## 

Ecole Nationale Des Sciences Appliquées -Kénitra-

## Mémoire de Projet Fin d’Etudes

Pour l’obtention du diplôme d’ingénieur d’état

### *Filière : Génie Informatique*

*Option : Génie logiciel et développement mobile*

## Promotion 2019 – 2020

Effectué à :

## 

|  |  |
| --- | --- |
|  | Conception et réalisation d’un Chatbot assisté par le RPA |
|  |  |
|  |  |

## Z

# Dédicaces

À nos parents

À nos frères

À nos ami

# Remerciements

Remerciement

Au terme de ce travail, nous tenons à exprimer notre profonde gratitude et nos sincères remerciements à notre à notre encadrant à l’Ecole Nationale des Sciences Appliquées de Kénitra **Pr. OUATIK Said**, qui nous a suivi efficacement tout au long de ce projet, pour ses conseils et orientations, pour ses judicieuses directives, son sens d’orientation, sa disponibilité, sa sympathie et ses remarques pertinentes qui nous ont été fort précieuses.

Nos plus vifs remerciements s’adressent aussi à tout le cadre professoral et administratif de l’Ecole Nationale des Sciences Appliquées Kénitra pour leurs efforts en vue d’assurer une formation de haut niveau pour l’ensemble des étudiants.

Nous remercions **Mme. JBILI Nawal** notre tutrice et manager de l’équipe RPA, pour ses conseils et ses directives, et aussi pour le temps qu’elle nous a consacré tout au long de cette période. Nous serons vaniteux si nous nous devons énumérer en ces quelques lignes vos remarquables qualités humaines et professionnelles, veuillez trouver ici l'expression et le témoignage de ma gratitude ressentie.

Nous profitons de cette occasion pour remercier vivement tous les collaborateurs du pôle techno, qui n’ont pas cessé de nous encourager, et pour leurs soutiens.

Nous tenons à remercier tout particulièrement et à témoigner toute notre reconnaissance aux personnes suivantes, pour l’expérience enrichissante et pleine d’intérêt qu’elles nous ont fait vivre durant ces mois au sein de Deloitte Nearshore :

* **M. Olivier MAUDUIT**, associé dédié à l’Audit et Contrôle Interne des systèmes d’information et ses conseils, pour son accueil et ses conseils.
* **M. Abdellatif SAHLALI**, Senior RPA, pour son aide et son encadrement technique et ses encouragements tout au long du stage.
* **M. Mohammed HADIK**, consultant RPA, pour son assistance et partage d’expérience et son orientation.
* **Mlle. Imane OURRARHI**, consultante RPA, pour son effort et ses conseils et directives.
* **M. Jean Francois Charpentier**, Senior IT, et toute son équipe.
* Et tout le personnel de Deloitte Nearshore pour leur gentillesse et leur compréhension.

Nos remerciements vont enfin à toute personne qui a contribué de près ou de loin à l’élaboration de ce travail.

Résumé

PROJET FIN D’ETUDE

ECOLE NATIONALE DES SCIENCES APPLIQUEES

-Kénitra –

Conception et Réalisation d’un Chatbot assisté par la RPA :

Résumé

Dans le cadre de notre projet de fin d’étude au sein de l’Ecole Nationale des Sciences Appliquées Kénitra, on a effectué ce stage de durée de 6 mois qui avait pour objectif, la mise en place d’un chatbot qui va servir faciliter les tâches quotidiennes/hebdomadaires/mensuelles des collaborateurs du pôle techno de Deloitte, en se basant sur la technologie RPA (Robotic Process Automation) qui est le sujet principal de notre mémoire PFE.

Abstract

As part of our end-of-study project within the National School of Applied Sciences Kénitra, we did this 6-month internship which aimed to set up a chatbot which will be used to facilitate the daily / weekly / monthly tasks of the collaborators of the techno pole of Deloitte, while being based on technology RPA (Robotic Process Automation) which is the main subject of our memory of our End of Study Project.

Mots-clés : Chatbot, RPA, IA,NLP,NodeJs,UiPath

Liste des figures

[Figure 1 : Logo Deloitte 11](#_Toc43460906)

[Figure 2 : Méthodologie RPA - Phases du projet 17](#_Toc43460907)

[Figure 3 : Diagramme de Gantt 20](#_Toc43460908)

[Figure 4 : RPA (Robotic Process Automation) 23](#_Toc43460909)

[Figure 5 : Avantages du RPA 24](#_Toc43460910)

[Figure 6 : Tableau de bord de UiPath Orchestrator 26](#_Toc43460911)

[Figure 7 : Fonctionnement d'un Chatbot 27](#_Toc43460912)

[Figure 8 : Fonctionnement d'un Moteur de traitement de langage naturel (NLP) 29](#_Toc43460913)

[Figure 9 : Interface de gestion d'un agent sur DialogFlow (onglet intents) 31](#_Toc43460914)

[Figure 10 : Vue sur un intent sous DialoFlow 31](#_Toc43460915)

[Figure 11 : Langages de programmation supporté par les librarires de DialogFlow 32](#_Toc43460916)

[Figure 12 : Action et paramètres - DialogFlow 33](#_Toc43460917)

[Figure 13 : Vue sur le code du traitement de la réponse reçue de chez DialogFlow 33](#_Toc43460918)

[Figure 14 : Vue sur les files d'attente dans l'orchestrateur 34](#_Toc43460919)

[Figure 15 : Vue sur les détails d'une transaction 35](#_Toc43460920)

[Figure 16 : Diagramme de cas d'utilisation général 37](#_Toc43460921)

[Figure 17 : Diagramme de cas d'utilisation « Services Skype » 38](#_Toc43460922)

[Figure 18 : Diagramme de cas d'utilisation « Services Redmine » 39](#_Toc43460923)

[Figure 19 : Diagramme de cas d'utilisation « Services Outlook » 40](#_Toc43460924)

[Figure 20 : Architecture globale du Chatbot 43](#_Toc43460925)

[Figure 21 : Architecture logicielle du chatbot 44](#_Toc43460926)

[Figure 22 : Diagramme de séquence général 45](#_Toc43460927)

[Figure 23 : Diagramme de séquence « Services Skype » 46](#_Toc43460928)

[Figure 24 : Diagramme de séquence « Services Redmine » 47](#_Toc43460929)

[Figure 25 : Digramme de séquence « Services Outlook » 48](#_Toc43460930)

[Figure 26 : Diagramme d’activité « services Skype « 49](#_Toc43460931)

[Figure 27 : Diagramme d'activité « Service Redmine » 50](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460932)

[Figure 28 : Diagramme d'activité « Services Outlook » 51](#_Toc43460933)

[Figure 29 : Interface d'accueil (Web et mobile) 56](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460934)

[Figure 30 : Liste des choix pour la fonctionnalité Skype 57](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460935)

[Figure 31 : Envoyer un message à un contact 57](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460936)

[Figure 32 : Choix de la date de la réunion 58](#_Toc43460937)

[Figure 33 : Confirmation de la réunion 59](#_Toc43460938)

[Figure 34 : Message de succès (réunion créée) 59](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460939)

[Figure 35 : Vérifier le statut d'un contact sur Skype 60](#_Toc43460940)

[Figure 36 : Liste des choix fonctionnalité Redmine 61](#_Toc43460941)

[Figure 37 : Liste déroulante (choix de la personne qu'on veut afficher ses tickets) 61](#_Toc43460942)

[Figure 38 : Résultats de la recherche (tableau) 62](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460943)

[Figure 39 : procédure de création du ticket 62](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460944)

[Figure 40 : Message de création du ticket 63](#_Toc43460945)

[Figure 41 : Création réussie du ticket sous redmine 63](#_Toc43460946)

[Figure 42 : Procédure modification du ticket Redmine 64](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460947)

[Figure 43 : Message de succès (modification du ticket Redmine) 64](#_Toc43460948)

[Figure 44 : Modification du ticket réussie sous Redmine 65](#_Toc43460949)

[Figure 45 : Procédure de création d'une réunion Teams 65](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460950)

[Figure 46 : Message de succès (Envoie du mail) 66](file:///C:\Users\omouaad\Desktop\pfe\RapportDeStage_V4.docx#_Toc43460951)

Liste des tableaux

[Tableau 1 : Comparaison des méthodes classiques et agiles 19](#_Toc43460952)

[Tableau 2 : Table des fonctionnalités 35](#_Toc43460953)

[Tableau 3 : Présentation des acteurs 36](#_Toc43460954)

[Tableau 4 : Description textuelle du cas « Créer une réunion Skype » 39](#_Toc43460955)

[Tableau 5 : Description textuelle du cas « Chercher un ticket Redmine » 40](#_Toc43460956)

[Tableau 6 : Description textuelle du cas « Envoyer un mail » 41](#_Toc43460957)

Table des matières

[Remerciement 1](#_Toc43461048)

[Résumé 2](#_Toc43461049)

[Liste des figures 3](#_Toc43461050)

[Liste des tableaux 4](#_Toc43461051)

[Table des matières 5](#_Toc43461052)

[Introduction Générale 7](#_Toc43461053)

[CHAPITRE I : Présentation générale 10](#_Toc43461054)

[I. Présentation de l’organisme d’accueil 11](#_Toc43461055)

[I.1. Historique 11](#_Toc43461056)

[I.2. Services 11](#_Toc43461057)

[I.2.1. Audit 11](#_Toc43461058)

[I.2.2. Financial Advisory 12](#_Toc43461059)

[I.2.3. Risk Advisory 12](#_Toc43461060)

[I.2.4. Expertise Comptable 13](#_Toc43461061)

[I.2.5. Fiscalité 13](#_Toc43461062)

[I.2.6. Consulting 13](#_Toc43461063)

[I.3. Unité d’accueil du stage 14](#_Toc43461064)

[I.4. Valeurs 14](#_Toc43461065)

[I.5. Structure et organisation interne 15](#_Toc43461066)

[II. Présentation du projet 16](#_Toc43461067)

[II.1. Périmètre du projet 16](#_Toc43461068)

[II.2. Objectif du projet 16](#_Toc43461069)

[II.3. Méthodologie de travail 17](#_Toc43461070)

[II.3.1. Phases d’un projet RPA 17](#_Toc43461071)

[II.3.2. Les méthodes agiles 18](#_Toc43461072)

[II.4. Planification du projet 19](#_Toc43461073)

[Conclusion 20](#_Toc43461074)

[CHAPITRE II : Etude préalable 21](#_Toc43461075)

[I. Etude de l’existant 22](#_Toc43461076)

[I.1. Description et critique de l’existant 22](#_Toc43461077)

[I.2. Solution proposée 22](#_Toc43461078)

[II. RPA 22](#_Toc43461079)

[II.1. C’est quoi le RPA ? 23](#_Toc43461080)

[II.2. Cas d’usage RPA 24](#_Toc43461081)

[II.3. Outils RPA 24](#_Toc43461082)

[I .3.1. L’outil UiPath 25](#_Toc43461083)

[III. Agent conversationnel (Chatbot) 27](#_Toc43461084)

[III.1. Définition 27](#_Toc43461085)

[III.2. Traitement du langage naturel NLP (Natural Language Processing) 28](#_Toc43461086)

[III.2.1. Définition 28](#_Toc43461087)

[III.2.3. Moteurs NLP /NLU 29](#_Toc43461088)

[III.2.4. Choix du moteur NLP 30](#_Toc43461089)

[IV. Interaction Chatbot/RPA 31](#_Toc43461090)

[IV.1. Choix de Node.Js 32](#_Toc43461091)

[V. Spécifications des besoins 35](#_Toc43461092)

[V.1. Besoins fonctionnels 35](#_Toc43461093)

[V.2. Besoins non fonctionnels 36](#_Toc43461094)

[V.3. Diagrammes de cas d’utilisation 36](#_Toc43461095)

[V.3.1. Présentation des acteurs 36](#_Toc43461096)

[V.3.2. Diagramme de cas d’utilisation général 37](#_Toc43461097)

[V.3.3. Principaux diagrammes de cas d’utilisation 38](#_Toc43461098)

[Conclusion 41](#_Toc43461099)

[CHAPITRE III : Conception détaillée 42](#_Toc43461100)

[IV. Conception architecturale 43](#_Toc43461101)

[IV.1. Architecture adoptée 43](#_Toc43461102)

[IV.1. Architecture logicielle 44](#_Toc43461103)

[V. Conception aspect dynamique 44](#_Toc43461104)

[V.1. Diagrammes de séquence 44](#_Toc43461105)

[V.1.2. Diagramme de séquence général : 44](#_Toc43461106)

[V.1.2. Principaux Diagrammes de séquence : 45](#_Toc43461107)

[V.2. Diagrammes d’activité 48](#_Toc43461108)

[Conclusion 52](#_Toc43461109)

[CHAPITRE IV : Réalisation 53](#_Toc43461110)

[I. Environnement du travail 54](#_Toc43461111)

[I.1. Environnement matériel 54](#_Toc43461112)

[I.2. Environnement logiciel 54](#_Toc43461113)

[I.2.1. Outils de développement 54](#_Toc43461114)

[I.2.2. API et bibliothèques 55](#_Toc43461115)

[I.2.3. Gestionnaire de dépendances 55](#_Toc43461116)

[I.2.4. Serveurs Web 55](#_Toc43461117)

[I.2.5. Frameworks 55](#_Toc43461118)

[III. Interaction avec les différentes interfaces graphiques 56](#_Toc43461119)

[III.1. Interface d’accueil 56](#_Toc43461120)

[III.2. Interface Skype 56](#_Toc43461121)

[III.2.1 Envoyer un message à un contact 57](#_Toc43461122)

[III.2.2 Organiser une réunion Skype 57](#_Toc43461123)

[III.2.3 Vérifier le statut actuel d’un contact 59](#_Toc43461124)

[IV.1. Interface Redmine 60](#_Toc43461125)

[IV.1.1 Chercher un ticket Redmine 60](#_Toc43461126)

[III.3.2 Créer un ticket Redmine 62](#_Toc43461127)

[III.3.3 Modifier un ticket Redmine 63](#_Toc43461128)

[III.4. Interface Outlook 65](#_Toc43461129)

[III.4.1. Créer une réunion Microsoft Teams 65](#_Toc43461130)

[III.4.2. Envoie d’un mail Outlook 66](#_Toc43461131)

[Conclusion 66](#_Toc43461132)

[Conclusion Générale 67](#_Toc43461133)

[Webographie 68](#_Toc43461134)

Introduction Générale

L

e but de ce projet est de montrer la puissance des **chatbots** et comment ils peuvent être une alternative à l'utilisation d'une application ou même d'un site web.

Les chatbots (front office bots) conversent avec des clients ou des employés pour envoyer des informations, effectuer des tâches ou capturer leurs demandes. En fonction du cas d'utilisation, un bot doit s'intégrer à différents systèmes d'entreprise et y accéder.

Ces systèmes peuvent être de type help desk, intranet, CRM, Business Intelligence, LOB, bases de connaissances RH, etc. Si ces systèmes ont des API modernes, le chatbot peut accéder aux informations requises indépendamment et sans aucun problème. Cependant, si les systèmes ne disposent pas d'API modernes, le chatbot peut ne pas être en mesure d'intégrer et de récupérer des informations.

C'est là que **RPA** entre en jeu. L'intégration de RPA aide les chatbots à naviguer efficacement dans les systèmes d'entreprise hérités qui ne disposent pas d'API modernes.

RPA permet aux chatbots de récupérer des informations à partir de ces systèmes et de gérer à plus grande échelle les demandes et les requêtes clients / employés plus complexes et en temps réel.

De la même manière, les chatbots, à la demande d'un utilisateur, peuvent déclencher RPA pour effectuer des tâches banales spécifiques sans les acheminer vers un agent humain.

La combinaison de chatbots et de RPA peut résoudre plusieurs problèmes courants auxquels les entreprises sont confrontées aujourd'hui. Il aide les organisations à répondre aux attentes croissantes des clients à moindre coût en combinant les capacités d'automatisation de RPA et les fonctionnalités de libre-service d'un chatbot. Cela se traduit également par une productivité accrue des agents, car les agents n'ont pas à consacrer de temps à des activités banales et routinières telles que la collecte de données client, la copie d'informations, la rédaction de documents, etc.

Le présent rapport sera donc organisé en cinq chapitres en vue de détailler le projet :

• Le premier chapitre intitulé « Présentation générale » qui présente l’organisme d’accueil Deloitte, présentation du projet et la conduite du projet.

• Le deuxième chapitre intitulé « Etude préalable » qui décrit l’étude de l’existant ainsi que l’étude de la solution y compris les besoins fonctionnels et non fonctionnels.

• Le troisième chapitre intitulé « Etude Détaillée » qui contient une analyse profonde de la solution proposée ainsi que la conception du projet.

• Le cinquième chapitre intitulé « Réalisation » qui présente l’environnement de travail et les différentes captures d’écran du projet.

CHAPITRE I : Présentation générale

Dans ce chapitre, nous décrivons le contexte général du projet. Nous présentons d’abord l’entreprise au sein de laquelle se déroule le stage et nous enchainons par la suite par l’exposé du sujet en expliquant ses différents modules.

1. Présentation de l’organisme d’accueil

I.1. Historique

*Deloitte au niveau mondial :*

**Deloitte** est un des [quatre plus importants cabinets d'audit et de conseil mondiaux](https://fr.wikipedia.org/wiki/Big_Four_(audit)) avec un chiffre d’affaires mondial de 46,2 milliards de dollars en 2019. Il s'agit également du plus grand cabinet d'audit au monde en effectifs avec 312 000 employés, et du plus ancien du [*Big Four*](https://fr.wikipedia.org/wiki/Big_Four_(audit)).

Au sein des [Big Four](https://fr.wikipedia.org/wiki/Fat_four), **Deloitte** est le plus ancien des cabinets d'audit, l'une de ses entités remontant aux [années 1840](https://fr.wikipedia.org/wiki/Ann%C3%A9es_1840). Le premier cabinet a été fondé par William Welch Deloitte en [1845](https://fr.wikipedia.org/wiki/1845) à [Londres](https://fr.wikipedia.org/wiki/Londres). William Welch Deloitte a par la suite ouvert un bureau à [New York](https://fr.wikipedia.org/wiki/New_York) en [1880](https://fr.wikipedia.org/wiki/1880).

*Deloitte Maroc :*

Crée en 1994, à l'initiative de Deloitte & Touche France, le cabinet d'audit Deloitte & Touche Maroc voit le jour et devient affilié au réseau international de Deloitte, leader mondial des métiers d'audit et de conseil. La diversité des missions et des clients, le dynamisme et les relations de confiance tissées au plus haut niveau sont parmi les caractéristiques les plus attrayantes de notre firme marocaine. En 2003, suivant la décision de la firme internationale de Deloitte Touche Tohmatsu, la firme marocaine devient « Deloitte ».



Figure 1 : Logo Deloitte

I.2. Services

I.2.1. Audit

L’audit financier consiste à donner une opinion sur l’information financière mise à la disposition des marchés financiers et des différentes parties prenantes de l’entreprise.

L’activité audit est organisée autour d’expertises techniques et de savoir-faire variés suivants :

### Audit légal et contractuel.

### Investigation & Consolidation.

### Asset Management.

L’ensemble des collaborateurs spécialisés dans les métiers d’Audit est regroupé comme suit :

### Aviation et Transports

### Distribution et produits de grande consommation

### Energie et Ressources

### Institutions financières

### Secteur public

### Technologie, Médias et Télécommunications

I.2.2. Financial Advisory

Formidable accélérateur de croissance et de transformation, l’activité de fusions-acquisitions est certainement l’une des plus complexes qu’une entreprise exécute.

Il s’agit de la conduite et les négociations d'opérations de fusions/acquisitions, depuis l'évaluation jusqu'aux conseils pour l'intégration des sociétés acquises.

Deloitte Maroc intervient dans les opérations sur le capital, les montages financiers, mais aussi auprès des entreprises en difficulté pour les assister dans leur plan de restructuration, de redressement ou de reprise.

Deloitte dispose d’une organisation efficiente en 5 grandes lignes de services :

### Transactions Services

### Fusions-acquisitions et ingénierie financière.

### Evaluation financière.

### Litiges financiers et fraudes.

### Restructuration d'entreprises en difficulté

I.2.3. Risk Advisory

Les fonctions liées à la maîtrise des risques et à la gouvernance se déclinent de manière spécifique à chaque entreprise et secteur d’activité. Malgré la diversité des fonctions et des pratiques, les enjeux sont in fine les mêmes pour tous.

