UNIVERSITY OF KENT

Tool for Creating Financial Forecasting Datasets

AXEL GUEDJ
ag535@kent.ac.uk
M.Sc. Networks and Security
11 September 2014

Supervisor:
MICHAEL KAMPOURIDIS

Co-Supervisor:
FERNANDO OTERO

ACKNOWLEDGMENTS

I would like to thank Michael Kampouridis, my supervisor, and Fernando Otero, my second supervisor, for their guidance, comments, suggestions and criticisms that helped me to accomplish with success this project. Moreover, they were always present to help me at any time of the day or night, so as to answer to all my questions.

ABSTRACT

Since several years, the software products have an important part into the financial field. Several techniques were developed over the years, in order to forecast the stock market. One of these techniques is the technical analysis, this technique base its prediction by using past data of companies, in order to determine the trends of these companies. To help the predictions of trends, several genetic algorithms had been developed, in order to have better results with the prediction. One of these algorithms is named Evolutionary Dynamic Data Investment Evaluator (EDDIE), this one permits to help the experts in finance in their quest of investments. This algorithm needs some information about the historic of the company, and derives from these historical information several indicators, in order to operate the EDDIE algorithm. To get these information, the expert has to download a dataset that contains the closing price of the stock, and calculates other indicators derived from the information contained into the dataset. With all these data, the expert creates a file that contains the closing price of the stock, the indicators, and the signal; and submits this file to the EDDIE tool. Creating this file is a time-consuming task for the experts because they have to download a reliable dataset, and apply several formulas in order to get the indicators and signals from the closing price of the stock. This M.Sc. project has to aim to help the experts in finance to generate automatically the several data required by the EDDIE tool. But also, this tool could be used to calculate several technical indicators, for any user who wants to forecast the stock exchange.

LIST OF FIGURES

LIST OF ABBREVIATIONS

CLR Common Language Runtime.

CVS Concurrent versions system.

DLL Dynamic Link Library.

EDDIE Evolutionary Dynamic Data Investment Evaluator.

GP Genetic Programming.

HTML Hypertext Markup Language.

HTTP Hypertext Transfer Protocol.

IDE Integrated development environment.

JVM Java Virtual Machine.

MVC Model View Controller.

MVP Model View Presenter.

MVVM Model View ViewModel.

SVN Subversion.

UI User Interface.

UML Unified Modeling Language.

URL Uniform Resource Locator.

WPF Windows Presentation Foundation.

XAML eXtensible Application Markup Language.

Contents

I. Introduction

I.1. Background and Context

The information technologies are very important in the stock exchange fields, nowadays. Indeed, it exists many software that help the finance experts in their quest to forecast the markets. Over the years, and with the technological progresses, many algorithms were developed for this purpose.

The technical analysis (see section II.2.ii) is one of the most known and used techniques to analyse and determine the market. This technique base its prediction by using the data from the past of a company, in order to calculate several indicators. Since the democratisation of the computers, several algorithms were developed in order to get better results of prediction. One of these algorithms, which interests us for the development of this dissertation and the project, is named EDDIE, and is more detailed in the section II.2.iii.A. EDDIE is a Genetic Programming (see section II.2.iii) which learns and extracts knowledge from the past information of a company (Brookhouse, Otero, and Kampouridis 2014).

The information required by EDDIE are the closing price of a stock, several attributes and the signals. The closing price can be found online on several websites which provide this information (e.g. `https://uk.finance.yahoo.com/`). The attributes are technical indicators (see section II.2.ii.B) derived from the closing price of a stock. And finally, the signal is calculated by looking ahead of the closing price within the n previous days, in order to detect if the price of the stock has increased of $r\%$.

I.2. Motivation

To use the EDDIE tool, the experts have to generate files that contain several data (i.e. closing price of the stock, the attributes, and the signals). To write one of these files, the experts have to download a dataset from a financial website (e.g. `https://uk.finance.yahoo.com/`). And following the download of this dataset, the experts have to apply several formulas in order to get the attributes (i.e. technical indicators, see section II.2.ii.B), and the signals. With all these data, the experts can write a file workable by the EDDIE tool. However, writing this file is a heavy and time-consuming task for the experts.

The goal of this dissertation and particularly the tool named "Fincasting" is to generate several files which can be used by the EDDIE tool. Indeed, the files will contain the several data listed above (i.e. closing price, signal and the attributes). This tool has the advantages to avoid the heavy task of getting a dataset from the Yahoo! Finance website (describe in the section II.3.i), and apply the formulas to the user. To do that, we had developed a tool which automates all the steps that I described, and finally calculates the signal and several technical indicators useful for the EDDIE algorithm.

I.3. Overview of Dissertation

The dissertation is structured in six section. The first section that I just presented you throughout this chapter is an introduction of the project that has aims to create a tool for creating financial forecasting datasets. The second chapter present the literature review, in this chapter I will make a short introduction about the financial forecasting. Following this explanation of the background of the project, I will present an analysis of the existing situation. In the third chapter, I will present to you the design of the system. The chapter four will present the implementation and tools used to develop the software. The fifth chapter present the tests set up into the project. And finally, the last chapter presents the conclusion and discusses about the future works.

II. LITERATURE REVIEW

II.1. Overview

This chapter presents in a first time an overview about the financial forecasting, and presents the two main methods to forecast, which are the fundamental analysis and the technical analysis. In a second time I will do an analysis of the existing situation.

II.2. Background

The main problem for all the people that worked in the field of stock exchanges (speculators, investors and businesses) is to try to forecast the price movements in stock exchanges and commodity products. Many solutions exist to help these people in their choice of investments. The financial forecasting permits an estimate of the future outcomes of a company or a country, this categorisation of problems interests us for the development of this dissertation (Abu-Mostafa and Atiya 1996).

In order to forecast the financial markets, we have to believe that the present, and past events, together with the data about a company, have an impact on the future of the market. However, many economists emit the hypothesis that the change of price is totally independent of the present and past data of a company. This means that the hypothesis to forecast the market is impossible and the changes of price are unpredictable (see the Efficient Market Hypothesis of Burton Malkiel). However, according to some experiences on the markets, done by some economists proved that the changes price is something predictable. The speculators use essentially two main methods in order to forecast and trade, these methods are the fundamental analysis and the technical analysis.

II.2.i Fundamental Analysis

The fundamental analysis consists to go through the financial statements. To do that, this method analyses the financial statements (revenue, expenses, dividends), the new products, the research, the management, the competitive advantages, and the competitors on the market for a company. This technique uses all these data in order to try to have an overview of the future performance of the company. This method has the advantage to perform on industries or on the economy as a whole. This technique is more focued on the analysis of the economic well-being of a financial entity and not only on the price movements of a company. The fundamental analysis aims to respond to simple questions like if the revenue of the company is growing, if the company is making profit, if the company is able to repay its debts, if its position on the market permits to be superior to its competitors currently and in the future.

II.2.ii Technical Analysis

The technical analysis aims to try to determine the trends of the future outcomes for a company. However, this method uses a different approach as compared to the

fundamental analysis, because this technique does not use any information about the company or a commodity. This methodology admits that several patterns exist in historical data of a business and these patterns will repeat (Otero Fernando 2014). To determine the trend of the future outcomes of a company, this methodology uses the past data of the market and the company. These data will be used in order to calculate and determine several patterns. That is why it is important to have reliable data from the past. One of the data that interesting us for the development of this dissertation and the project "Fincasting", is the closing daily closing price of a stock. With this information from the financial past of the company, it permits us to derive useful information, essentially in order to calculate several technical indicators (Kampouridis and Tsang 2010). The daily closing price could be obtained online on some website (e.g. *Yahoo! Finance* 2014). In this method, there are two different types of indicators. The first one is by using chart patterns, and the other one is by using technical indicators.

II.2.ii.A Chart Patterns

A chart pattern is a pattern which is formed within a stock chart. It indicates a trading signal, or a sign of a future price movement into the market. The economists use these patterns to determine the trends and to try to determine the signal for buying and selling stocks. This technique is based on historic trend of the market, that is why the chartists try to determine the trading opportunities by using these patterns contents into the stock charts. The problem is there are no chart patterns which are 100% reliable to determine the trend or the moment to sell or buy stocks. That is why chart patterns are more seen as an art that a real science.

II.2.ii.B Technical Indicators

The technical indicators are mathematical calculations based on several data that we can find within a given dataset (e.g. price, volume). The result of these indicators are used to anticipate the future changes in price of the stocks or forex (*Technical Analysis From A To Z* 2006). The indicators are used in order to try to determine the signals for buying and for selling stocks. The choice of the indicators is very important, because it can make a difference on the prediction (Martinez-Jaramillo 2007). For our study, each technical indicator uses two different periods, known as short-term and long-term period. These two periods are used in order to compare the result of the technical indicator calculated with the long-term period and with the short-term period. However, it is not guaranteed that the result of a technical indicator for a short-term period is more efficient that the result for a long-term period (Kampouridis and Tsang 2010). The technical indicators that interests us for the development of this dissertation are the following:

- **Moving Average (MA)**: The moving average is a widely used in technical analysis, because this technical indicator helps to smooth out the price of the stock by removing the "noise" from changes in the price of the action. This technical indicator is a trend indicator with delay, because this one is based on the past

prices. The main utility of the Moving Average is to identify the trend direction, in order to determine the support and the resistance levels. This calculation can be performed from the closing price of the stock of a company, and yield a value that can be used to try to anticipate the future changes of the price (*Moving Average - MA* 2014). The figure 1 display the MACD chart, which is a kind of Moving Average.

- **Trade Break Out (TBO)**: The breakout trading is one of the most used indicators in technical analysis. This indicator tries to determine the suitable period to buy stocks. The indicators that permit to define when it is interesting to buy is when the price of a stock "break out" above a past indicator level or to sell when the price "break down" under this level indicator (*Breakout Trading: Technical Analysis Primer* 2014). The figure 2 display a chart that using this technique.

- **Volatility (Vol)**: The volatility displays on a chart the changes in the price of the market. This indicator is interesting and very helpful in order to try to determine the potential reversal into the market. This indicator is based on the price of the closing stock. The volatility is represented by a standard deviation from the historical price of the stocks. This indicator measures the amount of variability or the dispersion around an average (*Volatility-Based Indicators* 2014). The figure 3 displays an example of volatility on a chart.

- **Momentum (Mom)**: The momentum is the difference between the price of the stock at the closing for the day and the price of the closing L days ago. This indicator measures the rate of the rise or fall of the prices of a stock. The momentum indicator is a very useful indicator, because it determines the strength or weakness of the price of the action. Moreover, this indicator is one of the most efficient indicators during financial crisis, because the prices rise more often than they fall (*Momentum And The Relative Strength Index* 2014).

In the Appendix "Technical Indicators", we can find all the formulas used in the project "Fincasting". We have chosen these indicators because they have proved that they are the most useful during periods of financial crisis and also during previous works (Kampouridis and Tsang 2010, Otero Fernando 2014, Martinez-Jaramillo 2007).

