# Table des matières

[Table des matières 0](#_Toc50804912)

[Introduction générale 1](#_Toc50804913)

[Chapitre 1 : Etude de l’existant 3](#_Toc50804914)

[1 Introduction 4](#_Toc50804915)

[2 Organisme d’accueil 4](#_Toc50804916)

[2.1 Historique 4](#_Toc50804917)

[2.2 Fiche d’identification de l’entreprise 5](#_Toc50804918)

[2.3 Missions et Objectifs de l’entreprise 5](#_Toc50804919)

[2.4 L’organisation de Naftal : 6](#_Toc50804920)

[3 Présentation du système MSIQ 8](#_Toc50804921)

[3.1 Les demandes traitées par MSIQ 8](#_Toc50804922)

[3.2 Le flux suivi par les demandes : 9](#_Toc50804923)

[3.3 Reporting : 16](#_Toc50804924)

[4 Problématique 17](#_Toc50804925)

[5 Solutions proposées 18](#_Toc50804926)

[6 Conclusion 20](#_Toc50804927)

[Chapitre 2 : Analyse et Spécification des besoins 21](#_Toc50804928)

[1 Introduction 22](#_Toc50804929)

[2 Le langage UML 22](#_Toc50804930)

[3 Spécification des besoins 22](#_Toc50804931)

[3.1 Spécification des Besoins fonctionnels 22](#_Toc50804932)

[3.2 Spécification des Besoins non fonctionnels 23](#_Toc50804933)

[3.3 Spécification des Besoins semi formels 23](#_Toc50804934)

[4 Conclusion 31](#_Toc50804935)

[Chapitre 3 : Conception 32](#_Toc50804936)

[1 Introduction 33](#_Toc50804937)

[2 Diagramme de séquence 33](#_Toc50804938)

[2.1 Les diagrammes de séquence de notre système 34](#_Toc50804939)

[3 Règles de gestion 40](#_Toc50804940)

[4 Dictionnaire de données 40](#_Toc50804941)

[5 Modèle relationnel de données 43](#_Toc50804942)

[5.1 Du diagramme de classes au modèle relationnel de données : 43](#_Toc50804943)

[5.2 Notre modèle relationnel de données 45](#_Toc50804944)

[6 Conclusion 45](#_Toc50804945)

# Table des figures

[Figure 1 Organisation générale de naftal 7](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804969)

[Figure 2 Localisation géographique de la branche GPL 7](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804970)

[Figure 3 La structure de la branche GPL 8](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804971)

[Figure 4 Représentation d'activités 10](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804972)

[Figure 5 Représentation des éventements 10](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804973)

[Figure 6 Représentation du Flux séquentiels 10](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804974)

[Figure 7 Représentation des portes 10](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804975)

[Figure 8 Représentation des Lanes 11](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804976)

[Figure 9 Représentation des annotations 11](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804977)

[Figure 10 Représentation des messages 12](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804978)

[Figure 11 Représentation des signaux 12](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804979)

[Figure 12 Représentation du minuteries 12](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804980)

[Figure 13 Représentation des erreurs 12](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804981)

[Figure 14 Représentation des boucles 13](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804982)

[Figure 15 Représentation graphique de flux de la demande client en BPMN 13](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804983)

[Figure 16 Représentation graphique de flux de la demande fourniture en BPMN 14](#_Toc50804984)

[Figure 17 Représentation graphique de flux de la demande véhiculet en BPMN 14](#_Toc50804985)

[Figure 18 Représentation graphique de flux de la demande de tirage en BPMN 15](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804986)

[Figure 19 Représentation graphique de flux de la demande PEC en BPMN 15](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804987)

[Figure 20 Représentation graphique de flux de la demande Relex en BPMN 15](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804988)

[Figure 21 Représentation du cycle de reporting 17](#_Toc50804989)

[Figure 22 Diagramme de cas d'utilisation général 25](#_Toc50804990)

[Figure 23 Diagramme de classes 31](#_Toc50804991)

[Figure 24 Représentation graphique d’un objet 33](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804992)

[Figure 25 Diagramme de séquence d’une demande véhicule 34](#_Toc50804993)

[Figure 26 Diagramme de séquence d’une demande client 35](#_Toc50804994)

[Figure 27 Diagramme de séquence d’une demande Fournitures 36](#_Toc50804995)

[Figure 28 Diagramme de séquence d’une demande Tirage 37](#_Toc50804996)

[Figure 29 Diagramme de séquence d’une demande De Prise En Charge 38](#_Toc50804997)

[Figure 30 Diagramme de séquence d’une demande D'Activité Relex 39](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804998)

[Figure 31 Représentation du principe de transformation 44](file:///C:\Users\Yacin\Desktop\Mémoire%201.0.2.docx#_Toc50804999)

# Introduction générale

Les architectures informatiques ne cessent d'évoluer et faire preuve d'une grande flexibilité. L'automatisation logicielle vous permet de gagner en efficacité, de générer de la valeur plus rapidement et de résoudre les problèmes liés aux workflows informatiques et métier.

Le workflow permet à ceux qui participent au processus d’effectuer un suivi du processus et de mettre en place les procédures adéquates permettant d’atteindre des objectifs escomptés.

Le workflow ne doit pas nécessairement être automatisé. Cependant, l’assistance d’une technologie permettant l’automatisation est sans doute la meilleure manière d’utiliser le workflow et d’obtenir les résultats souhaités. Workflow et automatisation sont bien souvent deux thématiques liées. Pour ce faire il existe plusieurs outils et méthodes à suivre comme ITIL.

ITIL est le sigle pour Information Technology Infrastructure Library qui se traduit par Bibliothèque pour l’infrastructure des technologies de l’information. C’est un projet ambitieux développé en Grande-Bretagne dans le milieu des années 1980 dans le but de documenter et d’archiver un maximum d’informations et de connaissances possibles au sujet des meilleures pratiques et développements du[BPM](https://www.heflo.com/fr/outil-de-modelisation-des-processus-metiers/) (Business Process Management) ayant un rapport avec l’IT (Information Technology)[01], que ce soit par le biais d’études de cas pratiques, ou via des manuels et des ouvrages spécialisés sur le sujet. La définition ITIL désigne l’ajout d’une série d’avantages et fournit la gestion des services et des Technologies de l’information de manière beaucoup plus efficace et efficiente en les alignant correctement par rapport aux risques et aux exigences de votre entreprise.

Dans ce cadre, nous nous intéressons a ce présent travail à étudier et développer un système de traitement de demandes client selon ITIL v3.Nous étudions cette norme et ses principales formalisations. Cette dernière va permettre à notre système de bénéficier de plusieurs avantages comme la bonne qualité de service, la cohérence des processus, l’efficacité de prestations des services et la réduction des couts des technologies de l’information, ce qui le permettra d’être intégré au sein de différentes type d’entreprises.

Notre étude s’articule autour de 4 chapitres :

* Le premier chapitre : Introduction et étude de préalable
* Le deuxième chapitre : Spécification des besoins
* Le troisième chapitre : Conception
* Le quatrième chapitre : Implémentation

# Chapitre 1 : Etude de l’existant

## Introduction

Dans ce chapitre, nous présentons une étude globale sur l’organisme d’accueil « NAFTAL branche GPL », son historique et le système d’automatisation existant  « MSIQ ».. Nous définissons par la suite les imperfections existantes dans MSIQ, et proposer des solutions pour développer le système qui a pour but l’automatisation de traitement des différentes demandes clients.

## Organisme d’accueil

Naftal est une entreprise pétrolière algérienne, spécialisée dans la distribution des produits pétroliers. Elle est aussi spécialisée dans la conception, l'élaboration et la distribution de lubrifiants pour moteurs (deux-roues, automobiles et autres véhicules) ainsi que pour l’industrie. [02]

### Historique

Issue de SONATRACH, l’entreprise ERDP (entreprise national de raffinage et de distribution des produits pétroliers) a été créé par Le décrit N° : 80/101 du 06 avril 1981, entrée en activité en 1er Janvier 1982, L’ERDP est chargée de l’industrie du raffinage et de la commercialisation et distribution des produits pétroliers.

Le 25 Août 1987, L’activité raffinage est séparée de l’activité distribution. La raison sociale de la société change suite à cette séparation des activités. Naftal est désormais chargée de la commercialisation et de la distribution des produits pétroliers et dérivés, par la suite elle était transformée en société par actions au capital social de 6 650 000 000 DA et filiale à 100% du holding Sonatrach Valorisation des Hydrocarbures (SVH) en 18 Avril 1998. [03]

L’appellation NAFTAL se compose de :

* NAFT : terme arabe désignant le pétrole.
* AL : en référence à ALDJAZAIR ou ALGER. [04]

### Fiche d’identification de l’entreprise

|  |  |
| --- | --- |
| Création | 6 avril 1981 |
| Forme juridique | SPA |
| Slogan | Une relation de confiance |
| logo |  |
| Siège social | Route des dunes 16002 Chéraga (Alger)-Algérie |
| Direction | Kamel Benfriha |
| Actionnaires | Sonatrach (100%) |
| Activité | Distribution des produits pétroliers |
| Produits | Carburants, lubrifiants, bitumes, pneumatiques |
| Société mère | Sonatrach |
| Filiales | BAG |
| Effectif | 31 583 (2018) |
| Site web | [www.naftal.dz](http://www.naftal.dz) |
| Chiffre d'affaires | 40 000 000 000 DA |

Tableau 1 Fiche d’identification de l’entreprise

### Missions et Objectifs de l’entreprise

Naftal a pour mission principale : la distribution et la commercialisation des produits pétroliers et dérivés sur le marché national.

* Elle intervient également dans le domaine de :
  + L’enfûtage des GPL.
  + La formulation des bitumes.
  + La distribution, le stockage et la commercialisation des carburants, GPL, lubrifiants, bitumes, pneumatiques, GPL/carburant, produits spéciaux.
  + Le transport des produits pétroliers.
* Pour assurer la disponibilité des produits sur tout le territoire, Naftal met à contribution plusieurs modes de transport :
* Le cabotage et les pipes, pour l’approvisionnement des entrepôts à partir des raffineries.
* Le rail pour le ravitaillement des dépôts à partir des entrepôts.
* La route pour livraison des clients et le ravitaillement des dépôts non desservis par le rail.

A 1’ère de la mondialisation, Naftal a jugé indispensable la mise en place d’une nouvelle organisation par ligne de produit (bitumes, lubrifiants, réseau, logistique, GPL, pneumatique, Aviation, Marine).

Naftal fournit près de 13,3 millions de tonnes de produits pétroliers par an, un chiffre appelé à augmenter avec une demande en constante croissance.

Elle a également mis en place une nouvelle vision stratégique à moyen terme orientée client avec un plan de mise en œuvre.

A travers son plan de développement NAFTAL vise un double objectif :

Poursuivre sa mission de distribution des produits pétroliers.

Améliorer sa qualité de service.

### L’organisation de Naftal :

#### Organisation générale :

La nouvelle organisation de NAFTAL est datée du 1 janvier 2006 [02]. Dans un contexte de marché marqué par de profonds bouleversements, NAFTAL, pour assurer sa pérennité à l’ambition de se hisser au niveau des meilleures entreprises pétrolières. La dernière organisation de la société NAFTAL, résultat de la séparation des activités commercialisation et carburant est structurée comme suit : Une direction générale qui comporte deux types de structures :

• Des structures fonctionnelles, elles sont subdivisées en deux (02) types de directions qui sont :

- Direction exécutive.

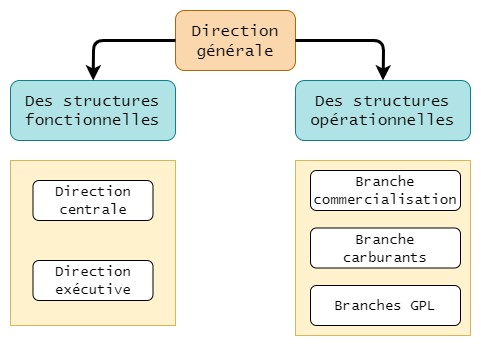
- Direction centrale.

• Des structures opérationnelles, contenant quatre (03) branches :

- Branche carburants.

- Branche commercialisation.

- Branches GPL.



Figure

I.1

–

Organisation

Générale

De

naftal

#### Organisation de la branche GPL d’el Mohammedia

Le stage a été effectué dans le siège dans la branche GPL d’el Mohammedia, plus précisément dans la direction informatique qui est responsable de mettre des solutions traitant les différentes problématiques.

#### Location :

Figure 2 Localisation géographique de la branche GPL

#### Structure :

La

branche

GPL

d’el

Mohammedia

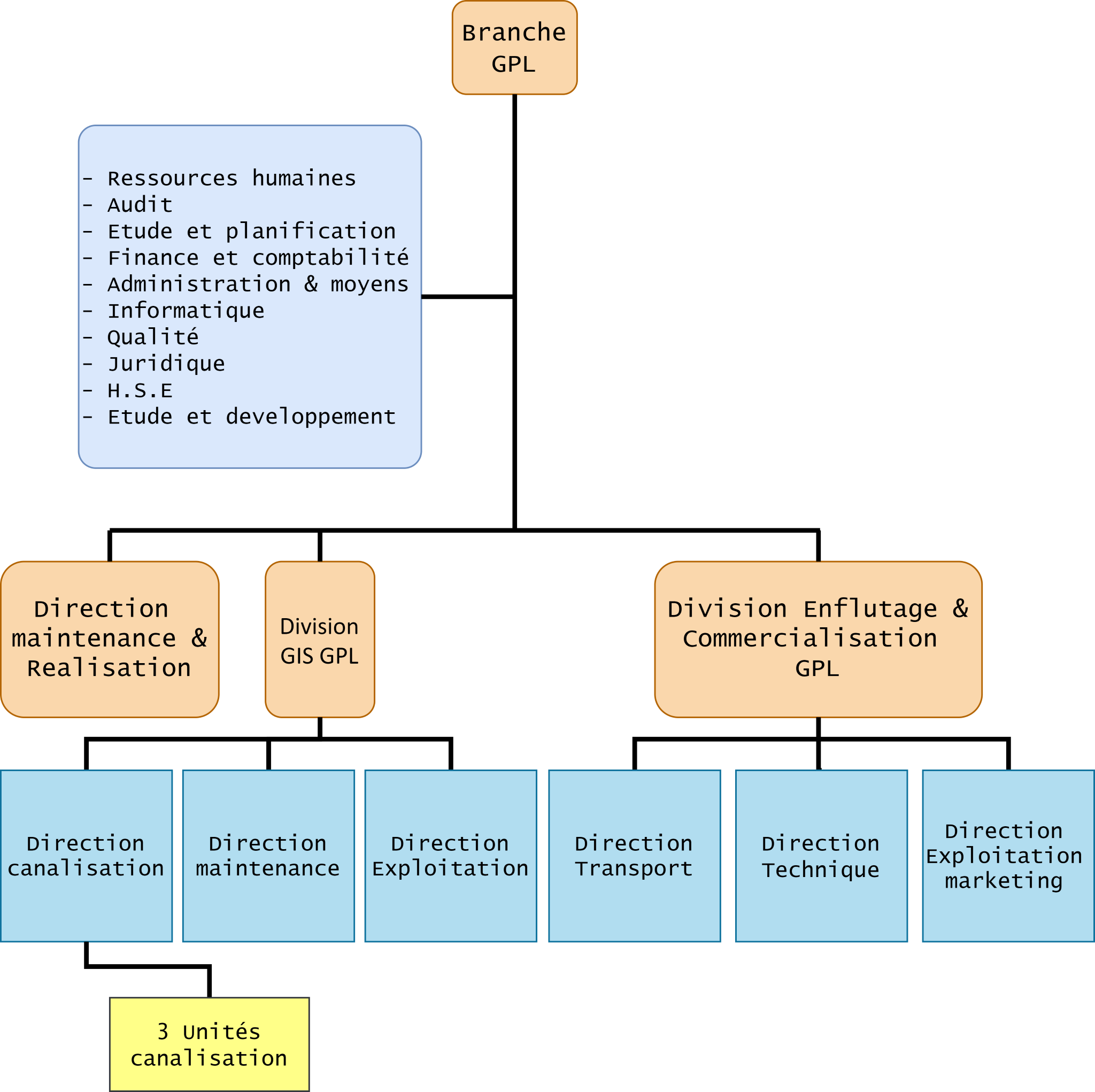
est

structurée

comme

suit

:



Figure

I.3

–

La

structure

de

la

branche

GPL

Notre système est basé sur ce modèle globale qui structure toutes les branches GPL.

## Présentation du système MSIQ

MSIQ est un système utilisé dans l’annuaire de l’entreprise traitant différents types de demandes envoyées par ses utilisateurs, permettant ainsi aux supérieurs la validation ou le rejet de ces demandes selon la hiérarchie et l’organisation de l’entreprise.

### Les demandes traitées par MSIQ

#### Demande client :

Les travailleurs ont besoin des produits comme les chaises, les tables, les ordinateurs …etc. et parfois la réparation de certaines machines comme les PC et les imprimantes, et même l’installation de certains logiciels nécessaires pour le travail.

Le travailleur peut faire soit une demande d’un service ou d’un produit et il décrit cette demande dans son formulaire.

