RAPPORT TECHNIQUE
PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
DANS LE CADRE DU COURS LOG792
PROJET DE FIN D'ÉTUDE EN GÉNIE LOGICIEL

SIGNAUX POUR AUTOMATES DE TRADING

CHARLY SIMON
SIMC28069108

DÉPARTEMENT DE GÉNIE LOGICIEL ET DES TI

Professeur-superviseur

Alain April

MONTRÉAL, 13 NOVEMBRE 2016
AUTOMNE 2016

REMERCIEMENTS

Ce projet est la continuité d'un travail promu par le professeur Alain April, professeur à l'École de Technologie Supérieure.

Ce projet est sous la réalisation de Mr. Thomas Maketa, étudiant en Doctorat à l'École de Technologie Supérieure.

Un partenaire de travail qu'il ne faut pas oublier est Mr. Nicolas Hubert de par sa partition active au développement du produit.

SIGNAUX POUR AUTOMATES DE TRADING
PYALOGTRADE ET INTERACTIVE BROKER

CHARLY SIMON
SIMC28069108

RÉSUMÉ

Le marché de la bourse est un monde complexe qui permet aux investisseurs de placer leur argent en achetant et en vendant des actions, des options, les obligations ou autres types de contrats. Pour connaitre le bon moment d'acheter ou de vendre, les investisseurs utilisent des indicateurs boursiers afin de pouvoir appliquer des stratégies de placements.

La compagnie Interactive Broker a été choisi afin de récupérer les données et de placer les ordres. Cette compagnie est ce que l'on appelle un courtier en bourse. Il nous fournit plusieurs plateformes de placements en particulier, Trader Work Station (TWS). Cette plateforme fournit également des interfaces de programmation (API) afin de pouvoir en étendre les fonctionnalités.

PyAlgoTrade est un outil de placements automatisés qui permet à leurs utilisateurs de faire ce que l'on appelle du BackTesting. Nous en avons étudié l'architecture ainsi que les problèmes de qualité de code avec SonarQube. Nous avons implémenté la gestion des options qui n'était pas encore supporté. Nous avons par la suite amélioré le module de BackTesting et commencé l'intégration du module Live afin de permettre l'interaction avec le courtier Interactive Broker en utilisant la librairie Open Source IbPy.

TABLE DES MATIÈRES

Page

CHAPITRE I LES STRATÉGIES DE PLACEMENT EN BOURSE 2
1.1 Le marché de la bourse 2
1.2 Les types de placement 3
1.2.1 Les actions 3
1.2.2 Les obligations 3
1.2.3 Les devises 4
1.2.4 Les options 4
1.3 Les indicateurs boursiers 5

CHAPITRE II INTERACTIVE BROKER 6
2.1 La compagnie Interactive Broker 6
2.2 La plateforme TWS 6
2.3 Les interfaces de programmation propriétaire 7
2.3.1 Les Interfaces de programmation applicative (API) 7
2.3.2 L'API de TWS 8

CHAPITRE III PYALGOTRADE : UN OUTIL DE TRADING AUTOMATIQUE 9
3.1 Un outil de trading automatique 9
3.2 Architecture de PyAlgoTrade 11
3.3 Le module de BackTesting 15
3.4 Les options avec PyAlgoTrade 16
3.5 IbPy : communication avec Interactive Broker 17

LISTE DE RÉFÉRENCES 19

LISTE DES FIGURES

Page

Figure 1 : Conception d'une API 7

Figure 2 : Langages de programmation des API fournit par Interactive Broker 8

Figure 3 Exemple d'implémentation de stratégie 10

Figure 4 : Sommaire de projet dans SonarQube 11

Figure 5 : Vue détaillé de projet dans SonarQube 12

Figure 6 : Matrice de dépendance de PyAlgoTrade 13

Figure 7 : Diagramme de dépendance de PyAlgoTrade 14

Figure 8 : Aperçu du fonctionnement du BackTesting dans PyAlgoTrade 15

Figure 9 : Diagramme de séquence de la création d'une option longue 16

Figure 10 : Diagramme de séquence de la création d'une option courte 16

Figure 11 : Aperçu du fonctionnement dans PyAlgoTrade avec IbPy 17

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

<Texte interligne simple>

Abbréviation	Description
MMS	Moyenne Mobile Simple
MME	Moyenne Mobile Exponentielle
MACD	Mobile Average Convergent Divergent
BandesB	Bandes de Bollinger
RSI	Relative Streng Index
TWS	Trading Worker Station
API	Application Programming Interface

INTRODUCTION

Notre projet est réalisé dans le cadre de notre projet de fin d'étude pour notre Baccalauréats en génie Logiciel à l'École de Technologie Supérieure à Montréal à la session d'automne 2016. Ce présent rapport fait état du travail effectué durant ce projet.

Dans un premier temps, nous aborderons le monde de la bourse afin d'introduire les différents prérequis et concepts boursiers afin de vous mettre dans le contexte de la réalisation de notre projet.

Par la suite, nous introduirons le courtier qui a été choisi dans le cadre de notre projet : Interactive Broker. Nous vous expliquerons les différents outils que cette compagnie fournit publiquement, ainsi que les différentes interfaces de programmation qui permettent l'extensibilité de la plateforme de Interactive Broker.

Et pour terminer, nous vous expliquerons les différentes librairies et applications que nous avons utilisé pour atteindre nos objectifs.

CHAPITRE I

LES STRATÉGIES DE PLACEMENT EN BOURSE

1.1 Le marché de la bourse

La Bourse est un marché financier sur lequel s'effectuent des transactions sur les valeurs mobilières et les marchandises. C'est le lieu où les actions et les obligations, cotées en Bourse, sont vendues et achetées par des investisseurs. Les valeurs mobilières ont une cote qui définit leur prix de vente et leur prix d'achat.

Très souvent, les investisseurs ont recours à des intermédiaires pour participer à la Bourse. Ces intermédiaires peuvent être des banques, des sociétés de gestion ou des conseillers financiers.

