Université De Tunis El Manar
Faculté des Sciences Economiques et de Gestion de
-Tunis -

Rapport de Projet de Fin d'Études

En vue de l'obtention du diplôme

Licence Fondamentale en informatique de gestion

Mise en place d'un système partagé et sécurisé dédié à l'enseignement à distance

---

Réalisé au sein de l'Agence digitale N3RD

Réalisé par :
Khalfi Khawla,Kouki Hamza

Sous la direction de :
Pr. Mohamed Amnai (ENSA)
Pr. Mohammed Nasri (ENSA)
M[lle]. Halima Hadi (SQLI)
M. Faouzi Amri (SQLI)

Soutenu le 01 juillet 2020, Devant le jury :

Pr. Meriem Mandar :	ENSA Khouribga	- Présidente
Pr. Mohamed Amnai :	ENSA Khouribga	- Encadrant
Pr. Mohammed Nasri :	ENSA Khouribga	- Encadrant

Année universitaire
2019/2020

Dédicace

“

Je dédie ce travail :

À ma chère mère et à mon cher père qui n'ont jamais cessé de me supporter, me soutenir et m'encourager durant mes années d'études et de moments de faiblesse et de maladie.

Qu'ils trouvent ici le témoignage de ma profonde gratitude et reconnaissance.

À mon frère et ma sœur qui me donnent de l'amour et de la vivacité.

Àmon binôme . À tous ceux qui m'ont aidé - de près ou de loin - ceux qui ont partagé avec moi les moments d'émotion lors de la réalisation de ce travail et qui m'ont chaleureusement supporté et encouragé tout au long de mon parcours.

Et Pour finir, à tous mes amis qui m'ont toujours encouragé.

Merci !

”

- Khawla

Dédicace

“

Je dédie ce travail :

À ma chère mère, et à mon cher père ,Qui n'ont jamais cessé, de formuler des prières à mon égard, de me soutenir et de m'épauler pour que je puisse atteindre mes objectifs.

À A mes frèresr Pour ses soutiens moral et leurs conseils précieux tout au long de mes études.

Àma chère binôme ,Khalfi khawla, Pour leurs indéfectibles soutiens et leurs patiences infinies.

Et tous ceux que j'aime et ceux qui m'aiment.

Merci !

”

- Hamza

Remerciements

Nous adressons nos remerciements les plus sincères à tous ceux qui de près ou de loin ont participé à la réalisation de ce travail.

Nous tenons à exprimer notre profonde gratitude, tout particulièrement, à notre encadrant académique Monsieur **Mounir Nefzi** nous a fait profiter de ses larges connaissances et ses précieux conseils tout au long de ce travail. Il a toujours été à notre écoute et a su nous apporter un soutien sans faille.

Nous souhaitons ensuite adresser nos remerciements au corps professoral et administratif de la Faculté de Sciences Economiques et de Gestion de Tunis, pour la qualité de l'enseignement offert et le soutien de l'équipe administrative.

Nos Vifs remerciements sont adressés également, à notre maître de stage Monsieur **Ouesleti Nizar** et le directeur (à verfier) Monsieur **Bchini Tarek** au sein de la Société d'Ingénierie et de Développement Web «N3RD» pour leurs accueils, le temps passé ensemble et le partage de leurs expertises . Nous saisissons également cette occasion pour adresser notre profond respect à toute l'équipe de travail qui ont fourni le cadre nécessaire pour la réalisation de notre projet. .

Pour finir, Nous voudrions adressons nos sincères remerciements aux membres du jury pour avoir bien voulu examiner et juger ce travail.

Table des matières

Table des figures

Liste des sigles et acronymes

DAM	*Digital Asset Management*
ERP	*Enterprise Resource Planning*
EAI	*Enterprise Aplication Integration*
HAC	*Hybris Administration Console*
MEP	*Mise En Production*
OAT	*Operational Acceptence Testing*
OOTB	*Out Of The Box*
PIM	*Product Information Management*
PCM	*Product Content Management*
TFS	*Team Foundation Server*
UAT	*User Acceptence Testing*
WFJ	*Watches and Fine Jewelry*

Introduction générale

L'enseignement est un mode d'éducation permettant de développer les connaissances d'un élève par le biais de la communication verbale et écrite. Il est centré sur le cours magistral.

Les systèmes traditionnels d'enseignement imposent à tous les apprenants une unité de lieu, une unité de temps, une unité d'action, une unité de rythme ce qui implique une rigidité des mécanismes et une difficulté d'adéquation avec la réalité quotidienne. La tendance à l'amélioration du système sur le plan pédagogique par le recours aux moyens audiovisuels classiques (projections de diapositives, de transparents, séquences vidéo) n'a pas résolu le problème. Il existe en effet une solution de rechange à l'enseignement traditionnel. Cette forme d'enseignement relativement jeune, c'est la formation à distance qui permet d'acquérir des connaissances et de développer des habiletés sans avoir à fréquenter un établissement d'enseignement et sans la présence physique d'une personne qui enseigne. Dans ce cadre s'intègre notre projet de fin d'étude qui est effectué au sein de la société N3RD. Notre objectif est de concevoir et mettre en place un système partagé et sécurisé dédié à l'enseignement à distance.

Ce rapport s'articulera donc, autour de quatre chapitres comme suit :

Le premier chapitre permet de placer notre projet dans son contexte générale. Il comportera une description de l'organisme d'accueil, exposera l'étude de l'existant, mettra l'accent sur la solution proposée et abordera la méthode adoptée. Il présentera également quelques notions théoriques importantes pour la réalisation de notre projet.

Dans le deuxième chapitre, nous dégagerons les besoins les besoins fonctionnels et non fonctionnels, comme nous spécifierons un diagramme de cas d'utilisation général du produit avec le langage de modélisation unifié UML.

Le troisième chapitre elaborera l'étude conceptuelle de notre projet dont lequel nous allons presenter quelues diagrammes.

Le quatrième chapitre « La phase de clôture » sert à présenter l'ensemble des différents outils utilisés pour concevoir et développer notre application ainsi que le diagramme

de déploiement et nous terminons par des captures d'écrans des principaux interfaces de l'application pour illustrer la version finale de notre produit.

Finalement, nous clôturerons ce rapport par une conclusion générale dans laquelle nous évaluerons le travail réalisé au sein de la société et nous proposerons des perspectives dans le but d'améliorer notre travail.

Chapitre 1

Contexte général du projet

> *ce chapitre introductif est dans le but de mettre le travail dans son contexte général. Nous commençons tout d'abord par une présentation de l'entreprise d'accueil. Ensuite, nous mettons l'accent à décrire le sujet autour du quel se déroule notre projet de fin d'études. Enfin, nous allons faire la critique du système actuel et de proposer les solutions adéquats avant de mettre à disposition le langage et la méthodologie de conception.*

1.1 Présentation de l'organisme d'accueil

1.1.1 Cadre général de travail

FIGURE 1.1 – Logo de la société N3RD.

N3RD Créée en 2011 est une société d'ingénierie et de développement web qui, grâce à l'utilisation des nouvelles technologies du web, est capable aujourd'hui d'assurer les différentes tâches. [1].

1.1.2 Domaines d'activité de l'entreprise

N3RD est une société très active dans le domaine du développement Web et mobile (sites et applications), elle n'utilise que les dernières technologies :

- Web 2.0 et plus.
- Langages de développement côté serveur : PHP5, MySQL5, CSS3, HTML5
- Frameworks : Symphony2, Zend, Laravel et Code Igniter
- CMSs : Prestashop, Drupal et WordPress.
- Animation dynamique côté client : CSS3, JavaScript, Bootstrap, Ajax et JQuery.

1.1.3 Mission de l'entreprise

- Développement de sites et applications web pour tout business.
- Développement des solutions web personnalisées.

- Conseils.
- Boutiques en ligne et e-commerce.
- Design sur mesure.
- Création multimédia.
- Infogérance (hébergement, gestion d'infrastructure...).
- Création d'outils collaboratifs.
- Optimisation pour les plateformes mobiles.
- Création d'applications mobiles (Android IOS).

1.2 Étude de l'existant

1.2.1 Cadre du projet

Nous ne pouvons pas commencer ce travail sans avoir des informations et des idées claires et précises sur l'existant. L'étude de l'existant est une phase d'analyse qui consiste à faire un diagnostic sur les points forts et faibles du projet et déterminer des objectifs du nouveau système et l'ébauche de solution. Pour ce faire, il faut étudier le système existant lui même ainsi que l'environnement dans lequel il baigne.

Après observation des plusieur site web université actuel nous avons constaté qu'il ne contient que les emplois du temps et les résultats en ligne ou quelques autres informations pour les etudiants(Stages..) .

1.2.2 Problématique

La formation à plusieur université se fait actuellement de façon traditionnelle (des enseignants, des étudiants et des cours sur place), et leur site ne contient ni des supports de cours ni des séries d'exercices et n'offre pas la possibilité de passer les examens en ligne ce qui gêne les étudiants qui travaillent et veulent poursuivre leurs études en parallèle. Alors notre mission est de résoudre ce problème.

1.2.3 Solution proposée

En tenant compte des différents problèmes que nous avons évoqués, nous sommes amenés à proposer une solution qui répond aux objectifs et qui pallie aux lacunes constatées aux niveau de processus de l'existant.

L'idée de notre projet consiste à mettre en place une application web qui facilite l'étude à distance qui a pour principales missions de faciliter l'apprentissage à distance en partageant les cours et les travaux dirigés entre l'étudiant et l'enseignant et en permettant de passer les examens et d'avoir les notes et les résultats en ligne.

1.3 Planification et conduite du projet

1.3.1 Démarche adoptée

Dans cette section, nous dévoilons le processus simplifié que nous préconisons pour la modélisation du système. Nous présenterons également les principes fondamentaux du EXtreme Programming (XP) et Scrum, afin d'éclairer les idées fortes auxquelles se rattache la démarche pratique adoptée dans la suite du rapport.)

.....	Scrum	EXtreme Programming (XP)
Description	(La méthode s'appuie sur le découpage d'un projet en «sprint», ainsi que l'autoorganisation de l'équipe de développement. Chaque sprint commence par une estimation suivie d'une planification opérationnelle.Le sprint se termine par une démonstration de ce qui a été achevé, et contribue à augmenter la valeur d'affaires du produit.)	(Ensemble de «Best Practices » de développement (Idéal pour le travail en groupe). - Cible des projets de moins de dix personnes.
Points forts)	-Amélioration de la communication. -Règles définies clairement. -Augmentation de productivité.	-Simple à mettre en œuvre. -Fait une large place aux aspects techniques :prototypes, règles de développement, tests. . .
Points faibles	- Violation de responsabilité. -L'équipe ne se prete pas au SCRUM.	(- Ne couvre pas les phases en amont et en aval au développement : capture des besoins, support, maintenance, tests d'intégration. . . -Assez flou dans sa mise en œuvre.)

TABLE 1.1 – Etude comparative entre les approches agiles

presentation de la methode Scrum

Scrum : Aujourd'hui « Scrum » est la méthode agile la plus populaire. Ce terme signifie « mêlée » au rugby. La méthode scrum s'appuie sur des « sprints » qui sont des espaces temps assez courts pouvant aller de quelques heures jusqu'à un mois. Généralement et de préférence un sprint s'étend sur deux semaines. À la fin de chaque sprint, l'équipe présente ce qu'elle a ajouté au produit.

