client. Nous avons partagé les progrès réalisés et démontré le bon fonctionnement de la fonctionnalité de prédiction d'arrêt de pipelines. Le client a eu l'opportunité de donner son avis sur les fonctionnalités livrées et d'exprimer sa satisfaction quant à la qualité du travail accompli.

Conclusion

Dans le sprint 2, nous avons réussi à développer la fonctionnalité de prédiction d'arrêt de pipelines en appliquant notre méthode Scrum adaptée. Nous avons réalisé les tâches de prétraitement des données, d'apprentissage des modèles (KNN, régression logistique, SVM et Random Forest), d'ajustement des hyperparamètres et d'évaluation des performances. Nous avons également travaillé sur la modélisation des interfaces homme-machine (IHM) pour permettre aux utilisateurs d'accéder facilement à la fonctionnalité de prédiction. Grâce à nos efforts, nous avons atteint notre objectif principal et le sprint 2 a été un succès. Nous sommes maintenant prêts à passer à la prochaine phase de développement.

Chapitre 7

Sprint 3

Introduction

Dans le sprint 3, nous finalisons le projet avec la mise en place des fonctionnalités de planification de mission et de gestion des utilisateurs. Ces nouvelles fonctionnalités répondent aux besoins des utilisateurs pour une gestion efficace des incidents sur les pipelines. Ce sprint clôture notre cycle de développement avec une solution complète et prête à être déployée.

7.1 Réunion de planification

Lors de la réunion de planification du sprint 3, notre équipe s'est réunie pour discuter des fonctionnalités à développer dans ce sprint du projet. Les principales fonctionnalités à mettre en œuvre sont : la planification de missions, le suivi des missions, la consultation des missions à venir, la consultation des missions en cours et la gestion des utilisateurs.
En travaillant ensemble, nous avons créé un backlog détaillé pour chaque fonctionnalité, en établissant les User Stories associées, ainsi que les critères de finition (Definition of Done, DoD).

7.2 Diagramme de cas d'utilisation

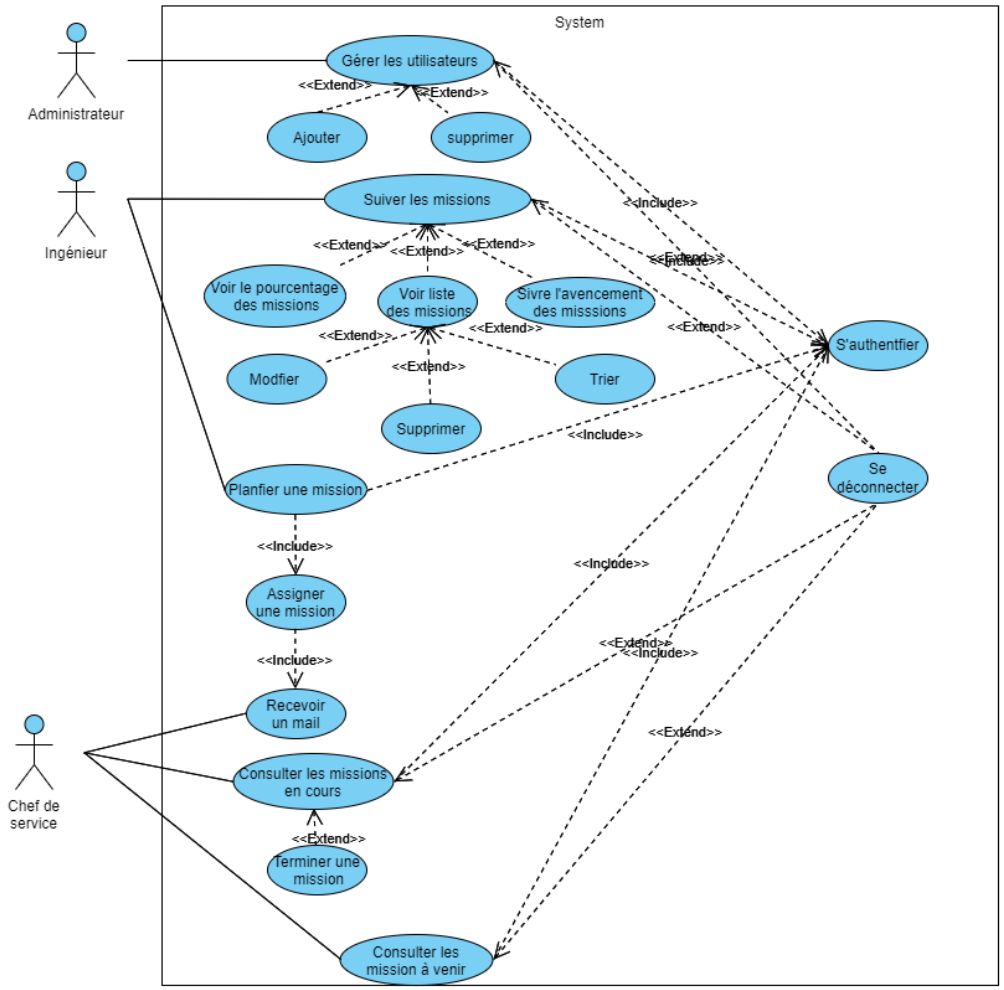

FIGURE 7.1 – Diagramme de cas d'utilisation sprint 3

7.3 Backlog de Sprint

Numéro de tâche	Description des tâches avec des User stories	Durée estimée	Qui réalise
1	En tant que membre de l'équipe de développement, je veux construire le diagramme de cas d'utilisation pour modéliser les fonctionnalités de ce sprint.	1h	Gali Kinza
2	En tant que membre de l'équipe de développement, je veux rédiger la description des cas d'utilisation , pour les expliquer.	1h	Gali Kinza
3	En tant que membre de l'équipe de développement, je veux construire les diagrammes d'interaction pour les cas d'utilisation, pour expliquer le scénario des fonctionnalité.	1h	Gali Kinza