#### Demande de fourniture :

Ce type de demandes concerne les produits bureautiques comme les crayons, papiers... etc.

Un employé prépare une liste des produits à demander avec leurs quantités.

#### Demande de tirage :

Afin d’éliminer le cout d’impression des documents telle que les rapports, les cahiers de charges, les appels d’offre…, les employés peuvent faire des demandes de tirage pour que documents s’impriment dans la salle tirage. Le formulaire de cette requise au client de décrire la structure qui a fait la demande et la décrire elle-même (photocopie, reluire, tirage de plan, autres) et les informations de document (titre, nombre feuilles, nombre exemplaire).

#### Demande Véhicule :

Les employés qui sortent dans des missions ou bien transporte des articles ont besoin des véhicules. Et donc il fait une demande de véhicule dans laquelle il doit citer la structure où il travaille, la liste des demandeurs, la destination, la date et heure de départ et ramassage et aussi l’organisme d’accueil, le motif de déplacement et la nature de marchandise transportée (dans le cas de transport de marchandise).

#### Demande de prise en charge :

Les employés qui sortent dans des missions hors willaya doivent prendre des vols, la structure dont ces employés font partie fait donc une demande de prise en charge pour tous les employés concernés par une même mission.

Lors de l’établissement de cette demande, la structure doit spécifie la destination, l’objet et date de la mission, Moyens de transport, heure du vol, aéroport.

#### Demande activité relex

Dans les missions, les employés peuvent demander la restauration, l’hébergement et transport, dans ce cadre, ils doivent établir une demande activité relex.

Le demandeur doit citer ses informations personnels et l’objet mission avec la destination, la date et l’heure de départ, et la date et l’heure retour et moyens de transport

Si le demandeur utilise un véhicule d’entreprise, il synchronise une demande de véhicule avec cette demande.

### Le flux suivi par les demandes :

Quand la demande est soumise, elle suit un flux de traitement, nous le représentons par l’aide du BPMN 2 (Business Process Model & Notation).

#### Qu’est-ce que BPMN :

Commençant par la définition du BPM (Business Process Manager) :

* BPM – Business Process Management : est Une approche de gestion des processus visant à améliorer les performances métier.
* BPMN – Business Process Model and Notation : est Une représentation graphique permettant de spécifier des processus métier à partir d'un modèle.

La notation consiste en un ensemble de symboles graphiques qui représentent des actions, des flux ou des comportements de processus

* **Les éléments BPMN en 4 catégories :**

Nous pouvons l'aborder en classant les éléments BPMN dans plusieurs grandes catégories.

*Workflow* : Ils incluent les activités, portes et événements, ainsi que les flux séquentiels qui les lient.

* Activités: Tâches réalisées dans un processus par un humain, un système ou qui activent des sous-processus

Activité

Figure 4 Représentation d'activités

* Evénements: Utilisés pour débuter ou finir un processus et pour gérer des actions spécifiques pendant un workflow ; ils déclenchent ou sont le résultat d'une action extérieure au flux du processus.

Début

Intermédiaire

Fin

Figure 5 Représentation des éventements

* Flux séquentiels : Utilisés pour indiquer la progression du workflow.

Activité 1

Activité 2

Figure 6 Représentation du Flux séquentiels

* Portes : Utilisées pour dissocier ou réunir des flux de processus



Figure 7 Représentation des portes

*Organisation*: Ils incluent les pools et les lanes. Ce sont des conteneurs pour le flux de processus.

* Pools : Contiennent un processus unique et complet. Un workflow ne peut pas sortir d'un pool : il convient de transférer les actions ou les données d'un pool/processus à un autre par d'autres moyens
* Lanes : Utilisées pour organiser le processus en fonction de qui fait quoi (acteurs). Dans une piscine, les lignes d'eau servent à séparer les nageurs afin qu'ils ne se percutent pas

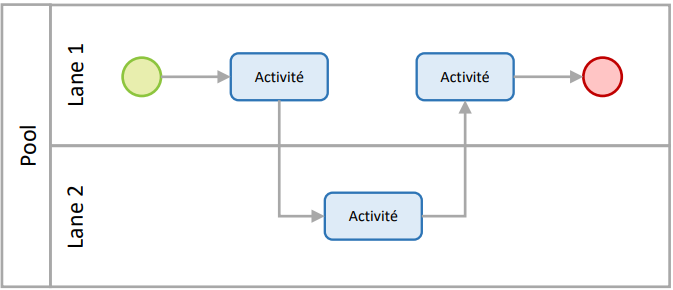
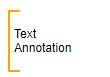


Figure 8 Représentation des Lanes

*Lisibilité* : Ils incluent les annotations et les liens. Ces éléments servent à faciliter la lisibilité d'un diagramme. Ils n'ont aucun effet sur le flux de processus.

* Annotations : Permettent de rajouter des notes sur un diagramme afin de le clarifier (un excellent outil pour les débutants !)



Lien destination

Lien source

Figure 9 Représentation des annotations

* Liens : Ils permettent de couper un processus qui est devenu trop long afin qu'il soit facilement lisible, et de continuer le processus sur une autre ligne dans le même pool.

*Comportements spécifiques* : Les comportements spécifiques incluent un ensemble d'événements et de marqueurs de tâches. Ces éléments nous permettent de concevoir un workflow exécutable se comportant de manière complexe.

* Messages : La corrélation est utilisée pour coordonner l'avancement entre deux instances de processus en cours et mettre en correspondance les événements des messages.



Envoi de message

Reception de message

Figure 10 Représentation des messages

* Signaux : Utilisés pour diffuser des informations d'un processus donné vers plusieurs autres processus.



Envoi de signal

Envoi de signal

Figure 11 Représentation des signaux

* Minuteries : Utilisées pour démarrer périodiquement des activités ou pour vérifier qu'une activité s'est déroulée dans un délai défini.
* Erreurs : Utilisées pour définir le comportement lorsque le système rencontre une erreur.

Figure 13 Représentation des erreurs



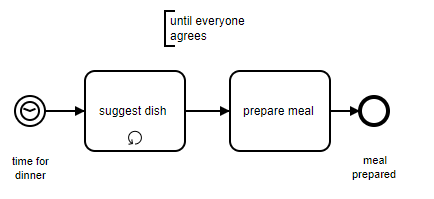
Minuterie



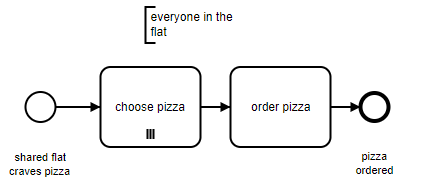
Erreur

Figure 12 Représentation du minuteries

* Boucles et Multi-instance : Utilisées pour répéter des tâches telles que de multiples lancements de la même tâche (multi-instance) ou la répétition de la même tâche (boucle).



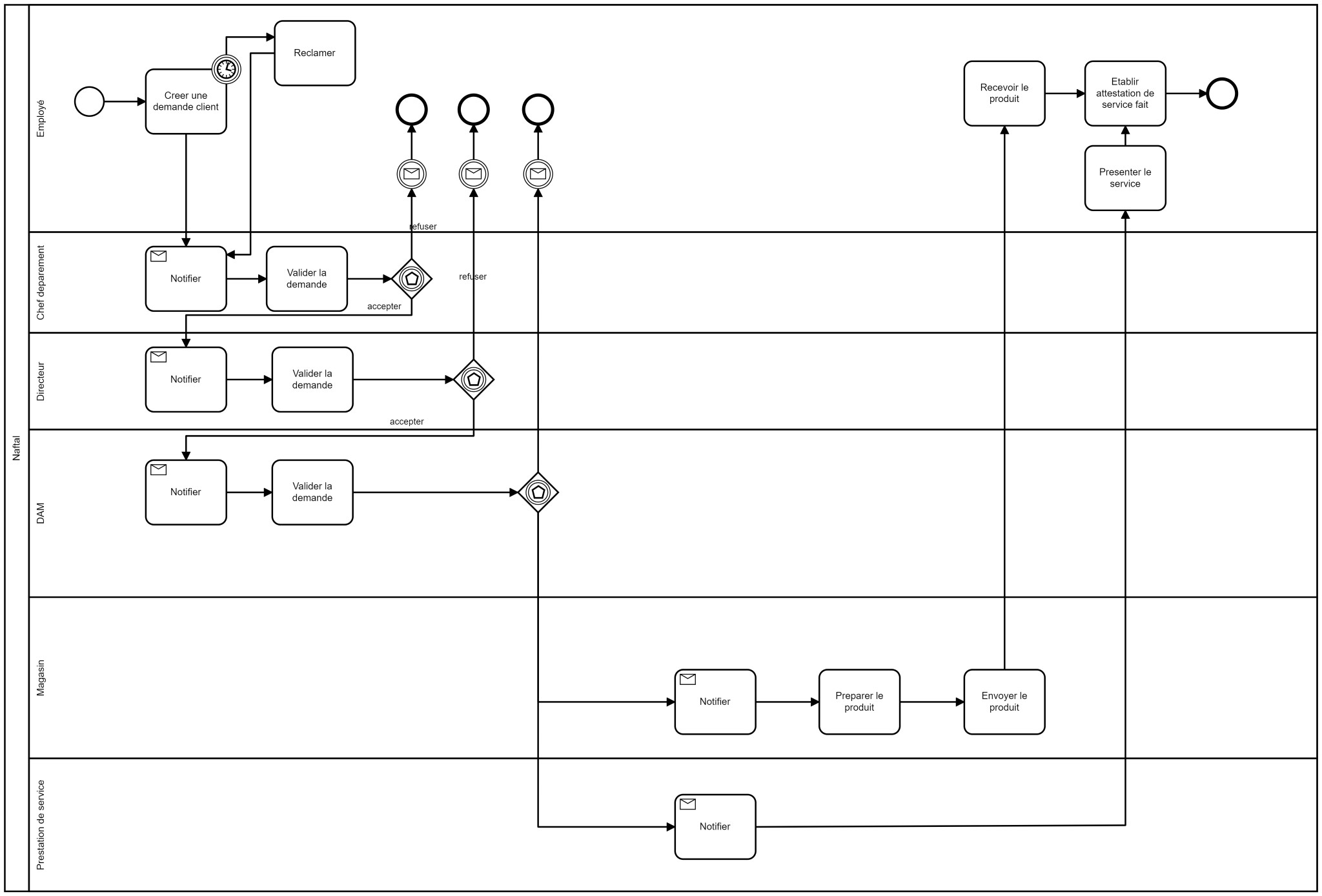
Boucle



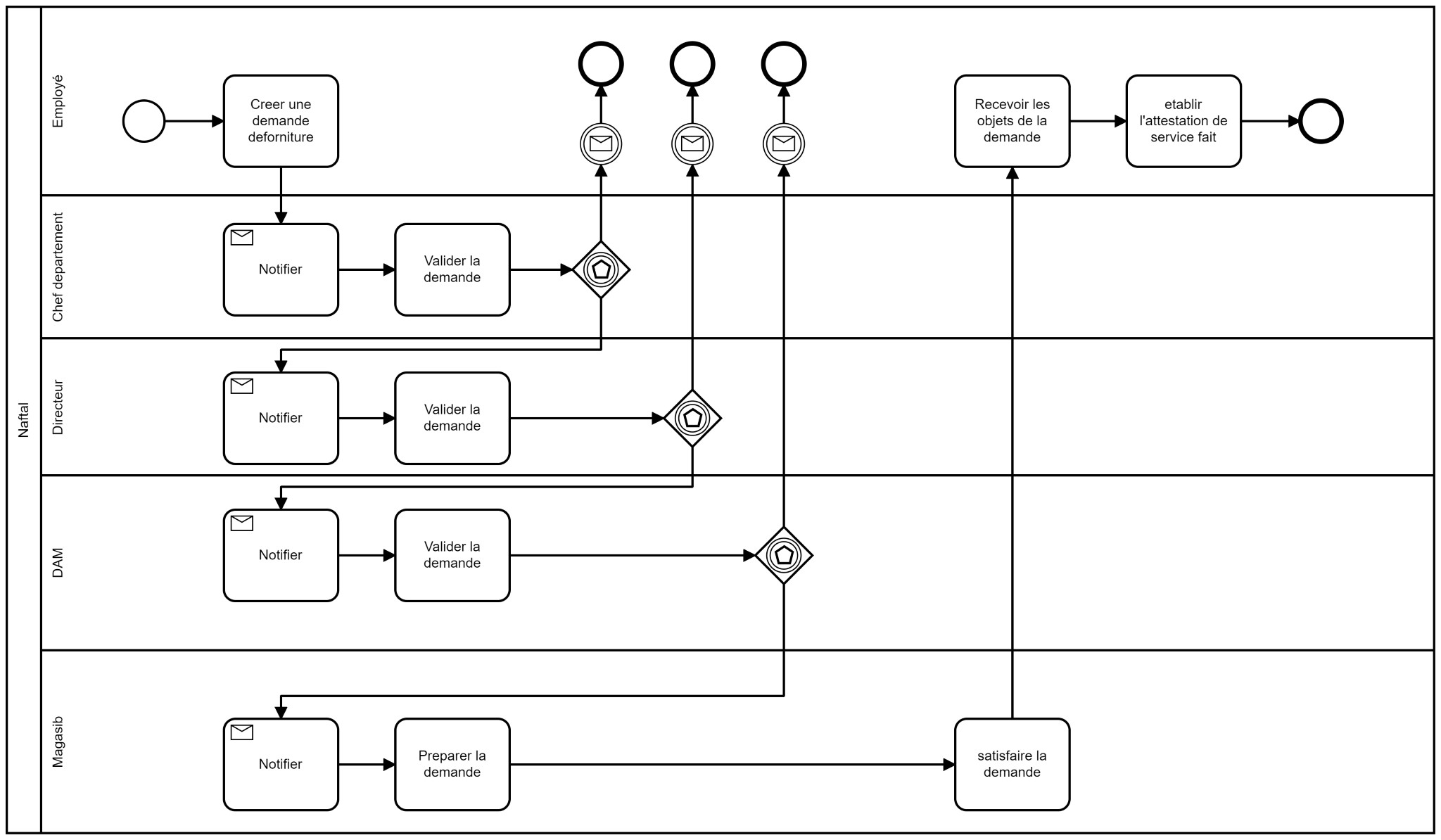
Multi-instance

Figure Représentation des boucles

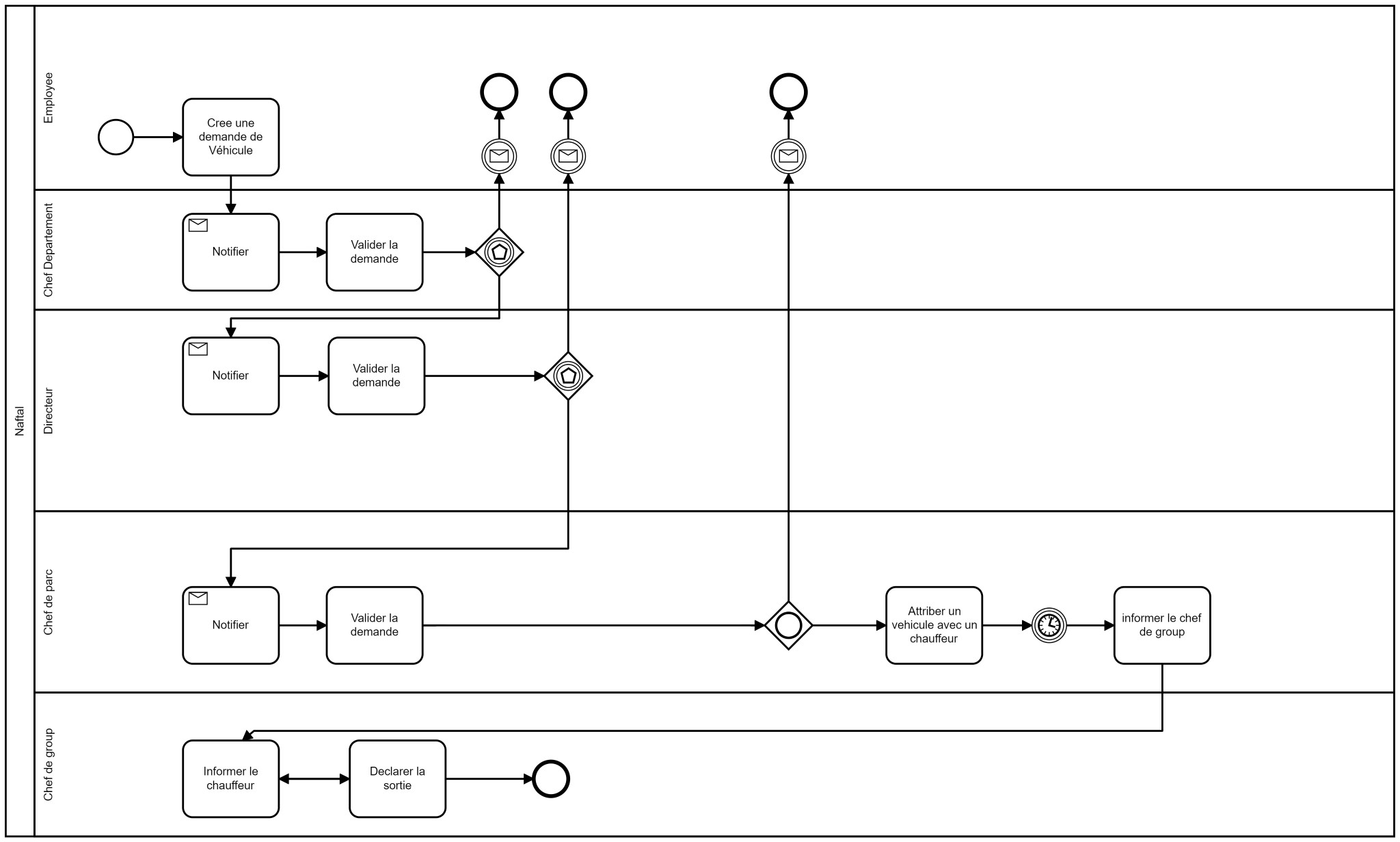
1. ***Flux de la Demande client en BPMN***



