**UNIVERSITÉ DU QUÉBEC À MONTRÉAL**

**CONOPS**

**TRAVAIL**

**PRÉSENTÉ**

**À**

**MILI HAFEDH**

**DANS LE CADRE DU COURS**

**PROJET D’ANALYSE ET DE MODÉLISATION**

**INM5151**

**PAR**

**de VIPART Lukas ...............................DEVL17109700**

**PINEAULT Yannick...............................PINY01059709**

**THIEMONGE Christophe……………….THIC05089507**

**11 OCTOBRE 2019**

## 

**Historique des version :**

11/09/2019. version 1

**Préface:**

Ce ConOps est un document montrant l’intérêt du développement d’un bot discord nommé Personal Assistant destiné à unifier des fonctions qui sont habituellements fragmentées entre plusieurs bots voir d'utilisation manuelles.

## 

**Table des matières**

[**Introduction**](#_yhlfdvzd35u1) **4**

[Objectifs du document](#_ie4z2zmr7ajd) 4

[Vue d’ensemble du système](#_kuunxyra8hyk) 4

[**La Situation ou le système actuel**](#_i5fzq3ir0tmv) **5**

[Contexte, objectifs et portée du système actuel](#_ppghdyxrxo7p) 5

[Politiques opérationnelles et contraintes](#_9w6zl2wh3e23) 5

[Description du système actuel ou de la situation courante](#_krpdwe9146t3) 5

[Les modes d’opération du système ou de la situation actuelle](#_izw8x880ffi) 6

[Les classe d’utilisateurs et les autres personnels impliqués](#_padayyt15he9) 7

[L’environnement de support](#_7pfn56wzyeu3) 7

[**La justification et la nature des changements**](#_b78s32arywy3) **8**

[Justifications des changements](#_6ra2szxuozct) 8

[Description des changements](#_6fc6u5ptcw2j) 8

[L’ordre de priorité dans les changements](#_edit97qcogm6) 9

[Les changements considérés mais non-inclus](#_myjmc9982b01) 11

[Hypothèses et contraintes](#_ywe993lzuppo) 12

[**LE CONCEPT DU SYSTÈME PROPOSÉ**](#_ussf724vaszu) **13**

[Contexte, objectifs et portée du système proposé](#_vy3v50a01du1) 13

[Politiques opérationnelles et contraintes](#_rcawvfpo834e) 13

[Description du système proposé](#_mqg22er0zmm) 13

[Les modes d’opération du système proposé](#_g4piyj9rnkei) 14

[Les classes d’utilisateurs et les autres personnels impliqués](#_o1zks7v7e9h8) 14

[La structure organisationnelle](#_vnro7r92j99l) 14

[Les interactions entre les utilisateurs](#_1y0gdq1wkb7u) 15

[**Les scénarios d’opération**](#_x6chzevi7u68) **16**

[**Le sommaire des impacts**](#_hqcrhsrz1ksb) **24**

[Les impacts opérationnels](#_o2e1z0pirpmc) 24

[Les impacts organisationnels](#_gav1hclaprmi) 24

[Les impacts durant le développement](#_t4sc38hextu) 24

[**L’analyse du système proposé**](#_tx8f8ilh7p9q) **25**

[Le sommaire des améliorations](#_dyro0d10e6oq) 25

[Les inconvénients et limites](#_j8ajrllfl7zt) 25

[Les alternatives et compromis considérés](#_r0zgcomuhwli) 25

[**Notes**](#_lu0vw7eeop36) **26**

[**Annexes**](#_64ipk78dtkgt) **26**

[**Glossaire**](#_ibu4usyu9kf9) **26**

**Liste des figures**

Figure 1 : Le fonctionnement du bot …………………………...…..P5

Figure 2 : Le lancement d’une commande …………………...…...P6

Figure 3 : La création d’un rôle ………………………………...…..P11

Figure 4 : Le paramétrage d’un salon ………………………….….P12

Figure 5 : Une structure organisationnelle discord basique …….P14

**Listes des acronymes et abréviations**

PA : Personal Assistant (le nom du bot)

## 

## 

## Introduction

Le système présenté par ce document est un bot discord nommé Personal Assistant.

### **Objectifs du document**

Ce document servira à décrire les améliorations proposées par rapport aux bots actuels en expliquant les besoins typiques des utilisateurs discord et leurs attentes. En outre il est destiné notamment aux utilisateurs de ladite plateforme mais aussi à ceux qui ne connaitraient pas la plateforme et souhaite comprendre l’intérêt de PA.

