Chapter 2
How to Use the U-Mart System Network Edition

2.1 Introduction

Using the market simulator, chapter one introduced how to use one computer in order to deal with multiple machine agents preinstalled by default. The U-Mart system makes it possible for multiple agents to transact via a network. See the preface for the relation between the exchange and agents.

This chapter assumes that the trading terminal is used mainly by a human as an agent (trader) to make transactions in a computer-practice room in universities where multiple computers are connected to an available LAN. Conducting an actual trading experiment, this chapter introduces the flow from installing the market server that serves as an exchange to the consequential analysis.

2.2 Preparation

First, experimental equipment and the operational environment should be prepared. The U-Mart system is a platform-independent system since it has been developed in Java language. However, a server machine that serves to an exchange is required to handle a vast amount of orders from clients, which burdens the server heavily. Adequate performance and stable system environment are very essential.

2.2.1 Network Environment

Experiments through a network require a server machine that serves as the exchange and multiple client machines as an agent. The market server is an application program serving as an exchange on the server machine, and the trading terminals serve as the agent in the client machines. These machines can communicate via TCP/IP,

but recent computer practice rooms connected to the LAN are believed to meet the requirements.

2.2.2 Market Server Operation Environment

The market server is independent from the execution-platform since it has been developed in Java language. Operational checks have been conducted with Windows XP / 2000, Linux (i386) and MacOS X (PowerPC) on the U-Mart system contained in the attached CD-ROM. Especially convenient is that Java Runtime Environment (JRE) is also contained for the U-Mart systems for Windows and Linux, so the systems can be started with only the attached CD-ROM.[1] If your computer environment can operate the market simulator as related in Chapter 1, it likewise can operate the market server.

Confirm the specifications regarding the machine performance required. However, the machine load depends on the number of agents that are connecting, so a high-spec server machine is required to conduct such a large-scale experiment. Particularly, in connection with the operation on Java platform, a larger memory size is always better.

You can refer to the past results of transaction experiments with simultaneous connections that have been collected from 200 client computers simultaneously connected to the market server on a server machine (Pentium4 2.4GHz, 512MB RAM, Windows 2000). The number of simultaneous connections depends on the server machine's RAM and/or the network conditions.

Operation Specifications

Windows XP/2000

800MHz of Pentium3 processor, 512MB of system memory, and hard drive capacity of 256MB free space or more are required. The network connections are necessary for the server to communicate with client PCs via TCP/IP.

Linux

800MHz of Pentium3 processor, 512MB of system memory, and hard drive capacity of 256MB free space or more are required. X-Window (X11) environment is required to be installed for the GUI operation of the market server and trading terminal. The network connections are necessary for the server to communicate with client PCs via TCP/IP.

The operation check of the U-Mart system contained in the attached CD-ROM was conducted with FedoraCore 4/5 preinstalled by default.

[1] Java environment is preinstalled in MacOS X by default.

MacOS X

500MHz of PowerMac G4/G5 or more are required, and MacOS X version 10.3 or greater should be installed. The network connections are necessary for the server to communicate with client PCs via TCP/IP.

The operation check of the U-Mart system contained in the attached CD-ROM was conducted with MacOS X version 10.4.6/10.3.9 PowerPC edition.

2.2.3 Client PC Requirements

A PC that serves as each agent is also needed. As a client, the trading terminal is used. Java Runtime Environment is included in the U-Mart systems for Windows and Linux, so the trading terminal can be operated on the same environment as that of the market simulator. There is no problem if the client's PC is using Windows XP/2000, Linux, and MacOS X, and being connected to the network to communicate with the server via TCP/IP.

2.3 Market Server Installation

Windows

Copy the UMartSystem-Windows folder (Fig. 2.1) in the attached CD-ROM to the hard drive of the server machine. Since the market server operation generates a vast amount of experimental results (logs), confirm whether the hard drive has sufficient free space.

If the experimental results are not saved, the market server can also be operated from the CD-ROM.

Fig. 2.1 UMartSystem-Windows

Linux

Copy the UMartSystem-Linux.zip file contained in the attached CD-ROM to the hard drive in the server machine. Unzipping the copied file since this file is compressed in the ZIP format. The recent version of Linux provides GUI operation like Windows, which can decompress or copy such files by the GUI interface. Fig. 2.2 shows the example of the UMartSYstem-Linux folder after being decompressed by FedoraCore5.

Since the market server operation generates a vast amount of experimental results (logs), confirm whether the hard drive has sufficient free space.

Linux provides command operations to copy and decompress files. The following shows how to use the command prompt when the terminal is started to mount the CD-ROM and decompress the file after copying them under '/usr/local.'

```
% mount –t iso9660 /dev/cd/mnt/cd                (Mount the CD)
% cp /mnt/cd/UMartSystem-Linux.zip/usr/local/    (Copy the file)
% unzip UMartSystem-Linux.zip                    (Unzip the file)
```

MacOS X

Copy the UMartSystem-Mac.zip file contained in the attached CD-ROM to the hard drive of the server. Unzip the file after copying it since this file is compressed in the ZIP format. MacOS X can unzip the compressed file in the ZIP format by double

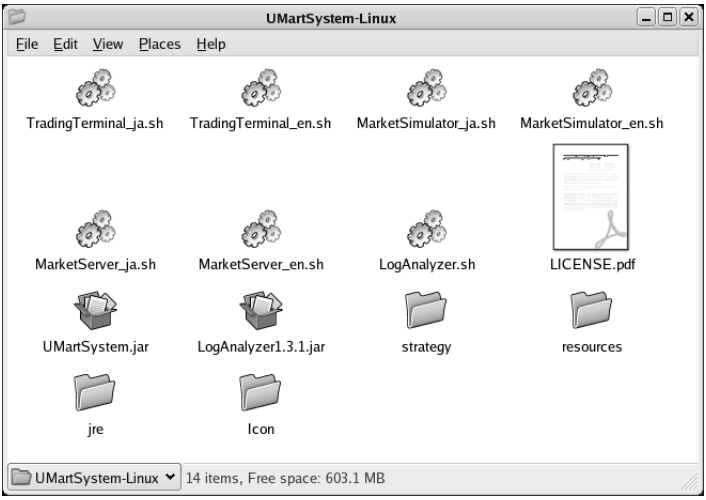


Fig. 2.2 UMartSystem-Linux

clicking on it. Fig. 2.3 shows an example of the UMartSystem-Mac folder after unzipping.