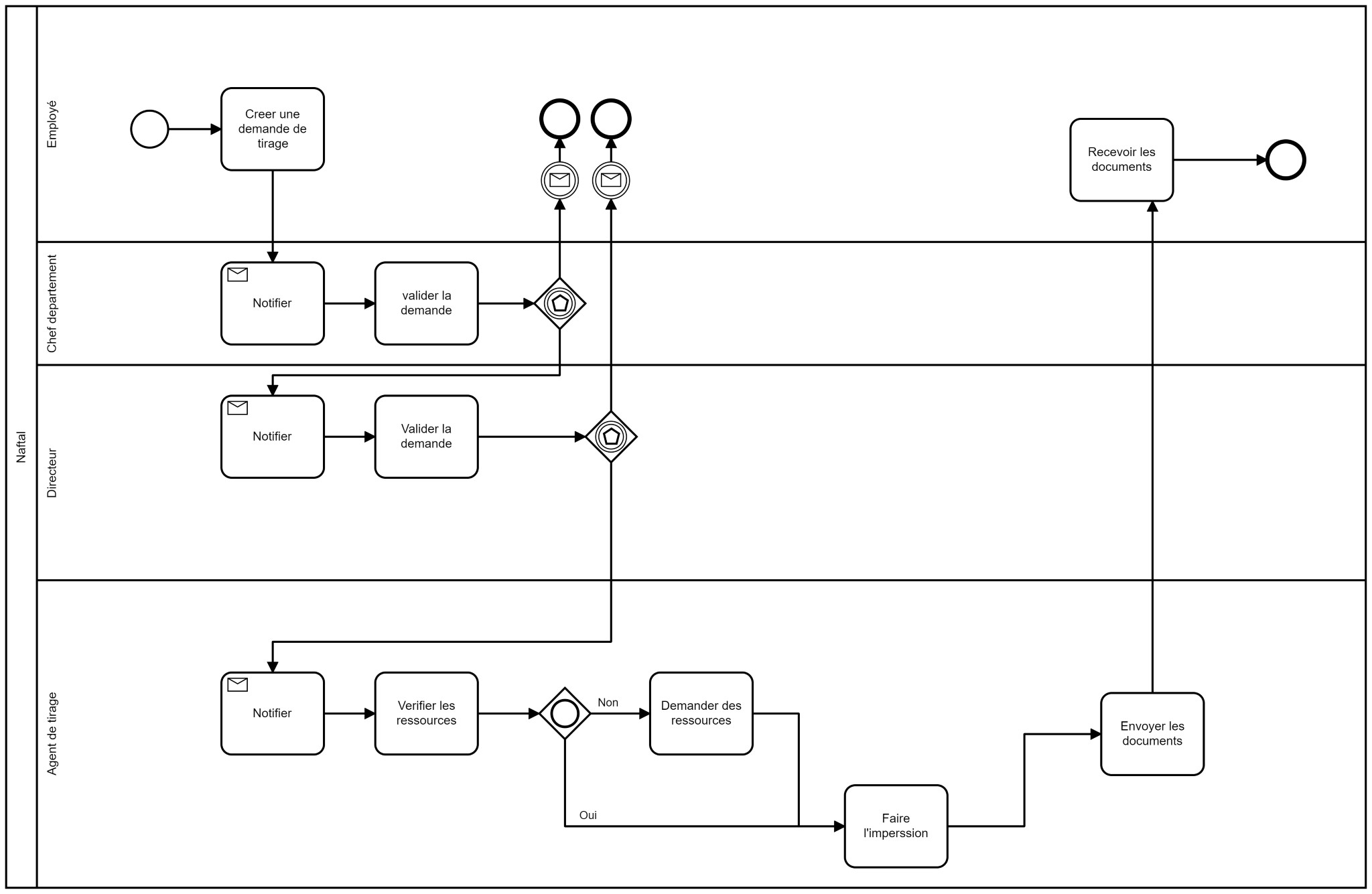
***2. Flux de la Demande de fourniture en BPMN***



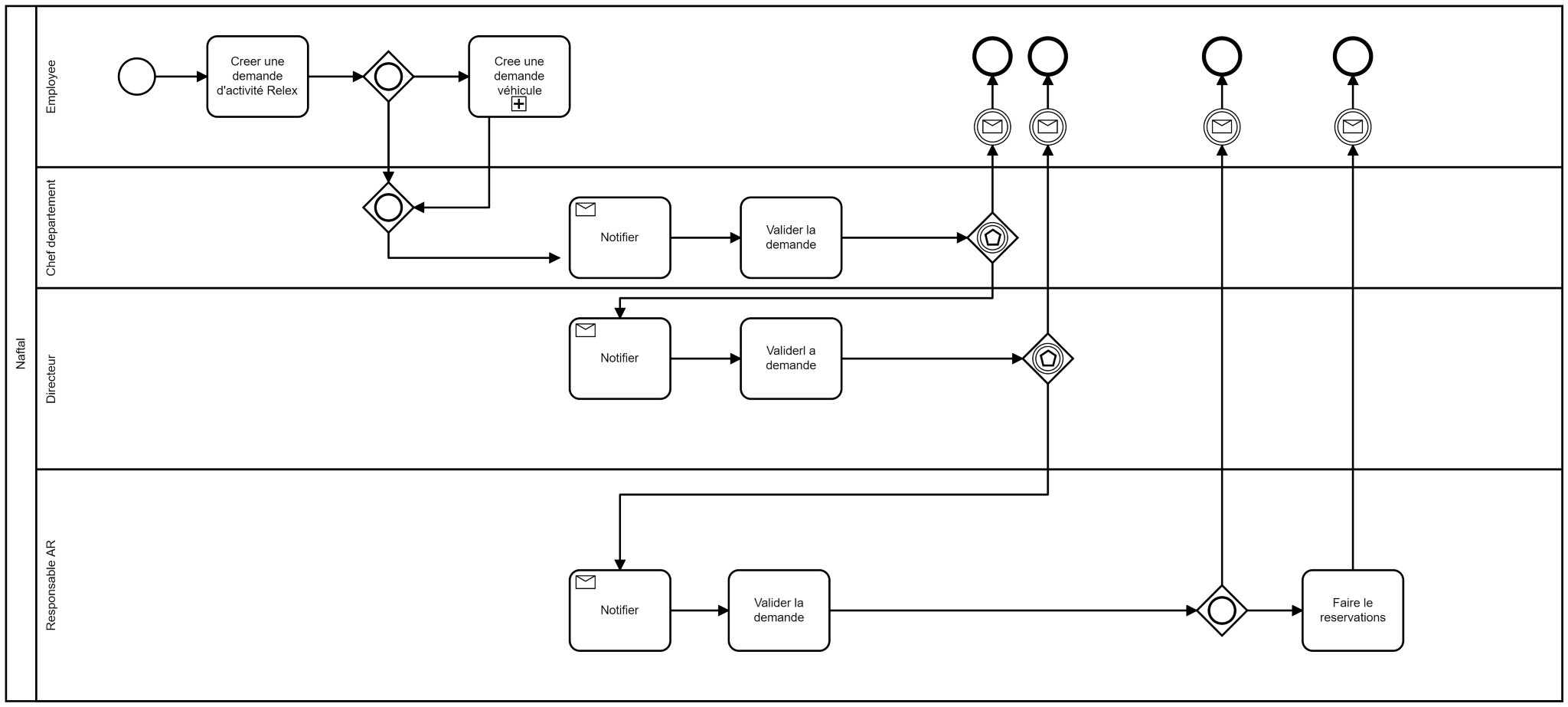
**3. flux de la Demande de véhicule en BPMN**



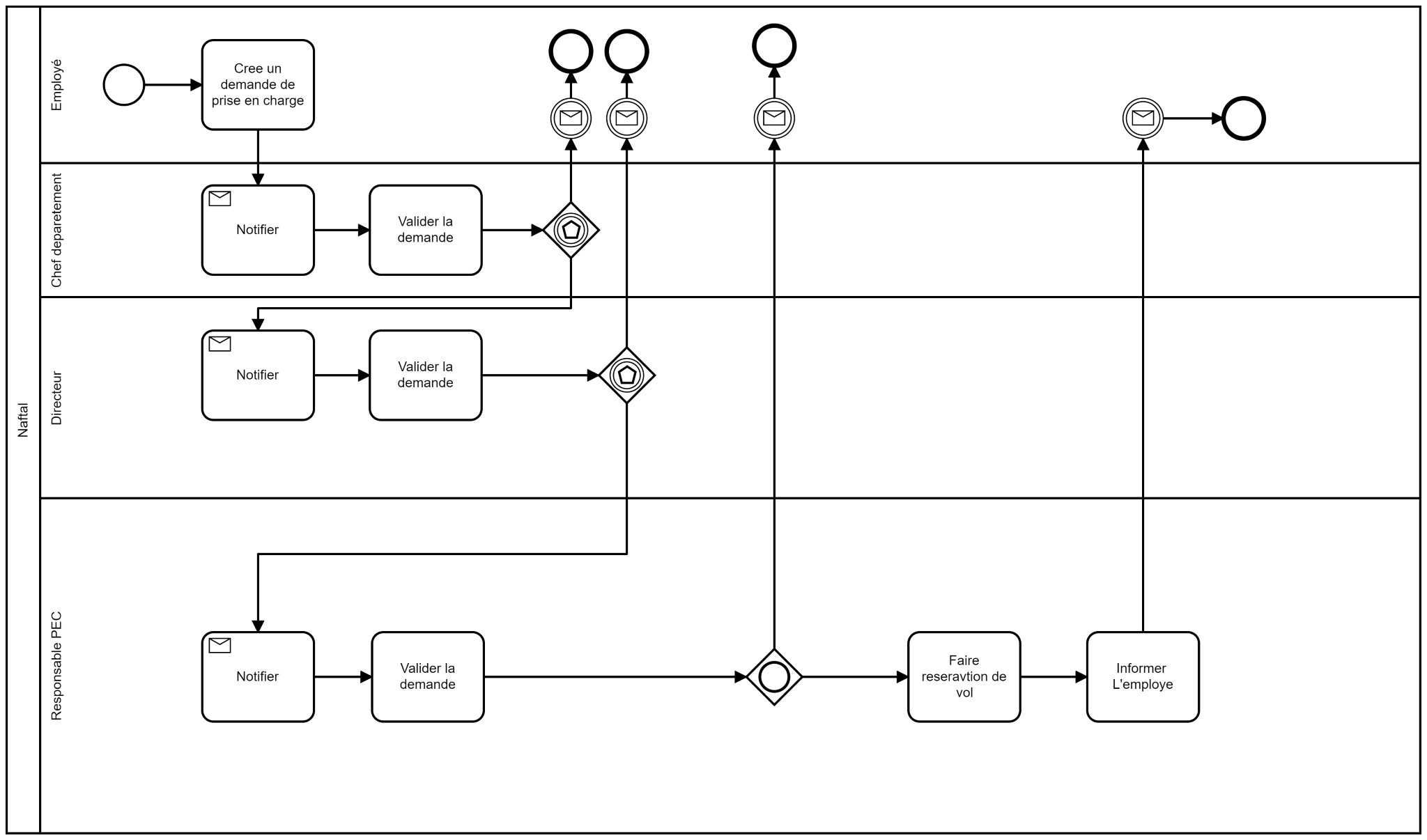
***4. Flux de la Demande de tirage en BPMN***



***5. Flux de la Demande d’activité Relex en BPMN***



***6. Flux de la Demande de prise en charge en BPMN***



Envoi de signal

Comme il représenté dans les graphes, les demandes client et de fourniture ont un flux similaire depuis la soumission de jusqu’à l’arrivée au chef de département de service.

Les demandes de véhicule, tirage, activité relex et de prise en charge sont traitées différemment.

Quand un client fait une demande, il la soumet, et elle s’envoie à son chef du département, ce dernier va par la suite la transférer au directeur, qui va soit l’accepter et l’envoyer au département DAM, soit la refuser quand il y a des motifs de refus.

Minuterie

Minuterie

Le département des Moyens DAM va par la suite transférer la demande au chef du service correspondant :

* Si c’est une demande client ou de fourniture, ça va être transféré au magasin

Pour les demandes véhicule, tirage, d’activité relex et de prise en charge, quand le directeur accepte la demande, elle va être transférée vers :

* Le chef de parc qui traite la demande de véhicule.
* L’agent de la salle de tirage qui prend charge de la demande de tirage.
* le service RELEX qui va établir l’ordre de mission pour la demande Activité RELEX.
* Le service PEC qui va traiter la demande de prise en charge.

### Reporting :

Comme tout budget qui essaye d’anticiper l’avenir, le reportant sert à analyser le passé, c’est une manière d’échange d’information au sein de l’entreprise. Parfois les chiffres sont produits par la direction générale puis redescendu vers les entités opérationnelles, on appelle ça top-down, et parfois c’est le sens inverse, et on l’appelle bottom-up [17].

Le reporting n’est pas une observation figée de la réalité à un temps T, c’est une série d’observation à intervalles réguliers qui permet de comprendre des évolutions et de construire les budgets de demain.

Le reporting ne néglige pas la partie ‘explication des chiffres’.

Les processus des budgets reporting sont les cycles qui se répètent dans le temps :

1- Agir : On fait des plans

2- Mesurer : On regarde comment ça se passe dans la réalité

3- Ajuster : On prend des actions en conséquence

4- Prévoir : On refait des plans.

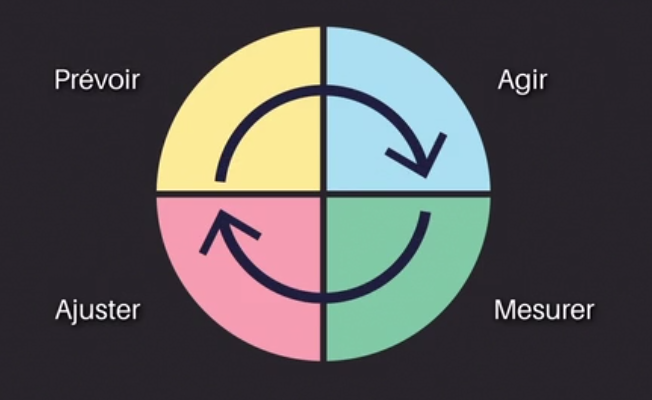


Figure 21 Représentation du cycle de reporting

La réduction des coûts demeure la principale préoccupation des responsables informatiques. Les départements informatiques ont l’injonction d’en faire davantage avec moins pour leur organisation. Par chance, les barrières entravant l’efficacité du reporting ont elles aussi le mérite de pointer vers la solution. Pour réduire leur coût total de possession, les organisations ont tout avantage à utiliser une seule et même application de reporting qui réduirait pour l’informatique la complexité de l’environnement de reporting tout en apportant davantage de choses à ses utilisateurs en entreprise. Pour être complète, une solution de reporting doit [18]:

• savoir identifier et traiter différentes sortes d’utilisateurs.

• couvrir tous les types de rapports.

• accéder à la totalité des données de l’entreprise, quelle que soit leur source.

## Problématique

Suite à notre étude et analyse du système existant MSIQ, nous avons constaté qu’ils présentent des lacunes et des insuffisances qu’on peut les articuler comme suit :

Le système MSIQ est construit à partir des idées brutes :

* L’absence de système de profilage.
* L’absence d’une définition claire de certaines demandes comme la demande client et la demande de fourniture (la différence n’est pas assez claire)
* Le manque de l’information exacte sur le flux de traitement.
* Les utilisateurs ne sont pas bien informés sur le flux.
* Lors d’une demande client, la prestation des services informatique (installation, configuration...) est faite par un autre système carrément.

Le système MSIQ n’a pas pris en considération des besoins importants des clients :

* Le responsable de Parc doit savoir la liste des chauffeurs et véhicules disponibles.
* Lors d’une demande de tirage, le responsable de cette tache doit faire un travail manuel qui prouve qu’il a bien servi le demandeur.
* Le responsable de parc doit avoir la possibilité de gérer le parc : supprimer ou ajouter des véhicules ou des chauffeurs.
* Le chef poste de garde de parc doit déclarer la date et l’heure réelle de sortie et de retour de véhicule.
* L’agent de tirage doit faire une demande papier.
* Les clients doivent avoir un système de messagerie intégré dans le site
* Les utilisateurs doivent être notifiés sur l’état de leurs demandes
* Les directeurs et chef de départements doivent aussi être notifiés lors de la soumission des nouvelles demandes.