Processus de développement

La nature du projet et sa forte dépendance aux acteurs du domaine sont les raisons qui expliquent le fait d'être toujours à l'écoute du client et prêt à répondre à ses nouveaux besoins. C'est pour cela, l'équipe du projet a opté pour un cycle de développement agile et plus précisément SCRUM.

Le principe de la méthodologie SCRUM est de développer un logiciel de manière incrémentale en maintenant une liste totalement transparente des demandes d'évolutions ou de corrections à implémenter (backlog).

Avec des livraisons très fréquentes, toutes les 4 semaines en général, le client reçoit un logiciel à chaque itération. Plus nous avançons dans le projet, plus le logiciel est complet et possède de plus en plus de fonctionnalités.

Pour cela, la méthode s'appuie sur des développements itératifs à un rythme constant d'une durée de 2 à 4 semaines (2 semaines pour notre cas) comme le montre la figure 1.2 :

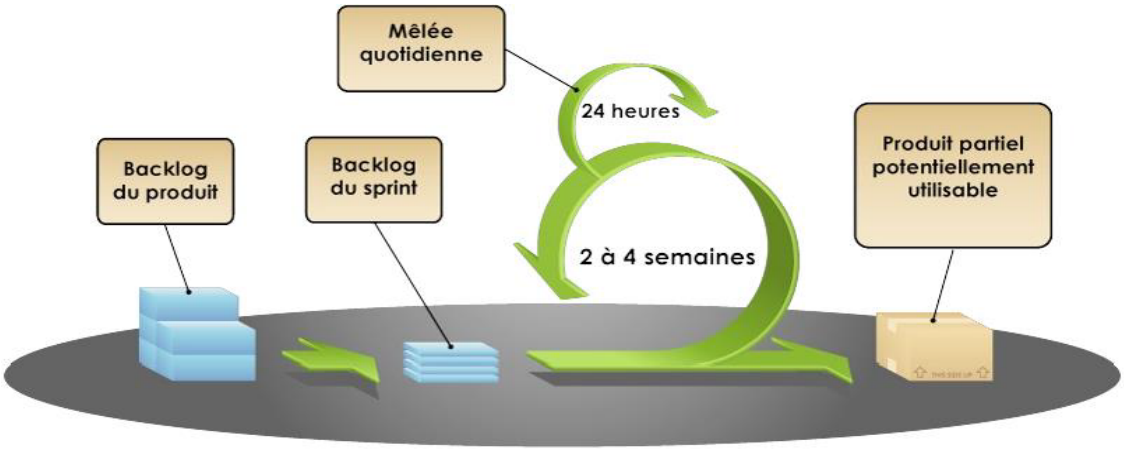

FIGURE 1.2 – Cycle de vie de la méthodologie scrum.

Le schéma illustre un exemple de planification en Scrum : les itérations (sprints) durent en pratique entre 2 et 4 semaines, et possède chacune un but. Le but de chaque sprint

une liste d'items du backlog de produit ou de fonctionnalités à réaliser. Ces items sont décomposés par l'équipe en tâches élémentaires de quelques heures.

Comme nous pouvons le remarquer dans cette figure, pour mettre en place la méthode SCRUM, il faut tout d'abord définir les différentes fonctionnalités de notre application qui forment le backlog du produit. Ensuite, vient l'étape de la planification du sprint pour définir le plan détaillé d'une itération.

Durant un sprint, il y a toujours des réunions quotidiennes entre les différents collaborateurs du projet afin de présenter l'état d'avancement des différentes tâches en cours, les difficultés rencontrées ainsi que les tâches restantes à réaliser. Une fois le produit partiel est prêt, nous vérifions la conformité de ce qui a été fait durant le sprint et nous pouvons alors l'améliorer en procédant à l'étape de rétrospective.

l'equipe Scrum

Scrum est considéré comme un cadre ou un «framework» de gestion de projet. Ce cadre est constitué d'une définition des rôles, il s'articule autour des trois rôles qui sont principalement les suivants :

- **Product Owner** :(Dans la majorité des projets, le responsable produit (product owner) est le responsable de l'équipe projet client. C'est lui qui va définir et prioriser la liste des fonctionnalités du produit et choisir la date et le contenu de chaque sprint sur la base des valeurs (charges) qui lui sont communiquées par l'équipe.)

- **ScrumMaster** : Véritable facilitateur sur le projet, il veille à ce que chacun puisse travailler au maximum de ses capacités en éliminant les obstacles et en protégeant l'équipe des perturbations extérieures.

- **Équipe de dévelopement** : elle regroupe l'ensemble des rôles habituellement nécessaires à un projet, à savoir le concepteur, le développeur, le testeur, etc. L'équipe s'organise elle-même et elle reste inchangée pendant toute la durée d'un sprint.

  Dans notre cas, les rôles sont répartis comme suit :

Rôle	Personne
Product owner	La société N3RD
Scrum Master	Mr Ousleti nizar
Équipe de développement	Mr kouki hamza , Mlle khalfi khawla

TABLE 1.2 – Equipe Scrum

les événements Scrum

Dans le cadre du scrum, il y a 05 événements pour créer de la régularité et minimiser le besoin de rencontres. Tous les événements sont classés dans le temps (time-boxed), ce qui signifie que ces événements ont une durée maximale.

- ☐ Sprint : c'est le cœur du scrum. Un sprint est lorsqu'un incrément de produit est créé. Un incrément est une partie utilisable et potentiellement fonctionnelle du produit final. Les sprints sont classés dans le temps pour un mois ou moins. Chaque sprint a un objectif de ce qui doit être construit.
- ☐ La planification du sprint (Sprint planning) est l'événement où l'équipe scrum définit ce qui sera livré dans un sprint. Cet événement est limité dans le temps à un maximum de huit heures pour un sprint d'un mois. Dans la planification du sprint, l'équipe scrum définira ce qui peut être livré au prochain incrément et combien de travail est nécessaire pour atteindre cet objectif. L'entrée principale de cette réunion est le Product Backlog où le Product Owner et le DevTeam choisiront les éléments qui seront inclus dans ce sprint pour atteindre l'objectif du sprint.
- ☐ Le Daily scrum est une réunion quotidienne de 15 minutes pour DevTeam où ils mettront à jour le statut de travail et les plans pour les prochaines 24 heures. Cette réunion est utilisée pour inspecter l'état d'avancement du sprint. Lors de cette réunion, chaque membre du DevTeam répondra aux questions suivantes :
  1. Qu'est-ce qui a été accompli depuis la dernière réunion ?
  2. Que fera-t-on avant la prochaine rencontre ?
  3. Voyez-vous un obstacle qui vous empêche, ou le DevTeam, d'atteindre l'objectif de sprint ?
- ☐ La revue de sprint (Sprint review) est utilisée par l'équipe scrum pour présenter et inspecter l'incrémentation à la fin d'un sprint. Il est limité à quatre heures pour un sprint d'un mois. Cette réunion est destinée à obtenir des commentaires et des demandes de changement du client et des parties prenantes. Seuls les éléments considérés comme « terminés (Done)» sont inclus dans la revue de sprint.
- ☐ La rétrospective Sprint (Sprint retrospective) est un événement destiné à traiter l'amélioration. Les améliorations pourraient concerner les personnes, les relations, les processus et les outils. Il faut trois heures pour un sprint d'un mois.

1.3.2 Planning du projet

La planification du projet est une phase importante d'avant-projet. Elle consiste à prévoir le déroulement de ce dernier tout au long des phases constituant le cycle de développement.

- **Le diagramme de Gantt :**

  Le diagramme de Gantt, couramment utilisé en gestion de projet, est l'un des outils les plus efficaces pour représenter visuellement l'état d'avancement des différentes activités (tâches) qui constituent un projet. Ce diagramme permet donc de visualiser d'un seul coup d'œil :

  ⋆ Les différentes tâches à envisager.

  ⋆ La date de début et la date de fin de chaque tâche.

  ⋆ La durée escomptée de chaque tâche.

  ⋆ Le chevauchement éventuel des tâches, et la durée de ce chevauchement.

  ⋆ La date de début et la date de fin du projet dans son ensemble.

  Le diagramme de Gantt dans la figure 1.3 illustre le déroulement du stage dans le temps :

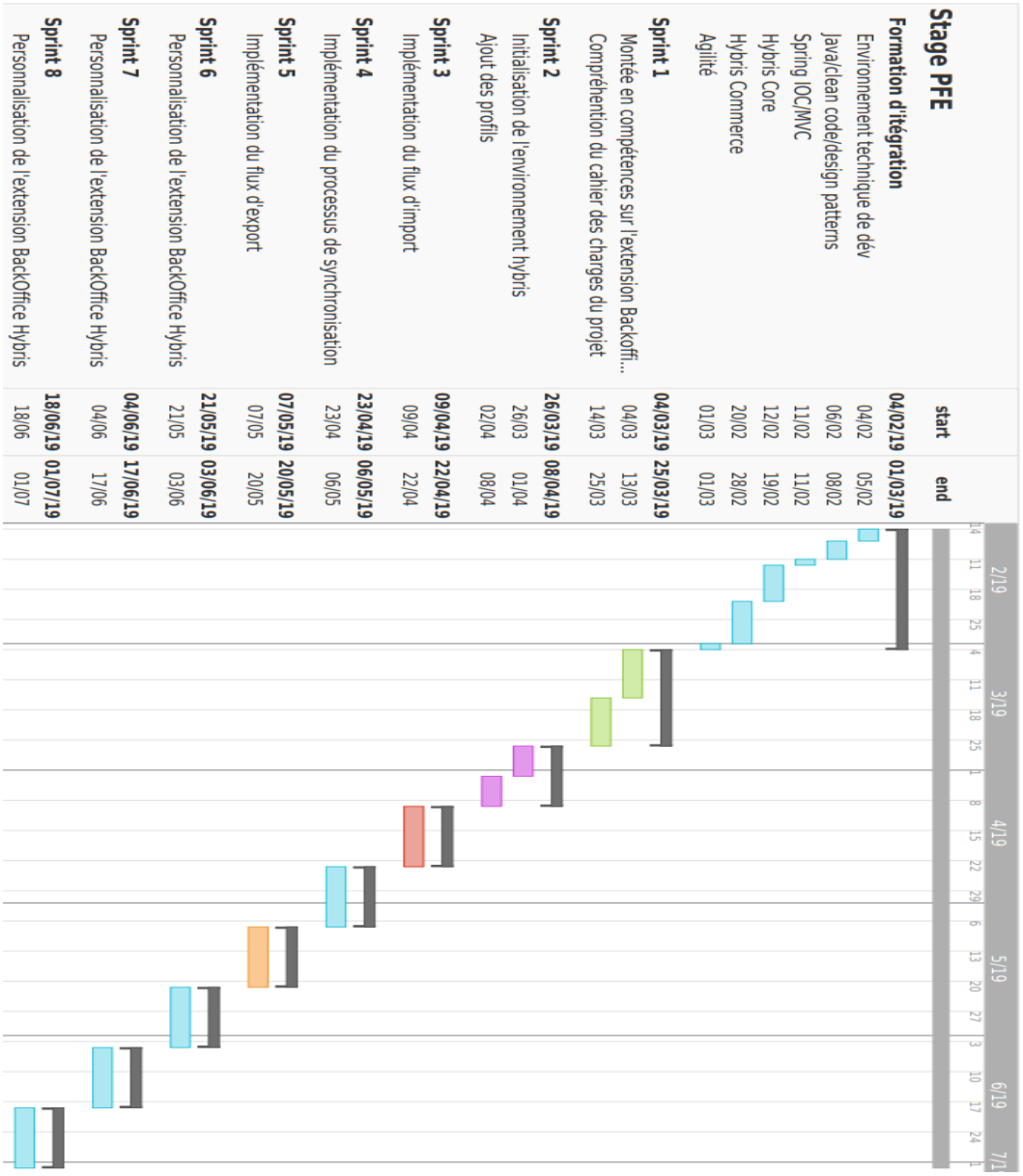

FIGURE 1.3 – Diagramme de gantt du projet.