Cette activité concerne principalement la sécurité des systèmes, processus et organisations à travers les services suivants :

### Audit des systèmes d'information.

### Qualité et sécurité des processus.

### Sécurité des systèmes, des réseaux, du commerce électronique, de l'information.

### Externalisation de la fonction audit interne.

### Audit de projet.

### Actuariat.

### « IAS/IFRS ».

### Gestion des risques de taux et change, évaluation d'instruments financiers, passage aux nouvelles normes comptables.

### Management des risques opérationnels & conseil en respect des réglementations.

I.2.4. Expertise Comptable

L’activité Expertise Comptable comporte les métiers suivants :

### Tenue des comptes, supervision comptable.

### Conseil en organisation et systèmes d’informations comptables.

I.2.5. Fiscalité

Les investisseurs internationaux attirés par les opportunités sur le continent africain au même titre que les compagnies locales en quête d’internationalisation doivent porter une attention particulière à la fiscalité locale et régionale en Afrique.

I.2.6. Consulting

Deloitte, leader mondial du conseil, se distingue par sa capacité à aider ses clients à résoudre leurs problèmes les plus complexes, de la définition de leur stratégie à la sécurisation de leur mise en œuvre.

L’activité Consulting comporte les métiers suivants :

### Stratégie, Analytics et M&A

### Client & Marketing

### Opérations commerciales de base

### Capital humain

### Technologie et performances d'entreprise

I.2.7. Juridique

L’activité Juridique se décline à travers les différents axes suivants :

### Diagnostics et recommandations juridiques, fiscales et sociales.

### Montages juridiques.

### Gestion des opérations sociales et patrimoniales.

### Procédures contentieuses.

### Due diligence.

I.3. Unité d’accueil du stage

Comme montré dans la partie précédente (Services), Deloitte offre 7 services différents à ses clients, parmi ces services on trouve le *consulting* : c’est un service dont le but derrière est de trouver les solutions des problèmes complexes des clients, où la nécessité de l’intervention de la technologie.

Sur ce, Deloitte s’intéresse à un nombre de technologies qui sont très tendances, telles que : SAP, Salesforces, DataMining, Digital, RPA...

Durant ce stage, nous avons été affectés à l’équipe RPA, qui était le nouveau-né de Deloitte et dont le but de notre intégration était d’apporter de nouvelles idées et solutions à l’équipe en exploitant nos expériences en développement et en programmation.

I.4. Valeurs

Le succès de Deloitte repose sur un ensemble de valeurs plus que jamais nécessaires pour répondre aux attentes et aux enjeux de nos clients. Ces valeurs constituent le socle de leur double ambition : la confiance des clients et l'excellence des collaborateurs.

# Lead the way :

Deloitte n’est pas que leader de sa profession, nous la réinventons pour demain. Nous nous engageons à créer des opportunités, à innover, à ouvrir la voie à un monde plus durable.

# Serve with integrity :

En agissant avec intégrité et dans le respect des règles d’éthique, Deloitte a gagné la confiance de ses clients, des régulateurs et du public. Préserver cette confiance est de nos responsabilités la plus importante.

* Take care of each other:

Nous nous soucions de chacun et donnons la priorité au respect, à l’équité, au bien-être et au développement.

* Foster inclusion :

Nous faisons le maximum pour encourager une culture basée sur l’inclusion et favoriser la diversité sous toutes ses formes. Attirer les meilleurs talents est source d'innovation et d'accélération de la performance.

* Collaborate for measurable impact :

Nous abordons nos travaux dans un esprit collaboratif en nous appuyant sur des équipes pluridisciplinaires et internationales pour offrir des résultats concrets, mesurables et identifiables.

I.5. Structure et organisation interne

1. Présentation du projet

II.1. Périmètre du projet

Le sujet de notre projet fin d’études est la conception et la réalisation d’un agent conversationnel (Chatbot), qui va servir faciliter les tâches des collaborateurs au sein de Deloitte.

Notre chatbot dans ce projet répondra à des demandes qui sollicitent l’interaction avec des outils de communication, management de projet, saisie feuille de temps…, dans le but de l’automatisation des tâches répétitives en permettant aux collaborateurs de se consacrer aux taches à plus forte valeur ajoutée.

II.2. Objectif du projet

Le sujet s’intitule « Conception et Réalisation d’un chatbot assisté par le RPA »  
L’objectif du projet est de concevoir, réaliser et implémenter une solution d’automatisation sous forme d’un agent conversationnel (Chatbot), qui permettra de remplacer les employés des charges ennuyantes. Ce robot doit assurer :

* *Organisation des réunions :* réunions sur skype ou teams
* *Recherche des tickets suivant des filtres :* par auteur, nom de projet, statut…
* *Création /modification des tickets :* créer des tickets sous la solution de gestion de projet.
* *Saisie de la feuille de temps :* remplir la feuille de temps chaque quinzaine.

## *Renseignement sur les statuts des collaborateurs sur skype.*

* *Guider les nouveaux entrants chez Deloitte :* informations sur les personnes à rejoindre en cas de besoins RH, administration, Help Desk …

Ainsi la réalisation de ce chatbot pourra aussi être utile pour les clients de Deloitte, où il servira une Interface Homme Machine à travers laquelle le client pourra demander l’exécution d’une tâche souhaitée, ensuite le chatbot lancera le processus adapté à la demande de l’utilisateur, comme ça l’utilisation sera plus fluide et moins complexe.

II.3. Méthodologie de travail

II.3.1. Phases d’un projet RPA

Dans l’équipe RPA, un projet passe par un ensemble séquentiel de phases, dont le nom et le nombre sont déterminés en fonction des besoins de projet, permettant généralement le développement d’un produit ou d’un service. Le modèle de cycle de vie est la description d'un processus couvrant les phases de :***Discovery, Design, Modélisation/Développement, Mise en œuvre/Déploiement, Stabilisation***. Comme illustré dans la figure N°2 :

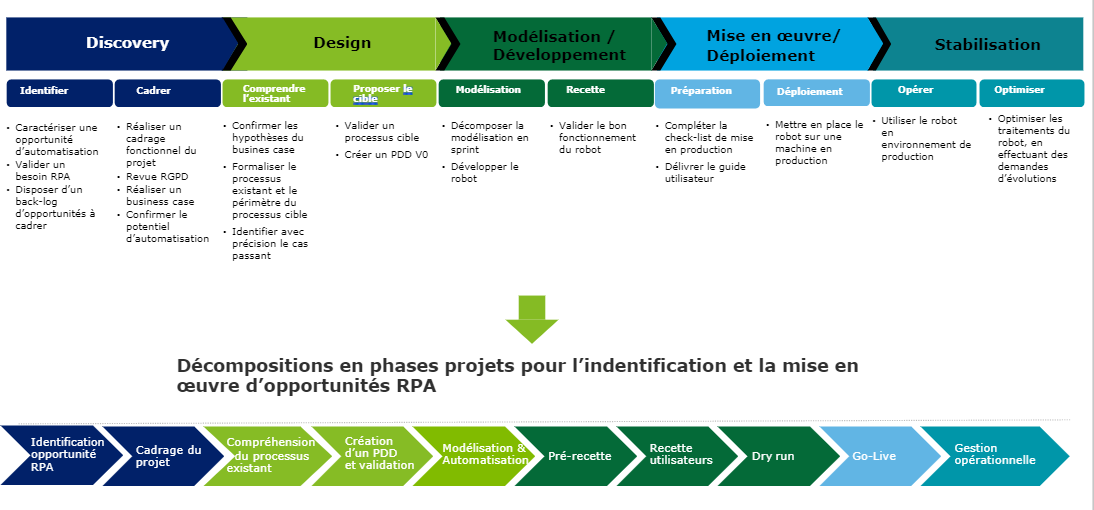


Figure 2 : Méthodologie RPA - Phases du projet

**Discovery :**

Analyse d’une opportunité RPA :

* Complétude du questionnaire de recensement avec le product owner.
* Validation de l’éligibilité du processus.
* Réalisation d’un recording du processus existant (cas passant).
* Complétude du business case.
* Rédaction du PQD pour validation.

**Design :**

* Définition des prérequis projets et techniques.
* Définition des besoins métiers.
* Réalisation de sessions de recording.
* Définition des sprints.
* Rédaction du PDD cible et validation.

**Modélisation / développement :**

* Décomposer la modélisation en sprint.
* Développement.
* Rédaction du cahier de recette (définition de la stratégie de recette…)
* Réalisation d’un plan de tests et validation par le product owner de la bonne couverture des scénarii de test.
* Réalisation d’un PV de recette.

**Mise en œuvre / Déploiement :**

* Rédaction d’un guide de lancement du robot à destination du product owner.
* Mise en place du robot dans une machine de production.

**Stabilisation :**

* Optimiser des traitements du robot, en effectuant des demandes d’évolution.

II.3.2. Les méthodes agiles

Les méthodes agiles représentent un mouvement novateur qui vise à apporter plus de valeur aux clients et aux utilisateurs, ainsi qu’une plus grande satisfaction dans leur travail aux membres de l’équipe.

Le but d’une méthode agile est de maximiser la valeur ajoutée :

|  |  |
| --- | --- |
| Méthode | Caractéristiques |
| Méthodes classiques | * Modèles stricts. * Etapes très clairement définies. * Documentation très fournie. * Fonctionne bien avec les gros projets et les projets gouvernementaux. |
| Méthodes agiles | * Modèles incrémentaux et itératifs. * Petites et fréquentes livraisons. * Accent sur le code et moins sur la documentation. * Convient aux projets de petites et moyennes tailles. |

• Le développement s’effectuant par itérations successives, il est possible, à la fin de chaque itération, de changer les priorités en faisant en sorte que les éléments apportant le plus de valeur soient réalisés en premier.

• Un meilleur accomplissement des personnes impliquées dans un développement est rendu possible par la notion d’équipe autoorganisée.

Tableau 1 : Comparaison des méthodes classiques et agiles

II.4. Planification du projet

Le diagramme de GANTT ou diagramme à barres est l'un des outils de planification le plus utilisé. Cet outil permet de situer les tâches à accomplir dans le temps. Il permet restreindre les délais, grâce à la possibilité de chevauchement des tâches qu'ore son tracé. La construction du GANTT retient en abscisse, le temps nécessaire à l'exécution des opérations. A l'ordonnée, on représente soit les tâches soit les ressources affectées aux différentes opérations. La figure suivante représente l’enchainement des tâches. La figure 3 contient le diagramme de GANTT réel pour notre stage.

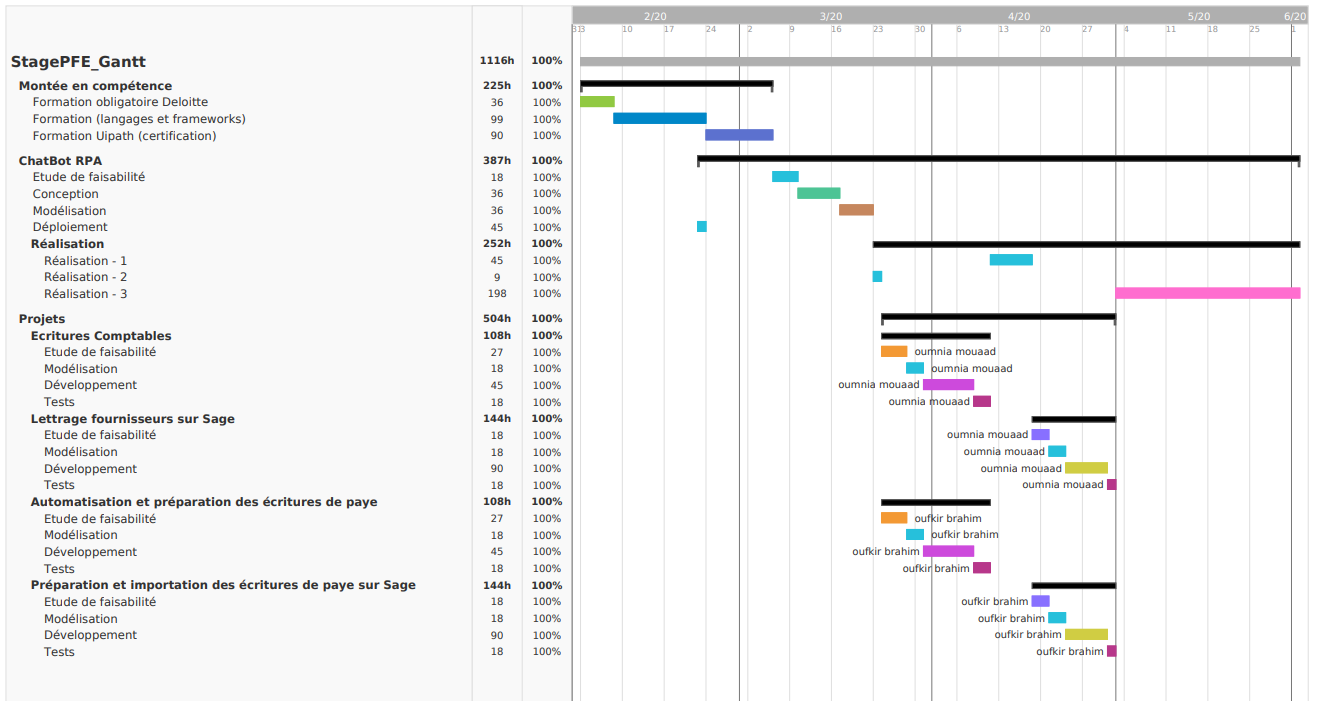


Figure 3 : Diagramme de Gantt

Conclusion

Dans ce chapitre, nous avons essayé d’expliquer le contexte du projet et dévoiler l’objectif du travail demandé. Avant de développer toute application informatique, il est absolument nécessaire d’étudier le système existant, de dégager ses fonctionnalités, et surtout d’en connaître les failles afin de trouver les solutions adéquates pour le travail demandé. Cette étude sera le sujet du chapitre suivant.

CHAPITRE II : Etude préalable

L’étude préalable constitue une étape préliminaire pour la réalisation d’une application. En effet, elle permet d’analyser, d’évaluer et de critiquer le fonctionnement habituel, tout en élaborant la liste des solutions possibles afin d’en retenir une comme solution finale. Nous commençons ce chapitre par une étude de l’existant. Ensuite, nous réservons le reste du chapitre à l’étude de la solution adoptée.

1. Etude de l’existant

I.1. Description et critique de l’existant

Puisque Deloitte fait partie des plus grandes entreprises digitales au monde, elle doit être au top de la technologie pour faciliter le travail de ses collaborateurs. Normalement, un collaborateur regarde son emploi du temps et planifie ses réunions, réserve la salle selon sa disponibilité de façon manuelle. A la fin de chaque quinzaine, le collaborateur doit soumettre ses feuilles temps, créer des tickets …

Ces activités prennent beaucoup de temps quand on est une personne à grande responsabilité et ayant un planning charge, il est difficile d’organiser une réunion parce que les disponibilités des participants se chevauchent. Aussi pour les équipes techniques, ils ont besoin de créer des tickets soit pour remonter des anomalies ou bien pour des évolutions.

Le processus actuellement adopté n’assure pas la rapidité d’exécution des tâches des collaborateurs, du coup Deloitte a remarqué qu’elle pourrait gagner plus de temps de productivité si elle arrive à les automatiser.

I.2. Solution proposée

L’idée est de proposer une aide active sans effectuer un tas de manipulation clavier/souris. Cette aide devrait être proposée aux collaborateurs au moment voulu. D’où vient l’idée de créer un Chabot (un agent conversationnel), ce dernier peut guider l’utilisateur rapidement afin de lui trouver la solution sur les différentes plateformes/applications utilisées, de ce fait une optimisation de temps, de coût et de qualité des réponses.

La valeur ajoutée de cet agent conversationnel, est que les processus qui seront déclenchés vont être exécutés par des robots RPA (Uipath) ou robots Selenium, c’est une idée qui n’est pas trop consommée, d’où l’originalité de ce projet, chose qui nous a poussé à prendre le challenge de faire interagir le RPA avec un agent conversationnel et surtout de le faire réussir.

Dans la partie qui suit, nous allons expliquer c’est quoi le RPA ainsi qu’un agent conversationnel et la liaison entre les deux.

1. RPA

Avant de parler de la solution et de passer aux détails, on a pensé qu’il est nécessaire d’expliquer tout d’abord le RPA et les outils avec lesquelles on travaille, ainsi le fonctionnement des robots RPA et l’environnement de travail.

II.1. C’est quoi le RPA ?

L'automatisation robotisée des processus (Robotic Process Automation, RPA) consiste à utiliser des logiciels dotés de fonctionnalités d'intelligence artificielle ([IA](https://www.lemagit.fr/definition/AI)) et d'[apprentissage statistique](https://whatis.techtarget.com/fr/definition/Machine-Learning) pour traiter des tâches volumineuses et répétitives qui, par le passé, étaient réalisées par des humains. Il peut s'agir notamment de requêtes, de calculs, et de tâches de maintenance des enregistrements et des transactions.

Les solutions RPA – Robotic Process Automation – répondent pleinement aux challenges des entreprises qui sont en recherche constante d’accélérateurs afin d’améliorer la performance opérationnelle, le service client et l’engagement des collaborateurs. Challenge en automatisant les tâches répétitives et permettant ainsi aux collaborateurs de se consacrer aux taches à plus forte valeur ajoutée.

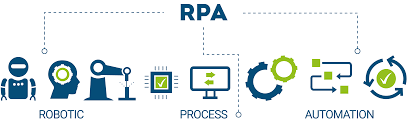


Figure 4 : RPA (Robotic Process Automation)

Ces solutions peuvent traiter n’importe quelles données d’entrée en exécutant, telle une macro, une série d’actions préprogrammées et en suivant des règles métiers prédéfinies. Son champ d’application est vaste puisqu’elle peut s’appliquer à n’importe quel type de processus métier (banque, assurance, administration…) ou de fonctions supports (Finance, Ressources humaines, Achats, IT…). Ces technologies utilisent les systèmes legacy en l’état et peuvent être déployées de façon agile très rapidement et à moindre coût.

En d’autres termes, ces nouvelles solutions permettent de :

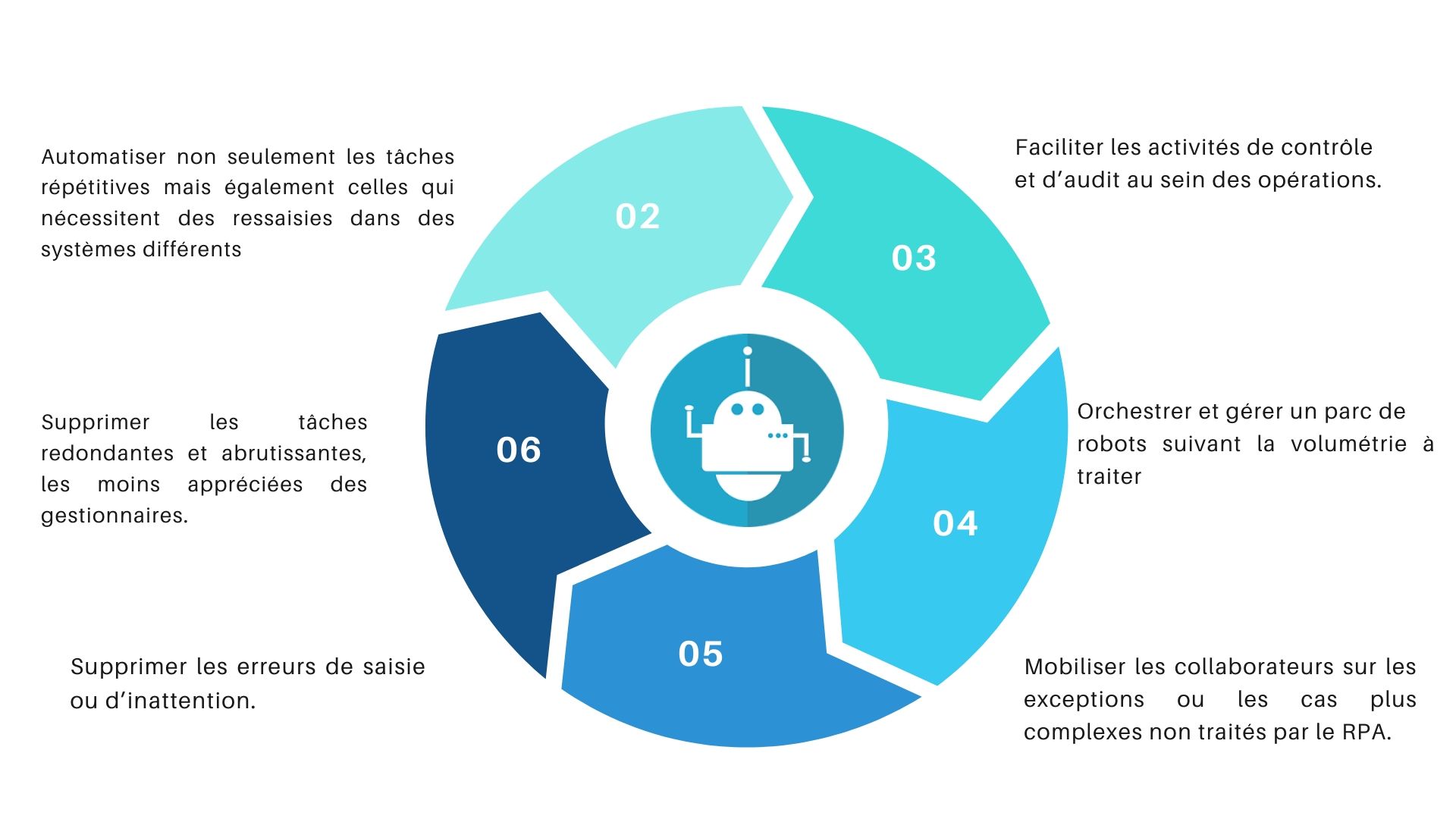


Figure 5 : Avantages du RPA

II.2. Cas d’usage RPA

Les outils RPA sont **utilisés dans une large variété de secteurs**. Ils sont notamment utiles pour offrir un meilleur service client en automatisant par exemple les processus de centres de contact tels que la vérification d’informations.