Les fonctionnalités qui ne sont pas mise en œuvre dans le système MSIQ :

* La demande prise en charge
* La demande activité relex
* Reporting
* Manque d’archivage automatique des taches.

## Solutions proposées

En suivant le référentiel d’itil, on va pouvoir déterminer qui fait quoi dans notre système et comment.

ITIL est avant tout un ensemble de proposition à adapter

L’avantage qui donne ITIL :

Centré utilisateurs

Qualité de service

Couts réduits

Adaptable

ITIL est un très bon point de départ quand il s’agit d’organiser & d’améliorer la qualité de son service.

Les concepts clés d’ITIL sont : processus, fonction, rôle.

Processus : sert à transformer un ou plusieurs éléments d’entrée en éléments de sortie, il est mesurable qualitativement et quantitavement.

Les activités d’un processus sont réalisées par des intervenants

Ces intervenants sont les fonctions, ils peuvent etre des :

* Personnes
* Groupes
* Outils

A chaque fonction on attribue un ou plusieurs rôles.

Un rôle est un ensemble de responsabilités et d’autorités.

Dans sa mise à jour V3, ITIL a introduit la notion ‘cycle de vie d’un service’, Il s’agit des étapes que le service passe depuis sa naissance à son exploitation quotidienne.

Il existe 5 phases d’exploitation d’un service

1- stratégie du service : 5 processus dans cette première phase de service nous élaborons la direction stratégique de service :

* Exigences,
* Stratégie,
* Couts.

2- conception du service : 8 processus :

* Niveau de service ;
* Capacité ;
* Disponibilité ;
* Sécurité.

3- transition du service :7 processus :

* Conception,
* Test,
* Déploiement du service.

4- Exploitation du service : 5 processus :

* Maintenance,
* Coordination des activités

5- l’amélioration continue du service : 1 processus :

* Audit des processus,
* Plan d’action

Cela permet de voir les forces et faiblesse des services et permet d’établir un plan d’action.

Ça permet de comprendre les points suivants :

* Pourquoi le client a besoin de ce service ?
* Quel type de service fournir ?
* Quelle solution mettre en œuvre ?
* Comment exploiter ce service au quotidien ?

Et dons nous avons commencé par se remettre en question, par mesurer l’efficacité du service existant que MSIQ fournit, et là nous allons identifier le standard de qualité à atteindre.

Par la suite notre stratégie pour se faire se base sur l’amélioration de la qualité de service actuelle en proposant des nouvelles fonctionnalités, décris comme suit :

* Clarifier la prestation des services lors d’une demande client.
* //ajouter
* //ajouter.

## Conclusion

Dans ce premier chapitre, nous avons fait l’étude de l’existant en spécifiant les détails nécessaires pour commencer le travail sur notre système.

En pensant à l’intégration des référentiels D’ITIL V3, nous avons met une stratégie dans ce chapitre en spécifiant toutes les problèmes et lacunes rencontrés dans l’existant et en planifiant des solutions possibles pour les remédier.

Dans le chapitre suivant, nous allons commencer la spécification des besoins et l’analyse fonctionnelle qui appartient à l’étape deux (02) de référentiel ITIL qui est l’étape de conception

# Chapitre 2 : Analyse et Spécification des besoins

## 

## Introduction

L'étape de l'analyse des besoins est considérée comme l’une des étapes les plus importantes de cycle de vie d’un projet.

Notre objectif dans cette étape est donc d'exprimer les besoins attendus de le future Système à développer ; nous allons commencer par définir le langage de modélisation UML, ensuite nous procéderons à l’élaboration des diagrammes des cas d'utilisation, mettant en évidence la réalisation de notre projet.

## Le langage UML

UML (*Unified Modeling Language* ou langage unifié de modélisation) est un langage graphique destiné à la modélisation de systèmes et de processus. UML est un langage basé sur l’approche par objets, celle qui a d’abord conduit à la création des langages de programmation comme Java, C++ ...etc. [06]

UML est unifié car il provient de plusieurs notations qui l’ont précédé. Aujourd’hui, UML est promu par l’OMG (*Object Management Group*), un consortium de plus de 800 sociétés et universités actives dans le domaine des technologies del’objet [07].

Notre idée est qu’UML deviendra un langage de modélisation très répandu, notamment grâce à sa richesse sémantique qui le rend abstrait de nombreux aspects techniques.

## Spécification des besoins

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

### Spécification des Besoins fonctionnels

Les besoins fonctionnels expriment une **action** que doit effectuer le système en réponse à une demande (sorties qui sont produites pour un ensemble donné d’entrées).

Notre système doit permettre à un employé qui le client de faire plusieurs types de demandes. Les demandes se ressemblent sur quelques champs et se diffèrent sur d’autres, ces derniers sont à remplir par l’utilisateur, qui par la suite les soumet en attendant soit une réponse favorable ou non favorable par son supérieur.

Le système doit permettre au chef des départements et aux directeurs de recevoir les demandes envoyées par les employés, les traiter et les valider ou refuser selon certains motifs.

Les utilisateurs auront le pouvoir de consulter la réponse envoyée en cas du refus avec le motif, sinon ils reçoivent l’attestation de service fait.

Le système doit générer un reporting.

### Spécification des Besoins non fonctionnels

Ce sont des besoins qui caractérisent le système. Notre système doit répondre aux critères suivants :

* La sécurité : il devra être bien sécurisé, l’accès aux rôles des chefs de départements et des directeurs devra être restreint, chaque employé aura son compte spécial avec son nom d’utilisateur et son propre mot de passe.
* La performance : en assurant un temps de réponse optimal, c’est-à-dire le système devra réagir dans un délai précis quel que soit l’action de l’utilisateur.
* La disponibilité/Fiabilité : exigence de disponibilité sur toutes les périodes de travail.
* L'ergonomie de l'interface : l'interface devra être simple à utiliser.
* Utilisabilité : Capacité pour un utilisateur d’exécuter une tâche dans un temps donné après une formation d’une durée déterminée.

### Spécification des Besoins semi formels

Dans cette étape nous allons introduire les diagrammes UML correspondants, mais avant ça nous commençons par l’identification des acteurs de notre système

#### Identification des acteurs

Les acteurs sont les utilisateurs de système. Dans le cadre de notre étude, les acteurs sont :

* Le client :

C’est un employé quelconque au sein de l’entreprise, peu importe son grade, il s’authentifie et peut soumettre plusieurs types de demandes qu’il souhaite.

Le client peut consulter la liste de ses demandes, supprime et modifie une demande

Le client peut faire des modifications a son profile et aussi peut changer son mot de passe

* Le chef de département :

Il a le droit de consulter les demandes soumises par les employés de son département, par la suite il peut soit l’accepter et la transmettre à son supérieur, ou bien la refuser selon un motif précis.

* Le Directeur :

Il possède les mêmes droits qu’un chef de l’un de ses départements, sauf que dans le cas d’accord il transmet la demande au directeur de DAM.

Le directeur peut consulter reporting de sa structure.

Le directeur peut consulter la liste des employées de sa structure, il peut supprimer un employé ou modifie le type d’employé.

* Le responsable de DAM :

Il consulte les demandes reçus et les transmet aux exécutants finaux de la tâche.

* Agent de tirage :

Il reçoit une demande de tirage, il l’exécute et il envoie l’attestation de service fait au demandeur.

* Chef du parc :

Il reçoit une demande de véhicule, il l’exécute et il envoie l’attestation de service fait au demandeur, il peut consulter la liste des véhicules disponibles, ajouter des chauffeurs et plusieurs d’autres fonctionnalités.

* Agent de DAM :

Il traite les demandes client (voir les demandes de fournitures), ce type de demande requise l’intervention de magasin et son gérant qui fournit le service au demandeur.

* Responsable de service relex-PEC :

Il traite les demandes d’activité RELEX et la demande de prise en charge.

#### Analyse Fonctionnelle

Pour l’analyse fonctionnelle, nous allons utiliser les deux diagrammes UML suivants :

* Le diagramme de cas d’utilisation.
* Le diagramme de classe.

##### Le diagramme de cas d’utilisation

Le diagramme des cas d'utilisation (Use Case Diagram) constitue la première étape de l’analyse UML en :

- Modélisant les besoins des utilisateurs.

- Identifiant les grandes fonctionnalités et les limites du système.

- Représentant les interactions entre le système et ses utilisateurs.

Le diagramme des cas d’utilisation apporte une vision utilisateur et absolument pas une vision informatique. Il ne nécessite aucune connaissance informatique et l’idéal serait qu’il soit réalisé par le client. Le diagramme des cas d’utilisation n’est pas un inventaire exhaustif de toutes les fonctions du système. Il ne liste que des fonctions générales essentielles et principales sans rentrer dans les détails [10].

###### Diagramme de cas d’utilisation de notre système :

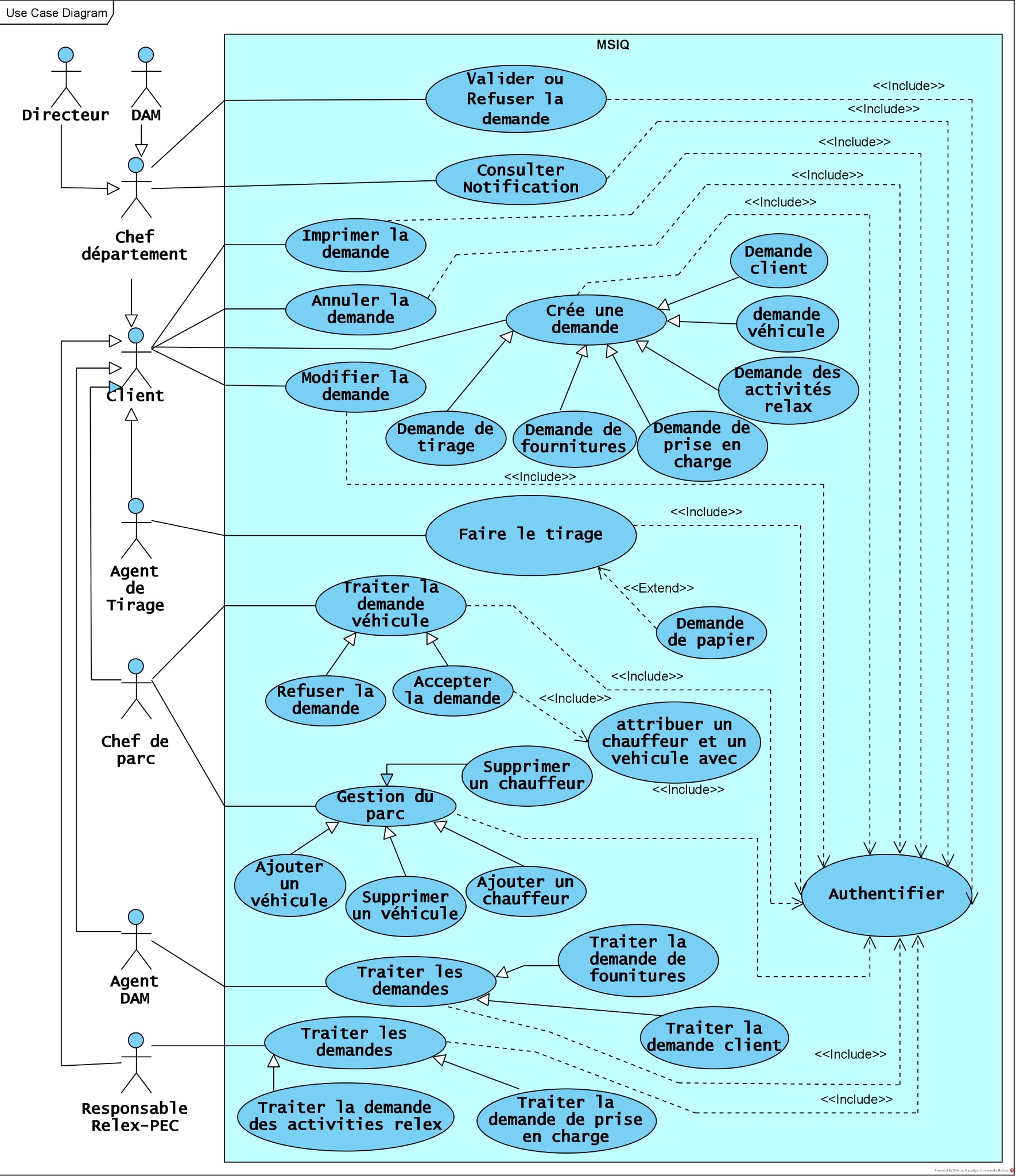


Figure 22 Diagramme de cas d'utilisation général

La figure 2.1 représente le diagramme de cas d'utilisation général du système, c'est-à-dire toutes les fonctionnalités que doit fournir ce dernier aux acteurs.

###### Description textuelle des cas d’utilisation essentiels :

* A- Authentification

|  |  |
| --- | --- |
| **Titre** | Authentification |
| **Acteur** | Client, chef département, directeur, directeur DAM, chef département DAM, agent DAM, chef du parc, Agent de tirage |
| **But** | Accéder à la première page du système |
| **Précondition** | Avoir un compte |
| **Scénario nominal** | 1. L'utilisateur accède à la page d'authentification.  2. Le système affiche le formulaire d'authentification.  3. L'utilisateur saisie son nom et son mot de passe.  4. Le navigateur envoie les données vers le serveur web.  5. Le serveur web envoie la requête vers la base de données.  6. La base de données cherche si le nom et le mot de passe existent et revoie le résultat vers le serveur web  7. Le serveur web ouvre une nouvelle session et envoie la page approprié vers le navigateur.  8. Le navigateur affiche la page appropriée. |
| **Scénario alternatif** | 7.1 - Si le nom et le mot de passe ne sont pas vérifiés :  -Le serveur n'ouvre pas une session et envois la page d'authentification vers le navigateur.  -Le navigateur affiche la page d'authentification avec un message d'erreur. |
| **Post-Condition** | L'utilisateur accède à la page appropriée |

Tableau 2 Description textuelle de cas d'utilisation « authentification

Cette table "Tableau 2" représente la description textuelle qui décrit toutes les étapes permettant à

Un utilisateur de s'authentifier pour qu'il puisse accède à la première page du système.

* B- Créer une demande client

|  |  |
| --- | --- |
| **Titre** | Créer une demande client. |
| **Acteur** | Client |
| **But** | Un service ou un produit. |
| **Précondition** | Le client s’authentifier. |
| **Scénario nominal** | 1. Le client s’authentifie.  2. Le client entre dans l’espace demande et clique sur demande client.  3. Le client remplis le formulaire avec les informations qui lui sont demandés.  4. Le client soumet la demande qui va être transférée à son supérieur afin de la valider. |
| **Scénario alternatif** | 1.1Le nom d’utilisateur ou mot de passe erroné et donc échec d’authentification.  **-** message d’erreur, demande de réauthentification.  3.1 erreur de remplissage d’un champ :  - demander au client de remplir le champ une nouvelle fois |
| **Post-condition** | - demande validée par les supérieurs et service fait. |

Tableau 3 Description textuelle de cas d'utilisation Créer une demande client

Cette table "Tableau 3" représente la description textuelle qui décrit toutes les étapes permettant à un client de faire une demande client et la soumettre.

* C- traitement d’une demande de véhicule

|  |  |
| --- | --- |
| **Titre** | Traitement d’une demande de véhicule |
| **Acteur** | Chef de parc |
| **But** | Satisfaire une demande véhicule d’un client |
| **Précondition** | Le chef de parc s’authentifie |
| **Scénario nominal** | 1. Le chef de parc s’authentifie.  2. Le chef de parc consulte les demandes envoyées.  3. Le chef de parc traite la demande de véhicule selon la disponibilité des chauffeurs et des véhicules.  4. Le chef de parc envoie une attestation de service fait au client. |
| **Scénario alternatif** | 1.1Le nom d’utilisateur ou mot de passe erroné et donc échec d’authentification.  **-** message d’erreur, demande de réauthentification. |
| **Post-condition** | - demande validée par les supérieurs et service fait. |

Tableau 4 Description textuelle de cas d'utilisation Traitement d’une demande de véhicule.

Cette table "Tableau 4"représente la description textuelle qui décrit toutes les étapes permettant à le chef de parc de traiter une demande de véhicule soumise par un client.

/ / / ajouter la description textuelle d’autres cas d’utilisation

##### Le diagramme de classe :

Le diagramme de classe constitue un élément très important de la modélisation : il permet de définir quelles seront les composantes du système final. [ ]

Il représente les classes intervenant dans le système. Une classe décrit les responsabilités, le comportement et le type d’un ensemble d’objets, les éléments de cet ensemble sont les instances de la classe.

* Son utilisation :

Le diagramme de classe est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces d’un système ainsi que les différentes relations entre celle-ci. Ce diagramme fait partie de la partie statique d’UML car il fait abstraction des aspects temporels et dynamiques.

