



FIX Price Server for TOCOM market data system Configuration and Management Guide

Date: 05 January 2004
Version: 1.0 (Release)

Orchid Technology K.K.
5F. Landic II Toranomom Bldg., 3-7-8 Toranomom, Minato-ku, Tokyo 105-0001 Japan

This document contains information proprietary to Orchid Technology K.K. and may not be reproduced, disclosed or used in whole or part without express permission of Orchid Technology K.K.

Orchid Technology K.K., by publishing this document, does not guarantee that any information contained herein is and will remain accurate or that use of the information will ensure correct and faultless operation of the relevant service or equipment. Orchid Technology K.K., its agents and employees shall not be held liable to or through any user of this document for any loss or damage whatsoever resulting from reliance on the information contained herein.

This document refers to the products and trademarks of manufacturers. Acknowledgement is made of all trademarks, registered trademarks and trading names that are referred to in the text.

© 2004 Orchid Technology K.K. All rights reserved.

Published by
Orchid Technology K.K.

Table of Contents

1. Preface.....	4
2. Prerequisites	5
3. Introduction.....	6
3.1 Purpose	6
3.2 Scope	6
3.3 Functional overview	6
4. Gateway operation	8
4.1 Startup	8
4.2 Termination	8
5. Configuration management.....	9
5.1 Configuration file format overview.....	9
5.2 Active service section	9
5.3 Global parameters context.....	9
5.4 FIX connection settings context.....	10
5.5 Exchange connectivity	11
5.5.1 Exchange common settings context	11
5.5.2 Relay Terminal settings context	12
6. Client side integration	13
6.1 Sequence number management.....	13
6.2 Implemented message types	13
6.3 Market Data Request implementation.....	13
6.4 Implemented instrument types	13
7. Reference documentation.....	14

1. Preface

This document provides detailed information about configuration and deployment aspects of FIX price server for TOCOM exchange market data interface. Following sides of the system are presented:

- Command-line parameters, startup/shutdown
- Configuration file syntax and parameter reference
- Client-side integration (FIX protocol compliance)
- Load balance and failover
- Exchange connectivity

2. Prerequisites

In order to address the topics, described in this document, reader should be familiar with the following:

- FIX Protocol version 4.2
- TOCOM market data system architecture, protocols and client side infrastructure maintenance

General knowledge of UNIX administration is highly desirable. Reference documentation is presented in reference section.

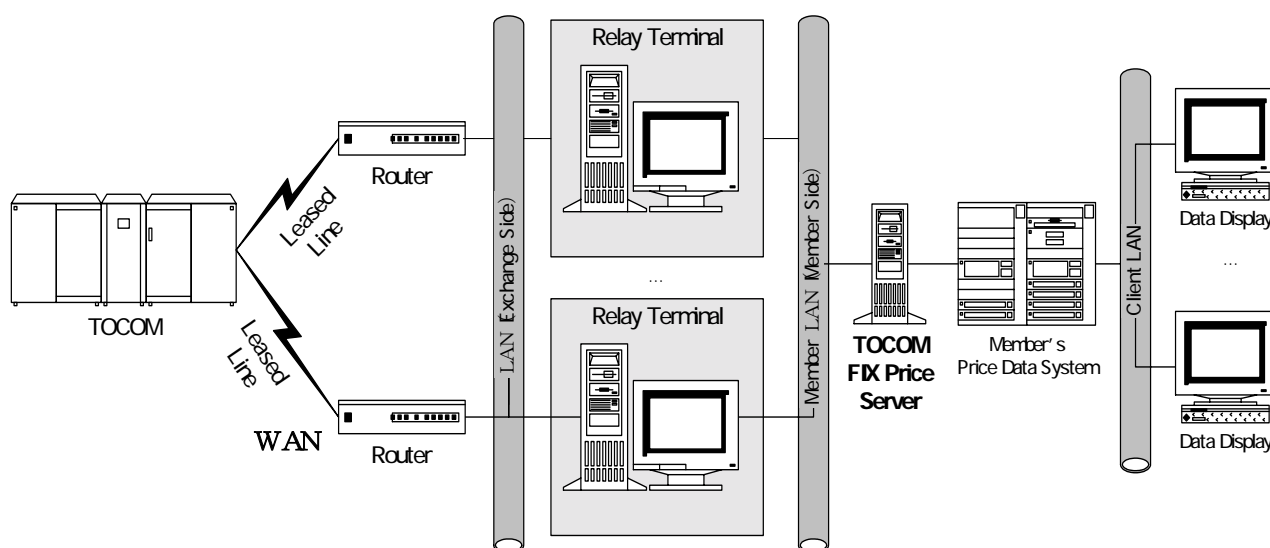
3. Introduction

3.1 Purpose

FIX price server for TOCOM market data system was developed to provide standard interface to proprietary commodities pricing data publishing engine developed by NTT Data. Interface is based on industry-standard FIX protocol version 4.2 deployed via transport level of TCP/IP stack.