### Vue d’ensemble du système

Discord est une application permettant aux gamers de discuter tout en jouant, en outre, un chat vocal est proposé, mais aussi un chat textuel avec de nombreuses options d’administrations, les utilisateurs rejoignent un serveur, puis se voient attribuer des rôles qui leurs permettent de discuter, partager des images, des vidéos, des liens, créer des salons privés, etc…

Afin de palier à la difficulté d’administration d’un grand nombre d’utilisateurs (notamment sur des discords publiques) mais aussi pour ajouter des fonctionnalitées qui ne sont pas présentes à l’origine (mini jeu sur chat texte ou musique en chat vocal), des bots ont été introduits, ils sont développés par la communauté et offrent un panel de fonctionnalité extrêmement large, allant de la simple attribution de rôles automatiquement à des utilisateurs, à la gestion poussé des messages et lancement de serveurs de jeux.

PA sera un bot discord, ce qui signifie qu’il ne sera pas utilisable en dehors de cette application. Nous souhaitons par son développement améliorer la fluidité d’utilisation des serveurs sur lesquels il se trouve (administration) mais aussi permettre plus d’interaction avec les utilisateurs (mini-jeux, discussion, musique…).

## La Situation ou le système actuel

### Contexte, objectifs et portée du système actuel

Les bots discord peuvent exister pour un nombre astronomique de raisons en considérant le nombre de fonctionnalités infinis qu’un programmeur peut y ajouter. Toutefois, l’idée fondamentale derrière la conception de bots peut être définie comme étant une facilité d’accès à différents services dans le cadre d’un serveur discord. Les objectifs, bien que propres à chaque bot, peuvent être regroupés sous quelques catégories, tels que le divertissement, l’accès rapide à de l’information ou encore l’aisance d’utilisation des outils d’administration. Le système actuel ne permet toutefois pas de changer les fonctionnalités d’un bot pour correspondre à ses propres besoins ou encore de changer les commandes qui permettent d’appeler chacune des fonctions.

### Politiques opérationnelles et contraintes

Peu de contraintes sont reliées aux bots discord. Toutefois, il faut host le service sur un ordinateur pour que les utilisateurs puissent l’utiliser, comme n’importe quel service web. Ainsi, une utilisation interne qui prévoit de limiter l’utilisation du bot au sein de notre équipe ne possède pas de contraintes sauf un ordinateur de base agissant comme serveur. Par contre, pour rendre accessible un bot à qui le désire, il faut s’assurer d’avoir un ordinateur suffisamment puissant pour gérer un nombre de requêtes pouvant dépasser les centaines, voir les milliers.

### Description du système actuel ou de la situation courante

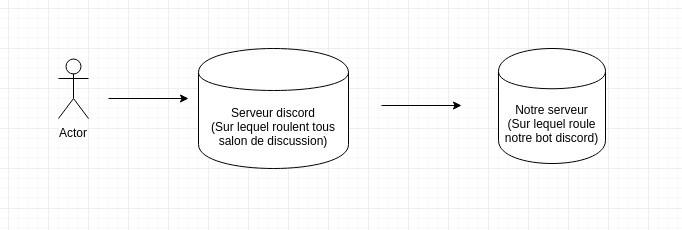


Fig 1 : le fonctionnement du bot

Il est important de comprendre que les fonctionnalités que le système actuel propose sont infinies et qu’un utilisateur peut sans doute trouver un bot qui contient la fonctionnalité qui l’intéresse sur le web. Toutefois, c’est la combinaison de fonctionnalités dans un même bot qui fait objet de la question, combinaison que nous adapterons aux besoins de notre équipe.

Il est relativement simple de conserver des performances correctes puisque les fonctionnalités qu’un bot discord offre sont normalement peu gourmandes au niveau du CPU. La vitesse de réponse d’un service peut s’estimer alors en considérant le temps de transmission de l’ordinateur d’un utilisateur, jusqu’au serveur discord (normalement automatiquement construit le plus près possible du créateur, donc nous), jusqu’à l’ordinateur où le bot est hosté, suivi de l'exécution du code que la fonctionnalité fournie par le bot. Le temps de transmission sera bref, puisque les serveurs discord sont répartis partout dans le monde, donc près de nous et que l'exécution du code à travers le bot est relativement simple.

Le soucis vient de la flexibilité puisque des modifications sont difficilements réalisables sur les bots discords actuels. Pour obtenir un bouquet de fonctionnalité personnalisé il faut alors ajouter plusieurs bots à son serveur chacun avec leur manière de réaliser des commandes.

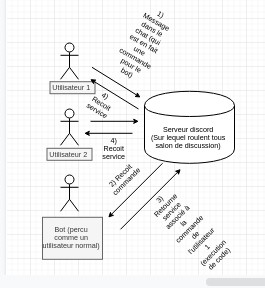


Fig 2: Le lancement d’une commande

### Les modes d’opération du système ou de la situation actuelle

Le mode largement utilisé est le mode normal, c’est-à-dire quand la machine sur lequel roule le bot fonctionne de manière suffisante. Si toutefois la machine se trouve surchargée pour quelconque raison, le service ralentit, mais reste généralement accessible. Parfois la machine qui host le bot ferme (update, panne de courant, etc.) le service est alors rendu inaccessible. Il y a parfois des backups pour les plus gros bots.