Les tâches de comptabilité [telles que le reporting](https://www.lebigdata.fr/tagetik-reporting-financier) ou la budgétisation peuvent également être automatisées. Dans le domaine des services financiers, les RPA peuvent permettre d’automatiser des tâches comme l’ouverture ou la fermeture d’un compte ou la gestion des requêtes d’audits ou des réclamations d’assurance.

Dansl’industrie de la santé, les RPA permettent d’automatiser la prise en charge des rapports de patients, des réclamations, ainsi que la facturation, la gestion de compte, le reporting ou l’analyse de données.

Les ressources humaines peuvent s’en servir pour automatiser certaines tâches ou pour mettre à jour les informations sur les employés. Cette technologie sert aussi dans le domaine de la logistique pour automatiser les traitements et paiements de commandes, la surveillance des niveaux d’inventaires ou le suivi d’expéditions.

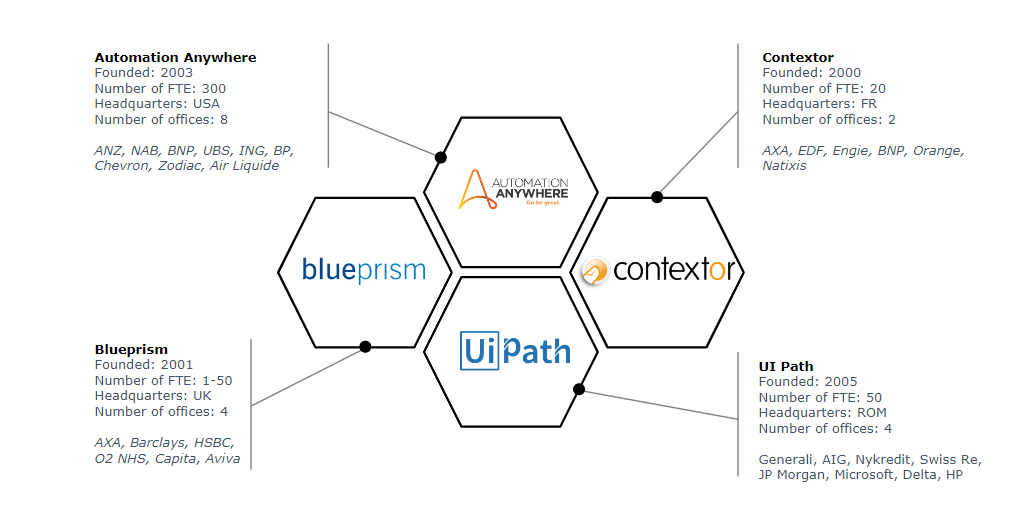
II.3. Outils RPA

Il existe de nombreux vendeurs de logiciels RPA, et le marché est en pleine croissance. Parmi les plus populaires, on compte Automation Anywhere Inc**.** et sa plateforme de main d’œuvre numérique dédiée aux ressources humaines, à la prise en charge de réclamations et autres processus back-office.

De son côté, Blue Prism propose une main d’œuvre virtuelle agile pouvant être gérée de façon centralisée.

Un autre vendeur est Contextor qui permet aux entreprises d’automatiser des tâches sans avoir besoin de code informatique.

La plateforme ouverte d’UiPath permet également d’automatiser les processus d’entreprise.



I .3.1. L’outil UiPath

**Deloitte est l’une des entreprises qui utilisent *UiPath* comme outil d’automatisation RPA, vu que cette plateforme présente plusieurs avantages. Tout d’abord, elle peut être hébergée sur des environnements Cloud ou des machines virtuelles. De plus, elle est compatible avec une large variété d’applications web et PC.**

**Son répertoire centralisé se révèle aussi très pratique pour gérer des flottes entières de robots. La sécurité est également au rendez-vous, avec la nécessité de s’identifier pour exécuter les robots.**

**La plateforme de RPA d’UiPath se décline en trois produits différents :** UiPath Studio**,** UiPath Robot**, et** UiPath Orchestrator**.**

1. UiPath studio

**Tout d’abord,**UiPathStudio**permet de concevoir les processus d’automatisation de façon visuelle à l’aide de différents diagrammes représentant les différentes catégories de tâches.**

1. UiPath Robot

La suite UiPath Robot quant à elle permet d’exécuter les processus conçus à l’aide du studio. Les robots se chargeront d’effectuer les tâches de manière autonome, ou lors de l’activation du processus par un humain.

1. UiPath Orchestrator

Enfin, l’UiPath Orchestrator se présente comme une application web. Elle permet de déployer, de programmer et de gérer les robots et les processus. Il s’agit donc d’une plateforme de gestion centralisée pour la RPA.

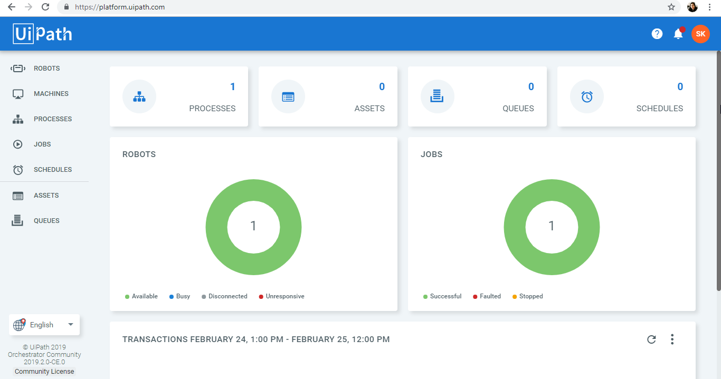
**

Figure 6 : Tableau de bord de UiPath Orchestrator

1. Agent conversationnel (Chatbot)

De la même manière, puisque notre projet contient le chatbot comme sujet principal, on a vu qu’en plus de parler de l’RPA, il est primordial de décortiquer un peu le terme agent conversationnel.

III.1. Définition

Afin d’établir une conversation entre homme et machine, une conversation homme machine en langage naturel est utilisée, c’est cela que l’on appelle communément un chatBot ou agent conversationnel. Un programme informatique écrit en langage quelconque répond instantanément, en se basent sur un corpus (ensemble de question / réponse).

Chatbots est la contraction de « chat » et de « robot ». On trouve également l’expression « Chatterbot » dans la littérature, et il est aussi courant de parler d’agent conversationnel.

Un Chatbot est une application complexe composée de plusieurs éléments, voici un schéma détaillé du fonctionnement classique d’un Chatbot.

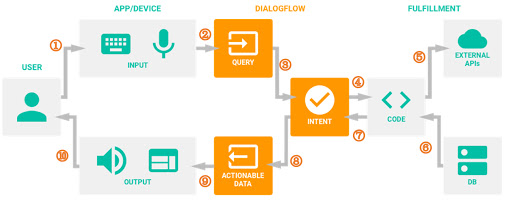


Figure 7 : Fonctionnement d'un Chatbot

De manière très classique, le fonctionnement d’un Chatbot est le suivant :

* L’utilisateur questionne le Chatbot, vocalement ou textuellement via une interface quelconque.
* Un moteur de traitement du langage naturel (NLU) reçoit le texte brut correspondant à la question de l’utilisateur, l’interprète et renvoie une donnée (la plupart du temps au format JSON) structurée, sous la forme intention/entités.
* Un moteur conversationnel côté serveur ou côté client traite la donnée retournée par le moteur NLU et construit une réponse en fonction de celle-ci. Le moteur conversationnel est en lien avec une base de connaissance ou tout autre support pour obtenir des informations pour construire la réponse.
* La réponse est retournée à l’utilisateur.

III.2. Traitement du langage naturel NLP (Natural Language Processing)

III.2.1. Définition

Le traitement automatique du langage naturel (abr. TALN), ou traitement automatique de la langue naturelle, est un domaine multidisciplinaire impliquant la [linguistique](https://fr.wikipedia.org/wiki/Linguistique), l'[informatique](https://fr.wikipedia.org/wiki/Informatique) et l'[intelligence artificielle](https://fr.wikipedia.org/wiki/Intelligence_artificielle), qui vise à créer des outils de traitement de la langue naturelle pour diverses applications. Il ne doit pas être confondu avec la [linguistique informatique](https://fr.wikipedia.org/wiki/Linguistique_informatique), qui vise à comprendre les langues au moyen d'outils informatiques.

III.2.2. Fonctionnement

1. Intention (intent)

L’**intention** (ou intent) est ce que souhaite savoir ou faire l’utilisateur.

Les énoncés (ou utterances) sont différentes formulations de la même intention.

Exemples d’énoncés pour l’intention “Obtenir la météo”

* Quel temps fait-il aujourd’hui à Paris ?
* Quelle est la météo à Paris aujourd’hui ?
* Est-ce qu’il fait beau aujourd’hui à Paris ?

1. Entité (Entity)

Les **entités**, appelées aussi parfois **slots**, sont les éléments de contexte. C’est ce sur quoi repose l’intention.

On peut voir cela comme les paramètres qui seront transmis à une API métier.

Dans l’exemple précédent, il y a donc deux entités :

* Aujourd’hui : entité de type temps
* Paris : entité de type lieu

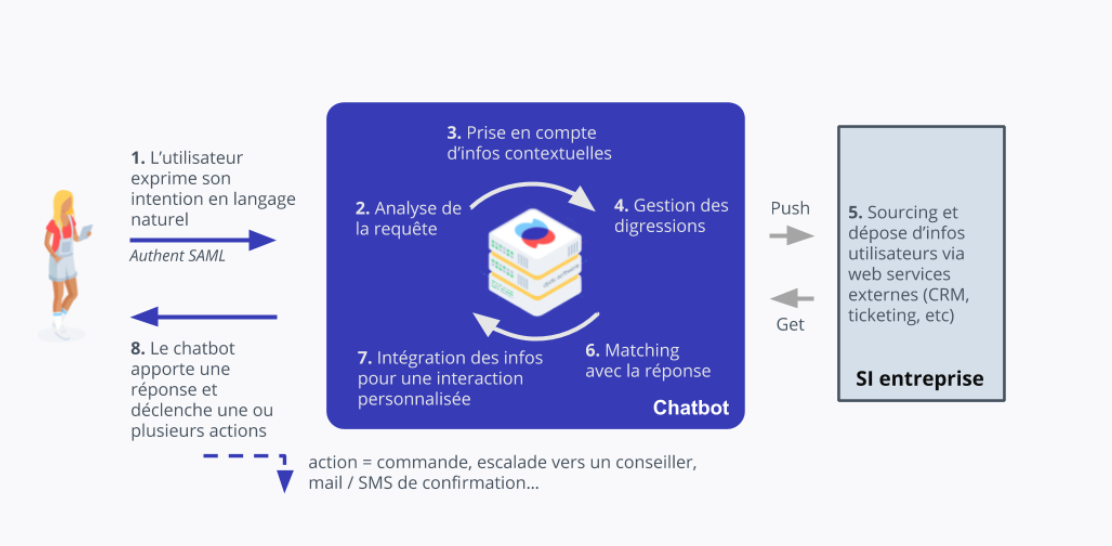


Figure 8 : Fonctionnement d'un Moteur de traitement de langage naturel (NLP)

III.2.3. Moteurs NLP /NLU

Nous distinguons les deux acronymes suivants :

* **NLU** – Natural Language Understanding : discipline de l’intelligence artificielle liée à la compréhension du langage
* **NLP** – Natural Language Processing : discipline de l’intelligence artificielle liée à l’interaction complète incluant la compréhension et la génération de langage naturel (NLG)

##### DialogFlow



*Dialogflow* est une startup rachetée par Google fin 2016.

Le nom de la marque a changé récemment, vous avez peut-être plutôt entendu parler d’**api.ai**.

## Avantages :

* Gestion de réponses spécifiques par plateforme (ex : slack, google assistant…) permettant de renvoyer des messages à un format compréhensible par celle-ci
* Très nombreux connecteurs pour déployer le chatbot sur de nombreuses plateformes (Messenger, Slack, Viber, Twitter, Skype, Alexa…)
* Fenêtre de chat de test très complète, affichant utterance, réponses du moteur, contextes, intent, action, entités détectées, valeurs ainsi que le flux JSON
* Très nombreux prebuild domains (template de chatbot en anglais). Exemple : commande au restaurant
* Intégration facile d’APIs métier via un webhook exposé sur internet
* Possibilité de coder un webhook simple via un éditeur web dans la console d’administration \*. Il est ensuite déployé dans Google Cloud Function.

##### Luis

*LUIS* fait partie des [Microsoft Cognitive Services](https://www.microsoft.com/cognitive-services/en-us/).

C’est un moteur NLU (Natural Language Understanding) hébergé sur le cloud de Microsoft.LUIS ne propose que le moteur NLU, c’est à dire la compréhension du langage, et ne permet pas de générer de réponse.Il doit être intégré dans le développement custom d’un chatbot.

## Avantages :

* Interface d’administration proposant un tableau de bord présentant les indicateurs principaux de l’agent
* Moteur complété par le [bot framework](https://dev.botframework.com/) (SDK open source en C# ou NodeJS) pour développer son chatbot
* Déploiement des bots sur différentes plateformes et publication sur un store publique grâce au [bot directory](https://dev.botframework.com/bots)
* Séparation de la configuration du moteur (définition des intents et entities) et de son entraînement.



##### Wit.AI

Startup rachetée par Facebook en janvier 2015.

Contrairement aux autres moteurs, il n’y a pas de distinction entre intention et entités, car tout est intent :

* *Free-text* et ou *keywords* pour les entités
* *Trait* pour les intentions

## Avantages :

* Logs des historiques des conversations passées sous forme d’un dialogue chat.
* Gestion d’un très grand nombre de langues.

III.2.4. Choix du moteur NLP

Notre choix s’était orienté vers DialogFlow, qui proposait également les mêmes services que luis.ai, sauf que l’interface de DialogFlow était plus attractive et rendait agréable le développement des intent/entities. De plus, il permet également aux développeurs d’intégrer l’agent conversationnel à travers différentes librairies clientes (le plus important pour nous). Quant à wit.ai, on a tout de suite écarté ce moteur de NLU à cause de son interface étrange et repoussante pour l’utilisateur, il était difficile de cerner le fonctionnement du service à cause de ce facteur, bien que celui-ci semble très performant.

L’image dessous visualise l’interface qui permet d’interagir avec l’agent crée dans DialogFlow, on trouve aussi les intents et les entités relatives.



Figure 9 : Interface de gestion d'un agent sur DialogFlow (onglet intents)

Voici à quoi ressemble un intent :

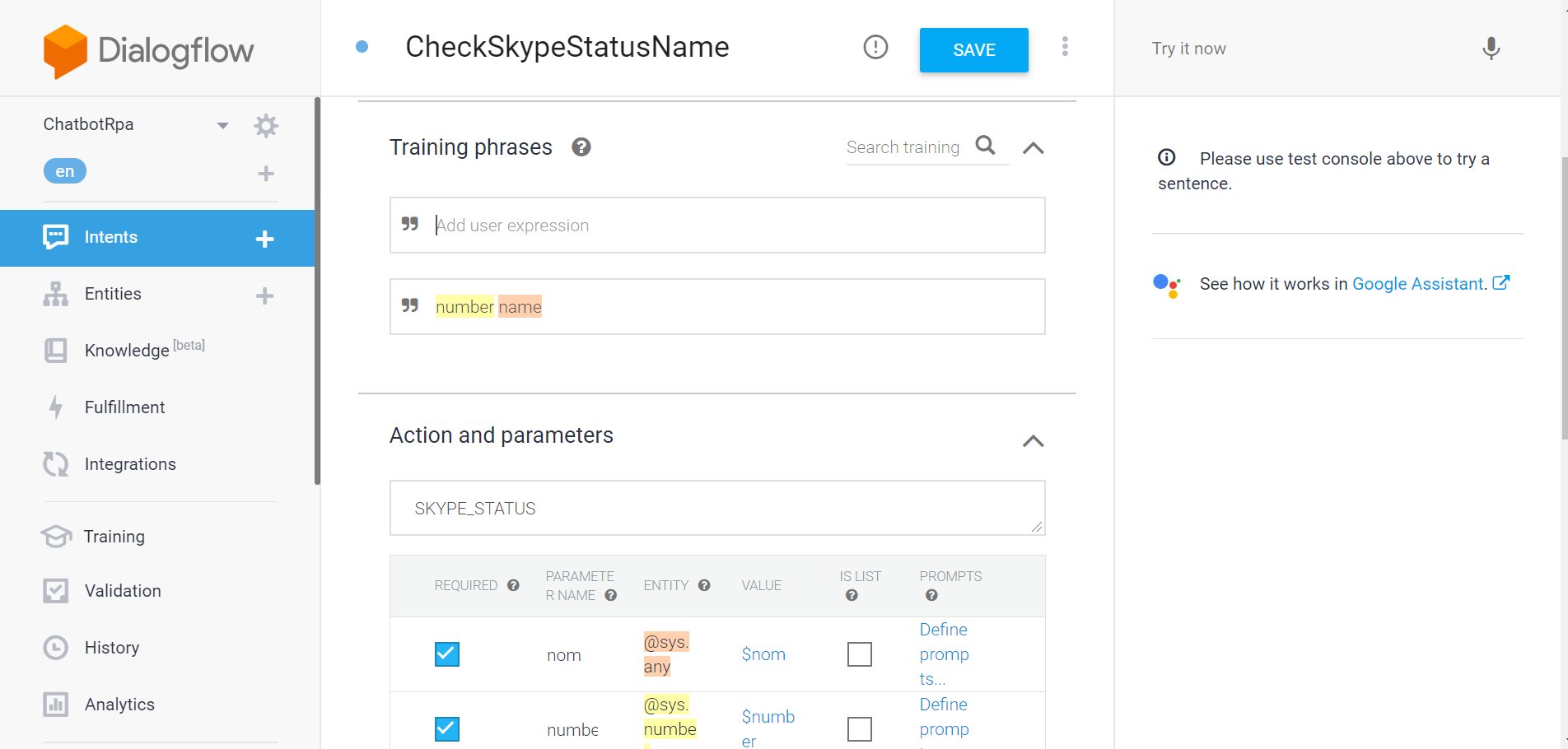


Figure 10 : Vue sur un intent sous DialoFlow

IV. Interaction Chatbot/RPA

Pour mettre en place notre projet et concrétiser notre idée de chatbot assisté par le RPA, nous avons pensé à une solution qui pourra nous permettre de faire interagir les deux d’une manière performante, et vu que DialogFlow offre la possibilité d’intégrer l’agent conversationnel à travers différentes librairies clientes avec un choix libre du langage de programmation, comme le montre la figure ci-dessous :