Since MacOS X is a UNIX-based OS, starting the terminal provides the command prompt operation as in the case of Linux.

Column: Client Installation

Since the U-Mart system clients use the trading terminal contained in the UMartSystem folder, prepare a U-MartSystem folder on the hard drive in accordance with respective environments just as the case of the market server. If the market simulator was already operated, the trading terminal should also be installed in the same environment.

If the experimental results (logs) do not have to be saved, the trading terminal can be operated on the CD-ROM.

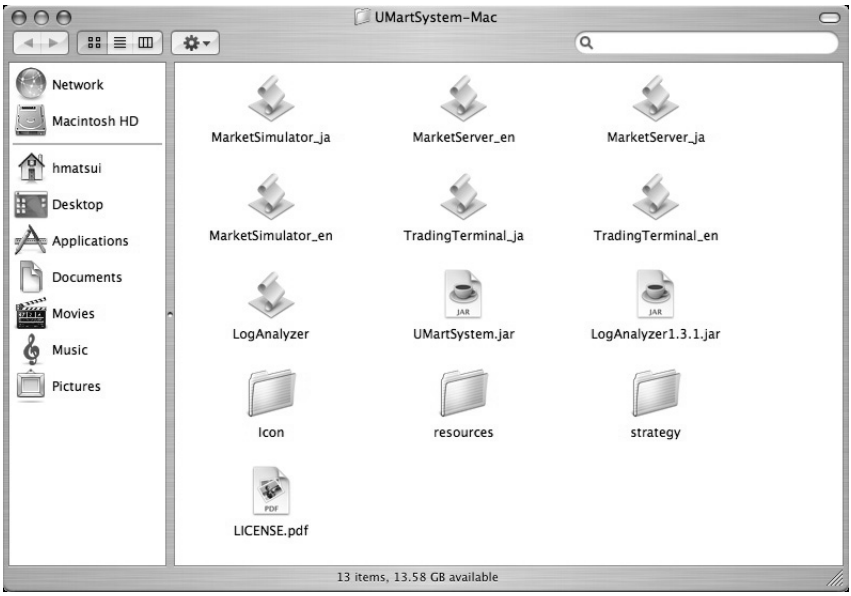


Fig. 2.3 UMartSystem-Mac

2.4 Starting Market Server

2.4.1 How to Start

Windows

The UMartSystem-Windows folder contains two market servers – the Japanese version (MarketSerer_ja.exe) and the English version (MarketServer_en.exe). If the server is unable to display Japanese characters when using an English version of Windows, switch to the English version of the U-Mart server.

Double click on the MarketServer_ja.exe icon to start the Japanese version. After a short time, the U-Mart project logo is displayed, and then a dialogue box appears in order to configure settings such as trading-start time and various preset default values.

Linux

Where the GUI environment is available, the Japanese version of market server can be started by double clicking on the MarketServer_ja.sh icon just like in Windows. Double click on the MarketServer_en.sh icon when starting the English version.

When the program fails to start even by double clicking, you need to start the program by using the command prompt. In such a case, enter the following command after moving to the UMartSystem-Linux folder from the terminal, under the X-Window environment.

```
% ./MarketServer_ja.sh      [return]
```

Enter the following command when starting the English version of market server.

```
% ./MarketServer_en.sh      [return]
```

After the market server is started, the U-Mart logo is displayed just like in Windows, and then a dialogue box appears in order to configure settings such as trading-start time and various preset default values.

MacOS X

As with Windows, double click on the MarketServer_ja.app contained in the UmartSystem-Mac folder. The Japanese version of market server is started. Double click on the 'MarketServer_en.sh' icon when starting the English version.

In addition, as is the case of Linux, you can start the program by command prompt since MacOS X is a UNIX-based OS.

Column: Starting the server with GUI on Windows/MacOS X

In Windows or MacOS X, the U-Mart sever can be started with a command. After starting Command Prompt for Windows or Terminal for MacOS X, enter the command to start the server. Refer README.pdf contained in the attached CD-ROM for the details of the commands to be entered.

2.4.2 Experimental Environment Settings

When starting the market server, the settings dialogue box as shown in Fig. 2.4 appears after the U-Mart project logo is displayed.

After setting each item, trading is begun by clicking on the 'OK' button. The detailed settings are configured with the configuration files (see paragraph 2.6). In this dialogue box, the file that registers experimental settings can be specified. Additionally, the basic experimental environment settings can be configured, such as trading experimental period, starting date, and the number of trading sessions per day.

Each setting item is described as follows.

Setting menu and File button This enables you to select the registered experimental environment (NickName). Clicking the 'File' button enables you to specify the file that registers the experimental settings. By default, TimeSeriesDefinitions.csv in the csv folder of the resources folder is specified.

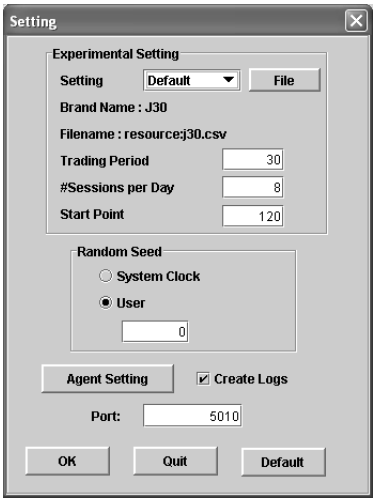


Fig. 2.4 Environment Settings window

Brand Name	Displays the name of the price time series, which is specified in the experimental configuration file.
Filename	Displays the filename of the price time series to be used.
Trading Period	Specifies the trading period of the experiment.
Sessions per Day	Specifies the number of trading sessions per day.
Start Point	Specifies the starting day of the price time series data. Enter the value of 120 or more.
Random Seed	Specifies the initial set value of random numbers that the system uses. Fixing the set value enables the user to obtain the experimental reproducibility.
Agent Setting button	Takes you to the setting window that configures human and machine agents used in the trading experiments (Fig. 2.5).
Create Logs	Experimental records are saved by checking this checkbox.
Port	Specifies the port number used in the network experiments. Number 5010 is specified by default.
OK button	The market server is started when this button is clicked (Fig. 2.6).
Quit button	The market server stops operating when this button is clicked.
Default button	Restore the default settings by clicking on this button.

