|  |  |
| --- | --- |
| SMA  Négociation | Résumé  Dans ce rapport est exposé une modélisation d’un système automatique de négociation à l’aide d’un système multi agents.  Jérémy Dollé, Etienne Debard  5A Informatique |

Table des matières

[Introduction 2](#_Toc501876718)

[Librairie MadKit 2](#_Toc501876719)

[Multithreading 2](#_Toc501876720)

[Système de messagerie 2](#_Toc501876721)

[Affichage 3](#_Toc501876722)

[Système de négociation 3](#_Toc501876723)

[Différents types d’agents 3](#_Toc501876724)

[Le client 3](#_Toc501876725)

[Le broker 3](#_Toc501876726)

[Le fournisseur 3](#_Toc501876727)

[Instanciation des différents agents 4](#_Toc501876728)

[Processus de négociation 4](#_Toc501876729)

[Recherche de fournisseurs 4](#_Toc501876730)

[Négociation 5](#_Toc501876731)

[Achat d’un ticket 5](#_Toc501876732)

[Stratégie de négociation 5](#_Toc501876733)

[Stratégie des clients 5](#_Toc501876734)

[Stratégie des fournisseurs 5](#_Toc501876735)

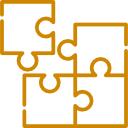
[Implémentation des stratégies de négociation 6](#_Toc501876736)

[Résultat final 6](#_Toc501876737)

[Exemples de négociations 7](#_Toc501876738)

[Conclusion 8](#_Toc501876739)

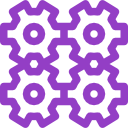
# Introduction

Dans ce rapport nous allons exposer les choix que nous avons pris pour la réalisation de ce projet. Dans un premier temps nous allons détailler pourquoi avoir choisi d’utiliser une librairie. Ensuite, nous allons expliquer notre implémentation et quelques pistes d’amélioration.

# Librairie MadKit

Le but de ce TP étant l’implémentation d’un algorithme d’apprentissage, nous avons décidé de choisir une librairie pour tout ce qui est question d’affichage pour éviter de perdre du temps à concevoir une interface et aussi pour vérifier rapidement nos résultats. La librairie MadKit est très simple d’utilisation est colle parfaitement à nos besoins, c’est pour cette raison que nous avons opté pour cette dernière.

## Multithreading

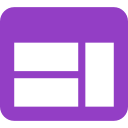
La librairie Madkit permet de générer des agents sur plusieurs threads sans se soucier de leur cycle de vie. On peut à tout moment instancier un agent, en supprimer un autre, etc… sans que l’on affecte le bon déroulement du cycle de vie d’un autre agent. Ceci parait être important à la vue du système que nous avons à implémenter. En effet, nous verrons que nous allons instancier plusieurs types d’agents à différents moments de la simulation, une stabilité des agents en vie est donc importante.

## Système de messagerie

La base d’un system de négociation est faite sur l’aptitude des différents agents à pouvoir dialoguer entre eux et cacher certaines informations. Madkit offre un system de messagerie avec adressage et réponse. En effet, chaque agent peut appartenir à une « société ». Dans chaque société, on peut ensuite définir des groupes et dans chaque groupe, chaque agent peut se voir attribuer des rôles. Ceci semble important dans notre cas, dû à la différence fonctionnelle des agents que nous allons développer.

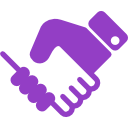
Ensuite, du fait que chaque agent, se trouve exécuté sur un thread indépendant, il est possible d’instaurer une messagerie avec attente de réponse. Et on peut également décider de continuer le cycle de vie d’un agent si la réponse à un message envoyé est trop longue, afin qu’un agent ne reste pas bloqué dans un état indéfiniment.

## Affichage

Madkit offre la possibilité de créer entièrement comme l’on souhaite l’affichage d’un agent au sein d’une fenêtre indépendante. On peut par exemple décider de changer la couleur de cette fenêtre en fonction de l’état de l’agent.