Numéro de tâche	User stories	Durée estimée	Qui réalise
4	En tant que membre de l'équipe de développement, je veux construire le diagramme de classe, pour pouvoir réaliser le modèle logique de données.	1h	Gali Kinza
5	En tant que membre de l'équipe de développement, je veux construire le modèle logique de données, pour pouvoir implémenter la base de données.	1h	Gali Kinza
6	En tant que membre de l'équipe de développement, je veux implémenter la base de données, pour bien sauvegarder et gérer les données.	1h	Lagab Sara
7	En tant que membre de l'équipe de développement, je veux modéliser l'interface homme/machine de planification de mission, pour satisfaire le client.	2h	Lagab Sara
8	En tant que membre de l'équipe de développement, je veux modéliser l'interface homme/machine de suivi des missions, pour satisfaire le client.	2h	Lagab Sara
9	- En tant que membre d'équipe de développement, je veux modéliser l'interface homme/machine de gestion des utilisateurs, pour satisfaire le client.	2h	Lagab Sara
10	- En tant que membre d'équipe de développement, je veux programmer la fonctionnalité gérer les utilisateur , pour satisfaire le besoin de client.	2h	Lagab Sara
11	En tant que membre de l'équipe de développement, je veux programmer la fonctionnalité de planification de mission, pour satisfaire le besoin du client.	2h	Lagab Sara
12	En tant que membre de l'équipe de développement, je veux programmer la fonctionnalité de suivi des missions, pour satisfaire le besoin du client.	2h	Lagab Sara
13	En tant que membre de l'équipe de développement, je veux tester la fonctionnalité de planification de mission, pour garantir son bon fonctionnement.	2h	Lagab Sara
14	En tant que membre de l'équipe de développement, je veux modéliser l'interface homme/machine de consultation des missions à venir, pour satisfaire le client.	2h	Gali Kinza
15	En tant que membre de l'équipe de développement, je veux programmer la fonctionnalité de consultation des missions à venir, pour satisfaire le besoin du client.	2h	Gali Kinza
16	En tant que membre de l'équipe de développement, je veux tester la fonctionnalité de consultation des missions à venir, pour garantir son bon fonctionnement.	2h	Gali Kinza
17	En tant que membre de l'équipe de développement, je veux modéliser l'interface homme/machine de consultation des missions en cours, pour satisfaire le client.	2h	Lagab Sara

Numéro de tâche	User stories	Durée estimée	Qui réalise
18	En tant que membre de l'équipe de développement, je veux programmer la fonctionnalité de consultation des missions en cours, pour satisfaire le besoin du client.	2h	Lagab Sara
19	En tant que membre de l'équipe de développement, je veux tester la fonctionnalité de consultation des missions en cours, pour garantir son bon fonctionnement.	2h	Lagab Sara
20	En tant que membre de l'équipe de développement, je veux tester la fonctionnalité de suivi des missions, pour garantir son bon fonctionnement.	2h	Lagab Sara
21	Réunion de revue de sprint.	2h	Lagab Sara, Gali Kinza, Product Owner
22	Réunion de rétrospective de sprint.	2h	Lagab Sara, Gali Kinza
23	Rédaction de la documentation de sprint 3.	2 jours	Lagab Sara, Gali Kinza

TABLE 7.1 – Backlog de sprint 3

Voici la définition de "Done" pour les fonctionnalités du Sprint 3 :

Consulter les missions à venir :

- L'ingénieur peut accéder à la liste des missions à venir.
- Les missions à venir sont affichées avec leurs descriptions.
- Le temps restant avant le début de chaque mission est correctement calculé et affiché.

Consulter les missions en cours :

- L'ingénieur peut accéder à la liste des missions en cours.
- Les missions en cours sont affichées avec leurs descriptions.
- Le temps restant avant la fin de chaque mission est correctement calculé et affiché.

Planifier une mission :

- Le chef de service peut accéder à l'interface de planification de missions.
- Le chef de service peut choisir une date de début et une date de fin pour la mission.
- Le chef de service peut remplir la description de la mission et l'assigner à un ingénieur.
- Le système vérifie que la date de début est antérieure à la date de fin.
- Le chef de service peut valider la planification de la mission.

Gestion des Utilisateurs (Administrateur) :

- Les utilisateurs peuvent être ajoutés ou supprimés par l'administrateur.

— Les informations des utilisateurs sont correctement validées lors de l'ajout.

— L'interface utilisateur est intuitive et conviviale pour permettre une gestion aisée des utilisateurs.

— Les erreurs ou messages d'alerte sont correctement affichés en cas d'actions incorrectes.

7.4 Description textuelles des cas d'utilisation

Nous allons présenter seulement les cas d'utilisation "Planifier une mission", "Gérer les utilisateurs" et "Suivi des missions" par manque d'espace.

Cas d'utilisation "Planifier une mission"

Acteurs	Chef de service
Description	Planifier une mission
Date	1/06/2023
pré-condition	Le chef de service doit être authentifié
Description des scénarios	
Scénario nominal	**Planifier :** 1. Le chef de service accède à l'interface de planification de missions. 2. Il choisit une date de début et une date de fin pour la mission, remplit la description de la mission, et assigne la mission à un ingénieur. 3. Il valide la planification de la mission.
Scénario alternatif	**Cas "Date de début est antérieure à la date de fin "** 3.1 Si la date de début est antérieure à la date de fin, le système affiche un message d'erreur.

Table 7.2 – Description textuelle du cas d'utilisation "Planifier une mission"

Cas d'utilisation "Gérer les d'utilisateurs"

Acteurs	Administrateur
Description	Gérer les comptes des utilisateurs
Date	01/06/2023
pré-condition	l'administrateur doit être authentifié
Description des scénarios	
Scénario nominal	**Ajouter :** 1. L'administrateur accède à l'interface de gestion des utilisateurs. 2. Il remplit le formulaire avec les informations de l'utilisateur. 3. Si nécessaire, il choisit une photo de profil pour l'utilisateur à partir de son appareil. 4. Ensuite, il clique sur le bouton "Sauvegarder" pour ajouter le nouvel utilisateur. 5. Le système vérifie les informations fournies pour s'assurer qu'elles sont valides et complètes. 6. Une fois les informations vérifiées, le nouvel utilisateur est enregistré dans la base de données. 7. L'utilisateur ajouté apparaît désormais dans la liste des utilisateurs affichée sur l'interface de gestion. **Supprimer :** 1. L'administrateur accède à l'interface de gestion des utilisateurs. 2. Il fait défiler la liste des utilisateurs pour trouver celui qu'il souhaite supprimer. 3. Une fois qu'il a identifié l'utilisateur à supprimer, il clique sur le bouton "Supprimer" qui apparaît à côté de cet utilisateur dans la liste. 4. Le système supprime l'utilisateur de la base de données et met à jour la liste des utilisateurs affichée sur l'interface de gestion. 5. L'utilisateur supprimé n'apparaît plus dans la liste des utilisateurs.
Scénario alternatif	**Cas "Nom d'utilisateur existant :"** 4.1 Si le nom d'utilisateur existe déjà, le système affiche un message d'erreur à l'administrateur, l'informant que ce nom d'utilisateur est déjà pris.