When clicking on the 'OK' button without changing any settings, the market server is started with the default settings, and waits for the connection to the client's trading terminals.

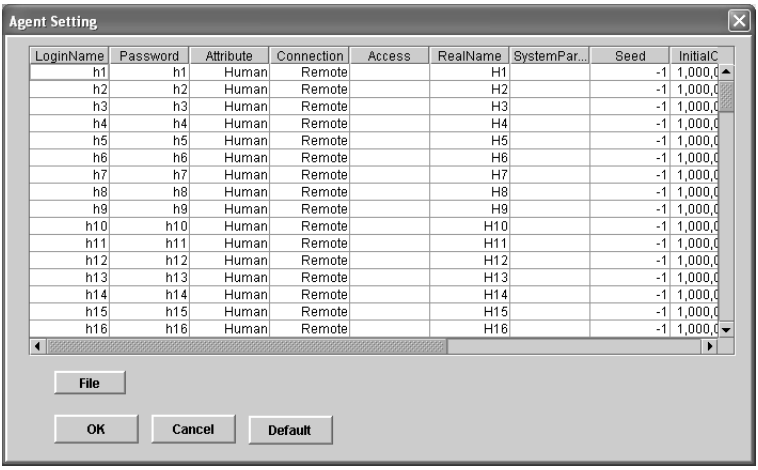


Fig. 2.5 Agent Settings window

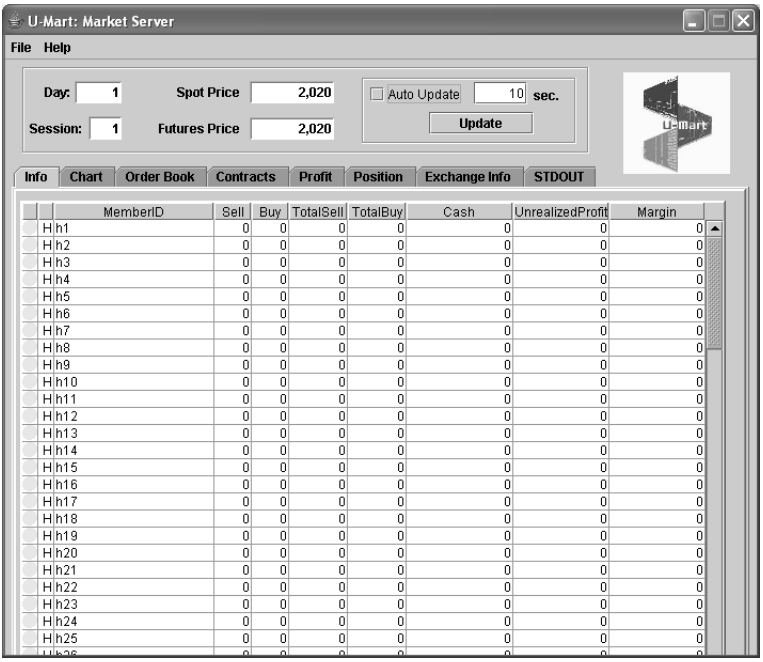


Fig. 2.6 Start window of market server

2.5 Trading Experiments

2.5.1 How to Connect to the Exchange Using a Trading Terminal

Clients use the trading terminals. The Japanese and English versions are prepared for the trading terminals.

With Windows, as is the case with starting the market server, double click on 'TradingTerminal_ja.exe' for the Japanese version, and double click on 'TradingTerminal_en.exe' for the English version. The trading terminals for OSs other than Windows can also be started in the same way as starting the market server.

After the U-Mart project logo is displayed, the dialogue box that confirms the destination appears (Fig. 2.7). In this dialogue box, enter the IP address (hostname) and the port number of the computer running the market server. The default port number is 5010; however, sometimes the port number is changed when starting the market server.

When a wrong IP address (hostname) is entered or the market server is not ready, the following message is displayed: "Can not connect the server. Try again." In this case, reenter the right IP address (hostname), or try to reconnect once the market server has been started.

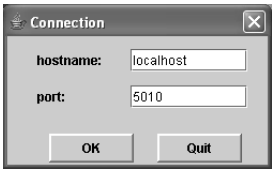


Fig. 2.7 Destination confirmation dialogue box of trading terminal

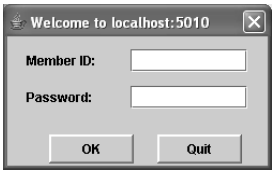


Fig. 2.8 Trading terminal authentication dialogue box

When the trading terminal succeeds in connecting to the market server, another dialogue box appears to enter the member ID and password (Fig. 2.8). The dialogue box header displays the IP address and port number of the market server.

Click on the 'OK' button after entering the member ID and password assigned to each client. The password entered is not displayed. When entering a wrong member ID or password, the message is display that prompts you to reenter the correct member ID or password again.

Column: Member ID and Password

Member ID specifies an agent that participates in the transaction. The upper limit of agents possible to participate is determined by the configuration files. Since the number of agents also includes machine agents, notice that the member IDs of human agents cancel and overwrite machine agent's member IDs.

The member ID of an agent that has gone bankrupt is not available during the ongoing experiment. It is recommended to prepare in advance member IDs for those agents gone bankrupt so that they can come back to make transactions again.

The trading terminal starts successfully when it connects to the market server and actual transactions can be started if the market is open (Fig. 2.9).