II.2.iii Genetic Programming

The genetic programming is an automated method inspired by the mechanism of the natural selection established by Charles Darwin. This technique permits to a software to learn by using an evolutionist algorithm, and optimise bit by bit a population of other software to increase the degree of adaptation and realised a task requested by a user. The Genetic Programming (GP) is traditionally represented in memory as a tree structure, and the tree is evaluated in a recursive manner (see figure 5). A genetic decision tree could be seen as a set of rules, every tree node has a condition and every leaf has an action (e.g. buy, sell and so one). The GP is commonly used in financial forecasting. Indeed, these algorithms are used in order to make predictions about companies.

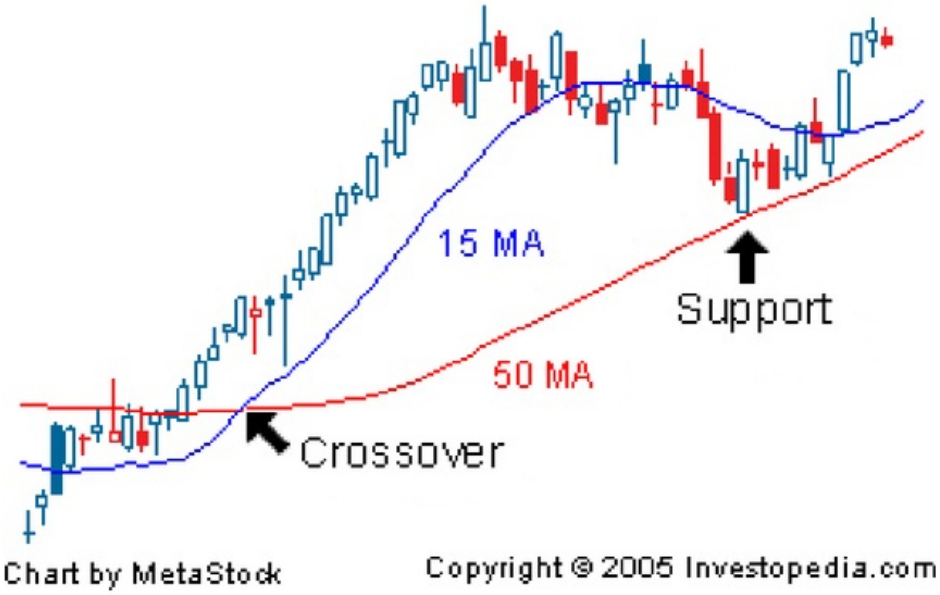

Figure 1: Moving Average Convergence Divergence

II.2.iii.A EDDIE

EDDIE or Evolutionary Dynamic Data Investment Evaluator is a genetic programming, and used as forecasting tool. Indeed, this algorithm is used in financial forecasting, but also in sports betting (Tsang, Li, and Butler 1998). The first version of EDDIE (named EDDIE - 1) was applied to horse racing. But also, this genetic programming is suitable for forecasting. Indeed, betting on horse races is quite the same as investing in a financial market. Some investments are highly risky, like a bet on a race. However, the use of EDDIE in financial forecasting does not replace forecasting experts. The main objective of this algorithm is to generate and respond to hypothesis. All of that, in order to help the investors in their quest to find the best company to invest in. The hypothesis are like the following form: *"Will the price of share X rise by r% within the next n trading days?"* (Tsang, Yung, and Li 2004). To respond to these hypothesis, the algorithm calculates by looking ahead of the closing price of a stock for a time horizon of n days, and determine if there is an increase of the price of the closing price of $r\%$. The user (expert) has to specify a set of factors that the system has to consider. With the factors specified by the expert, the EDDIE algorithm will generate a genetic decision tree that the user could approve or reject, as we can see in the figure 5 (Tsang et al. 2000). The set of data used by the EDDIE are the closing price of the stock, a number of attributes and signals. The attributes are technical indicators like explained in the section II.2.ii.B. The indicators to take into account depend of the choice of the expert. And the signal is the result of the hypotheses *"Will the price of share X rise by r% within the previous n trading days?"*. It exists several versions of the EDDIE

Figure 2: Trade Break Out

algorithm, the latest version is the version "EDDIE 8" (Kampouridis and Tsang 2010).

II.3. Analysis of the Existing Situation

As we have seen in the section II.2.ii ("Technical Analysis"), I explain that we need a reliable dataset in order to calculate the technical indicators. This dataset must contain some important information (mainly the daily closing price of a stock). This information can be found on some website, for this dissertation, we are using the `https://uk.finance.yahoo.com/` website to get the information and thus create a reliable dataset.

II.3.i Calculate the Technical Indicators Manually

If a person that is working in the stock market field (e.g. trader, speculator, and so on) wants to calculate some technical indicators, he/she needs to get the daily closing price of a stock from the Yahoo! Finance website. The problem is that to get these data could be a heavy task for this person.

Figure 3: Volatility

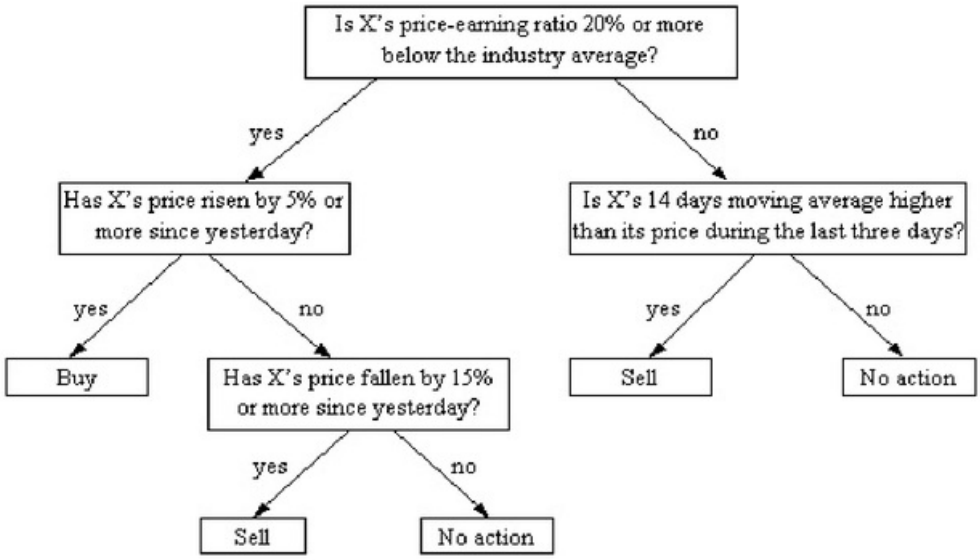

Figure 4: The EDDIE algorithm represented as tree structure

The quest to get these data is composed of several steps. In a first time the user has to go on the `https://uk.finance.yahoo.com/` website. The user normally will be directed on the main page. After that, the person has to enter the ticker of a company and click on the button look up, in order to get more information about it, as we can

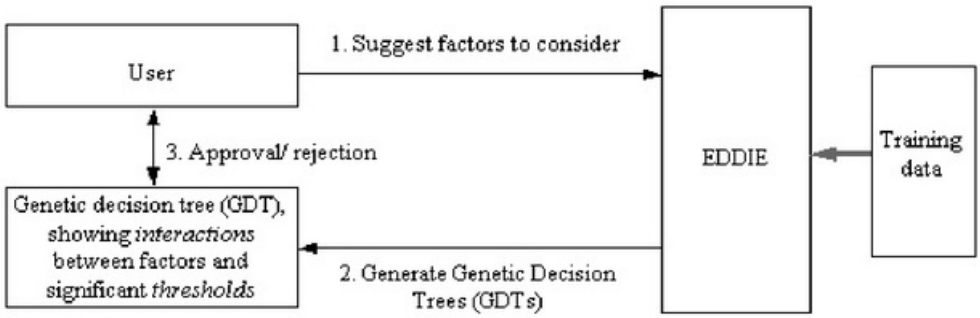

Figure 5: The role of the EDDIE algorithm

see on the figure 6. In a second time, the user will arrive on a page that contains all the essential information about the company selected (e.g. price of the stock, chart that contains the price of the stocks during a defined period, and so on). After that, the user has to click on the button "Historical Prices" which is on the left navigation bar of the website (see figure 7). Following this, the user arrives on a page that displays several information about the company (e.g. the price at the opening of the stock exchange, the highest price of the stock, the lowest price of the stock, the price at the close of the stock exchange, the average volume and the adjusted closing price), all these information are available for the day, week or for the month. To get these data the user has to specify the date range to which he wants the information, and select one option to specify if he wants these data for each day, week or month during the range that he selected (see figure 8). And finally, to get all these information display, the user has to scroll down the webpage (which could be very long if the date range is large), and click on the button "Download the Spreadsheet", as we can see in the figure 8. By clicking on this button, the user will download a CSV file that contains the dataset. After getting the dataset, the user can apply some formulas in order to calculate some technical indicators. For calculating the signal, the user has to spend a lot of time adjusting the value of $r\%$, in order to find the best value of this one, that respects the range of percentage of positive signal that the expert wants. After getting the signals, attributes and the closing price of the stock, the user can generate a file usable by the EDDIE tool.

II.3.ii Existing Software

It exists many different software that do financial forecasting. Most of them are used as a web based software, this solution has the advantages to give the portability of the solution on every platform both desktop and smartphone or tablet. However, most of these companies proposed their solution with a renewal subscription paid every month/year.