TABLE 7.3 – Description textuelle du cas d'utilisation "gérer les d'utilisateurs"

Cas d'utilisation "Suivi des missions"

Acteurs	Chef de service
Description	Suivi des Missions dans le Temps
Date	02/05/2023
pré-condition	Le chef de service doit être authentifié

Description des scénarios	
Scénario nominal :	**Voir le pourcentage pour chaque type de mission** 1. Le chef de service accède à l'interface de suivi des missions. 2. Le système affiche une vue listant les différents types de missions (par exemple : "en cours", "à venir", "fini","fini en retard"). 3. Pour chaque type de mission, le système calcule et affiche le pourcentage de missions de ce type par rapport au total des missions. 4. Le chef de service peut visualiser ces pourcentages dans des graphes en anneau (ring graphs) pour avoir une vue d'ensemble. **Suivi de l'avancement des missions dans le temps** 1. Le chef de service accède à l'interface de suivi des missions. 2. Il sélectionne un utilisateur et une date de début. 3. Le système affiche un graphe linéaire d'avancement d'état des missions pour l'utilisateur sélectionné, depuis la date de début jusqu'à la date courante. **Consulter la liste des missions** 1. Le chef de service accède à l'interface de suivi des missions. 2. Le système affiche une liste des missions avec des détails tels que la description, la date de début, l'état, etc. 3. Le chef de service peut consulter la liste pour avoir une vue détaillée des missions. **Modifier une mission** 1. Dans la liste des missions, le chef de service sélectionne la mission qu'il souhaite modifier. 2. Il modifie les champs nécessaires (par exemple : la description, la date de début, l'ingénieur assigné). 3. En cliquant sur le bouton "Modifier", le système enregistre les changements et met à jour la mission. **Supprimer une mission** 1. Dans la liste des missions, le chef de service sélectionne la mission qu'il souhaite supprimer. 2. En cliquant sur le bouton "Supprimer" correspondant à cette mission, le système confirme la suppression. 3. Une fois confirmée, la mission est supprimée de la liste.

	Trier les missions 1. Le chef de service peut utilise les options de tri par date de début et par état dans la liste des missions. 2. En sélectionnant une option de tri, le système réorganise la liste des missions en conséquence.
Scénario alternatif :	Aucun

TABLE 7.4 – Description textuelle du cas d'utilisation "Suivi des Missions"

7.5 Diagrammes d'interaction

Nous donnons les diagrammes d'interaction correspondants aux cas d'utilisation décrits précédemment.

Diagramme d'interaction pour "Planifier une mission"

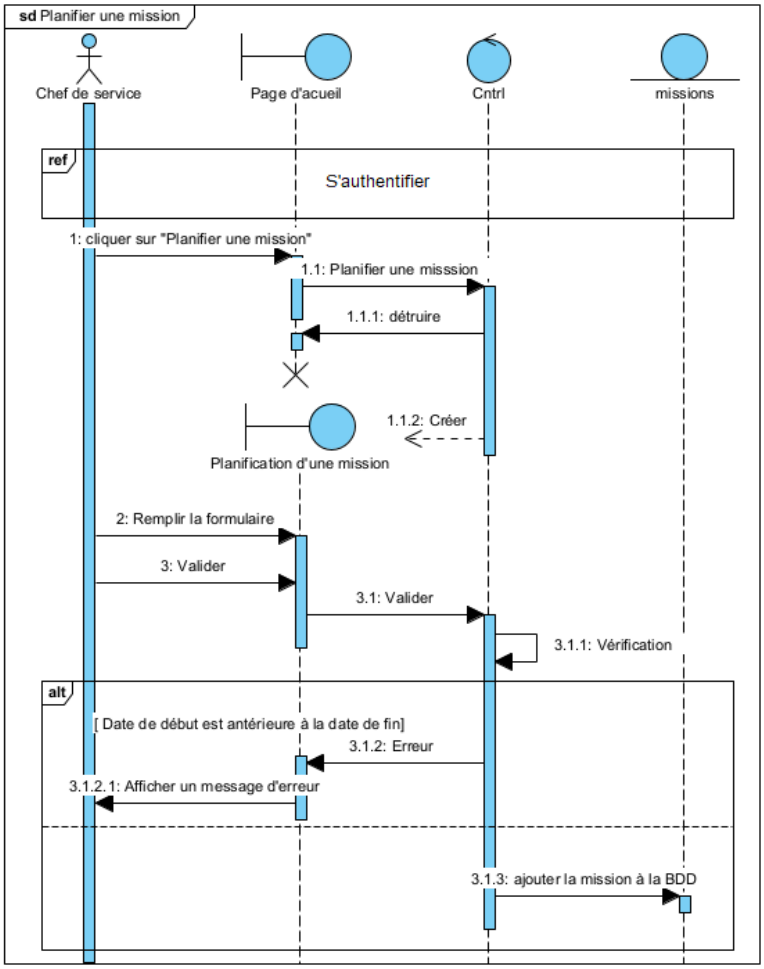

FIGURE 7.2 – Diagramme d'interaction pour"Planifier une mission"

Diagramme d'interaction pour"Suivi des mission"