* Identification des classes :

Une classe est une description d'un groupe d'objets partageant un ensemble commun de propriétés (les attributs), de comportements (les opérations) et de relations avec d'autres objets (les associations et les agrégations). [08]

* Une classe contient : [09]
* **Des attributs :**(ou champs): L’attribut d'une classe est une caractéristique de ses objets (instances).
* **Des méthodes :**(ou opérations de la classe) : Les méthodes décrivent les opérations qui sont applicables aux instances de la classe. C’est un service dont un objet peut demander l’exécution.

###### Dictionnaire des classes de notre système :

Les attributs et les méthodes de chaque classe sont définis comme suit :

|  |  |  |
| --- | --- | --- |
| **Classes** | **Attributs** | **Méthodes** |
| Utilisateurs | email, mobile, fonction,  userPassword,  nomUtilisateur,  typeUtilisateur, structure,  prenomUtilisateur,  posteTelephonique,  dateNaissance,  departement | authentifier(), register(),  modifierProfile(),  ModifierMotDePasse() |
| Demande | demande\_ID, etat, motif,  demande\_Date,  utilisateurs\_ID, seen | getDemands(),  getDemand(), getEtat(),  setEtat(),  update() ,  delete(),  print(),  valider() |
| DemandeClient | demande\_C\_ID,  demande\_C\_description,  nature, objet,  mise\_disposition,  date\_mise\_dispostion,  achat, nAchat, date\_achat,  oAchats | getDemandClient(),  setDemandClient(),  updateDemandClient() |
| DemandeTirage | demande\_T\_ID,  numero\_ordre,  date\_prestation,  document\_ID | getDemandTirage(),  setDemandTirage(),  updateDemandTirage() |
| DemandeFouniture | demande\_F\_ID,  date\_recu | getDemandFouniture(),  setDemandFouniture(),  updateDemandFouniture() |
| DemandePriseEnCharge | demande\_P\_ID, aeroport,  collegue1\_ID,  collegue2\_ID,  collegue3\_ID,  collegue4\_ID,  collegue5\_ID, destination,  objet\_mission, startDate,  EndDate,  moyen\_transport,  heureDeVol | getDPEC(),  setDPEC(),  updateDPEC() |
| DemandeRelex | demande\_R\_ID,  destination, date\_depart,  objet\_mission,  date\_retour,  prise\_en\_charge,  demande\_V\_ID | getDemandeRelex(),  setDemandeRelex(),  updateDemandeRelex() |
| Objet | code\_object, designation,  quantite | getObjects(), setObject() |
| DemandeFournitureObject | demande\_F\_ID,  code\_object, qty\_demande  int, qty\_accordee,  observation | setDFO() |
| Document | document\_ID, autre,  nom\_document,  nombre\_feuille,  nombre\_exemplaire,  total\_feuille,  type\_document,  stored\_name,  format\_fichier | - |
| Vehicule | matricule, nom, annee,  type\_vehicule | getVehicules(),  setVehicule() |
| Chauffeur | Chauffeur\_ID,nom,Prenom,type\_permis  Telephone,email | getChauffeurs(),  setChauffeur() |

Tableau 5 Liste des classes avec leurs attributs et méthodes

* *Diagramme de classes de notre système :*

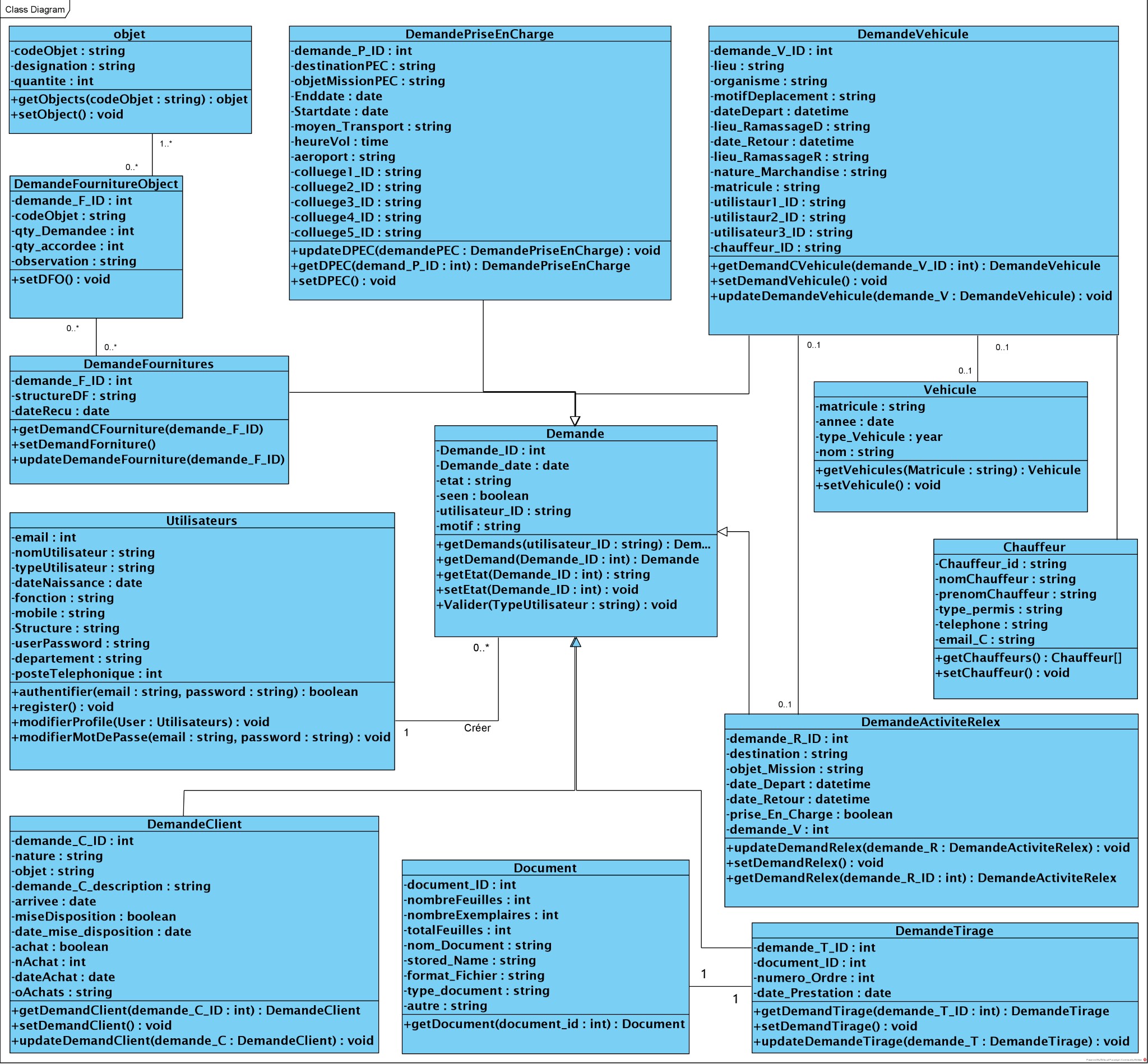


Figure 23 Diagramme de classes

## Conclusion

Dans ce chapitre, nous avons fait la spécification des besoins et l’analyse fonctionnelle avec le diagramme de cas d’utilisation et le diagramme de classes correspondants à notre thème qui concerne l’étude et la réalisation d’un système de traitement des demandes client, et dans le prochain nous allons entamer un niveau de conception plus profond.

# Chapitre 3 : Conception

## 

## Introduction

Dans le référentiel d’ITIL, la phase de conception est une phase très importante qui sert a mener a une bonne transition du service avant de l’exploiter. Nous avons commencé cette étape dans le chapitre précédent, par l’introduction de l’analyse fonctionnelle.

Dans ce chapitre nous allons entamer la conception à un niveau très élevé. En ayant comme objectif une compréhension approfondie, pour cela nous allons utiliser le diagramme de séquence.

## Diagramme de séquence

Un diagramme de séquence représente une interaction entre objets en insistant sur la chronologie des envois des messages. La notation est dérivée des Object Message Sequence [12] de siemens pattern group. Un objet est matérialisé par un rectangle et une barre verticale appelée ligne de vie des objets.

Nom : Classe

Figure 24 Représentation graphique d’un objet

Le diagramme de séquence d’objets est utilisé pour modéliser des flux de contrôle ordonnés selon le temps. Il ne décrit toutefois pas le contexte des objets. Les principales informations contenues dans un diagramme de séquence d’objets sont les messages échangés entre les lignes de vie, présentés dans un ordre chronologique. Ainsi, contrairement au diagramme de communication, le temps y est représenté explicitement par une dimension (la dimension verticale) et s’écoule de haut en bas. [13]

### Les diagrammes de séquence de notre système

#### Demande véhicule

En respectant et suivant le flux de la demande, le client interagit directement avec le système.

Le système va par la suite transférer cette demande aux supérieurs qui vont décider l’accord ou désaccord de la demande.