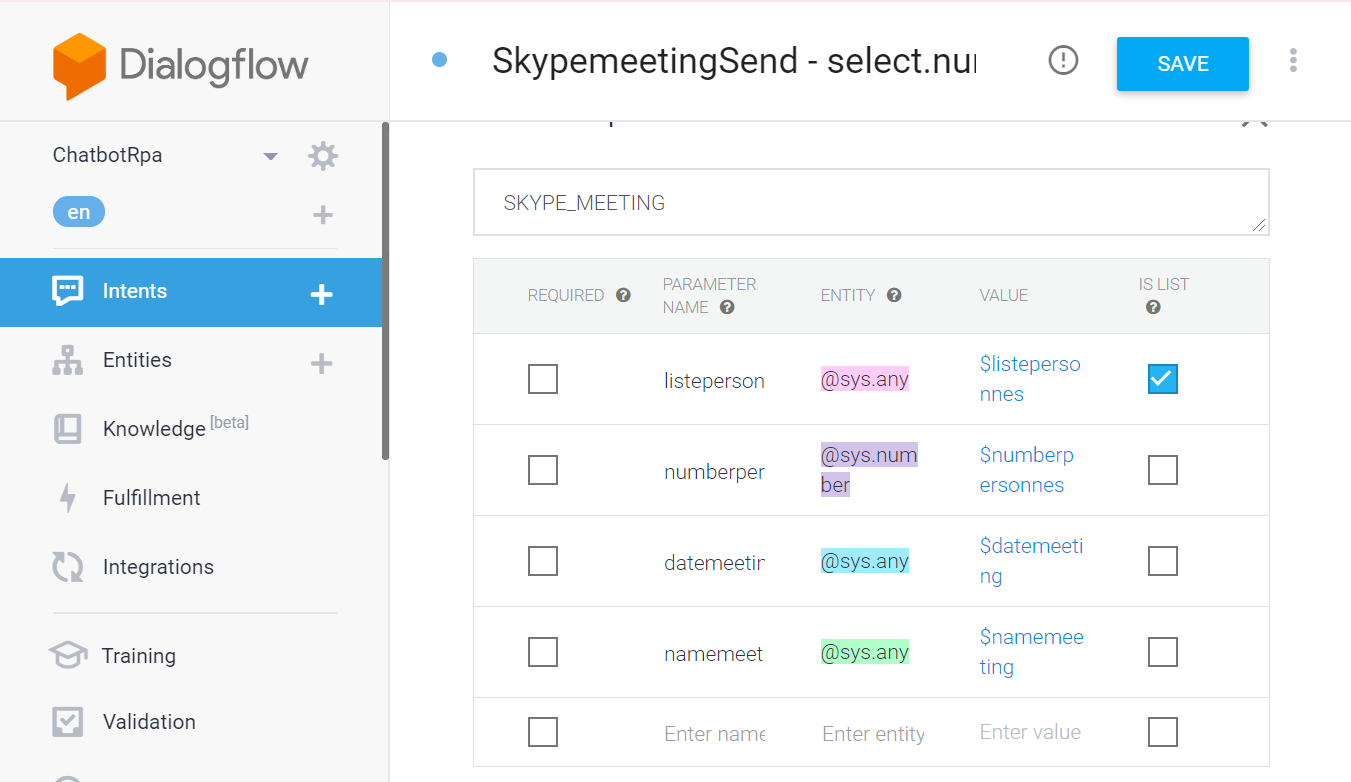
Figure 11 : Langages de programmation supporté par les librarires de DialogFlow

On a pensé à faire appel à l’une des SDK que DialogFlow offre et spécifiquement Node.js.

IV.1. Choix de Node.Js

Node.Js offre la possibilité d’utiliser des bibliothèques JavaScript orientée qui vont nous permettre de communiquer avec DilogFlow (dialogflow.js) et l’orchestrateur (orchestrator.js), ce sont des librairies gratuites et qui permettront d’envoyer des requêtes à DialogFlow et à l’orchestrateur.

NodeJs envoie une requête à DialogFlow pour récupération de la réponse de l’intent déclenché, chaque intent a des intents de suivi, qui se succèdent pour arriver à la fin au dernier intent qui est responsable du déclenchement de l’action qui portent tous les paramètres nécessaires pour exécution du processus.



Les paramètres nécessaires pour l’exécution du processus

Le nom de l’action

Figure 12 : Action et paramètres - DialogFlow

Le champ ***action*** est destiné à faciliter l'exécution de la logique de notre service pour plus de commodité.

Lorsqu'un intent est mis en correspondance au moment de l'exécution, Dialogflow fournit la valeur de l'action à la requête de NodeJs, ainsi que les valeurs extraites de l'expression de l'utilisateur final en tant que **paramètres**. Cela permet de déclencher la logique spécifique dans notre service.

Au niveau de NodeJs, lors de la réception de la réponse de requête, un traitement permet d’identifier l’action reçue et de récupérer les paramètres passés, pour faire appel à la fonction qui permettra l’ajout d’un élément dans la file d’attente de l’orchestrateur. Jetons un coup d’œil sur le code derrière ce traitement :

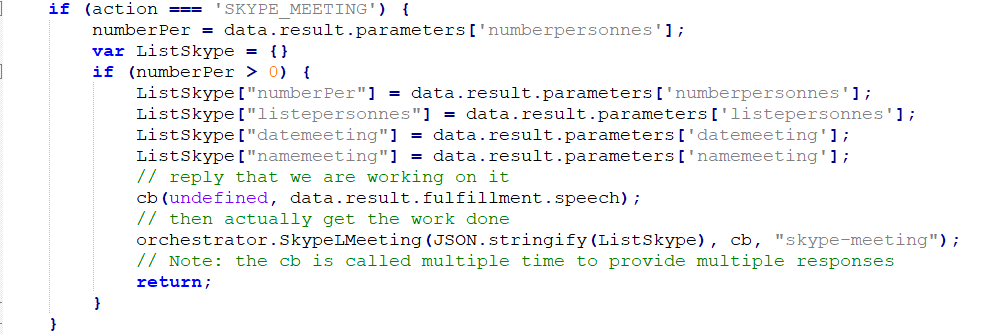


Figure 13 : Vue sur le code du traitement de la réponse reçue de chez DialogFlow

On remarque que les noms des paramètres sont les même sur DialogFlow. On rassemble ces paramètres dans une liste et on les convertit en objet Json (JSON.Stringify).

C’est à cette phase que le module de l’orchestrateur intervient pour alimenter la file d’attente de l’orchestrateur.

Tout d’abord expliquons ce que c’est une file d’attente de l’orchestrateur, aussi nommé « queue ».

La « queue » est un conteneur qui permet de contenir un nombre illimité d'éléments, qu’on pourra modifier leurs statuts et aussi les traiter en utilisant des activités de Uipath Studio. Dès que les éléments de la file d'attente sont traités, ils deviennent des transactions.

Voici à quoi ressemblent les files d’attente dans l’orchestrateur :

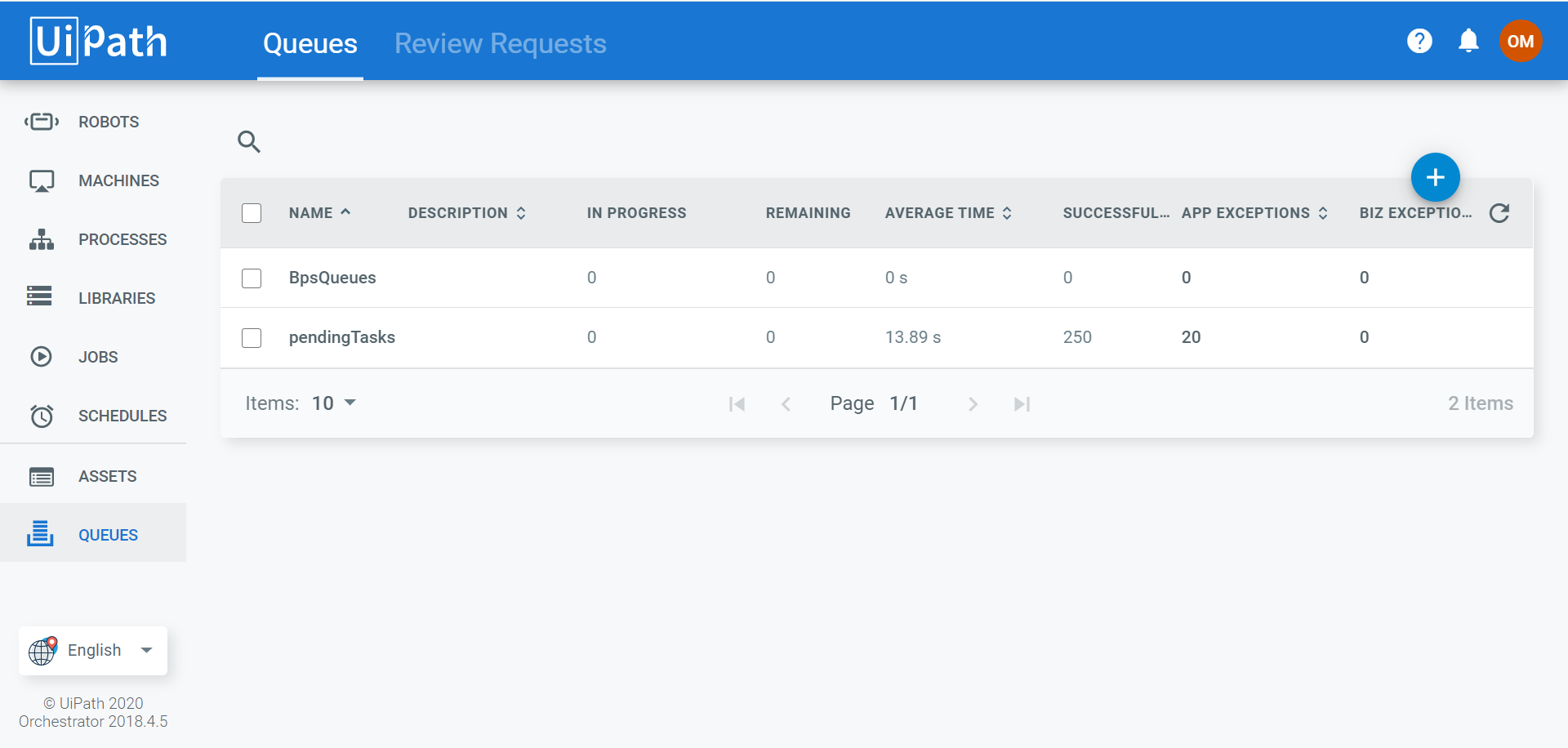
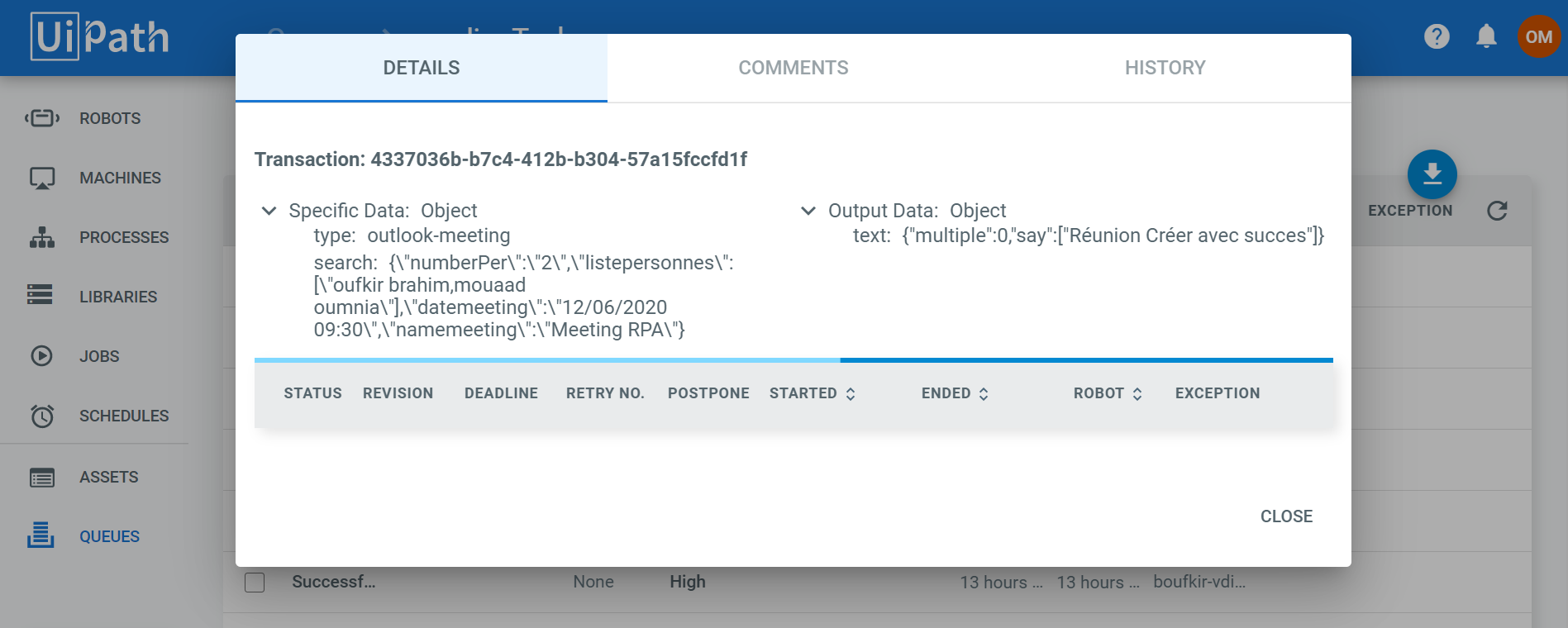


Figure 14 : Vue sur les files d'attente dans l'orchestrateur

La page **Files d'attente (Queues)** permet de créer des files d'attente. Elle fournit également un accès en consultation aux files d'attente auparavant créées, aux graphiques avec la progression de l'état des transactions dans le temps et à d'autres détails divers, tels que le délai d'exécution moyen et le nombre total de transactions réussies.

Exemple de transaction :



L’objet envoyé depuis nodeJs, il contient le type et le paramètre (il peut être un seul ou une liste de paramètres)

Figure 15 : Vue sur les détails d'une transaction

L’objet retourné (output) après fin d’exécution du robot invoqué. On le stocke sous format d’objet *Json* qui désigne la réponse envoyée par le chatbot.

V. Spécifications des besoins

V.1. 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 application est répartie sur 4 principaux axes. Chaque axe possède ses propres besoins fonctionnels.

|  |  |
| --- | --- |
| Axe | Fonctionnalités |
| Skype | * Organiser un skype meeting * Vérifier le statut d’un collaborateur * Envoyer un message sans devoir être connecté à skype. |
| Outlook | * Organiser un Teams meeting * Envoyer un mail |
| Redmine | * Créer un ticket sur redmine * Modifier un ticket * Recherche un ticket suivant des filtres |

Tableau 2 : Table des fonctionnalités

V.2. Besoins non fonctionnels

Les besoins non fonctionnels sont importants car ils agissent de façon indirecte sur le résultat et sur le rendement de l’utilisateur d’où leurs importances.

Pour cela il faut répondre aux exigences suivantes :

* Fiabilité : l’application doit fonctionner de façon cohérente sans erreurs.
* Gestion des erreurs : l’application doit les signaler par des messages d’erreurs.
* Ergonomie et bon IHM : l’application doit être adaptée à l’utilisateur sans qu’il fournisse trop d’effort (utilisation claire et facile).
* Efficacité : l’application doit permettre l’accomplissement de la tâche avec le minimum de manipulations.
* Sécurité : l’application doit être sécurisée au niveau des données.
* Convivialité : l’interface utilisateur doit être conviviale pour assurer un accès aisé aux données.

V.3. Diagrammes de cas d’utilisation

V.3.1. Présentation des acteurs

Un acteur est une abstraction extérieure au système à modéliser (l’organisation lorsque l’on modélise le métier, le système informatique dans un second temps) qui interagit avec lui. Il s’agit de rôles joués par des personnes, de logiciels, de matériels, etc.

Les acteurs en interaction avec notre système sont :

|  |  |
| --- | --- |
| Acteur | Description |
| Utilisateur | L’un des collaborateurs Deloitte |
| UiPath Orchestrator | Permet (dans ce projet) d’envoyer des paramètres au robot |
| UiPath Robot | En se basant sur les paramètres reçus, lance un processus UiPath |
| DialogFlow Connector | Construit les intents/entities et fait la liason entre eux |
| DialogFlow Runtime | Transmet l’action et retourne la réponse |
| Messaging Channel | Le pont entre le chatbot, l’orchestrateur et DialogFlow Runtime |

Tableau 3 : Présentation des acteurs

V.3.2. Diagramme de cas d’utilisation général

Chaque usage que les acteurs font du système est représenté par un cas d’utilisation. Chaque cas d’utilisation représente une fonctionnalité qui leurs est offerte afin de produire le résultat attendu. Ainsi, le diagramme de cas d’utilisation décrit l’interaction entre le système et l’acteur. Il détermine les besoins de l’utilisateur et ce que doit faire le système pour l’acteur.

Nous présentons, dans la figure N°4, le diagramme de cas d’utilisation global de notre chatbot :

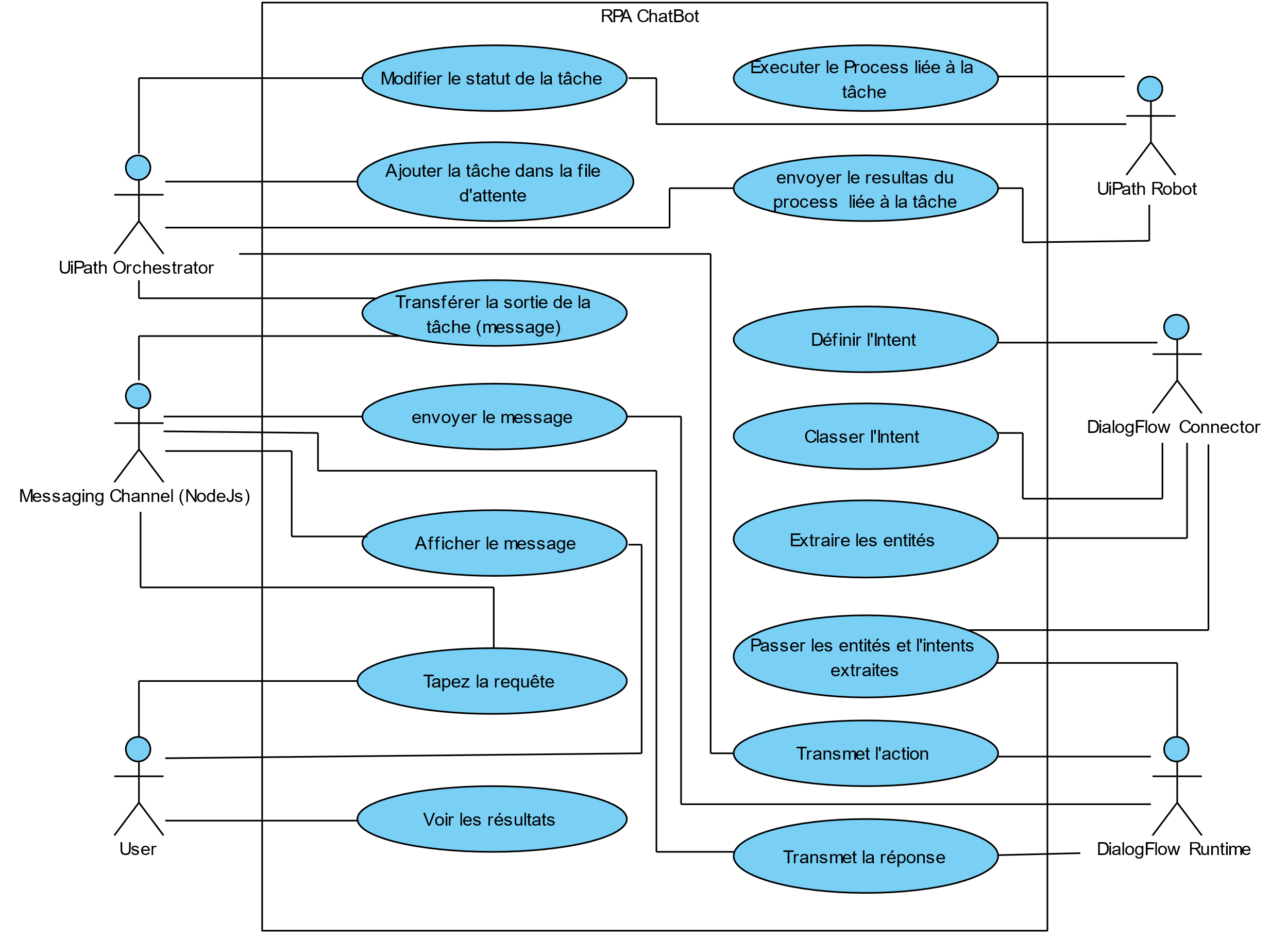


Figure 16 : Diagramme de cas d'utilisation général

Le diagramme ci-dessus montre le fonctionnement de notre travail. L’agent ***User*** va taper une requête ou choisir l’une des propositions du chatbot, par exemple organiser une réunion, créer un ticket, remplir la feuille de temps... Derrière cette action on trouve l’acteur ***Messaging Channel (NodeJs)*** qui joue le rôle d’un contrôleur, et qui va ensuite envoyer le message à ***DialogFlow Runtime.*** Cet agent collabore avec ***DialogFlow Connector*** *pour extraire l’intent correspondent* au message envoyé , ensuite après définition de l’intent, *DialogFlow Runtime* soit transmet la réponse à *Messaging Channel* ou déclenche une action (qui va permettre de lancer le traitement automatique), celui-ci (*Messaging Channel*) va interpréter la réponse reçue et soit il  renverra une réponse ou il ajoutera un élément dans la file d’attente de ***Uipath Orchestrator,*** qui va alors stocker la transaction portante du type du processus et aussi les paramètres, pour qu’elle soit repérée par ***Uipath Robot*** (le robot qui englobe tous les processus), pour qu’il exécute le traitement lié à la tâche demandée suivant le type de la transaction, finalement le robot va retourner la réponse du traitement, il stockera le résultat dans la transaction de *l’orchestrateur* ensuite le *Messaging Channel* va lire le message retourné par l’orchestrateur et le reformuler pour l’afficher à l’utilisateur.