L'achat ou la vente d'actions et d'obligations en Bourse nécessite, pour l'investisseur et le vendeur, de donner un ordre de Bourse. Cet ordre contient des informations telles que la nature de l'opération (vente ou achat), la quantité d'actions ou d'obligations concernées par la transaction, la date limite à laquelle l'ordre doit être effectué.

1.2 Les types de placement

Sur le marché de la bourse, il existe plusieurs types de placements. Les principaux sont les suivants :

- Les actions
- Les obligations
- Les devises
- Les options

1.2.1 Les actions

Par placement en actions, on entend généralement l'achat et la possession d'actions cotées en bourse par un particulier ou une société en vue d'en tirer un revenu qui peut prendre la forme de dividendes, de gains en capital ou d'un accroissement de la valeur des actions.

1.2.2 Les obligations

L'obligation est définie comme un titre négociable émis par une entreprise publique ou privée ou par le Trésor (l'Etat) et donnant à son souscripteur le droit de créance sur l'émetteur (l'entreprise).

Une entreprise qui émet donc un emprunt obligataire* sur le marché boursier donne un titre à chaque acheteur. Ce titre (Obligation) donne le droit à son détenteur de percevoir un revenu. Ce revenu est calculé en fonction de la valeur nominale de l'obligation et du taux d'intérêt calculé sur la valeur nominale de celle-ci, et des modalités de remboursement.

1.2.3 Les devises

Le marché des devises est plus communément appelé « Forex ». Ce terme vient de la contraction du terme anglais « Foreign Exchange », signifiant en français « marché des changes ». C'est sur ce marché qu'est déterminé le taux de change d'une devise par rapport à une autre.

Lorsque l'on consulte le marché des devises, on ne peut donc pas consulter le taux d'une devise seule, mais d'une paire de devises. Par exemple, on verra la paire de devises EUR/USD qui correspond au taux de change d'un Euro libellé en dollars. En clair, si le cours de l'EUR/USD est de 1.25, cela signifie qu'un Euro équivaut à 1.25 Dollars. Cette cotation est exprimée en pips.

Le marché des devises est donc par définition un marché international, ce qui en fait l'un des marchés les plus volatiles car il est accessible partout dans le monde.

1.2.4 Les options

Une option est le droit, et non l'obligation, d'acheter ou de vendre un actif dans l'avenir à un cours fixé à l'avance. Ce droit est exerçable une fois au choix pendant une période fixe (cas des options américaines) ou seulement à une date fixe qui est souvent le dernier jour de la durée de vie de l'option (option Européenne).

Les actifs en question, appelés actifs sous-jacents, peuvent être des valeurs mobilières (action, obligation...), des contrats futurs, des indices boursiers, de l'or, et bien sûr des devises. Dans ce dernier cas, on les appelle options de change.

1.3 Les indicateurs boursiers

Un indicateur technique est une série de points utilisée pour analyser les titres boursiers dans le but de prédire l'évolution des cours. L'indicateur est une formule mathématique qui utilise le prix et/ou le volume d'un titre sur une période donnée pour mettre en évidence et exploiter des situations de marché.

Le problème lié aux indicateurs est qu'ils donnent facilement des signaux contradictoires. Certains sont plus pertinents sur des marchés directionnels alors que d'autre sont plus appropriés pour identifier des points de retournement. C'est pourquoi on les classe en deux catégories : les indicateurs de suivi de tendance et les oscillateurs.

Les indicateurs de tendance sont utilisés plutôt pour des marchés directionnels, mais ils peuvent s'avérer dangereux lorsque les marchés sont plats. Ils sont utiles pour entrer dans une tendance établie, trouver des niveaux de supports et de résistance et définir des objectifs.

Les oscillateurs captent les points de retournement ; c'est dans les phases de trading range ou non directionnelles que les oscillateurs se montreront les plus pertinents.

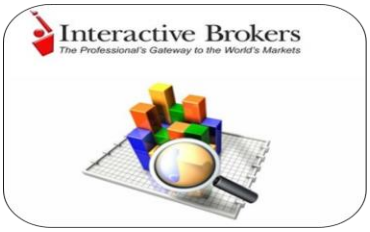
Les indicateurs boursiers les plus commun sont :

- Le Volume
- Les Moyennes Mobile Simple et Exponentielle (MMS et MME)
- La convergence et divergence de moyenne mobile (MACD)
- Les Bandes de Bollinger (BandesB)
- L'indice de force relative (RSI)
- Stochastique lent/rapide
- L'indice de fond de roulement
- Le taux d'encaisse

CHAPITRE II

INTERACTIVE BROKER

2.1 La compagnie Interactive Broker



La compagnie Interactive Broker (IB) a été fondé en 1977 par Thomas Peterffy. Son siège social est basé à Greenwich, Connecticut, États-Unis. Cette compagnie a été classée dans la catégorie investisseurs par Standards & Poor. Elle compte un total de 960 employés qui sont répartis dans le mondes sur l'ensembles des différentes places boursières. Elle est l'une des plus grandes sociétés de courtage avec des capitaux propres s'élevant à plus de 5 milliard de dollars.

Interactive Broker est ce que l'on appelle communément un courtier en bourse (Broker). Un courtier en bourse est une personne intermédiaire qui met en relation deux autres personnes qui souhaitent contracter l'une envers l'autre en contrepartie d'une commission versée au courtier. La rémunération du courtier prend le plus souvent la forme d'une commission comme contrepartie du service de mise en relation.

Interactive Broker axe principalement ses activités sur le courtage en ligne. Il fournit un accès pour ses clients à une plateforme leur permettant de visualiser l'ensemble des cours boursiers. Et par la suite, de pouvoir exécuter leur placement directement via la plateforme.

2.2 La plateforme TWS

La principale plateforme de Interactive Broker est la plateforme TWS : Trader Work Station. Elle permet la récupération en temps réel de l'ensemble des valeurs des marchés financiers dans le monde. Elle permet également de faire la gestion de son portefeuille, d'avoir une synthèse des comptes détaillée, passer des ordres sur le marché tel que les actions, les options, les obligations, les devises. Cette interface est disponible pour toutes sortes de plateforme, aussi boen sur ordinateur personnel que sur mobile ou tablette.