The manner of operation after connecting to the market server is the same as the market simulation. The difference is that the 'STDOUT' tab displays only your information. The market server contains pieces of order information collected from multiple trading terminals; however, it is impossible for the trading terminal to col-

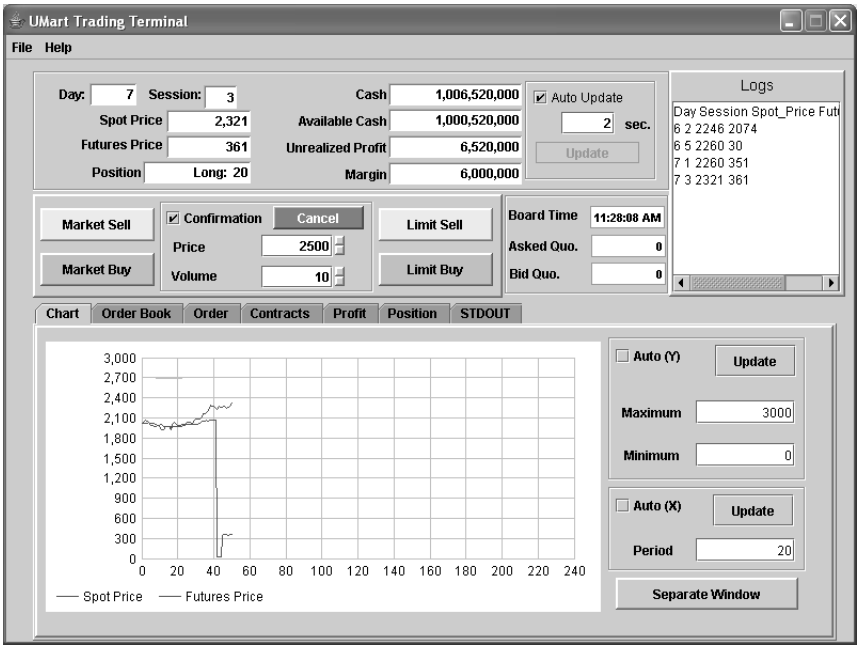


Fig. 2.9 Trading terminal view

lect trading information other than your own. The trading terminal can collect common information from the market and the information about the agent himself.[2]

The actual transactions are conducted in the same manner as with using the market simulator. In the market simulator, the participants in transactions are only the embedded machine agents and 'you.' However, where the network server is used, it means that there are other agents participating in the transactions, other than 'you.'

Furthermore, when the transaction is made via the network, the trading terminal might suddenly crash or become unresponsive due to various difficulties that occur on the network. In such a case, where the market server is operating correctly, you can continue to participating in the transactions resuming at the previous point and with your trading information by restarting the trading terminal and reconnecting to the server again. However, where the trading terminal does not response because of bankruptcy, it is impossible to connect to the market server with the same member ID.

[2] The information an agent can display in the trading terminal can be controlled by each agent at the market server configuration files. For example, the trading terminal can divide the agents into two kinds, the agent that can display *Itayose* information and the agent that cannot do so.

2.5.2 Market Server Functions

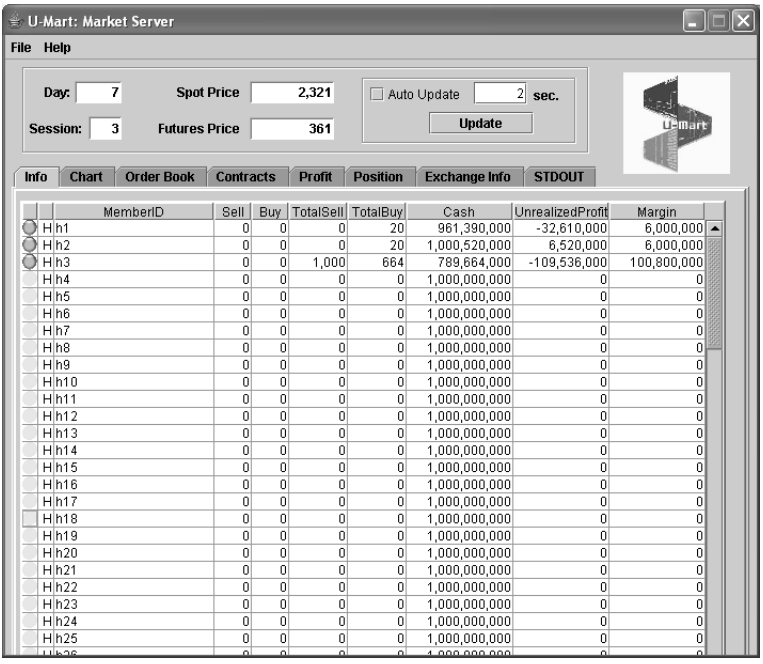
From the market server side, when an agent logs on to the server, the green indicator on Member ID of the "Info" tab lights up to show which member ID is logged on.(Fig. 2.10). If the market server is running, you can log into the server even before the transactions start.

The market server is able to control the progression of market transaction. Checking the 'Auto Update' checkbox allows the making transactions in the specified time interval (by seconds). The transactions can be progressed manually by clicking on the 'Update' button.

There are only two ways to start a transaction. One is to activate the automatic update function checking the 'Auto Update' checkbox with the time interval specified. The other one is to progress the transaction manually by clicking the 'Update' button. The transactions do not progress unless you click on the 'Update' button when it is controlled manually. On the other hand, when the automatic update function works, the transaction can be paused temporarily by unchecking the 'Auto Update.'

During transactions, as is the case with the market simulator, the market server can display a variety of information, such as charts, market conditions, such as the order book, contract information of each agent, and profit and position information.

The following information can be displayed by the market server.



