THE UNIVERSITY OF NEW SOUTH WALES
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

A User-Driven Environment for Financial Data Analysis Workflows: Design and Implementation of ADAGE Trading Simulator

Author: Pei Wang,
CSE ID: pwan823
UNSW ID: z3290805

Abbreviation for degree: 3653

Submission Date: October 2013

Supervisor: Fethi Rabhi

Assessor: Helen Paik

Contents

1. Introduction

Ever since the introduction of the stock exchange, people have been trying to invest in such market to increase the yield of their fund. However, trading is like any other skills, which need constant practise to make perfect. Practising in the market means paying our hard earned money as tuition fee. Not to mention the skill, time, and effort barrier lies between the novice traders and the ones who could earn a profit. That is, for a regular person who has no financial knowledge, it takes significant amount of time and monetary cost to learn such skill. Is there an effective way to practise without carrying such cost?

Over the year, people have been developing solutions to this problem. Simulation software has been made to facilitate the training process with minimal cost. These software normally use historical or real time (but delayed) data so the user get a real sense of reality to win the trade. This allows user to practise trading whenever it is convenient. By doing so in a simulated environment, traders could try their trading strategy and become better before putting money to work.

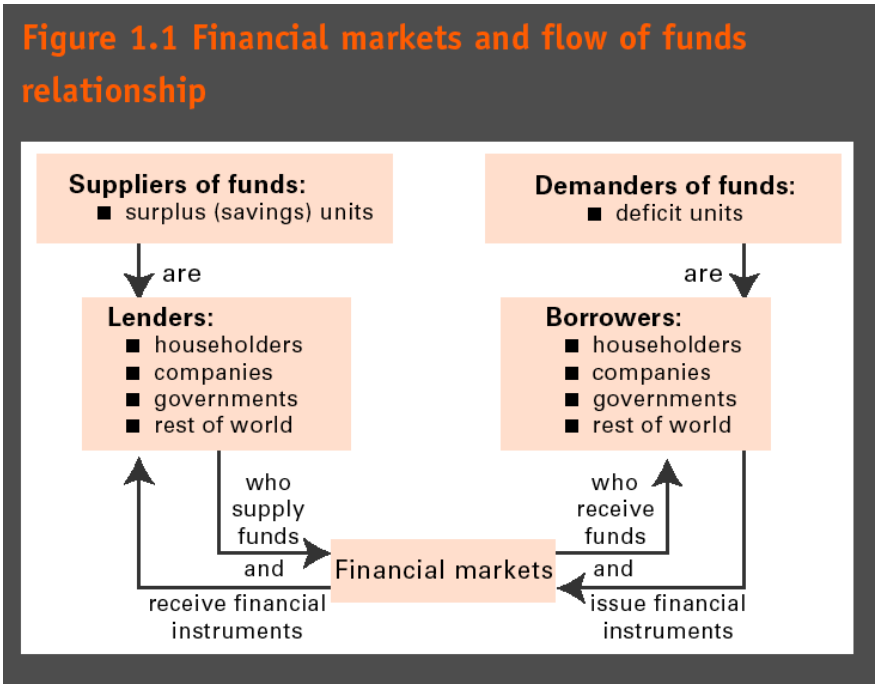
The purpose of this project is to examine existing software systems and study their strengths and weaknesses. The project will propose and implement a new system that captures the essence of existing systems and fill in their functional gap.

2. Background

2.1 Financial System

The financial system is part of a country's economic system. A financial system comprises a range of financial institutions, financial instruments and financial markets which interact to facilitate the flow of funds.

Its main function is to facilitate the transfer of funds from surplus to deficit economic units, in primary financial market, by the creation of new financial assets. It also provides arrangement for trading in existing financial assets in secondary financial market. An efficient financial system should ensure that savings will be directed to the most efficient users of those funds.



The largest financial institution in the Australian financial system is Commercial Banks. In Australia, they are in possession of 52.0% of the total financial assets (2008), seconded by superannuation funds 17.2%.

2.2 Financial Assets

A financial asset is defined as entitlement to future cash flows and financial security is a financial asset that can be traded in secondary market.

Equity

Equity can be described as an ownership interest in an asset. Types of equity include ordinary share and hybrid security such as preference share, or convertible notes. Focus of this project is on ordinary share.

The stock market today is one of the most important sources for companies to increase their leverage. Businesses with outstanding performance which agreed to comply with tighter regulation would go public to gain access to more financial capital, in a way such as issuing shares to public in IPO (initial public offering) in the primary market.

After the IPO, shares can be traded in the secondary market. An exchange market is a highly organized market to trade large variety of securities/financial instruments between buyers and sellers, thus providing a marketplace virtual or real. It facilitates the flow of fund from the surplus units to deficit units. Notice that

there is no additional capital flows to the firm which issues the stock after the IPO, but the fluctuation in price reflects public's future perspective of the firm. The focus of this project is on the secondary market.

Shares are the most well-known and commonly traded securities on the market. They give returns to the stock holders in form of dividends, which is issued by the firm. The reason why stocks normally give higher return than bonds or deposits is because they carry more risk, in term of the company might have not been profitable to issue enough dividend, sometimes the firm does not issue dividends at all. More return is expected from taking such risk, hence generating a higher return than bonds.

Debt

A debt is the contractual claim to periodic interest payments and repayment of principal. Some examples are loans, commercial bills and bonds. Government bond is considered as a risk free investment, and hence used frequently in financial models to evaluate portfolio performance. Its application is beyond the scope of the project.

Derivatives

Derivative instruments are different from equity and debt in terms that they do not provide actual funds for a borrower, but rather facilitate the management of certain related risks. It includes futures, forwards, options, swaps and commodities. Options and futures are similar in the way that both refer to an agreement on the trading of the underlying asset at a certain price sometime in the future. However, an option gives the right, but not the obligation to the holder of the options to execute, whereas a future gives obligation to both writer and the holder of the future contract to trade the underlying asset.

The scope of the project lies within shares and options. Other securities are not in our concern.

Option in greater detail

An option is a derivative taken in form of a contract which gives the option holder the right, but not the obligation, to buy or sell an underlying asset or instrument at a specified strike price on or before a specified date. There are two main types of options: call options and put options. The holder of the call option has the right to buy the underlying asset for a specified price on a specified date. The holder of put option has the right to sell the underlying asset for a specified price on a specified date.

The specified price and date in the option contracts are called strike price (or exercise price) and the expiration (or maturity) date respectively. Based on the execution time, the options are classified within two categories: American options and European options. The former can be exercised at any time up to the expiration date, while the latter can only be exercised on the specified expiration date. Most of the options traded on exchanges are American options, but European options are easier to analyse than American options. In this report, by options we mean European options unless specified.

There are two main differences between options and forward/future contracts. The holder of options does not have to exercise but in forward/futures the holder is obliged to sell/buy the underlying asset. Moreover, there is a cost (premium) to buy/sell an option contract while there is no premium to enter into a futures/forward contract.

Options do not have a secondary market. If an option holder wants to close the position before expiration, a new option needs to be written.

2.3 Trading Strategy Fundamentals

2.3.1 Incentive behind trading

The most fundamental incentive is to "increase leverage". Leverage is the use of various financial instruments or borrowed capitals to increase the potential return of an investment. In this case, since an option gives the right to perform a transaction on the underlying asset, without manipulating the ownership of the firm. This gives the same exposure to the movement of the price as the underlying asset, namely stock, but at a lower price. Leverage helps both the investor and the firm to invest and operate. However, it comes with greater risk, because leverage magnifies both gains and losses. In the business world, a company can use leverage to generate shareholder wealth, but if it fails to do so, the interest expense and credit risk of default may ruin shareholder's investment.

In addition, due to the special nature of options, trading strategies has been developed and adapted in financial portfolio managements. In general, trading strategies are adopted to make profit on the speculation of future price of stock or market. Option strategies are the simultaneous, and sometimes mixed, buying or selling of one or more options that differ in one or more of the options' variables. This is often done to gain exposure to a specific type of opportunity or risk while eliminating other risks as part of a trading strategy. For instance, if a speculation predicts the price of a stock is to decline, getting into a long position of a put option would give the right to sell the underlying stock in our portfolio at the strike price at expire date. This gives us the opportunity to sell the stock at strike price, which could be above its future market price, hence making a profit at the cost of buying the option.

The reason why investors need to understand option trading strategy is because its versatility to build all sorts of trading strategies to apply on different circumstances. And we make money by doing so correctly. However it is also complicated for people without a finance background to comprehend. Trading is a skill which takes practise to make perfect. This project aims to educate and improve users' trading skill, so they would be better prepared in the real financial markets. Typically speaking, the underlying financial assets of trading strategies include stocks options and bonds. Each asset carries a distinctive nature. Some primitive or complex trading strategies consist of different combinations of the said assets. The example we used in the previous paragraph is an example of "protective put". It forms a portfolio consists of a put option and a stock. In addition, the skill being covered is transient, because they are also applicable to other derivatives such as forwards or future. However, our focus stays with option.

2.3.2 Nature of option

We have had a brief preview of a simple option trading strategy in the previous session. In this section we will explore the objectives and examples of some more option trading strategies.

Option strategies are the simultaneous, and sometimes mixed, buying or selling of one or more options that differ in one or more of the options' variables. The objective of adapting an options strategy is to make profit or minimize loss from movements in the underlying that are bullish, bearish or neutral.

The notations used in this material have been summarised in table 1.

Variable	Description	Value (Data available or Calculation needed)
X or K	Strike price of the option	Data in the option contract
S_T	Price of the underlying asset of the option at the maturity time of the option	Unknown
S_0	Current price of the underlying asset of the option	Available Data
P or C	Value (price) of the option	Data in the option contract
T	Expiration date of the option	Data in the option contract (in years)

Table 1: Notations and assumptions

Each option contract has two sides: option writer (seller) and option holder (buyer). The investor who has bought the option takes the long position and the one who has sold (or written) the option takes the short position. The purchaser of the option pays the premium (in front) to the writer of the option. Considering these two positions and two types of option contract, there are four types of positions: a long position in a call option (long call), a long position in a put option (long put), a short position in a call option (short call), and a short position in a put option (short put).

The nature of options in different positions will be explained in the following with assistance of payoff diagrams. On the x axis we have the future market price of the underlying asset S_T. Y axis the payoff. A payoff on the top half indicates profit. Bottom indicates loss. 0 indicates no gains and no loss on the y axis and has no indication on the x axis. The orange line is the payoff of the option and blue line is the profit, that is calculated as the payoff – (+) price of the option in a long (short) position.

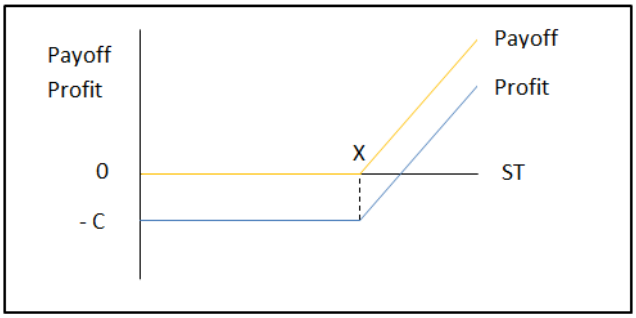
The flat part of the payoff line below the black line suggests the market price is more competitive than the strike price on the option, so the holder would let the option go lapse (expire worthless). The holder will only execute the right if the strike price is more favourable than the market price, that resulting in a positive or negative payoff.

2.3.3 Call option

Definition: A derivative which gives the holder the right to purchase a security at the exercise price.

·Pay off for the long position in a call option: max (S_T- S_k , 0)

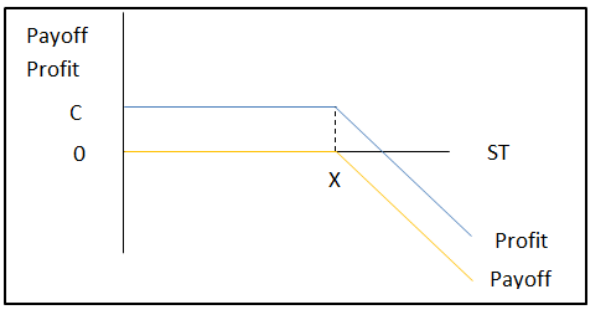
Payoff to call holder = $S_T – X$ if $S_T > X$
0 if $S_T <= X$
Profit = Payoff – C



·Pay off for the short position in a call option: min (S_k - S_T, 0)

Payoff to call writer = - ($S_T – X$) if $S_T > X$
0 if $S_T <= X$

Profit = Payoff + C



Strategy:

When to get into a long call position: stock price is expected to grow, so the call option holder could execute the right to purchase at a relative lower strike price comparing to ST.

When to get into a short call position: stock price is expected to fall, so the call option holder let the right to purchase at strike price to expire worthless.

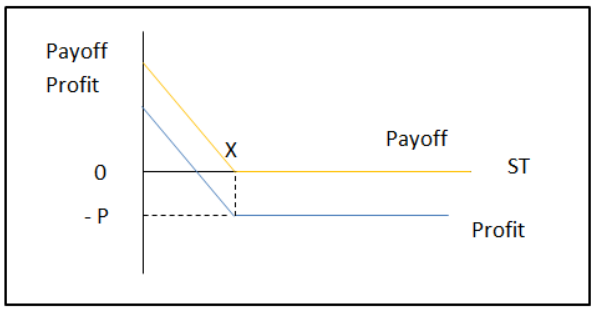
2.3.4 Put option

Definition: A derivative which gives the holder the right to sell a security at the exercise price.

Payoff for the long position in a put option: max (S_k - S_T, 0)

Payoff to put holder = 0 if S_T >= X
$X - S_T$ if S_T < X

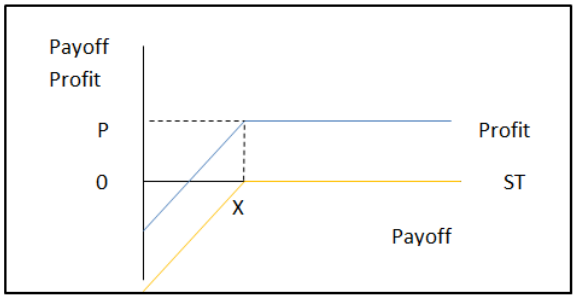
Profit = Payoff – P



Payoff for the short position in a put option: min (S_T - S_k, 0)

Payoff to call writer = 0 if S_T >= X
$-(X - S_T)$ if S_T < X

Profit = Payoff + P



Strategy:

When to get into a long put position: stock price is expected to fall, so the put option holder could execute the right to sell at a relative higher strike price comparing to ST.

When to get into a short put position: stock price is expected to grow, so the put option holder would let the right to sell at strike price to expire worthless.

2.4 Primitive Option trading strategies

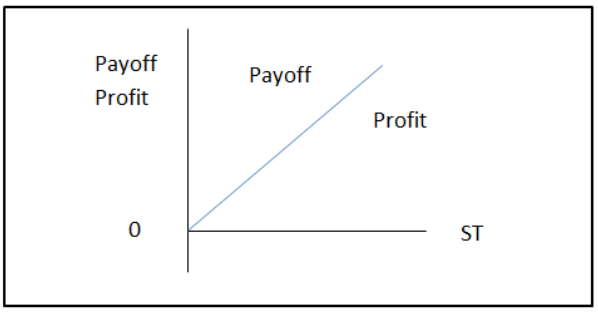
2.4.1 Protective Put

Definition: purchasing of a stock and a put option of the stock

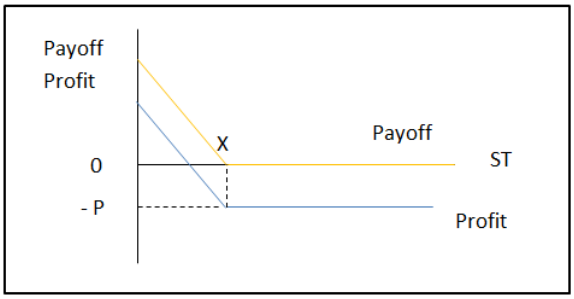
Payoff	$S_T <= X$	$S > X$
Stock	ST	S_T
+ Put	$X – S_T$	0
= TOTAL	X	S_T

Profit = Payoff – P

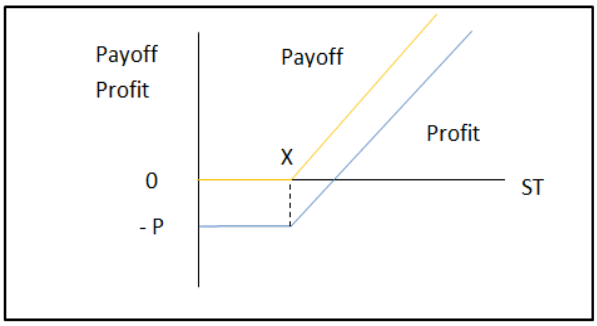
Stock



Long Put



Total



Strategy: when you want to invest in a stock but unwilling to bear potential losses beyond some given level.

Example: Suppose you have 100 BHP shares on hand selling at $60 per share and you signed a put contract (for 100 shares) for $200 of the BHP stock with strike price at $50, expire in 6 months. The put contract

ensures the value of your portfolio will not fall below $50 * 100 = $5000 by the end of the expiration. The maximum loss will be the premium you paid for the put contract.

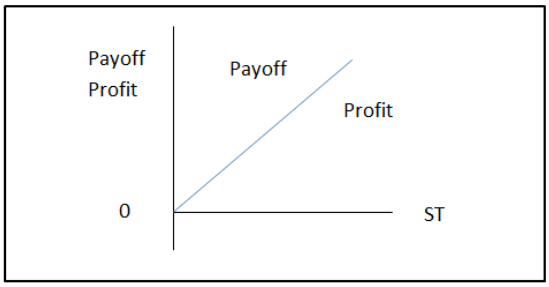
2.4.2 Covered Call

Definition: Purchase of a share of stock with a simultaneous sale of a call on that stock. This "covers" the potential obligation to deliver the stock. Naked option writing refers to writing a call option without an offsetting stock.

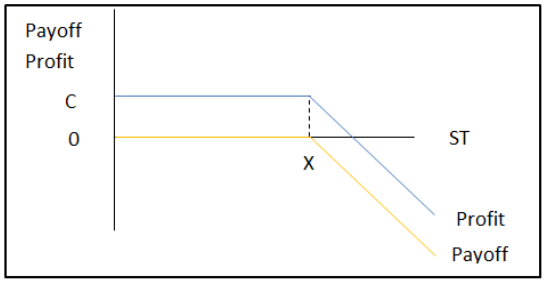
Payoff	$S_T <= X$	$S_T > X$
Stock	ST	S_T
+ Written call	-0	$-(S_T - X)$
= TOTAL	S_T	X

Profit = Payoff + C

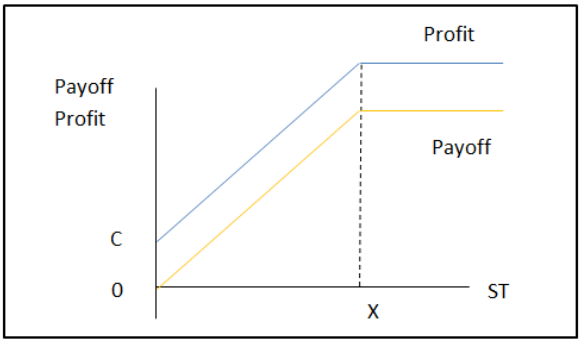
Stock



Short Call



Total



Strategy: Popular among institutional investor because it boosts income by collecting the premium.

Example: Suppose you have some BHP stock on hand and believe the price will either fall or not increase above X. Then you could write call option to maximize your profit. If the stock does go above X, and the holder of the call option you wrote decides to exercise the right, your BHP stock will have it covered.

Complex trading strategies can be found at the appendix of this report.

2.5 Trading Simulators

2.5.1 Why do we need simulator

Financial management is a skill which needs time and effort to master. Instead of paying real money as tuition fee, people seek ways to practise without bearing such cost. Trading simulators have been developed to answer to this demand. It allows user to test their trading strategy before putting real money into the market.

2.5.2 Review of existing simulators

In this section we assess the nature of existing simulation systems. Three systems are studied and their strength and weakness are being assessed.

TradingSim

TradingSim is a web based trading simulator on Nasdaq. It provides the ability to simulate day trading 24 hours a day from anywhere in the world. Every stock on the Nasdaq is being tracked by their system.

The playback control in TradingSim allows a user to pause, play and fast forward through the market action. The fast forward mode allows traders to see if positions would have gone in their favour without sitting through every tick.