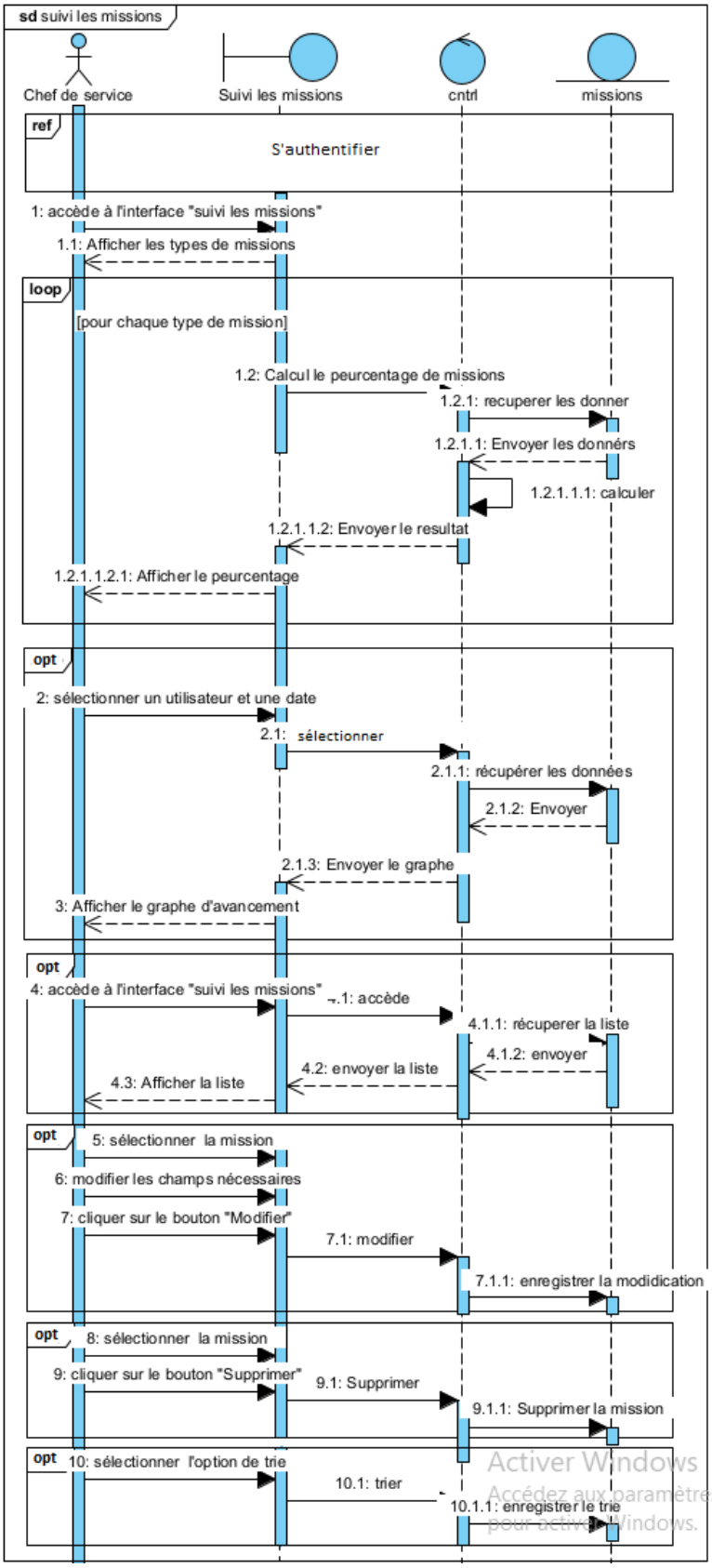

FIGURE 7.3 – Diagramme d'interaction pour"suivi des mission"

Diagramme d'interaction pour "Gérer les utilisateurs"

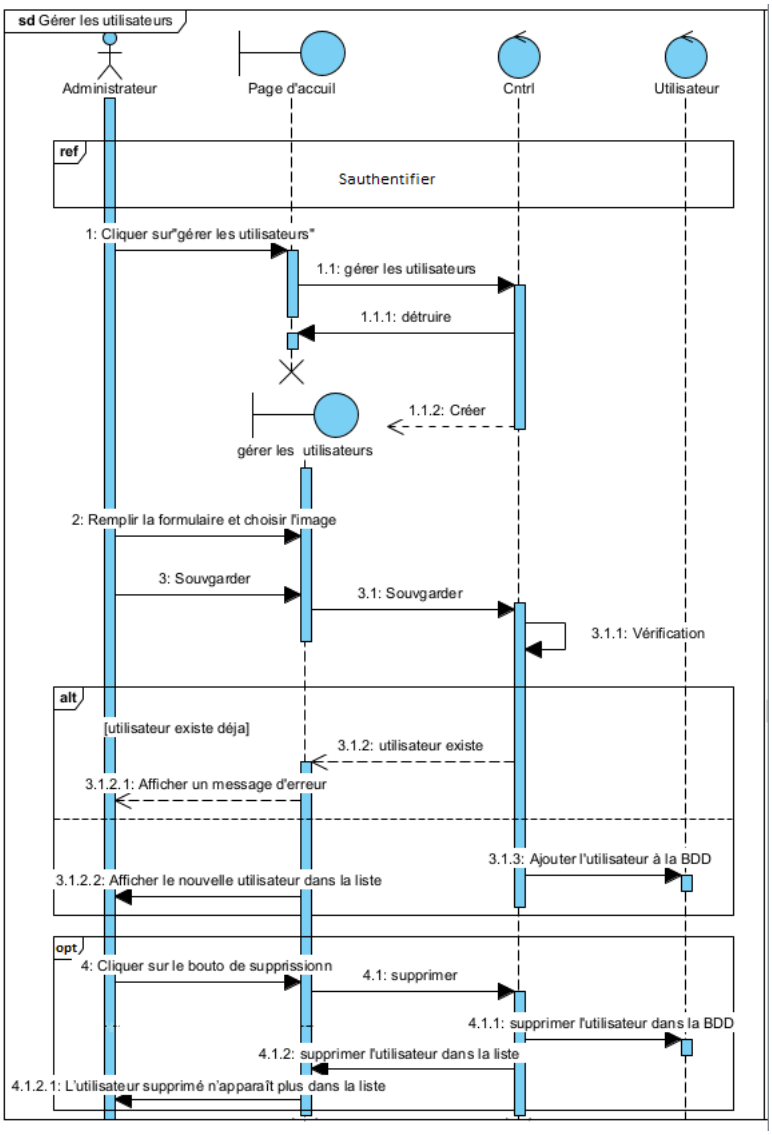

FIGURE 7.4 – Diagramme d'interaction pour "Gérer les utilisateurs

7.6 Diagramme de classe

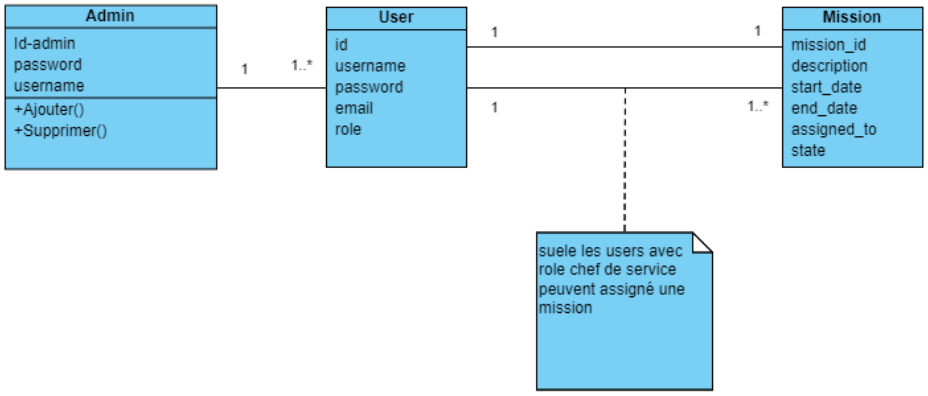

FIGURE 7.5 – Diagramme de classe de sprint4

7.7 Passage au modèle relationnel

Dans cette section, nous allons construire le modèle relationnel ainsi que le dictionnaire de données correspondant au sprint 3.

Modèle relationnel

D'après le diagramme de classe, nous pouvons constater deux entités : "Mission" et "User". Pour l'entité "User" (utilisateur) que nous avons déjà vue dans les sprints précédents, nous avons ajouté un attribut "rôle" qui permet de différencier entre le chef de service et l'ingénieur.