MemberID	Sell	Buy	TotalSell	TotalBuy	Cash	UnrealizedProfit	Margin
H|h1	0	0	0	20	961,390,000	-32,610,000	6,000,000
H|h2	0	0	0	20	1,000,520,000	6,520,000	6,000,000
H|h3	0	0	1,000	664	789,664,000	-109,536,000	100,800,000
H|h4	0	0	0	0	1,000,000,000	0	0
H|h5	0	0	0	0	1,000,000,000	0	0
H|h6	0	0	0	0	1,000,000,000	0	0
H|h7	0	0	0	0	1,000,000,000	0	0
H|h8	0	0	0	0	1,000,000,000	0	0
H|h9	0	0	0	0	1,000,000,000	0	0
H|h10	0	0	0	0	1,000,000,000	0	0
H|h11	0	0	0	0	1,000,000,000	0	0
H|h12	0	0	0	0	1,000,000,000	0	0
H|h13	0	0	0	0	1,000,000,000	0	0
H|h14	0	0	0	0	1,000,000,000	0	0
H|h15	0	0	0	0	1,000,000,000	0	0
H|h16	0	0	0	0	1,000,000,000	0	0
H|h17	0	0	0	0	1,000,000,000	0	0
H|h18	0	0	0	0	1,000,000,000	0	0
H|h19	0	0	0	0	1,000,000,000	0	0
H|h20	0	0	0	0	1,000,000,000	0	0
H|h21	0	0	0	0	1,000,000,000	0	0
H|h22	0	0	0	0	1,000,000,000	0	0
H|h23	0	0	0	0	1,000,000,000	0	0
H|h24	0	0	0	0	1,000,000,000	0	0
H|h25	0	0	0	0	1,000,000,000	0	0

Fig. 2.10 Info-tab view of the market server

Info	Displays the information about the agent currently logged on. The green indicator lights up when an agent is connected, and the red indicator lights up when an agent has gone into bankruptcy. In addition, a list of buy and sell information and asset information of each agent is displayed.
Chart	Displays the transition of spot price and futures price.
Order Book	Displays the present board information and the board graph of previous pricing.
Contracts Information	Displays the contract information of each of agents.
Profit	Displays the graph of agent-specific profit transition.
Position	Displays the graph of agent-specific positions transition.
Exchange Info	Displays the IP address and port number of the market server.
STDOUT	Displays the information of each agent recorded in log files.

2.5.3 Transaction Experiment Termination

How to quit the market server

The transaction results are recorded as logs if the 'Create Logs' checkbox was checked when starting the market server.

As is the case of creating logs with the market simulator, logs are saved as the CSV format file in the "UMARTxxxxxx (the executed date and time on the millisecond time scale are entered in 'xxxxxx' part)" folder, which is created in the same hierarchy location of the market server.[3] With these log files, more detailed analysis can be conducted easily by utilizing Log Analyzer that will be mentioned later, or by loading the log files into spreadsheet software like Microsoft Excel.

In the 'Session' textbox of the market server, the mark '-' is entered at the close of each transaction period (Fig. 2.11).

When selecting 'Quit' from File menu on the menu bar, a confirmation message appears. You can quit the market server by clicking on the 'OK' button. However, notice that if you quit the market server before logging off the trading terminals, the trading terminals will not operate (the terminals get hung-up) which prevents the clients from performing such operations as saving transaction logs. It is advisable for trading terminals to be logged off before quitting the market server.

[3] In MacOS X, where starting the market server by double clicking from GUI, the log folder is created in the root directory (usually, Macintosh HD:). Notice that it causes an error depending on user's authority. When starting the market server from the command prompt, the log folder is created in the same hierarchy with the market server.

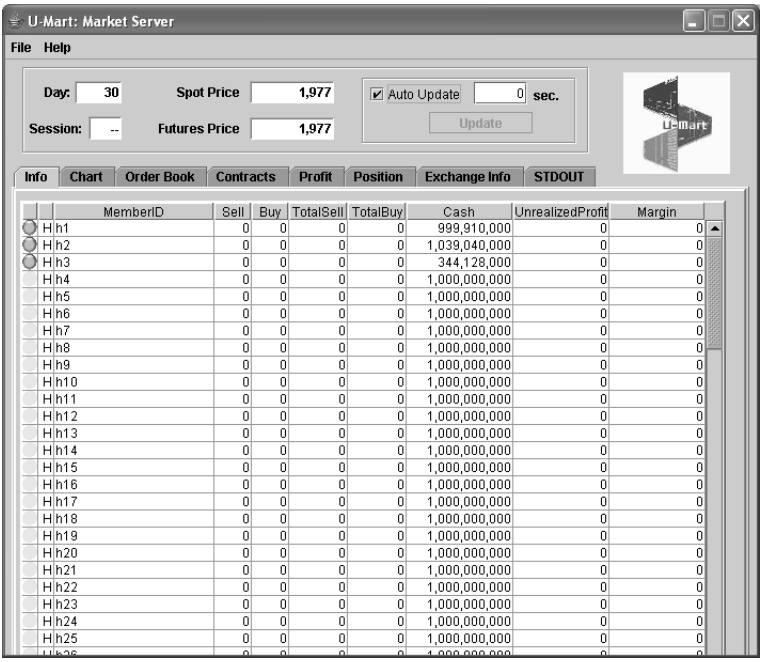


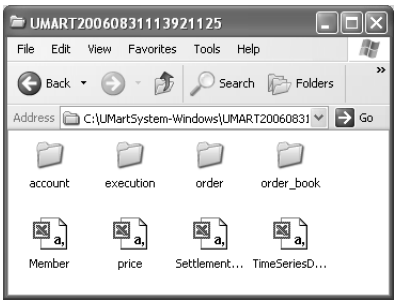
Fig. 2.11 Display of the market server at close of transaction

How to quit the trading terminal

After closing your transactions, you can quit directly or select to save the transaction records (logs, Fig. 2.12). To save your own transaction results, select 'Save' from File menu. The window comes up to save logs (Fig. 2.13).

Specifying the filename to save here, your transaction results are saved in four CSV files (***_order.csv, ***_posisiont.csv, ***_price.csv, ***_profit.csv). Fig. 2.14 shows the example of the files that are created with the filename 'Test' specified.

Fig. 2.12 Saved Market Server's logs



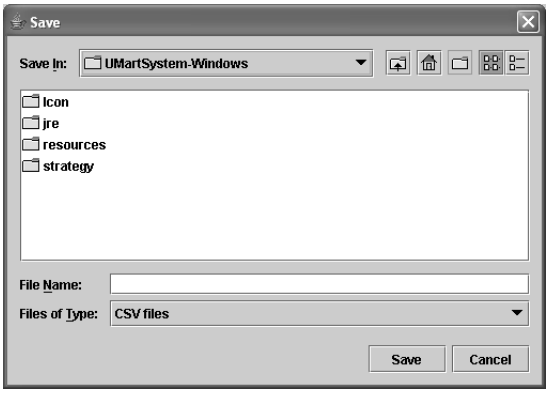


Fig. 2.13 Save window of your trading logs

Fig. 2.14 Saved CSV files

To quit the trading terminal, select 'Quit' from File menu. Then, the final results of all the agents who participated in this transaction are displayed (Fig. 2.15). You can confirm the final results by selecting 'Cancel' here, and your transaction results and others can be compared. Click on the 'OK' button to exit viewing the final results.