2.3 Les interfaces de programmation propriétaire

Un des principaux avantages de cette plateforme est qu'elle nous fournit une interface de programmation qui permet le développement de fonctions personnalisable à travers une API.

2.3.1 Les Interfaces de programmation applicative (API)

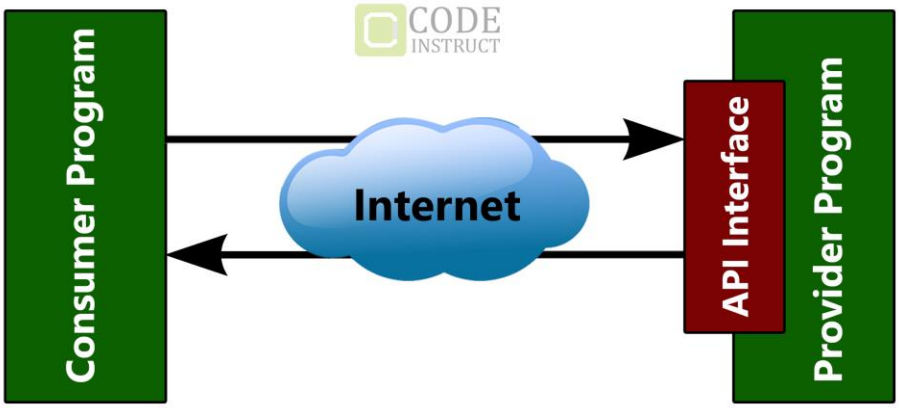


Figure 1 : Conception d'une API

Une API est un ensemble de classes, méthodes et/ou de fonctions accessible qui sert de façade à un logiciel afin d'offrir des services à d'autres logiciel. Le logiciel qui offre cette fonctionnalité s'appelle le fournisseur (provider) et le logiciel qui les utilise s'appelle le consommateur (consumer). Les informations transmises entre les deux logiciels sont habituellement transmises via un réseau internet ou intranet.

2.3.2 L'API de TWS

L'application TWS de Interactive Broker fournit une API qui permet aux utilisateurs d'accéder à toutes les fonctionnalités de la plateforme de Trading. Cette API propriétaire est disponible dans plusieurs langages de programmation : Java, C#, C++, VB, etc.

Malheureusement, cette interface de programmation n'est pas disponible dans le langage de programmation Python.

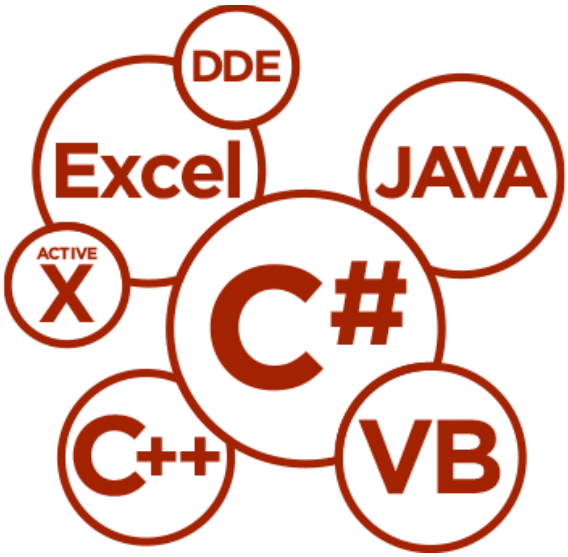


Figure 2 : Langages de programmation des API fournit par Interactive Broker

Le langage Pyhton est un langage très utilisé dans le domaine de la recherche, car celui-ci offre beaucoup de librairie, dites scientifiques, qui permet de faire des calculs que d'autres langages de programmation ne permettent pas. Il est également un langage très performant.

Après plusieurs recherches faites sur le net, nous avons trouvé une implémentation de l'interface en Python. Celle -ci se nomme « IbPy ». Celle-ci-nous fournit toutes les possibilités des autres API.

CHAPITRE III

PYALGOTRADE : UN OUTIL DE TRADING AUTOMATIQUE

PyAlgoTrade, abrégé de **Py**thon **Algo**rithmique **Trad**ing Library, est une application réalisée dans le langage Python qui est orienté sur le BackTesting et qui supporte le trading simulé et en direct.

3.1 Un outil de trading automatique

PyAlgoTrade fournit plusieurs outils pré implémentés de calculs. Il permet de créer des stratégies de placement d'achat et de vente d'action automatiquement. Celles-ci peuvent être, par défaut, basé sur des calculs d'indicateur boursiers tel que vues précédemment : Moyenne mobile simple et exponentielle, convergence et divergence de moyenne mobile, bande de Bollinger, l'index de force relative, etc.

PyAlgoTrade fournit différents exemples d'implémentation de ces stratégies dont nous nous sommes servis pour l'apprentissage de la solution logiciel.

```
from pyalgotrade import strategy
from pyalgotrade.barfeed import yahoofeed
#from pyalgotrade.barfeed import googlefeed
#from pyalgotrade.barfeed import ibfeed
from pyalgotrade.technical import ma

class MyStrategy(strategy.BacktestingStrategy):
    def __init__(self, feed, instrument, smaPeriod):
        super(MyStrategy, self).__init__(feed, 1000)
        self.__position = None
        self.__instrument = instrument
        # We'll use adjusted close values instead of regular close values.
        self.setUseAdjustedValues(False)
        self.__sma = ma.SMA(feed[instrument].getPriceDataSeries(), smaPeriod)

    def onEnterOk(self, position):
        execInfo = position.getEntryOrder().getExecutionInfo()
        self.info("BUY at $%.2f" % (execInfo.getPrice()))

    def onEnterCanceled(self, position):
        self.__position = None

    def onExitOk(self, position):
        execInfo = position.getExitOrder().getExecutionInfo()
        self.info("SELL at $%.2f" % (execInfo.getPrice()))
        self.__position = None
```

```
    def onExitCanceled(self, position):
        # If the exit was canceled, re-submit it.
        self.__position.exitMarket()

    def onBars(self, bars):
        # Wait for enough bars to be available to calculate a SMA.
        if self.__sma[-1] is None:
            return

        bar = bars[self.__instrument]
        # If a position was not opened, check if we should enter a long position.
        if self.__position is None:
            if bar.getPrice() > self.__sma[-1]:
                # Enter a buy market order for 10 shares. The order is good till
canceled.
                self.__position = self.enterOptionLong(self.__instrument, 10, True)
        # Check if we have to exit the position.
        elif bar.getPrice() < self.__sma[-1] and not self.__position.exitActive():
            self.__position.exitMarket()

def run_strategy(smaPeriod):
    # Load the yahoo feed from the CSV file
    feed = yahoofeed.Feed()
    feed.addBarsFromCSV("orcl", "orcl-2000.csv")
#    feed = ibfeed.Feed()
#    feed.addBarsFromCSV("bac", "bac_short.csv")

    # Evaluate the strategy with the feed.
    myStrategy = MyStrategy(feed, "orcl", smaPeriod)
#    myStrategy = MyStrategy(feed, "bac", smaPeriod)
    myStrategy.run()
    print "Final portfolio value: $%.2f" % myStrategy.getBroker().getEquity()

for i in range(10, 30):
    run_strategy(i)
```