1.3.3 La modélisation objet

Depuis quelques années, la modélisation objet avec le langage UML est devenue une pratique courante sur de nombreux projets informatiques. En effet, Le recours à la modélisation est une pratique indispensable au développement logiciel, car un modèle est prévu pour arriver à anticiper les résultats du codage.

Définition de UML

UML se définit comme un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, spécifier et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue. UML unifie à la fois les notations et les concepts orientés objet. Il ne s'agit pas d'une simple notation graphique, car les concepts transmis par un diagramme ont une sémantique précise. UML s'articule autour de treize types de diagrammes, chacun d'eux étant dédié à la représentation des concepts particuliers d'un système logiciel [3].

Processus de modélisation

Le processus que nous allons présenter et appliquer tout au long de ce rapport :

- Conduit par les cas d'utilisation ;
- Relativement léger et restreint, comme les méthodes agiles, mais sans négliger les activités de modélisation en analyse et conception ;
- Fondé sur l'utilisation d'un sous-ensemble nécessaire et suffisant du langage UML, conformément à méthodes agiles.

1.4 Conclusion

Tout au long de ce chapitre, nous avons décrit l'organisme d'accueil , nous avons aussi formulé une petite présentation de notre projet pour vous mettre en contexte. Nous avons ensuite fait une étude de l'existant dans l'entreprise pour pouvoir dégager les différentes lacunesainsi que les solutions envisagées. A la fin de ce chapitre, nous avons dévoilé le langage et laméthodologie de conception de notre système. Le chapitre suivant sera consacré à la planification du projet ainsi que la spécification des besoins

Chapitre 2

Analyse et spécification des besoins

> *Dans ce chapitre, j'aborde les phases d'analyse et de spécification des besoins du projet, dans le but d'avoir une vision globale claire du comportement du projet ainsi que les attentes des utilisateurs.*

2.1 Spécifications des besoins

La spécification des besoins va nous permettre d'avoir une meilleure approche des utilisateurs, des fonctionnalités et de la relation entre les deux. Elle sera sous forme de besoins. Pour cela nous allons procéder comme ceci :

- Identification des acteurs du nouveau système.
- Identification des besoins fonctionnels.
- Identification des besoins non fonctionnels .

2.1.1 Identification des acteurs

Un acteur est une personne, un matériel ou un logiciel qui interagit avec le système. L'analyse du présent projet commence par une identification des acteurs agissants sur les différentes parties du système. Les acteurs présentés dans la figure 2.1 sont des employés du clients en plus du serveur qui est la solution adoptée par le client pour la communication et le partage des informations entre ces systèmes et départements .

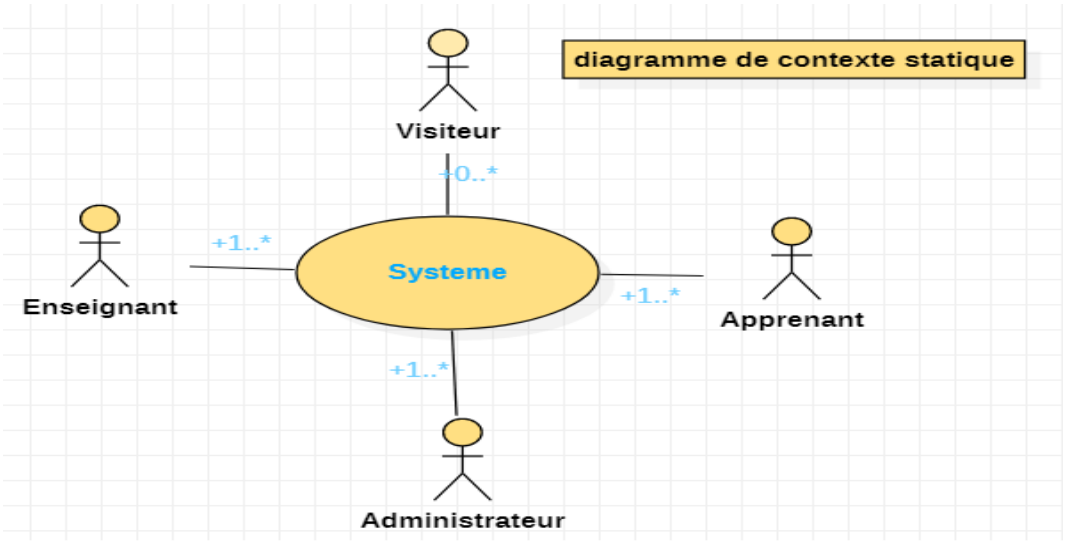

FIGURE 2.1 – Héirarchie des profiles humaines.

Le tableau 2.1 récapitule les acteurs en interaction avec le système en spécifiant le rôle de chacun avant de définir plus précisément leurs interactions avec le système en utilisant

des diagrammes de cas d'utilisation.

Acteur	Fonction
Administrateur	L'administrateur est la personne responsable de gérer la totalité du système.
Enseignant	C'est un acteur principale qui interagit avec notre application. C'est l'acteur qui a pour rôle de gérer les cours, les travaux dirigés (TD) et les examens des étudiants.
Apprenant	L'apprenant inscrit, il va pouvoir consulter les cours et faire les tests qui lui sont proposés. L'apprenant peut avoir la possibilité de participer aux forums, d'envoyer un message à un tuteur, à un autre apprenant ou même à l'administrateur. Ainsi la possibilité de discussion en ligne avec le tuteur, la modification de son profil et la consultation de sesrésultats.
Visiteur	N'importe quel visiteur qui veut télécharger des cours via un compte personnelle a condition de faire les inscriptions pour avoir un compte.

TABLE 2.1 – Acteurs en interaction avec le système

2.1.2 Les besoins fonctionnels

Les besoins fonctionnels expriment une action que doit effectuer le système en réponse à une demande .
Les besoins principaux à couvrir par le système sont les suivants :
Si l'acteur est un Administrateur, il peut :

- S'authentifier :l'Administarateur entre son « username » et son « password » avant d'accéder à l'application pour assurer la confidentialité des informations.
- Gérer les comptes des utilisateurs : l'Administrateur peut ajouter des comptes pour les nouveaux Utilisateurs(Enseignant ou Etudiant) , modifier leurs informations et supprimer les comptes des anciens utilisateurs.

Si l'acteur est un Enseignant, il peut :

- Gérer les Matières :l'Enseignant peut ajouter, modifier et supprimer les matières.
- Gérer les Cours :l'Enseignant peut ajouter, modifier et supprimer les cours.
- Gérer les Traveaux dirigés(TD) :l'Enseignant peut ajouter,modifier et supprimer les Traveaux dirigés(TD).
- Gérer les Examens :l'Enseignant peut ajouter des examens en choisissant une durée de temps determiné.

Si l'acteur est un Etudiant, il peut :

- Consulter les Cours :l'Etudiant peut consulter et télecharger les cours.
- Consulter les Traveaux dirigés(TD) :l'Etudiant peut consulter et télecharger les Traveaux dirigés(TD).
- Passer les Examens :l'Etudiant peut passer les examens en respectant une durée de temps determinée.

2.1.3 Les besoins non fonctionnels

Les besoins non fonctionnels impressionne directement sur déroulement réelle de l'application. Ce sont des besoins techniques décrivant la majorité des contraintes (qu'on a déjà approuvée dans le chapitre précèdent) auxquelles est soumis le système pour sa réalisation et son bon fonctionnement. Pour cela l'ensemble des extensions à réaliser doivent respecter les besoins suivants :

- La Sécurité : La solution proposée permet à l'utilisateur une navigation sécurisée. Elle n'est accessible qu'avec une authentification.
- Ergonomie de l'interface : L'ergonomie est un élément important de l'application : les écrans de saisie doivent être clairs, organisés avec cohérence, de façon à ce qu'une prise en main soit la plus rapide possible.
- Maintenance : L'une des plus importantes besoins de notre application est la facilité de modification pour s'adopter aux nouveaux besoins.
- Portabilité : L'application doit être accessible via n'importe quel navigateur.

2.2 Backlog produit

Le Backlog produit est une liste ordonnée de tout ce qui pourrait être nécessaire dans un produit et constitue l'unique source d'exigences pour toutes les modifications apportées au produit. Le Product Owner est responsable du Backlog produit, y compris son contenu, sa disponibilité et son ordonnancement.
Ses toutes premières moutures ne font qu'esquisser les besoins tels qu'initialement connus et compris. Le Backlog Produit évolue au fur et à mesure que le produit et le contexte dans lequel il sera utilisé évoluent. Le Backlog Produit est dynamique ; il change constamment pour identifier ce que le produit requiert pour être approprié, compétitif et utile. Tant et aussi longtemps qu'un produit existe, son Backlog Produit correspondant existe.
Les caractéristiques fonctionnelles sont appelées des histoires utilisateurs (user story). Les user stories sont caractérisés par :

☐ ***Identifiant*** Il détermine un identifiant unique pour l'histoire en question.

☐ ***Description***Elle décrit le besoin d'un acteur.

☐ ***Critères d'acceptation***À chaque user story sont associés des critères permettant au client de tester l'histoire. Ces critères d'acceptation peuvent être formalisés, pour aller un peu plus loin dans l'aide fournie à l'équipe que l'énoncé de ces critères.

☐ ***Estimation***Est une estimation de la complexité, elle est une valeur entière qui appartient à la suite de Fibonacci.

☐ ***Priorité*** Les priorités sont utilisées pour définir l'ordre de réalisation, elles permettent de constituer le flux de stories qui va alimenter l'équipe. Pour prioriser nos user stories, nous avons pris en compte les critères suivant :

1. *La valeur apportée (Business Value)*

2. *La fréquence d'utilisation*
3. *La réduction des risques*
4. *L'incertitude sur des besoins des utilisateurs qu'un user story permettra de diminuer*
5. *La contribution à la qualité. Les travaux visant à garantir la qualité du produit devraient être prioritaires*
6. *Les dépendances entre stories*

Lors de la création de notre Backlog, nous avons essayé de produire des user stories qui respectent les critères réunies dans le mot INVEST, c'est à dire

- ⋆ Independant : Ne dépend de rien (réduire les liens entre items)
- ⋆ Negociable : Je n'ai pas une solution technique figée
- ⋆ Valuable : pour le client (a une valeur Business)
- ⋆ Estimable : Estimation en complexité
- ⋆ Small / Sized Appropriately : De petite taille (A définir en interne de l'entreprise)
- ⋆ Testable : Pour la validation de l'item

2.2.1 Planification des Sprints

-Duree des Sprints

2.2.2 Backlog des Sprints

Le Sprint Backlog comporte la liste des tâches du Sprint (son périmètre donc) ainsi que la charge de travail associée à ces dernières. Chaque jour, le Reste A Faire de chaque tâche est actualisé par l'équipe de développement afin de tracer le graphique d'avancement de Sprint.