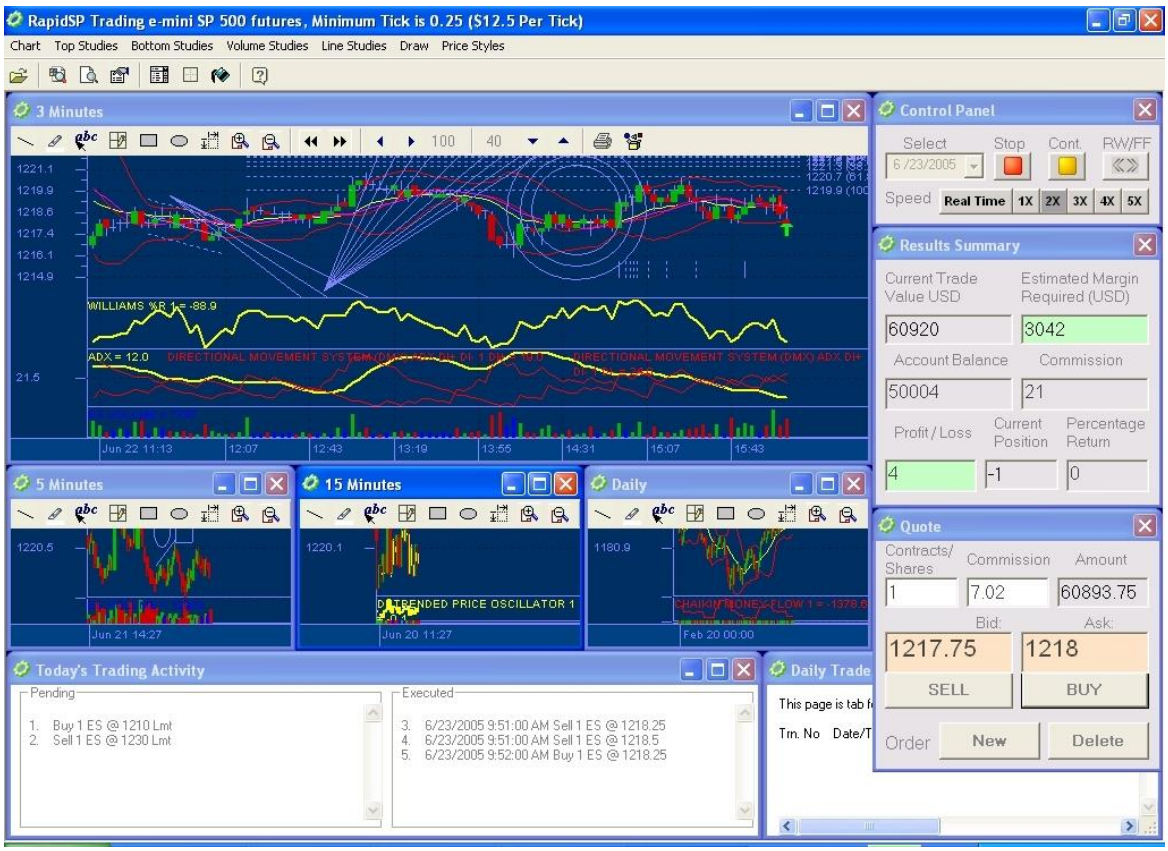
It allows trader to place different types of order in respect to the real exchange market. All past transaction are recorded.

TradingSim allows users to simulate trades directly over the web browser, without the hassles of downloading any software.

RapidSP Day Trading Simulators

RapidSP is a powerful yet easy to learn day trading simulator that can be used for realistic tick-by-tick paper trading of stocks, futures and currencies. Test your trading abilities with virtual money and see how much you can make by predicting the intra-day price movements correctly. With day trading skills there is no limit on how much you can make even in real life. Users can download more data for many stocks, currencies and futures for free.

Trade currencies, futures, commodities and stocks - Many years of data on many different instruments are available to download. It offers 100+ different technical studies and oscillators. Users can add technical indicators via the menu and indicator property dialog. The technical analysis is updated at every tick. All the technical studies in RapidSP are highly regarded for their mathematical accuracy.



TradeStation

TradeStation's award-winning trading platform, trading software and online financial brokerage services set the standard for fundamental and technical analysis of financial markets, and for direct access online trading across multiple asset classes. Through our proprietary and customizable trading tools, TradeStation helps trader to create custom trading strategies, back-test them against our extensive historical database and automate the real-time monitoring and execution of your strategies. It made online stock trading, online options trading and online futures trading available. TradeStation is also a leader in currency trading, offering some of the lowest forex spreads – together with some of the most sophisticated automated forex trading software – available to online forex traders.

Currently the best stock trading system, best options trading system and best futures trading system available for intermediate and advanced traders.

	TradingSim	RapidSP	TradeStation
Data type supported	Real time & Historical Price Data	Historical Price Data	Real time & Historical Price Data
Play Back Control	Y	Y	N
Different order types	Y	Y	Y
Portfolio management	Y	Y	Y
Charting	Y	Y	Y
Tradable securities	Stock	Stock and derivatives	Stock and derivatives
Additional functions	Web based	Technical indicator	Automatic scanning, Customizable view, Multiplatform, Trading Strategy, Analysis report
Price	$199 life time	$99 life time	$60/month

Table 2

2.5.3 Discussion

Table 2 presents a complete summary according to the following criteria.

- For the purpose of simulation, data should be as realistic as possible. The first row "Data type" indicates whether the simulator runs on real time data (the data being collected from the market with some delay) or historical data (market data generated some time ago).
- "Play Back Control" refers to being able to rewind the simulation back to previous states. This function is not supported by TradeStation. One argument could be TradeStation uses both historical and real time data. That means the user could start another simulation from a certain time in the past, hence compromising the lack of a control panel. In addition, it is not considered as good practise to make the option available so that the user might abuse this functionality, defeating the purpose of self-improvement.
- "Different order types" refers to Market, Limit, Stop, Stop Limit order type. They are essential functionality to capture the timing of stock trading.
- "Portfolio management" allows user to review financial asset on hand. Charting gives visual presentation of the market and stock movement. These two criteria are critical to the usability of the simulator.
- "Tradable securities" is self-explanatory. Simulator must have financial instruments to run.
- "Additional functionalities" covers the functionality that improves the user experience but no essential to create the simulation environment.

No doubt TradeStation is the best trading simulator on the market. It offers extraordinary amount of functionalities. Even for the features that have been achieved by the other products, TradeStation simply has them done better. However, one drawback of the system is that its target customers are above intermediate level. A novice trader will definitely be overwhelmed by the level of complexity. Therefore, it is not the ideal tool for beginners. The other two trading simulators offer the necessary function to create a virtual environment and allow user interactions.

Objectively speaking, the level of complexity remains high in RapidSp and TradeStation on the level of trader who has no experience. TradingSim has a simple and refreshing interface, but it has also sacrificed the functionality to trade derivatives.

2.6 Conclusion

Financial management is a useful skill to have but it takes time and effort to learn the fundamentals. Financial market is deep and complex. Each financial security has its own nature. It takes even longer to master the strategy to maximize one's capital. This section only covers a fraction of the whole. It takes time to educate an investor with required knowledge, but too many people have entered the financial market blind folded and got reaped clean. Therefore, there is definitely a demand for a righteous tool to do so. Many software systems have been developed and each has its own advantages and short comings.

The following section is my proposal to answer this demand.

3. Proposed work

3.1 Problems

As mentioned in the previous section, people need to be educated about the nature of the exchange markets in order to make a gain without suffering too much lost during the learning process. Existing simulators on the market are either too complex or does not have enough functionalities. Therefore, this project will provide a simulator for this demand.

A simulator is either run on historical data or real time data. Each implementation has its own advantages, but all depends on the movements of the exchange markets. And what affects the movements of the markets? In finance, the efficient-market hypothesis (EMH) asserts that financial markets are "informational efficient". The semi-strong-form EMH claims both that prices reflect all publicly available information and that prices instantly change to reflect new public information. In other words, information such as news, financial reports and announcement of the executive has critical effect on the market movements.

Simulator runs on real time data does not have to worry about the supply of the latest information since technology has made them available online Web site such as Reuters or from news reader like RSS provide plenty news. However, all simulators running on historical data fails to collect what was happening during the simulated time frame. Imagine the confusion created by the moving chart to the trader with no financial knowledge, and the users are not even aware of what is missing, because they are overwhelmed by the amount of functionality and technical terms.

This creates a functionality gap. No simulator embraces historical events, such as significant news which may have impacted the market. From the trader's point of view, the fluctuation of the stock is mystical and beyond comprehension. In fact, market is driven by events such as new government policy, release of financial reports, dividend announcements, shifts of strategic plan, changes in executive etc. This might not be an issue if the simulator uses real time data, since such information is easily obtainable from the internet, but it is absolutely crucial if historical data is being used. Therefore, the proposed project will fill in this gap.

3.2 Objectives

The main objective of this project is to design and implement a stock exchange simulation module called the "ADAGE Trading Simulator". The system will be developed to cover the following goals:

- Allow user to get familiar with financial market
- Allow user to apply different trading strategies
- Allow user to replay historical data
- Functionality to maintain user profile

Individual investors are mostly familiar with the concept of stock. However, stock is not the only derivative existing on the market. Option is a derivative to be used for hedging, and speculating purposes and it massively increase the leverage of the investment. It gives the holder the right but not the obligation to buy or sell at exercise price on a specific date. In order to take advantage of its properties, our first goal will encourage users to undergo a series of comprehensive tutorial to strengthen their knowledge of the said derivative and the exchange market.

With sufficient knowledge, users could start trading on the simulator embedded in the system. The goals are to exam and improve the skill of our users in the simulated environment. The simulator has a graphic user interface. It displays all the essential information of the market and details of every listed firm, which will guide users to make judgement on the profitability and future perspective. Different trading strategies could be applied. A portfolio is also made available for users to manage.

Another feature of the system is that it uses real historical data to run the simulation. The data includes a timeline of all the parameters of the market and every listed firm, and all the news headlines which may have affected the market. Unlike other existing trading simulators, some run on delayed real time data, some simulate the movement of an artificial stock. The ADAGE trading simulator could traverse through a specific time frame, creating a realistic virtual trading environment in respect to what actually happened in history.

Last but not least, users could check their performance of a specific session and the detail of every trade, along with the news and stock status at the time. This allows users to relive the decision making moment and assess their performance from the outcome, so they could learn from their mistake and improve their trading skill. System also records user's most significant achievement as a form of encouragement.

3.3 Methodology

Agile Programming

For this project we are to develop an "ADAGE trading simulator" where small functionalities are built up over time. Due to this, the methodology for this project we found most suitable for this situation would be the "Agile Programming" methodology. "Agile Programming" breaks tasks into small increments and makes the project flexible for future changes. During the development of our program, each component is guaranteed to work properly and is corrected to the specification given before moving on to the next. Furthermore, by applying this methodology, it is reassured that our product is flexible and can be changed in the future if required without modifying the full behaviour of the entire product.

Test Driven Development

Each time a feature is completed it is pass to our QA (Quality Assurance) test and it will go through a series of tests. These tests will be based on the specifications given which the feature must satisfy. As correctness is a vital aspect of this project, a dedicated testing process is mandatory.

The QA process requires testing each feature independently, so the correctness is guaranteed. Hence, the sum of the independent parts is integrated with consummate quality.

Technology

Version control

In order to maintain efficiency and documents, Git GUI is used for version keeping. This tool provides the functionality to track all modifications made to the project. It also provides the functionality to revert back to a previous version of software for backup purposes. Best of all, it could operate offline, allowing maximum flexibility for development.

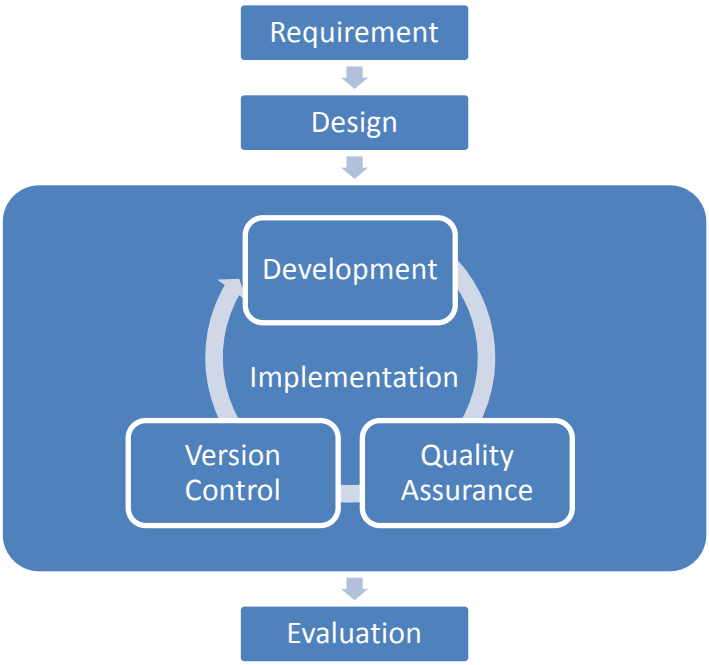


Figure 4-2 Project Development Methodology

The following sections will be Design, Implementation and Evaluation.

- Design section covers the requirements, architecture, use cases design and development methodology. This is covered in section 4.
- Implementation section presents the development environment, code structure, use case updates and any second and third party package embedded in the system. This is covered in section 5.
- Evaluation section will include validation of the system and conclusion on the project. This is covered in section 6.

4. Design

This section is a project overview containing information on the project specifications. The report will start with the requirements, then project objectives, architecture, design and use cases, and class diagram.

4.1 Requirements

Goal Level

ReqID	Requirement	Short Description
GL-1	Allow user to get familiar with trading stock market and option strategy	System provides tutorial and links to other website.
GL-2	A simulator to allow user to interact with	To play the trading session in a simulated environment.
GL-3	Allow user to play on real historical data	Data used in the simulation was from historical data.
GL-4	Functionality to maintain user profile	The profile record user's detail and past performance.

Table 4-1: Table of Goal-Level Requirements

Domain-Level

ReqID	Requirement	Short Description
DN-1.1	Multiple choice to exam the knowledge of the user	Feedback on user's performance will be given at the end of the test.
DN-1.2	Provide links to external websites	External link includes links to the system tutorial and ASX
DN-1.3	User select different tutorial session through GUI	There are sessions for Market, Stock, Option.
DN-2.1	Implement a GUI for user to interact with	The GUI needs to be concise, informative and responsive.
DN-2.2	User could select a simulation session from a collection of configurations	Configuration comes from the downloaded packages
DN-2.3	Functionality to check portfolio of financial security on hand	User needs to know the value of the security being purchased before.
DN-2.4	Allow trading of stock and option	A separate view is displayed upon trading request
DN-2.5	Functionality to display detail of the stock market and every listed firm	The simulation pond will display/link to everything user need to know about the market
DN-3.1	Functionality to read stock data, option data and news data from the package	The package contains three kinds of data for simulation.
DN-3.2	Functionality to download new content	New content is downloadable to extend the usability of the module.
DN-4.1	Allow CRUD of accounts	Allow user to create, read, update and delete accounts
DN-4.2	User could check performance of a specific session and the detail of every trade	Price, volume, profit, news at the session, parameter of the company etc. will be revisited
DN-4.3	System records user's most significant achievement	As in the most profit made in one trade and session.

Table 4-2: Table of Domain-Level Requirements

4.2 Architecture

4.2.1 Language and Platform

The language used was Java.

Java is the ideal choice of language, because it has many native advantages. It provides a massive built in library. It runs on virtual machine so developers do not have to worry about the lower level specifications. Java Applet and Servlet also give the freedom to run on different platforms. It also has automatic garbage collector so developers are free from the task of trivial memory management. In addition, it has a potent library for GUI design.

Several plans were perceived before the implementation of the simulator. It could be implemented as a web application like the web based simulator TradingSim introduced in the research section of the report. However, this idea was dropped because it requires the user to always maintain an internet connection, hence limiting the usability of the simulator when internet is not available or not stable. For example, recent online campaign against DRM (Digital Right Management) due to synchronizations, loss of data, absent of services during stress hours shows the potential pit falls of a web application. This leads to my decision to use the Java swing library for GUI and Java Applet for the model.

Therefore, considering the nature of this project, Java is the ideal language.

4.2.2 MVC model

MVC is a methodology that is widely accepted and applied in software system developments.
It has been proven to be one of the best practices in software developments. The project will be conducted under the principles of MVC architecture.

Following the guidelines of MVC makes the development process proceeds more effectively, because having the project divided into three sections of model, view and controller means that three schedules could run in parallel. Another advantage of applying MVC is that it provides a certain degree of abstraction, making the system more secured. Therefore, MVC is the ideal architecture for the project.

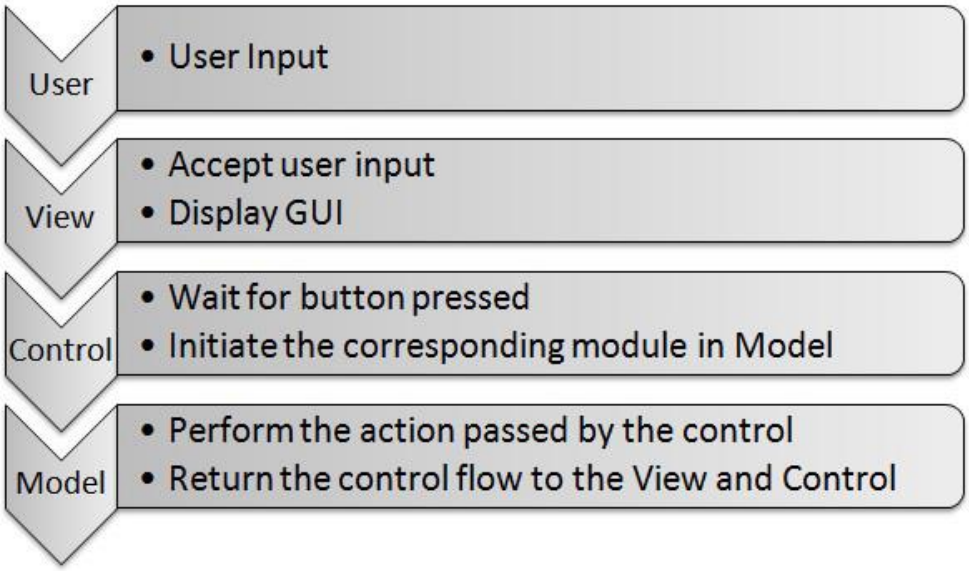
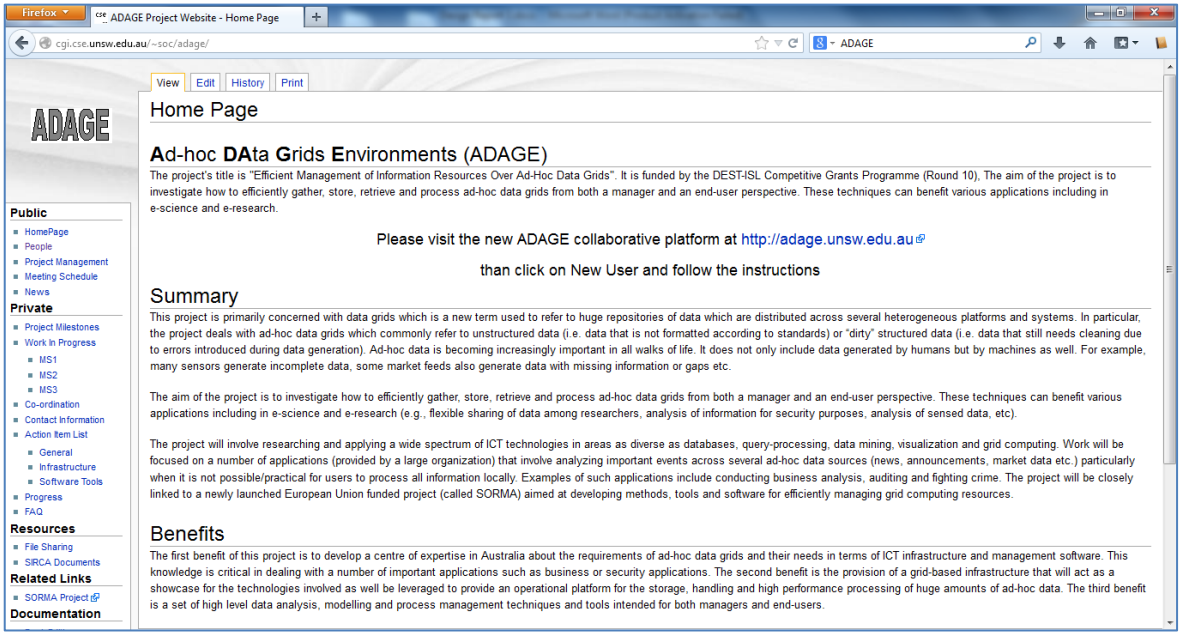


Figure 4-1: This figure shows the flow of the program

4.2.3 Architecture Design

Ad-hoc Data Grid Environment

This project is built as a part of an integrated software system in Ad-hoc Data Grid Environment (ADAGE). ADAGE is open architectural framework based on service-oriented computing (SOC) principles. It is a framework that guides the development of domain-specific problem-solving environments or systems to support data analysis activities. It addresses the issue of interoperability, providing a superior user experience by working synergistically as a coherent group, rather than an unrelated collection of incompatible tools. The ADAGE framework has been applied on the development of a prototype which focuses on analysis of financial data. In the prototype, different functionalities are created as modules, which massively increase the flexibility of the system by allowing the user to create new system or customize and extend existing system.