II.3.ii.A ProRealTime

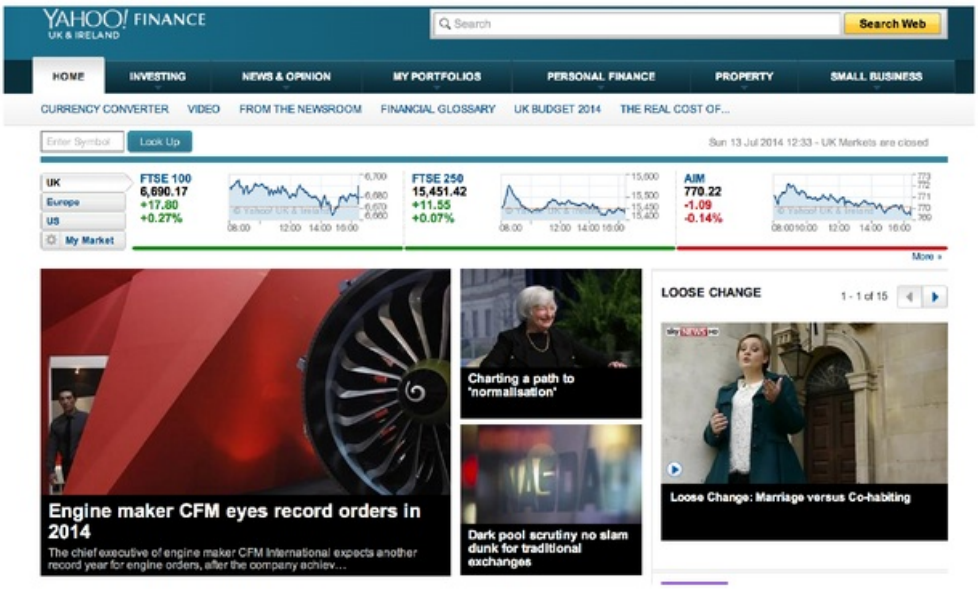

Figure 6: Yahoo! Finance, Main Page

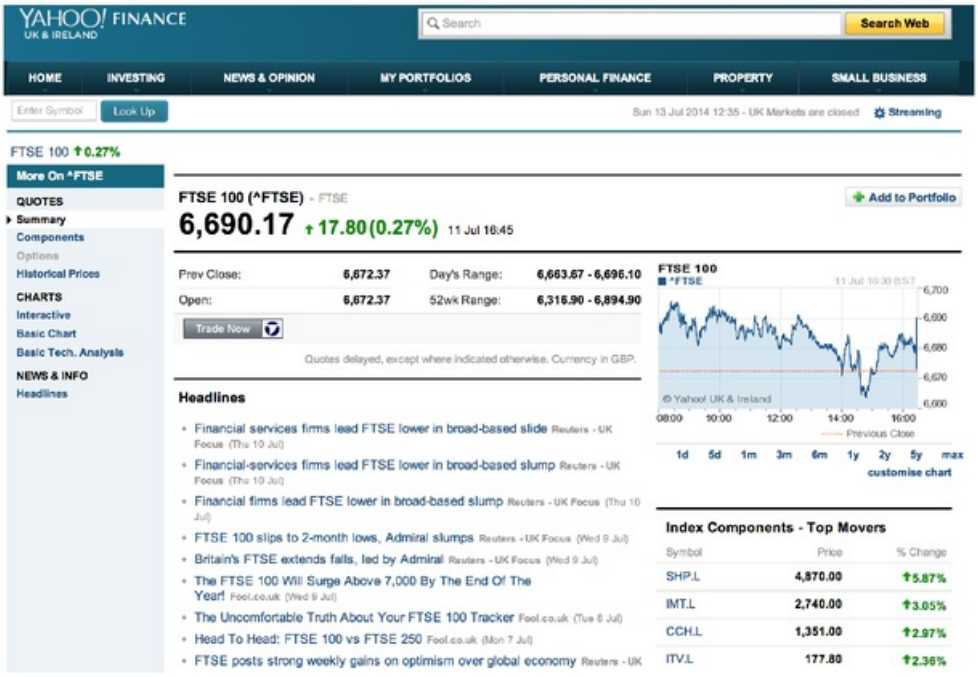

Figure 7: Yahoo! Finance, Information about a company

This software is proposed as web solution. However, this solution may dissuade some user for two reasons. The first reason is that a user who wants to use this solution in order to have some information must create an account on their website. The

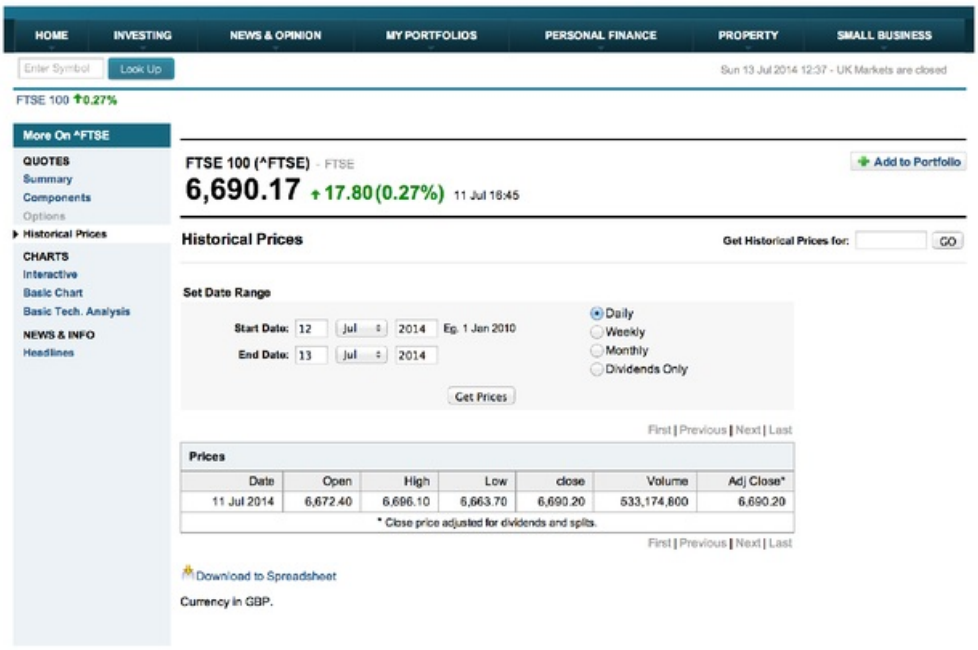

Figure 8: Yahoo! Finance, Historical Information

second reason is, if the user wants to use the service provided by "ProRealTime", he must have to install a plugin for his web navigator. Moreover, the plugin could not be installed easily because it uses a Java Virtual Machine (JVM) and the installation of this one is mandatory to use ProRealTime, which could be a drag for the user that does not have a JVM installed on their computer.

At first sight, this tool looks like a desktop application as we can see in the figure 9. This tool displays around hundred indicators to the user, for any company and for Forex market. In addition of displaying indicators and charts, this tool allows the users to automate the trading systems, the system is allowed to place orders for the user of the software even if the workstation is off. Another interesting feature proposed by the solution is the export of the indicators or trading systems.

II.3.ii.B IStock

IStock is a web based solution asset trading platform. This solution aims to be easy and simple to use. This software could be used to trade forex currency, commodities, stock market, and so on. Just like "ProRealTime" this solution uses a JVM, which means that the JVM is mandatory for the use of IStock.

As we can see in the figure 10, this application looks like a desktop application. The main advantages of this solution is that it provides a simple graphical user interface. It allows the customisation of the interface and permits the displaying of different charts

Figure 9: ProRealTime

and around thirty technical indicators. Moreover, this solution permits the export of historical data of any stock and even the import of from several format (.csv, .xls).

II.3.ii.C Our Project's situation

II.3.ii.C.a What we will have more

The tool named "Fincasting", will have the main advantages to be free and open source. Firstly, free in order to allow all the people who work into the stocks exchange (experts in finance), or even for a personal usage to do some forecasting with the technical indicators proposed by the tool. It is the main targeted public of this project because it permits to have the necessary information required by the EDDIE tool, easily, and automatically; and all of this for free, which is not the case of most of all solutions presented before. The second aim of this project is to do a software easy to use, because the aims of this project is to automate and facilitate the life of the experts in finance in order to use the EDDIE tool. That is why this software will display only the main

Figure 10: IStock

technical indicators to export and not complicate the display and the user experience with no frills, which complicate them. Another advantage, that we have compared to competitor is the non creation of account for using our solution. Moreover, this software does not need a constant access to internet to use it compared to the other softwares, since the software downloads and saves the information required by the system and does not require any connection to the internet after that (if the software has saved a dataset that contains the necessary information for the periods requested by the user). Besides, this project is open source in order to permit the continuity of the project, and thus the adding of features into the project in the future. Finally, this project is different to other projects because it was designed to assist the researchers and other people who work in the stock exchange fields in their quest of trying to determine the opportunity for buying and selling stocks.

II.3.ii.C.b What will not be covered

In the project "Fincasting" there are several features available on other software that I described before that will not implemented into this version of the project. The main reason is because the aims of this project is totally different of the software products presented above. Indeed, "Fincasting" is a tool that helps the experts and researchers in finance to generate files that contains the information required (see section II.2.iii.A) by the EDDIE tool in order to perform it and generate a genetic decision tree, useful for the experts in their quest of forecasting. The most interesting feature available on the other projects presented before and that will not be covered in our tool is the displaying of the chart, but it is not the utility of our project. However, it could be a feature added for the future of the project (see section VI.2). Another feature not covered in this version of the software and also significant is the number of technical indicators provide by the software which is equal to six in our project, whereas some

tools (e.g. “ProRealTime”) provide more than hundred of technical indicator, and could improve the results of prediction of the EDDIE tool. Finally, the feature that permits to place orders for the user, which is available in “ProRealTime”, will not be implemented. However, this is not the aim of the project “Fincasting”, since the targeted public of this project are experts and researchers in finance.

II.4. Conclusion

As was seen throughout this chapter, in order to use EDDIE tool, the expert in finance has to create a reliable file that contains several data (i.e. closing price of the stock, attributes, and signals). Creating this file manually is a time consuming task, that is why this tool is important for the experts in finance, because it permits to automate this task. Moreover, as was seen in the section that talks about analysis of the existing situation, our tool does not have any competitor on the market.

III. Analysis and design of the system

III.1. Overview

In this chapter, I will present in a first time, all the features included into the system. Following this, I will talk about the constraints of the system. In a third time, I will present the methodology used to design and develop the project. After that, I will talk about the general architecture of the system. And finally, I will show the global view of the project by designing several diagrams.

III.2. Description of the objectives

The tool named "Fincasting" is a desktop application, and it is available on Microsoft Windows. This software has aim to help the experts and researchers in finance in their quest to forecast the market. To do that, this tool generates files that could be exploited by the EDDIE tool, in order to try to predict the stock exchange.

The files generated by the "Fincasting" tool contain three important data for the operation of the EDDIE tool. These three data are the closing price of the stock, several attributes and the signal. The signal is calculated by looking ahead of the closing price within the n previous days, in order to detect if the price of the stock increase of r%. Finally, the attributes are the results of the technical indicator (see section II.2.ii.B). "Fincasting" allows the users of the software to have six different technical indicators for any company available on the stock market. These data could be exported into two different files (Train and Test file) for each technical indicator calculated. Obviously, the user could change some parameters according to his/her exigency (e.g. long term period, short term period, percentage of distribution of the export into each file). The formulas for all the technical indicators presented in this dissertation are available in the Appendix (Technical Indicators). The six technical indicators are the following:

- Moving Average
- Trade Break Out
- Filter
- Volatility
- Momentum
- Momentum Moving Average

III.3. Functional requirements

Functional Requirements
Get automatically a reliable dataset from the Yahoo! Finance website.
Calculation of each technical indicators (i.e. moving average, trade break out, filter, volatility, momentum and momentum moving average). And also, the calculation of the signal.
Implement a dynamic process that calculates the best value of r (the percentage of increasing price), for a company. According to the range of positive percentage specified by the user, and the value of n (the horizon time).
Export files which contain the good values for the closing price of the stock, the attributes, and the signals. This file contains these data for each day, according to a number of given days.
The exported files (i.e. *txt*) have to respect a standard in order to be open by Excel.
Change the preference parameters.
The system has to take into account the preference parameters of the software (i.e. periods, number of days we want the information, and so on).
Implementation of a favourite list, which contains all the companies that the user wants the information. An error handling, in the case that the company does not exists. And management of the favourite list.
Search bar to add a company to the favourite list by specifying the ticker of the company. Or specified the name of the company (only for the company that are parts of the FTSE).
Let the choice to the user for the value of r between the value find by the dynamic process, or the value specified into the preferences, in order to calculate the signal for the exporting file.