### Les classe d’utilisateurs et les autres personnels impliqués

Le type d’utilisateur primaire de discord est celui qui utilise les services offerts par les bot qui ne donnent aucun privilège sur les autres. En effet, la vaste majorité des commandes sont accessibles à tous, comme la musique ou encore une information à la météo rapide. Un utilisateur de base peut ainsi exécuter ces commandes.

Le deuxième genre d’utilisateur est celui qu’on peut classifier d’administrateur. Ceux-ci ont accès aux mêmes fonctionnalités que l’utilisateur de base, mais aussi à certaines autres commandes spécifiques, telles que pouvoir expulser un autre utilisateur du serveur discord, ou encore avoir certaines informations plus privées d’un utilisateur.

La dernière classe d’utilisateur serait celle en lien avec la maintenance des bots. Ceux qui ont accès aux bots (que ce soit les hôtes ou les développeurs) et qui peuvent faire des modifications et ajouter des fonctionnalités.

### L’environnement de support

L’environnement nécessaire pour s’assurer du bon fonctionnement à court et à long terme est simplement une machine de puissance moyenne qui reste en fonctionnement de manière constante (de préférence, pas un portable qui subit le trajet maison-école chaque jour par exemple). La machine doit également avoir un accès Internet minimalement rapide (10-15 mB/s) et fiable pour pouvoir recevoir les requêtes des autres utilisateurs et les renvoyer. Les bots sont parfois rendu accessibles à plusieurs personnes en les affichant sur un site web rassemblant les bots discord.

## La justification et la nature des changements

### Justifications des changements

Comme nous l’avons vu, le système actuel est peu flexible; la modification d’un bot existant n’est pas vraiment possible. Afin de palier à ce problème l’utilisation de multiples bots ou l’accomplissement de certaines tâches redondantes à la main sont parfois envisagés. La création d’un bot personnalisé répondant uniquement à nos besoins nous permettra de solutionner ce problème.

### Description des changements

Les fonctions d’administrations vont tout d’abord être regroupées au sein d’un seul bot:

* + - Autosuppression des commandes déjà réalisées
    - Arrêt du bot
    - Ramener les logs d’un utilisateur
    - Bannissement/kick d’un utilisateur
    - Auto-attribution de rôles par réactions
    - Attribution de rôle par message
    - Attribution automatique d’un rôle de base
    - Suppression automatique de certains messages par mots clés
    - Création de channels textuels/vocaux
    - Envoie de messages privés aux utilisateurs du discord

Puis un certain nombre de commandes de divertissement vont être ajoutées :

* + - Lecture de musique par youtube
    - Auto apprentissage de réponses automatiques à certains messages (ou de réactions)
    - Lancement de mini jeux textuels (pierre papier ciseau/ lancer de pièce)
    - Recherche d’image google image/reddit
    - Recherche de définitions (ou plus généralement de morceaux de textes)
    - Lancement et gestion d'applications sur la machine hôte
    - Demande de la météo pour une zone X
    - Demande de l’heure pour une timezone X
    - Ramener le résultat d’une recherche youtube
    - Ramener les statistiques de joueur pour un jeu
    - Ramener des infos sur un film

### L’ordre de priorité dans les changements

Il est essentiel d’implémenter ces fonctions:

- arrêt du bot : cette fonction est une très importante pour pouvoir faire des

tests notamment mais aussi pour pouvoir par la suite redémarrer la machine

sur laquelle est hébergée le bot ou pour corriger des bugs si besoin

- la lecture de musique youtube: Discord étant à la base une application basée sur son chat vocal, l’ajout de musique par youtube pour distraire les utilisateurs lorsqu’ils sont plusieur en chat vocal (que ce soit en musique de fond ou en topic principal de la conversation) est primordiale; c’est la première feature ajoutée sur la plupart des serveurs communautaire

- l’attribution de rôle par message; elle permettra aux utilisateur de s'auto attribuer des rôles de bases pour participer aux différents canaux de discussion ainsi qu’aux administrateurs d’attribuer certains rôles qu’ils jugent nécessaire plus facilement (il faut normalement cliquer sur le nom de l’utilisateur dans la liste puis chercher le rôle qu’on souhaite lui attribuer), les administrateurs seront alors délesté de la tâche de l’attribution d’un rôle à chaque nouvel utilisateur

- la commande de bannissement/kick; il sera bien plus aisé de bannir ou expulser un utilisateur via une simple commande qu’en passant par tout le système de validation et justifications normalement demandé par l’application, en outre c’est une fonction de base d’administration d’un chat qui ne peut pas être négligée

Il serait souhaitable d’implémenter ces commandes :

- auto suppression des commandes déjà réalisées; cette commande permet de maintenir le chat textuel propre sans avoir besoin de créer un chat dédié uniquement aux commandes (ou tout du moins en limiter l’usage).