Figure 3 Exemple d'implémentation de stratégie

Cet exemple définit une stratégie de base dans PyAlgoTrade. Afin d'implémenter une stratégie, vous devez avant tout hériter de la classe BaseStrategy, puis hériter la méthode __init__() afin de définir vos paramètres. Vous pourrez par la suite redéfinir différentes méthodes de cette classe afin de pouvoir intercepter les événements de création et de sortie d'ordre, d'arriver de données boursières.

3.2 Architecture de PyAlgoTrade

PyAlgoTrade est un logiciel contenant du code presque complétement documenté. Afin d'obtenir une vue d'ensemble de l'application nous avons installé et configuré SonarQube[1]. Ce logiciel d'analyse permet de lancer un examen complet du code source d'une application. Il permet de trouver les différents problèmes dans le code et de vous fournir un suivi sur l'avancement de la correction de ces problèmes, et de s'assurer que le code reste conforme à un ensemble de règle définit, et que l'ensemble des tests couvre l'ensemble des lignes de codes.

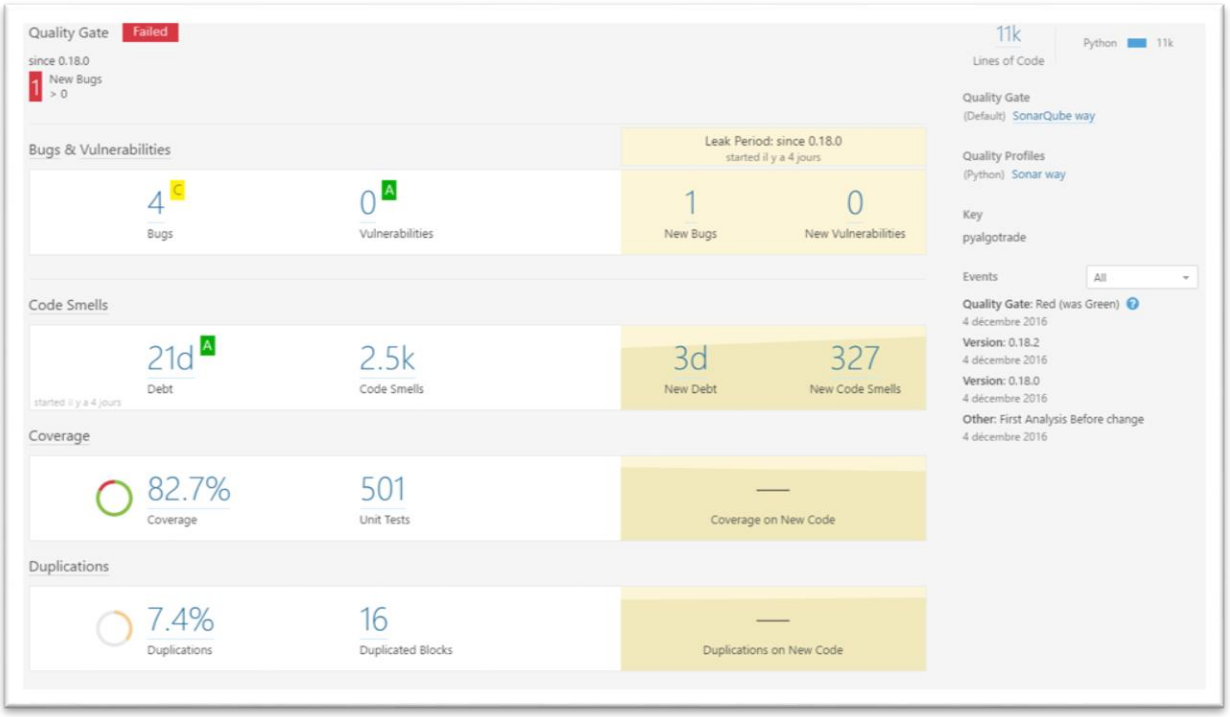


Figure 4 : Sommaire de projet dans SonarQube

Vous pouvez obtenir plusieurs vues détaillées tel que présenté plus bas.