III.4. Non functional requirements

One of the non functional requirements is to have a minimum network bandwidth, in order to get the dataset from the Yahoo! Finance website. Another non functional requirement is that the save of the company contains into the favourite list, and the save of the preference of the software specified by the user. Finally, this tool must be easy to use, that is why the biggest constraint for this project is that software must be ergonomic and the easiest to use for the user.

III.5. Functional constraints

The functional constraints permit to highlight the sensible points, and the constraints that could prevent the use of the system. This section is important because it helps us to define the architecture of the project presented in the sections III.7 and IV.4. Indeed, by presenting the sensible points that could prevent the use of the solution change totally the implementation of the software in order to avoid the constraints and limitations of the system.

III.5.i Provider of the dataset

The biggest constraint of this project is the manner to get a reliable dataset, and therefore, find a provider to get this dataset. For this project, we use the website Yahoo! Finance which is one of the most known websites for getting information about stock exchange. However, using this provider is our biggest weakness of the project for two main reasons. The first one is this website is our only provider of this kind of information, in the case that this website close or does not provide these data, we will have to find another provider to get the same information. The second constraint, is in the case of the Yahoo! Finance website changed the manner to get these data. In this case, our software will be out of service during the implementation of a new provider that getting these data. That is why the architecture of the implementation of the provider is important, because it must be easy to change our provider into the software, in order to limit the time that the solution is out of service, and thus avoid that our users leave the software for another.

III.5.ii Internet

Even if the dependance with internet is low, the software requires an access to internet in order to get the datasets from the Yahoo! Finance website. Indeed, this constraint is very important in the case that the user does not have an internet connection. In this case, the software could not be used by the user and could frustrate him, and even in the worst case, the user could leave the software by another one.

III.5.iii Desktop Application

One of these constraints, that we impose to ourself for this project is the compatibility of the solution. Because this software is only available on desktop, and not online like most of the existing software that I talk in the section II.3.ii (“Existing Software”). Moreover, this solution is only available on the Microsoft Windows platform. This constraint limits the number of users of our solution, therefore, most of the people that use a computer use Microsoft Windows as operation system. I will explain more about this choice in the section IV.2.iii.

III.6. Methodology

Throughout the conception and the implementation of the project "Fincasting", I would like to establish some rules to follow in order to fulfil all the requirements of the project. That is why I am interested about the software engineering methodology. Although, the format of a team of one person is not really a context where we can apply and follow the software engineering methodology, I tried to apply the base concept of the methodology by extracting the interesting points and apply them to my workflow.

III.6.i Agile Method

The Agile method are groups of project practice for the development of software, which could be applied to several types of projects. These methods have to aim to increase the productivity and enhance the rate of success of software projects by simplifying the workflow. All these methodologies have in common the *Agile Manifesto* (Beck et al. 2001). These techniques involve more the client and permits to have a better reactivity in his/her request. Indeed, these techniques aim to satisfy the client. For the development of a project that does not use the Agile method, the developer team has a single deadline to develop the software. However, this technique presents a lot of fails for the fulfilment of the project, that is why the Agile method uses short development cycle. If we take the example of the Scrum methodology (which is the methodology which I used throughout the implementation of the project), it uses short development cycles named "sprints". These "sprints" may have a duration of several hours to four weeks (most of the time a sprint has a duration of two weeks). At the end of the cycle (sprints), the new features developed will be added in the new release version, and the team planned the features that will be developed for the new sprint. Another main feature of the Agile method is to not design the whole system, because it is complex and even impossible to design the entirety of a system. The Agile method advocates to design and develop feature by feature, in order to avoid the problems. These techniques permit more flexibility, and the success of the project.

III.6.ii Incremental Development

For the implementation and the design of the application, I was focused on developing and designing a feature. This technique is named incremental development methodology, this technique gives a better flexibility of the design, which permitted me to roll back on some features in order to improve it or to totally change the specification of the feature. Which would have been more difficult if I had done the whole design of the application at the beginning of the project. Also, throughout the project, I kept in mind two rules of the Agile method which are:

- **KISS** (Keep it simple Stupid): this rule is applied on everything, from the planning to the implementation. If we take the implementation essentially, the developer has to well separate the code and every class has to do what it is supposed to do and nothing more. This technique has for advantage to have a code more reliable, more easy to maintain, and more easy to debug.

- **DRY** (Don't repeat yourself): this rule has to aim to keep in mind to do not forget that the duplication of code is inadvisable and a lost of time. By following this rule, the developers will have to spend time by refactoring the code in order to optimize the source code, and therefore, increase the performance of the system.

III.7. Project Architecture

To operate, the application needs a reliable dataset in order to calculate the technical indicators. To get this reliable dataset, this software uses the database of the Yahoo! Finance website. It is the only period during which the software needs an internet connection, in order to get a dataset on the Yahoo! Finance website, as we can see in the figure 11. The database from Yahoo! Finance allows us to get several important data, like the price of the stock exchange at the opening/closing of the market, the number of sold stocks during the day/week/month, the highest/lowest price of the stock.

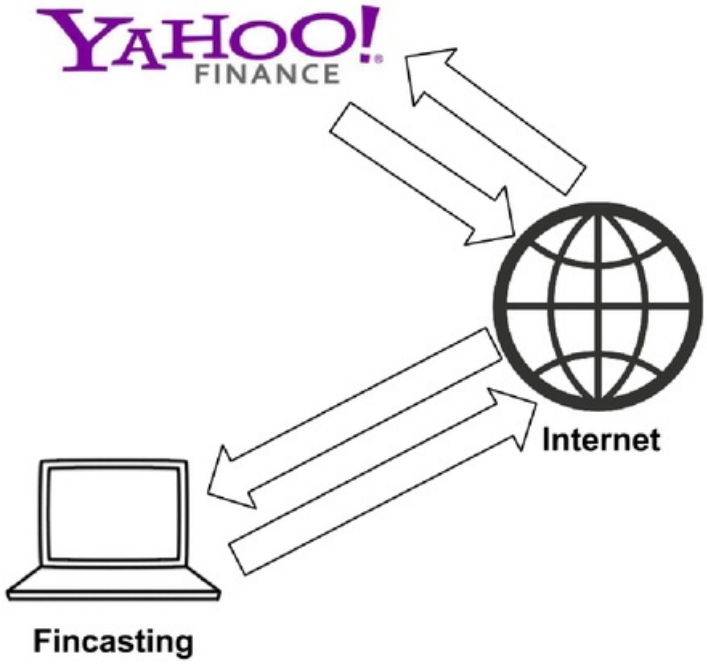

Figure 11: Global Architecture

III.8. Global view of the project

In order to define and have a better understanding of the system, we used tree types of Unified Modeling Language (UML) diagrams (e.g. use case, sequence diagram and activity diagram).

III.8.i Use case diagram

One of the challenge for the developers is to understand what the user wants and fulfil the requirements imposed by him/her. Most of the time, the requirements are not clearly specified to the developer, or the developer does not understand clearly what the user wants. That is why we are using the use case diagram for the project "Fincasting". Indeed, the use cases have been proven in most of the development project to avoid the problem of understanding between the developer and the clients in order to create the system that the user wants. The aim of the use case diagram is to define how to use the software, and thus describe the functional requirements. Each use case defines a scenario that defines how to use the system.

III.8.ii Sequence diagram

The sequence diagram represents a scenario of use and presents the interaction between the user and the system according to a chronological order. The sequence diagram permits to hide the interaction between the objects under the scenario of a Use Case diagram. Most of the time we represent the main actor at the left of the diagram, and the other actors at the right. The aim to this diagram is to represent the interaction between the actors or objects. The time is represented by the vertical dimension of the diagram, and permits to display the action sequences during the time, and thus specify the born and the death of an object.

III.8.iii Activity diagram

The activity diagram represents graphically the actions done by a method, or the progress of a use case. This diagram permits to represent the outbreak of events according to the state of the system. The activity diagram is quite the same as the organization chart. An activity diagram corresponds to the algorithmic representation of a use case.

III.8.iv Project modeling

As we can see in the figure 12, the only main features available for the user is to manage its favourite list of companies and do an export of the calculated technical indicators. In order to add a company to the favourite list, the system (Fincasting) has to request the Yahoo! Finance service, in order to see if the company exists, and display an error message, or add the company to the favourite list, as we can see in the figures 13, 16 and 19. As we can see in the figure figures 14, 17 and 20; in order to export the results of the technical indicators, the system has to check if it has a dataset on the hard drive, and checks if it is reliable for the calculations of the technical indicators. In the case that the dataset is not up to date, the system will get a dataset directly from the Yahoo! finance website. Finally, the system integrates a dynamic process in order to get the best percentage of increasing price (r) since the last n days. To do that, the system needs a reliable dataset, otherwise, the system will get the dataset directly from the Yahoo! Finance website. Following that, the system could try to determine the

best value of r for the range of percentage of positive signal that the user wants (see figure figures 15, 18 and 21).

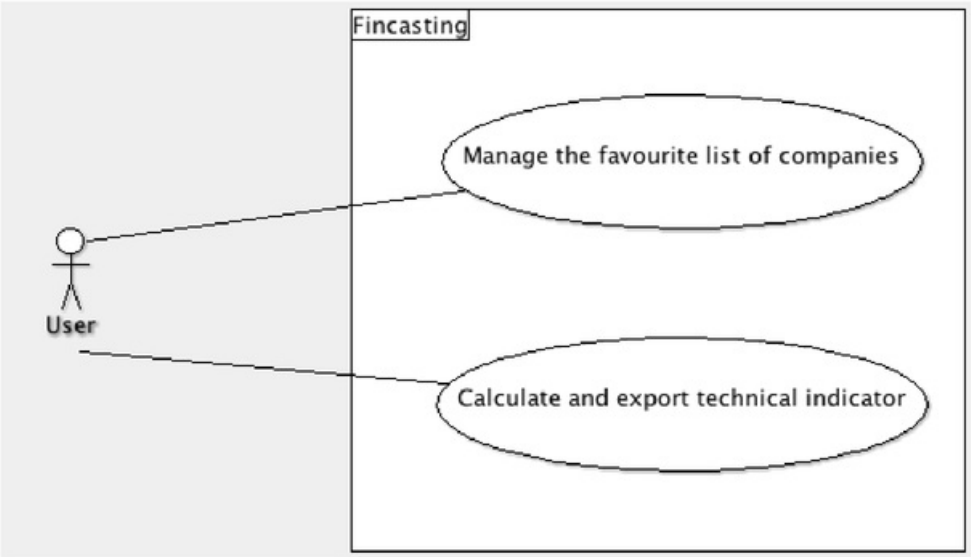

Figure 12: Use Case General

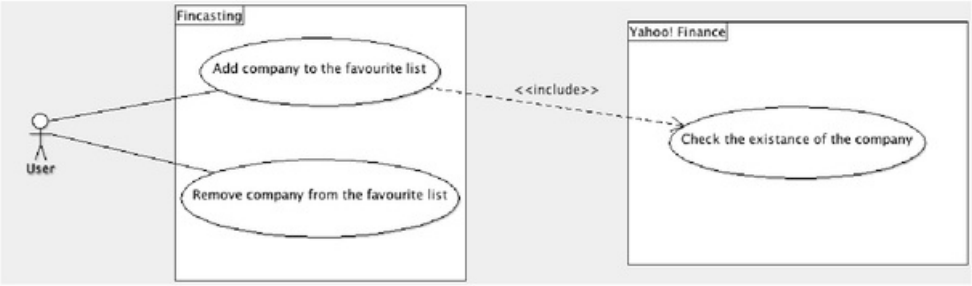

Figure 13: Use case diagram for adding a company to the favourite list

III.9. Conclusion

Throughout this chapter, we have seen that the system has to fulfill several functionalities. Moreover, the system has several constraints that we have to take into account in order to develop the system and meet its requirements. In order to have a better productivity and have a better chance of success to develop the project, I set up several methods (i.e. agile method). Finally, we have seen all the scenarios and functionalities that will be covered by the system, by the help of the several UML diagrams.