V.3.3. Principaux diagrammes de cas d’utilisation

Dans ce paragraphe, nous allons présenter quelques diagrammes de cas d’utilisation ainsi que leurs descriptions textuelles. Les autres diagrammes sont pareils à ceux que nous allons présenter dans ce qui suit.

1. Diagramme de cas d’utilisation « Services Skype »

Comme il est indiqué dans la figure suivante N° 17, l’utilisateur en choisissant le service Skype, il pourra soit créer une réunion Skype, vérifier le statut d’un contact, ou Envoyer un message.

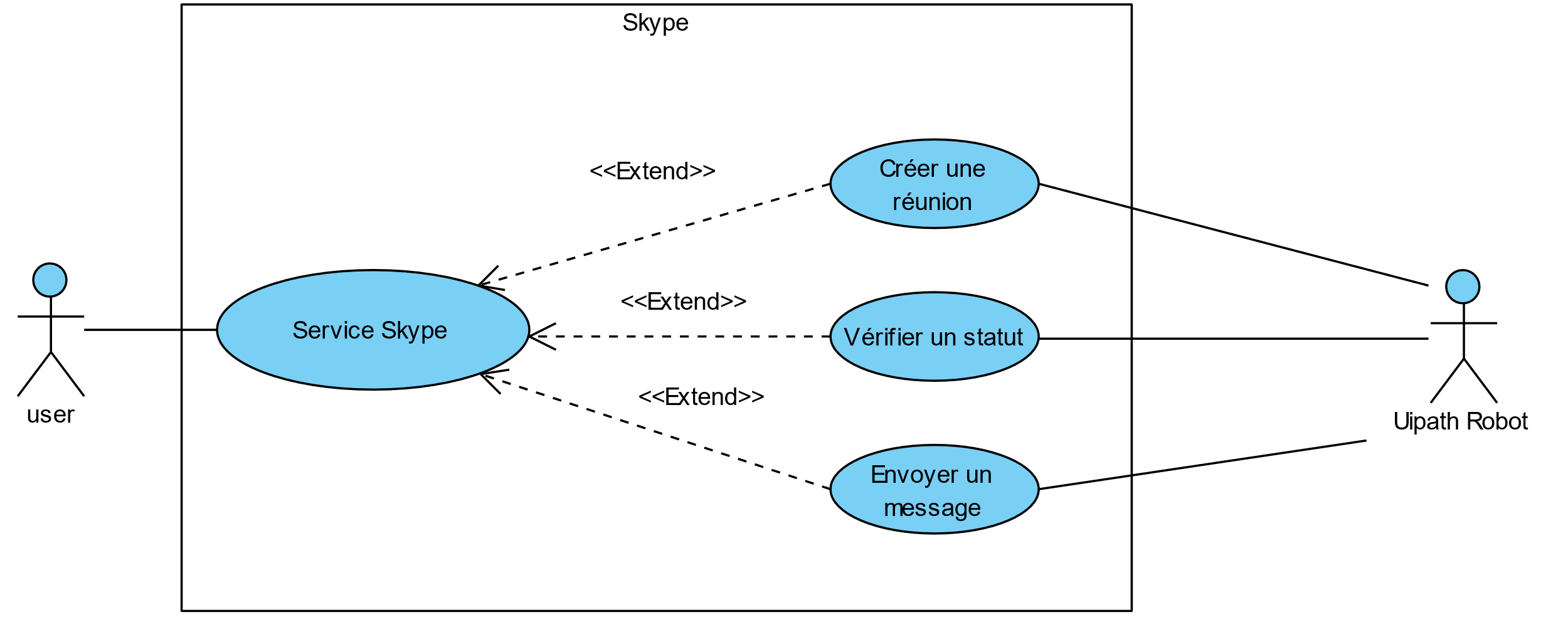


Figure 17 : Diagramme de cas d'utilisation « Services Skype »

* Description textuelle du cas d’utilisation « Créer une réunion Skype »

|  |  |
| --- | --- |
| Cas d’utilisation | Créer un réunion. |
| But | Organiser une réunion Skype. |
| Acteur | L’utilisateur. |
| Pré conditions | L’utilisateur doit soit choisir « Skype » 🡪 « Skype Meeting » dans les premières suggestions du chatbot ou tapez le message « Skype meeting» |
| Scénario nominal | * L’utilisateur choisit le service de création d’une réunion skype. * Le chatbot demande à l’utilisateur de renseigner les participants, le titre et la date/heure de la réunion. * Après la saisie,le Système fait le traitement et envoie les informations à Uipath orchestrator pour lancer le robot. * Le robot Uipath retourne la réponse au système et l’affiche à l’utilisateur. |
| Extension | L’utilisateur peut confirmer ou annuler la création de la rénuion. |

Tableau 4 : Description textuelle du cas « Créer une réunion Skype »

1. Diagramme de cas d’utilisation « Services Redmine »

Les services Redmine ainsi que toutes ses tâches sont illustrés par le diagramme de cas d’utilisation suivant :

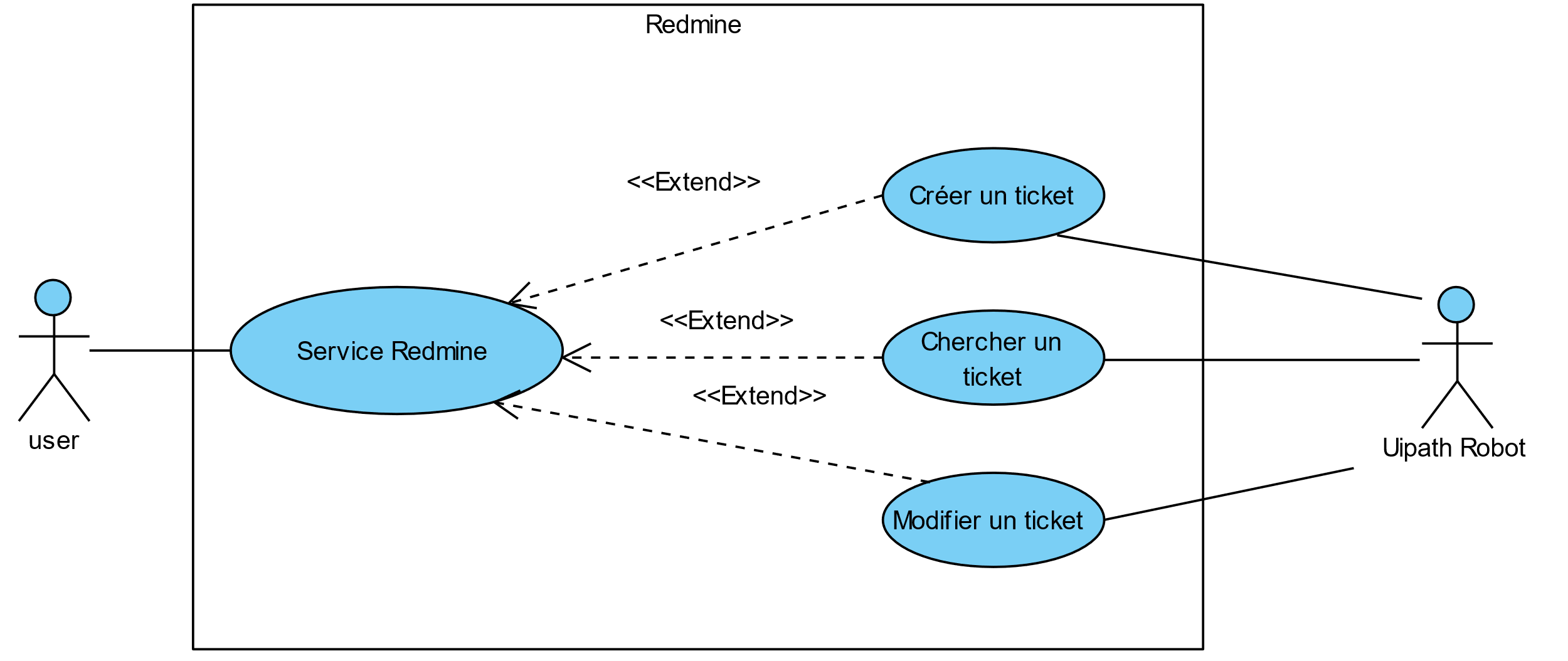


Figure 18 : Diagramme de cas d'utilisation « Services Redmine »

* Description textuelle du cas d’utilisation « Chercher un ticket Redmine »

|  |  |
| --- | --- |
| Cas d’utilisation | Chercher un ticket Redmine |
| But | Chercher un ticket Redmine par filtres |
| Acteur | L’utilisateur |
| Pré conditions | L’utilisateur doit soit choisir « Redmine » 🡪 « Rechercher ticket » dans les premières suggestions du chatbot ou tapez le message «  Rechercher ticket » |
| Scénario nominal | * L’utilisateur choisit le service de recherche d’un ticket Redmine. * Le chatbot demande à l’utilisateur de renseigner le filtre,et la valeur à chercher. * Après la saisie,le Système fait le traitement et envoie les informations à Uipath orchestrator pour lancer le robot. * Le robot Uipath retourne la réponse au système et ce dernier l’affiche sous forme d’un tableau à l’utilisateur. |
| Extension | L’utilisateur à la possibilté de choisir l’un des filtres proposés pour effectuer sa recherche. |

Tableau 5 : Description textuelle du cas « Chercher un ticket Redmine »

1. Diagramme de cas d’utilisation « Services outlook »

D’après la figure N°7, L’utilisateur peut effectuer des tâches sous Outlook après avoir choisi l’option Outlook. En effet, il peut créer une réunion sur Microsoft Teams ou envoyer un mail :

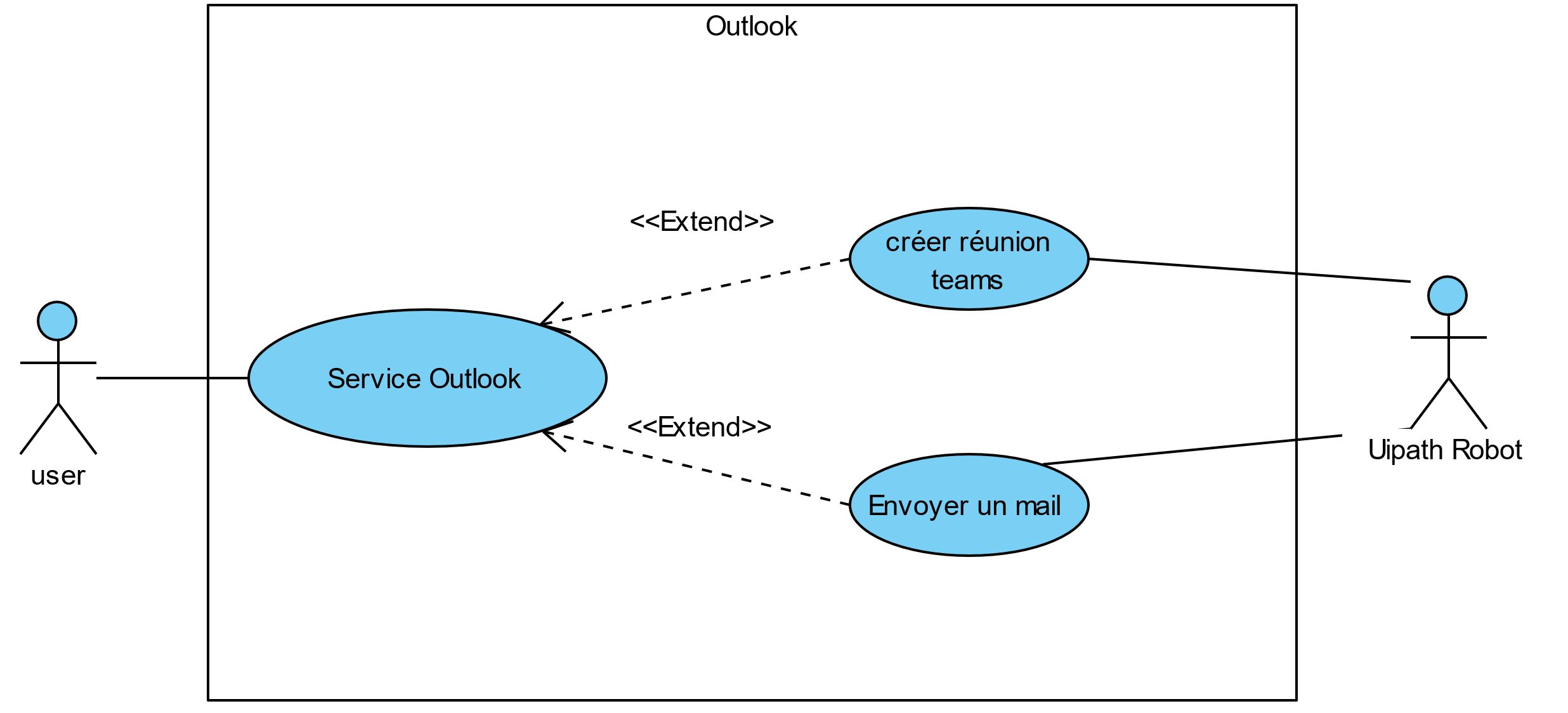


Figure 19 : Diagramme de cas d'utilisation « Services Outlook »

* Description textuelle du cas d’utilisation « Envoyer un mail »

|  |  |
| --- | --- |
| Cas d’utilisation | Envoyer un mail |
| But | Envoyer un mail à un contact |
| Acteur | L’utilisateur |
| Pré conditions | L’utilisateur doit soit choisir « Outlook » 🡪 « Envoyer mail » dans les premières suggestions du chatbot ou tapez le message «  Envoyer mail » |
| Scénario nominal | * Le système affiche des suggestions du service Outlook. * L’utilisateur choisit l’une des suggestions et saisit les informations demandées par le chatbot. * Le Système fait le traitement et envoie les informations à Uipath orchestrator pour lancer le robot. * Le robot retourne la réponse au système et l’affiche à l’utilisateur. |

Tableau 6 : Description textuelle du cas « Envoyer un mail »

Conclusion

La phase d’étude préalable nous a permis d’introduire notre projet et d’expliquer d’une façon générale le fonctionnement de notre chatbot, ainsi de mettre en valeur l’interaction entre l’agent conversationnel et l’RPA, et finalement déterminer les principales fonctionnalités de notre système à travers la présentation des besoins fonctionnels et non fonctionnels. Nous avons réussi à élaborer les diagrammes de cas d’utilisation relatifs aux besoins spécifiés et nous avons présenté quelques diagrammes de séquence afin de mieux comprendre le fonctionnement du futur système. Tout ce travail nous a permis de passer à la phase la plus importante dans notre travail : la conception.

CHAPITRE III : Conception détaillée

Après l’étape de l’étude préalable et la spécification des besoins, nous réservons cette partie à la conception détaillée de notre système. En effet, au cours de ce chapitre nous allons décrire les différentes étapes de conception.

1. Conception architecturale

IV.1. Architecture adoptée

La Figure N°17 illustre l’architecture adoptée pour notre chatbot. Tout d’abord, notre projet sera défalqué en quatre modules :

* NodeJs Server

**c**’est le coordinateur entre l’interface du chatbot**,** DialogFlow et l’orchestrateur, il communique avec les 3 modules pour envoyer les requêtes qui permettent l’envoie et réception des messages.

* Moteur de traitement de langage naturel (NLP)

Extrait les intents et retourne la réponse adaptée ou déclenche l’action portante des paramètres nécessaires à une logique précise.

* Uipath Orchestrato

Son rôle est de stocker les paramètres reçus depuis NodeJs dans une file d’attente, pour qu’ils soient repérés par Uipath Robot.

* Uipath Robot

C’est le robot qui rassemble les processus qui seront executés au lancement d’une action,il est toujours en écoute, et en attente que la queue de l’orchestrateur s’alimente, dès qu’il détecte une transaction dans la file d’attente, il récupère les paramètres et lance le processus convenable.

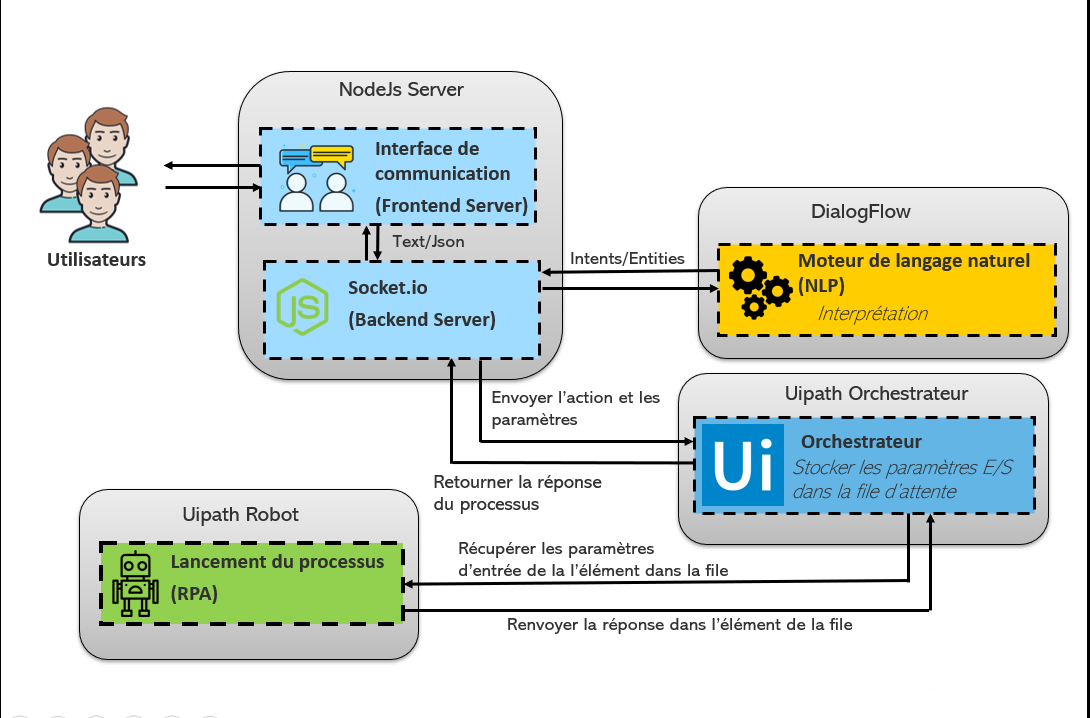


Figure 20 : Architecture globale du Chatbot

IV.1. Architecture logicielle

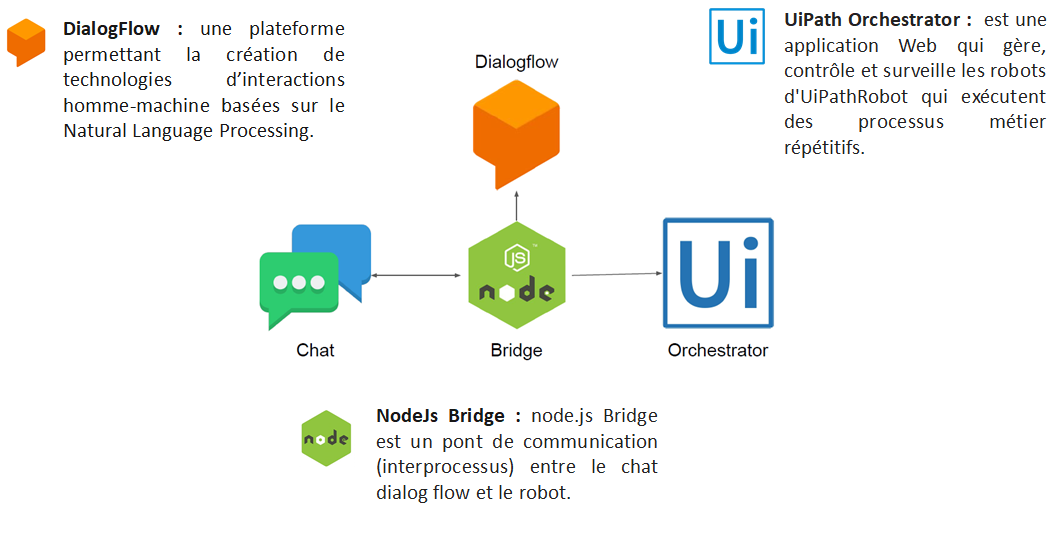


Figure 21 : Architecture logicielle du chatbot

V. Conception aspect dynamique

V.1. Diagrammes de séquence

Le diagramme de séquence met en valeur les échanges de messages entre acteurs et système de manière chronologique, Il modélise donc à la fois des activités (fonctionnalités) et des communications (interactions) pour les entités concernées.