- ramener le résultat d’une recherche youtube : si l’on a pas le lien youtube sous la main pour demander une lecture il est plus intéressant de demander au bot de rechercher pour nous la musique que l’on souhaite écouter plutôt que d’aller manuellement le faire sur youtube

- envoie de messages privés aux utilisateurs du discord: cette fonction permet surtout aux administrateurs et modérateurs d’envoyer des messages concernant la modération sans s’impliquer directement, ceci afin d’éviter les conflits d'intérêt, mais aussi afin de grouper certains messages

-Suppression automatique de certains messages par mots clés : cette fonction permet surtout d’éviter les débordements en supprimant des messages qui pourraient s’avérer indésirable, elle n’est pas absolument nécessaire mais est très utile pour un grand serveur (par exemple si la modération dort)

Ces fonctions sont optionnelles :

-recherche de définitions, ramener des infos sur un film, demander la météo pour une zone X, demander l’heure pour une timezone X : ces fonctions sont surtout là pour alimenter la discussion et rendre services aux utilisateurs, leur évitant de changer d’onglet sur mobile ou d’ouvrir le navigateur sur ordinateur

-recherche d’images google/reddit : cette fonction permet d’alimenter la discussion elle n’est pas d’une importance capitale

-lancement de mini-jeux textuels: c’est avant tout une fonction de divertissement pour les utilisateurs, de nombreux jeux sont envisageables comme un pendu ou un pile ou face par exemple;

-auto apprentissage de réponses automatiques : cette fonction sert elle aussi à alimenter la discussion mais aussi à se divertir, elle peut être plus ou moins compliquée à implémenter

-ramener les statistiques de joueur pour un jeu : discord étant à la base une application prévue pour les jeux il est intéressant de pouvoir consulter ses statistiques à la volée par une simple commande

-auto attribution de rôle par réaction et attribution automatique d’un rôle de base : ces deux fonctions sont intéressantes mais pas forcément nécessaire dans la mesure où une première fonction d’attribution de rôle est implémentée, il sont en outre plus complexes à mettre en place pour un intérêt pas forcément évident.

-Ramener les logs d’un utilisateur : c’est un outil d’administration qui permet de savoir ce qu’un utilisateur a dit même s’il a supprimé ses messages. Ils sont accessibles de base dans le menu des paramètres du serveur mais il peut être util d'avoir accès à une partie limitée des logs à l’aide d’une simple commande

-lancement et gestion d’applications sur la machine hôte : surtout utilisée pour gérer des serveurs de jeux tels que minecraft cette commande sera intéressante pour un serveur entre ami, elle pourrait aussi permettre de tester des fonctions par exemple sur une machine virtuelle

-Création de channels textuels/vocaux : la plupart du temps ces channels sont créés par l’administrateur dès le départ, il est rare d’avoir besoin de nouveaux channels mais il peut être intéressant d’en créer un ponctuellement sans paramétrage spécifique (par exemple pour noël), auquel cas le bot peut s’en charger

### Les changements considérés mais non-inclus

La création de rôles est une fonction qui aurait pu avoir un certain intérêt mais qui est finalement bien trop complexe à intégrer pour un grain minime par rapport à la simplicité de le faire manuellement ainsi qu’au nombre de fois où cette fonction est utilisée.

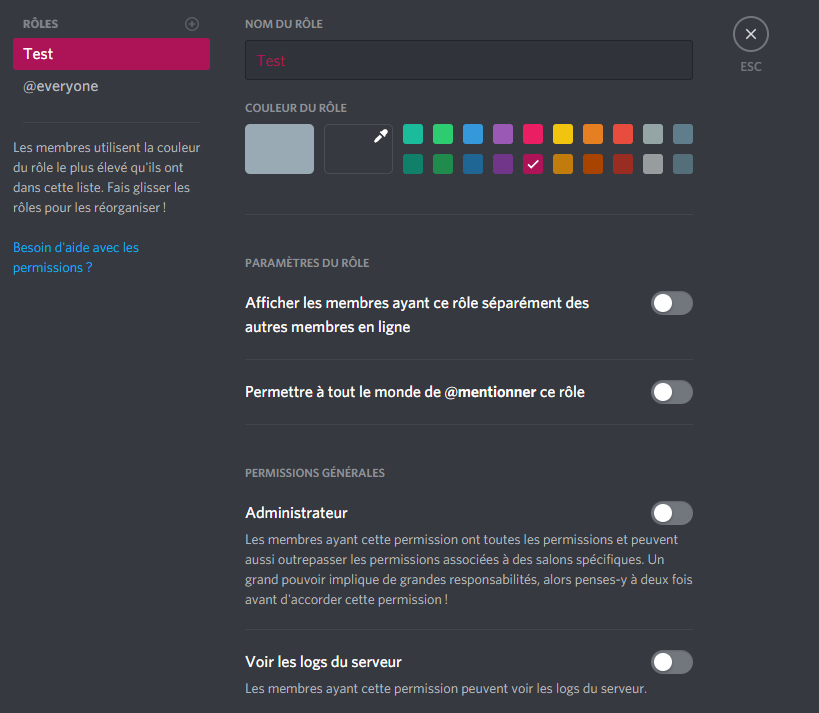


Fig 3: la création d’un rôle

De même la modification des attributions pour les salons (par exemple bloquer un salon de discussion de modérations aux utilisateurs lambdas) a été considérée mais comme précédemment prendrait trop de temps à implémenter pour un gain relatif; il ne faut pas oublier qu’il faudrait préciser chaque paramètre pour ce genre de fonction.