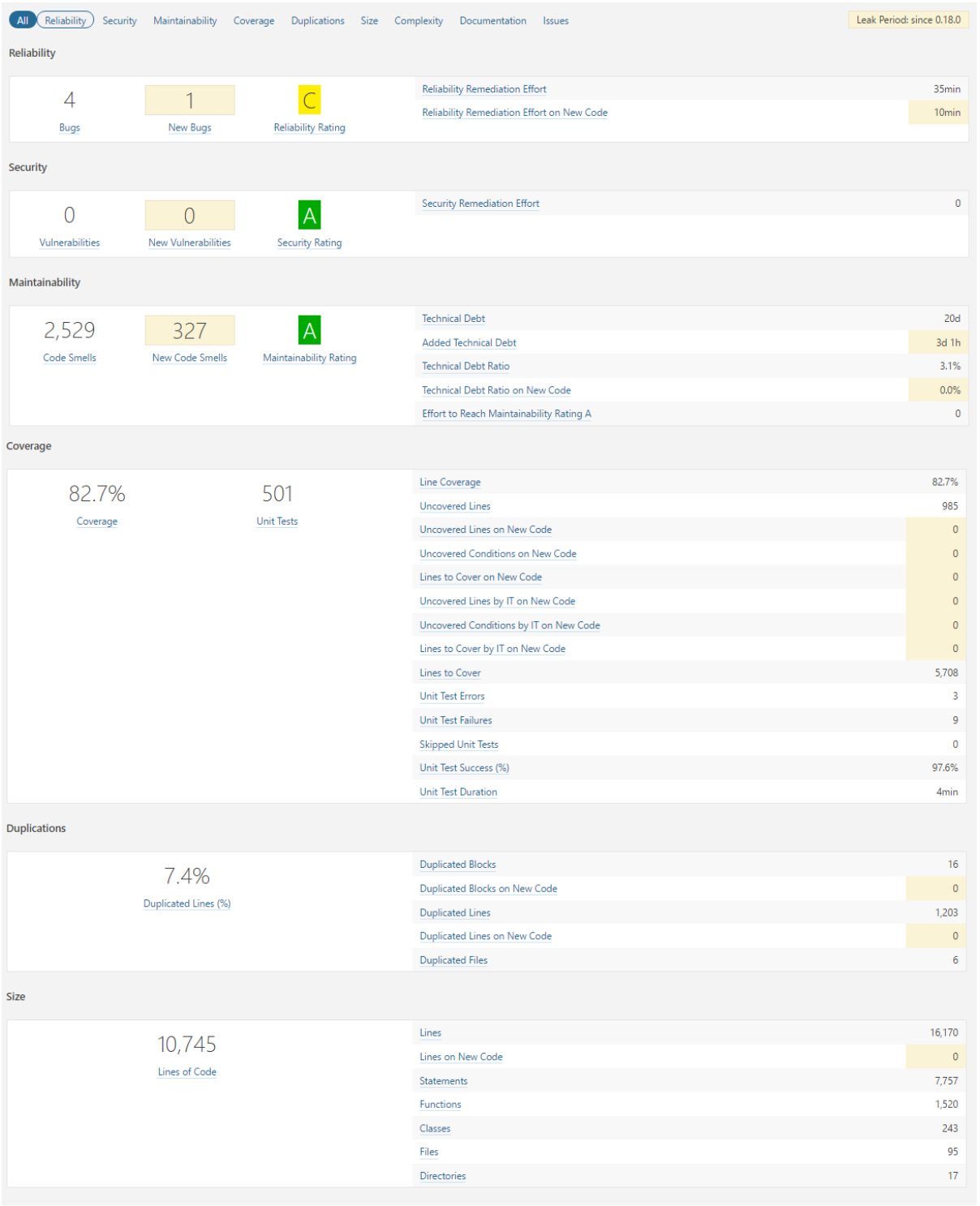


Figure 5 : Vue détaillé de projet dans SonarQube

Étant donné que SonarQube ne permet la génération de matrices de dépendances que pour le langage Java, nous avons effectué des recherches sur le net afin de trouver un outil nous permettant de générer un diagramme de dépendance pour PyAlgoTrade. Cette recherche s'est avéré infructueuse. Nous avons donc pris l'initiative de réaliser cet outil de génération de matrice de dépendance.

Cet outil, nous l'avons nommé PythonDSM. Cet outil a été réalisé dans le langage Java afin de simplifier les développements. Cet outil a plusieurs fonctions, lire la hiérarchie des dossiers de votre application, lire les fichiers dans cette hiérarchie, en extraire les importations, construire une matrice contenant les dépendances, puis trié l'ensembles des fichiers de la matrice par ordre de dépendance. Ce logiciel est Open Source et vous pourrez trouver le lien vers celui-ci dans les références de ce document.

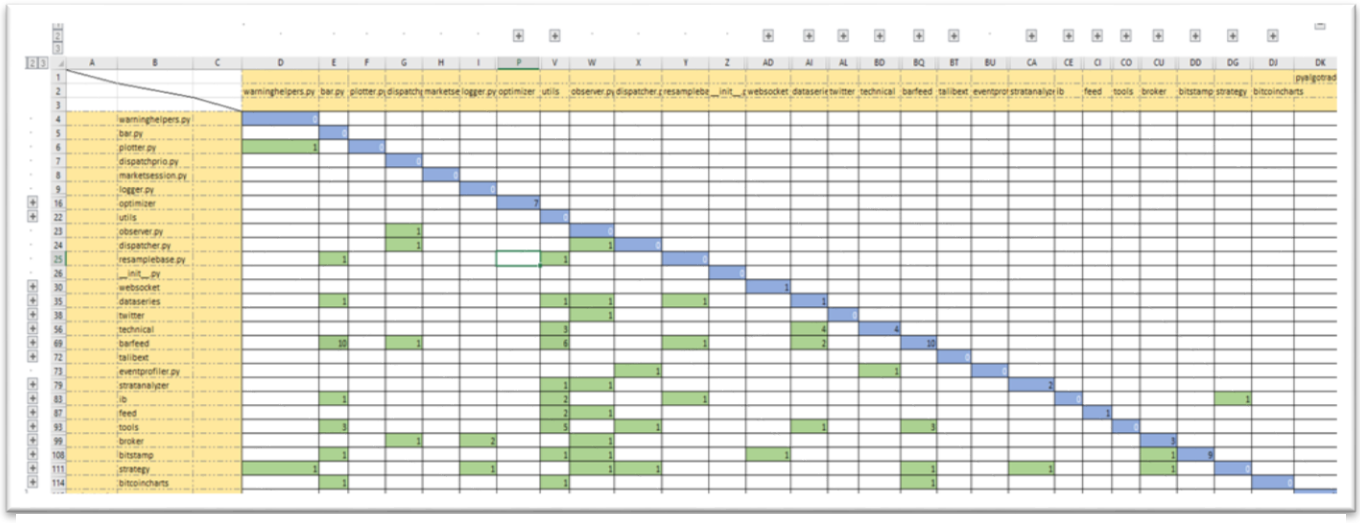


Figure 6 : Matrice de dépendance de PyAlgoTrade

Voici la matrice de dépendance généré pour l'application PyAlgoTrade. Vous pourrez trouver le fichier excel en document Annexe. Comme vous pouvez le constater que l'application ne contient aucune dépendance cyclique et nous n'avons qu'une seule

dépendance mal orienté dû à nos premiers tests d'intégration de IbPy dont nous parlerons plus tard. Ce qui est plutôt rare pour un programme en python.