Admin (Id_admin, password, username)

Mission (mission_id, description, start_date, end_date, assigned_to,state)

Users(ID,username,password,email,rôle)

Dictionnaire de données

Table utilisateur

id	INTEGER	"id" INTEGER NOT NULL UNIQUE
username	TEXT	"username" TEXT NOT NULL
password	TEXT	"password" TEXT NOT NULL
image	BLOB	"image" BLOB NOT NULL
email	TEXT	"email" TEXT NOT NULL
role	TEXT	"role" TEXT

FIGURE 7.6 – Table utilisateur

Table administrateur Table mission

id-admin	INTEGER	"id-admin" INTEGER NOT NULL UNIQUE
username	TEXT	"username" TEXT NOT NULL UNIQUE
password	TEXT	"password" TEXT NOT NULL

FIGURE 7.7 – Table administrateur

mission_id	INTEGER	"mission_id " INTEGER NOT NULL
description	TEXT	"description" TEXT
start_date	TEXT	"start_date" TEXT
end_date	TEXT	"end_date" TEXT
assigned_to	INTEGER	"assigned_to" INTEGER
state	TEXT	"state" TEXT CHECK("state" IN ('ongoing', 'completed', 'pending', 'done late'))

FIGURE 7.8 – Table mission

7.8 Interfaces Hommes/Machines

Interface planification de missions

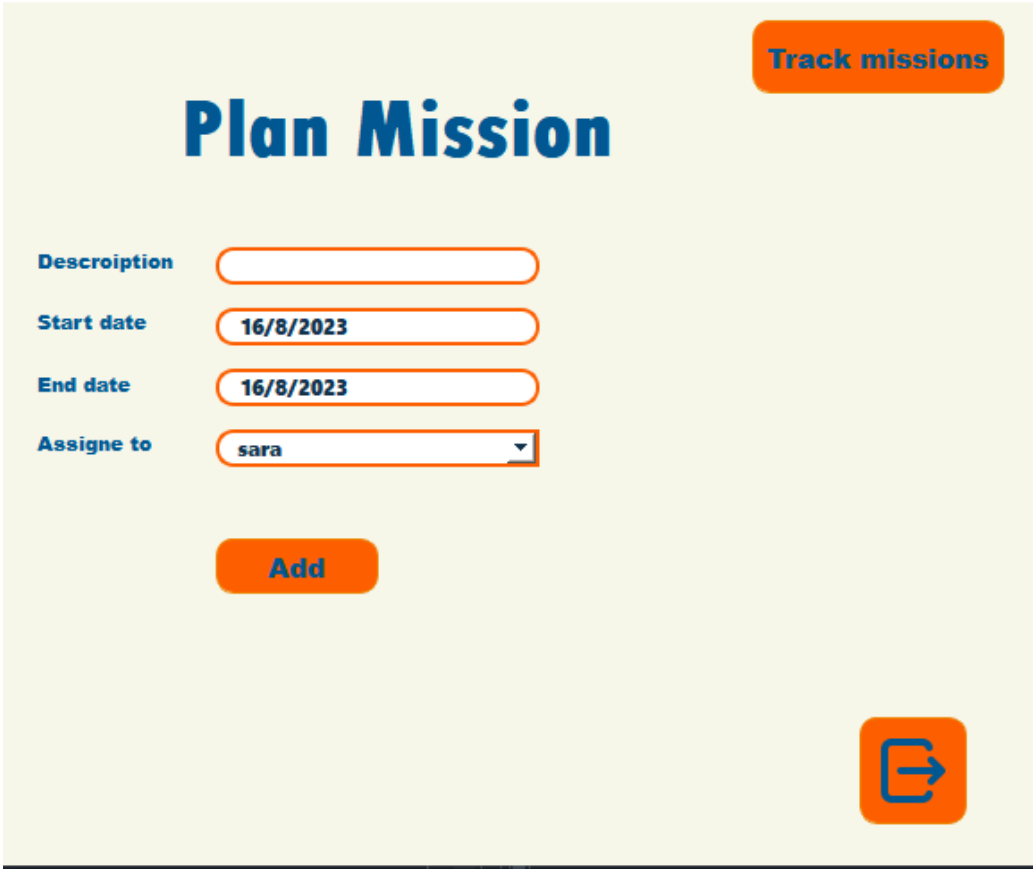

FIGURE 7.9 – Interface planification de missions

Interface suivi de missions

FIGURE 7.10 – Interface suivi de missions

Interface gestion des utilisateurs

FIGURE 7.11 – Interfaces gestion des utilisateurs

Interface "Page d'accueil"

FIGURE 7.12 – Interface page d'accueil

7.9 Évaluation de sprint

Dans le cadre de l'évaluation du sprint 3, notre équipe a également conduit deux réunions cruciales : la réunion de revue et la réunion de rétrospective. Lors de la réunion de revue, nous avons présenté les nouvelles fonctionnalités que nous avons développées, à savoir la gestion des utilisateurs avec un nouvel acteur "Administrateur", la planification de missions,le suivi d'avancement des missions , la consultation des missions à venir et la consultation des missions en cours. Nous avons exposé en détail ces nouvelles capacités au client et recueilli ses retours.
Par la suite, lors de la réunion de rétrospective, nous avons pris du recul pour évaluer notre performance au cours de ce sprint. Nous avons identifié les aspects positifs qui ont contribué au succès de nos efforts, mais également les domaines dans lesquels nous pourrions améliorer notre approche. En discutant ouvertement des enseignements tirés de ce sprint, nous avons déterminé des actions correctives pour optimiser davantage notre processus de développement et mieux répondre aux besoins futurs du projet.

Conclusion

Le sprint 3 marque la fin de notre parcours de développement, et il représente un aboutissement significatif pour notre équipe. Au cours de ce sprint, nous avons introduit des fonctionnalités essentielles pour l'achèvement de notre projet. La gestion des utlisateurs, la planification de missions,le suivi d'avencement des missions,la consultation des missions à venir et la consultation des missions en cours ont été mises en place avec succès, offrant une gestion complète des missions pour les utilisateurs.

Conclusion générale

L'objectif principal de ce mémoire était d'introduire de l'agilité dans le processus de développement de systèmes intelligents. Pour cela, nous avons, tout d'abord, exploré les systèmes intelligents et compris que leur rôle est essentiel dans les entreprises et plus spécialement dans la prise de décision. En outre, les systèmes intelligents font appel aux techniques d'apprentissage automatique pour l'extraction et le traitement des données complexes, ainsi que pour l'amélioration de la qualité des décisions. Ensuite, nous avons examiné les méthodes de développement logiciel et constaté qu'il existe plusieurs types dont les méthodes classiques ont montré leurs limites en termes de flexibilité et d'adaptabilité aux changements, contrairement aux méthodes agiles qui ont émergé comme une réponse efficace grâce à leur approche souple et itérative.

C'est deux études révèlent que les méthodes de développement ne répondent pas aux spécificités des systèmes intelligents, néanmoins les méthodes agiles présentent plus d'avantages, en particulier leur capacité à s'ajuster aux exigences changeantes. Ainsi, ce travail a proposé une adaptation de la méthode agile Scrum par l'ajout de sprints d'apprentissage pour le développement de systèmes intelligents tout en bénéficiant de la flexibilité des méthodes agiles.

Nous avons mis oeuvre l'approche proposée pour développer une application d'aide à la décision pour l'entreprise SONATRACH. Les cycles itératifs de l'apprentissage, comprenant le pré-traitement et la conceptualisation des données, ont appliqué pour la réalisation de la fonctionnalité "prédiction d'arrêt de pipelines". Les autres fonctionnalités de l'application ont été également réalisées pour répondre aux besoins exprimés.