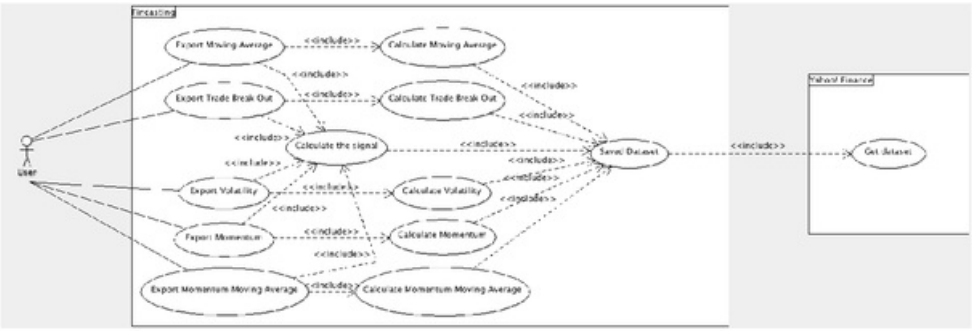

Figure 14: Use case diagram for calculate and export technical indicator

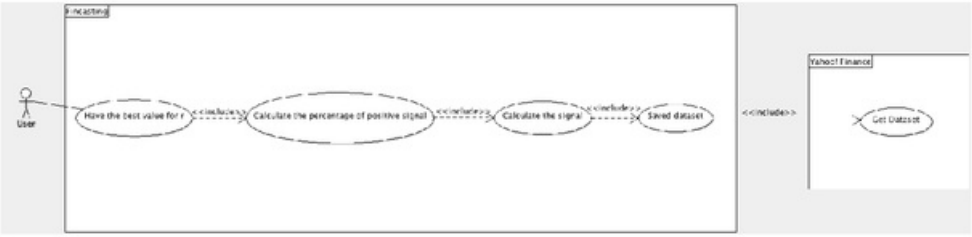

Figure 15: Use case diagram for the dynamic process

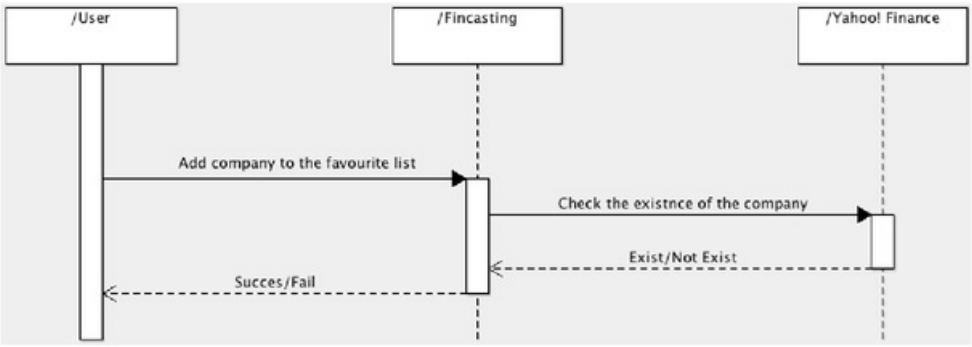

Figure 16: Sequence diagram for adding a company to the favourite list

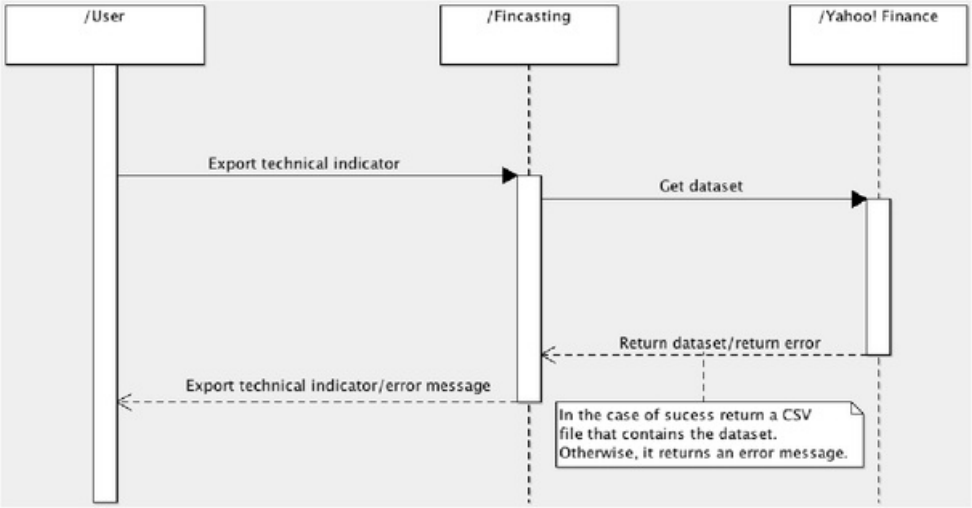

Figure 17: Sequence diagram for calculate and export technical indicator

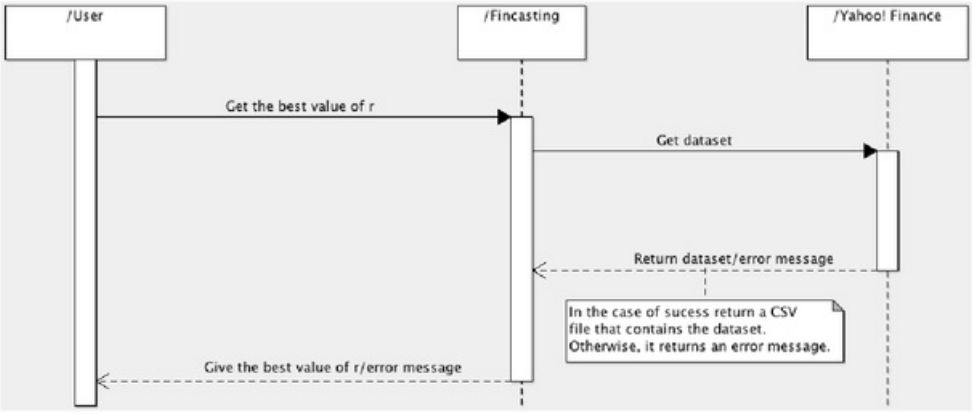

Figure 18: Sequence diagram for the dynamic process

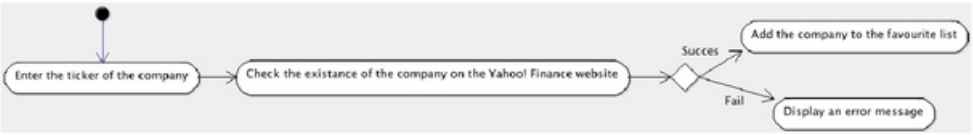

Figure 19: Activity diagram for adding a company to the favourite list

Figure 20: Activity diagram for calculate and export technical indicator

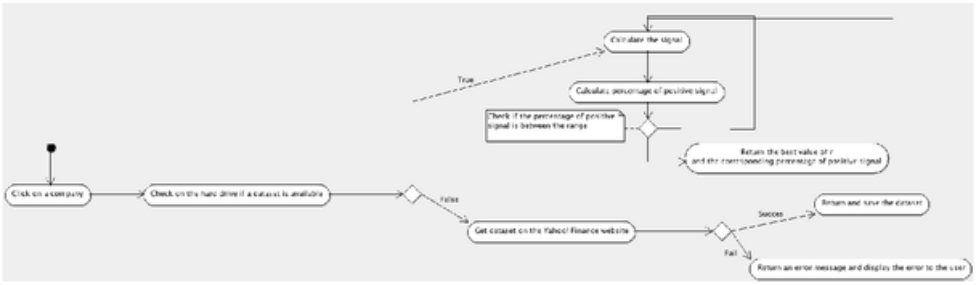

Figure 21: Activity diagram for the dynamic process

IV. Implementation of the system Fincasting

IV.1. Overview

In this chapter we will see the implementation of the system. In a first time, we will see the technologies (i.e. language used, framework, pattern and library) used to developed the system. After that, I will talk about the solutions available to get a dataset from the Yahoo! Finance website. Following that, I will present the architecture of the project. Finally, I will talk about all the tools used to developed and design the project "Fincasting".

IV.2. Technology

For the development of the software "Fincasting", we use a lot of different technologies, tools and libraries to develop this one. That is the thing we will see throughout this section.

IV.2.i C#

The C# is an oriented language programming with strong typing. This language was created in 2001 by the company Microsoft and essentially by Anders Hejlsberg who is the creator of the Delphi language. It was originally created in order that the Microsoft .NET platform had its own language. And in order that this language uses all of the capacities of the platform. This language has a lot similarities with the JAVA and C++ languages, and takes the general syntax, and most of their concepts.

The objects in C# are automatically cleaned and removed by a garbage collector, just like in the JAVA language and many other high level and modern languages. Moreover, just like the JAVA language use a JVM to execute a JAVA application, the C# use the Common Language Runtime (CLR) to execute a C# application. Obviously, just like the JVM the CLR is mandatory (Greene and Stellman 2013). However, it has a lot of differences with the C++, because the C# does not permit the manipulation of the pointer like in the C++ language (it allows but only into code marked *unsafe*) and it does not support the multiple inheritance. Finally, the C# is extremely dependant of the .NET platform. Indeed, even the native types correspond to those of the .NET platform. The latest version of .NET as I write this dissertation is the version 4.5 which is the version used to develop the software.

I have chosen the C# because this language proposed a large catalog of features directly integrated in the framework .NET, which is a big advantage compared to the other known languages (e.g. C++). Another advantage is its performance which is not the most efficient compared to the C language, but are not insignificant compared to other language (Sestoft 2010). Furthermore, Microsoft advises the developers to implement some interesting patterns that permit a better reusability of the code and a better portability of the project on other project that using the C# language, even on

another Microsoft technology. For example we could imagine a desktop application porting on a web-based, or even on a mobile application. All of this could be done easily, because the language C# uses the framework .NET which is quite the same on all the Microsoft platform.