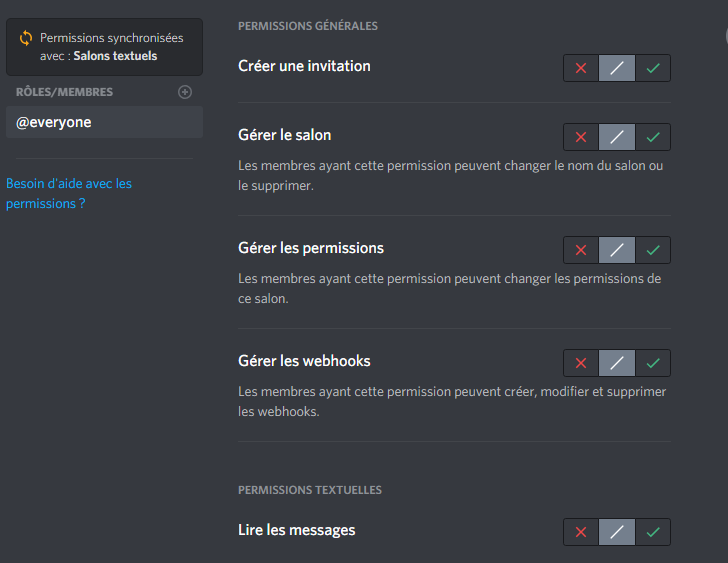


Fig 4: le paramétrage d’un salon

Comme on le voit il faut venir modifier les permissions pour chaque rôle, c’est fastidieux mais ce serait encore pire sur une commande, surtout si une erreur est commise dans la commande.

### Hypothèses et contraintes

La charge du système devrait rester globalement constante dans le temps; PA vise avant tout des petits serveurs privés ou communautaires (1000 utilisateurs grand maximum mais plutôt moins de 100 de manière générale).

De même PA vise des serveurs avant tout francophones, seule la langue française sera donc prise en compte. Afin de faire fonctionner certaines fonctionnalitées telles que la musique Youtube qui est l’application la plus utilisée sera la seule supportée par PA même si d’autres applications existent avec leurs propres API.

## 

## LE CONCEPT DU SYSTÈME PROPOSÉ

### Contexte, objectifs et portée du système proposé

PA a pour but premier de faciliter et fluidifier l’utilisation d’un serveur discord que ce soit du point de vue des utilisateurs mais aussi du point de vue des administrateurs et modérateurs. Cela passe par l’unification des fonctionnalités d’administration mais aussi des fonctionnalités de divertissement communément recherchées dans les bots.

### Politiques opérationnelles et contraintes

Le système devrait être disponible en permanence, de ce fait il devrait être hébergé sur une machine de puissance moyenne qui n’est jamais mise à l’arrêt (un ordinateur personnel peut suffire). Le système n’a pas besoin d’employés pour opérer mais peut nécessiter un gestionnaire pour le redémarrer en cas de soucis ou pour faire des mises à jour si besoin est.

### Description du système proposé

Les bots étant des systèmes assez simple et ne dépendant que de la machine hôte pour fonctionner le coût d’opération du système sera une fraction de la facture d’électricité et d’internet de la machine hôte ou le coût de location d’un serveur.

PA ne va opérer que sur discord et sera donc limité à un fonctionnement par chat textuel sur les divers serveurs autorisés. (voir Fig 2 en page 5)

En outre il roulerait sur un serveur node.js supporté par une machine Windows ou linux. Le bot obtiendra l’information par les commandes de l’API de Discord. Il renverra cette information au serveur Discord qui affichera le résultat sur son API.

Les ‘’features’’ de notre bot sont les mêmes qu’un bot Discord, mais complètement ‘’sur mesure’’ pour notre équipe.

Les caractéristiques de performances sont les mêmes que pour un bot normal puisque la principale feature de notre bot est de regrouper des fonctions.

Nous utiliserons la bibliothèque de Discord qui se charge déjà de la confidentialité; par exemple par encryptions des donnés envoyé par et à notre bot. Nous n’avons pas de système qui empêche une attaque directe sur l’ordinateur. Il serait probablement possible de corrompre le fichier du bot directement sur la machine si on pouvait s’y infiltrer.

### Les modes d’opération du système proposé

Il n’existe que le mode d’opération normal du système, il sera marqué connecté sur discord et toutes les fonctions seront disponibles. En cas de maintenance ou de mise à jour, PA sera simplement mis hors ligne et apparaîtra déconnecté sur discord.