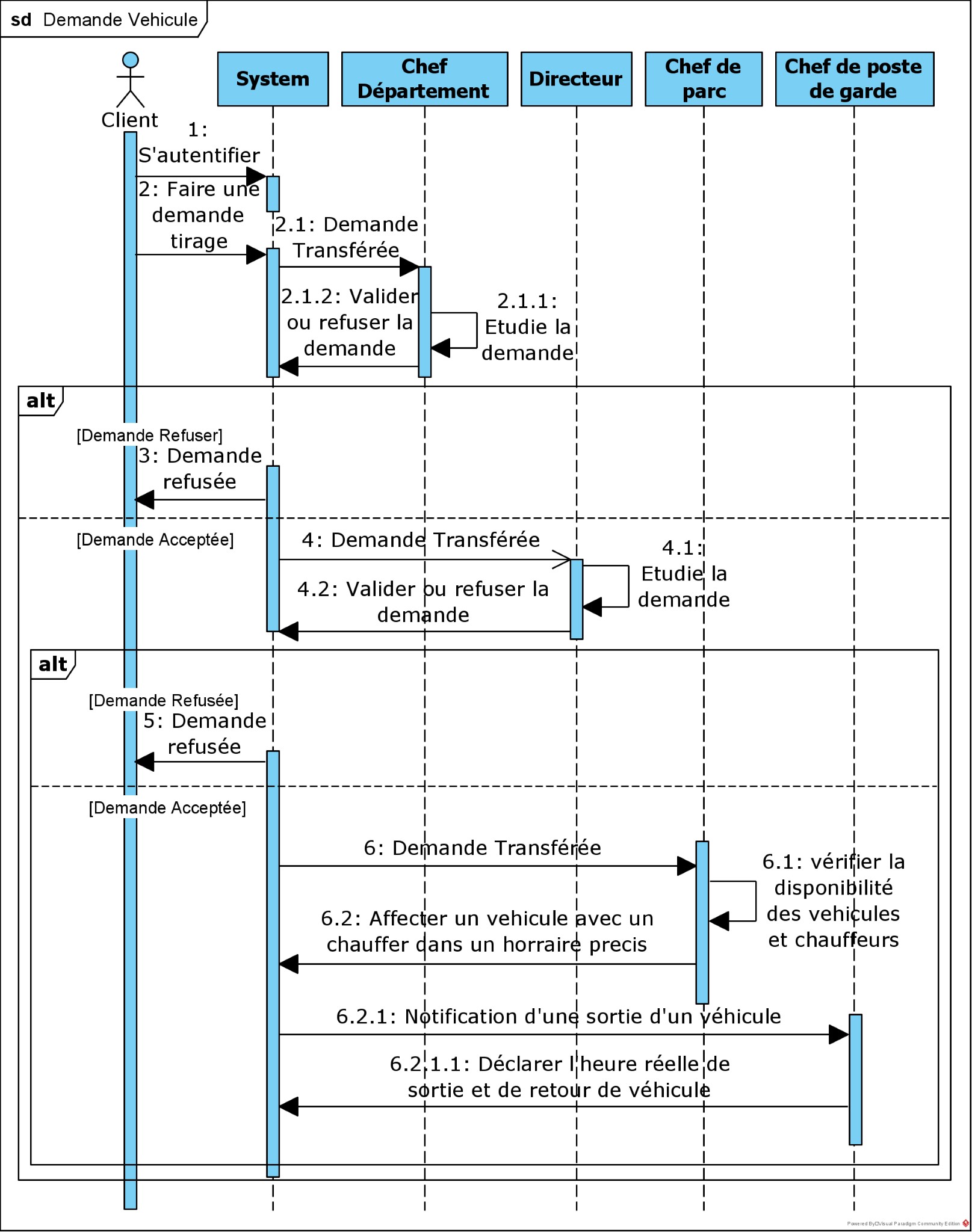


Figure 25 Diagramme de séquence d’une demande véhicule

#### Demande client

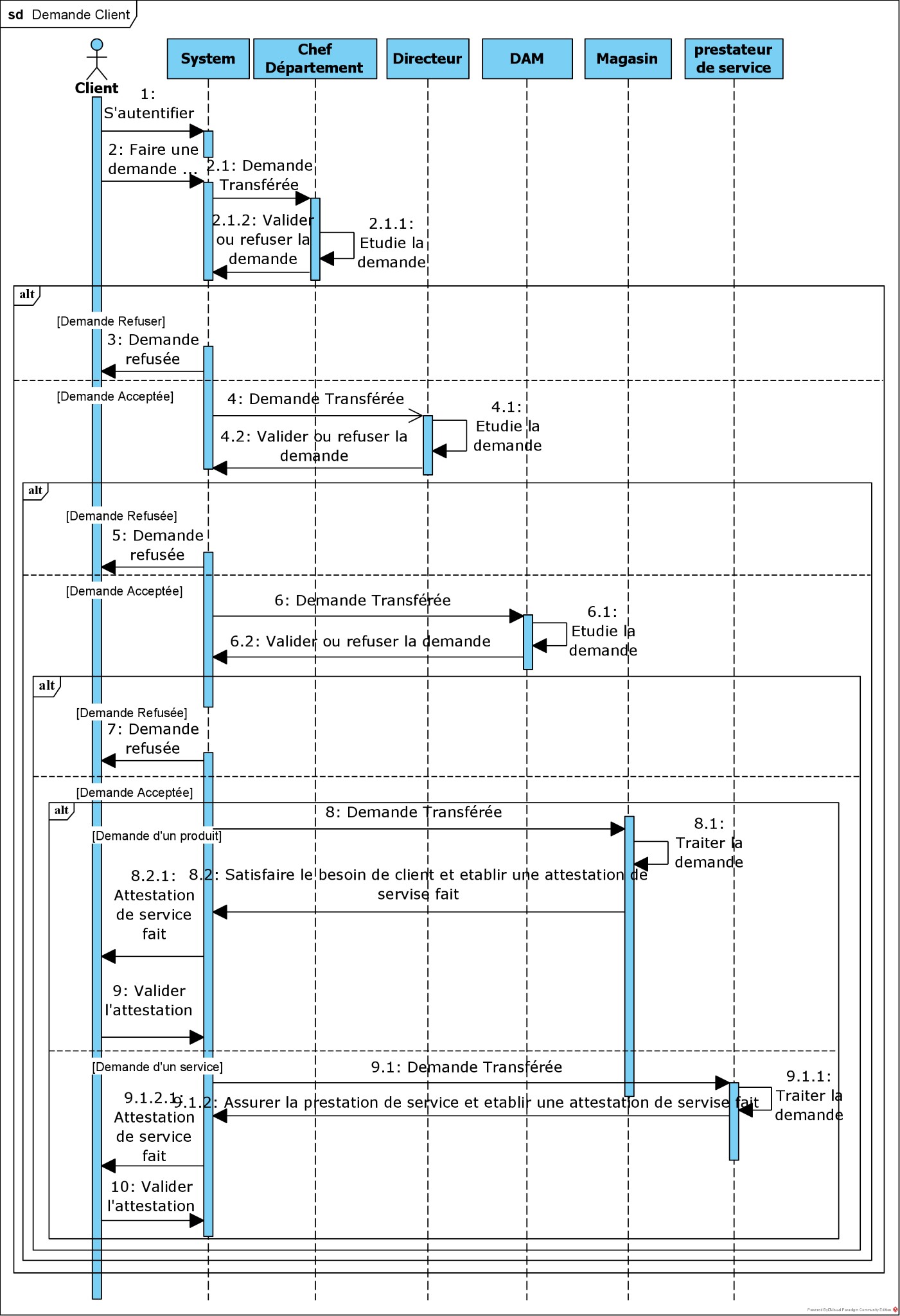


Figure 26 Diagramme de séquence d’une demande client

#### Demande de fourniture

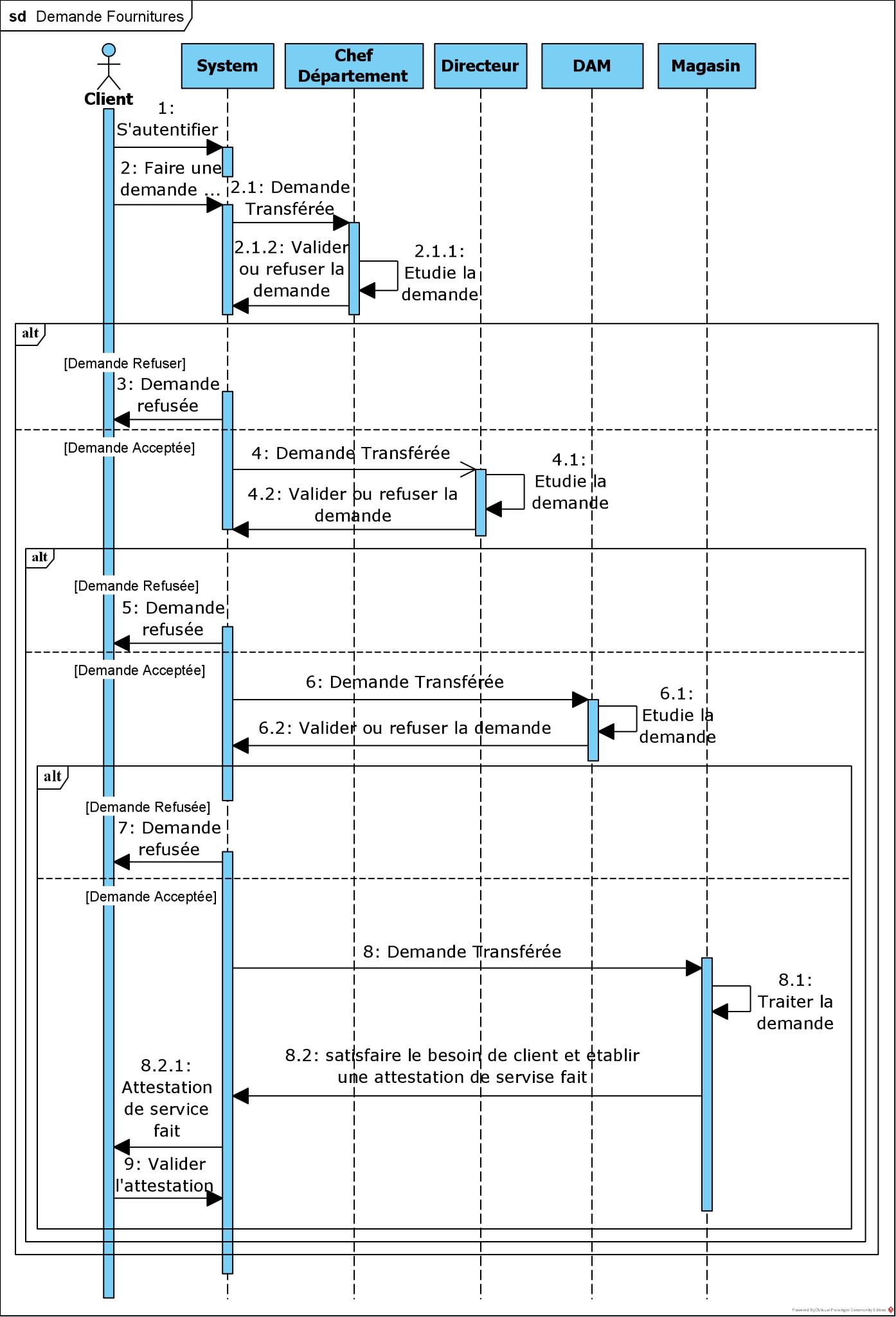


Figure 27 Diagramme de séquence d’une demande Fournitures

#### Demande de tirage

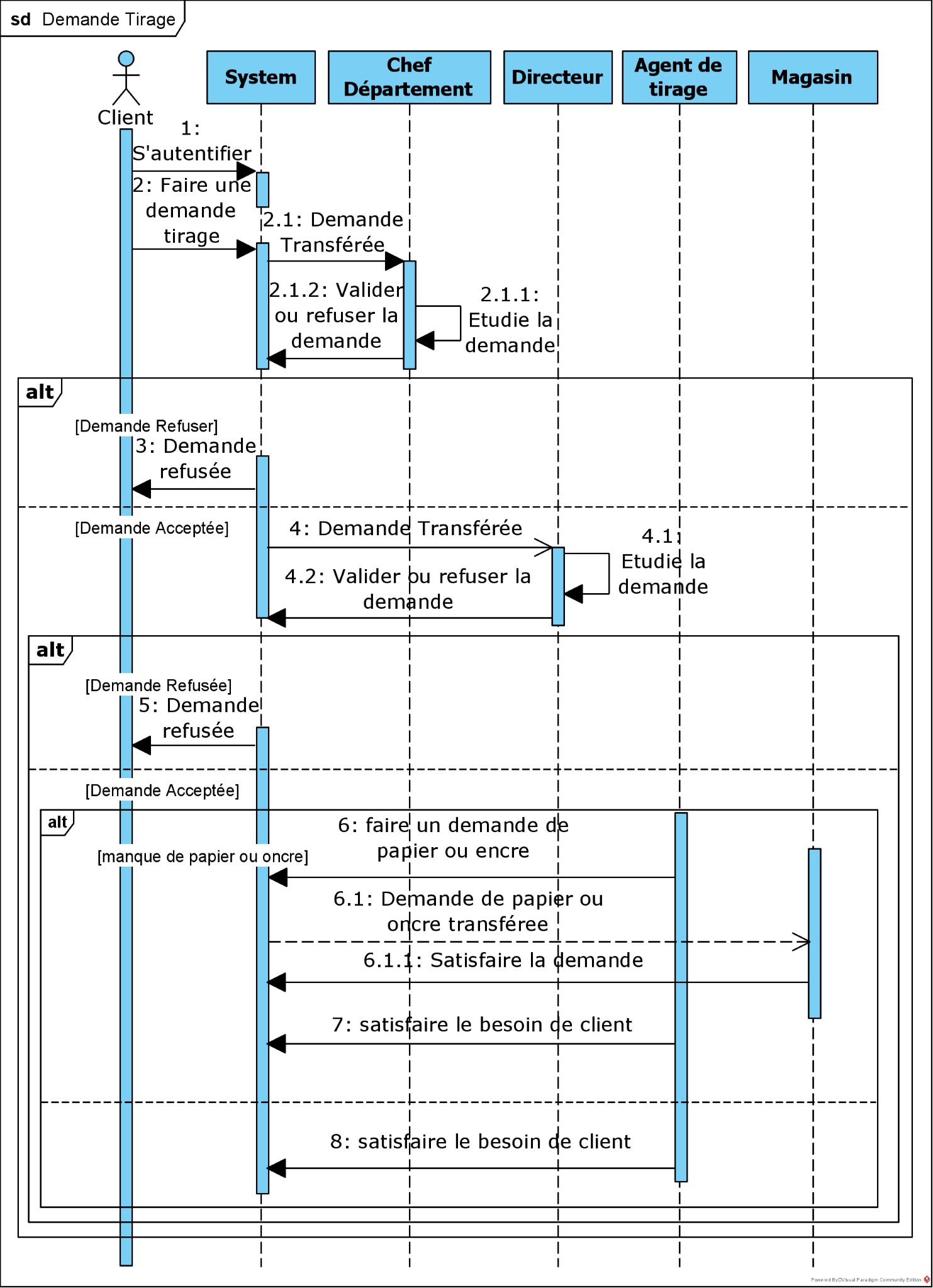


Figure 28 Diagramme de séquence d’une demande Tirage

#### Demande prise en charge

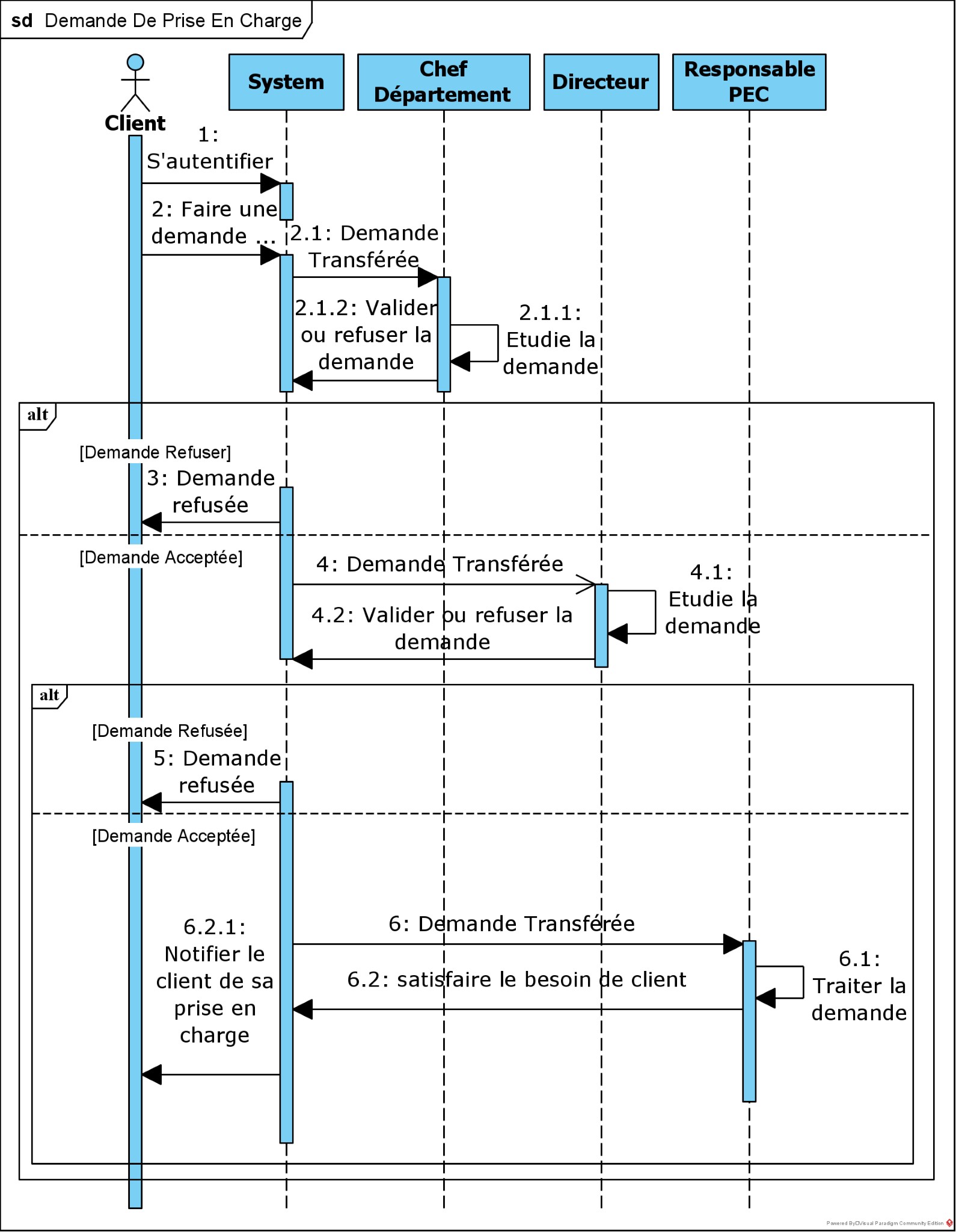


Figure 29 Diagramme de séquence d’une demande De Prise En Charge

#### Demande activité relex

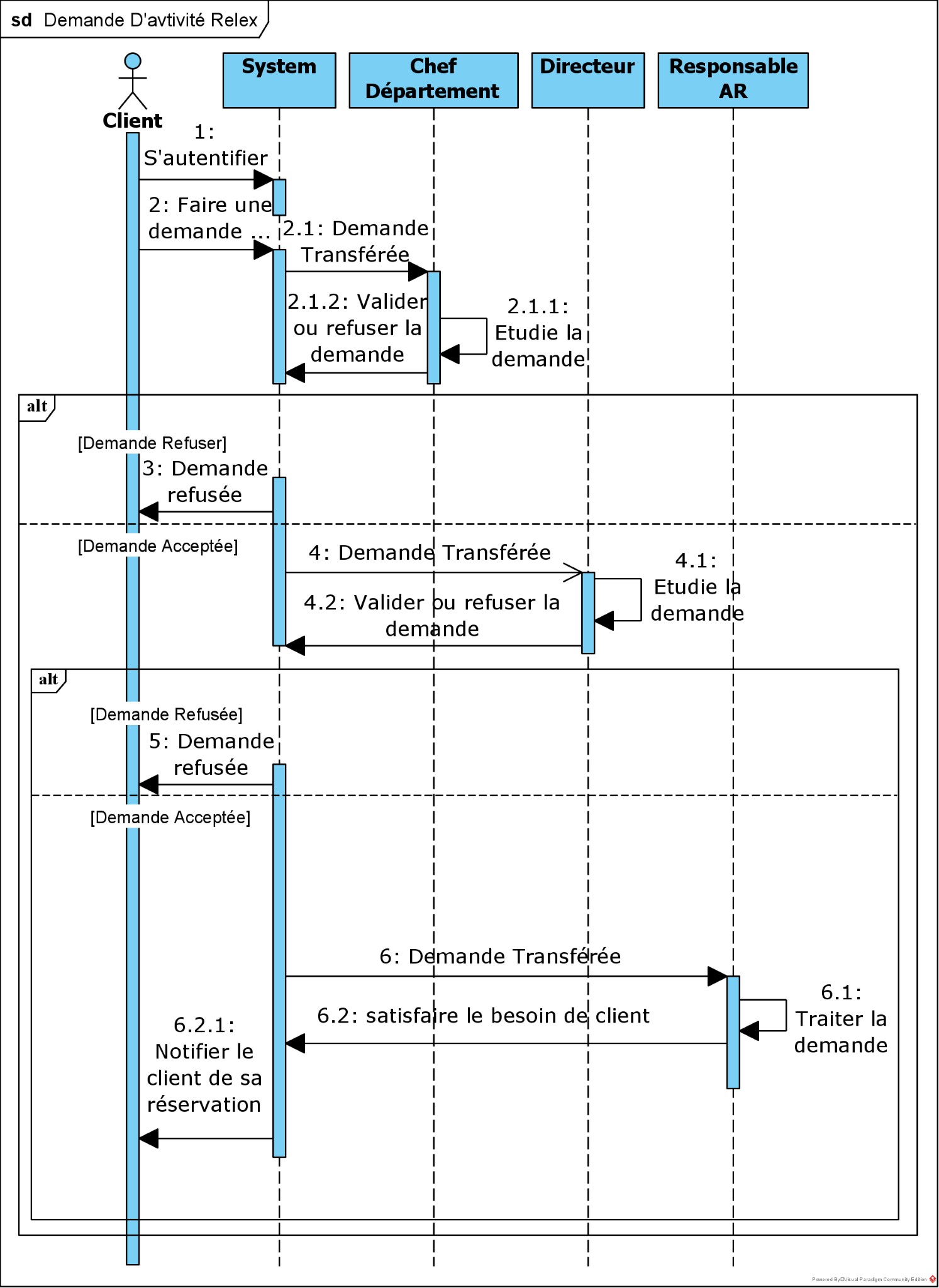


Figure 30 Diagramme de séquence d’une demande Activité relex

## Règles de gestion

Une règle de gestion est une règle suivie par la société. Il peut s'agir d'une disposition légale, d'une exigence formulée par un client ou d'un article de règlement interne. [14]

Les règles de gestion de notre système sont :

Une demande client peut être faite par un seul client, un client peut faire plusieurs demandes.

Une demande de fourniture est soumise par une structure, elle peut contenir plusieurs produits. Une structure peut réaliser plusieurs demandes de fourniture.

Une demande de véhicule peut être faite par un ou plusieurs clients appartenant à la même structure. Un véhicule et un chauffeur pouvons être attribuer à une demande véhicule.

Une demande de tirage est faite par une structure. Une structure peut faire plusieurs demandes tirage et elle contient un ou plusieurs documents.

Une demande de prise en charge peut être faite par un ou plusieurs clients appartenant à la même structure.

Une demande d’activités relex est faite par un seul client.

Une demande peut passer de plusieurs états dépend la responsable de validation. L’état peut être Directeur DAM, Agent de Tirage, Agent de magasin, Chef de parc, Acceptée, Rejetée et Chef Département.

Chaque utilisateur a un type, le type peut être Client, Directeur, Responsable DAM, Chef département, Chef de parc, Agent de magasin et Agent de Tirage

Un client doit remplir toutes les cases dans le formulaire convenablement sauf les cases associées aux responsables pour que la demande soit envoyée.

Un simple client (employé) n’a pas le droit d’accepter des demandes, la validation ou refus des demandes sont restreints aux responsables des départements et directeurs.

Un client possède un nom d’utilisateur et un mot de passe forte au moins de huit chaine caractères.