2.3 Diagrammes des cas d'utilisation global

Le modèle des cas d'utilisation décrit les fonctionnalités d'un système d'un point de vue utilisateur, sous la forme d'actions et de réactions ; l'ensemble des fonctionnalités est déterminé en examinant les besoins fonctionnels de tous les utilisateurs potentiels.
Ainsi, pour construire notre modèle, nous allons organiser les cas d'utilisation et les regrouper en ensembles fonctionnels cohérents. Pour ce faire, nous utilisons le concept général d'UML, le package.

Le profil Admin :

Ce diagramme illustre le cas d'utilisation générale de notre système. Ces cas d'utilisation seront par la suite expliqués en détaille. (voir la figure 2.2) :

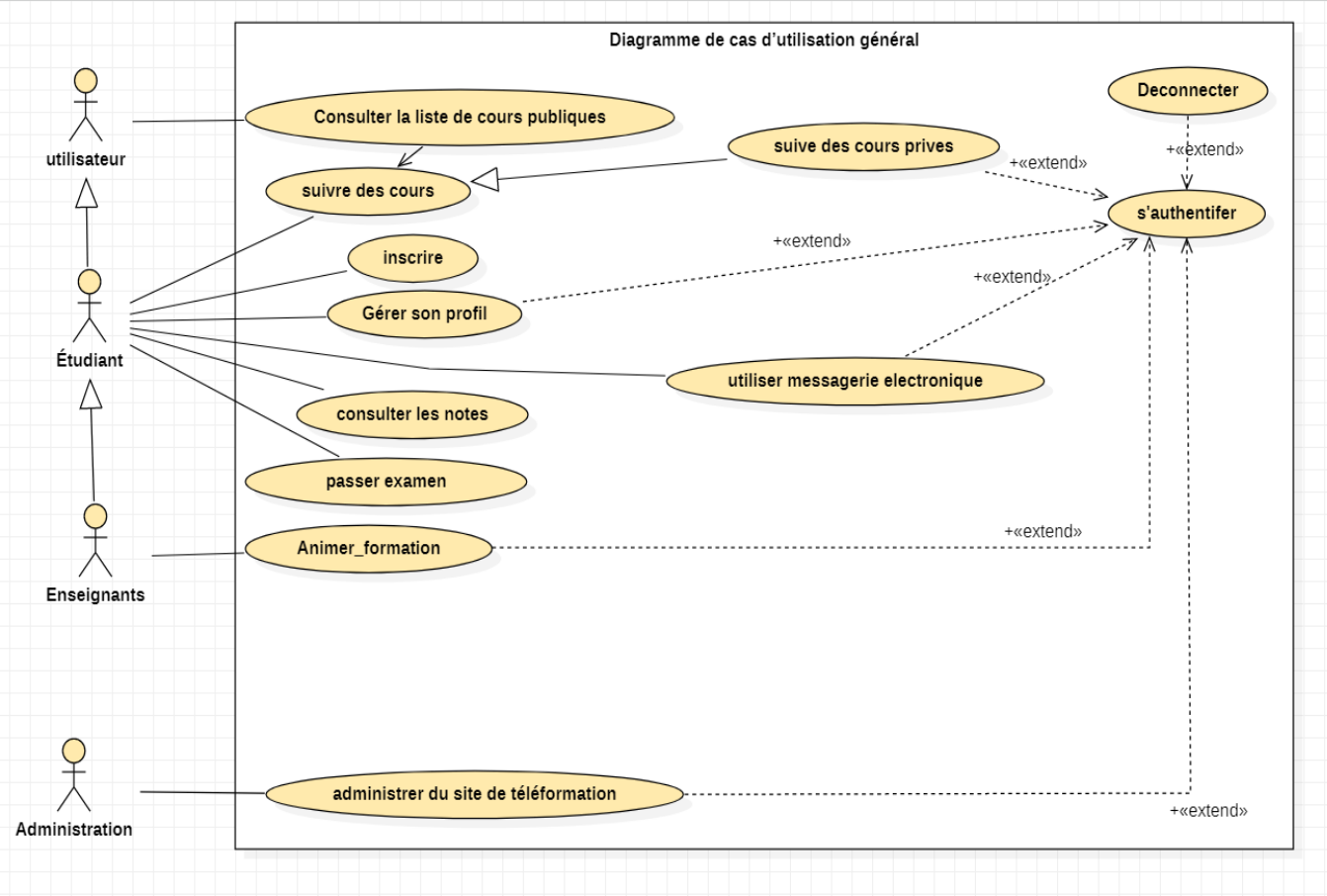

FIGURE 2.2 – Les cas d'utilisation global.

- **Cas d'utilisation S'authentifier :** permet aux utilisateurs de se connecter au système avec leurs logins et mots de passe afin de sécuriser la platforme.
  - ***Objectif :*** Cette fonctionnalité permet aux différents acteurs de se connecter.
  - ***Acteur :*** Tous les acteurs
  - ***Pré-condition :*** L'utilisateur existe dans la base de données.
  - ***Post-conditions :*** Utilisateur authentifié.
  - ***Scénario nominal :***
    1. L'acteur saisit son login et son mot de passe.

2. Le système vérifie les informations saisies.
3. Le système trouve que les informations saisies sont valides.
4. Le système vérifie le rôle de l'acteur.
5. Le système connecte l'acteur à son espace.

– ***Scénario d'erreur :***

1. L'acteur saisit son login et son mot de passe.
2. Le système vérifie les informations saisies.
3. Le système trouve que les informations saisies sont invalides.
4. Le système demande à l'acteur de vérifier les informations saisies.

- **Cas d'utilisation Gérer les employés :** L'administrateur prend en charge la gestion des utilisateurs de la plat-forme :

  – ***Objectif :*** Cette fonctionnalité permet à l'administrateur de gérer les utilisateurs du PIM qui sont des employés du client.

  – ***Acteur :*** Administrateur

  – ***Pré-condition :*** L'acteur se connecte au système.

  – ***Scénarios nominaux :***

  1. *Ajouter un utilisateur.*
     ⋆ L'acteur saisit les informations de l'utilisateur.
     ⋆ Le système confirme à l'administrateur l'enregistrement de l'utilisateur.
     ⋆ Le système affiche la liste des utilisateurs contenant l'utilisateur ajouté.
  2. *Modifier un utilisateur.*
     ⋆ L'acteur affiche la liste des utilisateurs.
     ⋆ L'acteur sélectionne l'utilisateur à modifier.
     ⋆ Le système affiche les informations de l'utilisateur sélectionné
     ⋆ L'acteur modifie les champs concernés.
     ⋆ L'acteur valide ses modifications.
     ⋆ Le système confirme à l'acteur la mise à jour des informations.
     ⋆ Le système affiche la liste des utilisateurs contenant l'utilisateur modifié.
  3. *Supprimer un utilisateur :*
     ⋆ L'acteur affiche la liste des utilisateurs.
     ⋆ L'acteur sélectionne l'utilisateur à supprimer.
     ⋆ Le système alerte l'acteur sur son action.
     ⋆ L'acteur valide son action.

⋆ Le système confirme à l'acteur la suppression de l'utilisateur.

4. *Chercher des utilisateurs :*
   ⋆ L'acteur affiche la liste des utilisateurs.
   ⋆ L'acteur clique sur le volet de recherche.
   ⋆ L'acteur choisit les critères de recherche.
   ⋆ L'acteur lance la recherche.
   ⋆ Le système affiche les résultats de recherche.

- **Cas d'utilisation Gérer les catégories :** L'administrateur peut gérer les catégories qui sert a classifier les produits selon leurs critères et typologies :

  – ***Objectif :*** Cette fonctionnalité permet à l'administrateur de gérer les catégories des produits disponibles dans le PIM.
  – ***Acteur :*** Administrateur
  – ***Pré-condition :*** L'acteur se connecte au système.
  – ***Scénarios nominaux :***

  1. *Ajouter une catégorie.*
     ⋆ L'acteur saisit les informations de la catégorie.
     ⋆ Le système confirme à l'administrateur l'enregistrement de la catégorie.
     ⋆ Le système affiche la liste des catégories contenant la catégorie ajoutée.
  2. *Modifier une catégorie.*
     ⋆ L'acteur affiche la liste des catégories.
     ⋆ L'acteur sélectionne la catégorie à modifier.
     ⋆ Le système affiche les informations de la catégorie sélectionnée
     ⋆ L'acteur modifie les champs concernés.
     ⋆ L'acteur valide ses modifications.
     ⋆ Le système confirme à l'acteur la mise à jour des informations.
  3. *Supprimer une catégorie :*
     ⋆ L'acteur affiche la liste des catégories.
     ⋆ L'acteur sélectionne la catégorie à supprimer.
     ⋆ Le système alerte l'acteur sur son action.
     ⋆ L'acteur valide son action.
     ⋆ Le système confirme à l'acteur la suppression de la catégorie.
  4. *Affecter/désaffecter des produits à une catégorie :*
     ⋆ L'acteur affiche la liste des catégories.

- ⋆ L'acteur sélectionne la catégorie souhaitée.
- ⋆ Le système affiche les informations de la catégorie sélectionnée
- ⋆ L'acteur ouvre l'onglet des produits inclut dans la catégorie.
- ⋆ L'acteur ajoute/supprime des produits à/de la catégorie.
- ⋆ L'acteur valide ses modifications.
- ⋆ Le système confirme à l'acteur la mise à jour des informations.

5. *Chercher une catégorie :*
   - ⋆ L'acteur affiche la liste des catégories.
   - ⋆ L'acteur sélectionne ouvre le voler de recherche.
   - ⋆ L'utilisateur choisit les critères de recherche.
   - ⋆ L'acteur lance la recherche.
   - ⋆ Le système affiche la liste des catégories conformes à la recherche.

2.4 Diagrammes de classes global

Le diagramme de classes est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que les différentes relations entre celles ci. Ce diagramme fait partie de la partie statique d'UML car il fait abstraction des aspects temporels et dynamiques. Une classe est un ensemble de fonctions et de données (attributs) qui sont liées ensembles par un champ sémantique [N4]. Dans ce qui suit nous allons décrire le diagramme de classes relatif à notre application. 2.3).

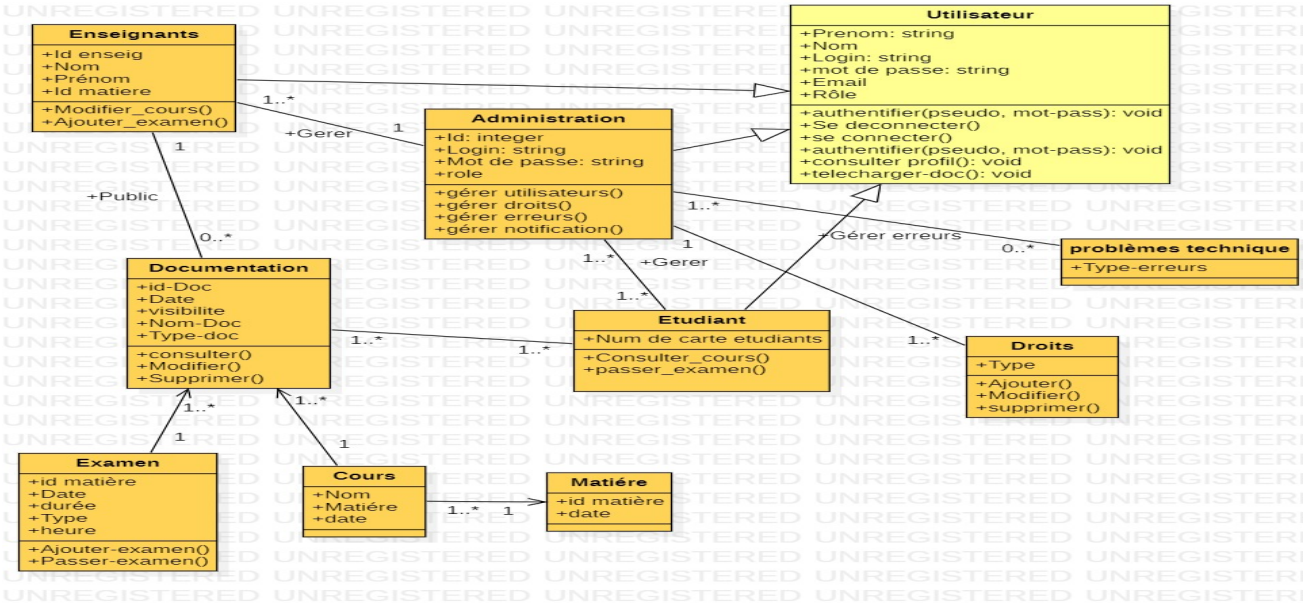

FIGURE 2.3 – Diagrammes de classes global.