Enfin, ce mémoire offre une opportunité intéressante pour la communauté informatique sur l'application des méthodes agiles dans le développement des systèmes intelligents. Voici quelques perspectives que nous envisageons pour ce projet :

- **Optimisation des Modèles :** Poursuivre la recherche pour optimiser les modèles d'apprentissage automatique utilisés. Cela pourrait inclure l'exploration de techniques d'optimisation avancées ou l'expérimentation avec des architectures de réseaux neuronaux plus complexes.
- **Interactivité Utilisateur :** Développer des fonctionnalités interactives pour l'interface utilisateur permettant à ce dernier de personnaliser davantage les prédictions en fonction de ses besoins spécifiques.

Bibliographie

[1] 16th state of agil report. https://info.digital.ai/rs/981-LQX-968/images/SOA16.pdf. Consulté le 8 septembre 2023.

[2] Agile alliance. https://www.agilealliance.org/. Consulté le 1 avril 2023.

[3] Categorical data. *Towards Data Science*.

[4] Centre national de ressources textuelles et lexicale. https://www.cnrtl.fr/definition/m%C3%A9thode. Consulté le 1 juin 2023.

[5] Cyclical features encoding. *Towards Data Science*.

[6] Diagramme de classes uml - lucidchart. https://www.lucidchart.com/pages/fr/diagramme-de-classes-uml. Consulté le 3 juin 2023.

[7] History : The agile manifesto. https://agilemanifesto.org. Consulté le 3 juin 2023.

[8] Kaggle. https://www.kaggle.com. Consulté le 22 mai 2023.

[9] Méthodologie xp. http://tvaira.free.fr/dev/methodologie/methodologieXP.pdf. Consulté le 30 juin 2023.

[10] Scrum.org. https://www.scrum.org/. Consulté le 5 avril 2023.

[11] Sqlite browser. https://sqlitebrowser.org/. Consulté le 5 juiilet 2023.

[12] Visual paradigm - uml, agile, pmbok, togaf, bpmn and more ! https://www.visual-paradigm.com/. Consulté le 13 juin 2023.

[13] Extreme programming. https://www.extremeprogramming.org/, 2022. Consulté le 15 juillet 2023.

[14] Elastic email, 2023. Consulté le 3 juillet 2023.

[15] Abdelkader Adla. *Aide à la Facilitation pour une prise de Décision Collective : Proposition d'un Modèle et d'un Outil*. PhD thesis, Université Paul Sabatier-Toulouse III, 2010.

[16] Randa Al-Maghraby and Mohammad Alshayeb. Extreme programming methodology : A literature review. *International Journal of Computer Applications*, 175(9), 2017.

[17] Laurent Alexandre. *Intelligence Artificielle : vers une domination programmée ?* Éditions Robert Laffont, 2018.

[18] Algotive. Systèmes intelligents : Ce qu'ils sont, comment ils fonctionnent et pourquoi ils sont si importants, April 2023. 01/09/2023.

[19] Emmanuel Ameisen. *Building Machine Learning Powered Applications*. O'Reilly Media, 2020.

[20] Inc. Anaconda. Anaconda distribution (version 2021.05). https://www.anaconda.com/products/distribution, 2021. Consulté le 04 juillet 2023.

[21] Sondra Ashmore and Kristin Runyan. *Introduction to agile methods*. Addison-Wesley Professional, 2014.

[22] L. Atif. *Une approche Collaborative d'Analyse des Besoins et des Exigences Dirigée par les Problèmes : Le Cas de Développement d'une Application Analytics RH*. Paris, France, 2017.

[23] Claude Aubry. *Scrum-3e éd. : Le guide pratique de la méthode agile la plus populaire*. Dunod, 2013.

[24] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. Gambling in a rigged casino : The adversarial multi-armed bandit problem. In Sara A. Solla, Todd K. Leen, and Klaus-Robert Müller, editors, *Advances in Neural Information Processing Systems 7 (NIPS 1994)*, pages 322–329. MIT Press, 1995.

[25] Neculai Bagiu, Silvia Avasilcai, and Bogdan Rusu. Traditional vs. agile : The challenge mindset shift in project management. 21 :301–311, 12 2022.

[26] Kent Beck. *Extreme Programming Explained : Embrace Change*. Addison-Wesley Professional, 2nd edition, 2000.

[27] Kent Beck. *Extreme Programming Explained : Embrace Change*. Addison-Wesley Professional, 2000.

[28] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks : Tricks of the trade*, pages 437–478. Springer, 2012.

[29] Akim Berkani. *Dynamiques de généralisation des méthodes agiles de gestion de projet : une analyse processuelle de quatre organisations complexes*. PhD thesis, Université Paris sciences et lettres, 2020.

[30] Michael Blaha. *Object-Oriented Modeling and Design with UML : For VTU, 2/e*. Pearson Education India, 2005.

[31] Anca Boboc and Jean-Luc Metzger. Les méthodes agiles et leurs contradictions. analyse de leurs effets sur les métiers de l'informatique. *SociologieS*, 2020.

[32] Barry Boehm. Get ready for agile methods, with care. *Computer*, 35(1) :64–69, 2002.

[33] Barry Boehm and Richard Turner. Using risk to balance agile and plan-driven methods. *Computer*, 36(6) :57–66, 2003.

[34] Barry Boehm and Richard Turner. Management challenges to implementing agile processes in traditional development organizations. *IEEE software*, 22(5) :30–39, 2005.

[35] Barry W. Boehm. A spiral model of software development and enhancement. *Computer*, 21(5) :61–72, 1988.

[36] AIT DAHMANE B BOUKEMOUCHE M, BENKHIDER N. Vers les méthodes agiles de gestion des projets : Cas de la sonatrach (rtc) bejaia, 2021.

[37] Chris Bubernak and Marc Schweikert. Lean software development. Presentation at CSCI 5828, March 2012. Consulté le 23 Avril 2023.

[38] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168, 2006.

[39] B. Chabbat. *Modélisation multiparadigme de textes réglementaires*. PhD thesis, Insa de Lyon, 1997.

[40] H. Chen, R. H. Chiang, and V. C. Storey. Business intelligence and analytics : from big data to big impact. *MIS quarterly*, pages 1165–1188, 2012.

[41] Alistair Cockburn. *Crystal Clear : A Human-Powered Methodology for Small Teams*. Addison-Wesley Professional, 2004.

[42] Alistair Cockburn and Jim Highsmith. Agile software development, the people factor. *Computer*, 34(11) :131–133, 2001.

[43] Edgar F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6) :377–387, 1970.

[44] Thomas Collonville. Les processus de développement ou cycles de vie du logiciel. `https://collonvillethomas.wordpress.com/2017/12/23/les-processus-de-developpement-ou-cycles-de-vie-du-logiciel/`, December 2017. Consulté le 5 février 2023.