ADAGE

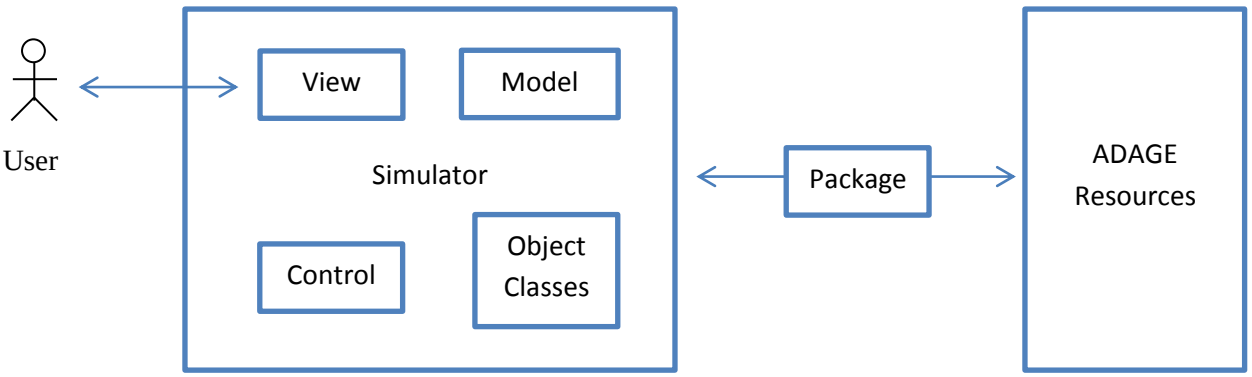


Figure 4-2: How ADAGE framework is perceived to interact with the simulator

4.3 User Interface Design

In this section we will focus on the core of the system, which is the trading simulator. The rest of the mock up design could be found on the appendix. Now assume a user has logged in and decides to do a simulation as shown in Diagram 4-1. The left panel displays the information of the period related with the chosen configuration, which is the financial crisis package. The simulation starts after user clicks the "Start" button. The software loads the information from the package.

The view shifts to the simulation pond as shown in Diagram4-2. This is the main simulation view, with the left panel displaying the movements of the listed stocks and market index. Clicking on the "Index Chart" button shows the chart of this market in a separate window as shown in Diagram 4-3. The top right panel displays the latest news that just happened according to the timeline of the configuration. Under the text "Detail" is a textbox showing the detail of a selected stock. The user now selects a stock "GOOG.O", then click "check detail", which populates this textbox.

There are four buttons under "Trade". User clicks on the "Check Chart" button to display the chart diagram as Diagram4-3, but this time the chart is plotted on the stock, not on the market index. Suppose the user finds the investment in this stock appealing, and an increase in stock value is likely to happen, he would like to purchase the stock and a put option. This could be down by clicking on the "Trade" button, which brings up a window, where the user could select type of security using the drop down menu. After click on the "Select" button, the view would be populated with text fields as Diagram 4-4 and Diagram 4-5. After keying in all the essential information, clicking the "Calculation" button will display the total amount of transaction in the "Total" text field of Stock view and "Price" and "Total" in the Option view. Price of the option is determined by the system. Two trades happen accordingly as illustrated in the diagram. Assuming the price of the stock at the moment is $104.00 per share.

Another button under "Trade" is "Strategy". Clicking it brings up Diagram4-6. This view assists user to choose an ideal trading strategy with diagrams illustrating the payoff of each security and the final outcome. The explain button on the top links to the tutorial page where detail explanation is articulated.

Now we are back to Diagram 4-2. With two derivatives on hand, we would like to check our portfolio after several simulation months. Click on the "My portfolio" button bring up Diagram 4-7. The "Onhand" column shows whether the order has been fulfilled. The value of unfulfilled order is not calculated into the total profit. Type column could have three values: "S", "C", "P" indicating stocks, call and put options. "X" and "T" show the expiring price and time of an option, so it is not applicable to stocks. Assume the stock price has increased to $107.00. The value of the call option would be $2.00, since its expiring price is $105. The stock order on the other hand, has not been fulfilled, because the price of the stock has never dropped to $100.00 per share.

Going back to Diagram 4-2, we have three buttons on the right bottom corner, namely "pause", "auto run" and "skip to next day". They control the advancement of time in the simulation. If the user want to terminate the current session and resume it later, he could click the "Save" Button. It would save the current session and allow the user to resume next time on Diagram 4-1by clicking the "Load" button.

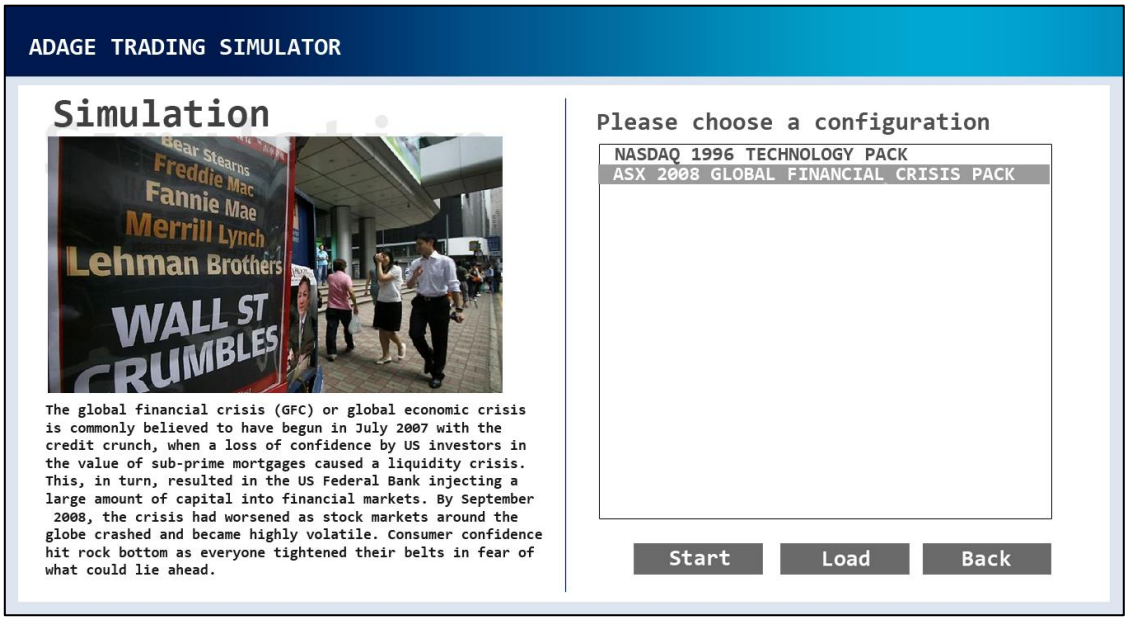


Diagram4-1: Simulation configuration

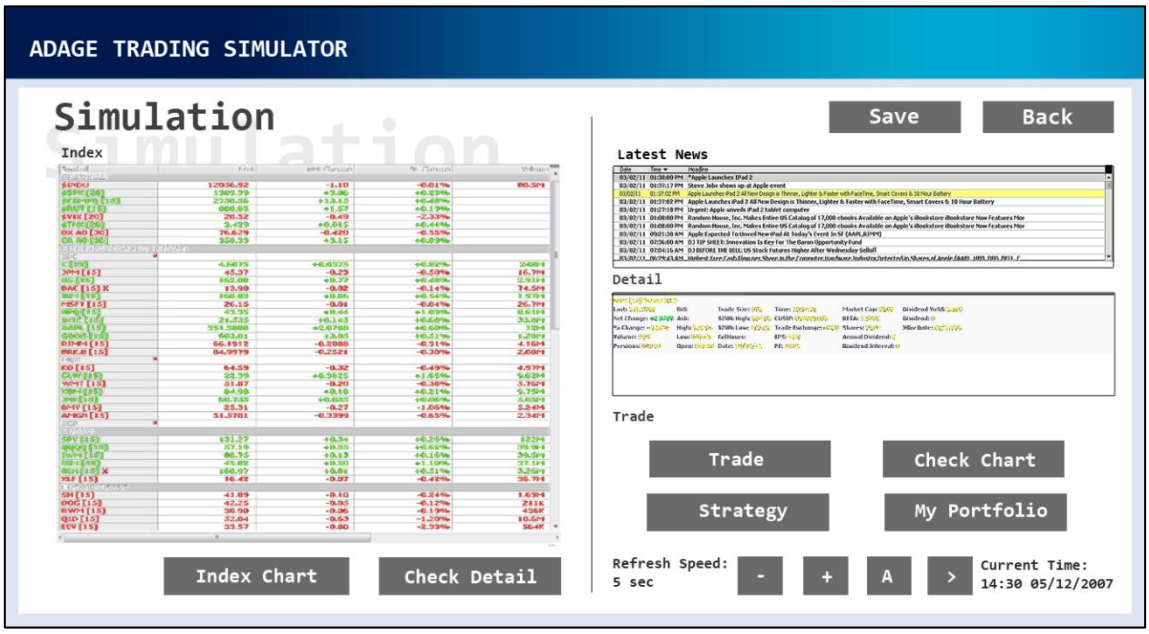


Diagram4-2: Simulation pond

Diagram4-3: Chart Diagram

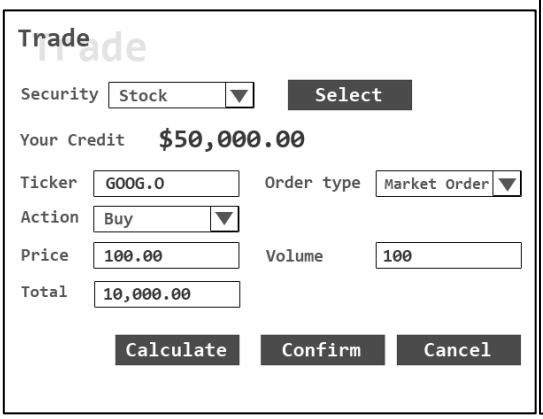


Diagram4-4: Stock

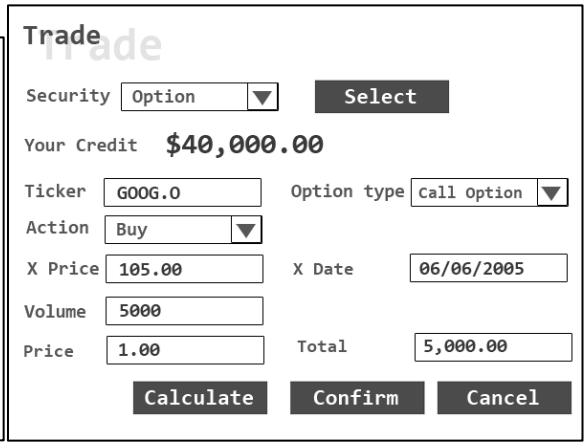


Diagram4-5: Option

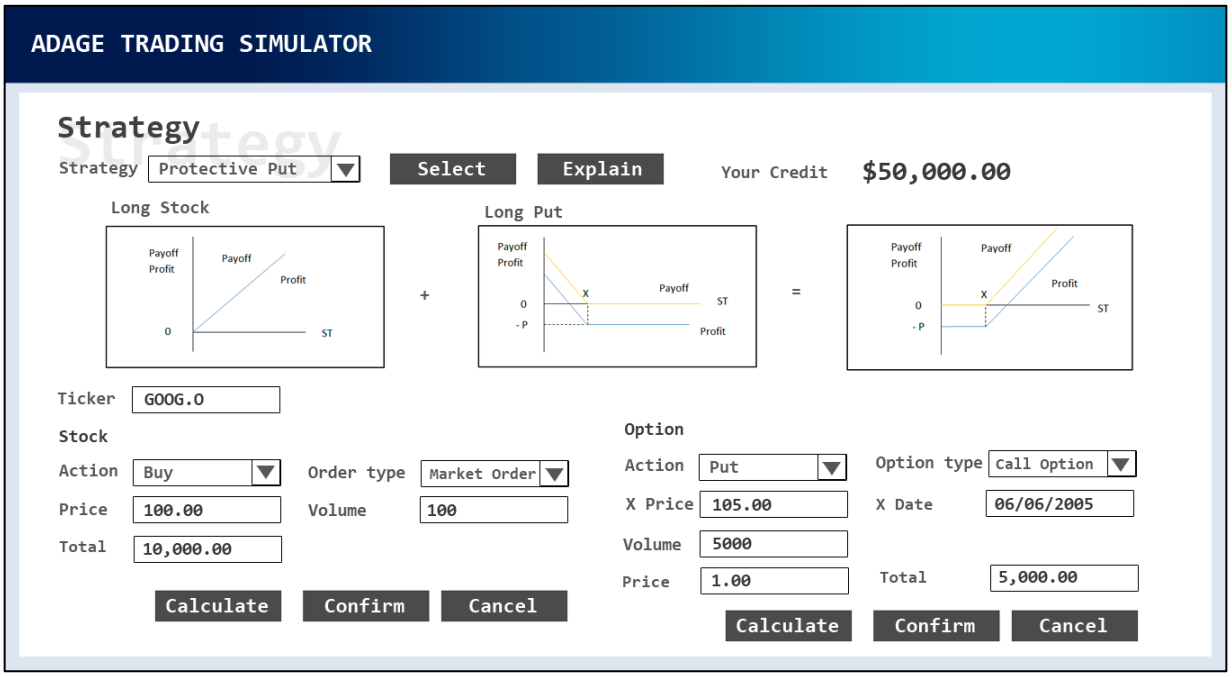


Diagram4-6: Strategy trading

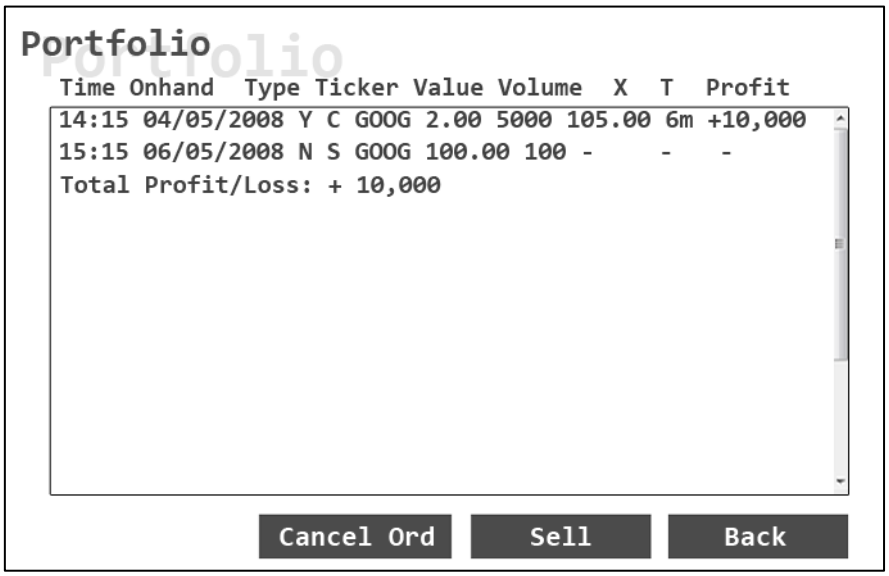


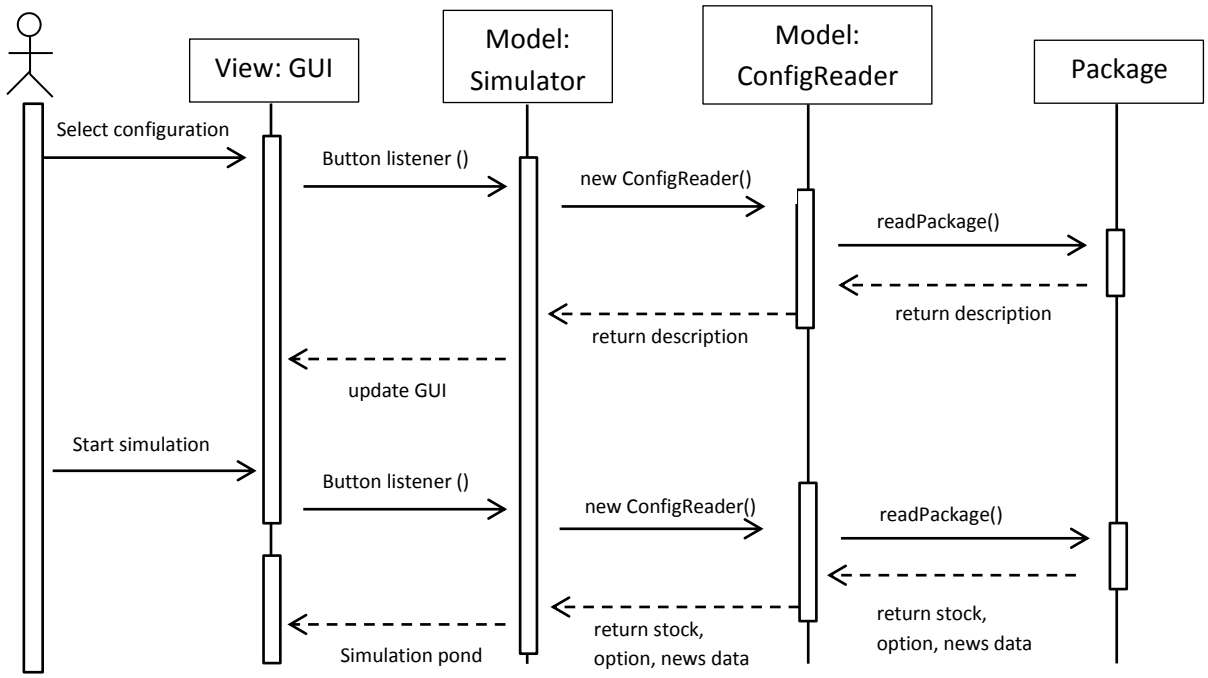
Diagram4-7: My Portfolio

4.4 Use Cases and Sequence diagram

The use cases focus on DN-2.2, 2.4 and 3.1 in the listed requirement from the following section.

DN-2.2	"Select configuration" Use case
Use Case Name	Select configuration
Description	Users could select a simulation configuration from a collection of configurations, which comes from the downloaded packages
Actor	Logged in user
Pre-Conditions	• A user has logged in • There are configuration packages available under simulator directory
Basic Flow	1. User selects the configuration package 2. System read and display description of the configuration on the left hand panel 3. User clicks on start button 4. System loads the configuration, read stock data, read option data, read news data, read parameters (see "Read configuration" use case), initializes user data
Post-Conditions	• The simulation pond is shown • Simulation starts
Alternative Flow	Simulation could resume by clicking "Load" button if there is saved session

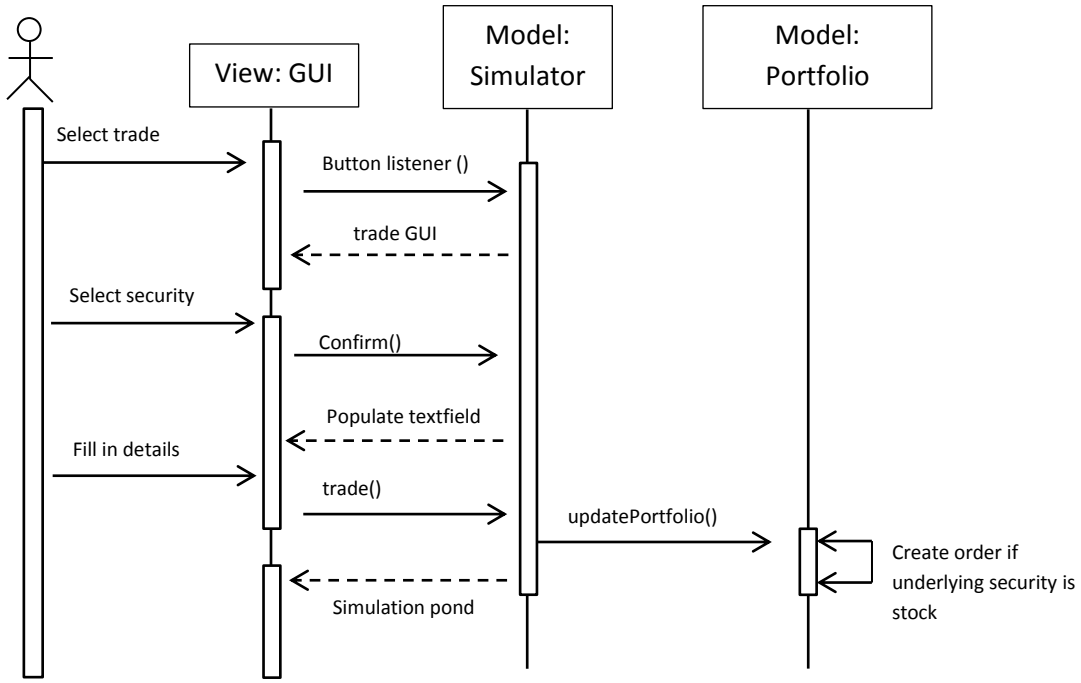
Use Case 4-1



Sequence diagram 4-1

DN-2.4	“Trade security” Use case
Use Case Name	Trade security
Description	Trading of stocks and options. A separate view is displayed upon trading requests
Actor	A user in a simulation session
Pre-Conditions	• Simulation is running • View of the system is at simulation pond
Basic Flow	1. User click on “Trade” button 2. “Trade View” window pops up 3. User selects the type of security from the drop down box in the top of the window, click confirm 4. System populates the bottom panel with text fields of the corresponding security 5. User inputs the detail, click confirm 6. If the security is a stock, then the order is placed. If it is an option, then the system will automatically assign a price and update the portfolio
Post-Conditions	• Both order list and portfolio are updated • Return to simulation pond
Alternative Flow	User clicks on “Strategy” button in the simulation pond

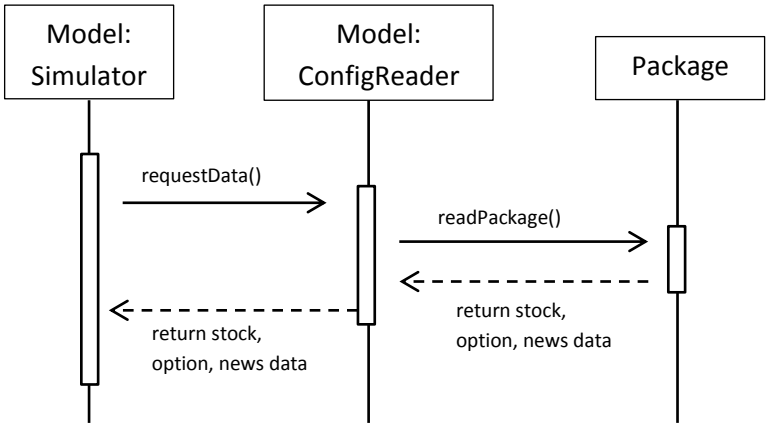
Use Case 4-2



Sequence diagram 4-2

DN-2.4	“Read configuration” Use case
Use Case Name	Read configuration
Description	The functionality to read stock data, option data, news data and other parameters from the package
Actor	System
Pre-Conditions	• User has selected a configuration or loaded previous saved session • A configuration must exist • Data of the following time unit is requested
Basic Flow	1. The system reads the file contained in the package 2. The system signals user from GUI when the loading is completed
Post-Conditions	Configuration data is loaded
Alternative Flow	None

Use Case 4-3



Sequence diagram 4-3

4.5 Class Diagram

Unified Modelling Language (UML) is a standardized, general-purpose modelling language in the field of software engineering. The Unified Modelling Language includes a set of graphic notation techniques to create visual models of object-oriented software-intensive systems.