### Les classes d’utilisateurs et les autres personnels impliqués

Les utilisateurs seront divisés en fonction de leurs rôles sur les serveurs.

L’hôte sera celui qui lance le bot sur sa machine, il aura donc aussi la possibilité de modifier le code de PA s’il en ressent le besoin.

Les Administrateurs auront accès à la totalité des fonctions.

Les Modérateurs auront accès à toutes les fonctions d’administration qui leur sont autorisé par le serveur (si ils ne peuvent pas bannir un utilisateur alors ils ne pourront pas le faire par le bot)

Les Utilisateurs n’auront accès qu’à l’attribution de rôle pour eux même et aux fonctions liées au divertissement.

Plus de classes d’utilisateurs peuvent exister en fonction des serveurs (animateurs par exemple), PA étant composé de fonctions d’administrations et de divertissement découlant directement des autorisations discord il peut exister autant de classe d’utilisateur que l’administrateur du serveur le souhaite avec chacun ses autorisations par rapport aux bots (un gestionnaire de bot, seul à avoir le droit aux fonctions d’administrations peut aussi exister).

### La structure organisationnelle

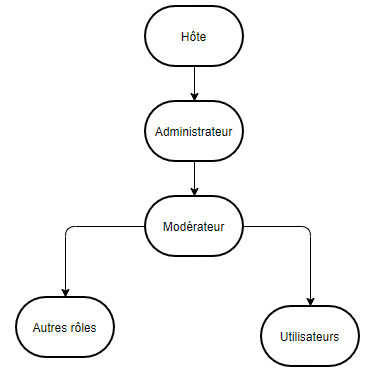


Fig 5: une structure organisationnelle discord basique

L’hôte est à part, pour le système proposé il est administrateur principal du serveur mais il pourrait être simple modérateur. Néanmoins si l’hôte quitte le serveur alors le bot quitte le serveur.

Les administrateurs ont tout pouvoir sur le serveur et donc sur le bot au sein d’un serveur défini. Les modérateur ont des droits limités vis à vis des fonctions d’administration, ce qui sera répercuté sur le bot. Les utilisateurs n’ont le droit que de s’auto administrer et uniquement de manière limitée (nombre limité de rôles, un utilisateur ne peut pas s’attribuer le rôle administrateur). Les autres rôles dépendent du serveur.

Les actions sur le bot dans un serveur peuvent avoir des répercussions sur un autre serveur. Notamment l’utilisation de la fonction de lecture de musique, le bot devant rejoindre un salon vocal, et étant considéré comme un utilisateur, un utilisateur ne peut être que dans un salon à la fois, si le bot joue de la musique sur un serveur X il ne pourra pas jouer sur un autre serveur Y.

### Les interactions entre les utilisateurs

Les interactions les plus importantes sont les droits de modération des utilisateurs modérateur et administrateurs (et hôte si celui-ci est modérateur ou plus) envers les utilisateurs de plus bas niveau; notamment la gestion des rôles des expulsions et des bannissements qui sera autorisé par le système uniquement si le grade de l’utilisateur l’y autorise.

### 

## Les scénarios d’opération

| Administration d’utilisateur | Toute fonction qui vise à agir sur un utilisateur (modification de rôle, expulsion, ban, demande de logs) |
| --- | --- |
| Acteurs | Acteur primaire : Administrateur/modérateur/toute personne ayant les autorisations nécessaires  Acteurs secondaires : utilisateur ayant un rang inférieur à l’acteur primaire |
| Préconditions | L’utilisateur de cette fonction doit avoir les autorisations nécessaires et le bot doit lui aussi avoir les autorisations nécessaires. |
| Postconditions | La commande demandée est réalisée ; exemple pour une expulsion l’utilisateur ciblé est retiré du serveur, pour une attribution de rôle, le rôle demandé est ajouté à l’utilisateur cible. |
| Scénario principal | 1. L’administrateur entre sa commande dans le chat textuel  2. Le bot reconnaît la commande  3. Le bot recherche l’utilisateur ciblé  4. La tâche demandée est réalisée sur la cible demandée  5. Un message de confirmation est mis dans le chat |
| Scénarios alternatifs | Si les autorisations nécessaires ne sont pas remplies par le demandeur, le bot envoie un message d’erreur. |
| Fréquence d’occurrence | Dépend de la taille du serveur :  Petit : rarement  Grand : souvent |