2.4.1 Conclusion

Le but de ce chapitre était de définir et d'analyser l'ensemble des besoins fonctionnels de notre solution. Cette étape est primordiale dans le développement d'un projet informatique puisque elle nous permet de définir le périmètre fonctionnel du projet, et de garantir la couverture de l'ensemble des fonctionnalités recensées.

Chapitre 3

Étude Conceptuel et architectural

> *Dans ce chapitre nous allons nous concentrer sur l'aspect conceptuel et architectural de la solution à mettre en place. Cette étude conceptuelle est basée sur l'élaboration et l'analyse des différents schémas et diagrammes de classes.*

3.1 Introduction

L'organisation de l'entreprise cliente repose sur le maillage de trois divisions : la Mode (Fashion), les Parfums Beauté (fragrance and beauty) et l'Horlogerie Joaillerie (Watches And Fine Jewelry). La plate-forme Hybris est potentiellement utilisée pour les 3 divisions.

Pour ce qui concerne la solution PIM, elle est principalement implémentée pour les deux divisions de Mode et d'Horlogerie Joaillerie. Pour cela, une partie de la modélisation de cette solution est partagée entre les deux solutions, et ce pour remédier à la répétition et mettre en valeur les éléments communs entre les deux divisions.

Le présent projet porte sur la solution PIM de la division Horlogerie Joaillerie. Pour des raisons de lisibilité, on préfixera dans le reste de ce chapitre les entités et notions propre à cette division avec le préfixe WFJ (Watches and Fine Jewelry) et pour les entités et notions communes entre les deux divisions avec le préfixe PIM.

3.2 Modèle de données

3.2.1 Catalogue et versions

Hybris est une solution logicielle centrée sur le principe de catalogue de produits afin de gérer les données des produits, connu aussi sous le nom de PCM (Product Content Management). Ce catalogue est composé de plusieurs versions et regroupe l'ensemble des produits avec leurs données relatives (voir la figure 3.1).

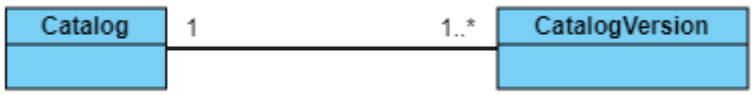

FIGURE 3.1 – Aperçu du diagramme de classes du catalogue et ses versions.

Notre solution PIM s'appuie sur ce principe. Par la suite, on utilise les deux versions de catalogue **Staged** et **Online** pour distinguer les données en cours d'enrichissement des données publiées (utilisables pour être exportées).

La réplication des données d'une version du catalogue à l'autre est effectué à partir d'un traitement de **Synchronisation** natif à hybris, qui est à son tour possible à travers le principe du **Catalog Aware** d'Hybris.

Ce processus est un job d'Hybris qui sélectionne les produits et les autres éléments du système qui sont **Catalog Aware** de la version intermédiaire ou **Staged**, vérifie la

présence de certaines règles (indiquant que l'élément est prêt à être mis en ligne ou publié) et crée (ou la met à jour s'il est déjà créé) une copie de l'élément avec la version En ligne ou **Online**.

Le principe du **Catalog Aware** sert à indiquer les produits ou les entités devraient avoir deux copies dans le système, afin que les utilisateurs professionnels puissent apporter les modifications appropriées avants leurs publication. Ces éléments sont appelés **Catalog Aware** et entrent dans le processus de synchronisation. Or les autres éléments du systeme qui ne participent pas ou ne devraient pas participer au processus de synchronisation sont appelés **Catalog Unaware**.

La figure 3.2 illustre le principe du catalogue et des versions de catalogue d'Hybris, implémenté dans la plate-forme PIM sous le nom de **WFJProductCatalog** :

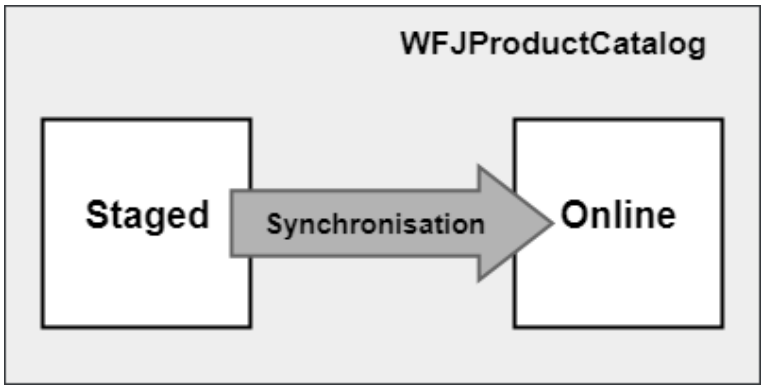

FIGURE 3.2 – Le principe du catalogue des produits d'Hybris.

On suit donc les règles de gestion suivantes pour adapter ce système aux besoins de notre solution PIM :

- L'import des données se fait sur la version Staged.
- L'enrichissement des données se fait sur la version Staged.
- Lorsqu'un produit est correctement enrichi, il est manuellement passé à l'état "approved".
- L'export des données se fait à partir de la version Online.
- Le système de synchronisation recopie les données de la version Staged vers la version Online pour tous les produits qui sont à l'état "approved".

- Si un produit change d'état d'approbation sur la version Staged
  - Il n'est plus synchronisé sur la version Online.
  - Mais il reste présent sur la version Online (avec les données de la dernière synchronisation).

On met en place un traitement (cronjob) de synchronisation, basé sur le fonctionnement natif hybris, permettant de répliquer les données du catalogue (produits avec leurs attributs, catégories, etc.) de la version **Staged** vers la version **Online**. Les entités qui sont synchronisées sont :

- Produits (et toutes les déclinaisons)
- Catégories (et toutes les déclinaisons)
- Médias (et toutes les données associées)

Les processus d'import, d'enrichissement et d'export se résument dans la figure 3.3 :

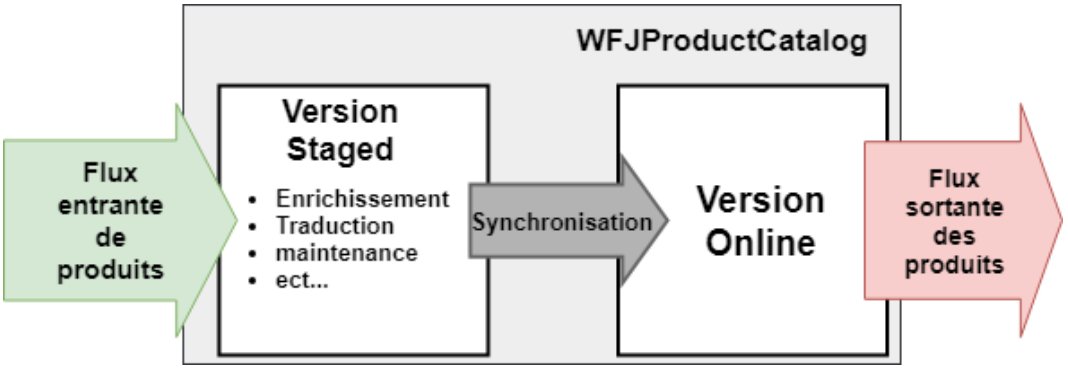

FIGURE 3.3 – Récapitulatif du processus de Synchronisation.

3.2.2 Les produits

Tous les produits de la division Horlogerie Joaillerie ou WFJ (Watches And Fine Jewelry) sont gérés à partir de l'ERP central, nommé CERP et basé sur la solution technique Microsoft AX. Indépendamment de leurs typologies, les produits WFJ sont déclinés suivant 2 à 3 niveaux de variante : **Référence** (entité produit de base qui contient la plupart des attributs de base du produit) déclinées en **SKU** (variation du produit de base selon un certain attribut comme la taille ou la couleur) et éventuellement **Matricule** (pour les

pièces numérotées).

Le schéma de la figure 3.4 illustre cette hiérarchie des produits :

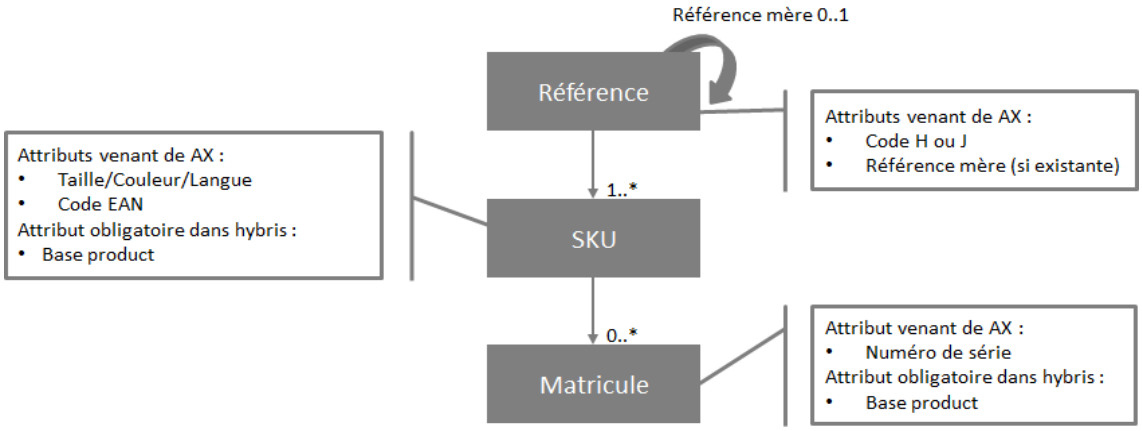

FIGURE 3.4 – Récapitulatif de l'abstraction du produit.

Les Références ou les produits de base :

La **Référence** est équivalente au **Product** dans le modèle de données natif d'Hybris. On définit donc la référence dans notre solution avec L'entité **WfjProduct**, en héritant de ce modèle.

Chaque référence est unique et possède un code (unique) défini dans l'ERP est utilisé telquel dans le PIM.

- Dans AX, l'horlogerie et la joaillerie ont chacune une base de données distincte même si certaines données référentielles sont communes aux deux.
- Les références de joaillerie ont un code commençant par un **J**.
- Les références d'horlogerie ont un code commençant par un **H**
- Une référence peut avoir une référence mère ; nommée masterProduct coté PIM.
- Les références sont identifiables par leur **code** (**H** ou **J**), il s'agit de leur clé d'unicité.

Pour définir les variations d'une certaine **Référence** on a introduit la notion de **Grille de Tailles**, modélisée par l'entité **WfjErpSizeGrid**. Cette entité est liée à la référence

(WfjProduct) et porte les différents valeurs possibles de la **Taille** modélisée par l'entité **WfjErpSize** et représentant le paramètre du variation de cette référence.