## Dictionnaire de données

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Codification** | **Signification** | **Type** | **Taille** | **Règle** |
| Email | Email et id du client | Alphanumérique | 50 | Unique |
| nomUtilisaeur | Nom du client | alphanumérique | 50 | - |
| prenomUtilisateur | Prénom du client | alphanumérique | 50 | - |
| dateNaissance | Date de naissance du client | date | - | AAAAMMJJ |
| Mobile | Numero de telephone personnel du client | numérique | - | - |
| posteTelephonique | Numéro de Tel de poste du client | numérique | - | - |
| Fonction | La fonction du client | alphanumérique | 50 | - |
| userPassword | Le mot de passe du client | alphanumérique | 50 | - |
| Structure | La structure où le client travaille | alphanumérique | 50 | - |
| typeUtilisateur | Le type d’utilisateur (client, directeur,…) | alphanumérique | 50 | **-** |
| departement | Le département où le client travaille | alphanumérique | 50 | - |
| Demande\_ID | L’id de la demande | numérique |  | UNIQUE |
| Demande\_Date | La date de la demande | Datetime | - | AAAAMMJJ 00 :00 :00 |
| Utilisateur\_ID | L’id de l’utilisateur qui a crée la demande | alphanumérique | 50 | - |
| Etat | L’état de la demande (acceptée, rejetée.) | alphanumérique | 50 | - |
| Motif | Le motif de refus | alphanumérique | max | - |
| Seen | Si la demande a été vu ou pas | Booléen |  |  |
| Demande\_C\_ID | L’id de la demande Client | numérique | - | UNIQUE |
| Demande\_C\_des | La description de la demande client | alphanumérique | Max | - |
| Nature | La nature de la demande client (produit, prestation du service) | alphanumérique | 50 | - |
| objet | L’objet de la demande client | alphanumérique | 50 | - |
| mise\_dispo | La mise à la disposition | booléen | - | - |
| date\_mise\_dispo | La date de la mise a disposition | Date |  | AAAAMMJJ |
| achat | - | booleen |  | - |
| nAchat | Numéro d’achat | numérique |  | - |
| date\_achat | Date d’achat | date |  | AAAAMMJJ |
| oAchats | Orientation d’achat | alphanumérique | Max | - |
| demande\_V\_ID | L’id de la demande véhicule | numérique |  | UNIQUE |
| Lieu | Lieu de déplacement | alphanumérique | 50 | - |
| organisme | Le nom de l’organisme d’accueil | alphanumérique | 50 | - |
| motif\_deplacement | Le motif du déplacement | alphanumérique | 50 | - |
| date\_depart | La date du départ | datetime | - | AAAAMMJJ hhmmss 00 :00 :00 |
| Lieu\_ramassage\_d | Le lieu de ramassage du départ | alphanumérique | 50 | - |
| date\_retour | La date de retour | datetime | - | AAAAMMJJ hhmmss 00 :00 :00 |
| Lieu\_ramassage\_r | Le lieu de ramassage de retour | alphanumérique | 50 | - |
| Nature\_marchandise | La nature de marchandise | alphanumérique | 50 | - |
| Utilisateur1\_ID | L’id du collègue num 1 | Numérique | 50 | - |
| Utilisateur2\_ID | L’id du collègue num 2 | Numérique | 50 | - |
| Utilisateur3\_ID | L’id du collègue num 3 | Numérique | 50 | - |
| matricule | Le matricule de véhicule | numérique | - | - |
| Chauffeur\_ID | L’id du chauffeur de véhicule. | numérique | - | - |
| demande\_T\_ID | L’id de la demande de tirage | numérique |  | UNIQUE |
| numero\_ordre | Numero d’ordre du bon | numérique |  | - |
| date\_prestation | La date de prestation | date |  | AAAAMMJJ |
| Document\_ID | L’id du document concerné par la demande de Tirage | numérique |  | - |
| nom\_document | Le nom de document | alphanumérique | 50 | - |
| nombre\_feuille | Le nombre de feuilles de document | numérique | - | - |
| nombre\_exemplaire | Le nombre d’exemplaires pour le tirage | numérique | - | - |
| total\_feuilles | Le nombre total des feuilles | numérique | - | - |
| Type\_document | Le type de document (photocopie, relieur..) | alphanumérique | 50 | - |
| autre | La description d’un autre type | alphanumérique | 50 | - |
| stored\_name | Le nom de stockage du fichier dans le serveur | alphanumérique | 50 | - |
| demande\_F\_ID | L’id de la demande de fourniture | numérique |  | UNIQUE |
| date\_recu | La date de reception de la demande | date | - | AAAAMMJJ |
| demande\_P\_ID | L’id de la demande de prise en charge | numérique |  | UNIQUE |
| aéroport | Le nom de l’aéroport | alphanumérique | 50 | - |
| colluegue1\_ID | L’id du collègue num 1 | numérique | - | - |
| colluegue2\_ID | L’id du collègue num 2 | numérique | - | - |
| colluegue3\_ID | L’id du collègue num 3 | numérique | - | - |
| colluegue4\_ID | L’id du collègue num 4 | numérique | - | - |
| colluegue4\_ID | L’id du collègue num 4 | numérique | - | - |
| quantiteAccordee | quantité de produit accordée | numérique |  |  |
| dest\_Pec | La destination de demandeur de prise en charge | alphanumérique | - | - |
| objet\_m\_pec | L’objet de la mission pour la prise en charge | alphanumérique | max | - |
| startDate | La date de commencement de la mission | date | - | AAAAMMJJ |
| EndDate | La date du fin de la mission | date | - | AAAAMMJJ |
| moyen\_transport | Le moyen de transport | alphanumérique | 50 | - |
| heureDeVol | L’heure de vol | time | - | 00 :00 :00 |
| demande\_R\_ID | L’id de la demande relex | numérique | - | UNIQUE |
| dest\_AR | La destination du demandeur RELEX | alphanumérique | 50 | - |
| date\_D\_AR | La date du départ | date | - | AAAAMMJJ |
| objet\_m\_AR | L’objet de la mission d’activité relex | Alphanumérique | max | - |
| Date\_retour | La date de retour | date | - | AAAAMMJJ |
| Prise\_en\_charge | La prise en charge liée avec relex | booleen | - | - |
| demande\_V | L’id de la demande véhicule liée à la demande relex | numérique | - | - |
| code\_objet | Le code de l’objet | alphanumérique | 6 | Unique |
| designation | La designation de l’objet | alphanumérique | 50 | - |
| quantite | La quantité de l’objet | numérique | - | - |
| demande\_FO\_ID | L’id de la demande fourniture | numérique | - | - |
| code\_obj | Code d’objet demandé | alphanumérique | 6 | - |
| qty\_demandee | Quantité demandée | numérique | - | - |
| qty\_accordee | Quantité accordée | numérique | - | - |
| observation | observations | alphanumérique | max | - |
| matricule | Matricule de véhicule | alphanumérique | 50 | - |
| mark | Marque de véhicule | alphanumérique | 50 | - |
| annee | L’annee de sortie de vehicule | Year | - | AAAA |
| type\_vehicule | Le type de véhicule | alphanumérique | 50 | - |
| chauffeur\_ID | L’id du chauffeur | numérique | - | UNIQUE |
| nom | Le nom du chauffeur | alphanumérqie | 50 | - |
| prenom | Prenom de chauffeur | alphanumérique | 50 | - |
| type\_permis | Le type de permis de chauffeur | alphanumérique | 40 | - |
| telephone | Le numero de telephone de chauffeur | numérique | - | - |
| email\_C | L’email de chauffeur | alphanumérique | 50 | - |

Tableau 6 Dictionnaire de données

## Modèle relationnel de données

Généralement, on ne génère pas le code du LDD SQL directement à partir d’un diagramme de

classes. On préfère transformer le diagramme de classes en un schéma représentant les données au travers de leurs éléments de modélisation relationnelle. Le code du LDD SQL est alors déduit du schéma relationnel. Un des intérêts est de pouvoir représenter la structure relationnelle des données sous une forme graphique donnant une vision générale des données, dégagée de toute contrainte syntaxique du langage du LDD SQL. [15]

### Du diagramme de classes au modèle relationnel de données :

#### Principe :

Intuitivement, tel que le suggère la figure suivante, à chaque classe on peut faire correspondre une relation telle que :

* Le nom de la relation reprend le nom de la classe,
* Le nom des attributs de la relation sont issus des noms des attributs de la classe.

—

Le

nom

des

attributs

de

la

relation

sont

issus

des

noms

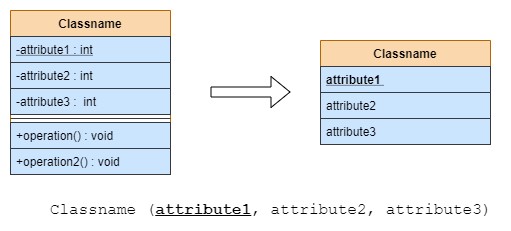
des

attributs

de

la

classe.



Figure

III.8

–

Représentation

du

principe

de

transformation

#### Quelques notions essentielles

* **Entité :** toutes entités sont transformées en table, les propriétés d’entités deviennent attribut de la table, l’identifiant de l’entité devient la clé primaire de la table.
* **Association :** relation entre deux ou plusieurs entités.
* **Domaine** : c'est l'ensemble des valeurs d'un attribut.
* **Relation** : c'est un sous ensemble du produit cartésien d'une liste de domaines. C'est en fait un tableau à deux dimensions dont les colonnes correspondent aux Domaines et dont les lignes contiennent des tuples. On associe un nom à Chaque colonne.
* **Attribut** : Propriété d’une entité ou d’une association.
* **Tuple** : c'est la liste des valeurs d'une ligne d'une relation.
* **Cardinalité** : c’est le nombre minimal et maximal d’associations possibles entre deux entités.
* **L'arité** : est le nombre d'attributs d'une relation.
* **Clé primaire** : ensemble d'attributs dont les valeurs permettent de distinguer les n-uplets les uns des autres (notion d'identifiant).
* **Clé étrangère** : Attribut qui est clé primaire d'une autre entité.

#### Transformation des associations :

Il existe trois types d'associations :

1. « Association 1.\* » : se traduit par la création d’une clé étrangère dans la relation correspondante à l’entité coté (1).

Cette clé étrangère est la clé primaire de la relation correspondant à l’autre entité (n).

1. « Association \*.\*» : Se traduit par la création d’une relation dont La clé primaire est composé des clés étrangères référençant les relations correspondant aux entités liées par l’association.
2. « Association 1..1 » : on ajoute un attribut de type clé étrangère dans la relation ayant la multiplicité minimale égale à un, qui porte le nom de la clé primaire de l’autre relation connectée à l'association. Si les deux multiplicités minimales sont à un, il est préférable de fusionner les deux relations en une seule.

### Notre modèle relationnel de données

—utilisateurs(**email,** userPassword, nomUtilisateur, prenomUtilisateur, typeUtilisateur, dateNaissance, mobile, fonction, structure, posteTelephonique, departement)

— demande(**demande\_ID**, demande\_Date, *utilisateurs\_ID*, etat, motif, seen)

—demande\_client(***demande\_C\_ID***, demande\_C\_description, nature, objet, mise\_disposition, date\_mise\_dispostion, achat, nAchat, date\_achat, oAchats)

— demande\_vehicule (***demande\_V\_ID***, lieu, organisme, motif\_deplacement, date\_depart, lieu\_ramassage\_d, date\_retour, lieu\_ramassage\_r, nature\_marchandise, utilisateur1\_ID, *utilisateur2\_ID*, *utilisateur3\_ID*, *matricule*, *chauffeur\_ID*)

— demande\_tirage(***demande\_T\_ID***, numero\_ordre, date\_prestation,

*document\_ID*)

— document(**document\_ID**, nom\_document, nombre\_feuille, nombre\_exemplaire, total\_feuille, type\_document, autre, stored\_name, format\_fichier)

— demande\_fourniture(***demande\_F\_ID***, date\_recu)

— objet **(code\_object**, designation, quantite)

— demande\_fourniture\_object(***demande\_F\_ID***, ***code\_object***, qty\_demande, qty\_accordee, observation)

—demande\_priseEnCharge(***demande\_P\_ID***, *collegue1\_ID*, *collegue2\_ID*, *collegue3\_ID*, *collegue4\_ID*, *collegue5\_ID*, destination, objet\_mission, startDate, EndDate, moyen\_transport, aeroport, heureDeVol)

— demande\_relex(***demande\_R\_ID***, destination, objet\_mission, date\_depart, date\_retour, prise\_en\_charge, *demande\_V\_ID*)

— vehicule(**matricule**, nom, annee, type\_vehicule)

— chauffeur(**chauffeur\_id**, nom, prenom, type\_permis, telephone, email)

Les attributs en **gras** sont des clés primaires.

Les attributs en *italique* dont des clés étrangères.

Les attributs en ***gras\_italique*** dont des clés étrangères et des des clés primaires.

### Conclusion

Dans ce chapitre, nous avons effectué et détaillé la conception de notre système de traitement de demandes client en s’adaptant en premier lieu sur le langage de modélisation et en se basant sur les diagrammes du formalisme UML, en commencent par le diagramme de séquence. En deuxième lieu nous avons présenté les règles de passage du modèle conceptuel au modèle relationnel de données.

## Chapitre 4 : Implémentation

### Introduction

Ce chapitre présente l’implémentation de notre Système de traitement de demandes client MSIQ. Tout d’abord, nous définirons l’environnement technologique de développement, ainsi que l’architecture du système implémenté, par la suite nous allons élaborer le diagramme de déploiement correspondant, et enfin nous présenterons les principales interfaces graphiques.

### Environnement de développement

* **Node JS** : Node.js est un environnement d'exécution à source ouverte pour l'exécution de code JavaScript côté serveur. Le principal avantage de Node.js est qu'il permet d'effectuer une programmation asynchrone très efficace en mémoire au moyen de websockets (conversation bidirectionnelle en temps réel entre le client et le serveur). Par rapport à l'exécution traditionnelle du code côté serveur, où chaque requête doit être créée dans un fil d'exécution distinct, Node.js traite des milliers de requêtes en utilisant un seul fil d'exécution. Apparemment, il permet d'économiser de la mémoire vive, mais comporte certains pièges, que vous apprendrez également à connaître et à gérer dans la liste des meilleurs livres sur Node.js.[19]
* **Vue JS** : Vue (prononcé /vjuː/, comme view) est un Framework progressif pour la construction d'interfaces utilisateur. Contrairement à d'autres Frameworks monolithiques, Vue est conçu dès le départ pour être progressivement adoptable. La bibliothèque centrale se concentre uniquement sur la couche Vue, et est facile à reprendre et à intégrer avec d'autres bibliothèques ou projets existants. D'autre part, Vue est également parfaitement capable de gérer des applications sophistiquées de page unique lorsqu'il est utilisé en combinaison avec des outils modernes et des bibliothèques de support. [20]
* **Plugin Vuetify :** Vuetify est la bibliothèque de composants #1 pour Vue.js et est en développement actif depuis 2016. Le but du projet est de fournir aux utilisateurs tout ce qui est nécessaire pour construire des applications web riches et engageantes en utilisant Material Design specification. Il accomplit cela avec un cycle de mise à jour cohérent, Support à long terme **(LTS)** pour les versions précédentes, l'engagement communautaire réactif, un vaste écosystème de ressources et un dévouement à des composantes de qualité. [21]
* **MySQL :** MySQL est un système de gestion de base de données relationnelle (SGBDR) à code source libre qui utilise le langage SQL (Structured Query Language). SQL est le langage le plus populaire pour ajouter, accéder et gérer le contenu d'une base de données. [22]

### Architecture du système

Pour développer le système, l’architecture a été proposée comme montré sur la figure

Figure 1 - architecteur générale

#### Coté Client

##### SPA : Single Page Application

SPA signifie Single Page Application, ou application monopage en français. C’est une application web accessible via une page web unique. Le but est d’éviter le chargement d’une nouvelle page à chaque action demandée, et de fluidifier ainsi l’expérience utilisateur.[24]

##### Différences entre une SPA et un site web classique

La différence entre une SPA et un site web classique réside dans leur structure et dans la relation entre le navigateur et le serveur.

|  |  |
| --- | --- |
| **Single Page Application** | **Web classique** |
| Composée d’une seul page | Composé de plusieurs pages |
| Le rôle du navigateur est beaucoup plus important que dans un site web, toute la logique applicative y étant déportée | Le rôle du navigateur est simplement d’afficher les pages envoyées par le serveur et de lui transmettre les actions de l’utilisateur |
| Le serveur est responsable de fournir les ressources de l’application et surtout d’exposer les données | Le serveur contient la logique applicative (il fournit les pages à afficher et réagit aux actions de l’utilisateur) |

Table 1-Comparison entre SPA et web classique

##### Single Page Application Framework (Vue JS)

Vue.js est une bibliothèque pour créer des interfaces Web interactives.

Techniquement, Vue.js se concentre sur la couche ViewModel du modèle MVVM a été montré sur la figure. Il connecte la vue et le modèle via des liaisons de données bidirectionnelles. [25]

* **Vue**: affiche des éléments visuels et des commandes à l'écran.
* **Vue modèle** : transformer les informations du modèle en valeurs qui peuvent être affichées sur une vue.
* **Modèle** : conserver les données d'application. Ce sont généralement des structures ou des classes simples.