Chaque agent possède également son propre système de log qu’il est possible d’afficher au sein de sa fenêtre. On peut ainsi aisément suivre l’activité de chaque agent durant tout son cycle de vie.

# Système de négociation

Le but de ce projet est d’implémenter un système de négociation complexe. Nous allons ici implémenter une négociation de tickets de voyages par différents fournisseurs (ex : voyage par train, avion, bus, etc…).

## Différents types d’agents

Dans notre implémentation du system de négociation, nous allons instancier 3 types d’agents ayant chacun un rôle bien défini, les clients, les brokers et les fournisseurs

### Le client

Le rôle du client est simplement d’acheter un ticket de voyage. Celui-ci veut donc acheter un ticket de voyage avec un mode de transport défini à son instanciation.

Le client possède également une contrainte de budget qu’il devra respecter durant sa négociation. Afin de rendre les négociations plus intéressantes, nous avons également attaché une liste de préférences et une liste noire pour les compagnies dans lesquels il pourrait acheter son ticket.

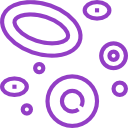
### Le broker

Le broker est un intermédiaire entre les fournisseurs et les clients, sont seul but est de mettre en relation les deux autres types d’agents.

### Le fournisseur

Finalement, le fournisseur est l’entité en possession des tickets, qu’il devra réussir à vendre aux clients suite à une négociation. Chaque fournisseur est associé à un type de transport (bus, avions, train, etc…) et à une compagnie.

## Instanciation des différents agents

Pour instancier les différents agents, nous avons mis enlace un fichier de configuration qui va définir le nombre d’instances de chaque type d’agent nus allons instancier au démarrage de la simulation.

Une fois ceux-ci instanciés, ils, vont demander un rôle au sein du groupe auquel ils appartiennent et ils vont vivre leur cycle de vie naturellement en échangeant avec les autres agents. Les clients auront le rôle de client dans le groupe client, les fournisseurs le rôle de fournisseur dans le groupe fournisseur. Et pour les brokers, ils vont demander le rôle de broker dans les deux groupes précédents afin de faire intermédiaire.

On peut citer comme exemple le cycle de vie d’un client. Lorsque que celui-ci à réussi une négociation, il se « détruit » mais lance l’exécution d’un autre agent client qui aura d’autre listes de préférences et d’autre contraintes propres à lui.

# Processus de négociation

Le processus de négociation est découpé en 3 parties majeures. C’est ce que nous allons voir dans la suite de ce rapport.

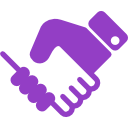
## Recherche de fournisseurs

La première partie de la négociation consiste à mettre en relation des fournisseurs avec un client.

Pour se faire, au début de sons cycle de vie, un client lance un appel d’offres pour un type de transport, ces appels d’offres sont alors captés par les brokers qui vont ensuite le faire transférer aux fournisseurs. Le broker va alors filtrer les réponses des fournisseurs pour récupérer que ceux qui sont en accord avec la demande du client (cf. liste noire). On génère alors un numéro de groupe aléatoire et on répond aux fournisseurs sélectionnés et aux clients qu’un « salon » de négociation a été ouvert pour qu’ils puissent échanger.

Si le client ne reçoit pas de réponse de la part d’un broker au bout d’un certain temps, il relance un appel d’offre avec un nombre maximum d’essai de 6. Si ce nombre est atteint, le client se détruit alors en relançant une autre instance de client.

## Négociation



Nous fournisseurs et notre client étant maintenant en relation, la négociation à proprement parlé peut commencer. Pour ce faire le client envoi en broadcast dans le salon un nouveau message « d’appel d’offres ». Il attend alors en réponse une offre de chaque fournisseur du salon. Chaque fournisseur va alors renvoyer une réponse avec un prix. Une fois toutes les réponses obtenues, ou après un temps maximal d’attente, le client peut départager les offres qu’il à reçut en fonction de ses préférences et contraintes. Il peut alors choisir de sélectionner une offre ou relancer un autre « appel d’offres » avec un maximum de 6 essais. Si le nombre maximum de tentatives est atteint, alors le client ferme le salon et se détruit en relançant une autre instance de client.