It is considered as good practise to illustrate a class diagram as a blue print before implementation. It also gives the reader of this report a more comprehensive overview.

The diagram outlines the classes of the system. Only the key MVC related classes have been shown for the purpose of simplification.

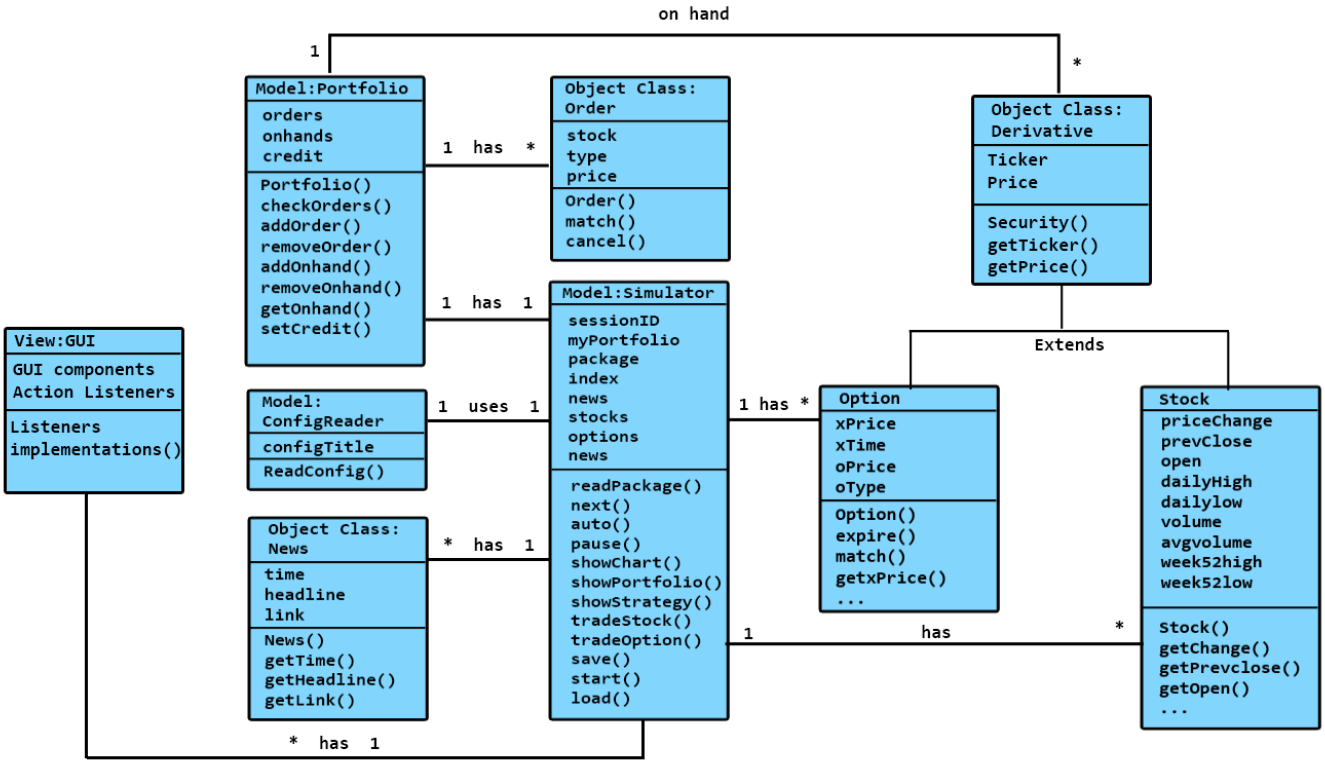


Figure 4-5-1 Class diagram

Control of MVC is integrated with the View. More elaboration can be found in the next section.

5. Implementation

The trading system has been successfully implemented after 10 weeks of development. This section outlines the language, environments, changes on the design and third party packages embedded in the system.

5.1 System development Environment

The system development kits include Eclipse, Netbeans, Git GUI and Photoshop.

Netbeans (Diagram 5-1) was used for the sole purpose of building GUI because it offers drag and drop functionality which has been proven handful for the tasks. However, it does not allow modifications on its automatically generated code with a friendly interface. That is where Eclipse comes into play. Eclipse (Diagram 5-2) was used as the main development kit simply because it is versatile and it has been the most familiar platform to me to write Java code. Git GUI (Diagram5-3) is the version control tool which keeps the working process on track. Photoshop was used to edit graphical material used in the application. This includes payoff diagrams and button backgrounds.

The overall development cycle follows three major steps: recreation of the proposed GUI design using Netbeans, copying the code generated to Eclipse and implementation, and committing in Git. This complies with the methodology proposed in 4.4. It ensures tasks are done efficiently with the best tool available.

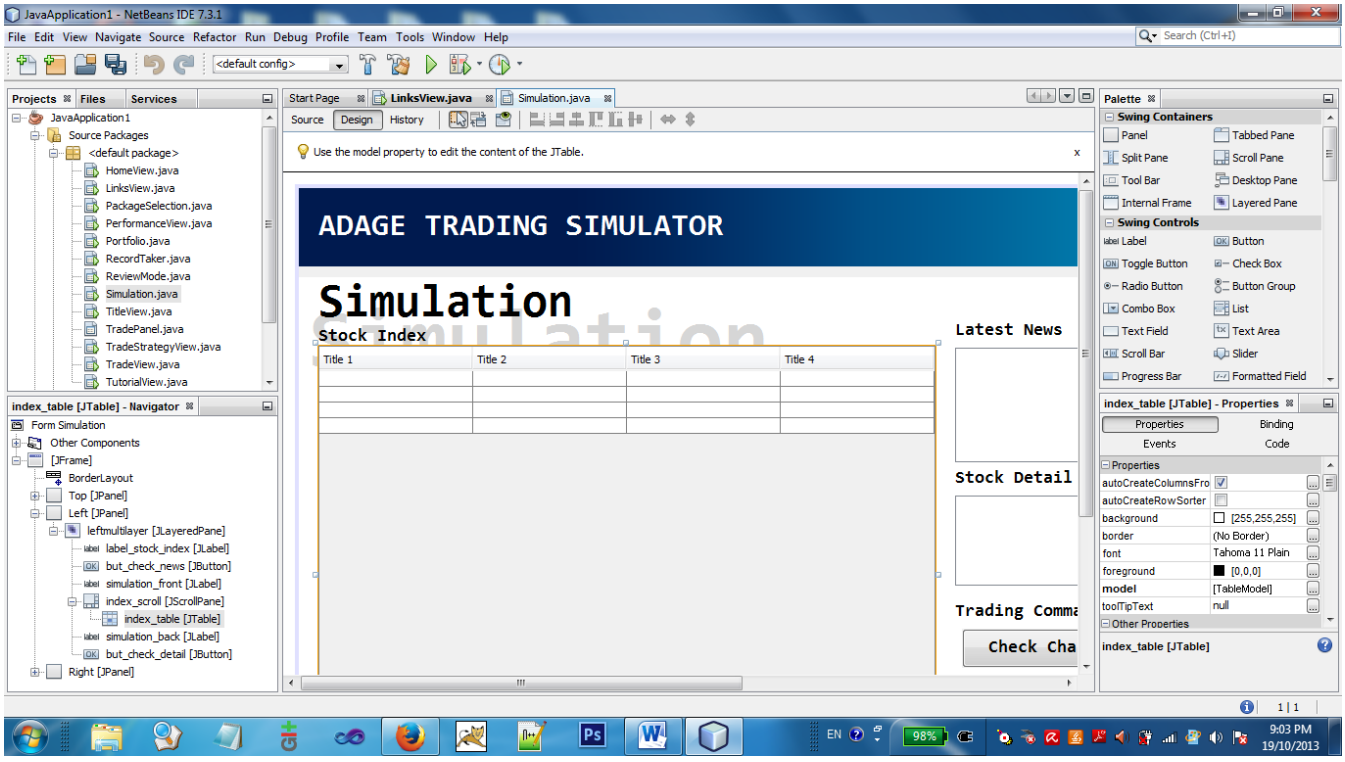


Diagram5-1: NetBeans

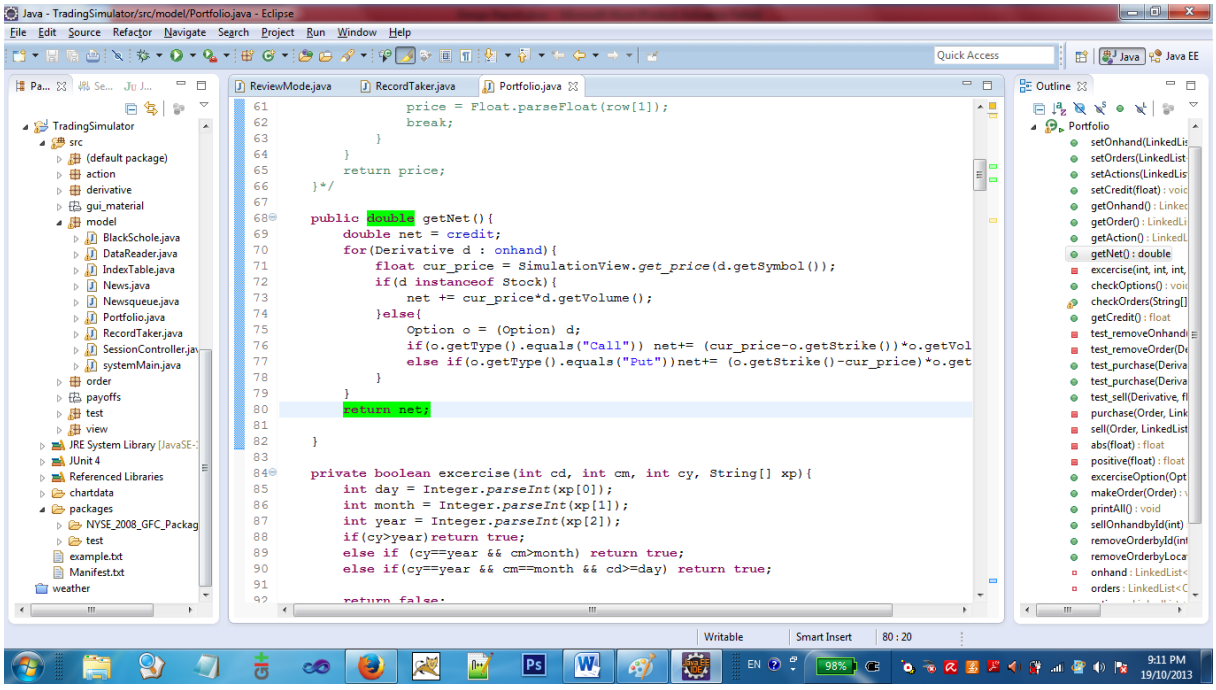


Diagram5-2: Eclipse

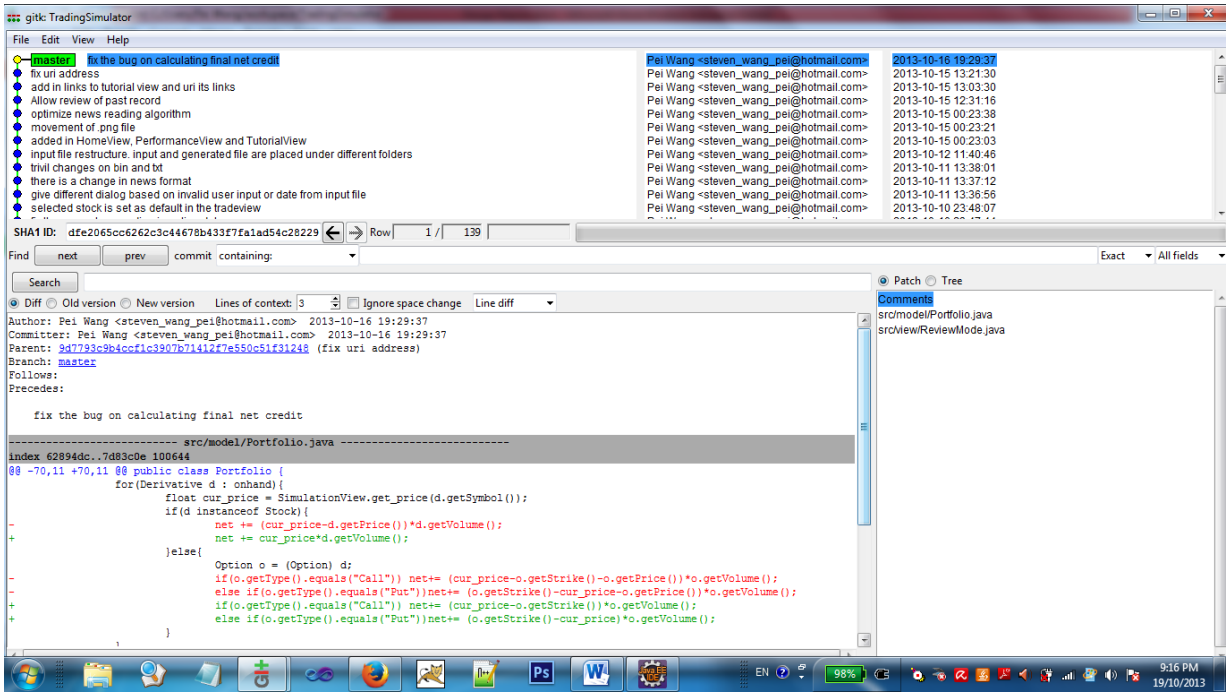


Diagram5-3: Git GUI

5.2 Code structure

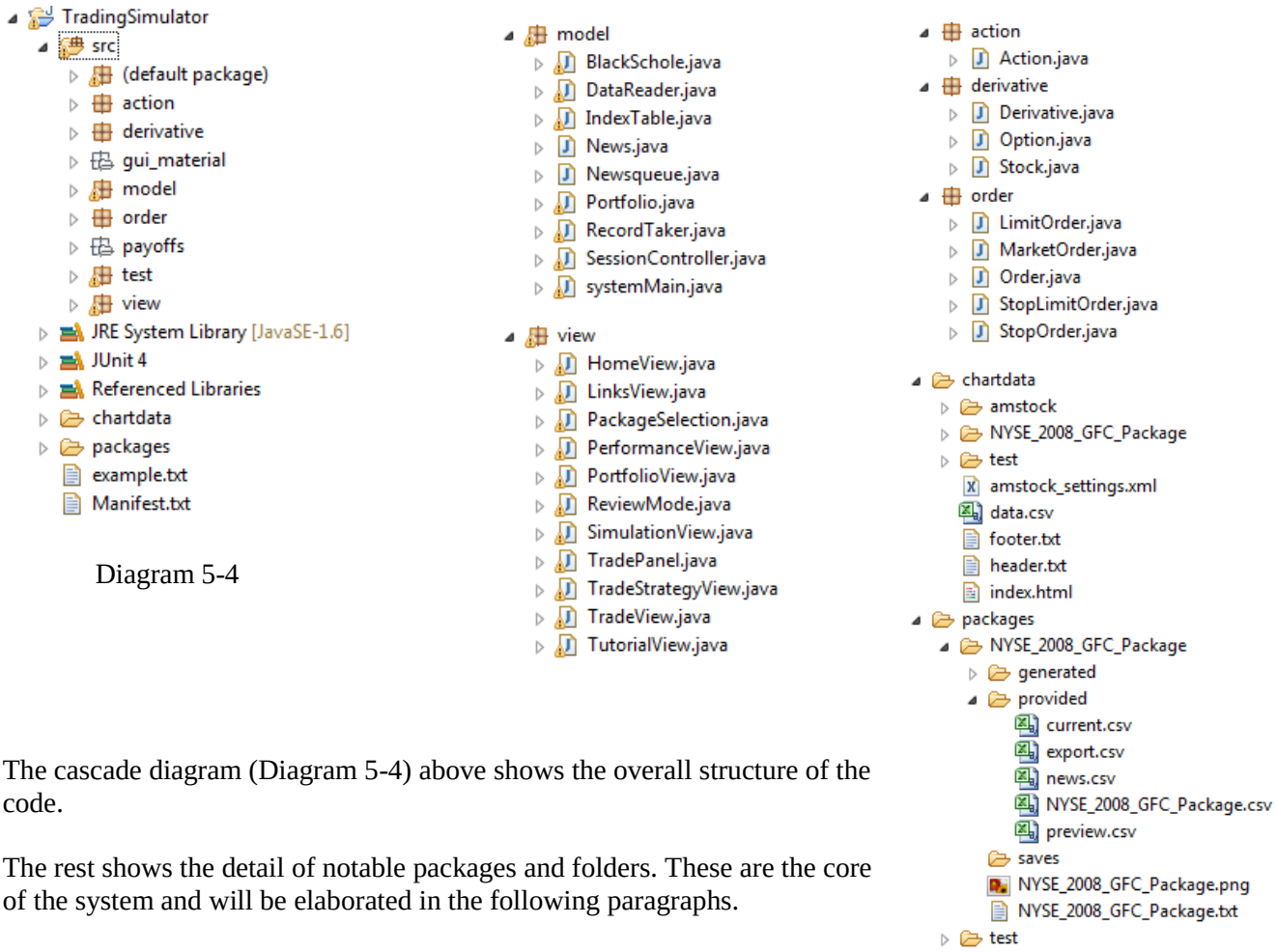


Diagram 5-4

The cascade diagram (Diagram 5-4) above shows the overall structure of the code.

The rest shows the detail of notable packages and folders. These are the core of the system and will be elaborated in the following paragraphs.

The system was built following a variation version of MVC methodology, which I would call “Design centered approach”. It is similar to the more renowned model “User centered approach”, but the focus is on the design rather than the user. The “design” in this case refers to the mock up diagrams created in the previous section of the report. Such model is followed for two reasons. Firstly, the design has already been refined for a whole semester to a degree, that it guarantees the feasibility and usability on a satisfactory level. Secondly, as the core and only developer of the system, I hold bachelor degree in both Software Engineering and Finance. My presence has met the criterion of “User centered approach”, so it is nature to go a level further.

As shown above, there is no “control” package in the project. The “control” package has been merged into the “view” package. This is done because as the code generated by Netbeans, corresponding listener calls are also generated in the same file (Diagram 5-5). It would be unwise to tamper the already well organized structure for the sole purpose of complying with the doctrines of the tradition model. Therefore, the “view” package contains GUIs and their corresponding listeners. For example, HomeView.java refers to the GUI first appears as the simulator launches. The buttons on the right hand side of the GUI invokes TutorialView.java, PackageSelection.java and PerformanceView.java respectively.