To quit the trading terminal, select the 'Quit' from the File menu once again. By clicking on the 'OK' button after the confirmation message appears, you can quit the trading terminal.

2.6 Configuration Files

The configuration files are contained in the csv folder (Fig. 2.16), under the 'resources' placed in the UmartSystem-Windows folder.[4] Since the market simulator

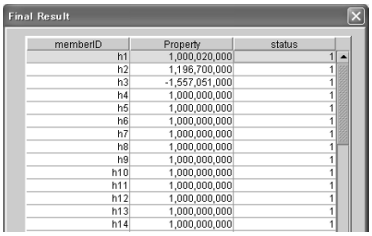


Fig. 2.15 Final results

[4] 'resources' is placed in the UMartSystem-Linux folder for Linux, and in the UMartSystem-Mac folder for MacOS X.

Fig. 2.16 csv folder in 'resources'

also uses the same configuration files, you can refer the following information for using not only the market server but also the market simulator.

Each configuration item of the file is as follows.

Institution.csv Player system configuration in the market simulator. See agent's settings part for each configuration item.

J30.csv Default time series data.

TimeSeriesDefinitions.csv Time series configuration that is used in the experiments and experimental condition settings.

MembersSA.csv Each setting of machine agents that are embedded in the market simulator by default.

MembersNet.csv Each setting of agents that participate in transaction experiments conducted by the market server.

SVMP.csv Connection settings between the exchange and agents. No need to change usually.

2.6.1 Time Series Configurations (j30.csv)

The U-Mart system is futures market, so the actual time series data should be provided in advance. As default time series data that the market server and the market simulator use, the past J30 stock index data provided by Mainichi Shimbun is prepared as j30.csv (Table 2.1).

Date '0' is entered. The date is output to the logs after the transaction experiments.

Table 2.1 j30.csv

Date	Session	SpotPrice	FuturePrice
0	0	2750	−1
0	0	2800	−1
⋮	⋮	⋮	⋮

Session	'0' is entered. The session number of each date is output to the logs after the transaction experiments.
SpotPrice	Spot price time series to be used when the transaction experiments are entered.
FuturePrice	Future price time series is entered. '-1' is entered when a price is not determined.

2.6.2 Configuration of Experimental Conditions (TimeSeriesDefinitions.csv)

Preparing a number of experimental settings with time series data, trading period, trading volume per a day (the number of sessions) and a start day of time series specified, each set of experimental condition can be named (with NickName) as experimental settings. This will allow the user to select experimental conditions when a transaction experiment is to be conducted (Table 2.2).

NickNames, as experimental settings, are displayed and can be selected from the 'Setting' form of the initial configuration dialogue when the market server is started. When it comes to the trading period, trading volume per a day (the number of sessions), and starting day of time series, these can be changed after selecting the desired NickName.

NickName	The experimental name given to experimental settings specified. When conducting various experiments, each experiment can be distinguished according to the different "NickName."
Filename	The time series data that an experiment uses is specified. Where using j30.csv in the csv folder placed in the 'resources' folder, specify like the following: "resource:j30.csv." When preparing a time series data in the same hierarchy of the market server, the time series file path is set by specifying it as the following: "file:***.csv."
BrandName	The name of price time series data is specified. J30 does not have to be changed.
StartStep	The start day of price time series data is specified. Enter the value of 120 or more.

Table 2.2 TimeSeriesDefinitions.csv

NickName	Filename	BrandName	StartStep	MaxDate	NoOfSessionsPerDay
Default	resource:j30.csv	J30	120	30	8
EX1	resource:j30.csv	J30	340	20	4
⋮	⋮	⋮	⋮	⋮	⋮

MaxDate	The number of days for transaction experiments is specified.
NoOfSessionsPerDay	The trading volume per a day (the number of sessions) is specified.

2.6.3 Agent Settings (MembersSA.csv, MembersNet.csv, Institution.csv)

The following two csv files configure the agents that participate in a transaction: MembersSA.scv for the market simulator and MemebersNet.csv for the market server (Table 2.3). These two csv files have the similar formats but different parameter settings. Many configuration items have been prepared, however, the items required to conduct transaction experiments on the network are limited.

Additionally, you can confirm the agent settings loaded by clicking on the 'Agent Setting' button on the initial settings dialogue box. Furthermore, you can also specify another agent's settings file.

LoginName	The user ID of an agent logs in is specified.
Password	The password is specified.
Attribute	Agent attributes ('Human' for a human agent, 'Machine' for a machine agent) are specified.
Connection	Agent connection methods ('Remote' for a network connection, 'Local' for a local connection) are specified.
Access	Controls information presented to each agent ('CHART' for a chart, 'BOARD' for the order book, 'ORDER' for a current order, 'ORDER_RESULT' for contract information, 'PROFIT' for a profit, 'POSITION' for a position, 'STDOUT' for standard output). The information which an agent does not want to display is listed. Separate each name with colons (:) in order to specify several names. Since the item 'Access' is blank by default, all information is presented to agents.

Table 2.3 MembersNet.csv

LoginName	Password	Attribute	Connection	Access	RealName	System Parameters	···
h1	h1	Human	Remote		h1		···
h2	h2	Human	Remote		h2		···
⋮	⋮	⋮	⋮	⋮	⋮		
TrendStrategy	passwd2	Machine	Local		Strategy. TrendStrategy	WidthOf Price=10	···
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

RealName	Specifies a real agent name. The real agent name is specified in the case of network connection, and the program name (class file) of machine agent is specified in the case of local connection. For example, when specifying the 'TrandStrategy.class' file in the strategy folder, enter as 'strategy.TrandStrategy.'
SystemParameters	The initial setting values of each machine agent under the local connection are specified. When specifying multiple settings, separate each setting name with colons (:). See the appendix regarding the initial setting for the machine agents embedded.
Seed	The initial setting values are specified in order to randomly specify the machine agents used under a local connection.
InitialCash	Initial retention asset. One billion yen by default.
TradingUnit	One unit for an order. 1,000 units by default.
FeePerUnit	Exchange fee. 0 yen by default.
MarginRate	Deposit (margin) money. 30,000 yen per one unit by default.
MaxLoan	The upper threshold of exchange debt when going bankruptcy. 30 million yen by default.
Interest	The interest rate of exchange debt. 10 percent by default.