Cette matrice nous a permis par la suite de pouvoir générer un diagramme de dépendance entre les différents packages afin d'avoir une vues d'ensemble des dépendances du programme.

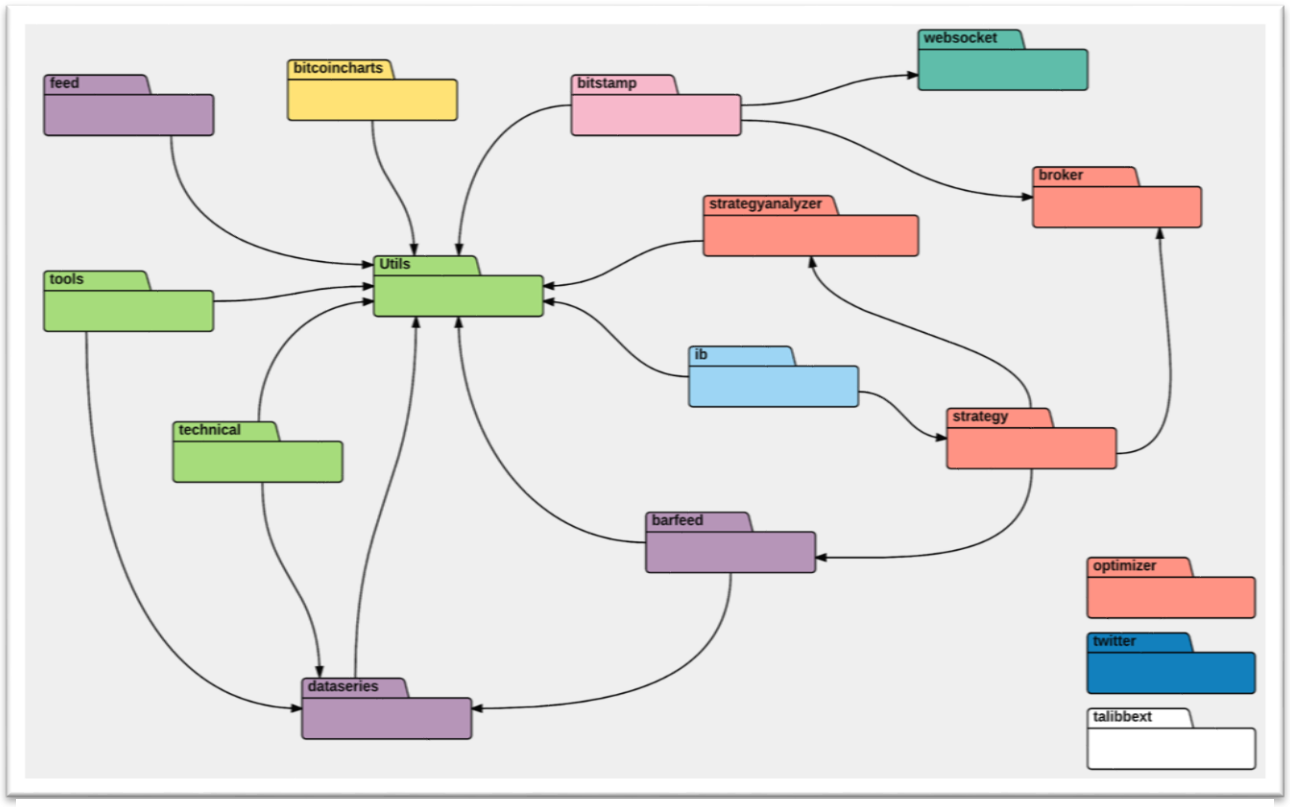


Figure 7 : Diagramme de dépendance de PyAlgoTrade

3.3 Le module de BackTesting

Le module de BackTesting est la partie principale de PyAlgoTTrade. Ce module sert à l'analyse de données historique importé dans le système. Suite à cet import, des données boursières sont simulées avant d'être envoyé à notre stratégie que nous avons précédemment implémentée. Si notre stratégie rencontre les conditions requises pour acheter ou vendre un ordre, alors cet ordre et créé par l'intermédiaire de la classe de base des stratégies : « BaseStrategy ». Une fois cet ordre créer, il est envoyé au module « Broker » afin que l'ordre soit passée et qu'une nouvelle position soit générée. Vous trouverez en annexe les diagrammes de séquence pour la créations d'un ordre ou d'une option.