Dans ce qui suit, nous allons présenter le diagramme de séquence général ainsi que les digrammes de quelques scénarios de notre chatbot.

V.1.2. Diagramme de séquence général

Ci-dessous (Figure N°22), le diagramme de séquence général de notre Chatbot, on a présenté en détails le flux de données qui circulent entre les agents de notre système :

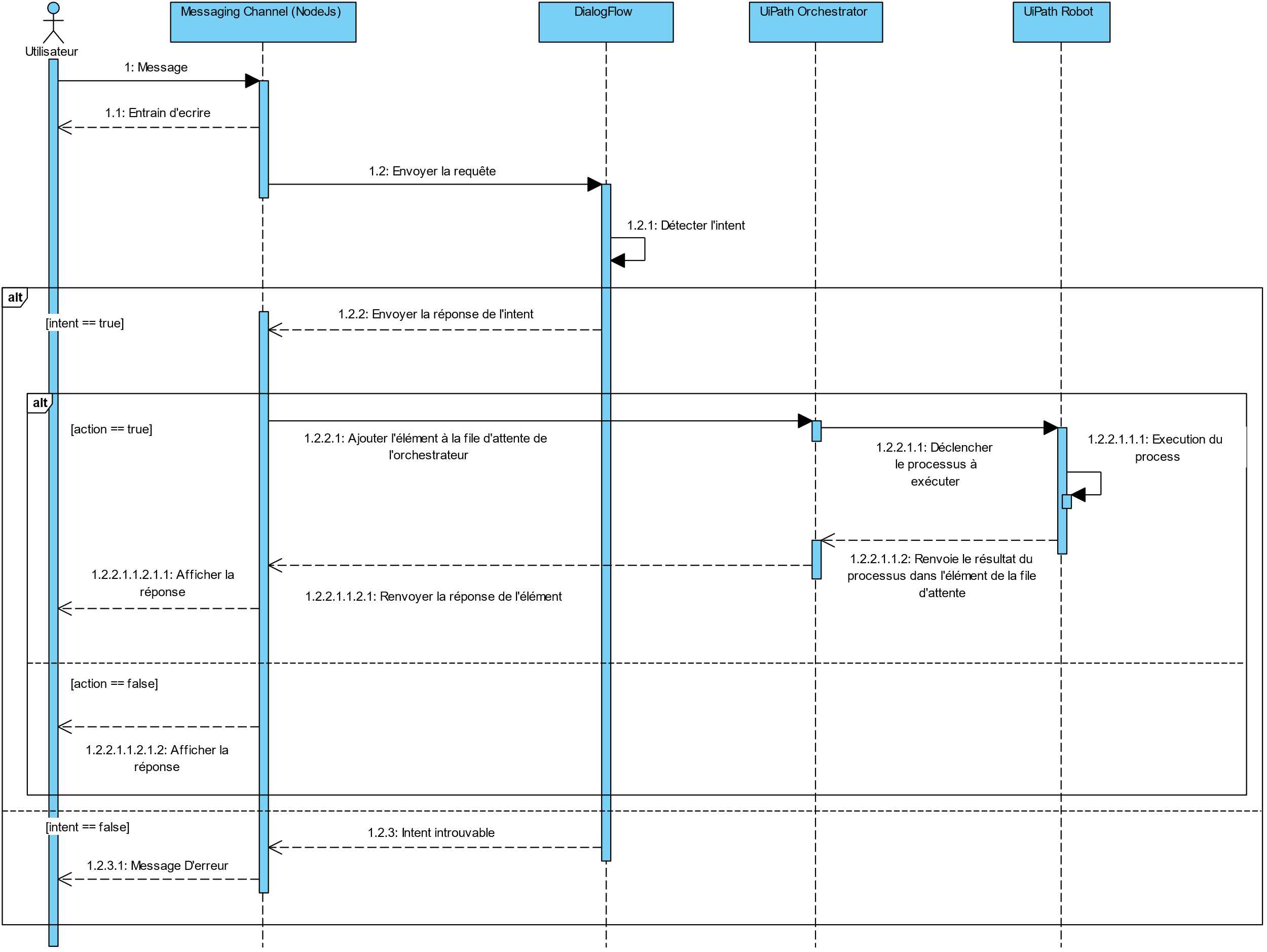


Figure 22 : Diagramme de séquence général

V.1.2. Principaux Diagrammes de séquence

Dans ce qui suit, nous allons présenter les diagrammes de séquences pour quelques scénarios

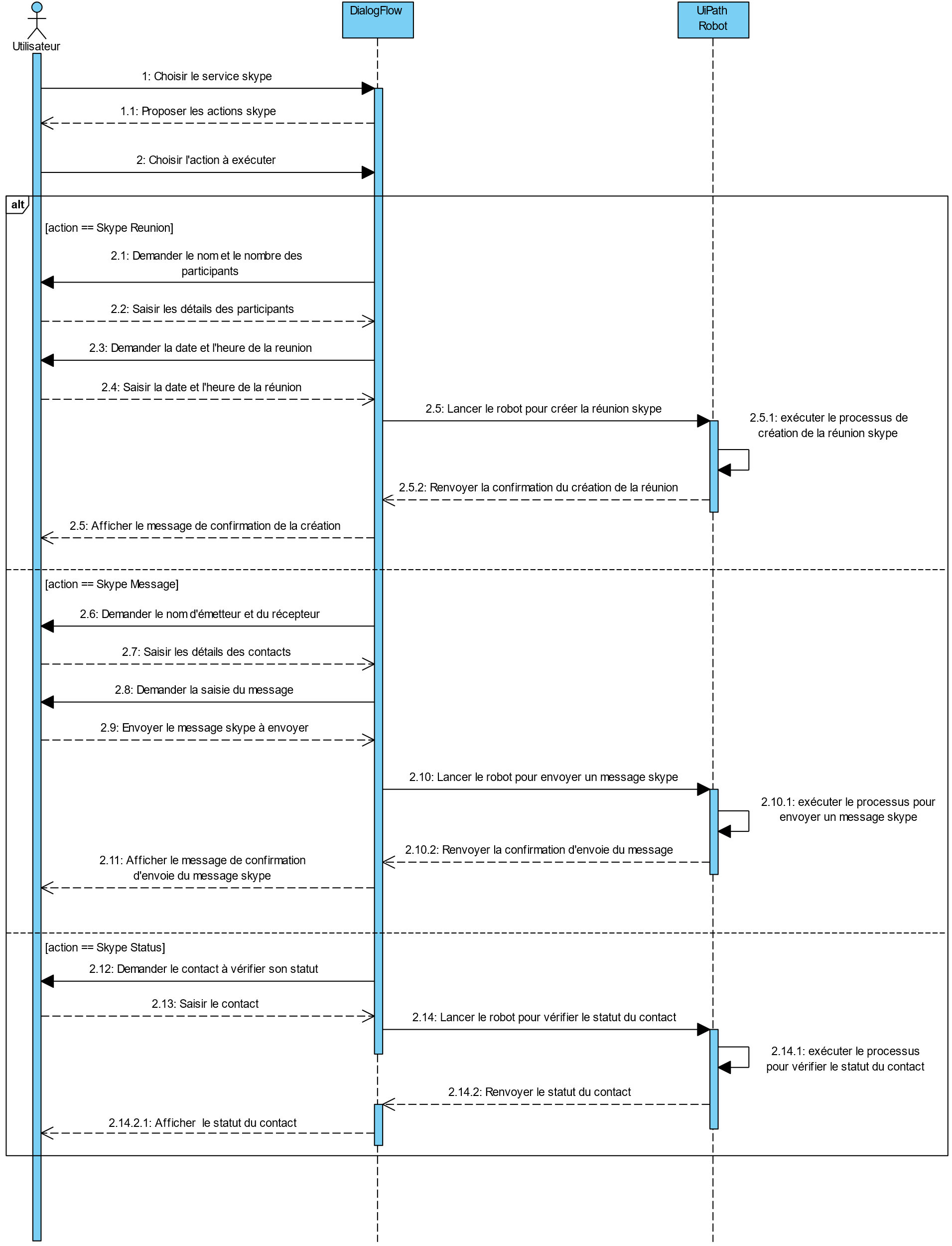


Figure 23 : Diagramme de séquence « Services Skype »

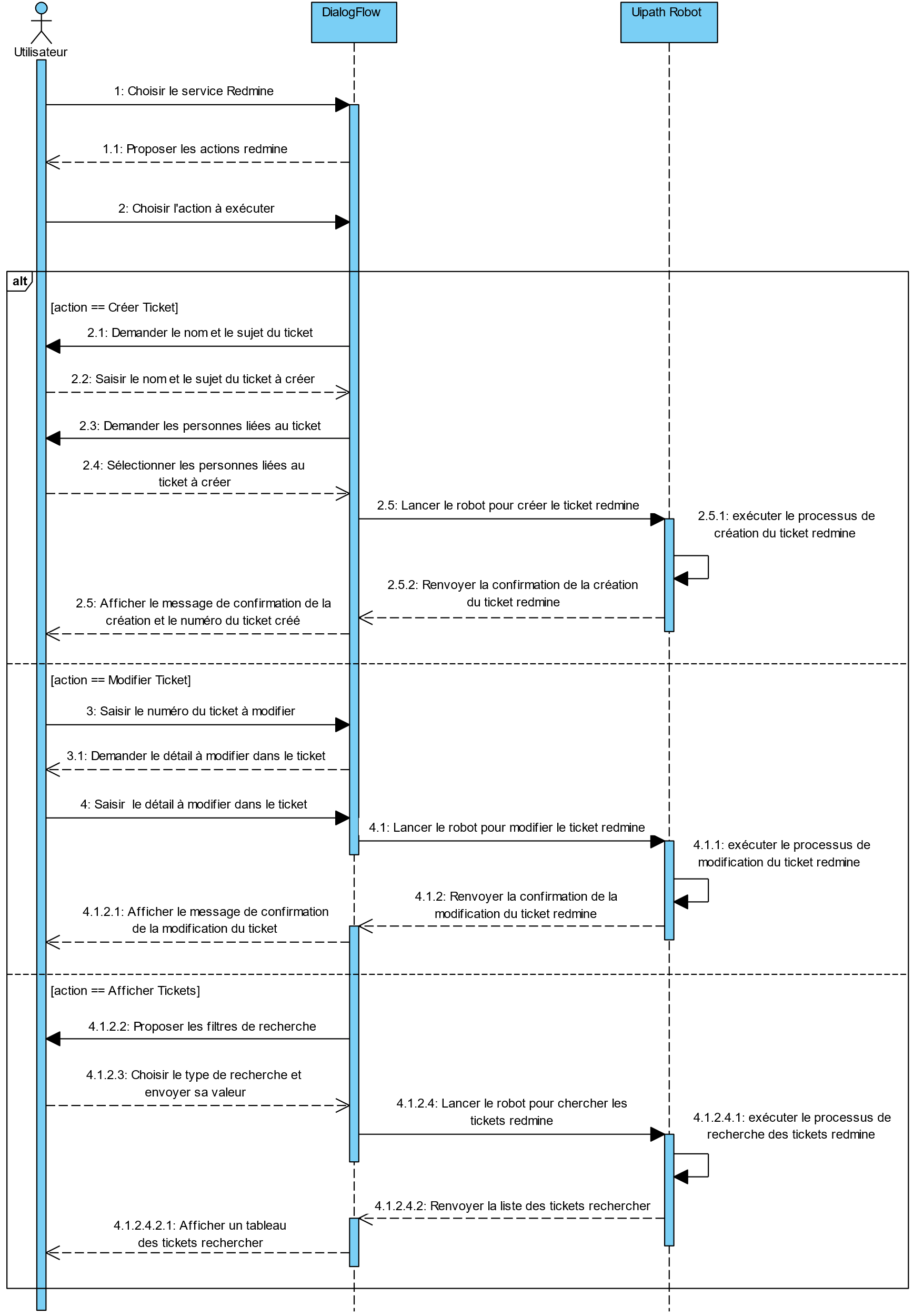


Figure 24 : Diagramme de séquence « Services Redmine »

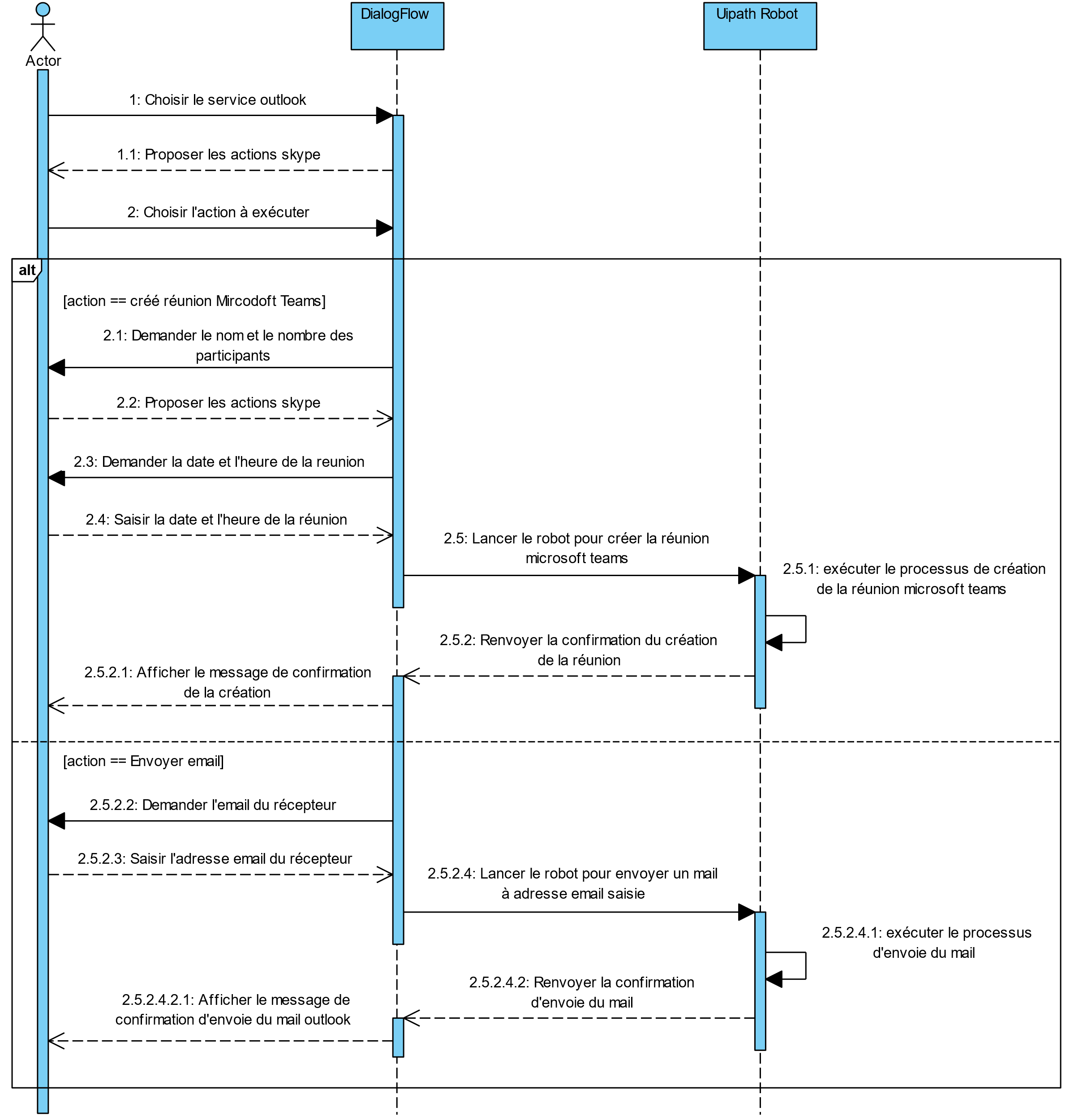


Figure 25 : Digramme de séquence « Services Outlook »

V.2. Diagrammes d’activité

Le diagramme d’activité représente la dynamique du système. Il montre l’enchaînement des activités d’un système ou même d’une opération. IL représente le flot de contrôle qui retrace le fil d’exécution et qui transite d’une activité à l’autre dans le système. Il est le plus approprié pour modéliser la dynamique d’une une tâche, d’un cas d’utilisation lorsque le diagramme de classe n’est pas encore stabilisé. Nous présentons dans ce qui suit quelques diagrammes d’activité.