[45] Communications. Learning loop for adopting agile. 1 2016. Consulté le 6 avril 2022.

[46] The Qt Company. Qt documentation. 2021. Consulté le 5 juillet 2023.

[47] DataScientest. K-nearest neighbors (knn), 2023. Consulté le 2 septembre 2023.

[48] C.J. Date. *An Introduction to Database Systems*. Pearson Education, 2005.

[49] L. Derczynski and E. Gaillard. Traitement automatique des langues. In A. Laurent et al., editors, *Encyclopédie de l'informatique et des systèmes d'information*, volume 2, pages 1–21. Hermes Science Publications, 2018.

[50] Leon Derczynski and Emmanuel Gaillard. Traitement automatique des langues. In Annie Laurent and et al., editors, *Encyclopédie de l'informatique et des systèmes d'information*, volume 2, pages 1–21. Hermes Science Publications, 2018.

[51] Gregory Ditzler, Manuel Roveri, Cesare Alippi, and Robi Polikar. Learning in nonstationary environments : A survey. *IEEE Computational Intelligence Magazine*, 10(4) :12–25, 2015.

[52] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10) :78–87, 2012.

[53] Joyce Duncan, Lesley Rackley, and Alexandria Walker. *SSADM in practice : a version 4 text*. Springer, 1995.

[54] Anass El-Haddadi, Said El Hazzat, and Mohamed Mahraz. Classification des systèmes intelligents et leurs domaines d'application. *Revue Africaine de la Recherche en Informatique et Mathématiques Appliquées*, 26 :57–72, 2017.

[55] Bernard ESPINASSE. Typologie des outils et méthodes de spécification en génie logiciel. 2008.

[56] Figma. Figma - the collaborative interface design tool, 2021. Consulté le 5 juillet 2023.

[57] Interaction Design Foundation. What is interaction design ? `https://www.interaction-design.org/literature/topics/interaction-design`, 2021. Consulté le 5 juin 2023.

[58] W Wayt Gibbs. Software's chronic crisis. *Scientific american*, 271(3) :86–95, 1994.

[59] G. A. Gorry and M. S. Morton. *A framework for management information systems*, volume 13. 1971.

[60] Muki Haklay and Patrick Weber. Openstreetmap : Collaborative mapping. *The Geographic Information Science & Technology Body of Knowledge (2nd Quarter 2017 Edition)*, 2017.

[61] Harry Hall. *Agile Waterfall : A Practical Guide to Using Agile with Waterfall Project Management Methodology*. CreateSpace Independent Publishing Platform, 2016.

[62] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2009.

[63] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning : Data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[64] Véronique Heiwy. Méthodes agiles, conception centrée utilisateurs : hybridation gagnante pour les projets innovants et pluridisciplinaires ? une application au cas du développement d'objets connectés. In *ERGO'IA 2018*, 2018.

[65] Lamia Ben Hiba and Mohammed Abdou Janati Idrissi. Tendances des méthodes de gestion des projets informatiques. *E-Ti : E-Review in Technologies Information*, 2012.

[66] S. Holtzman. *Decision support systems : A survey*, volume 3. 1989.

[67] J. D. Hunter. Matplotlib : A 2d graphics environment. *Computing in Science & Engineering*, 9(3) :90–95, 2007.

[68] IBM. K-nearest neighbors (knn). Consulté le 2 juin 2023.

[69] IBM. Logistic regression. Consulté le 6 septembre 2023.

[70] Grace Darlene Inakabanga. *La cohabitation entre les méthodes agiles et traditionnelles*. PhD thesis, Université du Québec à Trois-Rivières, 2020.

[71] Inconnu. Comprendre comment fonctionne un random forest en 5 min. *Devoteam France*. Consulté le 03 août 2023.

[72] PROJETS INFORMATIQUES. Yende raphael grevisse, ph. d. 2019.

[73] Internet Engineering Task Force. RFC 5321 – simple mail transfer protocol (smtp). `https://tools.ietf.org/html/rfc5321`, 2008. Consulté le 5 juillet 2023.

[74] Ivar Jacobson, Magnus Christerson, and Jean-Marie André. *Le génie logiciel orienté objet : une approche fondée sur les cas d'utilisation*. Addison-Wesley France, 1993.

[75] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning : with Applications in R*. Springer, 2013.

[76] Élyane. Jourdenais-Lemaire. *L'utilisation des méthodes agiles comme méthode de gestion de projet en communication stratégique*. PhD thesis, Faculté des sciences économiques, sociales, politiques et de communication, Université catholique de Louvain, 2020.

[77] Harold Kerzner. *Project management 2.0 : leveraging tools, distributed collaboration, and metrics for project success*. John Wiley & Sons, 2015.

[78] Carine Khalil. *Les méthodes" agiles" de management de projets informatiques : une analyse" par la pratique"*. PhD thesis, Télécom ParisTech, 2011.

[79] D. A. Klein. Integrating artificial intelligence and decision theory to forecast new markets. Technical report, IBM Research Division, Yorktown Heights, 1988.

[80] Alexander Kossiakoff, William N. Sweet, Samuel J. Seymour, and Steven M. Biemer. *Systems Engineering : Principles and Practice*. John Wiley & Sons, 2011.

[81] Corey Ladas. *Scrumban : Essays on Kanban Systems for Lean Software Development*. Modus Cooperandi Press, 2009.

[82] Dr HAMZA Lamia. cour de méthodologie de dévloppement d'un si.

[83] Craig Larman and Bas Vodde. *Large-scale scrum : More with LeSS*. Addison-Wesley Professional, 2020.

[84] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553) :436–444, 2015.

[85] Roderick J. A. Little and Donald B. Rubin. *Statistical analysis with missing data*. John Wiley & Sons, Hoboken, NJ, 2nd edition, 2014.

[86] Hela Ltifi. *Démarche centrée utilisateur pour la conception de SIAD basés sur un processus d'Extraction de Connaissances à partir de Données, Application à la lutte contre les infections nosocomiales*. PhD thesis, Université de Valenciennes et du Hainaut-Cambresis ; Ecole Nationale d ..., 2011.

[87] Alan MacCormack, Roberto Verganti, and Marco Iansiti. Developing products on "internet time" : The anatomy of a flexible development process. *Management science*, 47(1) :133–150, 2001.

[88] Mapbox. Mapbox, 2021. Consulté le 3 mars 2023.

[89] G. M. Marakas. *Decision support systems in the 21st century*. Pearson Education, 2003.

[90] Wes McKinney. Pandas : a foundational python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, 14(9) :7–10, 2011.

[91] Mozilla Developer Network (MDN). Javascript, 2021. Consulté le 10 juillet 2023.

[92] Zahir Megri and Mohamed Rahmani. Conception et réalisation d'une application d'apprentissage en ligne de la programmation par une évaluation automatisée. 2017.