IV.2.i.A WPF

Windows Presentation Foundation (WPF) is a graphical specification integrated into the .NET platform since the version 3.0. To create graphical interface, WPF uses eXtensible Application Markup Language (XAML) to define a page of the application. XAML is a declarative language that is used quite like an Hypertext Markup Language (HTML) page for the developer. WPF is preinstalled with Windows Vista and can be installed on Windows XP (since the version SP2) and Windows Server 2003 (Greene and Stellman 2013). WPF is an overlay of DirectX (DirectX is a library used for the programming of multimedia applications), and is used to develop graphical applications, except video games, and aims to replace Windows Forms. Moreover, WPF is not only used for desktop application, but also in a runtime environment of web page named "Silverlight" (Sells and Griffiths 2007).

The technology WPF permits to structure the system "Fincasting", and defines the User Interface (UI) of the tool. Indeed, just like explained above, the technology WPF permits to create graphical software on the Microsoft Windows platform.

IV.2.i.B MVVM

The most specific pattern set up in the project "Finacasting" is the Model View ViewModel (MVVM). But what the difference between the most known and renowned Model View Controller (MVC), it is what I am going to explain throughout of this section.

The developers have most of the time a problem for developing the user interface of a software, because the biggest work for that is to well cut the different tasks and other underlying problems like the security, the multithreading, the internationalisation, etc... There are some patterns that aim to resolve all these problems the most known are the MVC and the Model View Presenter (MVP). However, these patterns are sometimes complicated to implement, test, and maintain in time, according to the platform that there are implement (Smith 2014). The developers need a platform where they can simply build a UI, and where they can easily test the code. It is exactly the thing that proposes Microsoft with the graphical specification WPF, and the pattern created by Microsoft for the occasion, the "MVVM" (Sells and Griffiths 2007). Just like the MVC or the MVP the MVVM is decomposed in three components (see figure 22):

- **Model**: It is the data that we want to display into the View.
- **View**: It is where your elements of the user interface are designed.

- **ViewModel**: It is the logic part of the pattern. It is the link between the View and the Model. It could be consider like a special control that convert the data from the Model to the View.

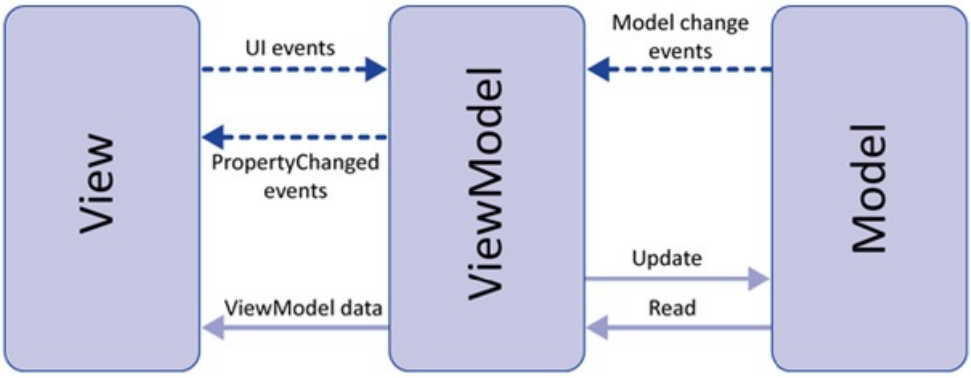

Figure 22: MVVM Pattern

But what are the differences between the previous patterns and the MVVM, because they fulfilled the same features. Take for example a software with a user interface, the developer has a tendency to write a lot of UI logic code behind the UI itself. In this case, it is hard to test and maintain the code UI. MVVM resolved this problem because it uses the concepts of "binding" and "event". The "binding" and the "event" permit to automatically update the view when a data from the model is changed and vice versa, when an element from the UI is changed, it is automatically reported into the ViewModel. This technique has several advantages, like the development in parallel of the Model, the View, and the ViewModel. Moreover, the ViewModel can be easily tested. But the most interested advantages is the level of abstraction of the View, and thus we can use the same ViewModel for several View or other project that using the same technology (Greene and Stellman 2013). This pattern is more and more spread in other languages and frameworks, which is the case with the Javascript and the framework Knockout.

The pattern MVVM was implemented into our system (i.e. "Fincasting"), and permit us to have a better maintainability of the project. Indeed, if we look at the architecture of the project (see section IV.4, and the code of the project itself), we can see that the project is divided into three entities. The Views are defined into XAML files (see section IV.2.i.A), and defines all the components of each View. The Models are defined into libraries (i.e. Dynamic Link Library (DLL)), and permits to get and manage the data. And finally, the ViewModel class that made the transition between the data from the Model to the View, but also get the event from the View (e.g. click on a button, fulfilled of a field, and so on). Moreover, it permits us to porting the project easily on another platform using the Microsoft .NET technology, just like I explained in the section IV.2.ii.

IV.2.ii Library used

To develop the software "Finacasting" we use several libraries in order to develop it.

The main library used in the project is "*MVVM Light Toolkit* 2014", this library permits to its user to implement easily, and accelerate the implementation of the pattern MVVM in several Microsoft technologies (e.g. WPF, Silverlight, Windows Store and Windows Phone). This library helps the developers to separate the *View* from the *Model* in order to create applications which are cleaner and easier to maintain and extend in the time. Moreover, this library allows to create testable applications easily. Finally, this toolkit allows to use easily the tool "Blend" which permits to edit user interface for all the Microsoft technologies.

IV.2.iii Desktop application

I chose to develop the project "Fincasting" on desktop for several reasons. The first reason, is because one of the specification which is an advantages compare to the other software provided by our competitors too, is to do not necessarily have an access to the internet for using the application (see section "Existing Software"). Another reason that I choose this solution, is because the other solution saw in the section "Existing Software" requires the creation of an account on their platform. I want that my solution proposed to the users, does not require the creation of an account to use the software. Another reason that I chose to develop this software on desktop is because if I had to develop the application as a web-based application, I needed some additional infrastructures and materials in order to develop and distribute the solution. This solution would have been more expensive, to develop, and maintain in the long-term. Moreover, this solution is more efficient than the web-based solution. However, the choice of the language C# put some restriction about the portability of the software, because this language can only be executed on the Microsoft Windows platform. However, it is not a big problem because according to the latest statistics, Microsoft Windows always represents more than 95% of the computer market (*Desktop Operating System Market Share* 2014).

IV.3. How to get a dataset?

IV.3.i Overview

For the project "Fincasting", we have made the choice to choose the website Yahoo! Finance as a provider for our project. We have choose this provider because it provides a lot of information (e.g. the price at the opening of the stock exchange, the highest price of the stock, the lowest price of the stock, the price at the close of the stock exchange, the average volume and the adjusted closing price), the other reason is because Yahoo! is a reliable service and we have less chance that the website or the service will close in the future.

IV.3.ii How to get a dataset from Yahoo Finance?

As I explained in the section "Calculate the Technical Indicators Manually", to get a reliable dataset from the Yahoo! Finance website (*Yahoo! Finance* 2014), the user has to pass by five steps (see the section II.3.i for more explanation about each step). Therefore, the aim of this project is to automate this process by the software. Throughout this section, I am going to explain you the research work in order to automate this process described above and the solution implemented into the project "Fincasting".

IV.3.ii.A Proposed solution

One of the most known techniques in order to automate the process described in the section II.3.i, is named "Web Scraping". A lot of websites contains a large number of pages auto-generated by using a base template. If we take for example a retail website, we can see that the pages that contain the products follow the same structure, only the data of the product (e.g. price, description, image of the product, category, and so on) change. In this case, the technique of "Web Scraping" could help us in order to get the information about a product and all of these automatically without any intervention of the user (Arasu, Garcia-Molina, and University 2003). To do this action, this technique could use a low level Hypertext Transfer Protocol (HTTP) client in order to get the information contained into a web page. This technique could even integrate an embedded version of a web browser (e.g. Microsoft Internet Explorer, Mozillia Firefox). By integrating an embedded web browser, the software could even send some actions, normally does by the user on its web browser (i.e. execution of the Javascript integrated into a web page). And thus, this technique could simulate the human exploration through a web site.

The technique of "Web Scraping" is in constant evolution, and more and more companies and developers work on this field, in order to improve the process of text processing and human computer interactions essentially (Salerno and Boulware 2006). It exists several methods in order to get information from a web page, but according to the method the performance and the efficiency is not the same. The most known technique are the following:

- **Copy and Paste**: in some cases the best method to get information from a website is the copy-pasting method. Indeed, the human examination is irreplaceable, and according to the website, this technique is the only practical method. For example, when the website has setup a barrier and the other techniques of web-scraping could not be performed.
- **DOM Parsing**: this technique is the more spread and use. The technique can be used for modify or inspect a web page. The software uses a client-side script in order to parse the content of the web page and uses a DOM tree. This technique embeds a web browser (e.g. Microsoft Internet Explorer, Mozilla Firefox) in order to retrieve the information.

- **HTTP Programming**: the use of sockets programming can be use to posting HTTP requests, and thus retrieve information from a static web page.
- **Semantic Annotation Recognising**: a lot of web pages contain some metadatas, semantic markups, and annotations that can be easily retrieved. The annotations could be organised and managed separately from the web page, in order to the scrapers can retrieve the data schema before scraping the web pages. This technique permits to improve the efficiency of the web scraping.
- **Text Grepping and Regular Expression Matching**: by using language like Python or Perl, we can use the UNIX command "grep", in order to extract information from the web pages.
- **Web Scraping Software**: instead of using one of the techniques described above, or by implementing another technique allowing us to get information from a web page, we can use a software that can do the web scraping for us, in order to get the information for us. These softwares can automatically get the information from a web page, and then convert the data into usable information, and store these information into a local database.

The tool that permits to extract the information from a web page is named "wrapper". To generate a "wrapper", the tool used a "wrapper generator" that uses scripts developed by the developer who wants to extract the information. The script contains the information and some part of the template of the web page as declarative rules in order to extract the data from it. The script is passed to the "wrapper generator" that will convert the script into a "wrapper". Moreover, most of these tools are developed by using script languages (e.g. perl, python, ruby, and so on), and even some of these tools are developed as extensions for the web browser Mozilla Firefox (Penman 2009). The reasons for the utilisation of these script languages, is because these languages are more flexible, in the case we want to do some modifications on the "wrapper". But essentially, because these script languages are cross platform, and they can be added to other compiled language (e.g. Java, C# and so on) (Ochal 2010).

Develop a web scraping tool that use one of these techniques explained above could be a tedious, time-consuming, and subject to error. That is why I have made a study to find the best web scraping tool available on the market. It exists a lot of scraping tools and a lot of them are open source (Zheng et al. 2007).