Figure 2-MVVM

#### Coté Serveur

##### Un serveur HTTP avec NodeJS et Express

###### Le protocole http : Le protocole HTTP (Hypertext Transfer Protocol) est un protocole de communication de type requête réponse synchrone. Le protocole est textuel : les messages échangés utilisent des caractères encodés sur 8 bits. Une session HTTP consiste en une série d'émissions alternées entre un client et un serveur.[26]

Figure 3- Protocol HTTP

* **Le serveur web :** Un « serveur web » peut faire référence à des composants logiciels (software) ou à des composants matériels (hardware) ou à des composants logiciels et matériels qui fonctionnent ensemble. [27]
* Au niveau des composants logiciels, un serveur web contient différents fragments qui contrôlent la façon dont les utilisateurs peuvent accéder aux fichiers hébergés. On trouvera au minimum un serveur HTTP. Un serveur HTTP est un logiciel qui comprend les URL et le protocole HTTP (le protocole utilisé par le navigateur pour afficher les pages web).
* Au niveau des composants matériels, un serveur web est un ordinateur qui stocke les fichiers qui composent un site web (par exemple les documents HTML, les images, les feuilles de style CSS, les fichiers JavaScript) et qui les envoie à l'appareil de l'utilisateur qui visite le site. Cet ordinateur est connecté à Internet et est généralement accessible via un nom de domaine tel que mozilla.org.
* **Express JS :** est une librairie qui vous permettra de créer une application Web plus simplement qu'avec l'objet http directement. Elle fournit un ensemble de méthode permettant de traiter les requêtes HTTP et fournit un système de middleware pour étendre ses fonctionnalisées. [28]

##### API REST

#### API : **API, trois lettres qu’on croise souvent sur le web et dont on ne connaît pas forcément le sens. Une API signifie**application programming interface (ou interface de programmation d’application, en français), ça **permet à deux applications de se communiquer entre elles. [29]**

* **API REST :**

REST signifie Representational State Transfer (ou transfert d’état de représentation, en français), Les API REST se basent sur le protocole **HTTP** pour transférer les informations. Les API REST constitue de lignes directrices **architecturales** qui structurent la façon de communiquer les données entre l’application et le reste du monde, ou entre différents composants de l’application. [30]

* **Les lignes directrices constituées par l’API REST**

Il faut garantir au minimum quatre lignes directrices, définies comme suit [31]:

**1- Séparation entre le client et le serveur**

**Un client** peut être une application, un navigateur ou un logiciel. Un **serveur**est un ordinateur distant capable de récupérer des données depuis la base de données, de les manipuler si besoin et de les renvoyer à l’API.

Une séparation entre le client et le serveur permet au client de s’occuper uniquement de la récupération et de l’affichage de l’information et permet au serveur de se concentrer sur le stockage et la manipulation des données. Chacun son rôle.

Les API REST offrent un**moyen de communication standardisé entre le client et les données**. En gros, peu importe comment le serveur est construit ou comment le client est codé, du moment qu’ils structurent tous les deux leur communication selon les lignes directrices architecturales REST, en utilisant le protocole HTTP, ils pourront communiquer entre eux.

**2- Sans état**

La demande du client doit contenir toutes les informations nécessaires pour en répondre.

**3- Sauvegardable**

La réponse doit contenir l’information sur la capacité ou non du client de mettre les données **en cache**, ou de les sauvegarder. Si les données **peuvent être mises en cache**, la réponse doit être accompagnée d’un numéro de version. Ainsi, si votre utilisateur formule deux fois la même requête (c’est-à-dire s’il veut revoir une page) et que les informations n’ont pas changé, alors le serveur n’a pas besoin de rechercher les informations une deuxième fois. À la place, le client peut simplement mettre en cache les données de la première fois, puis charger à nouveau les mêmes données la seconde fois.

**4- Système de couches**

Un client qui se connecte à un composant intermédiaire n’a aucune idée de ce avec quoi ce composant interagit ensuite. Par exemple, si vous faites une requête à l’API Facebook pour récupérer les derniers postes : vous n’avez aucune idée des composants avec lesquels l’API Facebook communique.

* **A- La requête REST :**

Chaque **requête** a une structure spécifique. Elle possède :

* ***URI et End points :***

Le **path (ou chemin)** que vous donnez à votre API lui permet de savoir exactement **où**se trouvent les données que vous voulez récupérer.

Les API REST stockent également les données de façon similaire, et un URI constitue le chemin pour y arriver.

Si une ressource est l’objet qui stocke vos données, pour les récupérer nous allons avoir besoin d’un identifiant de ressource uniforme, ou ***URI*** pour ***U***niform ***R***esource***I***dentifier. L’URI est le moyen d’identifier votre ressource, comme une étiquette.

 L’URI qui listerait tous les demandes pourrait être le suivant :

/demandes

Si nous voulons voir les informations sur un seul personnage, qui porte l’ID 123, notre URI serait le suivant :

/demandes/123

**L’URL de la requête**est l’end point complet que nous utilisons pour notre requête. Il associe le nom de domaine + le path de votre ressource.

http://localhost:3030/demandes

* **Les Verbes HTTP :**

Correspondent à différents types d’action que vous pouvez accomplir avec votre requête. Qui mappage pour create (créer), read (lire), update (mettre à jour) et delete supprimer (CRUD) opérations :

|  |  |  |
| --- | --- | --- |
| **Verbes HTTP** | **CRUD opérations** | **Action** |
| GET | lire | renvoie les données |
| POST | créer | crée un nouvel enregistrement |
| PUT | Mettre à jour | met à jour un enregistrement existant |
| DELETE | supprimer | supprime un enregistrement existant |

Table 2- Les verbes http

* **En-tête *HTTP*** *:*

Contient Des informations supplémentaires sur le message. Par exemple :

* De quel langage s’agit-il ?
* À quelle date l’envoyez-vous ?
* Quel logiciel la personne utilise-t-elle ?
* Quelle est votre clé d’authentification ?
* ***Body***

Pour formuler une requête, il n’est utilisé qu’avec **PUT** (mise à jour) ou **POST** (création). Il contient les données réelles de la ressource que vous essayez de créer ou de mettre à jour. Les données sont envoyées sous format JSON.

* **B- La réponse REST :**

Chaque **réponse** a une structure spécifique. Cette dernière possède :

* ***Code de réponse HTTP***

Le code de réponse HTTP aide le développeur et/ou le client à comprendre le **statut** de la réponse.

 Les règles de base pour les codes de réponse HTTP sont les suivantes :

* 100+ ➡ Information
* 200+ ➡ Succès
* 300+ ➡ Redirection
* 400+ ➡ Erreur client
* 500+ ➡ Erreur serveur
* **En-tête *HTTP*** *:*

Elle contient des informations supplémentaires sur le message.

* ***Body :***

Contient l’information que vous avez demandée et que l’API vous renvoie. Celle-ci est matérialisée au sein du body sous la forme d’un JSON ou en XML.

#### Le JSON (JavaScript Object Notation)

Le JSON est un format qui stocke des informations structurées et est principalement utilisé pour transférer des données entre un serveur et un client.[32]

## *Syntaxe JSON*

Un objet JSON commence et se termine par des accolades **{}**. Il peut contenir deux ou plusieurs paires **clé/valeur**, séparées par une **virgule**. Chaque clé est suivie d’un **deux-points** pour la distinguer de la valeur.

Voici un exemple JSON :

Figure 4-Exemple JSON

{

‘’Nom’’: “Ould bouchiba”,

“Prenom”:”Yakoub”

}

### *Types de valeurs*

* **Chaînes de caractères**

Une chaîne de caractères est une séquence définie de zéro ou plus de caractères Unicode. Elle est entourée de deux doubles guillemets.

* **Nombre**

Un Nombre en JSON doit être un nombre **entier** ou un **point flottant.**

* **Booléen**

**True** (vrai) ou **false** (faux) comme valeur.

* **Null**

C’est pour montrer qu’il n’y a pas d’information.

* **Tableau** (Array)

Un tableau est un ensemble ordonné de valeurs. Il est entouré de **crochets []** et chaque valeur qu’il contient est séparée par une virgule.

Un tableau de valeurs peut contenir des objets JSON, ce qui signifie qu’il utilise le même concept de pair clé/valeur. Par exemple :

Figure 5-Tableau JSON

"etudiants":[

{"prenom":"Tom", "nom":"Jackson"},

{"prenom":"Linda", "nom":"Garner"},

{"prenom":"Adam", "nom":"Cooper"}

]

Figure 7- Tableau JSON

* **Objet**

Un objet contient une clé et une valeur. Il y a deux points après chaque clé et une virgule après chaque valeur, ce qui permet également de distinguer chaque objet. Les deux sont entre guillemets.

L’objet, en tant que valeur, doit suivre la même règle qu’un objet. Comme ça :

Figure 6- Objet JSON

"employes": {"prenom":"Tom", "nom":"Jackson"}

#### Web Socket

Historiquement, la création d’applications Web qui nécessitent une communication bidirectionnelle entre un client et un serveur (p. ex., messagerie instantanée et applications de jeu) a nécessité le client de faire un **poll** c’est-à-dire que le client exécute une requête vers son serveur à intervalle de temps régulier, espérant ainsi être mis au courant d’une mise à jour, Cette approche montre les limitations suivantes :

* Certains évènements peuvent être manqués si l’intervalle entre deux requêtes est trop long
* A l’inverse, il est tout à fait possible de réaliser un appel à vide si aucun évènement n’est disponible
* Un intervalle de temps trop court entre deux requêtes aura tendance à surcharger le serveur

La solution le plus simple est d’utiliser le Web [33]

#### ****Définition de Web Socket****

Le protocole Web Socket est conçu pour remplacer les technologies de communication bidirectionnelles existantes qui utilisent HTTP comme couche de transport pour bénéficier de l’infrastructure existante (proxys, filtrage, authentification).

L’ouverture d’une connexion Web Socket s’effectue avec une requête HTTP qui demande au serveur « de mettre à jour la connexion » en connexion Web Socket.

GET /demo HTTP/1.1

Host: example.com

Connection: Upgrade

Sec-WebSocket-Key2: 12998 5 Y3 1 .P00

Sec-WebSocket-Protocol: sample

Upgrade: WebSocket

Sec-WebSocket-Key1: 4 @1 46546xW%0l 1 5

Origin: http://example.com

Figure 7- La requête web socket

Le serveur, s’il supporte le protocole Websocket, peut ainsi terminer l’ouverture de la connexion et la suite du dialogue entre le client et le serveur s’effectuera avec le protocole Websocket.

HTTP/1.1 101 WebSocket Protocol Handshake

Upgrade: WebSocket

Connection: Upgrade

Sec-WebSocket-Origin: http://example.com

Sec-WebSocket-Location: ws://example.com/demo

Sec-WebSocket-Protocol: sample

Figure 8- La réponse web socket

* **HTTP vs WebSocket :**

Figure 9- http contre web socket

La connexion en WebSocket reste ouverte jusqu’à le client ou serveur terminez-le

#### L’utilisation du WebSocket

Le WebSocket est utilisé dans les applications suivantes :

* Les applications en temps réel.
* Les jeux
* Les applications de chat

Le système que nous avons développé doit traiter les demandes client en temps réel, donc nous utilisons le protocole websocket en bénéficions de la bibliothèque socket.io.

#### La bibliothèque Socket.io

Socket.IO est une bibliothèque qui permet une communication en temps réel, bidirectionnelle et basée sur des événements entre le navigateur et le serveur. Il se compose de :

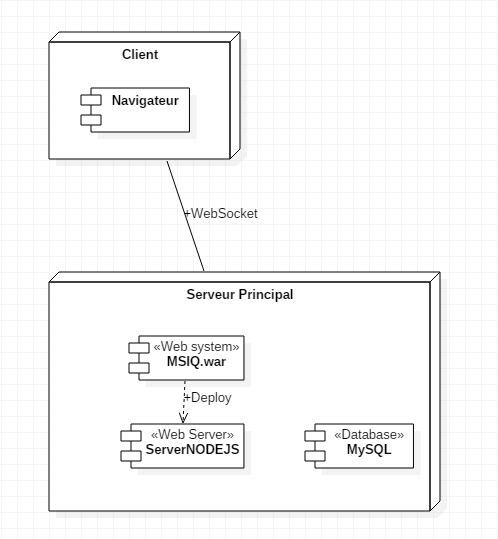
* Un serveur Node.js
* Une bibliothèque cliente Javascript pour le navigateur (qui peut également être exécuté à partir de Node.js) [34]

## Authentification

// ajouter demain nchalh ida heb rabi w kouna fi saha w 3la khiir

## Elaboration du diagramme de déploiement

// A REVOIR



## Présentation des principales interfaces graphiques du système

## Conclusion

Au cours de cette dernière étape de notre travail, nous avons présenté l’implémentation ainsi que les outils utilisés pour développer notre système, son architecture et son diagramme de déploiement.

Au final, nous avons terminé par présenter les interfaces graphiques du système, cette étape s’inclut dans la 3eme phase de processus ITIL qui est la phase transition du service.

**Conclusion & Perspective**

Dans ce mémoire, nous avons abordé le thème « Conception et Réalisation d’un système de traitement des demandes client selon ITIL V.3 ».

Notre étude était basée sur le référentiel ITIL qui avait pour but l’amélioration du service fourni par le système MSIQ qui gère différents types de demandes client faites par les employés de la branche GPL de Naftal situé à el Mohammedia.

Nous avons choisi par la suite le langage UML comme méthode de conception pour modéliser notre système se basant sur une expression des besoins en lien avec les objectifs du système.

Notre but est de fournir à l’utilisateur de système une interface simple à utiliser avec le niveau de sécurité nécessaire avec le respect des droits restreints à certains utilisateurs comme les directeurs contrairement aux autres comme les simples employés.

Vu la contrainte de temps et la situation exceptionnelle, ce travail reste ouvert à plusieurs extensions. Nous envisageons à moyen terme, de compléter l’implémentation des éléments suivants :

* Développer la partie Reporting.
* L’amélioration en ajoutant d’autres options à l’utilisateur comme la prestation du service.

Au terme de ce présent mémoire, nous considérons notre travail comme un pas en avant, et un moyen fort d’apprentissage de nouveaux Framework dans le monde de web.

**Bibliographie**

[01] pierre veyrat ,Definition ITIL pour entreprise et informaticiens, <https://www.heflo.com/>

[02] <https://www.naftal.dz/fr/>

[03] Historique, <https://www.naftal.dz/fr/>

[04] mémoire « le rôle de la communication interne au niveau de l'entreprise NAFTAL » , promotion 2014, Université ABDERRAHMENE MIRA de Bejaia.

[05] <https://www.naftal.dz/fr/>

[06] UML - Unified Modeling Language, Laetitia Matignon, [laetitia.matignon@univ-lyon1.fr](mailto:laetitia.matignon@univ-lyon1.fr)

[07] <http://www.omg.org/uml>

[08] Laurent Piechoki, diagrammes d’UML, édition 2007.

[09] Analyse, Conception Objet, diagrammes de Classes,

Une partie du matériau de ce cours est issue du cours de Stéphane Galland,

Septembre 2003.

[10] <https://explainsimple.com/uml-diagramme-des-cas-dutilisation/>

[11] Laurent Audibert, UML 2 de l’apprentissage à la pratique (cours et exercices), édition Ellipses.

[12] Wiley 1996, Pattern-Oriented software Architecture: A system of patterns.

[13] Claude Belleil, Le langage UML 2.0 Diagramme de séquence d’objets

[14]<http://infocenterarchive.sybase.com/help/index.jsp?topic=/com.sybase.stf.poweramc.docs_12.0.0/html/mcgu/mcgup29.html>

[15] Gregory Claude, UML et les Bases de Données, CNAM,2010

[16] André Corbillé & Vincent Dumas - Business Intelligence et portails (Dunod)

[17] Luc Walrafen - Réaliser des reporting et des budgets pour vous convaincre

[18] Le reporting pour les PME-PMI Des solutions qui travaillent à votre manière - Livre blanc avril 2009- IBM.

[19] <https://medium.com/@LaSoft/top-20-node-js-books-that-you-should-read-in-2018-2329d1dd64bc#:~:text=js-,Node.,conversation%20between%20client%20and%20server).&text=js%20books>.

[20] <https://vuejs.org/v2/guide/>

[21] <https://vuetifyjs.com/fr-FR/introduction/why-vuetify/>

[22] <https://www.siteground.com/tutorials/php-mysql/mysql/>

[23] <https://flaviocopes.com/node-websockets/>

[24] <http://www.opentuto.com/single-page-application/> client spa

[25] <https://012.vuejs.org/guide/> MVVM

[26] [https://wdi.centralesupelec.fr/appliouaibe/Cours/HTTP protocole http](https://wdi.centralesupelec.fr/appliouaibe/Cours/HTTP%20protocole%20http%20page%206)

[27] [https://developer.mozilla.org/fr/docs/Apprendre/Qu\_est-ce\_qu\_un\_serveur\_web /](https://developer.mozilla.org/fr/docs/Apprendre/Qu_est-ce_qu_un_serveur_web%20/) serveur web

[28] <https://www.grafikart.fr/tutoriels/express-798> express js

[29] <https://www.agencedebord.com/api-definition-utilisation/> api

[30] <https://openclassrooms.com/fr/courses/6573181-adoptez-les-api-rest-pour-vos-projets-web/6817216-identifiez-les-avantages-d-une-api-rest> REST api

[31] <https://www.sitepoint.com/developers-rest-api/> REST api

[32] <https://www.hostinger.fr/tutoriels/quest-ce-que-json/> JSON

[33] Reza Bisnis , The WebSocket Protocol , decembre 2011

[34] <https://socket.io/docs/>