Implementation of FIX Engine and Order Management Systems using ASP.NET C#

Shweta Swarnkar
Department of Computer Science, Montclair State University
Montclair, NJ 07043, USA

and

John Jenq
Department of Computer Science, Montclair State University
Montclair, NJ 07043, USA

ABSTRACT

Financial Information Exchange ("FIX") protocol, a public domain protocol, is a series of messaging specifications for the electronic communication of trade-related messages. FIX allows independently developed systems to communicate seamlessly and without licensing issues. It has been developed through the collaboration of banks, exchanges, broker-dealers, industry utilities and associations, institutional investors, and information technology providers from around the world. The vision of market participants is to standardize a common, global language for the automated trading of financial instruments. The objective of this project is to develop and implement a prototype FIX Engine and an Order Management System using Microsoft ASP.NET platform using C# programming language.

Keywords: FIX engine, Online Trading, ASP.NET

1.INTRODUCTION

Online Trading [1,2,3,4] is the buying and selling of stocks or other securities through the Internet. Instead of paying a broker to do the transaction for you, you type them in yourself. Brokerage fees are much lower, and transactions are completed more promptly.

Financial Information Exchange ("FIX") protocol [5, 6, 7], is a series of messaging specifications for the electronic communication of trade-related messages. FIX allows independently developed systems to communicate seamlessly and without licensing issues because FIX protocol is a public domain protocol. The FIX protocol has been developed through the collaboration of banks, exchanges, broker-dealers, industry utilities and associations, institutional investors, and information technology providers from around the world. These market participants share a vision of a common, global language for the automated trading of financial instruments.

A FIX engine is an application program which manages network connections, creates and parses communication messages both outgoing and incoming. A FIX engine also manages the session and application layers and is the single piece of software needed in order to FIX-enable trading or order management systems.

In the context of an online trading system, FIX engine serves as the interface to the outside world. Through connection of communication network, it connects you to the outside world and allows you to trade and exchange related information in a standard fashion. Thus, to FIX-enable an application refers to the integration of a FIX engine and connection to a routing network.

By maintaining electronic connectivity with trading counterparts, a FIX Engine sends and receives trading information and also monitors data integrity of the trading processes. The FIX engine actually is the component that handles low-level communications, and binds the protocol to your chosen programming language. A FIX Engine is a solution for companies who

plan to use the FIX protocol for electronic financial information exchange of financial data.

The Financial Interface exchange effort was initiated in 1992 by a group of institutions and brokers interested in streamlining their trading processes. FIX specifications are developed and managed by the members of organization known as "FIX Protocol Limited (FPL)". FPL is formed by major industry players.

The publication of the FIX standard in 1992 was followed by a number of products which can be categorized as first generation FIX engines. They were straightforward implementations of the FIX Protocol with their roots in traditional financial/accounting systems. They were typically built around a relational data base or on top of an application server. The most successful were Coppelia from Javelin [8], Financial Fusion from Sybase and Trinitech [9]. As FIX becomes more popular, the performance of these first generation solutions started to become an issue.

Second generation engines ran considerably faster than earlier versions, with Cameron FIX arguably being the first to market. Another significant second generation FIX engine which was the open source implementation, Quick FIX [10]. These FIX engines had raised the bar on performance but were still little more than straightforward implementations of the protocol.

The third generation FIX engine naturally implements the FIX protocol but it also serves as a platform for the processing of FIX messages. The FIX engine itself can be the most efficient and appropriate places to process those messages. Customers can configure in their own business logic for manipulating the FIX messages. FIX is now used by variety of firms and vendors. The latest FIX version is 5.0. Most of the current production versions are 4.1 to 4.4. FIX has become the de facto messaging standard for pre-trade and trade communication in the global equity markets, and is expanding into the post-trade space to support Straight through Processing, as well as continuing to expand into foreign exchange, fixed income and derivatives markets.

The FIX Engine can be combined with an Order management system that can store orders and operations in a database. In this project, the FIX Engine and an Order Management System for the Broker and Exchange is developed for online trading of stocks. This FIX engine is integrated with an Order Management System (OMS) for broker and exchange in order to generate and/or process FIX messages.

We outline of this report is as the following. Section two describes the system architecture and its implementation. Three-tier design was discussed. Section three list the system features. Section four concludes this report.

2. DESIGN AND SYSTEM IMPLEMENTATION

Three-tier design is adapted for our design of exchange server and OMS. These two servers exchange trading information through FIX engine. Any three-tier applications can be understood in terms of three different functional components: data management, application logic, and presentation. Different components are responsible for different purposes. The data management component is responsible for storing and managing all the persistent data used in the application. The presentation component is responsible for presenting the application data and functionality to the user and collecting the user's input to the application. The application logic component is the heaviest among all three. It implements the business logic of the whole application as well as addressing multiple functional and nonfunctional requirements by providing various services like caching, distributing computing, distributing transactions, data transformation, authentication, and many others.