IV.3.ii.B The solution implemented

In the previous section (Proposed solution) I showed and explained the technique of the web scraping, and how it could be useful for the software "Fincasting", in order to get the datasets from the Yahoo! Finance website. However, after an implementation of this solution to get the datasets and a deeper study of how to get the dataset from the Yahoo! Finance, I found another solution to replace the implementation of a web scraping tool. At the last step in order to get the dataset that I explained

in the section II.3.i, the user has to click on the button "Download to Spreadsheet", as we can see in the figure 8. In order to facilitate the implementation, we can go directly to this step in order to avoid the useless processing of the pervious steps. To do that in order to directly download the spreadsheet, the Yahoo! Finance website propose to get the dataset by doing only an HTTP GET request. This solution has the advantage to be easiest to implement, and above all, this solution has the advantages to be more efficient in order to get a dataset, and less prones to the errors. As we can see in the figure 23 the Uniform Resource Locator (URL) is just a simple GET request on the Yahoo! service. This URL is composed of the base "http://real-chart.finance.yahoo.com/table.csv?", and several GET parameters which are the following:

- **s**: this parameter contains the ticker of the company we want the dataset.
- **d**: it contains the month of the ending range, however this parameter is not exactly the number of month like 01 is january, in fact january is 00.
- **e**: this parameter contains the day of the end range.
- **f**: it contains the year of the ending of range.
- **a**: it contains the month of the beginning range, however this parameter is not exactly the number of month like 01 is january, in fact january is 00.
- **b**: this parameter contains the day of the beginning range.
- **c**: it contains the year of the beginning of range.
- **ignore**: this parameter contains the format of the output file, in our case .csv
- **g**: it contains the period we want the information, the possible value are "*d*" for daily, "*w*" for weekly, "*m*" for monthly, and "*v*" to only have the dividends.

By doing this HTTP GET request, we can get a dataset with the parameters we have specified. This solution is the one we have implemented into the software "Fincasting", in order to get the datasets of the several companies.

http://real-chart.finance.yahoo.com/table.csv?s=**TICKER**&d=**MM**&e=DD&f=YYYY&g=**OPTION**&a=**MM**&b=DD&c=YYYY&ignore=**FORMAT**

Figure 23: URL to get a dataset from the Yahoo! Finance website

IV.4. Architecture of the project

In order to respect the specifications that I described before (see sections "Global view of the project" and "Functional constraints"). I divided the software "Fincasting" into several projects. With the features provided by the language C# and the framework .NET (see section IV.2.i), I chose to split the project into one software project

("Fincasting") which is the core of the project, and each model used by the core is in a DLL (see figure 24). This architecture permits the reusability of the model for other C# project, because each model is in a DLL. The developer who wants to reuse the libraries has just need to put a reference on the DLL into his/her project and the libraries can be reused. Into the project we have the following libraries:

- **HttpWebRequestLibrary**: this library allows us to make GET request on the Yahoo! Finance service.
- **CompanyModel**: this library checks the existence of a company on the Yahoo! Finance website, but also it manages the favourite list of company into the software (add/remove a company from the favourite list). Moreover, this library handles the serialisation of the favourite list too.
- **DatasetsModel**: this model is the most interesting, because it is the library that gets the dataset from the Yahoo! Finance website and saves the dataset on the hard drive into csv files. Moreover, this library is used for getting the information which is into the csv file and converts these data into an array that contains all the useful data used for the calculus of the technical indicators.
- **SettingsModel**: this library is used to contain and save all the settings of the software.
- **TechnicalIndicator**: this library contains all the classes that doing the calculus of each technical indicator.
- **WriteInformationInFile**: This library is used to export each technical indicator result into two .txt files.

Moreover, each model uses the facade and factory design patterns in order to change the model easily. For example, if we take the library "DatasetsModel" and we want to change the provider of the dataset (Yahoo! Finance currently), the developer just has to change the code of library by adding a new class into the library (for a new provider) and to change the code of the factory. By using this architecture, the developer does not have to change the code of the core (Fincasting). This architecture permits a better flexibility, and reusability. Moreover, it permits to be more reagent in the case there is a problem into the software, and thus propose an update in order to fix the problem faster. Furthermore, this architecture permits to avoid the problem of functional constraints enumerate in the section III.5.

IV.5. Tools

IV.5.i Revision Control

For the development of this project, we have chosen to develop this project in Open Source. This designation means that the source code of the software is available to the general public. The use of Open Source is generally used in order to create derivative work from the original work, expand the features of the software, fix bugs, and so

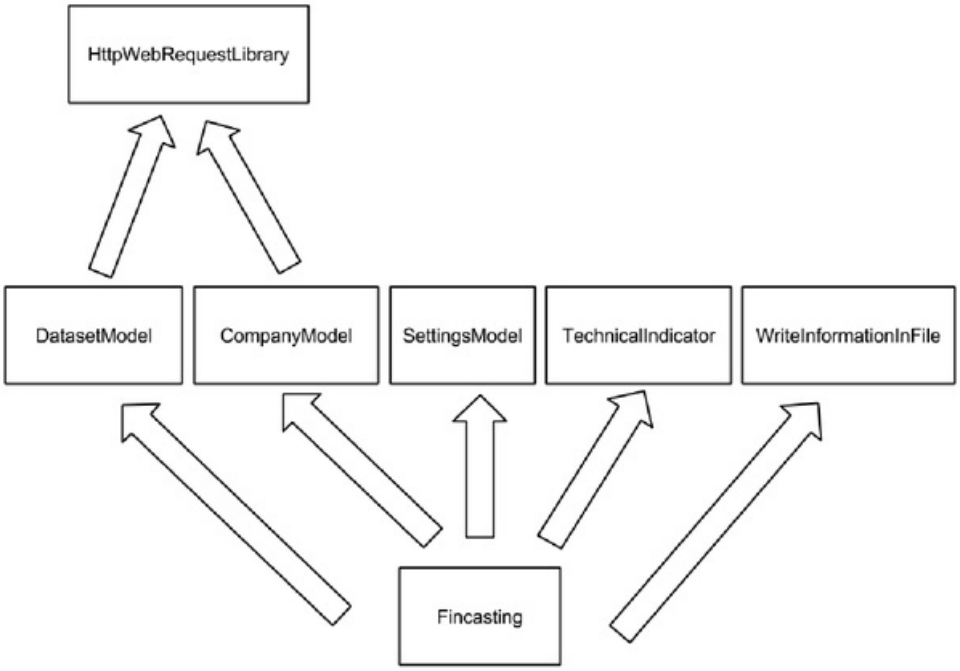

Figure 24: Architecture of the project Fincasting

on. Provide the source code to the general public is usually a collaborative effort where the developers improve the source code together and share the changes within the community and thus other members can contribute in the project.

The sharing of the source code is doing most of the time through a version control repository. The first step was to find the most efficient version control for our workspace. A qualitative and comparative study was doing in aims to find the best version control, and also in order to distribute our source code. In a first time, I selected the most known version controls for my study, which are Concurrent versions system (CVS), Subversion (SVN), GIT and Mercurial (*SVN, Git, Mercurial, and CVS – Comparison of Version Control Software* 2011). And finally, after the study of these different version controls, I have chosen GIT for several reasons. The first main reason that I chose GIT is because at the difference of SVN, GIT does not necessarily need an internet connection, because GIT is distributed and SVN is a client-server model. Moreover, GIT permits to implement easily the system of branching compared with the other version controls. This point is very important, because for this project I would like to implement a good workspace for my project. That is why I implemented the Vicent Driessen workspace (*A successful Git branching model* 2010). This workspace permits to be more reactive for the fixing of bugs and always have a stable version into the repository. However, this workspace is not easy to implement at the beginning of the project and the user needs a

good knowledge into GIT. Another disadvantages with GIT, is the number of different command (Loeliger 2009), and the command with GIT are more complex than with the other version controls (essentially compare to SVN). Finally, I chose GIT, because most of the collaborative version controls use this version control, that is why I chose GIT and *GitHub* 2014 to distribute this project.

IV.5.ii IDE

For the development of most of the project coded in the C# language most of the developers use the Integrated development environment (IDE) Visual Studio. Most of the developers use this IDE because it is the most widespread IDE for the development of any project developed in C#. The main reason, is because this tool is developed by Microsoft which created the C# language. The other reason is because Visual Studio offers to its users several interesting tools for the development (Randolph et al. 2010). The debugger is one of the most efficient on the market, the autocompletion, the refactoring, and many other features work very well compared to the other C# IDEs. Moreover, Microsoft proposes to the developers several versions of its IDE and one of these versions is free (*Visual Studio 2013* 2014).

IV.5.iii Project Management

For the project management of this project, I used the tool "Trello" which is a free web-based project management application. This tool allows to its users to set up the methodology Kaban invented by the car manufacturer Toyota. Every project is represented by a board, and every board contains several lists (a list correspond to a task lists). Finally, each list contains a list of cards (a cards correspond to a task). We could assigned several users on a card and write a comment to the task, which is very useful when the task aims to fix a bug, the person who creates the card can explain the scenario to reproduce the bug. The aim of this technique of management is to move the cards from one list to another, in order to see the progression of the task. This tool helped me throughout the project to define my tasks, bugs, and so on.

IV.5.iv Design

For the design of the application, I used the open source software "ArgoUML". This tool offers several interesting feature for the developers. The main interesting feature is this tool is cross-platform, and thus this tool could be used as well on Windows, Linux or Mac. Another interesting point is this tool is free unlike some other tools. Finally, this tool permits to design several kind of diagrams and respects the standard UML 1.4 (e.g. use case, class diagram, sequence diagram, activity diagram, and so on).

IV.6. Conclusion

We had seen throughout this chapter, the technologies and tools used to developed the software and the reasons of these choices. And finally, we had seen the architecture of the project, and the reasons of this architecture.

V. TESTING

V.1. Overview

Throughout the development of the project, a lot of tests have been set up in order to do not have regressions into the software. The development of tests permit to avoid the errors throughout the development of the system "Fincasting". Moreover, the tests allow to confirm the respect of the specification of the project (see sections III.3 and III.4). In addition of that, comments and C# documentation of code have been wrote throughout the implementation of the system in order to help the maintenance of the code, and to understand the code so as to find the causes of problems and resolve them more easily.

The manual testing was mainly used in order to check the functional tests (e.g. export of the files, add a company into the favourite list, remove a company from the favourite list, take into account the option parameters, and so on). The automated tests have been used to tests the respect of the functionalities and essentially all the formulas (i.e. moving average, trade break out, filter, volatility, momentum, momentum moving average and signals), and the main models (e.g. add a company that does not exist, test if the system has enough information about the company in order to calculate the technical indicators, and so on).

V.2. Manual testing

The manuals testing were done by following a scenario, in order to check that the feature is fulfil by the system, and all of that without any problems. Each scenario follows a normal case of use (e.g. add a company with a ticker name of a company which exist). But also, some extreme cases which could create problems into the system, and thus the software could not be used. Even in the worst case, it could distort the calculation and return a file which contains false values. In this case, the experts will give a file with wrong values to the EDDIE tool and thus distort the prediction.