Cette notion de grille de taille et détournée pour représenter non seulement les variations de vraies tailles (S, M, X, XL, ...) mais aussi des variations de couleurs ou de langues (pour les catalogues imprimés).

Les SKUs ou les variants du produit selon la taille :

Le **SKU** est la variante de produit basée sur l'attribut de taille en fonction de la grille de la taille renseignée sur la **Référence**. Les **SKU**s sont présentés dans le PIM par l'entité **WfjSkuVariant**.

- Un SKU est lié à une et une seule Référence via l'attribut *"baseProduct"* et une référence peut avoir plusieurs SKU.
- la notion de taille est légèrement détournée dans AX pour certains types de produits. En effet, certaines grilles "de taille" sont déclarées pour gérer des déclinaison de couleur ou de vraies tailles.
- Les SKU sont identifiables par leur **code** de référence de rattachement et leur **taille** (ou leur position dans leur grille de taille), les deux forment la clé d'unicité.

Les Matricules ou les pièces numérotées :

Les **Matricules** sont les produits réels dans le cas de pièce numérotées. Chaque matricule représente une version unique d'un SKU et possède donc un numéro de série unique (généré par AX). Les Matricules sont présentés dans le PIM par l'entité **WfjSerialNumberVariant**.

- Un matricule est lié à un et un seul SKU via l'attribut *"baseProduct"*. Un SKU peut avoir plusieurs matricules.
- Les matricules sont identifiables par (autrement dit, la clé d'unicité est le triplet) :
  - Leur **code** de Référence de rattachement.
  - Leur **taille** (ou leur position dans leur grille de taille).
  - Leur **numéro de série**.

Le diagramme de classes de la figure 3.5 présente les différentes classes utilisés pour représenter le modèle du produit. Il met en évidence avec trois couleurs les classes qui sont natives à Hybris (en bleu) et les classes qui sont communes avec les autres divisions où est implémenté le PIM (en orange et préfixées par PIM) et celles conçues spécifiquement pour les produits de la division WFJ (en rose et préfixées par WFJ).

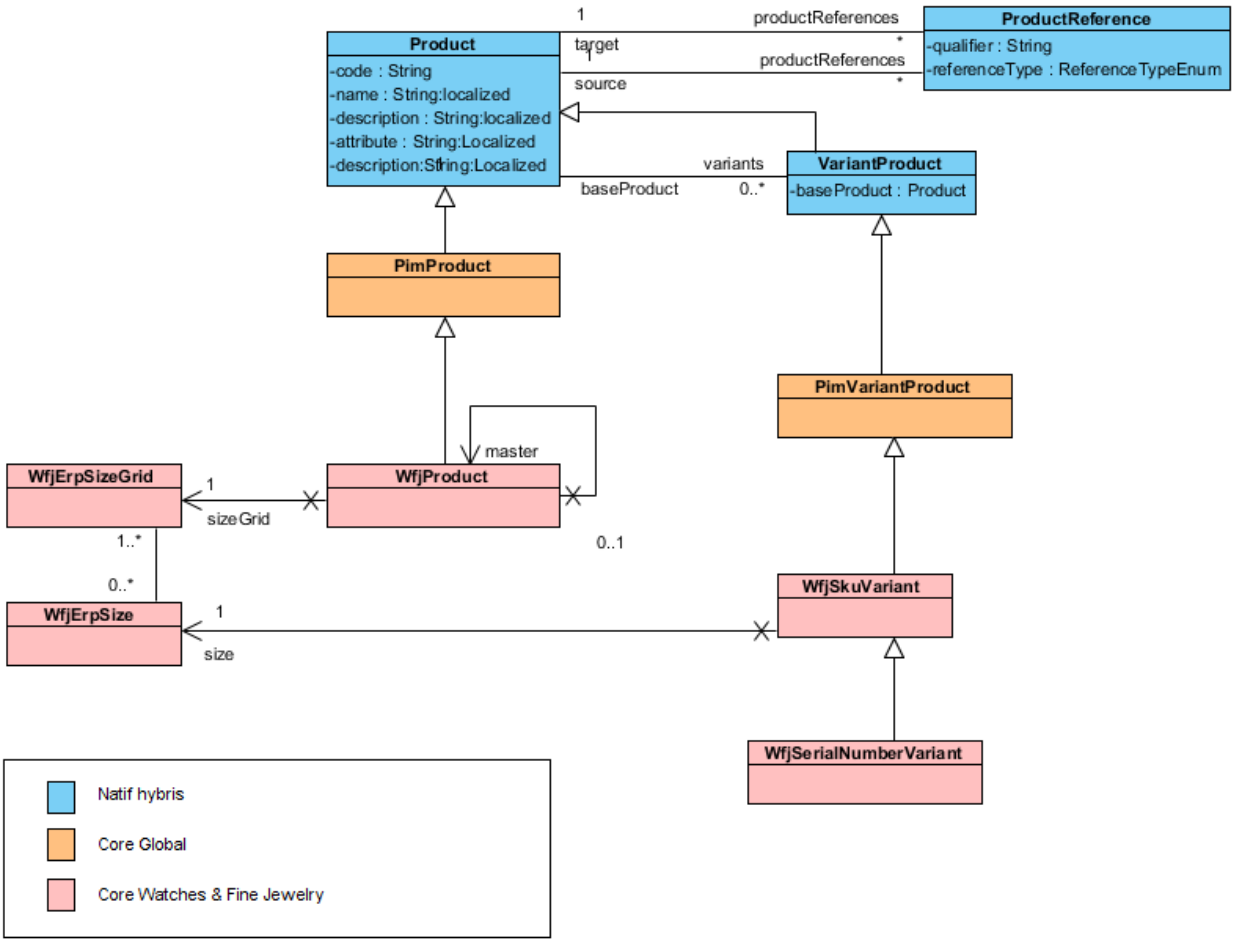

FIGURE 3.5 – Diagramme de classes des produits.

Comme le montre la figure 3.5,les différentes entités présentées dans ce diagramme sont comme suit :

- **WfjProduct**, hérite de **PimProduct** qui hérite à son tour, de L'entité **Product** d'Hybris est constitue la **Référence** présenté précédement.

- **WfjProduct** peut avoir un master product du même type permettant de mettre

en oeuvre des systèmes d'héritage des données, c'est à dire de déduire certaines informations d'une référence à partir de sa référence master.

- **ProductReference** permet d'établir un lien entre de produit en spécifiant le type de ce lien et les deux produits source et target.

- **WfjProduct** porte la notion de *"SizeGrid"* (grille de déclinaisons de taille).

- **WfjErpSizeGrid** Les grilles de tailles proviennent d'AX. Le concept de grille de taille est étendu (détourné) pour pouvoir gérer des tailles mais aussi des langues ou des couleurs.

- **WfjSkuVariant**, represente le **SKU** introduit précédemment et hérite de PimVariantProduct qui, à son tour, hérite du VariantProduct du modèle natif d'Hybris, représentant des variations du produit en question comme par exemple au niveau de la taille (S, M, L, XL...) ou la couleur.

- **WfjErpSize**, La taille indique l'information liée à la grille des tailles de la référence.
  - Si la référence est liée à la grille des tailles "Taille", alors l'id sera une taille.
  - Si la référence est liée à la grille des tailles "Couleur", alors l'id sera une couleur et la taille du produit sera unique.

- **WfjSerialNumber**, represente le **Matricule** introduit précédemment et hérite de **WfjSkuVariant**, il représente les produits réels dans le cas de pièce numérotées. Chaque matricule représente une version unique d'un SKU et possède donc un numéro de série unique (généré par AX).

On suit ainsi les règles de gestion suivantes pour les réferences et les grilles de taille :

- On affecte toujours une et une seule grille de taille à une référence (WfjProduct) et une référence doit toujours posséder une grille de taille (WfjErpSizeGrid).

- La grille de taille unique *"UNI"* permet de gérer les cas où il n'y a pas de variation de la référence.

- Les grilles de tailles sont importées dans le PIM à partir du flux AX.

3.2.3 Les catégories

La plate-forme PIM permet de regrouper les produits dans des catégories. Ces catégories peuvent être structurées en arborescences et on identifie trois grandes typologies de catégories (voir la figure 3.6) :

- **ERP Category** : catégories permettant de représenter l'arborescence de l'ERP. Ces catégories et leurs liaisons sont répliquées à partir de flux de données en provenance de l'ERP et ne sont pas suceptible d'être modifié dans le PIM.
- **Communication Category** : catégories permettant de constituer l'arborescence principale de communication, autrement dit, de créer une autre arborescence à partir de celle de l'ERP pour le modifier et l'évoluer au besoin.
- **Tags Category** : catégories moins structurées, permettant d'organiser les produits de manière plus libre et avec d'éventuelles redondances.

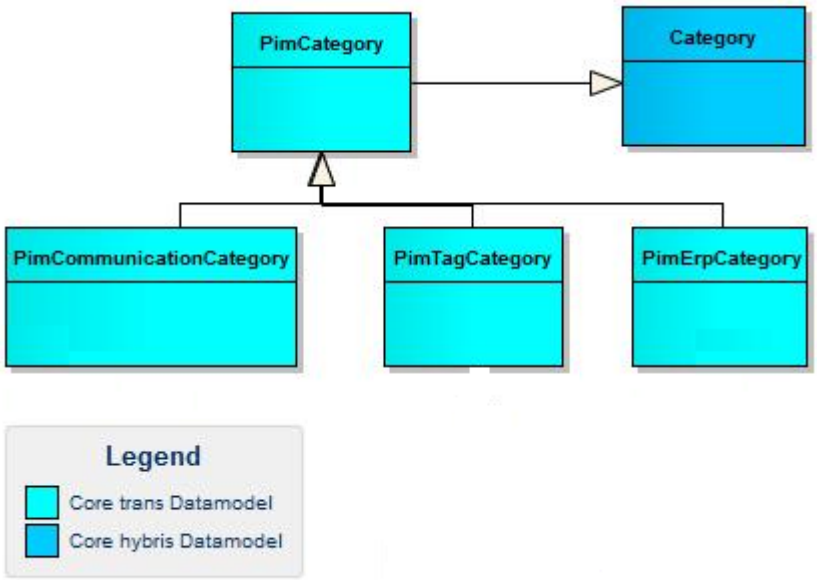

FIGURE 3.6 – Diagramme de classe des catégories.

3.2.4 Les medias

Le DAM (Digital Asset Management) permet de gérer des Assets ; c'est à dire des ressources numériques (principalement des images, scan PDF et vidéos) visant à être présentées sur des canaux digitaux ou autres supports papiers [2].

- Les Assets du **DAM** sont matérialisées à partir de l'entité native d'Hybris : **MediaContainer**.
- Tout ou partie des **métadonnées** d'un Asset sont stockées dans des attributs du **MediaContainer** correspondant.
- Les formats de déclinaison des Assets du DAM sont configurés en tant que **MediaFormat** hybris (entité native de formats des médias).
- Les différentes déclinaisons, dans différents formats, d'un même Asset du DAM sont chacune matérialisées par un **Media** dans hybris. Chaque Media concerne un format donné.
- Tous les Medias sont regroupés au sein du **MediaContainer** représentant l'Asset.