| Auto administration d’utilisateur | Toute fonction qui vise à agir sur soi-même (modification de rôle par réaction ou message) |
| --- | --- |
| Acteurs | Acteur primaire : tout utilisateur |
| Préconditions | L’utilisateur de cette fonction doit avoir les autorisations nécessaires et le bot doit lui aussi avoir les autorisations nécessaires. |
| Postconditions | La commande demandée est réalisée ; exemple pour une attribution de rôle par réaction, l’utilisateur reçoit son rôle après avoir cliqué sur la réaction demandée |
| Scénario principal | 1. L’utilisateur entre sa commande dans le chat textuel  2. Le bot reconnaît la commande  3. La tâche demandée est réalisée sur le demandeur  4. Un message de confirmation est mis dans le chat |
| Scénarios alternatifs | Si les autorisations nécessaires ne sont pas remplies par le demandeur, le bot envoie un message d’erreur. |
| Fréquence d’occurrence | Dépend de la taille du serveur :  Petit : rarement  Grand : souvent |

| Attribution d’un rôle de base | Attribue automatiquement un rôle à tout utilisateur qui rejoint le serveur |
| --- | --- |
| Acteurs | Acteur primaire : Tout le monde |
| Préconditions | Le bot doit être connecté au serveur. La personne n’a pas de rôle. |
| Postconditions | La personne se connectant pour la première fois se voit attribué un rôle |
| Scénario principal | 1. Un utilisateur rejoint le serveur  2. Le bot lui attribue tout de suite un rôle |
| Fréquence d’occurrence | Dépend de la taille du serveur :  Petit : rarement  Grand : souvent |

| Suppression de messages | Comprend la suppression de messages par mots clés et l’auto suppression de commandes |
| --- | --- |
| Acteurs | Acteur primaire : Tout usager |
| Préconditions | Un message inapproprié ou une commande a été entré dans le chat |
| Postconditions | Le chat est nettoyé |
| Scénario principal | 1. Un utilisateur entre une commande  2. Le bot détecte que le message est une commande  3. Le bot supprime le message |
| Scénarios alternatifs | 1. Un utilisateur entre un message  2. Le bot détecte un mot interdit  3. Le bot supprime le message |
| Fréquence d’occurrence | Souvent |

| Recherche d’infos | Toute fonction demandant de récupérer des informations (statistiques de jeu, recherche YouTube, météo, timezone, infos de films, définitions, images) |
| --- | --- |
| Acteurs | Acteur primaire : Tout usager |
| Préconditions | Pas de précondition |
| Postconditions | La commande demandée est réalisée ; les informations demandées sont affichées sur le chat textuel |
| Scénario principal | 1. L’administrateur entre sa commande dans le chat textuel  2. Le bot reconnaît la commande  3. Le bot recherche les informations demandées  4. Le bot affiche les informations demandées dans le chat textuel |
| Scénarios alternatifs | Si le bot ne trouve rien il affichera un message d’erreur |
| Fréquence d’occurrence | souvent |

| Lecture de musique | Fonction qui vise à lire de la musique pour les utilisateurs du canal vocal |
| --- | --- |
| Acteurs | Acteur primaire : Tout usager |
| Préconditions | L’utilisateur de la commande doit être dans un canal vocal |
| Postconditions | Le bot rejoint le canal et joue de la musique |
| Scénario principal | 1. L’utilisateur entre sa commande dans le chat textuel  2. Le bot reconnaît la commande  3. Le bot recherche le lien demandé  4. Le bot rejoint le canal vocal  5. Le bot lance la lecture de la musique demandée jusqu’à la fin |
| Scénarios alternatifs | Si les autorisations nécessaires ne sont pas remplies par le demandeur, le bot envoie un message d’erreur. |
| Fréquence d’occurrence | Dépend de la taille du serveur :  Petit : rarement  Grand : souvent |
| Autres commentaires/exigences | Si plusieurs musiques sont demandées à la chaîne elles devraient être mises en file pour être jouées à la suite. |

| Arrêt du bot | Cette fonction vise à mettre le bot hors ligne |
| --- | --- |
| Acteurs | Acteur primaire : administrateur |
| Préconditions | L’utilisateur de cette fonction doit être administrateur |
| Postconditions | Le bot est mis hors ligne et ne lit plus les commandes |
| Scénario principal | 1. L’administrateur entre sa commande dans le chat textuel  2. Le bot reconnaît la commande  3. Le bot se met hors ligne |
| Scénarios alternatifs | Si les autorisations nécessaires ne sont pas remplies par le demandeur, le bot envoie un message d’erreur. |
| Fréquence d’occurrence | Très rarement |

| Envoie de message privé aux utilisateurs du discord | Un message privé est envoyé aux utilisateurs par l’intermédiaire du bot |
| --- | --- |
| Acteurs | Acteur primaire : Administrateur  Acteur secondaire : Tout autre usager du serveur |
| Préconditions | L’utilisateur de cette fonction doit être administrateur |
| Postconditions | L’utilisateur cible reçoit un message privé de la part du bot |
| Scénario principal | 1. L’administrateur entre sa commande dans le chat textuel  2. Le bot reconnaît la commande  3. Le bot recherche l’utilisateur ciblé  4. Le bot envoie le message demandé à l’utilisateur sélectionné  5. Un message de confirmation est mis dans le chat |
| Scénarios alternatifs | Si les autorisations nécessaires ne sont pas remplies par le demandeur, le bot envoie un message d’erreur. |
| Fréquence d’occurrence | Rarement |