[93] Ministère de l'Économie et des Finances. Intelligence artificielle, 2022. Consulté le 5 Avril 2023.

[94] Henry Mintzberg. *The structuring of organization*. Prentice-Hall, 1979.

[95] Tom M. Mitchell. *Machine Learning*. WCB McGraw-Hill, 1997.

[96] Martin Molina. What is an intelligent system ? *arXiv eprint arXiv :2009.09083*, 2022.

[97] Nils J. Nilsson. *Principles of Artificial Intelligence*. Morgan Kaufmann, 1984.

[98] Votre nom. Description of rewards and penalties in reinforcement learning. Document non publié, Année. Description personnelle.

[99] Jean-Louis Peaucelle. Corig, histoire d'une méthode en informatique de gestion. *Systèmes d'information et management*, 7(3) :5, 2002.

[100] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn : Machine learning in python. *Journal of Machine Learning Research*, 12 :2825–2830, 2011.

[101] Fernando Pérez and Brian E. Granger. Ipython : a system for interactive scientific computing. *Computing in Science & Engineering*, 9(3) :21–29, 2007.

[102] Pablo Pernot. Lean software development mise en oeuvre. Version 0.8, September 2014. Initialisation September 2014 - Mise à jour September 2015.

[103] Gauthier Picard. Introduction à l'extreme programming et au développement agile. Presentation, October 2009. SMA/G2I/ENS Mines Saint-Etienne.

[104] Jean-Charles Pomerol. *Systèmes d'aide à la décision : Méthodes, concepts et outils*. Éditions Economica, 1990.

[105] Mary Poppendieck and Tom Poppendieck. *Implementing Lean Software Development : From Concept to Cash*. Addison-Wesley Professional, 2006.

[106] D. J. Power. *A brief history of decision support systems*, volume 44. 2008.

[107] D.J. Power. *Decision Support Systems : Concepts and Resources for Managers*. Greenwood Publishing Group, 2002.

[108] Project Management Institute. *PMI Agile Practice Guide*. Project Management Institute, Newtown Square, PA, USA, 2017.

[109] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 1–35. Springer US, 2011.

[110] Scott Robinson. Folium : Python data. visualize. leaflet.js maps. *The Journal of Open Source Software*, 3(21), 2018.

[111] Véronique Messager Rota. *Gestion de projet Vers les méthodes agiles.* Editions Eyrolles, 2011.

[112] Bernard Roy and Denis Bouyssou. *Aide multicritère à la décision.* Economica, 1993.

[113] Gilles Roy. *Conception de bases de données avec UML.* PUQ, 2007.

[114] WW Royce. Managing the development of large systems : Concepts and techniques. In *9th International Conference on Software Engineering. ACM*, pages 328–38, 1970.

[115] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv :1609.04747*, 2016.

[116] James Rumbaugh. What is a method ? *Journal of Object Oriented Programming*, 8 :10–10, 1995.

[117] Nayan B Ruparelia. Software development lifecycle models. *ACM SIGSOFT Software Engineering Notes*, 35(3) :8–13, 2010.

[118] Stuart J. Russell and Peter Norvig. *Artificial Intelligence : A Modern Approach.* Prentice Hall, 3rd edition, 2010.

[119] Andrew P Sage. Decision support systems : An open framework. *Decision Support Systems*, 7(3) :287–297, 1991.

[120] Arthur L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3) :210–229, 1959.

[121] Jake Schwaber and John Zeratsky Knapp. *Sprint : How to Solve Big Problems and Test New Ideas in Just Five Days.* Simon and Schuster, 2017.

[122] K. Schwaber and J. Sutherland. The scrum guide. `https://www.scrum.org/resources/scrum-guide`, 2020. Consulté le 3 juin 2023.

[123] Ken Schwaber and Jeff Sutherland. The scrum guide. `https://scrumguides.org/scrum-guide.html`, 2020. Consulté le 23 mars2023.

[124] R. Shapira and R. Kalakota. *Decision support and business intelligence systems.* Pearson, 2016.

[125] H. A. Simon. *The new science of management decision.* Prentice-Hall, 1977.

[126] John Smith. Classification of intelligent systems based on autonomy and complexity. *Journal of Artificial Intelligence Research*, 25(2) :123–145, 2020.

[127] Sonatrach. présantation de département entrain linges et bacs de stockage.

[128] Romaric Sossa. Méthodologie de projets de développement agile dans un environnement paas. Master's thesis, Ecole de technologie superieure, 2017.

[129] Spyder IDE. À propos de spyder. `https://www.spyder-ide.org/`. Consulté le 4 juin 2023.

[130] John D. Sterman. *Business Dynamics : Systems Thinking and Modeling for a Complex World.* Irwin McGraw-Hill, Boston, 2000.

[131] Mark Summerfield. *C++ GUI programming with Qt 4.* Prentice Hall, 2007.

[132] La Rédaction TechTarget. Système intelligent. Site web, 2018. Consulté le 12 mai 2023.

[133] E. Turban. *Decision support and expert systems : Management support systems.* Prentice-Hall, 1993.

[134] E. Turban, J. E. Aronson, and T. P. Liang. *Decision support systems and intelligent systems*, volume 7. Pearson Education, 2008.

[135] Efraim Turban, Ramesh Sharda, and Dursun Delen. *Decision Support and Business Intelligence Systems.* Pearson, 10th edition, 2018.

[136] ULT - Université Libre de Tunis. Chapitre II : Présentation du Langage de Modélisation UML. https://ult.bi/fr/chapitre-ii-presentation-du-langage-de-modelisation-uml, Consulté le 1er juin 2023.

[137] Vladimir N Vapnik and A Ya Chervonenkis. *Theory of Pattern Recognition.* Nauka, USSR, 1974.

[138] Norha M Villegas, Hausi A Müller, Gabriel Tamura, Laurence Duchien, and Rubby Casallas. A framework for evaluating quality-driven self-adaptive software systems. In *Proceedings of the 6th international symposium on Software engineering for adaptive and self-managing systems*, pages 80–89, 2011.

[139] Volle. Fonctionnement d'un système informatique d'aide à la décision (siad). In *Extraction et gestion des connaissances (EGC 2005)*. CEPAD, 2005.

[140] Dave Wells. What is extreme programming (xp) ? a methodology for agile software development. https://www.smartsheet.com/content/what-extreme-programming-xp-methodology-agile-software-development, 2019. Consulté le 5 Avril 2023.

[141] Patrick Henry Winston. *Artificial Intelligence.* Addison-Wesley Publishing Company, 1984.

[142] Haibo Yang, Sid Huff, and Diane Strode. Leadership in software development : Comparing perceptions of agile and traditional project managers. volume 3, page 184, 01 2009.

[143] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2017.

[144] IPI Écoles. La gestion de projet en mode hybride se dÉmocratise, mars 2018. Consulté le 2 avril 2023.