## Achat d’un ticket

Dans le cas où le client sélectionne une des offres qui lui est faire, il notifie le fournisseur qu’il est choisi en attendant une réponse de celui-ci. Une fois, la réponse reçut, le client envoie un message au fournisseur contenant le montant du ticket qui avait été convenu lors de la négociation puis ferme le salon en envoyant un message de fermeture à tous les fournisseurs du salon. Chaque fournisseur quitte ainsi le salon et peut de nouveau répondre à d’autres appels d’offres que fera transiter un broker.

# Stratégie de négociation

Afin de varier la nature des échanges de négociation, nous avons associé à chaque client et à chaque fournisseur un system de négociation.

## Stratégie des clients

Chaque client aura une stratégie de négociation sur le choix des offres qui lui sont faites par les fournisseurs en fonction de ses préférences et contraintes. Il aura également une stratégie lors du prix qu’il fait passer dans son appel d’offres.

## Stratégie des fournisseurs

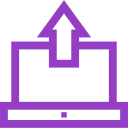
Un fournisseur aura une stratégie qui va déterminer le prix qu’il va choisir pour vendre ses tickets en réponse à un appel d’offres, en ayant connaissance des anciens appels d’offres et des anciennes offres qu’il a pu faire.

## Implémentation des stratégies de négociation

Nous n’avons pas eu le temps d’implémenter différentes stratégies de négociation. Le fournisseur va se contenter d’envoyer un prix aléatoire et le client de sélectionner le prix le plus avantageux. Néanmoins, tout le processus de relance et la création de client et de fournisseurs avec différentes stratégies au sein de la société d’agent à bien été implémenté.

On aurait pu imaginer différentes stratégies, comme un client qui garde constant le prix minimum d’achat d’un ticket, ou une basse constante ou aléatoire de celui-ci. Et de l’autre côté des fournisseurs qui proposent des coûts constants, ou variables dans un sens croissant ou décroissant en observant la variation du prix des offres que fait le client.

# Résultat final

Chaque agent possède donc une fenêtre où l’on affiche son type et le type de service qu’il offre ou recherche. On affiche également ses logs afin d’avoir un aperçu des échanges en cours. On peut également noter le changement de couleurs des différents agents qui définit des états importants. Un broker jaune sera en recherche de fournisseurs pour un client. Ensuite, un client ou un fournisseur en jaune veut dire que l’agent est en négociation. Puis un client vert signifie qu’il a réussi à acheter un ticket et qu’un fournisseur vert a réussi à vendre un ticket. Et un fournisseur rouge veut dire que celui-ci a perdu lors d’une négociation avec d’autres fournisseurs.