These settings can be configured according to each individual agent. For example, the transaction experiment among agents having different initial retention assets can be easily conducted. Additionally, it is possible to conduct a transaction experiment to confirm to differences between transactions, in which order book information is available and is not available.

2.7 Analyzing Log Files

After experiments using the U-Mart system, the logs that record the transaction details remain. The trading terminal leaves only the logs related to an agent's transaction. On the other hand, the logs recorded with the market server and market simulator contain all the transactions made by agents, so it takes considerable time to analyze those logs. For this reason, the program called 'Log Analyzer' is prepared in order to take extract particular agent's information.

2.7.1 Log Analyzer

Log Analyzer is an application program that graphically displays the transaction history of the entire market and the buy-and-sell history of the agents, This information can be loaded from the log folder created when quitting the U-Mart. Using Log Analyzer allows displaying the buy-and-sell results of each agent visually. Moreover, starting multiple Log Analyzers display the buy-and-sell results graphs among all

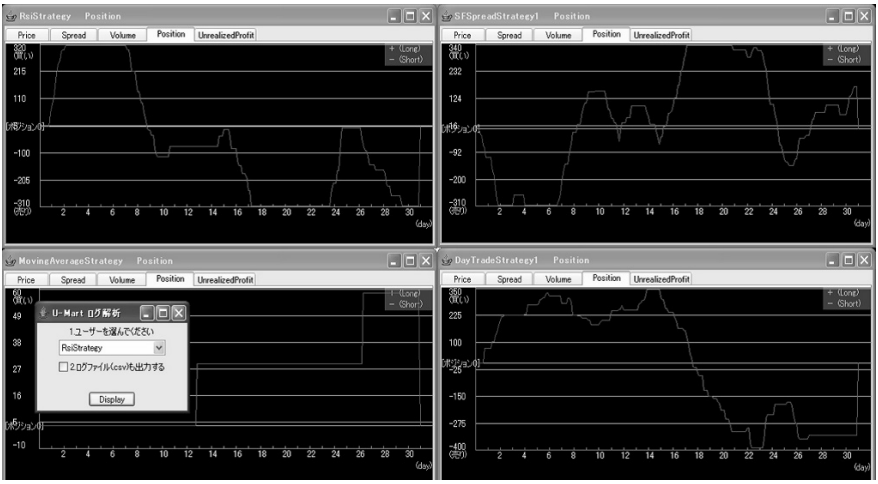


Fig. 2.17 Starting multiple Log Analyzers (Japanese Version)

or some of the agents in order to compare and analyze them (Fig. 2.17). Detailed numerical data can be outputted in csv files.

2.7.2 How to Use Log Analyzer

The use of Log Analyzer is common to the logs of trading terminal, market server, and market simulator.

1. Double click on 'LogAnalyzer.exe'.[5]
2. The folder 'UMARTxxxxxx' (the executed date and time on the millisecond time scale are entered in 'xxxxxx' part) is created in UMartSystem folder at the close of game, select this folder (Fig. 2.18). In the case of the market simulator, the 'Create Log' checkbox of the 'Setting' dialogue box is unchecked at the start of a session. Notice this checkbox should be checked before starting an experiment.
3. Right-click on 'Open' after selecting the appropriate log folder, then the 'U-Mart Log Analyzer' dialogue box appears (Fig. 2.19). Select the user that you want to display from the pull down menu of '1.' Check the checkbox of '2' if detailed buy-and-sell data of the selected agent is needed to be output in a CSV file (The file will be output with the filename " 'agent name'_log.csv"). Click on the 'Display' button after selecting the agent.
4. Each window is displayed graphically. Multiple agents can be displayed simultaneously by starting multiple Log Analyzers, and the windows that are displayed can be zoomed in and out by changing the window size.

[5] Double click on 'LogAnalyzer.sh' for Linux, and on 'LogAnalyzer.app' for MacOS X.

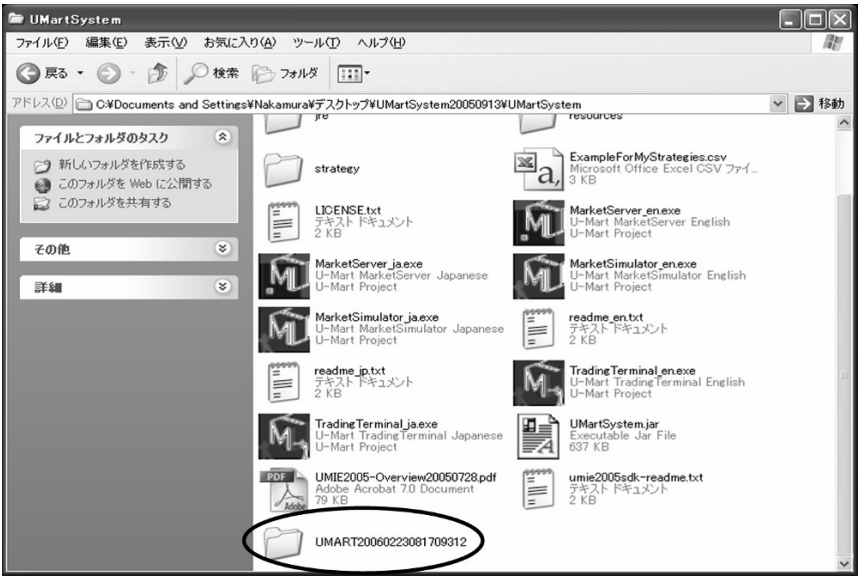


Fig. 2.18 Selecting the folder (Japanese Version)

Fig. 2.19 Start Dialogue Box (Japanese Version)

2.7.3 Display Contents

In Log Analyzer, the displayed contents can be switched by selecting the tabs. The 'Price,' 'Spread,' and 'Volume' tabs display the entire market results, so the common results are displayed by selecting any of these tabs. The 'Position' and 'Unrealized-Profit' tabs display the results of each agent.