On the other hand, the essence of MVC has been preserved in some level. The “model” package contains the modules which process the logic of the said model. For example, when the user initiates a simulation by selecting a package from view/PackageSelection.java to invoke view/SimulationView.java, an object of model/DataReader.java and an object of model/BlackSchole.java (section AP-2 for detail) are created. The prior read in the provided csv files from the configuration package and the later estimates the implied volatility for the each of the derivative in the csv file. However, there is no “Model: Simulation” class as proposed in the sequence diagram of design section. This makes the logic within the model package loosely correlated, as many logic processing happens under the view package. Static class such as “Indextable” was implemented as amendment. Such method of implementation should be revised.

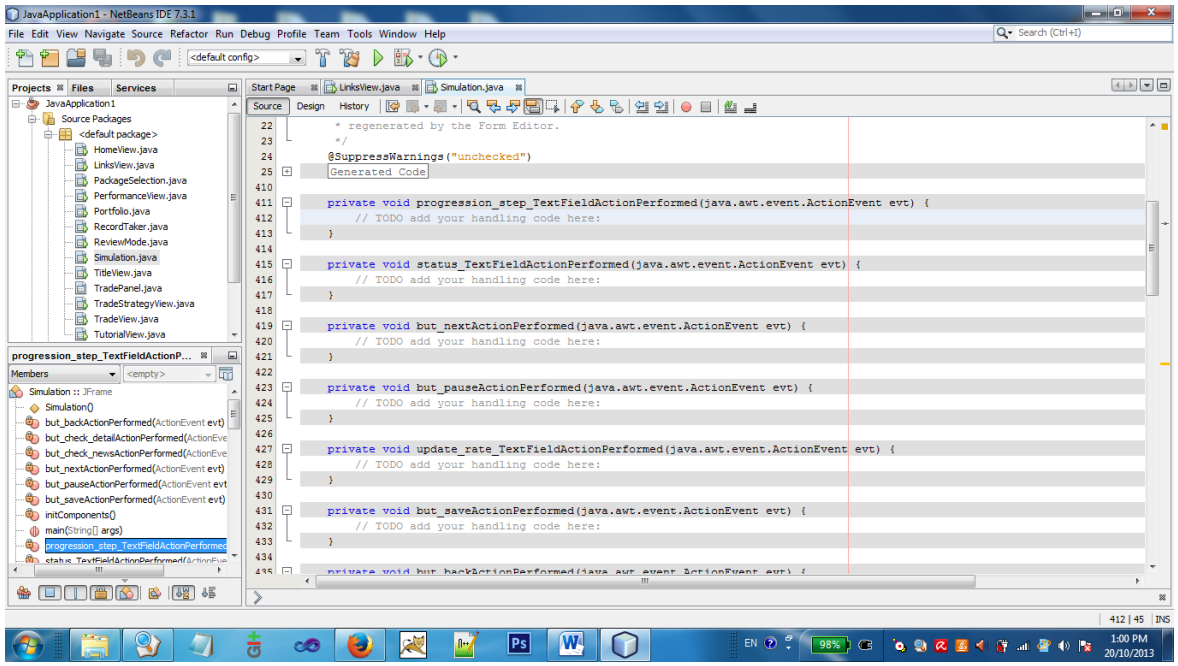


Diagram 5-5

Moving on to the "action", "derivative" and "order" packages, which contain class whose objects are created due to user's interactions with the simulator. For example, an object of "Action" class records everything happens on user's on hand orders and derivatives. This includes long or short order fulfilments, order cancelations, option exercises. Each action object also records the underlying derivative of each action for booking keeping and review. The "derivative" package contains Stock class and Option Class, which are extended from the Derivative class. The "order" package contains different orders on demands.

The folder "chartdata" contains the third party software amChart, which is used to display chart of a given stock.

Folder "packages" stores the configurations shown in the "PackageSelection" view. Assuming we use configuration named "NYSE_2008_GFC_Package" as the root folder. There are three sub folders and three files. Sub folder "provided" contains five files: NYSE_2008_GFC_Package.csv which contains the daily stock data obtained from TRTH, news.csv which contains the daily news data obtained from Sirca. The other three files contain information to calculate implied volatility required for option. More elaboration is necessary on these three files. "export.csv" and "preview.csv" each refers to the button on TRTH used to generate the csv spread sheet. "current.csv" contains the price of the stock on the first day being used to generate the other two csv files. It is called "current.csv" because the date is very likely to be a recent day of when the files are generated. That means the calculated volatility is the volatility outside the time frame of the simulation. For example, "NYSE_2008_GFC_Package" has the time frame from September 2008 to March of 2009, but the volatility of the stock are generated using the data from 2013, which is the year this report being written. What is the reason for this inconsistency? Unlike stock, which only has time and price, there is a unique option of the same stock for every maturity and expiration date. This creates up to hundreds of option price of the same stock to record in one time unit. Therefore, TRTH has no record on out-dated options. That is, if historical data is still available on their server, great. Otherwise we have to assume the volatility remains the same. It is a compromise we have to make.

Moving on to the "generated" folder, files are generated in the step "Write new files" to "generated" folder" in Diagram 5-6. This happens after DataReader reads in "NYSE_2008_GFC_Package.csv", where all stocks are listed in one file in time sequence. The generated files separate stocks to several individual files. Hence, it reduces memory space required during run time, because the system does not keep stock data inside system memory.

In addition, the "save" folder contains the data of the saved session, so user could resume the progress from last time.

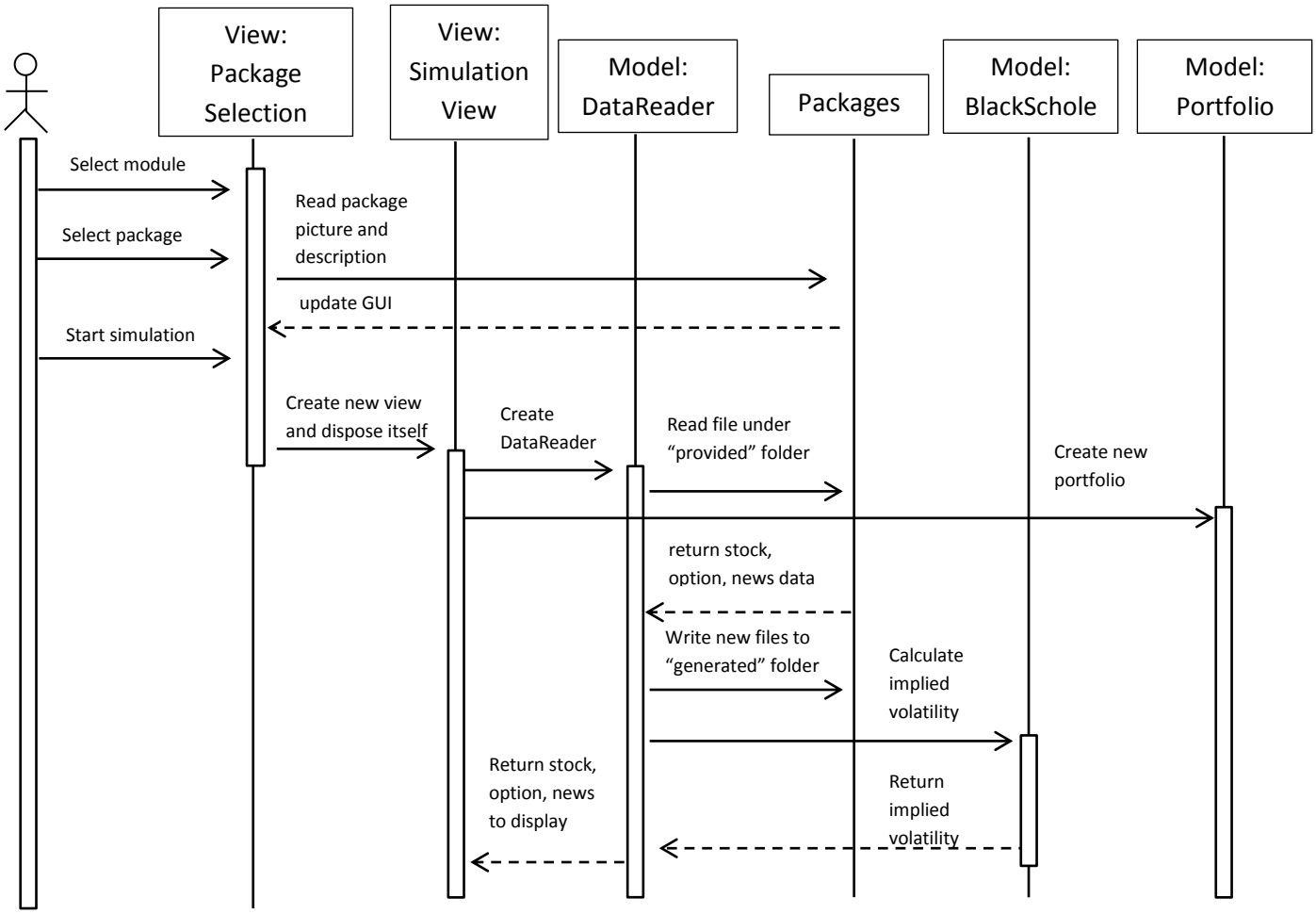
The png and txt files under the root folder are displayed in "PackageSelection" view. The "record" file is a leader board. It records the name of user and final credit of user's portfolio when a simulation has finished. The list is displayed under the "Performance" module.

5.3 Use Cases Implementation

This section refers to the use cases raised in 4.4. The followings are the actual implementation of the proposed use cases. The "Model: Simulation" entity has been replaced by classes under the View package. This is an imperfect realization of MVC design and should be revised.

"Select configuration" -- initialization of a simulation session

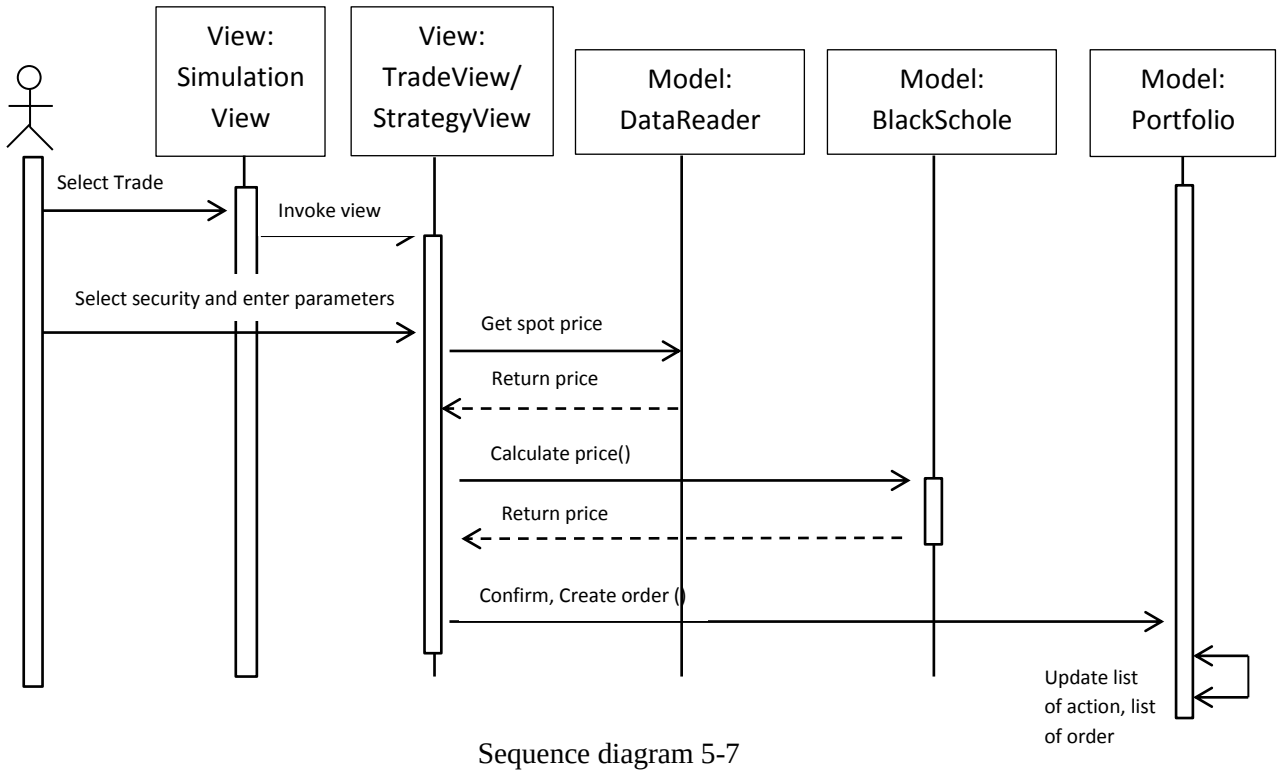
Use case manifest remains unchanged, except user no longer has to log in.



Sequence Diagram 5-6

“Trade security” – make an order

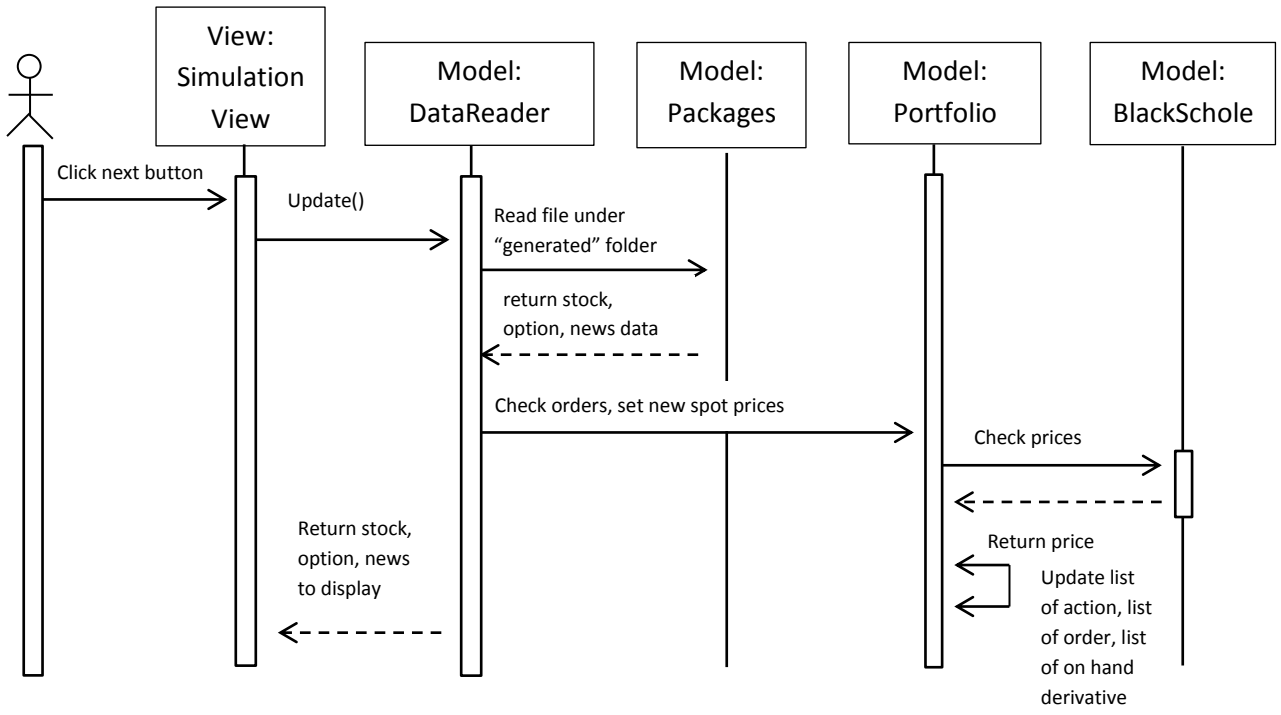
Use case manifest remains unchanged.



Sequence diagram 5-7

“Read configuration” – happens when simulation reads in data of the following day.

Use case manifest remains unchanged.



Sequence diagram 5-8

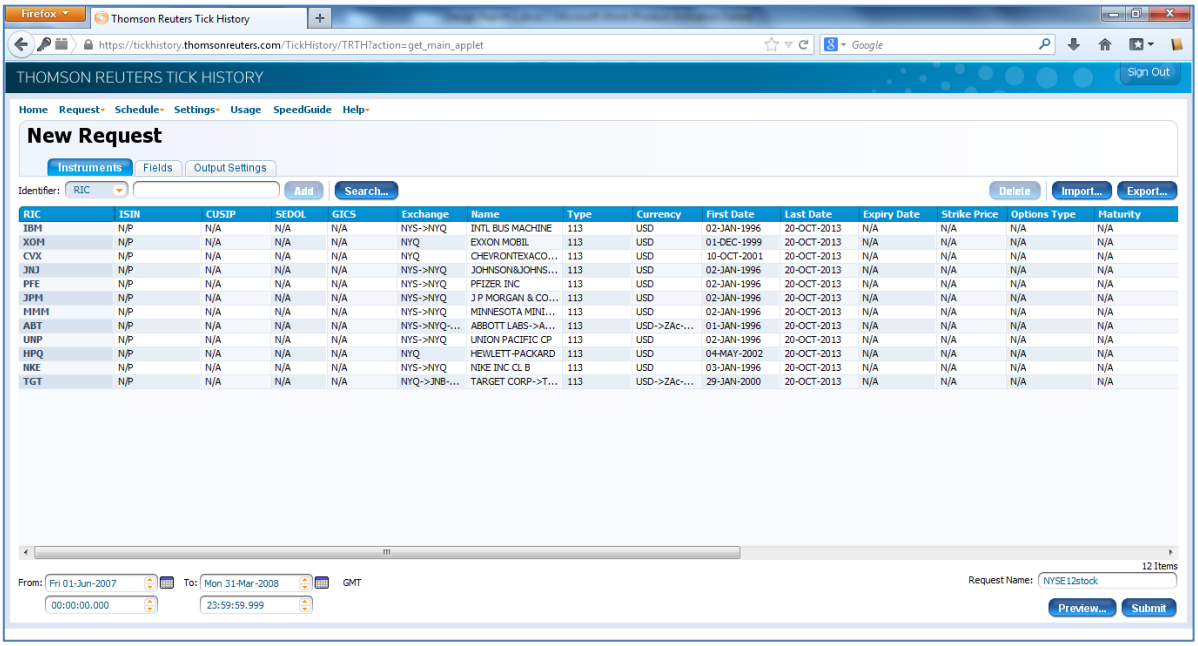
5.4 Third party package and services

The project is built on the shoulders of several third party packages and services. They are used either to reduce implementation difficulty or because it is the only source of information available. AmChar, Thompson Reuter Ticker History and Sirca Portal are used during implementation.

AmCharts is a robust charting tool that will suit many data visualization needs. A screen shot of the output chart has been shown in the previous part of the report. It gives an explicit view of the movement of stocks. This is considered a standard for a trading simulator, so such function is realized.

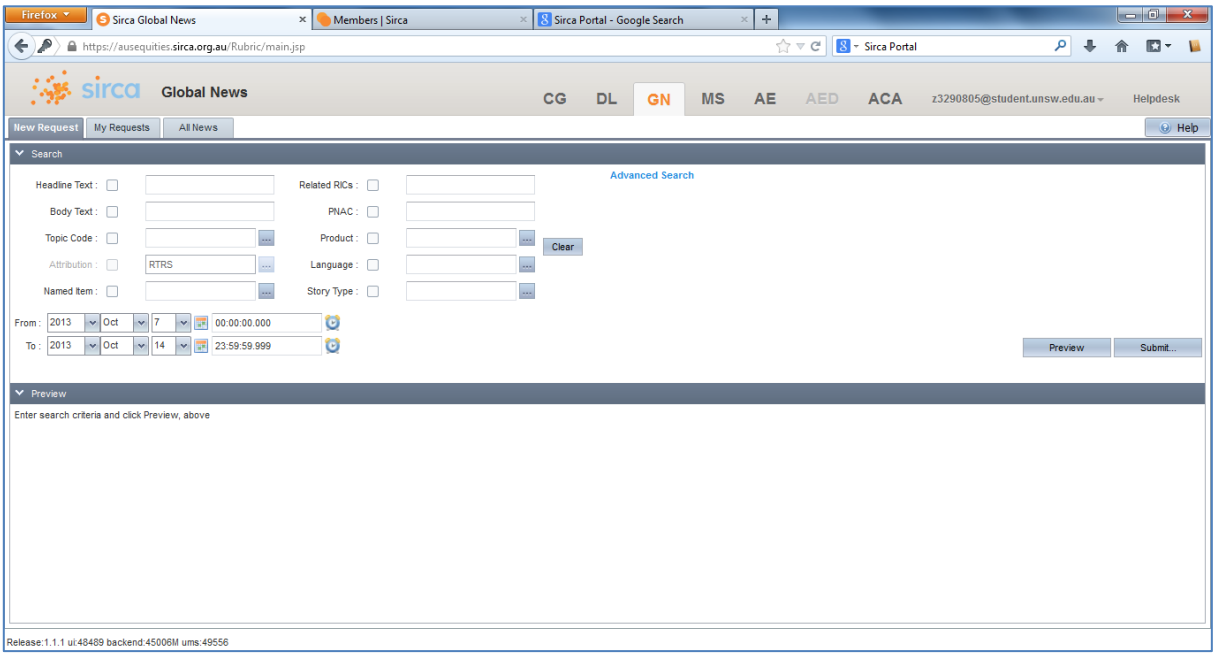
amChart

Thomson Reuter is the world's largest international multimedia news agency. Thomson Reuter Ticker History (TRTH) is one of their services to provide finance and stock market data. Input files under the “provided” folder of each package are all generated from their services. In another word, the system is built around the input data of their format. As shown in the screen shot below, you may find “Export” and “Preview” buttons. The output by clicking these two generates the file with the corresponding name under package folder “provided” with “csv” as extension. It is suggested when gathering data using their service, always make sure the data is daily (New Request tab > Message Type > End of the day) and Output Settings (tab) has Delivery Format as “Single file by RIC” and Data Format as “Results in RIC Sequence”. It is format supported by current version of the simulator.