As mentioned above, a FIX engine is integrated and link to an Order Management System (OMS) for broker and one FIX engine with exchange server. The Order management system which integrated with the FIX Engine will store orders and perform operations to the backend database. The Order Management System which links to the FIX Engine is used to generate and process FIX messages.

The FIX Engine establishes communication between broker's order management system and an Exchange's order management system. Orders placed by Broker OMS will be converted into FIX messages and transmitted by Broker's FIX engine to Exchange's FIX engine. The Exchange's OMS will receive all the orders after validation by an exchange's FIX engine. The FIX engine will also maintain TCP/IP and FIX sessions by generating heartbeat messages at regular time intervals. A Heartbeat monitors the status of communication link between broker's FIX engine and exchange's FIX engine.

This project is developed in Microsoft ASP.NET framework[11,12] and use of C# [13] language for development of the code. The information is stored in the Microsoft SQL Server [15] database management system. The web application is based on three tier architecture of software engineering. There is a clear separation of these three layers of the architecture. Most part of this web application is populated dynamically from the data stored in the database. This project uses a web server IIS server of Windows operating system.

3. SYSTEM FEATURES

3.1 Broker's Order management System

The home page of the Broker's Order Management system consists of a navigation menu that contains Home, Trade, My Account, Research & Ideas, Contact Us, About Us links. On clicking these menu links, it will open detail pages. User can access "Trade", "My Account", "Message Monitor" & "Message Archive" pages only after login to the application. Home page will contain "Login" and "Open An Account" submenu for user login. It will also have search buttons details by "Company Code" and "Company Name".

To open an account in the application user has to first create his user name and password. After submitting this information, a new page will open where user has to fill a form of his personal information like first name, last name, address, phone number, e-mail and his bank information. After submitting this form, user will get a message that account has been created successfully.

There is an Order Book for each user. User will be able to navigate to this page from Order Book link under the top navigation link Trade. This page will display order history of a user. This page will display company code of the company of which user has placed the order. It will also display quantity, order price, order date/time, side, open quantity, executed quantity and Time In force of the order. User can also modify or cancel his order before execution, if order status is active. **Figure 3.1** is a screen shot of an order book.

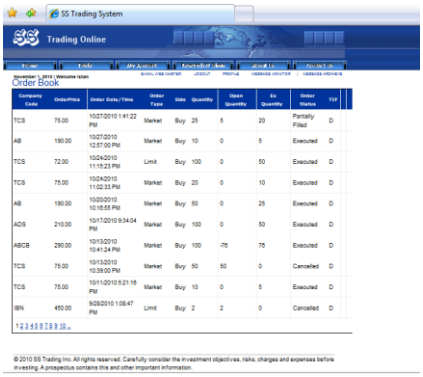

Figure 3.1 An order book

User will be able to navigate to this page from Trade Book link under the top navigation link Trade. This page will display all the executed orders. This page will display company code, company name, quantity, tax, brokerage, total amount and total trade date/time.

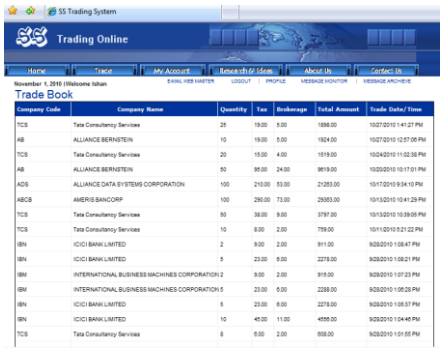

Figure 3.2 A trade book

This Message Monitor page as shown in **Figure 3.3** will display after login in the broker's order management system. This page displays the entire message communication between Broker's and Exchange's FIX engines.

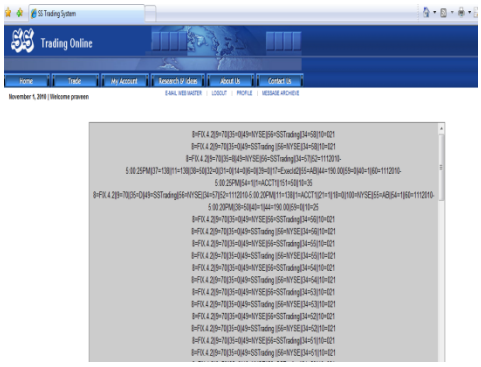

Figure 3.3 FIX message monitor page

For illustration purpose, Figure 3.4 demonstrate a message which buying 5000 shares of IBM with $110.75 each share.

Each part is a set of fields. Body contains actual application message. Checksum is used to avoid transmission errors.tag 35 is very important tag that tells what type of information a message is carrying. Tag 8 indicates the version of fix protocol used to create the message. 35 = D new order single means the sender is sending a new order to trade the single security.