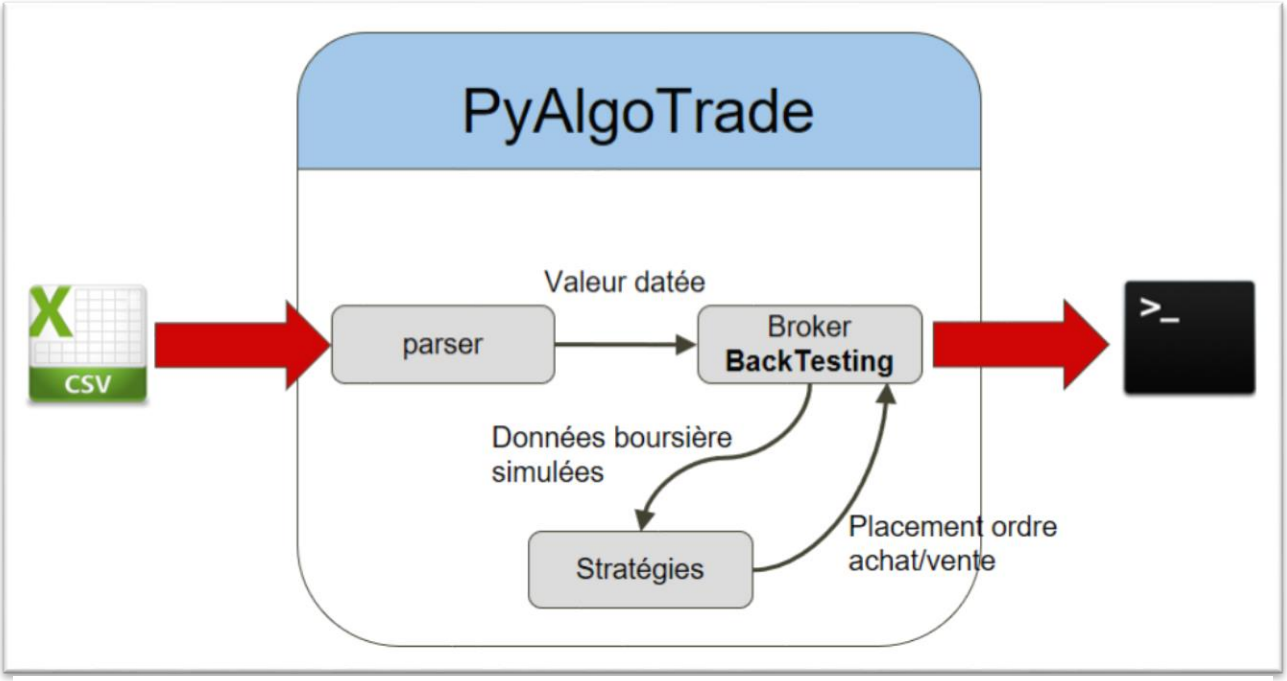


Figure 8 : Aperçu du fonctionnement du BackTesting dans PyAlgoTrade

3.4 Les options avec PyAlgoTrade

De base, PyAlgoTrade ne supporte pas les achats et ventes d'options. Nous avons donc dû pour les besoins de notre projet ajouter ces fonctionnalités. Pour ajouter ces fonctionnalités nous nous sommes basé sur les opérations d'achat et de ventes des actions. La modification s'est faite facilement car nous nous sommes basées sur l'exemple d'implémentation de stratégies.

Voici les diagrammes de séquences que nous avons généré afin de documenté notre travail. Vous trouverez les diagrammes pour les options longues et courtes

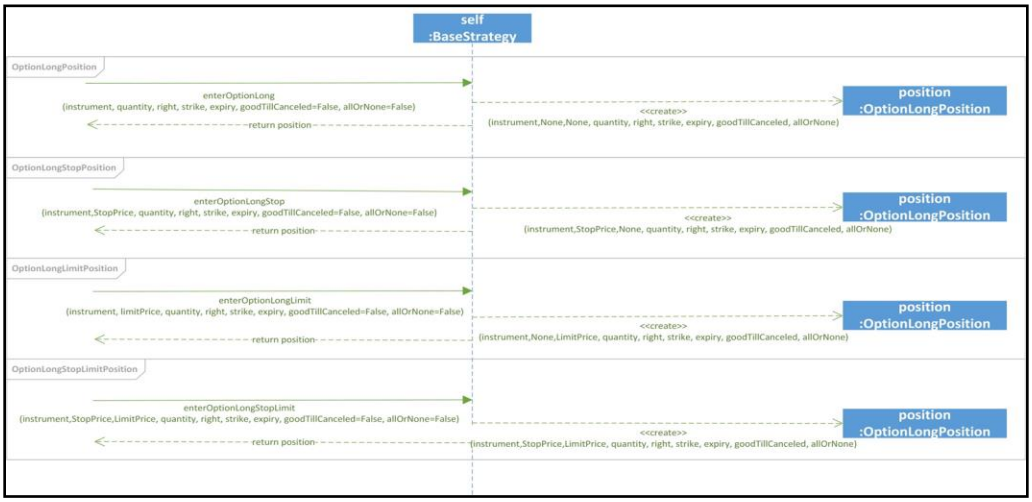


Figure 9 : Diagramme de séquence de la création d'une option longue

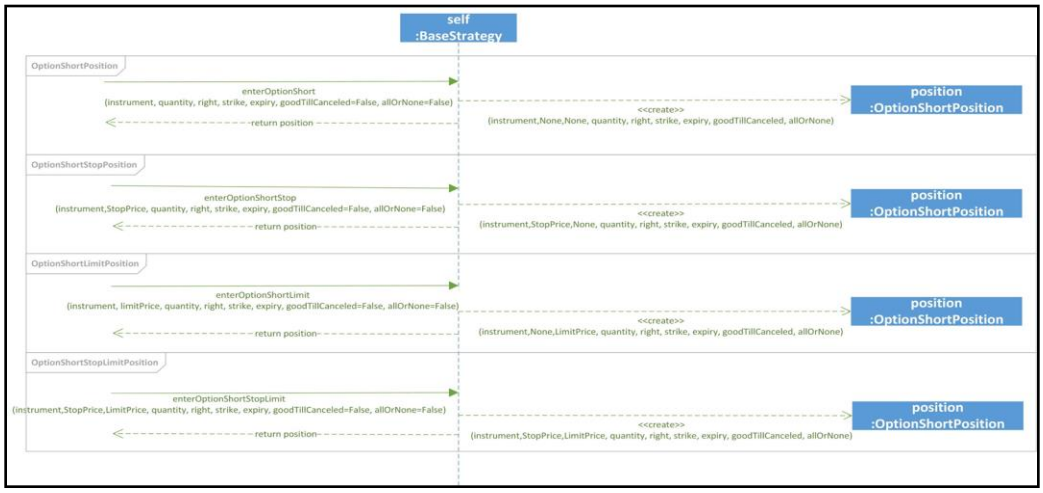


Figure 10 : Diagramme de séquence de la création d'une option courte

3.5 IbPy : communication avec Interactive Broker

IbPy est une tierce librairie Open Source qui a été développé dans le but de permettre aux applications Python de pouvoir utiliser l'interface de programmation fournit par IbPy. Elle permet de faire les requêtes de données en provenance de Interactive Broker, de soumettre des ordres. Toutes les fonctions fournies par les autres interfaces de programmation propriétaire.