TRTH

Sirca is a provider of online services to support financial and other data-intensive research at universities, Government and financial market participants. Their portal is used to gather news data. When using their service, it is advised to use advanced search then follow the syntax provided in their help document to aggregate news on several stock at once. For example: HT:IBM OR HT:XOM OR HT:CVX, where HT stands for Headline Title. The reason why not to search on the RIC, which supposes to give more accurate output, is because the real RIC is in the format of "IBM.N". The searching query confuses on the dot. It happens even when proper regular expression is supplied.



Sirca GN Portal

6. Evaluation

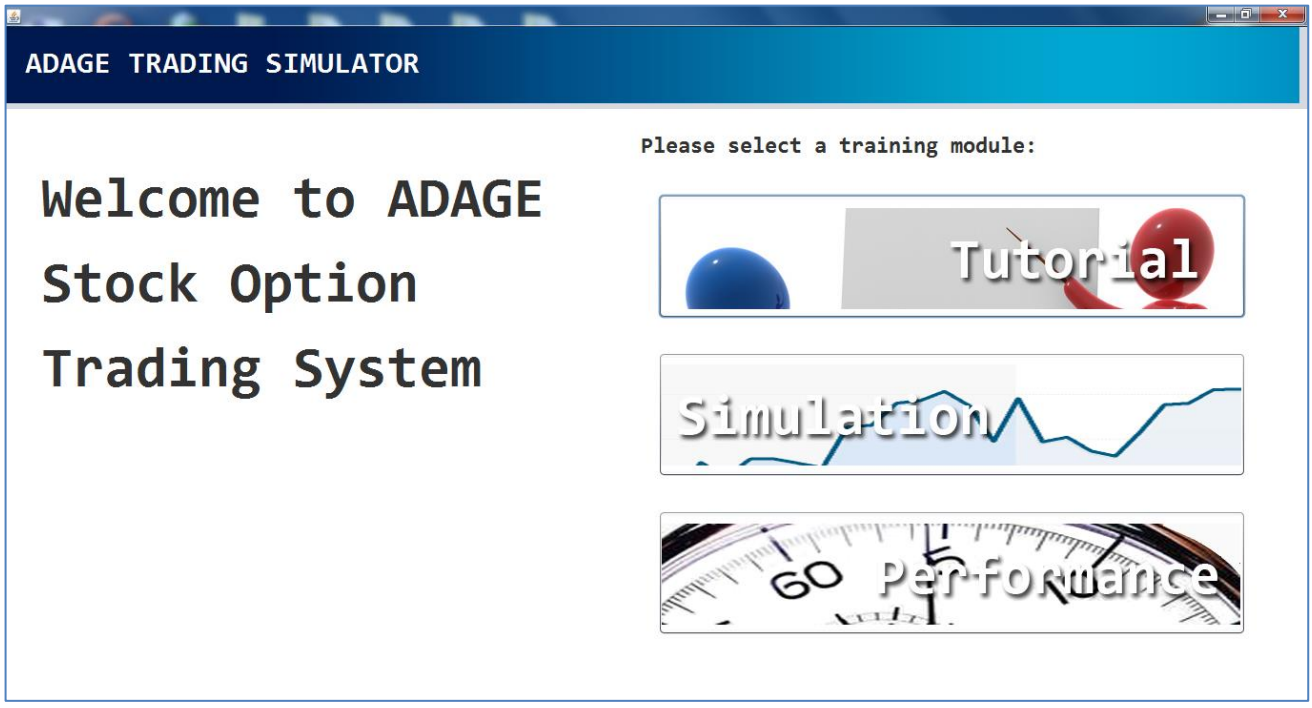
6.1 Validation

We are going to cover the usage of the project. The use case is very much similar to the one articulated in section 4.3. We are going to launch the system, and demonstrate the functionality with assistance of screen shots.

1. Extract and find the runnable jar

   It should look like the diagram above after the extraction of the compressed the file. Make sure three files are placed under the same directory.

2. Execute the runnable



   The Home page of the simulator is shown above. The size of the view is 1366 * 768. It has a welcome message on the left hand side and three clickable buttons on the right. Each button has its distinctive look. We are going to label the “Tutorial” module 3-1, “Simulation” 3-2 and “Performance” 3-3.

The Three Modules

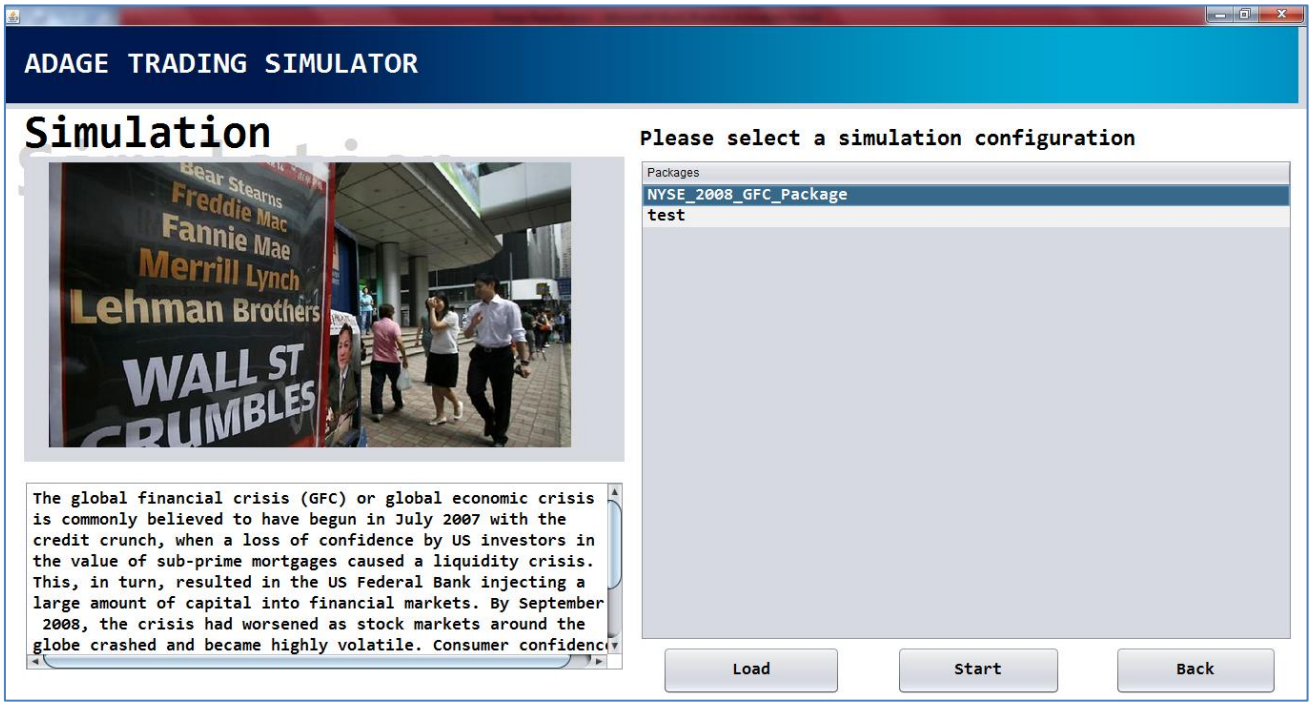
The “Tutorial” module contains links to web page which introduces requirement knowledge of financial trading. The “Simulation” module is the core of the simulator. It allows user interaction with historical data to practise their trading skills. The “Performance” module shows the leader board of user’s past performance. It also allows user to review their saved sessions.

5.5.1 Tutorial

Tutorial View

A default web browser is launched upon the click on the buttons to pages explaining the topics. The “Links” button starts a dialog which contains links to other parties such as UNSW CSE, ASB and email address for tech support. Clicking on the Back button goes to the previous view.

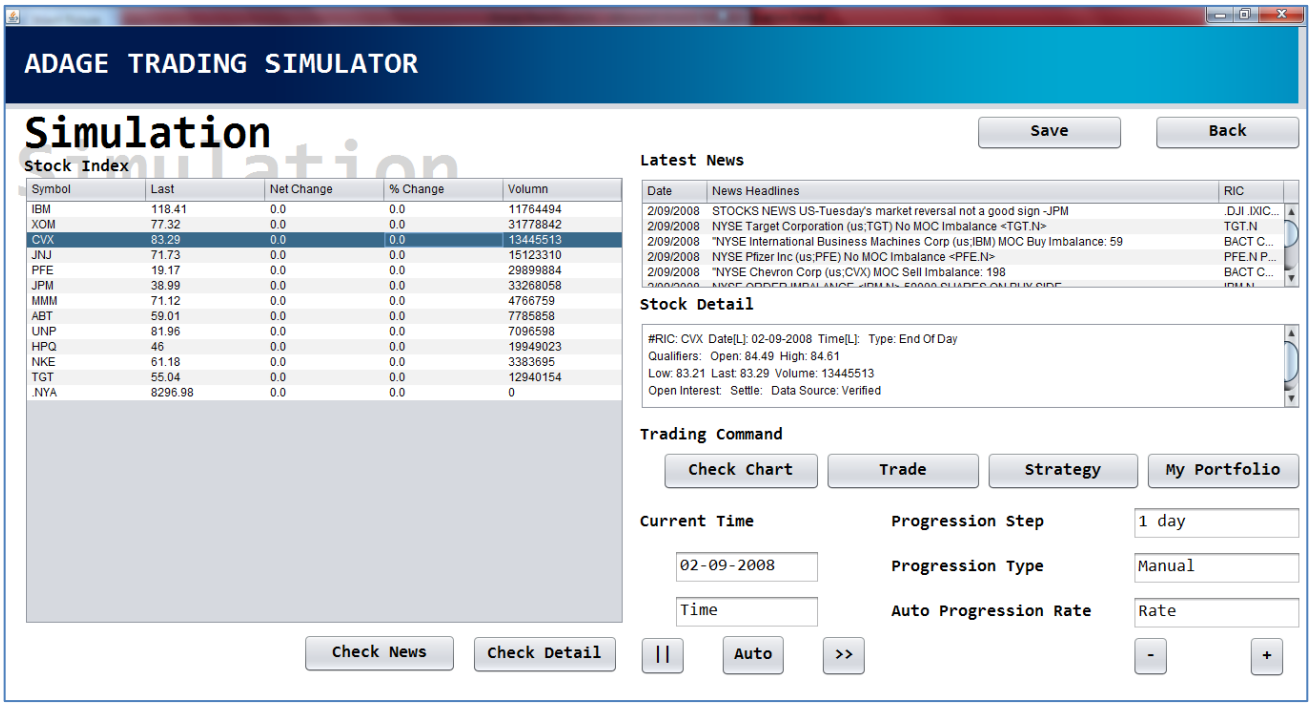
5.5.2 Simulation



Simulation Selection

Selecting the second module on the home view invokes the package selection view. Clicking on the package name on the right hand side populates the left half panel with picture and description. Clicking on the “Load” button invokes a selector window with the “save” folder under the corresponding package as the default path. Confirmation on the selected save file resume the saved session. Otherwise, clicking on the “Start” button starts a new simulation session, which is the next screen.

This view is relative more complicated comparing to the previous ones. Please bear with me as I explain it thoroughly.



Simulation View

Simulation starts on the date 02-09-2008.

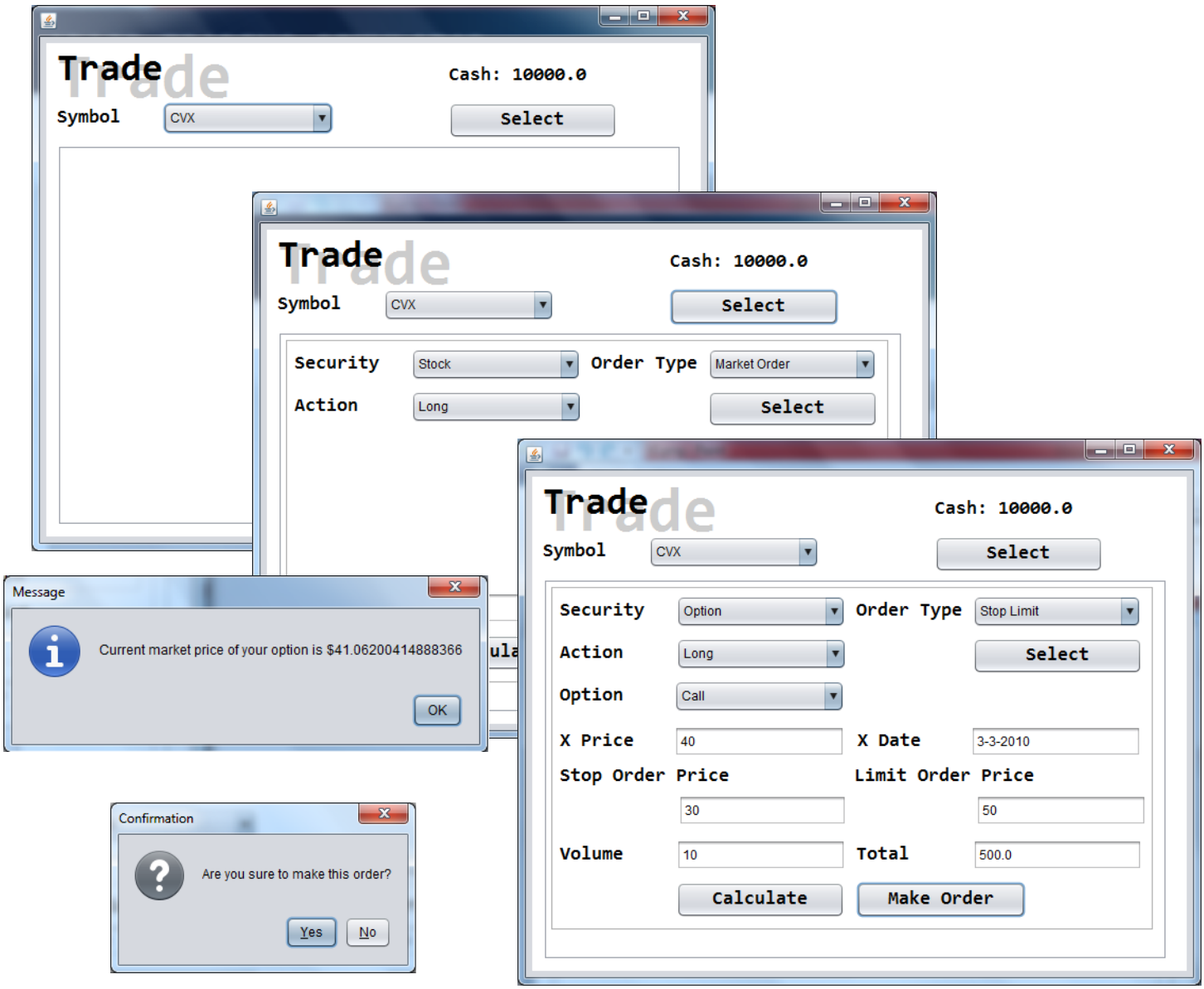
On the left hand side, we have the "Stock Index" table. It displays the current price of the stocks. "Symbol" column shows the RIC (Reuter Instrument Code) of the stock. "Last" shows the price. "Net change" shows the price difference from the previous day. "Net Change" shows the percentage of such difference. "Volume" shows the trading volume. Stock can be selected by simply clicking on the row. This action populates the "Stock Detail" text field on the right hand side. Beneath the "Stock Index" table you may find two buttons. "Check News" button displays and only displays the news of the selected stock in the Latest news table on the right hand side. "Check Detail" button invokes a web browser linking to Reuter showing the detail of the selected stock.

On the right hand side, you may find two buttons on the very top. "Save" button invokes a new selector window, with the "save" folder under the corresponding package as the default path. Key in a name and confirm will save the current session in the file with the name. "Back" button brings the user back to the package selection view. Beneath the buttons, we find the "Latest news" table. It has three columns: Date, Headline and RIC. This table is updated automatically as the simulator proceeds to the following day. Beneath that we find the "Stock Detail" table. It displays the detail of the selected stock by showing all the columns from the input file provided. This table is NOT updated automatically as the simulator proceeds to the following day.

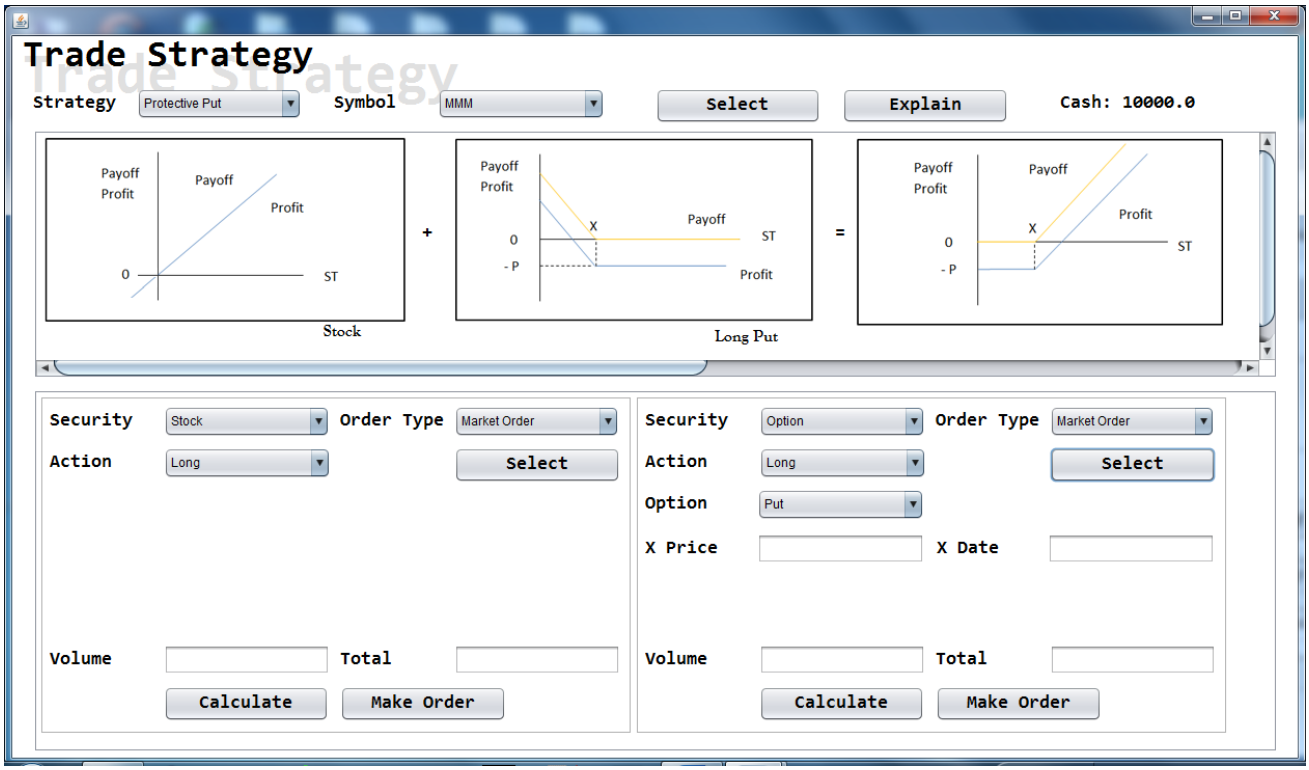
Moving on to the right lower quarter of the view, we have four buttons under the title "Trading Command". "Check Chart" button displays the chart of the selected stock from the starting date of the time frame in a web browser window. The Chart displayed on next page shows the chart of CVS on 08-09-2008, several days after the starting date.

"Trade" button starts a new window. Security combo has the default value the same as the selected stock, but the user is free to change to another security. Clicking the "Select" button populates the panel below. Then users could choose the type of derivative (stock or option) and the type of order (market, limit, stop, stop limit). Diagrams are shown on the next page. After key in the prices, volumes and dates in the text field, "Calculate" button calculates the price of the underlying asset of the order. If it is an option in concern, pressing the button will also invoke BlackSchole module (section AP-2 for detail) to make a fair calculation on the option. Such number is displayed in a Message dialog. A confirmation dialog will pop up after clicking on the "Confirm" button, asking whether the user truly wants to make such order.