Le diagramme de classes de la figure 3.7 représente ces différents classes et leurs relations :

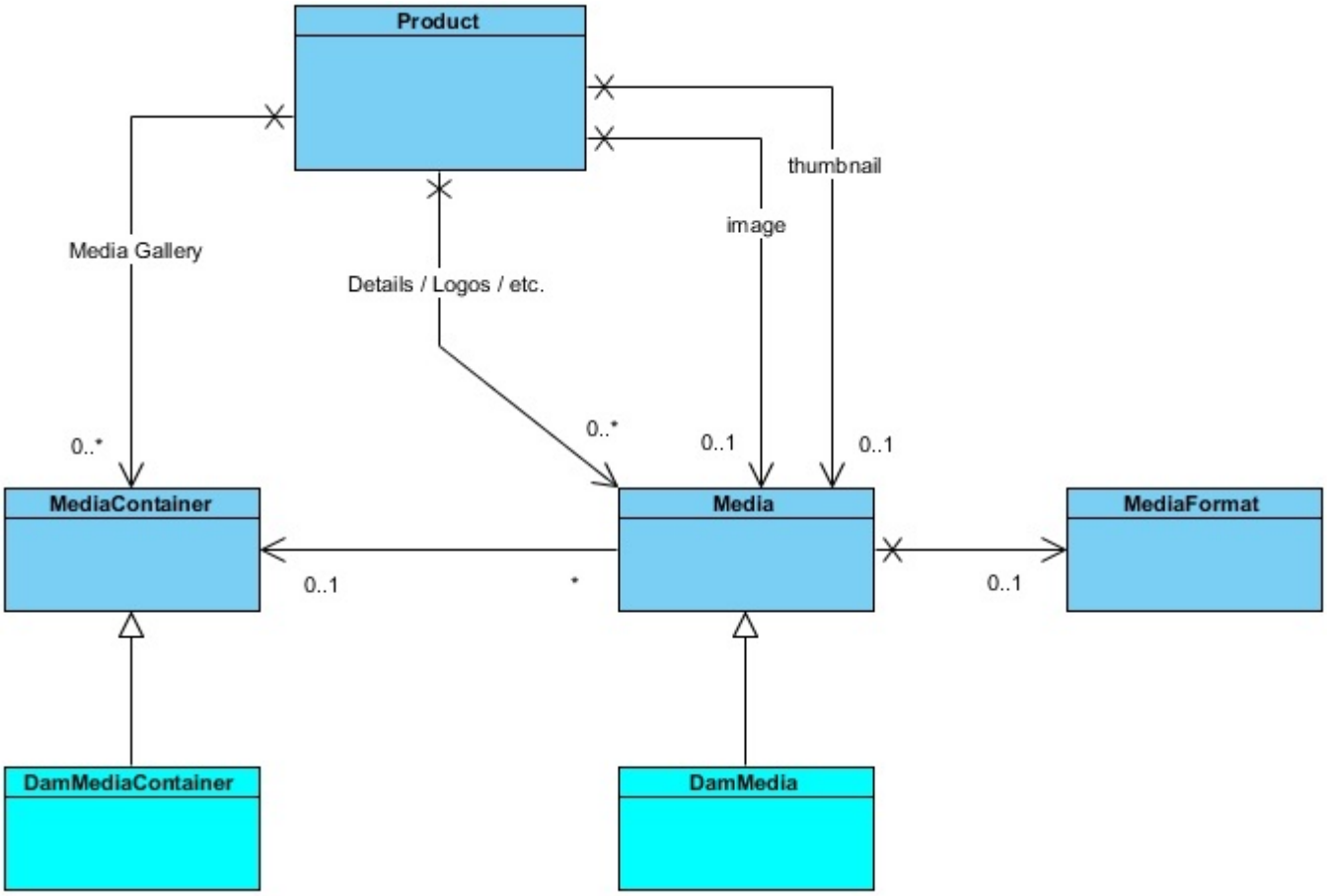

FIGURE 3.7 – Diagramme de classe des médias.

3.3 Le flux d'import

Le socle d'import permet d'importer des données depuis des systèmes tiers qui sont l'ERP centrale et le DAM du client vers le PIM. Il permet, par exemple, d'importer des produits ou des données référentielles ou encore des prix de produits.

Le socle d'import consomme des fichiers XML de données et plus précisément des fichiers au format CBL. En règle générale, les fichiers à importer dans le PIM sont mis à disposition par l'EAI Biztalk (qui prend donc en charge le transport des fichiers importés).

Le socle d'import fonctionne suivant un principe dit de "hotFolder" ; c'est à dire que les fichiers déposés dans les dossiers d'entrée (Splitting ou In) sont automatiquement consommés et traités dès lors qu'ils respectent une règle de nommage prédéfinie.

Le processus d'import comporte 3 principaux traitements comme le montre la figure 3.8 :

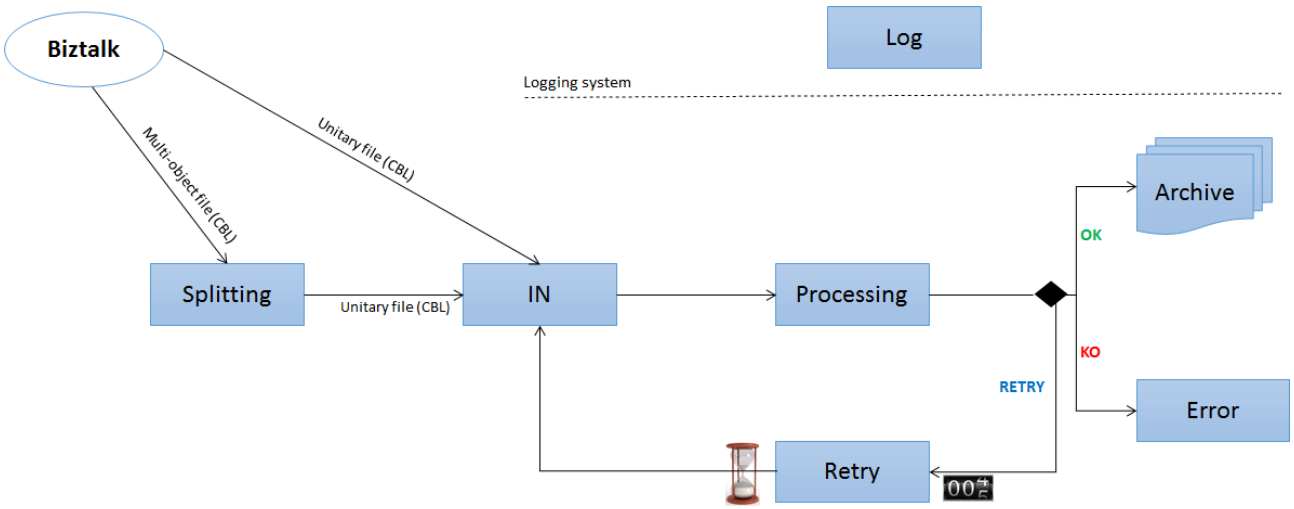

FIGURE 3.8 – Processus d'import des produits.

- **Splitting :** gère le découpage d'un fichier XML au format CBL [1] de collection d'objets en de multiples fichiers unitaires

  – L'opération de découpage n'est pas systématique. Pour la plupart des flux de données, Biztalk livre des fichiers unitaires qu'il dépose directement dans le dossier d'import.

- **Import :** étape d'injection des données d'un fichier unitaire vers un objet dans le PIM.

- **Retry :** traitement de relances multiples de l'import de fichiers unitaires ayant rencontré une erreur lors de l'import.

Une operation de **logging** ou journalisation et mise en place tout au long du processus d'import pour garder la traçabilité des différentes étapes et donc d'identifier les sources des éventuelles erreurs.

Traitement "splitting" :

Le Traitement de **splitting** est illustré dans la figure 3.9 :

1. Common Business Library : un format des fichiers XML utilisé dans l'ecommerce[3]

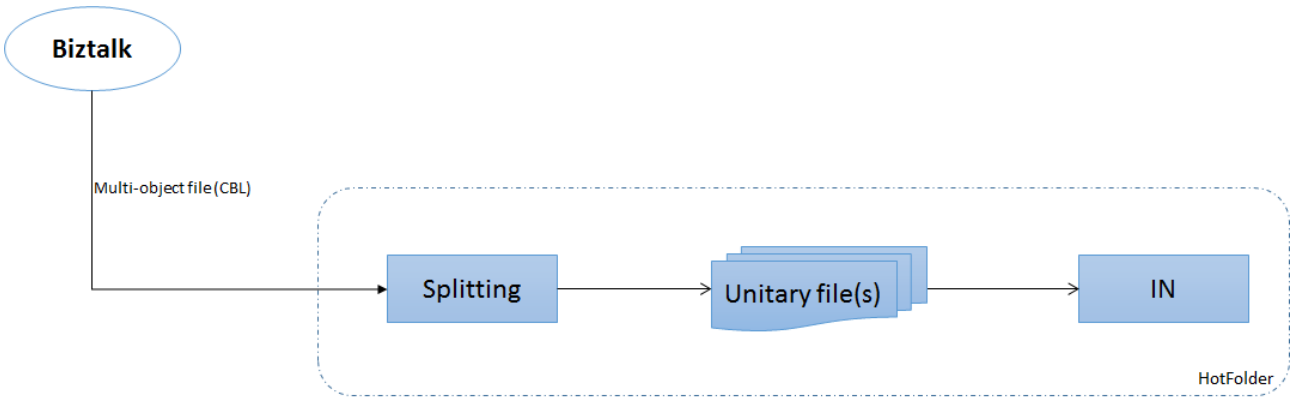

FIGURE 3.9 – Processus de splitting des fichiers xml.

Le traitement permet de découper des fichiers CBL contenant plusieurs objets de données (produits) en de multiples fichiers unitaires (contenant respectivement les données d'un seul produit) voir la figure 3.10.

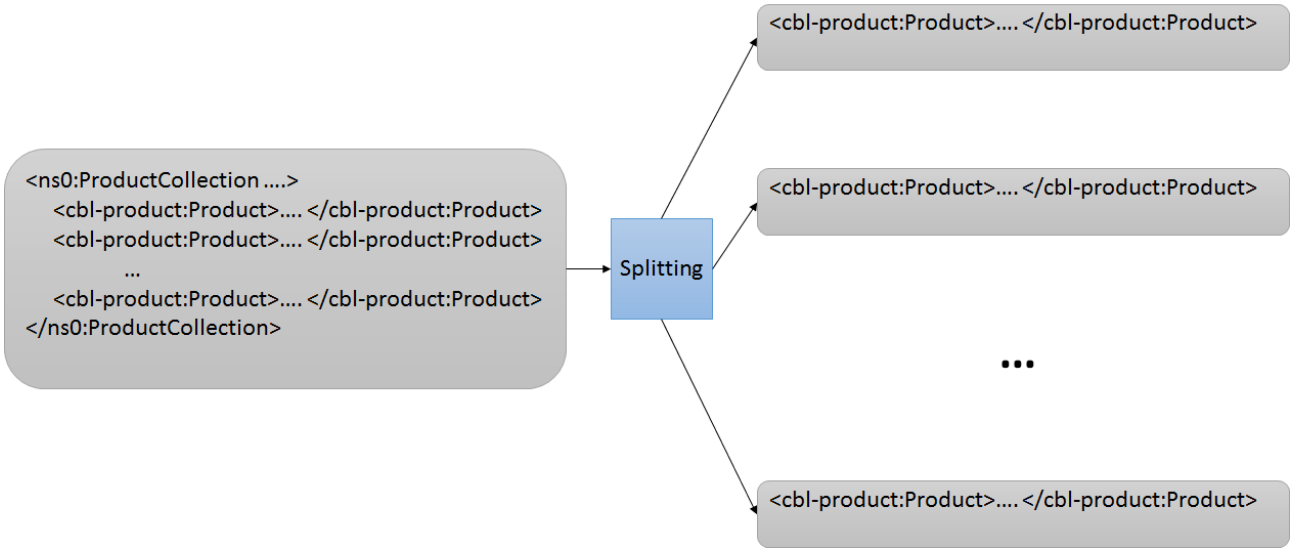

FIGURE 3.10 – Splitting d'un fichier multiple vers des fichier unitaire.