Suite à la découverte de cette librairie, nous avons décidé d'analyse la possibilité de l'intégrer à PyAlgoTrade. En nous basant sur les implémentations des autres API tel que celle de Yahoo Finances, de Quandl ou bien de Google Finance.

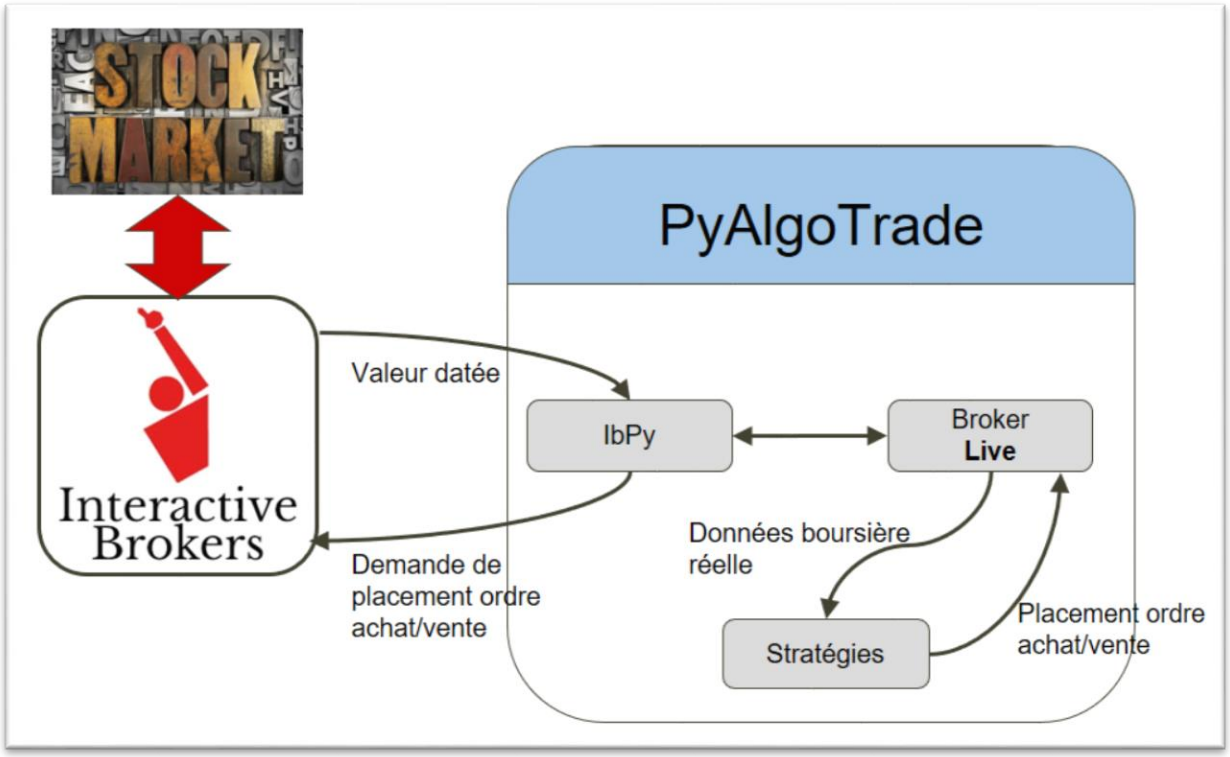


Figure 11 : Aperçu du fonctionnement dans PyAlgoTrade avec IbPy

Les principales modifications que nous devrons effectuer est au niveau du Broker Live. D'après les implémentations présentent dans l'application nous devons créer un Broker par API utilisée.

CONCLUSION

Ce projet a été essentiellement un projet de recherche et de découverte. L'ensemble des programmes utilisés dans ce projet ont été trouvé sur internet et possèdent des licences Open Source.

Nous avons effectué une partie du travail en équipe avec des rôles définis, ce qui nous a permis d'être efficace. L'autre partie du travail a été un travail personnel qui nous a permis de comprendre le monde de la bourse, les différents types de placement qu'il est possible de faire. Nous avons appris l'existence d'un courtier de rang mondial qui nous permet de faire des placements grâce des API réalisé pour plusieurs langages de programmation.

Nous avons pu mettre à profits nos connaissances dans les systèmes d'analyse de la qualité, comme SonarQube, ou même encore développer un outil de génération de matrice de dépendance pour Python. Cet outil développé étant Open Source, il pourra être utile pour d'autre architecte développant des application Python.

Ce projet est un tout de même un succès quant à la demande initiale, même si la définition initiale des besoins a été difficilement atteinte. Ceci nous amené une planification plutôt bancale mais nous sommes arrivés à nos fin.

Dans les prochaines étapes, il nous reste la fin de l'implémentation du module de BackTesting pour la gestion des options, principalement la gestion de l'expiration des options. Il nous reste également l'intégration de la librairie IbPy afin de pouvoir utiliser nos stratégies en direct avec Interactive Broker.

Nous aurons également une partie à considérer pour la réalisation de tests unitaires, afin de s'assurer que notre programme est fonctionnel.

LISTE DE RÉFÉRENCES

https://www.interactivebrokers.com

http://gbeced.github.io/pyalgotrade

https://pypi.python.org/pypi/IbPy2

https://github.com/blampe/IbPy

https://www.python.org

http://www.sonarqube.org

http://sonar.ddns.net:9000

https://github.com/sistorm/pythonDSM

https://github.com/sistorm/pyalgotrade/tree/contratsbacktesting

https://www.quantstart.com/successful-algorithmic-trading-ebook

http://www.comprendrelabourse.com/

http://www.strategie-bourse.com/

https://www.zonebourse.com

http://www.journaldunet.com/