Chart diagram



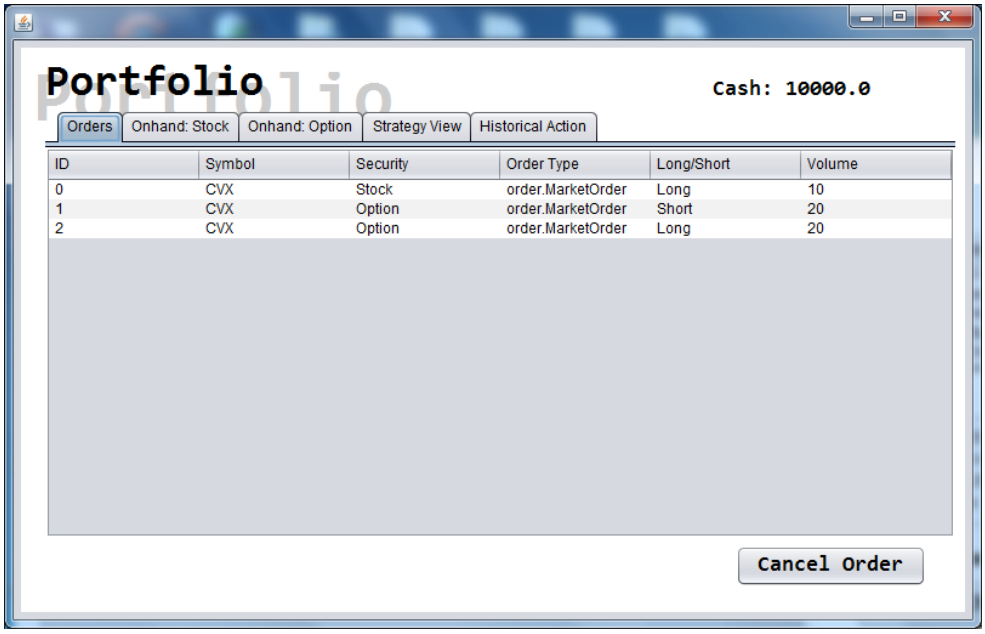
"Trade" button starts a new window, similar to the "Trade" button. However, it offers users to utilize trading strategies which they learned from the tutorial module. Several trading strategies are included at this stage but actual number of combination is beyond counting. Users are expected to use this view as a stepping stone to learn the universal rule, which is shaping their own trading strategy depending on market movements. For example, the diagram below shows the Trade Strategy view of a protective put strategy. Users have to select the strategy from the list and a Symbol then click on "Select" button to populate the lower half of the view with payoff diagrams and trading panel. The first two payoff diagrams depict a long stock and a long put, combining them form the third payoff diagram. The payoff diagrams serves as a reminder, a summary, but if users remains puzzled about the nature of the said strategy, the "Explain" button will bring forth a web page which explain the strategy in greater detail. The lower half is similar to the panel in Trade view, with some default value changed to conform to the chosen strategy for better user experience.

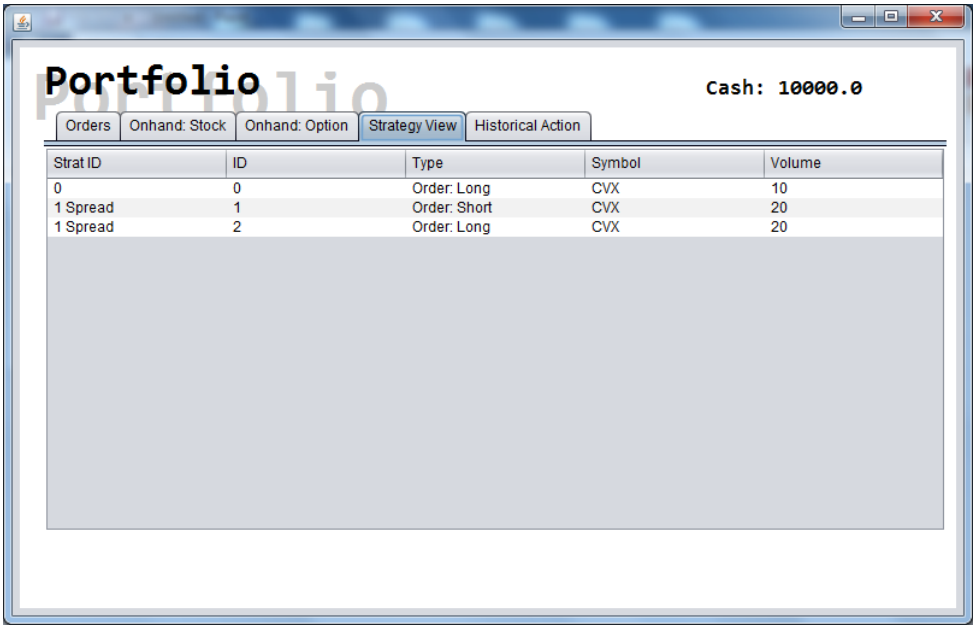


Trade Strategy View

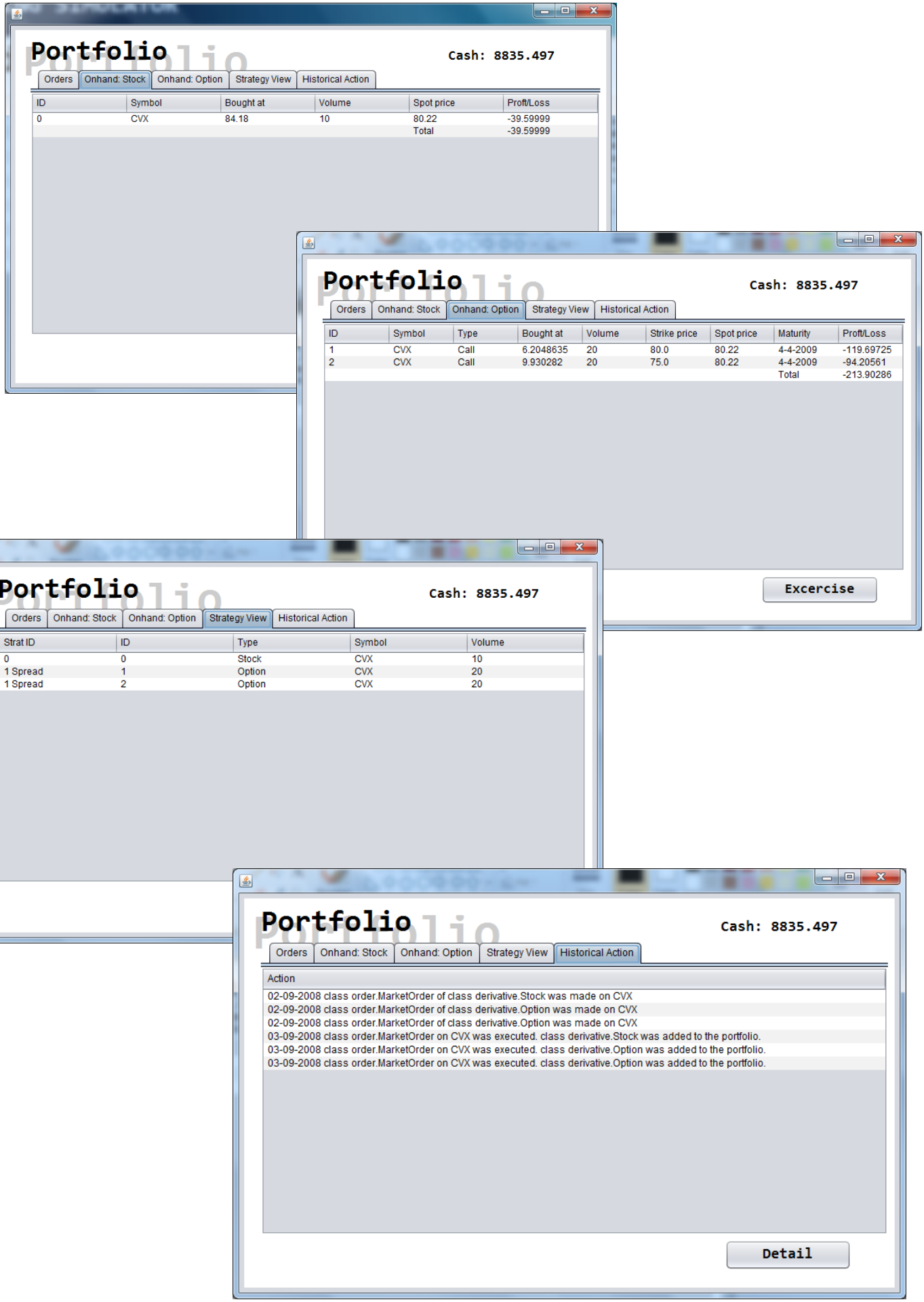
Last but not least, we have the "Portfolio" button. It starts a new window showing the current orders, derivatives on hand, strategies executed and all past actions. For example, imagine yourself in a position when you just made three orders. It consists of one standalone order to purchase 10 shares of CSV stock and two orders from a "Spread" trading strategy, as shown in the next page. The first screen shot shows the orders. The second shows the strategy utilized. Notice the standalone order only has its id in the first column. The third shows the actions.

Moving one day forward, the orders are executed. Screen shots are shown on the page two pages after this one. "Order" tab is now empty, whereas the "on hand: Stock" and "on hand: Option" gained their corresponding entities. "Strategy" tab changes the value in the "type" column. "Action" tab updated the latest changes, which are the execution of the orders. "Action" tab also records other changes such as cancelation of order, exercise of an option or execution of a stop limit order to form a limit order. However, these are not included in the screen shots. Users are recommended to try these features with the simulator.

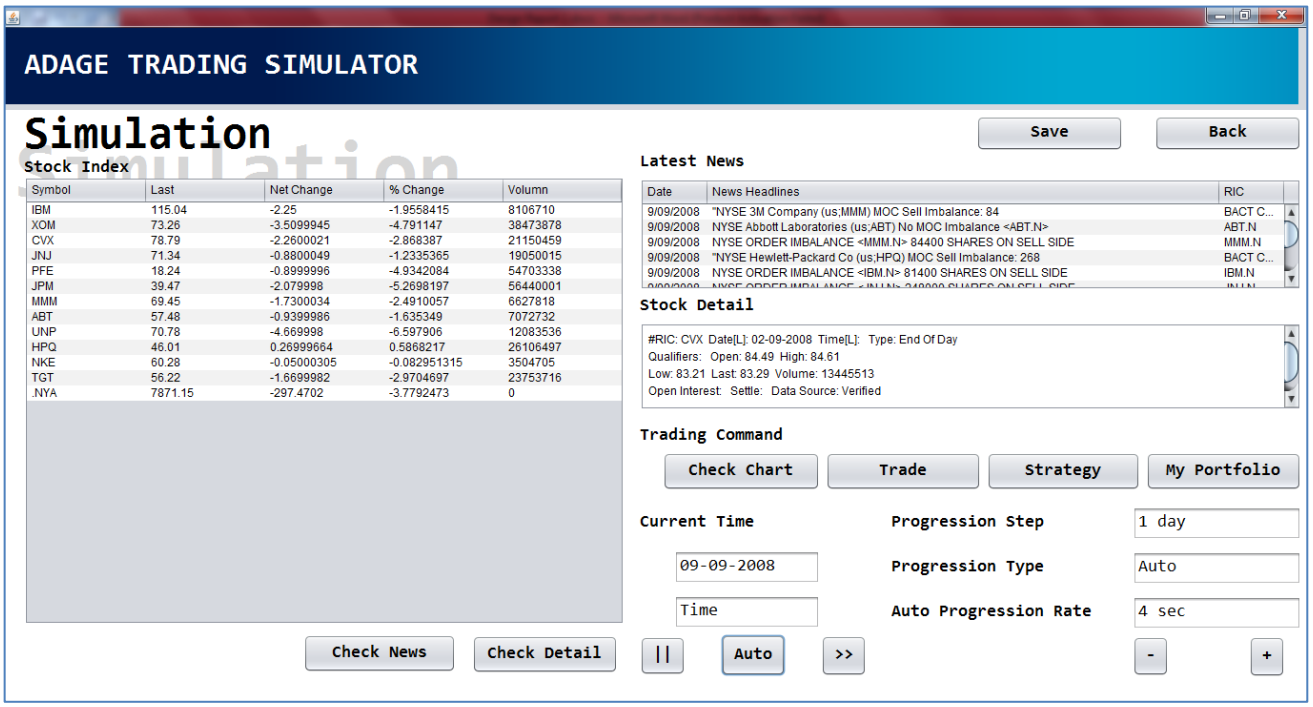

Portfolio
Cash: 10000.0
Orders
Onhand: Stock
Onhand: Option
Strategy View
Historical Action
ID
Symbol
Security
Order Type
Long/Short
Volume
0
CVX
Stock
order.MarketOrder
Long
10
1
CVX
Option
order.MarketOrder
Short
20
2
CVX
Option
order.MarketOrder
Long
20
Cancel Order


Portfolio
Cash: 10000.0
Orders
Onhand: Stock
Onhand: Option
Strategy View
Historical Action
Strat ID
ID
Type
Symbol
Volume
0
0
Order: Long
CVX
10
1 Spread
1
Order: Short
CVX
20
1 Spread
2
Order: Long
CVX
20


Portfolio
Cash: 10000.0
Orders
Onhand: Stock
Onhand: Option
Strategy View
Historical Action
Action
02-09-2008 class order.MarketOrder of class derivative.Stock was made on CVX
02-09-2008 class order.MarketOrder of class derivative.Option was made on CVX
02-09-2008 class order.MarketOrder of class derivative.Option was made on CVX
Message
Underlying derivative was class derivative.Option
Strategy:Spread
ID:1 RIC:CVX price:0.0 volume:20
OK
Detail

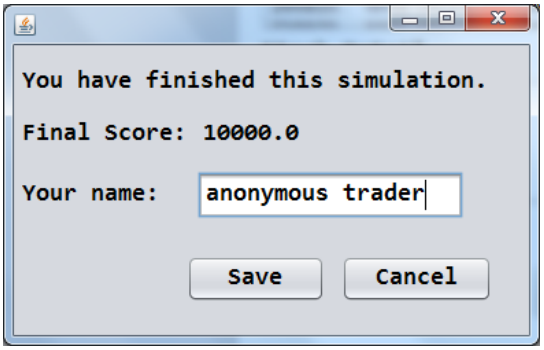

Portfolio
Cash: 8835.497
Orders
Onhand: Stock
Onhand: Option
Strategy View
Historical Action
ID
Symbol
Bought at
Volume
Spot price
Proft/Loss
0
CVX
84.18
10
80.22
-39.59999
Total
-39.59999
Portfolio
Cash: 8835.497
Orders
Onhand: Stock
Onhand: Option
Strategy View
Historical Action
ID
Symbol
Type
Bought at
Volume
Strike price
Spot price
Maturity
Proft/Loss
1
CVX
Call
6.2048635
20
80.0
80.22
4-4-2009
-119.69725
2
CVX
Call
9.930282
20
75.0
80.22
4-4-2009
-94.20561
Total
-213.90286
Excercise
Portfolio
Cash: 8835.497
Orders
Onhand: Stock
Onhand: Option
Strategy View
Historical Action
Strat ID
ID
Type
Symbol
Volume
0
0
Stock
CVX
10
1 Spread
1
Option
CVX
20
1 Spread
2
Option
CVX
20
Portfolio
Cash: 8835.497
Orders
Onhand: Stock
Onhand: Option
Strategy View
Historical Action
Action
02-09-2008 class order.MarketOrder of class derivative.Stock was made on CVX
02-09-2008 class order.MarketOrder of class derivative.Option was made on CVX
02-09-2008 class order.MarketOrder of class derivative.Option was made on CVX
03-09-2008 class order.MarketOrder on CVX was executed. class derivative.Stock was added to the portfolio.
03-09-2008 class order.MarketOrder on CVX was executed. class derivative.Option was added to the portfolio.
03-09-2008 class order.MarketOrder on CVX was executed. class derivative.Option was added to the portfolio.
Detail

Now we are moving on to the bottom right corner of the Simulation View as shown below, the progression control panel. Under the title Current Time, there are two text fields. The top one displays the "current" date and the bottom one displays the time. Since both packages are on daily basis and our simulator does not support intraday trading, the bottom text field only displays "Time". There are three buttons beneath the text fields. The right most button ">>" advance the simulation time by 1 day, causing new data to be read and displayed, portfolio to be updated. The middle button "Auto" allows the function described on ">>" button to run automatically. This is done by creating another thread calling the same function, which runs after clicking the ">>" button every n seconds. The value of n is shown on the Auto Progression Rate text field on the right. During automatic running, the text field next to the title "Progression Type" will display "Auto" as shown in the screen shot below. The left most button "||" will pause the auto running and switch back the "Manual" mode. In addition, the "+", "-" button increases and decreases the automatic updating rate.



Simulation View

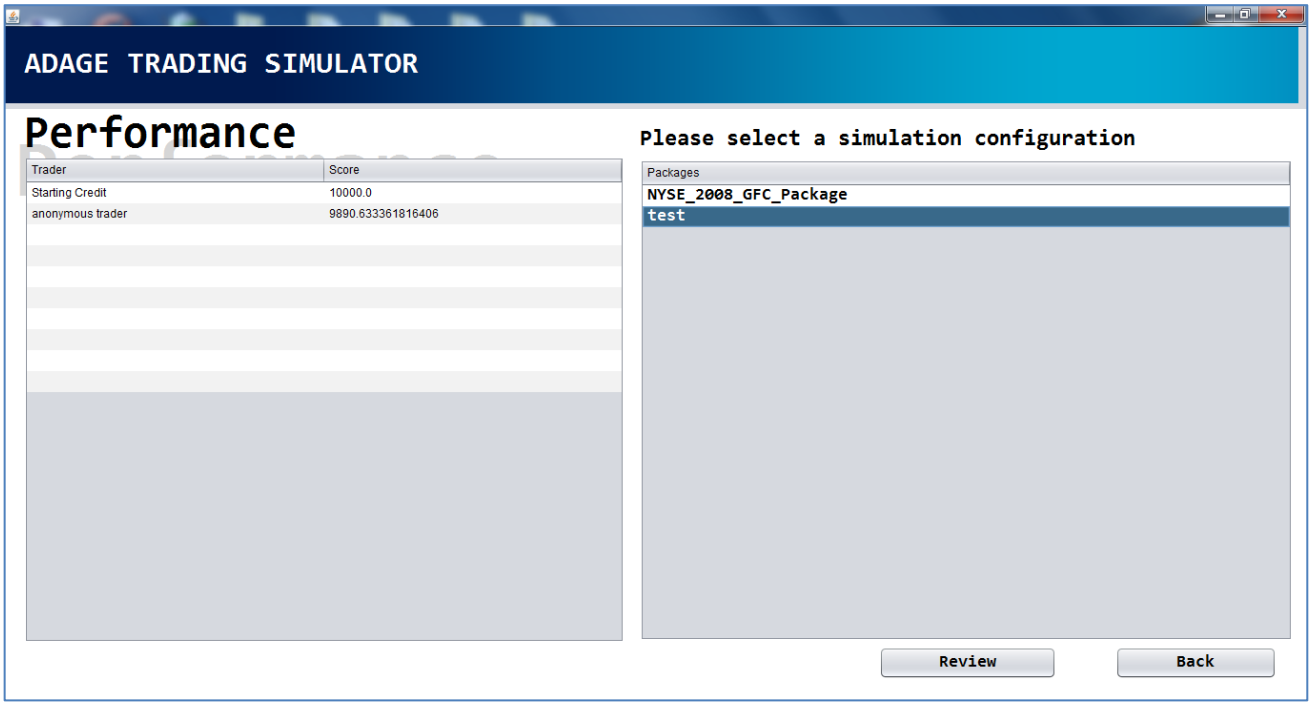
Another matter worth mentioning is that by the end of a simulation session, which is when the input data reaches an end. The system will notify and ask the user to key in their name. The name along with the credit, which is calculated by adding the current value of the derivative on hand to the cash, will be recorded in the "record" file under each configuration package. This score can be reviewed under the "Performance" module accessible from the Home View.