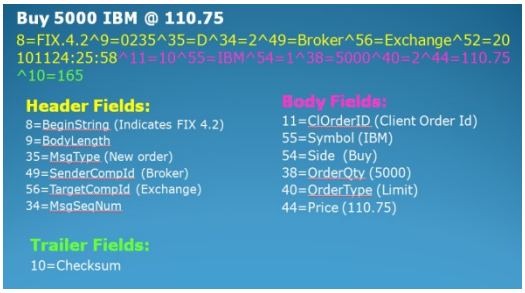

Figure 3.4 Example of purchase of IBM stock

There is a broker's Message Archive page displays messages generated on previous dates between FIX engines.

3.2 Exchange's Order Management System

We list some interesting pages for Exchange OMS.

The Order Monitor page will display orders placed from the Broker's side. This page will display Company Name, Order Id, Order Price, Order Date/Time, Order Type, Side Name, Ordered Qty, Last Share, Cumulative Qty, Leaves Qty, Order Status and TIF of the order. From this page Exchange can select Active or Inactive orders and will execute or cancel active order. This Page will also have submenus "Show Message" & "Logout" links.

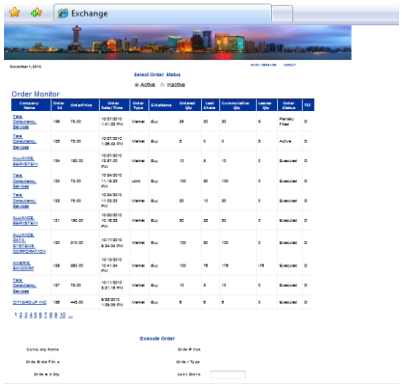

Figure 3.5: Order Monitor Page Screen

Similar to the broker System, the exchange system also can show message screen that displays after login in the exchange's order management system. This page displays the entire message communication between Broker's and Exchange's FIX engines. There are also Message archive for exchange server.

4. CONCLUSIONS and REMARKS

The benefits of a FIX engine include quick development and fast deployment for connectivity with new counterparties, easy and dynamic customization for maintaining and expanding the market reach. There are high flexibility and customizability in implementing the FIX Protocol and FIX variations. The efficiency of communication among financial

industry players is increased. As FIX engine utilizes FIX messages which reduces the time and complexity involved in connecting to multiple trading partners across different geographies. FIX engine will increase actual and potential completion between brokers for provision of trading services. Of course, it also reduces the explicit trading costs (e.g. brokerage commission and trading platform fees).

Over the years FIX has shown an extraordinary ability to evolve and adapt to the demands of its users, which has been key to its success. Its popularity has driven the evolution of FIX engine software. A FIX engine greatly simplifies connecting to exchanges or brokers. FIX engine implements all important aspects of the FIX protocol. FIX is a high performance, low-latency solution for order routing and market data.

Better FIX engines help to make the FIX Protocol more attractive to the market place. The result is a symbiotic relationship between the standard, and implementations of that standard, which bodes well for both. FIX is gaining increased attention within the Exchanges community as over three quarters of all exchanges surveyed supported a FIX interface. Currently, FIX is being used for equities, fixed income and foreign exchange trading for both cash and derivative products. FIX Engine is an enterprise-strength solution for companies who plan to use the FIX protocol for electronic financial information exchange of financial data, and Straight through Processing (STP).

Our implementation of FIX engine and OMS is very user friendly. More useful functionalities can be added in. As it will be a database application and will follow standard technologies, the maintenance of this application will be very easy. This application has lots of scope for future enhancements. One interesting area in FIX application is how to automate FIX message testing.

5. REFERENCES

[1] Online trading from All Business http://www.allbusiness.com/glossaries/online-trading/4946592-1.html

[2] Online trading from Answer.com http://www.answers.com/topic/online-trading

[3] How Online trading works http://money.howstuffworks.com/personal-finance/financial-planning/online-trading.htm

[4] Trading and broker related terms http://currencies.suite101.com/article.cfm/forex_online_trading

[5] Financial Information Exchange Wikipedia http://en.wikipedia.org/wiki/Financial_Information_eXchange

[6] What is Fix http://fixprotocol.org/what-is-fix.shtml

[7] FIX protocol Specifications http://www.fixprotocol.org/specifications/

[8] Javelin Technologies http://www.javelin-tech.com/main/index.html

[9] Financial Fusion of Sybase http://www.sybase.com/solutions/financialservicessolutions/banking

[10] Quick Fix: http://www.quickfixengine.org/

[11] Official ASP.NET web site http://www.asp.net/

[12] Zak Ruvalcaba, Build Your Own ASP.NET Website Using C# & VB.NET, Site Point Pty.Ltd. 2009

[13] Paul Deitel *Visual C# 2008 How to Program*, Pearson Publishing

[14] Christinan Nagel, Bill Evjen, Jay Glynn, Morgan Skinner, Karli Watson, Allen Jones, *Professional C# 2005*, Wiley Publishing Inc.

[15] Microsoft, *SQL Server 2005 Books Online*, http://msdn.microsoft.com/en-us/library/ms130214(SQL.90).aspx