- Le dossier d'entrée du traitement de découpage est le dossier "splitting".
- Le traitement est en mode "**hotFolder**".
- Le traitement lit le contenu du fichier initial (à découper) et génère à la volée (au fûr et à mesure) les fichiers unitaires.

- Chaque fichier généré est automatiquement déposé dans le dossier d'import.

- Les fichiers unitaires suivent une règle de nommage particulière et intégrant l'identifiant de l'objet qu'ils représentent

Traitement "Import" :

Le traitement d'import d'un fichier de données commence par déplacer le fichier à importer dans un dossier de travail : *Processing*. Certains contrôles (dépendant de l'entité de donnée importée) sont effectués lors de l'import.

En fonction du bon déroulement de l'import et des contrôles, le fichier est déplacé dans l'un des dossiers suivants :

- **Archive :** si l'import est effectué avec succès

- **Error :** si l'import à échoué

- **Retry :** si l'on souhaite refaire de nouvelles tentatives d'import du même fichier unitaire.

Traitement "Retry" :

Le traitement de Retry permet de relancer l'import d'un fichier qui aurait rencontrée un erreur lors d'un import.

Le traitement de Retry s'appuie sur 2 paramètres :

- Le délai d'attente avant un Retry

- Le nombre de tentatives de Retry

Le principe du traitement est :

- Après un import *"normal"* avec erreur, le fichier importé est placé dans le dossier **Retry**.

- Après le délai d'attente du retry, le fichier est de nouveau déplacé vers le dossier d'import (**In**) ; et le système comptabilise la nouvelle tentative d'import.

- L'opération est répétées tant que le fichier soulève une erreur (configurée pour le retry).
- Si le nombre de tentative est atteint, après un nouvel import avec erreur, le fichier est finalement déplacé dans le dossier d'erreur (**error**)

3.4 Architecture Logiciel du projet

Notre solution est basée sur la plate-forme SAP Hybris et suit donc son architecture logicielle présentée dans la figure 3.11 :

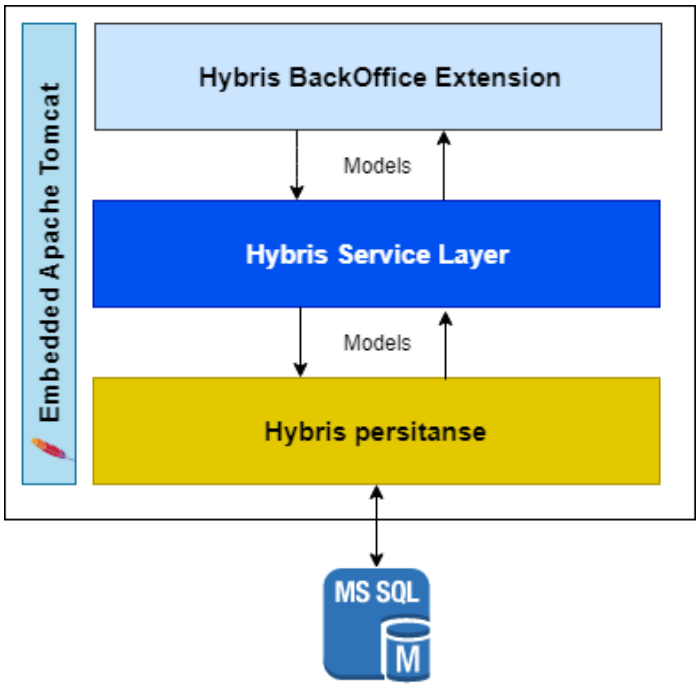

FIGURE 3.11 – Architecture logiciel du projet.

- la plate-forme Hybris inclut un serveur **Apache Tomcat** constituant son serveur Http.
- **l'extension Backoffice d'hybris :** c'est une extension Hybris qui permet aux utilisateurs métier d'avoir accès aux différentes fonctionnalités d'administration du contenu que se soit le catalogue, les catégories ou les produits ainsi que les différentes entités et types du système.
  - l'extension backoffice est basée sur le framework web **ZK**.
  - cette extension permet aux développeurs de créer ou bien de personnaliser les composants Hybris selon leurs besoins.
- **Le service Layer** de Hybris constituant la couche métier où il est implémenté la logique du business, il est constitué à sont tour d'un ensemble de couches de services et communique avec le backoffice et la couche de persistance à travers les **Models** (les entités de la logique métier).

- **La couche persistance** d'hybris est l'intermédiaire entre la base de données **MS SQL Server** est le Service Layer.

3.5 Architecture physique du projet

Cette partie présente l'architecture physique du projet, avec les différents serveurs utilisés. L'architecture de notre projet est composé de deux parties : interne et externe.

La partie interne est l'architecture implémentée au niveau de SQLI Rabat, elle est essentiellement composée d'un seul serveur d'intégration INT (Integation) en plus du serveur SonarQube [2] (un logiciel libre permettant de mesurer la qualité du code source en continu [4]) pour l'analyse du qualitée de code.

La partie externe est celle implémentés dans l'environnement client et composée de trois serveurs : UAT pour le User Acceptence Testing, OAT pour l'Operational Acceptence Testing et PROD le serveur de production.

La figure 3.12 illustre cette répartition :

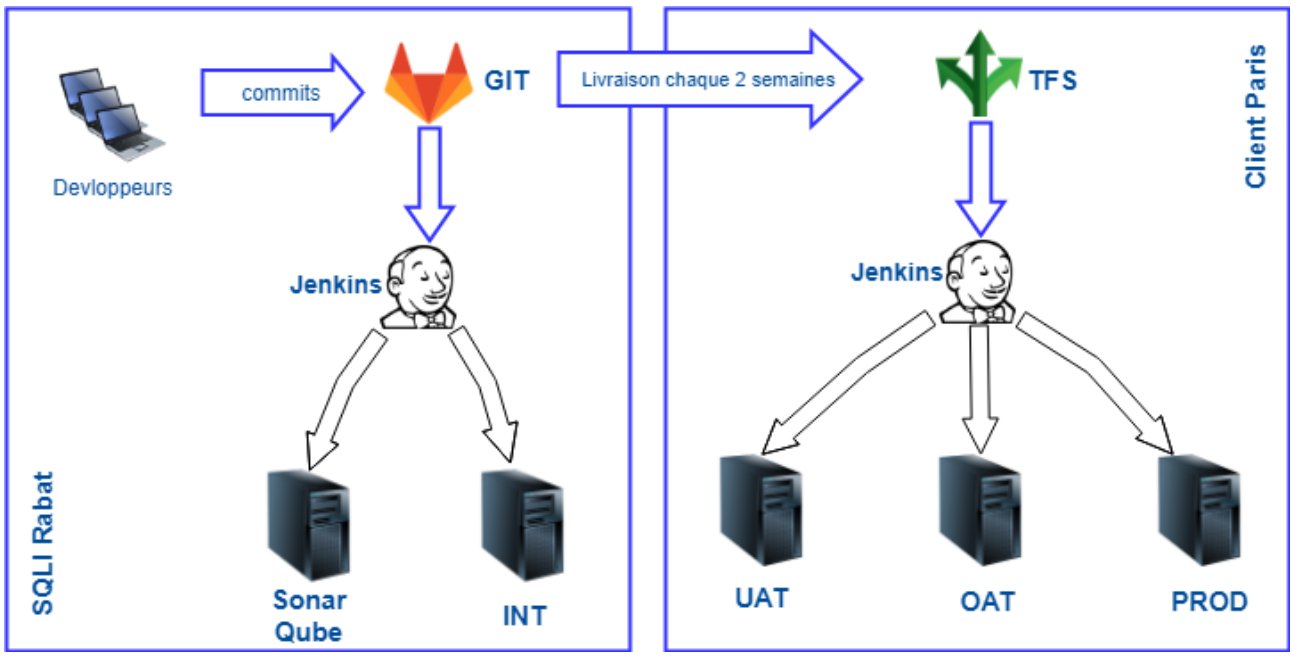

FIGURE 3.12 – Architecture physique du projet.

Les équipes de développement font des commits sur le serveur GIT (Gitlab), en suite un serveur d'intégration continue Jenkins (PIC ou Plate-forme d'Intégration Continue)

2. https://www.sonarqube.org/

récupère les versions produites depuis le GIT pour faire un build automatique avec ANT (un outil d'automatisation des opérations de build ou compilation des applications JAVA [5]) et lance une analyse Sonar depuis le serveur SonarQube sur le code afin de garantir sa qualité.

Après avoir fais les analyses Sonar le serveur vérifie que le nombre d'erreurs trouvé ne dépasse pas les limites prédéfinies selon le degré de gravité des erreurs et les quotas définis dans le serveur SonarQube et notifie les développeurs des résultats d'analyse, puis il fait le déploiement dans le serveur d'intégration (INT).

Ce processus d'intégration continue se répète tous au longe du sprint qui dure deux semaine et se termine par le livraison d'une nouvelle version au client dans son système de contrôle des versions TFS (Team Foundation Server).

Le client dispose de trois serveurs dont deux serveurs de test et un serveur de production :

- Le serveur UAT (User Acceptence Testing) sert à tester les livrables et s'assurer qu'ils correspondent aux attentes de l'utilisateur final.
- Le serveur OAT (Operational Acceptence testing) est un serveur qui permet de déterminer si les livrables sont opérationnelles et près à intégrer l'environnement de production.
- Après ces différents processus de test et dans chaque trois mois une opération de mise en production (MEP) s'effectue dans le serveur PROD pour qu'il soit disponible en ligne à l'utilisateur final.

3.6 Conclusion

Ce chapitre présente une vue conceptuelle de la solution à mettre en place. Il expose les différents diagrammes UML pour mieux comprendre les fonctionnalités offertes et pour mieux représenter la communication entre les différents objets du projet. Le chapitre suivant, présente la partie mise en œuvre de l'application.

Webographie

[1] Wikipédia, "Sqli — wikipédia, l'encyclopédie libre." http://fr.wikipedia.org/w/index.php?title=SQLI&oldid=158242840, 2019. [Consulté le 7 avril 2019].

[2] Wikipédia, "Gestion des ressources numériques — wikipédia, l'encyclopédie libre." http://fr.wikipedia.org/w/index.php?title=Gestion_des_ressources_num%C3%A9riques&oldid=156929136, 2019. [Consulté le le 16 juin 2019].

[3] C. Pages, "Xml common business library (xcbl)." http://xml.coverpages.org/cbl.html, 2001. [Consulté le le 16 juin 2019].

[4] Wikipédia, "Sonarqube — wikipédia, l'encyclopédie libre." http://fr.wikipedia.org/w/index.php?title=SonarQube&oldid=159160337, 2019. [Consulté le le 16 juin 2019].

[5] Wikipédia, "Apache ant — wikipédia, l'encyclopédie libre." http://fr.wikipedia.org/w/index.php?title=Apache_Ant&oldid=154452007, 2018. [Consulté le le 16 juin 2019].