| Création de canaux | Cette fonction permet de créer un canal basique |
| --- | --- |
| Acteurs | Acteur primaire : Administrateur/modérateur |
| Préconditions | L’utilisateur de cette fonction doit avoir les autorisations nécessaires et le bot doit lui aussi avoir les autorisations nécessaires. |
| Postconditions | Un canal est créé |
| Scénario principal | 1. L’administrateur entre sa commande dans le chat textuel  2. Le bot reconnaît la commande  3. Le bot crée un canal vocal/textuel avec le nom demandé |
| Scénarios alternatifs | Si les autorisations nécessaires ne sont pas remplies par le demandeur, le bot envoie un message d’erreur. |
| Fréquence d’occurrence | Très rarement |

| Lancement de mini jeux textuels | Une fonction visant à lancer un petit jeu divertissant (par exemple en attendant un chargement) |
| --- | --- |
| Acteurs | Acteur primaire : Tout usager |
| Préconditions | Pas de précondition |
| Postconditions | Un mini jeu est lancé et complété |
| Scénario principal | 1. L’usager entre la commande demandant le lancement d’un jeu  2. Le bot reconnaît la commande  3. Le bot lance le mini jeu demandé  4. L’utilisateur joue jusqu’à perdre/gagner  5. Un message de victoire/défaite est affiché |
| Fréquence d’occurrence | Moyenne |

## 

## Le sommaire des impacts

### Les impacts opérationnels

L’utilisation de PA va principalement changer la manière dont sont attribués les rôles et gérés les utilisateurs en général, sur un serveur sans bot les processus administratifs se verront grandement accélérés, sur un serveur qui possède déjà un ou plusieurs bots, les commandes se verront simplement altérées et unifiées (un seul bot donc une seule manière de faire des commandes “ex: /ban bob, /admin paul plutôt que .ban bob , !admin paul”).

En cas de soucis avec le bot il faudra contacter l’hôte plutôt que de simplement attendre (dans le cas d’un bot grand public).

### Les impacts organisationnels

Une documentation (pouvant être mise au sein d’une commande du bot) devra être mise en place afin que chaque utilisateur puisse utiliser le bot facilement.

### Les impacts durant le développement

Le bot pourrait être amené à être déployé en version test sur des serveurs avec plusieurs utilisateurs pour essayer des fonctionnalités et voir si tout répond lorsqu’il est sollicité par un nombre moyen ou élevé d’usagers.

## L’analyse du système proposé

### Le sommaire des améliorations

La combinaison de fonctionnalités est ainsi l’amélioration que nous apportons au système actuel de bots. Nous aurons ainsi un bot qui offre un service pour chacun de nos besoins auquel il sera possible de faire des modifications à notre gré.

### Les inconvénients et limites

Afin de pouvoir utiliser notre bot discord il sera nécessaire pour les utilisateurs d’apprendre comment utiliser chacune des fonctions, donc de connaître quel message est associé à quelle fonctionnalité. Il serait donc naturel pour nous de faire un document avec la liste des commandes et des fonctionnalités associées. De plus, pour rendre le bot accessible à plusieurs personnes, il peut être difficil de le faire gagner en popularité vu la grande quantité de bot déjà disponible.

En outre, puisque certaines fonctionnalités utilisent de l’information ou des services venant d’autres plateformes, telles que la musique venant de youtube, ou encore le service de météo utilisant un API, la vitesse pour ces fonctionnalitées est dépendantes des plateformes associées.

La flexibilité des fonctionnalités est entièrement limitée par nous (développeurs), ce qui nous permet d’avoir les fonctionnalités que nous désirons, mais qui peuvent ne pas convenir à tout utilisateur. Leurs *souhait* est donc limité par notre acceptation et notre implémentation de leurs idées dans le bot.

### Les alternatives et compromis considérés

Puisque un bot discord est défini par ses fonctionnalités, plusieurs fonctionnalités ont été considérées pour PA. Toutefois, il était également important de ne pas surcharger celui-ci puisqu’un utilisateur peut rapidement perdre de vue ce qui est possible de faire avec chacun des bots qu’il utilise. Nous voulions ainsi implémenter celles qui nous paraissaient les plus importantes. Par exemple, des fonctionnalités telles que pouvoir lancer différents jeux à travers le bot était une option possible et intéressante. Toutefois, le concept de jeu correspondait moins avec le service offert par notre bot, soit essentiellement de l’information et des services simples. De plus, il aurait été possible au niveau de la conception d’utiliser différents langage tels que python, C#, etc. qui sont plus récents et puissants. Toutefois, les bots discord ont été initialement produits avec Javascript, qui contient ainsi beaucoup plus de documentation et d’exemples en matière de bots.

## Notes

## Annexes

## Glossaire