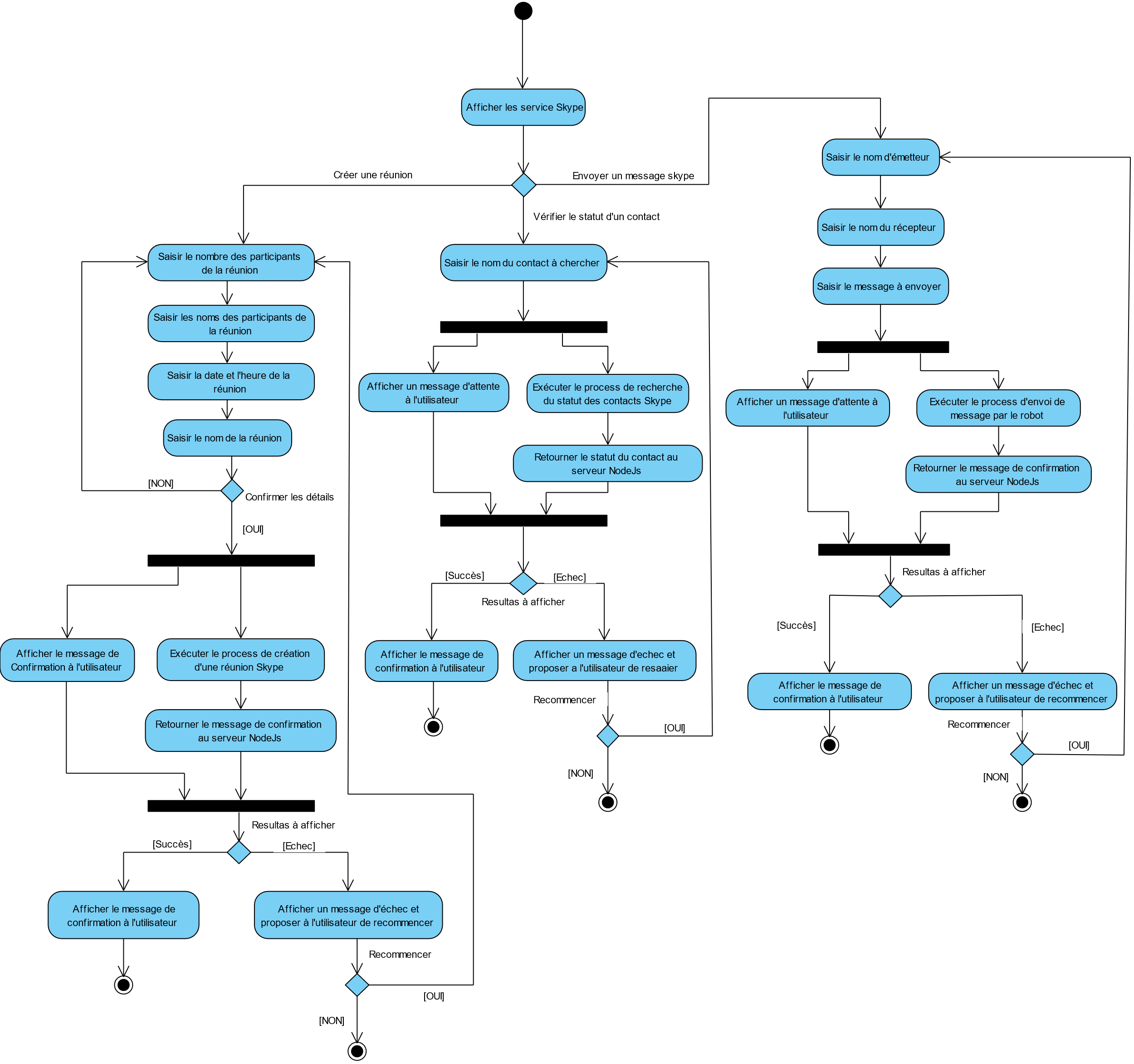


Figure 26 : Diagramme d’activité « services Skype «



Figure 27 : Diagramme d'activité « Service Redmine »

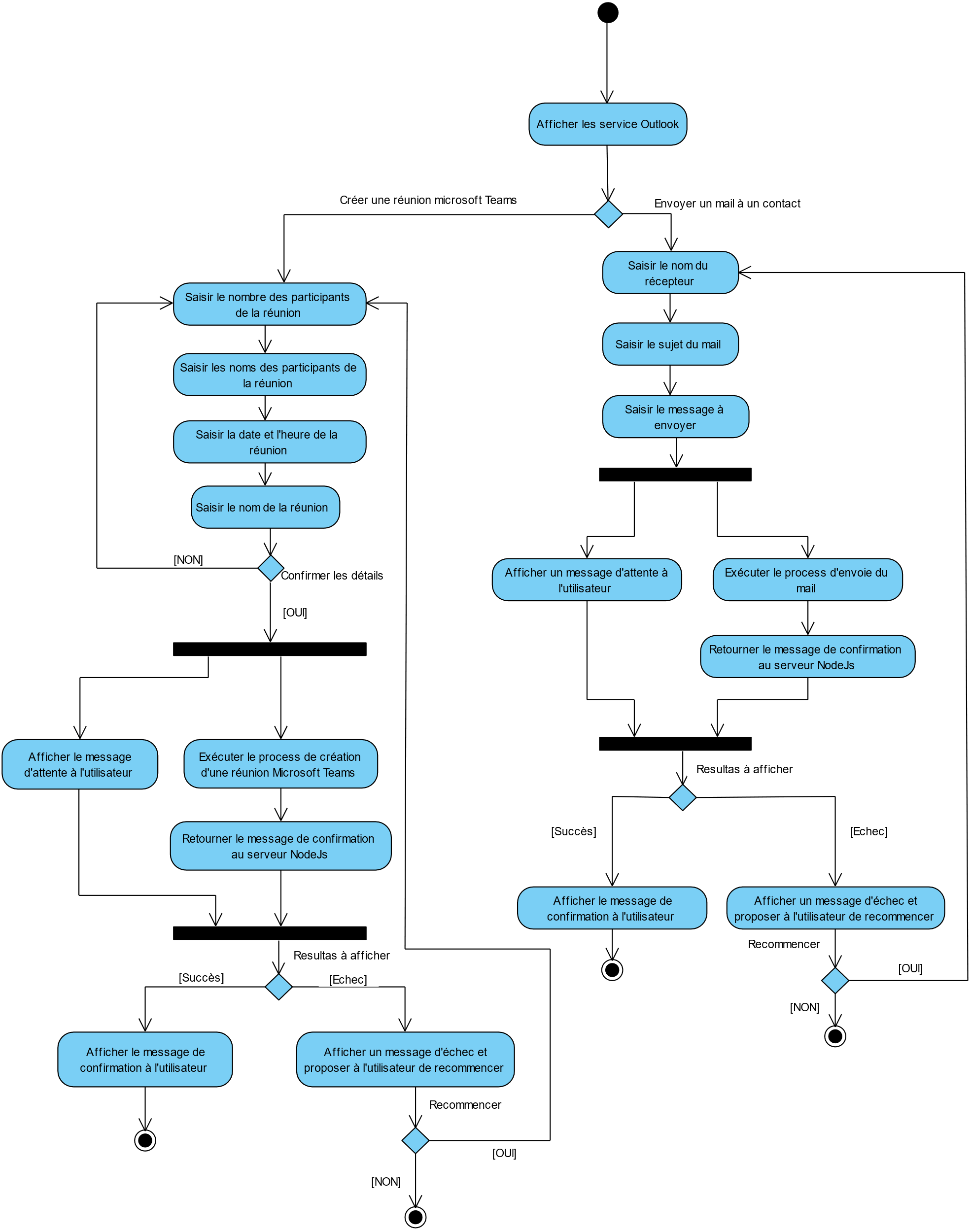


Figure 28 : Diagramme d'activité « Services Outlook »

Conclusion

Dans ce chapitre nous avons détaillé et expliqué le fonctionnement de notre chatbat et les différentes vues conceptuelles de l’application à réaliser. Cette conception est nécessaire à la phase réalisation qui fera l’objet du chapitre suivant.

CHAPITRE IV : Réalisation

Le chapitre suivant est dédié pour la présentation des outils et Frameworks utilisées pour le développement du produit soit technique ou de communication. Ce chapitre montra le résultat obtenu sous la forme des IHMS.

1. Environnement du travail

I.1. Environnement matériel

Afin de mener à bien ce projet, nous avons utilisé un ensemble de matériels dont les principales caractéristiques sont les suivantes :

### Machine locale

* Processeur: Intel(R) CoreTM i5-4310U CPU @2.00 GHZ 2.60 GHz
* RAM: 8.00 GO
* OS : Microsoft Windows 10.

### Machine à distance

* Processeur: Intel(R) CoreTM i5-4310U CPU @2.00 GHZ 2.60 GHz
* RAM: 8.00 GO
* OS: Microsoft Windows 10.

I.2. Environnement logiciel

I.2.1. Outils de développement

**a) Uipath Studio**

UiPath Studio - un outil avancé qui vous permet de concevoir des processus d'automatisation de manière visuelle, à travers des diagrammes. Il permet de modéliser visuellement les processus métier d'une organisation.

**b) Uipath Orchestrator**

UiPath Orchestrator est une application Web qui vous permet d'orchestrer vos robots UiPath lors de l'exécution de processus métier répétitifs. Orchestrator vous permet de gérer la création, la surveillance et le déploiement de ressources dans votre environnement. Il agit comme un point d'intégration avec des solutions et applications tierces. Sa puissance vient de sa capacité à gérer l'ensemble de votre flotte de robots.

**c) eclipse**

 Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

I.2.2. API et bibliothèques

**a) DialogFlow**

Dialogflow est une plate-forme de compréhension du langage naturel utilisée pour concevoir et intégrer une interface utilisateur conversationnelle dans des applications mobiles, des applications Web, des appareils, des robots, des systèmes de réponse vocale interactifs, etc.

I.2.3. Gestionnaire de dépendances

 **a) npm**

Npm est un gestionnaire de dépendances pour les projets JavaScript. Similaire à Gradle et Maven, il gère les dépendances d’un projet et son cycle de vie, en cherchant des packages depuis son répot cloud. Npm et définissant un ensemble des taches. Il nécessite l’installation du NodeJs sur la machine de développement.

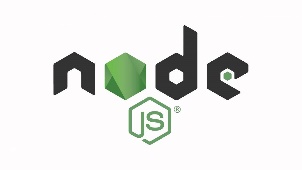
I.2.4. Serveurs Web

**a) IIS**

Internet Information Services (IIS) est un serveur Web polyvalent et flexible de Microsoft qui s'exécute sur les systèmes Windows pour servir les pages ou fichiers HTML demandés. Un serveur Web IIS accepte les demandes des ordinateurs clients distants et renvoie la réponse appropriée.

I.2.5. Frameworks

**a) NodeJs**

Node.js est une plateforme logicielle libre en JavaScript orientée vers les applications réseau événementielles hautement concurrentes qui doivent pouvoir monter en charge Elle utilise la machine virtuelle V8, la librairie libuv pour sa boucle d'évènements, et implémente sous licence MIT les spécifications CommonJS.

**b) Selenium**

Selenium est un framework de test informatique développé en Java.

Il permet d'interagir avec différents navigateurs web de même que le ferait un utilisateur de l'application. Il entre ainsi dans la catégorie des outils de test dynamique (à l'inverse des tests statiques qui ne nécessitent pas l'exécution du logiciel) facilitant le test fonctionnel.

III. Interaction avec les différentes interfaces graphiques

III.1. Interface d’accueil

Que ce soit en passant par la version Web ou la version Mobile, l’utilisateur recevra à sa première interaction avec l’interface, un Welcome message, ensuite le chatbot lui proposera ses différents services, l’utilisateur pourra choisir parmi ces services soit en cliquant dessus ou en écrivant son nom.



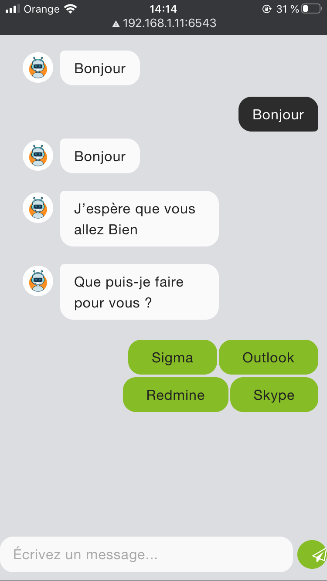


Figure 30 : Interface d'accueil (Web et mobile)

III.2. Interface Skype

En choisissant l’action Skype le chabot demande à l’utilisateur le processus qu’il veut exécuter en lui proposant 3 fonctionnalités Skype :

* Envoyer un message à un contact.
* Organiser une réunion Skype.
* Vérifier le statut actuel d’un contact.



Figure 31 : Liste des choix pour la fonctionnalité Skype

III.2.1 Envoyer un message à un contact

Ce service aide l’utilisateur à envoyer un message par Skype à un autre contact, en mentionnant son nom, le nom de récepteur et le message souhaité. Ce service va aider l’utilisateur dans le cas où il n’a pas accès direct à Skype et qu’il veut contacter un autre membre de la firme.

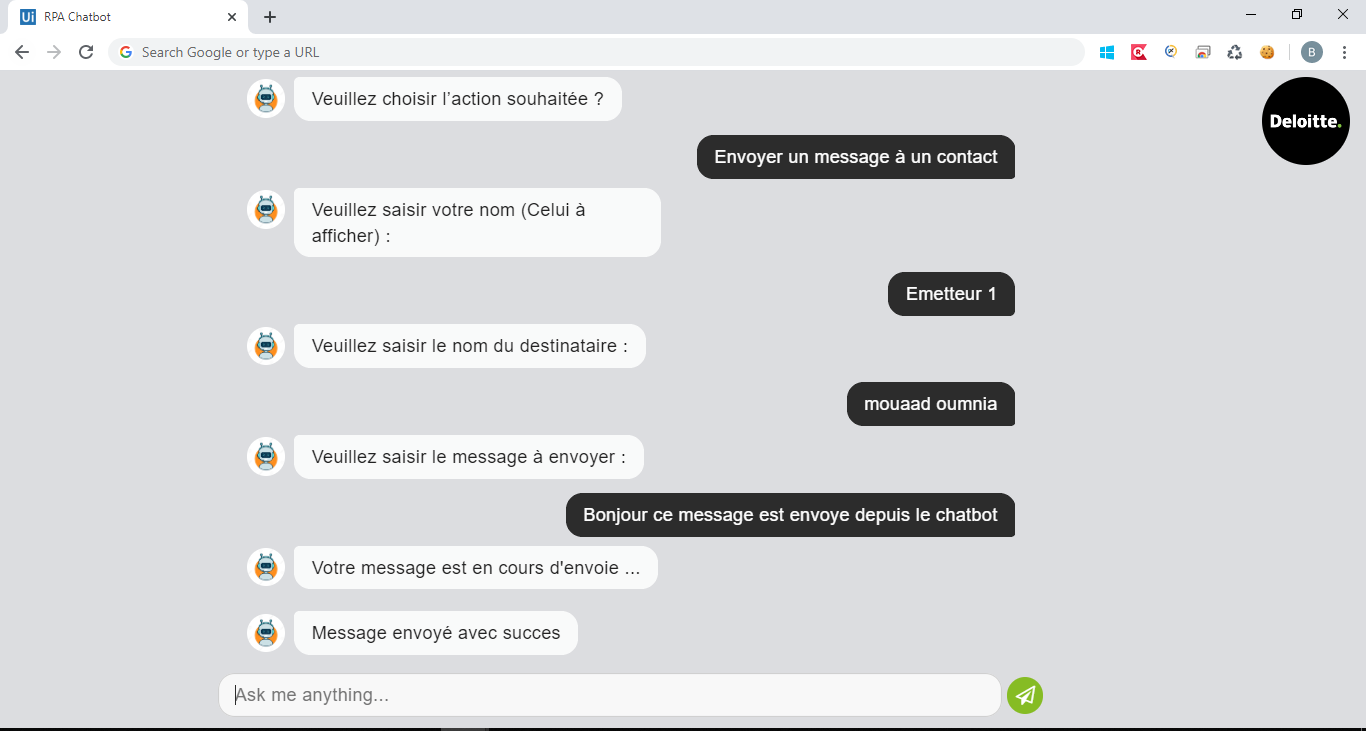
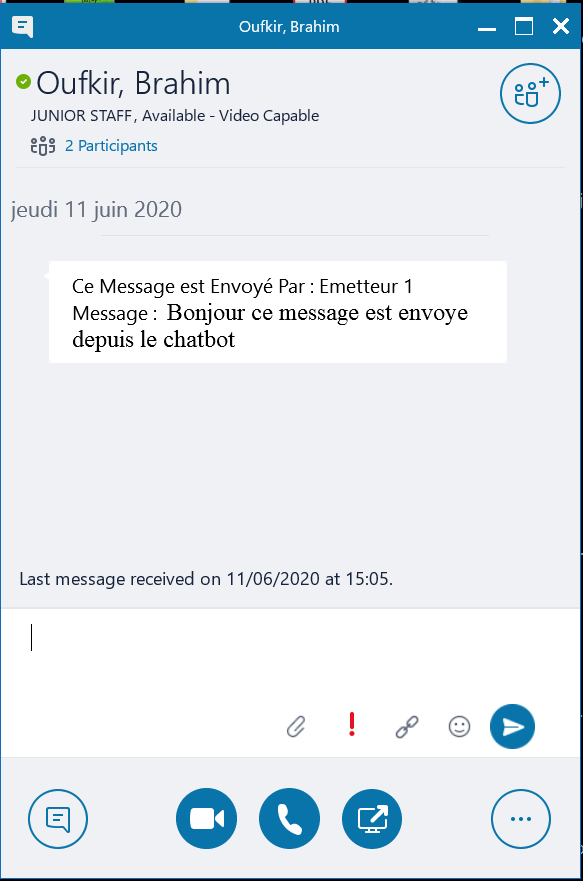


Figure 32 : Envoyer un message à un contact

III.2.2 Organiser une réunion Skype

Ce service aide l’utilisateur à planifier une réunion skype avec les participants qu’il va choisir tout en mentionnant la date, les noms des personnes invitées et le titre de la réunion.



L’utilisateur peut sélectionner la date et l’heure de la réunion depuis une liste déroulante :

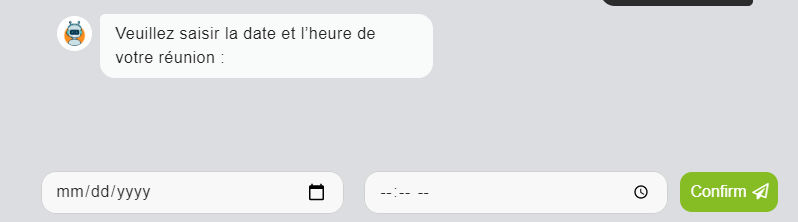


Figure 33 : Choix de la date de la réunion

Un récapitulatif est affiché à la fin de la saisie pour vérifier les données entrées par l’utilisateur

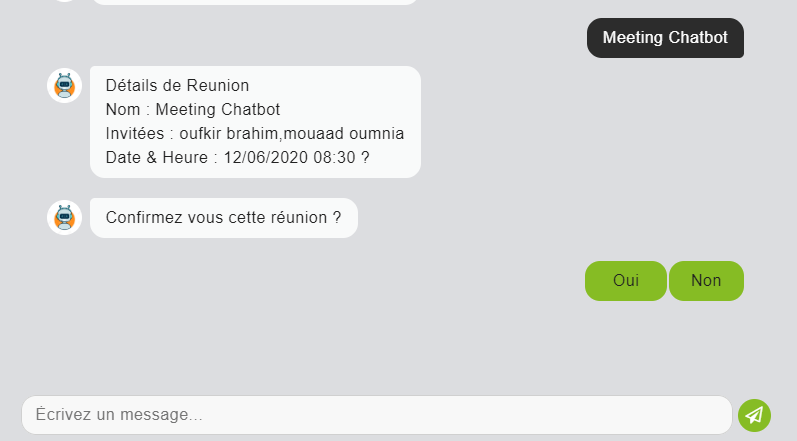


Figure 34 : Confirmation de la réunion

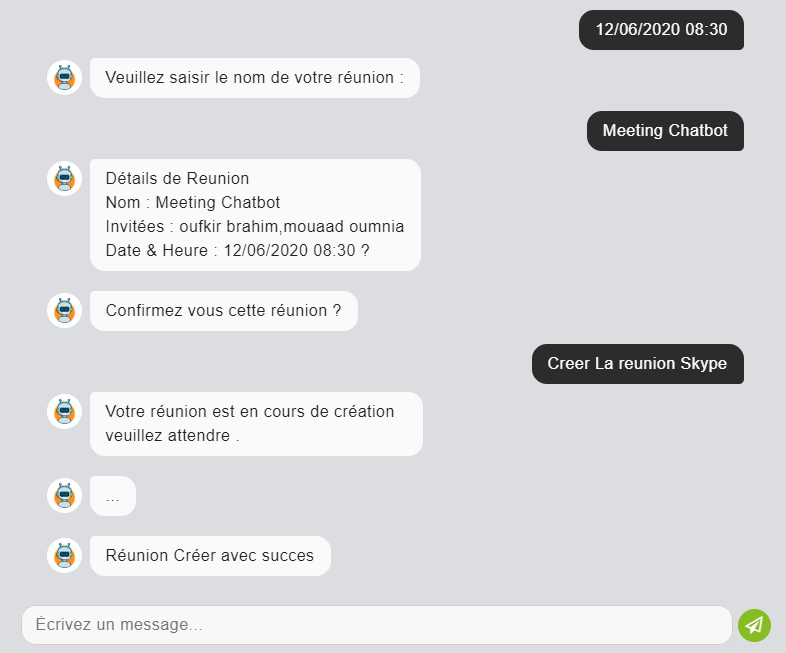
Un message de succès est affiché et l’utilisateur reçoit après l’invitation à la réunion Skype 

Figure 35 : Message de succès (réunion créée)

III.2.3 Vérifier le statut actuel d’un contact

L’utilisateur peut vérifier le statut actuel Skype des différents contacts à Deloitte. Ce service est bénéfique dans le cas où l’utilisateur n’as pas un accès direct à Skype.