## Exemples de négociations

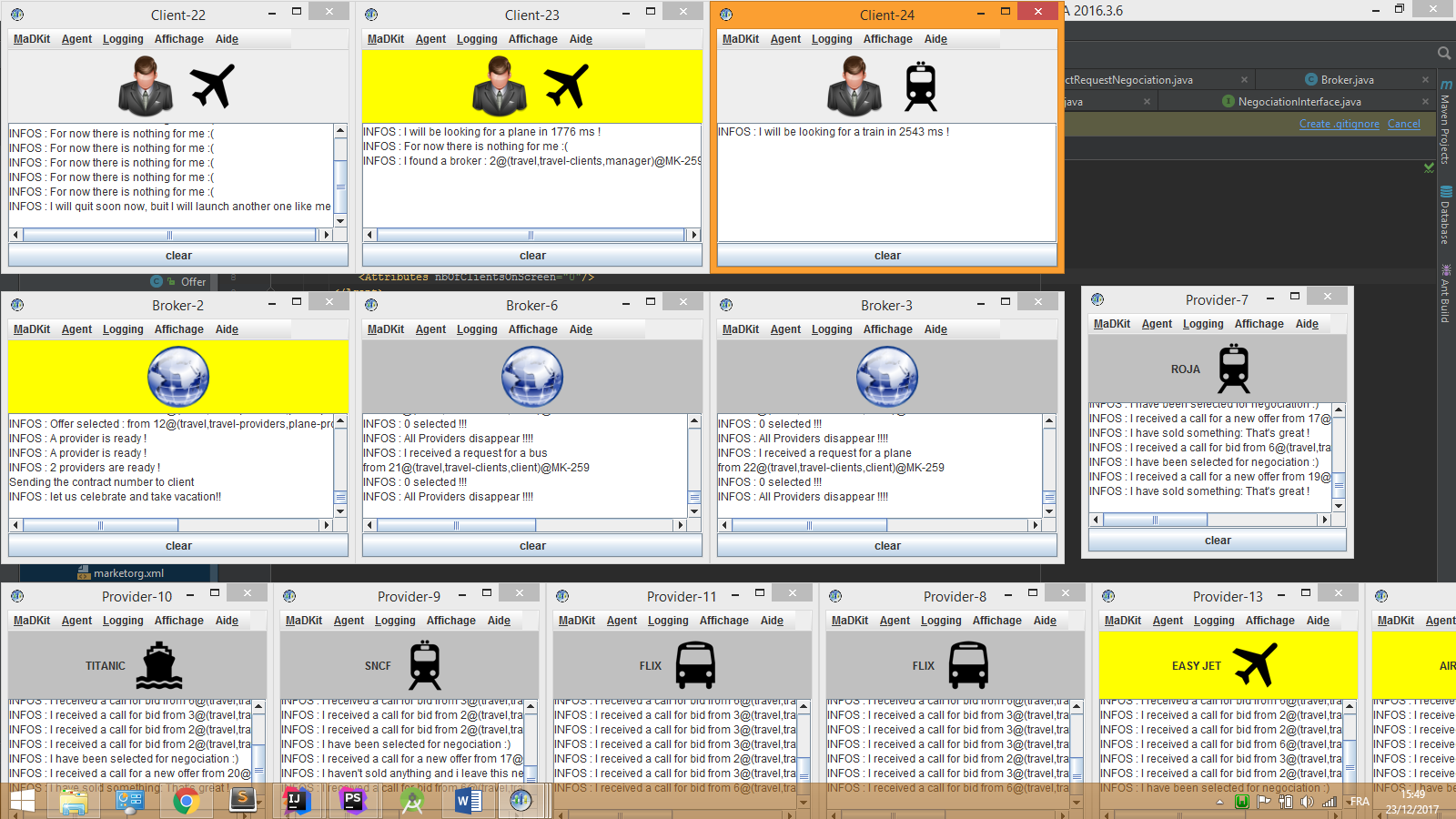


Figure : Exemple de fournisseurs et d'un client en négociation

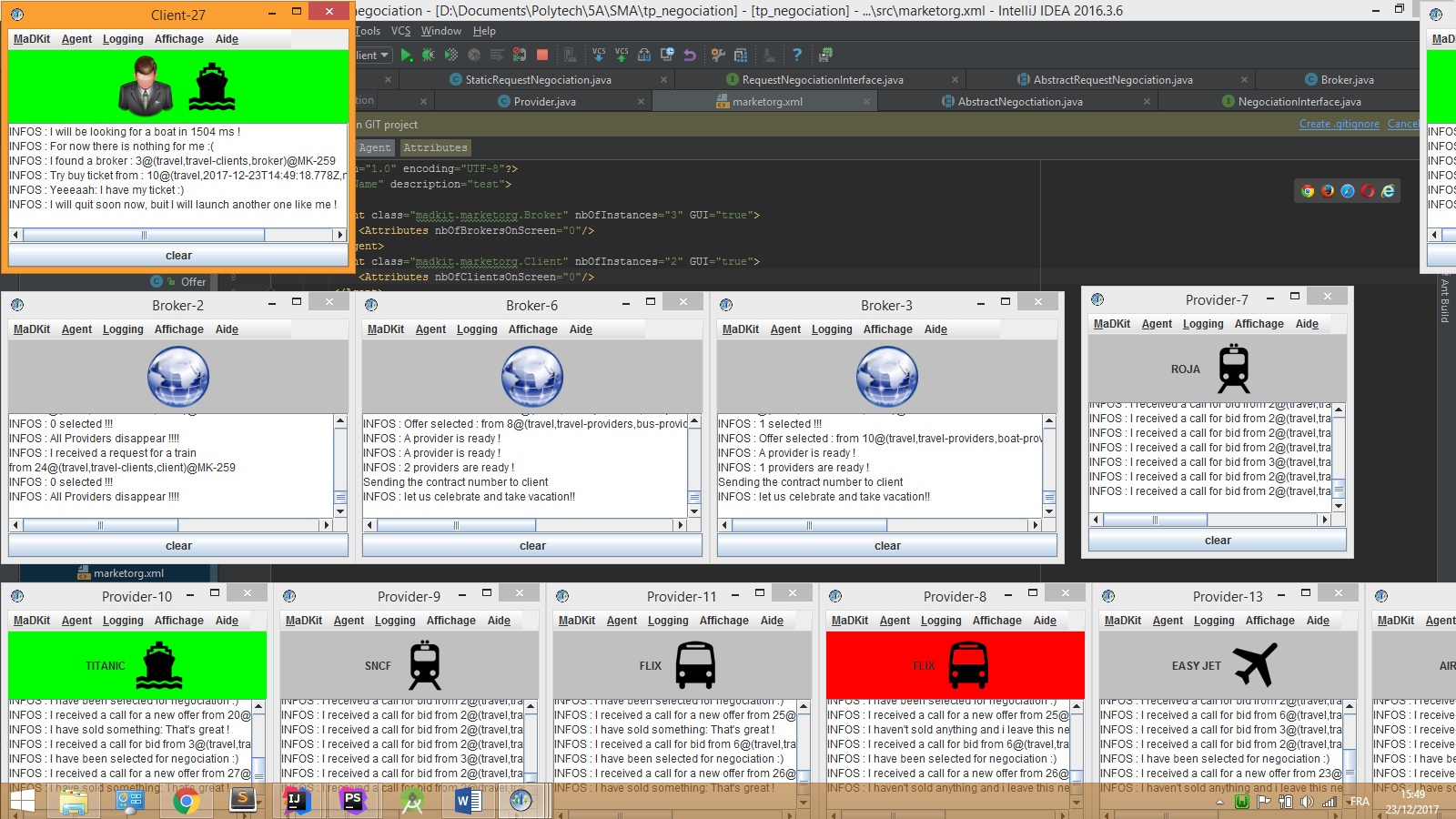


Figure : Exemple de réussite et d'échec de négociation

# Conclusion

Nous avons donc réussi à réaliser un system de négociation mettant en relations différents agents avec leurs propres contraintes et croyances. Couplé avec un système d’affichage et de logs, il est aisé de suivre l’évolution globale du système afin de comprendre le déroulement des négociations en temps réel.

Néanmoins, les négociations restent très basiques, l’évolution suivante serait donc d’ajouter un système plus complexe et varié de négociation des agents afin de les différencier dans leurs prises de décisions. De ce fait, on pourrait, avec une charge de travail conséquente, aboutir à une simulation plus ou moins proche du réel et ainsi permettre d’étudier les comportements et en extraire des conclusions et résultats.