V.3. Automated testing

Manual testing could not covered all the cases of using. Moreover, the manual testing is long and could be a heavy task to test for the testers. That is why, I have implemented several unit tests. The unit test has the advantages to be automated, very efficient and does not required any user interaction with the system. Each unit test in the project allow us to test each formula (e.g. moving average, trade break out, filter, volatility, momentum, momentum moving average and signals) and thus guarantee the non regression of each feature. Other unit tests had been setup to test the reliability of each model developed for the project (i.e. management of the favourite list, export of files, get a dataset, and so on). To develop the unit tests, the framework .NET provides to the developers several tools to write unit tests, which are very similar as the Java unit tests provides by the framework JUnit (Greene and Stellman 2013, Sells and Griffiths

2007). Moreover, the IDE Visual Studio integrated all the tools in order to automate the launching of the unit tests directly from the IDE (Randolph et al. 2010). However, this kind of tests could not covered all the cases of use, that is why the use of manual testing is mandatory for the sustainability of the project, and thus ensure that the system is reliable.

VI. Conclusions

VI.1. Objectives

This dissertation documented the design and the implementation of the tool "Fincasting". This new software presents the advantage to avoid to a finance expert to get a dataset from the Yahoo! Finance website, and calculate the derivative information from this dataset (i.e. technical indicators and signal). Indeed, the tool "Fincasting" fulfil all of these features present in the section "Functional requirements" and "Non functional requirements", and permits to the users to generate several files that contains these three data for any company available on the stock exchange.

The system permits to respond to the exigencies of the user, and permits to him/her to personalise the data he/she wants for the file which will be used with the EDDIE tool (e.g. the company, long term period, short term period, download period, horizon time n, the percentage of increasing price r, the start date, and so on). Another feature, that was added, is a dynamic process. The dynamic process permits to the user to find the best value of r in percent (the percentage of increasing price within the previous n days), this value obtained is normally between a range which represent the percentage of positive signal according the number of days requested by the user. Moreover, the architecture of the project presented in the chapter "Analysis and design of the system" and "Implementation of the system Fincasting", was implemented and permits to change easily all the models present into the project.

VI.2. Future directions

As we can see throughout this dissertation, this project is intended to experts in finance in order to use easily the EDDIE tool. But several new features could be added to the tool in order to improve it. For example we could add several other technical indicators, in order to improve the prediction of the EDDIE tool. Another feature, that could be added is to integrate directly into the tool, the EDDIE tool, in order to facilitate the usability for the experts in finance.

Furthermore, this tool could be open to a larger public, and do not limit to the experts in finance. Indeed, due to the fact that the "Fincasting" tool get a dataset from the Yahoo! Finance website, we could imagine several new features in order to open the software to a lay public. And implement the features available in many software that doing financial forecasting, like we had seen in the section "Analysis of the Existing Situation" (e.g. more technical indicators, charts, and so on).

VI.3. Conclusion

This project was very interesting and instructive. Indeed, this project was a very good learning experience to improve my knowledge in development and project management. Moreover, I have learned a lot of knowledge about the finance, which is a field

that I never worked before.

I have learned how the planning is important in software development. Indeed, have a deadline and respect it is not an easy task at the beginning. However, with time and practice, and by realising this project I learned to be more efficient and have a better estimation of the length to implement a feature.

Another fields in which I learned a lot of things, is in analysis and design. Indeed, in order to respect the specification and to avoid the constraints of this project, I had to implement a specific architecture in order to have more flexibility into the project. I had to implement that in the case that the Yahoo! Finance website does not provide dataset anymore and we have to implement another provider. This architecture was set up during the implementation of the project, and the design of the application helped me in this task.

Finally, this project helped me to improve my knowledge into the development of desktop application and the use of the C# language. Furthermore, this project introduced to me several knowledge about the finance, and essentially the financial forecasting.

REFERENCES

A successful Git branching model (2010). URL: http://nvie.com/posts/a-successful-git-branching-model/.

Abu-Mostafa, Yaser S. and Amir F. Atiya (1996). "Introduction to Financial Forecasting." In: *Appl. Intell.* 6.3, pp. 205–213. URL: http://dblp.uni-trier.de/db/journals/apin/apin6.html#Abu-MostafaA96.

Arasu, Arvind, Hector Garcia-Molina, and Stanford University (2003). "Extracting structured data from Web pages". In: *ACM SIGMOD international conference on Management of data*, p. 337. DOI: 10.1145/872797.872799. URL: http://portal.acm.org/citation.cfm?doid=872757.872799.

Beck, Kent et al. (2001). *Manifesto for Agile Software Development*. URL: http://www.agilemanifesto.org/.

Breakout Trading: Technical Analysis Primer (2014). URL: http://www.stockpickr.com/breakout-trading-technical-analysis-primer.html.

Brookhouse, James, Fernando E. B. Otero, and Michael Kampouridis (2014). "Working with OpenCL to speed up a genetic programming financial forecasting algorithm: initial results". In: *GECCO (Companion)*, pp. 1117–1124.

Desktop Operating System Market Share (2014). URL: http://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=0.

GitHub (2014). URL: https://github.com/Mamamouchi/Fincasting.

Greene, Jennifer and Andrew Stellman (2013). *Head first C sharp*. Ed. by O'Reilly Media. 3rd Edition. O'Reilly Media.

Kampouridis, Michael and Edward Tsang (2010). "EDDIE for Investment Opportunities Forecasting: Extending the Search Space of the GP". In: *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pp. 1–8. DOI: 10.1109/CEC.2010.5586094.

Loeliger, Jon (2009). *Version Control with Git: Powerful Tools and Techniques for Collaborative Software Development*. 2nd. O'Reilly Media, Inc. ISBN: 0596520123, 9780596520120.

Martinez-Jaramillo, S. (2007). *Artifical Financial Markets: An Agent Based Approach to Reproduce Stylized Facts and to Study the Red Queen Effect*. University of Essex. URL: http://books.google.fr/books?id=jnGbtgAACAAJ.

Momentum And The Relative Strength Index (2014). URL: http://www.investopedia.com/articles/technical/03/070203.asp.

Moving Average - MA (2014). URL: http://www.investopedia.com/terms/m/movingaverage.asp.

MVVM Light Toolkit (2014). URL: http://www.mvvmlight.net/.

Ochal, Zenon (2010). *Dynamic Language Integration in a C# World*. URL: https://www.simple-talk.com/dotnet/.net-framework/dynamic-language-integration-in-a-c-world/.

Otero Fernando, Michael Kampouridis (2014). "A Comparative Study on the Use of Classification Algorithms in Financial Forecasting". In: *EvoFin, EvoStar, Granada, Spain*.

Penman Richard Baron, Timothy Baldwin David Martinez (2009). "Web scraping made simple with sitescraper". In:

Randolph, N. et al. (2010). *Professional Visual Studio 2010*. Wiley. ISBN: 9780470632161. URL: http://books.google.fr/books?id=Kjt39v8AbQkC.

Salerno, J.J. and D.M. Boulware (2006). *Method and apparatus for improved web scraping*. US Patent 7,072,890. URL: http://www.google.com/patents/US7072890.

Sells, Chris and Ian Griffiths (2007). *Programming Wpf, 2Nd Edition*. Second. O'Reilly. ISBN: 9780596510374.

Sestoft, Peter (2010). "Numeric performance in C, C and Java". In:

Smith, Josh (2014). *WPF Apps With The Model-View-ViewModel Design Pattern*. URL: http://msdn.microsoft.com/en-us/magazine/dd419663.aspx.

SVN, Git, Mercurial, and CVS – Comparison of Version Control Software (2011). URL: http://biz30.timedoctor.com/git-mecurial-and-cvs-comparison-of-svn-software/.

Technical Analysis From A To Z (2006). McGraw-Hill Education (India) Pvt Limited. ISBN: 9780070636576. URL: http://books.google.fr/books?id=XJD8IbVj85oC.

Tsang, Edward, Paul Yung, and Jin Li (2004). "EDDIE-Automation, a decision support tool for financial forecasting". In: *Decision Support Systems* 37.4, pp. 559–565. URL: http://www.sciencedirect.com/science/article/B6V8S-4903GV9-1/2/d6ba531a46ce45526ff9015e4447409a.

Tsang, Edward P. K., Jin Li, and James M. Butler (1998). "EDDIE Beats the Bookies." In: *Softw., Pract. Exper.* 28.10, pp. 1033–1043. URL: http://dblp.uni-trier.de/db/journals/spe/spe28.html#TsangLB98.

Tsang, Edward P. K. et al. (2000). "EDDIE In Financial Decision Making". In: *Journal of Management and Economics*, October. URL: http://cswww.essex.ac.uk/CSP/finance/papers/EDDIE2000.htm.

Visual Studio 2013 (2014). URL: http://www.visualstudio.com/.

Volatility-Based Indicators (2014). URL: http://finviz.com/help/technical-analysis/volatility.ashx.

Yahoo! Finance (2014). URL: https://uk.finance.yahoo.com/.

Zheng, Shuyi et al. (Aug. 23, 2007). "Joint optimization of wrapper generation and template detection." In: *KDD*. Ed. by Pavel Berkhin, Rich Caruana, and Xindong Wu. ACM, pp. 894–902. ISBN: 978-1-59593-609-7. URL: http://dblp.uni-trier.de/db/conf/kdd/kdd2007.html#ZhengSWW07.

APPENDIX

A. TECHNICAL INDICATORS

In this section, I present to you all technical indicators used for the project, and for each of them the formulas. Given the closing price of a stock time series ($P(t)$) where $t \geq 0$, and the value L represent the length of the period. The equations (1) to (6) present these formulas.

Moving Average (MA)

$$MA(L,t) = \frac{P(t) - \frac{1}{L}\sum_{i=1}^{L} P(t-i)}{\frac{1}{L}\sum_{i=1}^{L} P(t-i)} \tag{1}$$

Trade Break Out (TBR)

$$TBR(L,t) = \frac{P(t) - max\{P(t-1), ..., P(t-L)\}}{max\{P(t-1), ..., P(t-L)\}} \tag{2}$$

Filter (FLR)

$$FLR(L,t) = \frac{P(t) - min\{P(t-1), ..., P(t-L)\}}{min\{P(t-1), ..., P(t-L)\}} \tag{3}$$

Volatility (Vol)

$$Vol(L,t) = \frac{\sigma(P(t), ..., P(t-L+1))}{\frac{1}{L}\sum_{i=1}^{L} P(t-i)} \tag{4}$$

Momentum (Mom)

$$Mom(L,t) = P(t) - P(t-L) \tag{5}$$

Momentum Moving Average (MomMA)

$$MomMA(L,t) = \frac{1}{L}\sum_{i=1}^{L} Mom(L,t-i) \tag{6}$$

Tool for Creating Financial Forecasting Datasets

GRADEMARK REPORT

FINAL GRADE

/100

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48