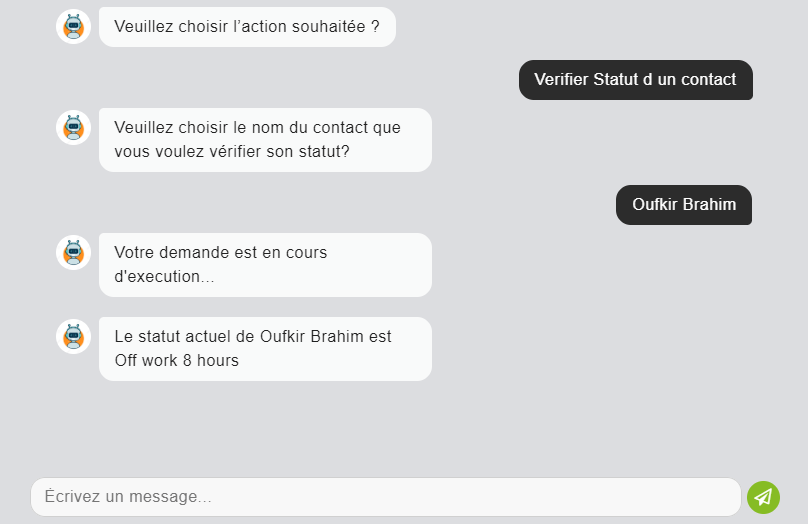


Figure 36 : Vérifier le statut d'un contact sur Skype

III.3. Interface Redmine

L’utilisateur peut choisir son choix parmi les services Redmine présentés soit en cliquant sur ce service ou en écrivant le nom du service. Le Chatbot propose 3 différents services :

* Chercher un ticket Redmine.
* Modifier un ticket Redmine.
* Créer un ticket Redmine.

III.3.1 Chercher un ticket Redmine

L’utilisateur peut chercher un ticket dans la plateforme Redmine à travers le chatbot sans accéder manuellement à la plateforme. Le type de filtres de recherche est sélectionné par l’utilisateur, on trouve 3 types de recherche :

* Recherche par - Auteur
* Recherche par - Sujet
* Recherche par – Assignée à



Figure 37 : Liste des choix fonctionnalité Redmine

En choisissant le filtre « Assignée à » Une liste déroulante est apparue pour sélectionner la personne assignée au tickets demandés

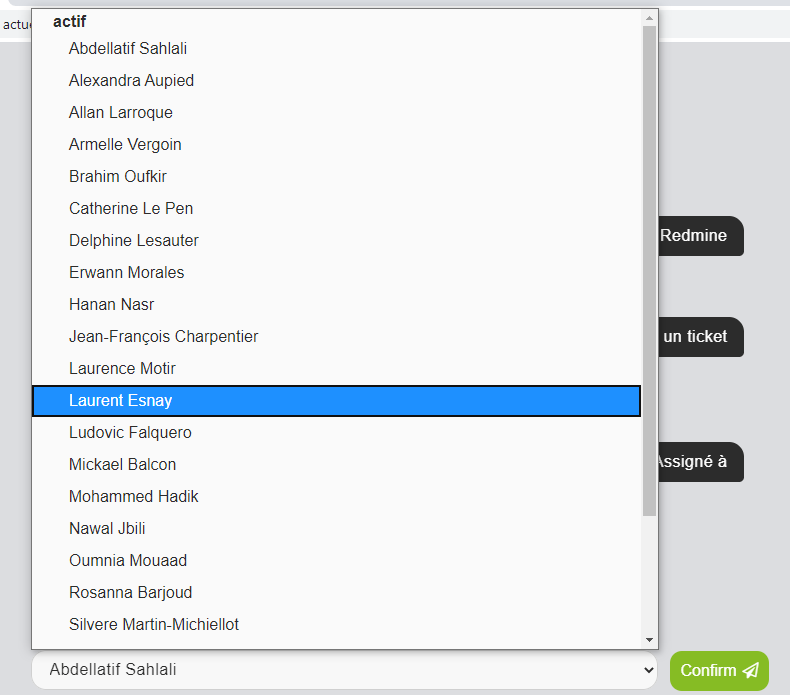


Figure 38 : Liste déroulante (choix de la personne qu'on veut afficher ses tickets)

Finalement le résultat est affiché sous forme d’un tableau adapté aux deux version web et mobile.

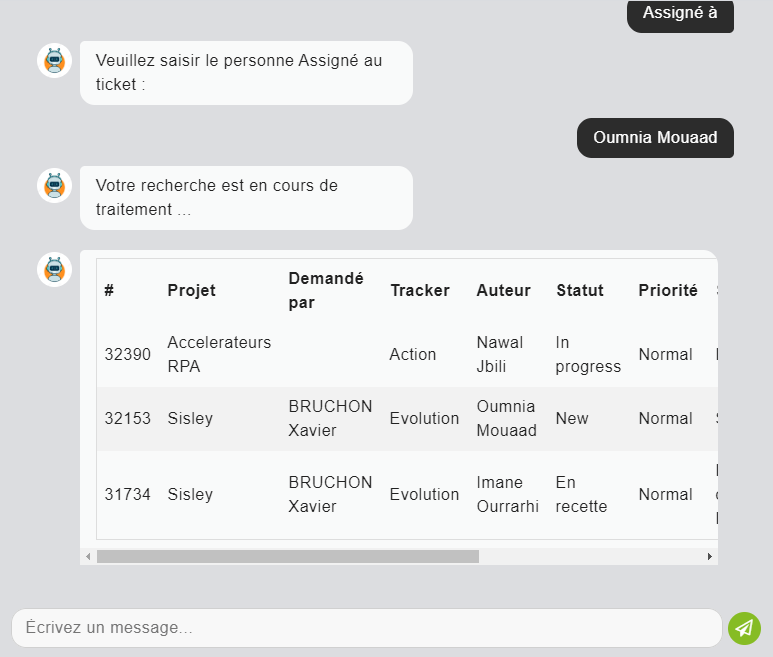
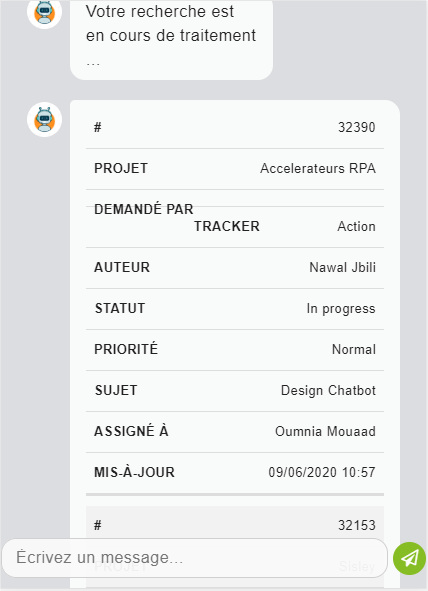


Figure 39 : Résultats de la recherche (tableau)

III.3.2 Créer un ticket Redmine

Cette action permet à l’utilisateur de créer un ticket sur la plateforme Redmine en indiquant la personne assignée au ticket, la personne « Demandé par », type du ticket et finalement le nom du ticket à créer.

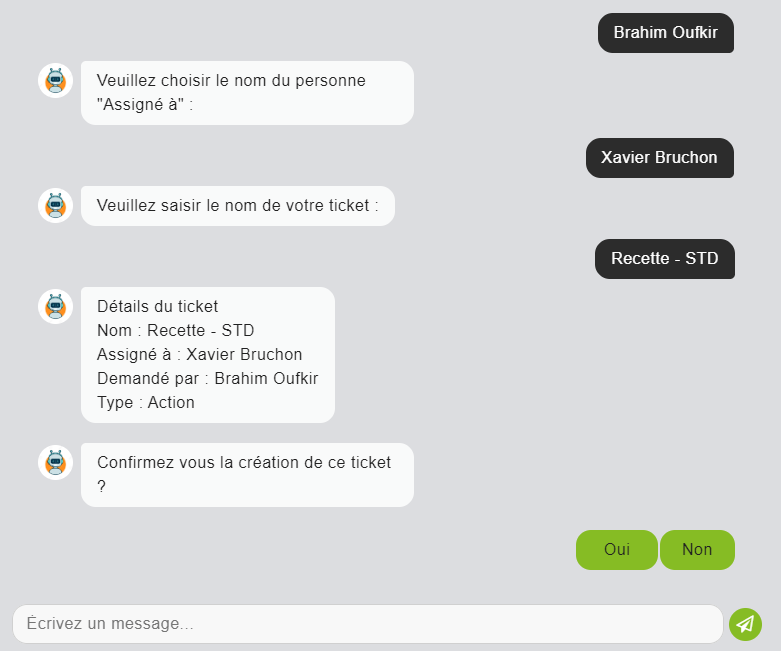


Figure 40 : procédure de création du ticket

Une fois utilisateur confirme la création du ticket Redmine, le processus se lance et le Robot UiPath renvoie la réponse de la création en indiquant le numéro du ticket crée.

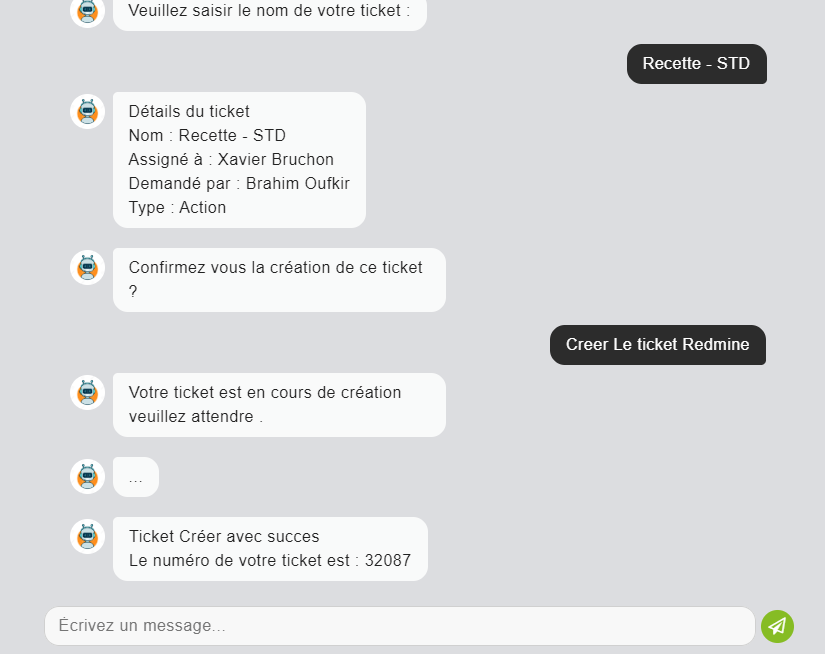


Figure 41 : Message de création du ticket



Figure 42 : Création réussie du ticket sous redmine

III.3.3 Modifier un ticket Redmine

L’utilisateur peut modifier un ticket en indiquant le paramètre à modifier (Nom du ticket, Assigné à…)

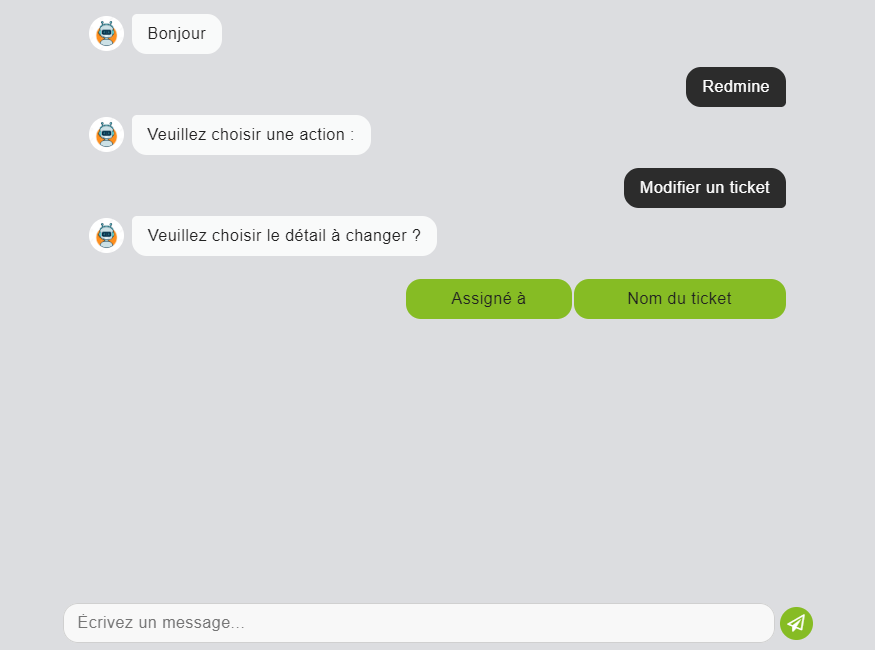


Figure 43 : Procédure modification du ticket Redmine

Une fois les détails du changement sont mentionnés, le chatbot renvoie le nom du processus au robot UiPath pour l’exécuter. Et un message de confirmation est renvoyé à l’utilisateur après le changement du ticket.



Figure 44 : Message de succès (modification du ticket Redmine)

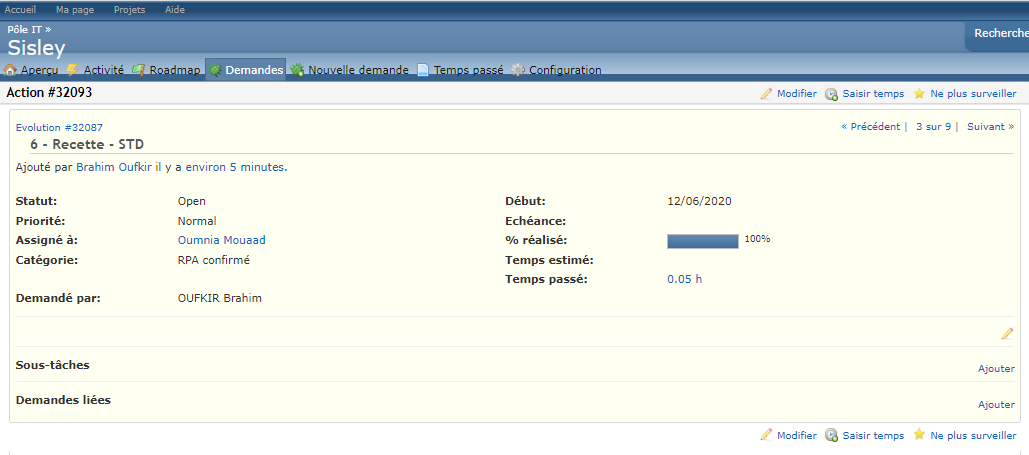


Figure 45 : Modification du ticket réussie sous Redmine

III.4. Interface Outlook

En choisissant l’action Outlook le chabot demande à l’utilisateur le process qu’il veut exécuter en lui proposant 2 processus Outlook :

* Envoyer un mail à un contact.
* Créer une réunion Microsoft Teams.

III.4.1. Créer une réunion Microsoft Teams

Ce service aide l’utilisateur à créer une réunion avec les participants qu’il choisit en mentionnant la date, les noms des personnes invitées et le nom de la réunion.

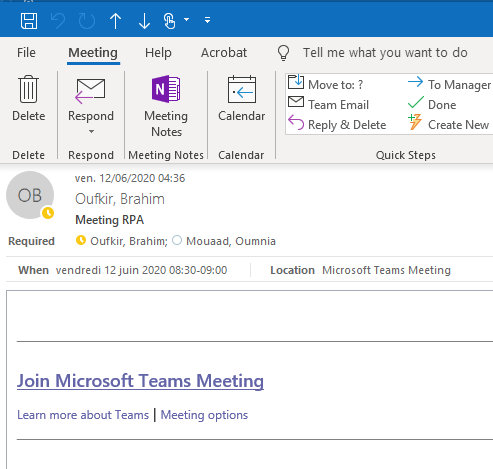
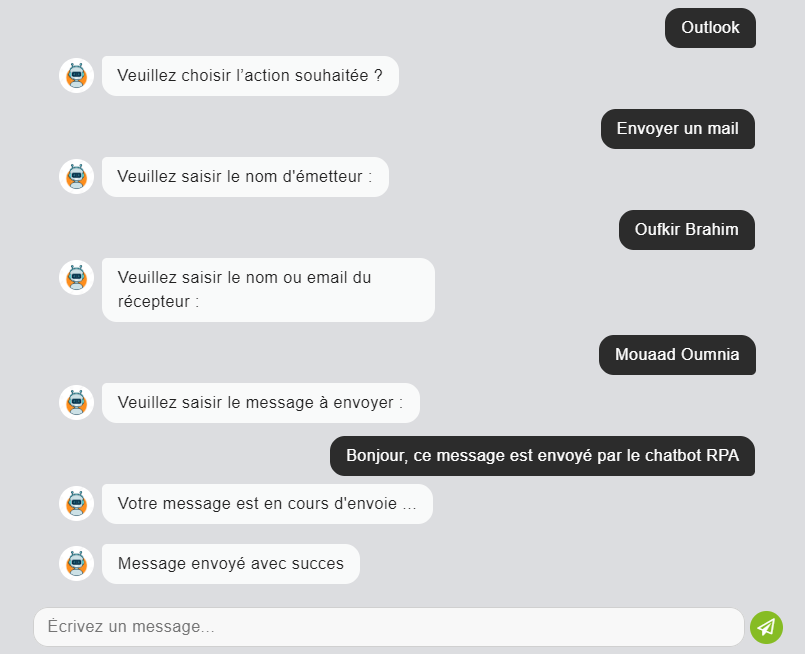
Le processus est similaire à la création d’un réunion Skype.

Figure 46 : Procédure de création d'une réunion Teams

III.4.2. Envoie d’un mail Outlook

Ce service aide l’utilisateur à envoyer un mail par Skype à un autre contact, en mentionnant son nom, le nom de récepteur et le message souhaité. Ce service va aider l’utilisateur dans le cas où il n’a pas accès direct à Skype et qu’il veut contacter un autre membre de la firme.

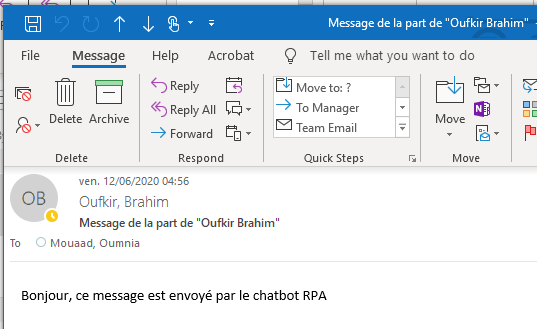


Figure 47 : Message de succès (Envoie du mail)

Conclusion

Au cours de ce chapitre, nous avons défini l’environnement du travail matériel et logiciel avec lequel nous avons pu implémenter cet agent conversationnel. Nous avons essayé par la suite d’expliquer le fonctionnement de notre système en présentant quelques imprimes écrans de chaque module. En achevant ce chapitre, nous nous approchons de la fin de ce mémoire qui sera clôturée par une conclusion générale.

Conclusion Générale

L’objectif de ce projet de fin d’études était de développer un chatbot implémenté en JavaScript (selon Node.JS) exposant une interface de conversation simple et interagissant avec DialogFlow pour reconnaître les tâches des utilisateurs un exécuter et les transformer en éléments de file d'attente (transmis via l'API Orchestrator) afin de transformer la demande de l'utilisateur en actions et processus RPA.

Cela couvre toute la chaîne, de l'interface du chat avec Orchestrator et NLP entre les deux, pour créer un chatbot compatible RPA basique mais fonctionnel et extensible.

Ce projet nous a donné la possibilité de découvrir et de maitriser une partie d’un domaine très compliqué qui est l’intégration de l’intelligence artificielle dans l’RPA, aussi de travailler avec une équipe d’experte.

Pour le moment, le Chatbot RPA est terminée en développement et en recette côté édition. Elle est maintenant arrivée à la phase de publication à l’équipe Techno de Deloitte.

Pour ce qui est des perspectives, on va basculer avec l’équipe, vers une évolution règlementaire tout en restant très réactif pour RPA Chatbot dans le cas où une anomalie est remontée.

L’expérience au sein d’un cadre professionnel, nous as été bénéfique. Ce stage nous as permis de se familiariser avec la vie professionnelle, de se développer en RPA, d’approfondir nos connaissances théoriques, acquises à l’Ecole Nationale des Sciences Appliquées Kenitra et de décrocher notre premier emploi au sein de Deloitte Maroc.

Webographie

<https://www.margo-group.com/fr/actualite/introduction-aux-chatbots-dialogflow/>

<https://www.lemagit.fr/definition/RPA-ou-Automatisation-robotisee-des-processus>

<https://www2.deloitte.com/fr/fr/pages/strategie-et-innovation/solutions/robotic-process-automation.html>

<https://www.lebigdata.fr/rpa-robotic-process-automation-definition>

<https://www.logpickr.com/fr/logpickr-accelere-la-RPA.html>

<https://www.dydu.ai/traitement-langage-naturel-fonctionnement/>

<https://blog.octo.com/comparatif-moteurs-chatbots/>

<https://www.wikipedia.org/>