1. Price tab (Fig. 2.20)

Displays the time series of spot and futures prices. The spot price is prepared by the server, whereas futures price is produced as a result of the transaction made by agents.

2. Spread tab (Fig. 2.21)

Displays the price difference (spread) between spot price and futures price with bar graphs. This tab is useful when evaluating the arbitrage transaction strategy to

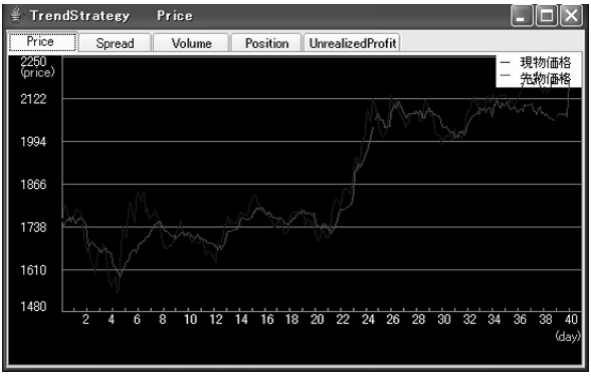


Fig. 2.20 Price tab (Japanese Version)

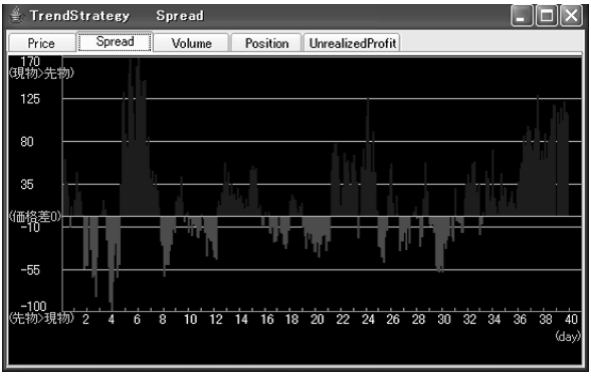


Fig. 2.21 Spread tab (Japanese Version)

determine to buy or sell with a price spread. It is possible to confirm whether you could enter a timely trade at a point of significant price discrepancy by using the arbitrage transaction strategy. In addition, you could evaluate whether you could close the positions when the price divergence dwindles in order to establish a profit.

3. Volume tab (Fig. 2.22)

Displays the contracted volume (trading turnover) of each session.

4. Position tab (Fig. 2.23)

Display user's positions (how many short position and long position the user has). In the trend strategy, it should be confirmed that the long position is increased as the price rises. With position transition, you can analyze whether you are alert by maintaining a good balance in positions, or taking risk preferences by being extreme in positions. Notice that the value on the graph returned to the location of '0' since

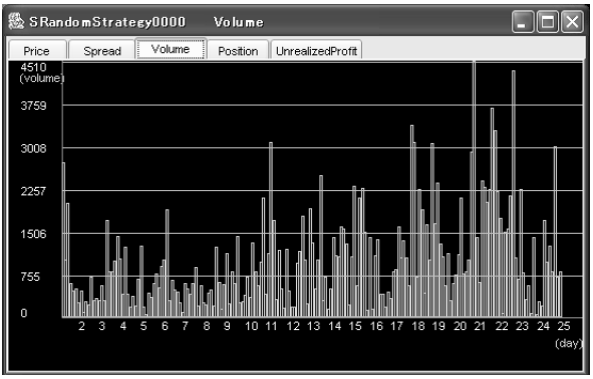


Fig. 2.22 Volume tab (Japanese Version)

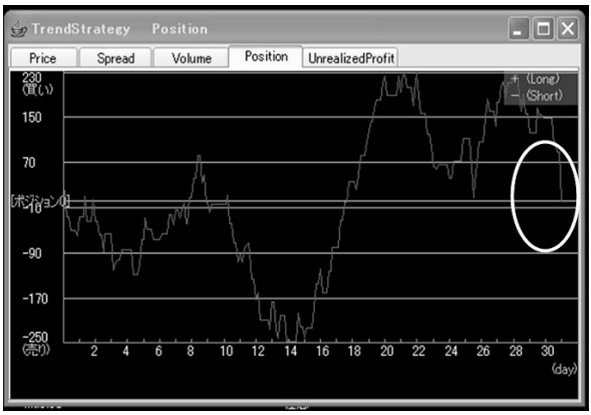


Fig. 2.23 Position tab (Japanese Version)

the positions that are carried over will be forced into settlement after the end of the final day transactions (the circle in Fig. 2.23).

5. Unrealized Profit tab (Fig. 2.24)

Displays the transition of user's unrealized profit and loss. In the U-Mart specifications, unrealized profit and loss are updated not after each session but each day, so this tab displays the transition of unrealized profit and loss day by day. The green bar in the graph indicates the final profit and loss after conducting Mark-to-Market on the positions carried over (the circle in Fig. 2.24). Where the transaction ends with a large amount of positions carried over, notice that the positions are forced into settlement at the final spot price and the results vary significantly.

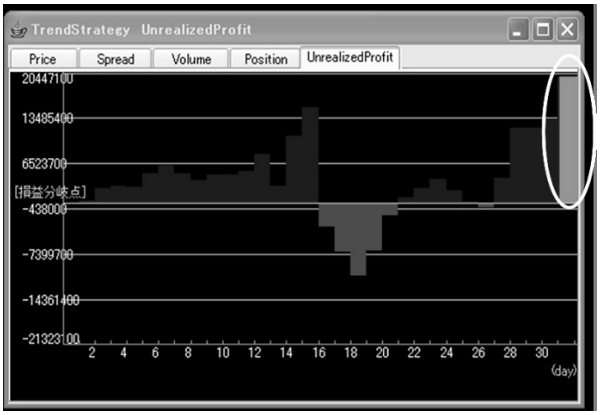


Fig. 2.24 Unrealized Profit tab (Japanese Version)