Record Taker

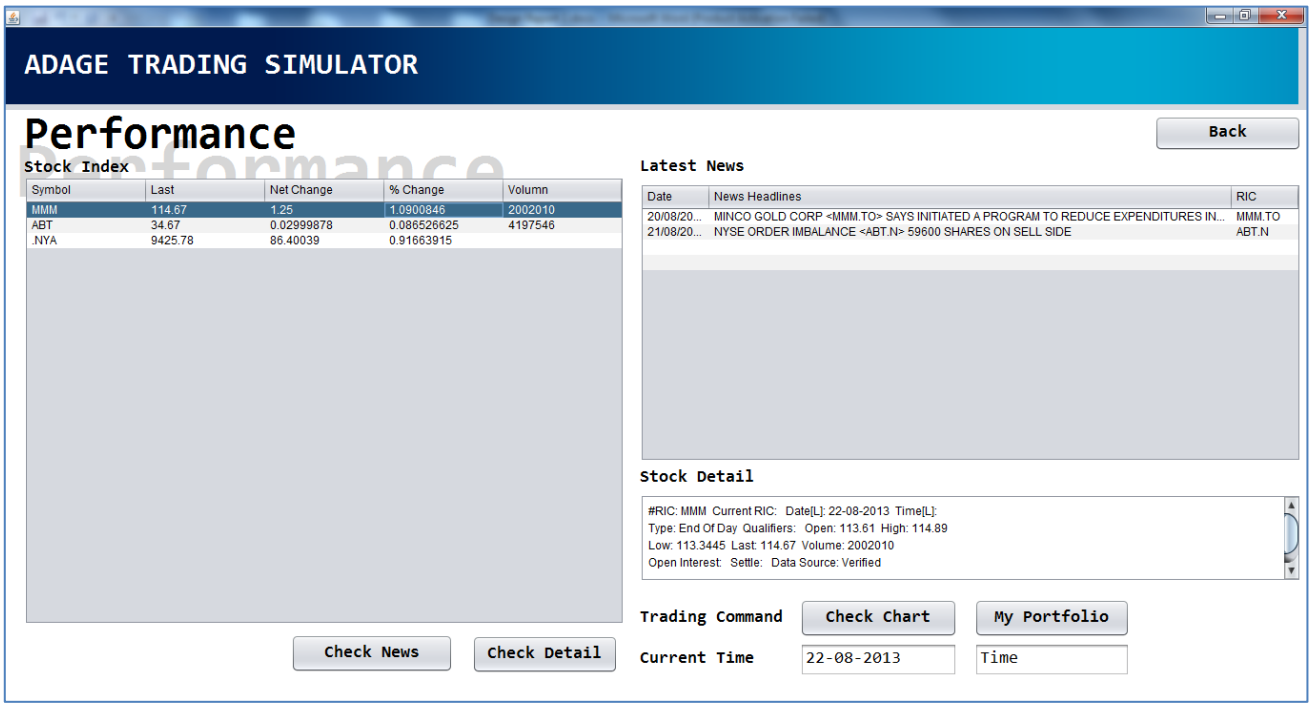
5.5.3 Performance

Performance is the third and the last module. It has a similar look as the Simulation Selection View, but the left hand side of the panel is a leader board displaying past session scores.



Performance View

Clicking on the “Review” button prompts a selector window with the “save” folder under the corresponding package as the default path. After confirmation it brings up another view as following for user to review the saved session. In this view, all components which allow time progression have been removed. The latest news table has been magnified to optimize usability.



Performance Review

6.2 Conclusion

The project delivered a working system that is able to be demonstrated. This section outlines achievements and possible improvements.

Objectives

The purpose of this project is to examine the existing software system by looking at their strength and weakness. Then propose and implement a new system to capture the essence of the existing system and fill in their functional gap, which is the lack of notification on the historical events. Has this problem been properly addressed? The answer would be yes but only partially. News data is displayed during the simulation, but only the headline is shown. This is because the raw data provided by Sirca was neither well organized nor user friendly. Only a small number of news has news body and some of the news simply does not make sense, because the lack of context or it is in a foreign language. That is, the initiative of the project was perfect, but the lack of support created a bottle neck.

In the discussion with my supervisor and assessor, the goal was to implement a simulator for the educational purpose of teaching people with no financial experience to trade derivatives using historical data. The simulator has lots of potential in education. It provides user real trading experience using historical data, and it allows customization on configuration package, so user may replay through the stock market in different eras. People could create their own package and test their trading skills. Despite the play value of the same package starts diminishing after the first attempt, it still offers an extraordinary learning experience.

Recalls in section 3.2, the objectives are defined as following:

- Allow user to get familiar with financial market
- Allow user to apply different trading strategies
- Allow user to replay historical data
- Functionality to maintain user profile

The project focuses strongly on the second and third points. This is demonstrated through validation from previous section and provision of the actual simulator. On the first point, users of the simulator get familiar with financial market in some degree as a side effect of using the simulator, but the simulator itself does not teach the user proactively. On the last point, user profile was seen as a distraction of the true purpose of learning. It was realized in another way as the leader board under performance module.

Hence, more user feedbacks are still required to make the verdict.

Functionality

The table of requirements from section 4.2 lists all the proposed functionalities. Everything was fulfilled except two: DN-1.1 Multiple choice to exam the knowledge of the user; and DN-3.2 Functionality to download new content. The first one is obsolete because ASX provides sufficient materials on their web site which includes the multiple choices. The second one is unachievable because neither TRTH nor Sirca provides command based portal. The only way to access their service is from web browser.

Apart from this, much new functionality were implemented during the development. The most noticeable one would be the portfolio view with its underlying derivatives, which ended up being more complicated than perceived. For example, the nature of option was known but not programmed before. Its implementation involves attention on multiple attributes such as expire date, expire price and volatility. These three are crucial parameters in Black Schole formula, which was used actively for calculating the price of any user generated option as accurately as possible. This avoided the association with tremendous raw option data from TRTH.

Code Structure

As mentioned in section 5, a variation version of MVC was applied during implementation. There are some obvious advantages of such adaptation such as reduction of complexity by merging View and Control. However, the implementation of Model package was not well handled. There was no central class used for logic processing. Instead, most of the work is done within the View package. This makes the classes under Model package loosely correlated and the code harder to maintain. Such structure should be revised. Apart from this, the code maintains a clean style. Classes are well organized under each package. Redundant code has also been minimized. Codes that may be used on multiple locations are made usable.

Performance

Even though a simulator running on historical data is not time critical, but was prioritized highly during development. Despite the flaw in the overall code structure which may cause some overheads, every bit of algorithms has been carefully devised and refined for better performance and memory management, so that efficiency is guaranteed. Such practise is expected to carry on in future projects.

Schedule

The project took 2 semesters to complete. There was a plan proposed was strictly followed, except the implementation stage took about 10 weeks, 2 weeks more than planned. At least 150 hours were spent on the development and more than 5000 lines of code were written. There was some time wasted on the implement basing on a false assumption of data format. There was also time spend on implementing the part conceived but not covered in section 4 of the report, namely the Home view and Performance module. A demo version was made by the time of the second presentation. The core of the system was done, but it would be better if I had the full version at disposal. Nevertheless, the progress was well documented and the deadline was met.

Feedbacks

All feedbacks are taken seriously and I always act responsibly to them. I worked closely with my supervisor. The core of the system was advised to be built first, and by doing the system was delivered on time. Feasibility of every detail was also refined, especially on the flexibility of input data. The code was made sure to be flexible enough to handle data of any format, as long as it contains the essential columns. In addition, we discussed the usability of the software. The derivative attributes in strategy view were suggested to be locked in when a strategy is chosen, so the default value was altered. The progression control panel under simulation view was not very intuitive, so a GUI was altered to a new design. Furthermore, the assessor proposed that the simulator should be able to display all the news of one stock. It was done accordingly. However, the other proposal which was to create a wizard for allocating provided files to their correct locations was not implemented.

A finance professor who has many years of experience in the industry was invited to test the system. However, he was unavailable until December this year. We look forward for him to give more feedback on the system.

Future Improvements

Out of all the possible improvements, the most essential one goes to Sirca news. Sirca needs to brush up its news service. It has poor functionality comparing to TRTH. It has no recording of previous request, it is not fully compatible with regular expression and it has ambiguous output format, where there are too many redundant entries and the news body does not go into one single row. The format should be as simple as having 5 columns: time, date, headline, news body and related stock. Instead, there are 20 columns, where most are very confusing to the user. And it does not give user the option to remove these columns. Taking a step back, even the data in the key column such as headline are often incomprehensible as it is out of context or of another language. Having this fixed would give a far more informative simulator. Thinking from another angle, raw data can only be gathered from Sirca at the moment. Relying on one source is not so wise. Future project should seek alternatives.

As mentioned above, one of the purposes that the simulator may be used is for education. This idea was not really enforced during development. It was built as a functional simulator rather than educational software. It takes extra measurements to build the second. Interactivity is one thing the current simulator is lacking. Apart from the occasional expiration of option dialog, the simulator rarely interacts with the user to tell them what is done right or wrong. Of course, the portfolio view is the quite informative, but it only provides a passive learning experience. Therefore, we could look beyond the traditional way of teaching, such as by introducing triggers to pop up dialog in news to advice user on this occasion, or have more informative news headlines. Utilize multi-threading a step further so every window is automatically updated as simulation goes on. It is hoped these ideas could provide directions on future extension on this project.

In addition, a wizard would be helpful for importing raw data to the configuration package. This would integrate the simulator with the ADAGE model more cohesively, hence improve the usability.

Therefore, there are still rooms for improvements. More users should test the system and all their feedback should be welcomed.

7. Bibliography

Paper and Textbook

Chapter 20, Chapter 21, Bodie, Z., A. Kane and A. Marcus, 2010, *Investments*, 9th edition, Irwin McGraw-Hill

Black, Fischer; Myron Scholes (1973). "The Pricing of Options and Corporate Liabilities". *Journal of Political Economy*. (Black and Scholes' original paper.)

Websites

Ad-hoc DAta Grids Environments, http://cgi.cse.unsw.edu.au/~soc/adage/ (visited 27, 10, 2013)

Adobe Photoshop, http://www.adobe.com/au/products/photoshop.html (visited 27, 10, 2013)

Java, http://www.oracle.com/technetwork/java/index.html (visited 27, 10, 2013)

Model-View-Controller, http://msdn.microsoft.com/en-us/library/ff649643.aspx (visited 27, 10, 2013)

Australian Securities Exchange, http://www.asx.com.au/ (visited 27, 10, 2013)

Amchart Home page, http://www.amcharts.com/ (visited 22, 10, 2013)

Thomson Reuters Tick History, https://tickhistory.thomsonreuters.com/TickHistory/login.jsp (visited 22, 10, 2013)

Sirca Member Service, https://ausequities.sirca.org.au/member_services/signin (visited 22, 10, 2013)

"Black-Scholes" in Multiple Languages, http://www.espenhaug.com/black_scholes.html (visited 27, 10, 2013)

Appendix

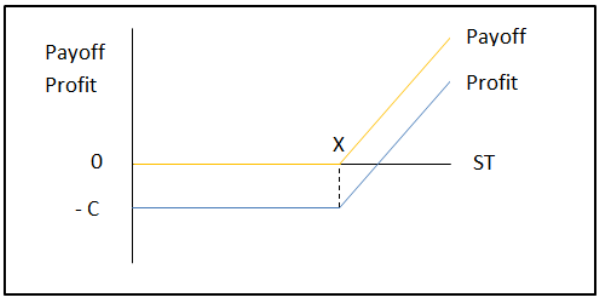
AP-1 Complex Trading Strategy continuous

Straddle
Definition: buying both a call and a put on a stock, each with the same exercise price and same expiration date.

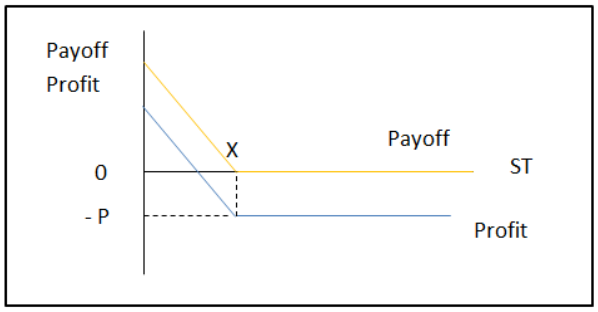
Payoff	$S_T < X$	$S_T >= X$
call	0	$S_T - X$
+ put	$X - S_T$	0
= TOTAL	$X - S_T$	$S_T - X$

Profit = Payoff – P

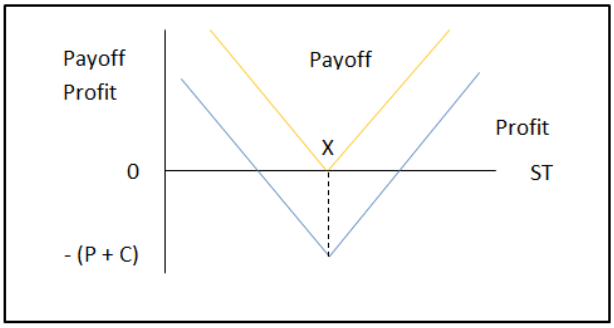
Call



Put



Total:



Strategy:
When believe a stock will move a lot in price but are uncertain about the direction of the move.

Example: Suppose you have 1000 BHP stock with S0 = $50 and you heard the firm is going to have a huge announcement with uncertain impact. You could long 10 call contracts for $150 each and 10 put contract for $200 each, both with the same strike price X = $50. That's $1.5 per stock for the call and $2 for the put. After the announcement, you earn a positive profit if the price either raises above 50+ 1.5+2 = $53.5 or drops below 50 - 1.4-2 = $46.5.

Spreads

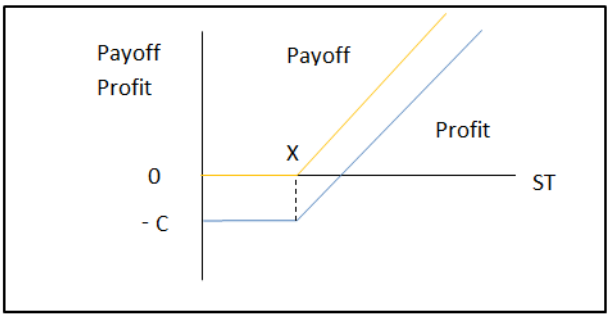
Definition: Combination of two or more call options (or two or more puts) on the same stock with differing exercise prices or times to maturity. Some are bought, whereas others are sold or written.
Money spread: purchase one option and simultaneous sale of another with a different exercise price.
Time spread: sale and purchase of option with differing expiration dates.

Money spread Payoff

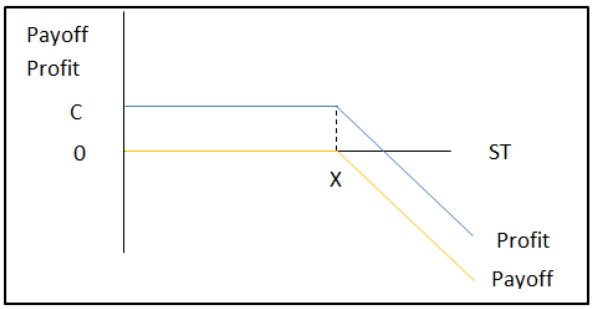
Payoff	$S_T <= X1$	$X1 < S_T <= X2$	$S_T >= X2$
Purchased call, exercise price = X1	0	S_T - X1	S_T – X1
Written call, exercise price = X2	-0	-0	-(S_T – X2)
= TOTAL	0	S_T – X1	X2 – X1

Profit = Payoff –C1 +C2

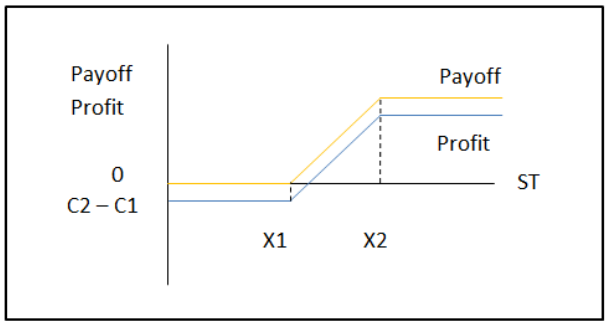
Long Call 1



Short Call 2



Total



Strategy:
When thinks one option is overpriced relative to another, or in other word, one option is cheaper than it should be.

Example:

Take an arbitrary stock XYZ currently priced at $100. Furthermore, assume it is a standard option, meaning every option contract controls 100 shares.

Assume that for the next month, a call option with a strike price of $100 costs $3 per share or $300 per contract, while a call option with a strike price of $115 is selling at $1 per share, or $100 per contract.

A trader can then buy a long position on the $100 strike price option for $300 and sell a short position on the $115 option (aka write a $115 call) for $100. The net debit for this trade then is *$300 - 100* = $200.

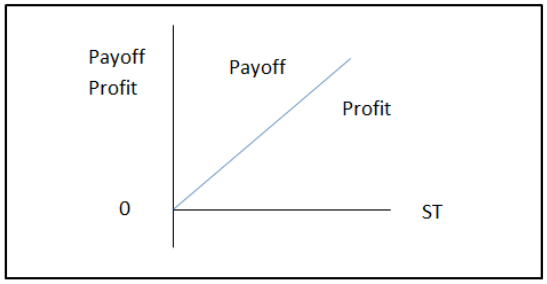
This trade results in a profitable trade if the stock closes on expiry above 102. If the stock's closing price on expiry is $110, the $100 call option will end at $10 a share, or $1000 per contract, while the $115 call option expires worthless. Hence a total profit of *$1000 - 200* = $800.

Collar
Definition: a strategy which brackets the value of a portfolio between two bounds.

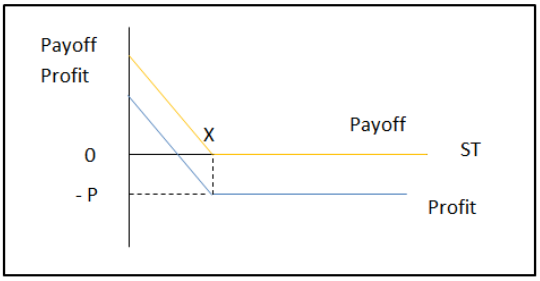
Payoff	S_T <= X1	X1 < S_T <= X2	S_T >= X2
Stock	ST	ST	ST
Purchased put, exercise price = X1	X1 - ST	0	0
Written call, exercise price = X2	-0	-0	-(S_T – X2)
= TOTAL	X1	ST	X2

Profit = Payoff – P + C

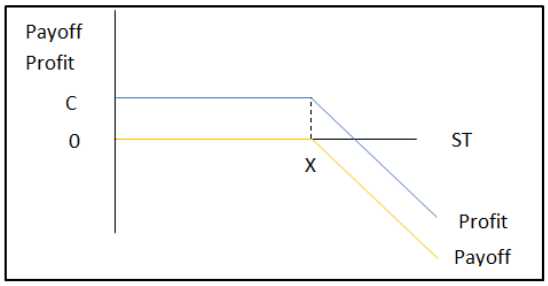
Stock



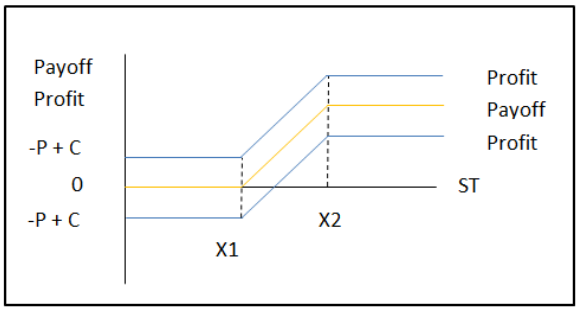
Long Put



Short Call



Total



*There are two profit lines because the value of -P+C is uncertain.

Strategy: Have a target goal in mind but is unwilling to risk loss beyond a certain level.

Example:

You current have $200,000 and you are unwilling to loss more than $20,000. You could purchase 2,000 shares selling at $100. Purchase 2,000 put option with X = $90. Write 2,000 call option at X = $110. This gives you a good chance to realize capital gain without losing more than $20,000.

More strategies will be added into the system depends on the time and scope of the project.

AP-2 Black Schole Formula

The Black–Scholes model is a mathematical model of a financial market containing certain derivative investment instruments. From the model, one can deduce the Black–Scholes formula, which gives a theoretical estimate of the price of European-style options.

The value of a call option for a non-dividend-paying underlying stock in terms of the Black–Scholes parameters is:

$$
\begin{aligned}
C(S,t) &= N(d_1)S - N(d_2)Ke^{-r(T-t)} \\
d_1 &= \frac{1}{\sigma\sqrt{T-t}}\left[\ln\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)\right] \\
d_2 &= \frac{1}{\sigma\sqrt{T-t}}\left[\ln\left(\frac{S}{K}\right) + \left(r - \frac{\sigma^2}{2}\right)(T-t)\right] \\
&= d_1 - \sigma\sqrt{T-t}
\end{aligned}
$$

The price of a corresponding put option based on put-call parity is:

$$
\begin{aligned}
P(S,t) &= Ke^{-r(T-t)} - S + C(S,t) \\
&= N(-d_2)Ke^{-r(T-t)} - N(-d_1)S
\end{aligned}
$$

For both, as above:

- $N(\cdot)$ is the cumulative distribution function of the standard normal distribution
- $T-t$ is the time to maturity
- S is the spot price of the underlying asset
- K is the strike price
- r is the risk free rate (annual rate, expressed in terms of continuous compounding)
- σ is the volatility of returns of the underlying asset

Volatility is not given, so we firstly gathered the first 4 parameters which are given. Risk free rate is calculated based on time to maturity and yield of US Treasury bond. Together with the price of the existing option, we can deduce the implied volatility using recursion version of binary search. That is, keep changing the value of volatility until the calculated price is extremely close to the real price. Then we applied this implied volatility to future option of any expiration price and time to maturity to find its price. This method has been tested with a less than 5% error in option price.