3.2 Scope

Following diagram depicts how TOCOM FIX price server fits into overall trading infrastructure.



TOCOM market data system employs 3-tier client server model, where up to two Relay Terminals, installed at member site behave as intermediate proxies. Thus FIX price server can interface up to two Relay Terminals.

3.3 Functional overview

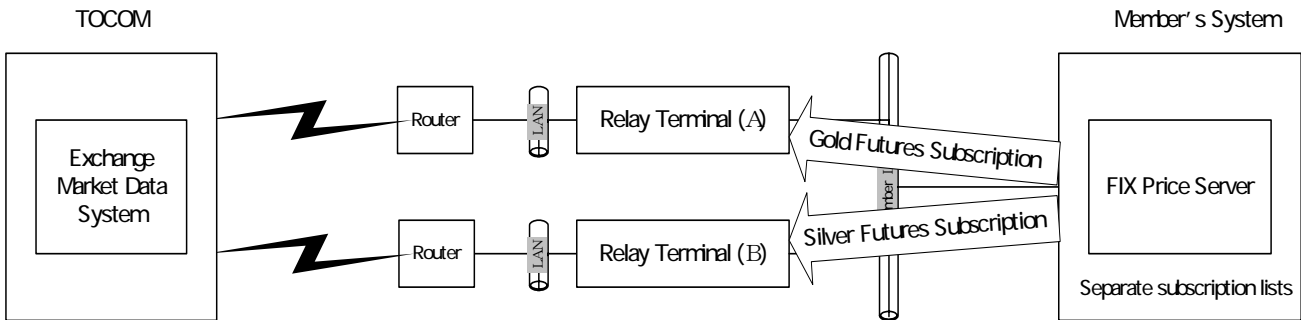
FIX price server provides market data publishing service by means of single FIX connection for business-level interactions with Member system. Each relay terminal connection (session) can be assigned its own list of instruments' subscriptions, providing load balancing and failover functionality. Once session is established, subscription assigned to this session is activated.

When real-time market data messages and other status notifications are received from TOCOM centre, they are processed and forwarded to Member System. In event of temporary member's system disruption, all incoming messages are discarded.

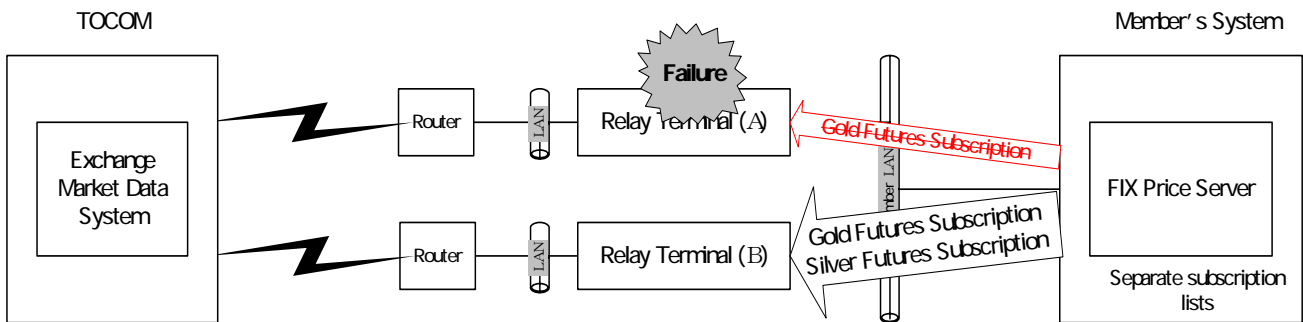
When FIX price server is configured to connect to two relay terminals, load balancing can be achieved by manipulating subscriptions lists.

If subscription list for one relay terminal connection differs from subscription list for the other connection, load balancing between two relay terminals is in place. The following diagram demonstrates configuration of FIX price server in which Gold Futures subscriptions and Silver

Futures subscriptions are load balanced between two relay terminals (A and B).



With two relay terminal connections configured, FIX price server can be configured to perform failover for subscription list from failed relay terminal connection on remaining active one. This is demonstrated on following diagram:



Once connection to failed relay terminal is recovered, FIX price server will automatically re-activate corresponding subscription list on recovered connection.

4. Gateway operation

Once started, the server can normally be left running around the clock. Special configuration parameters can be set to specify start and end of online time with TOCOM relay terminals.

4.1 Startup

The server's configuration is loaded from external configuration file and command line parameters are kept simple:

`./tocom_mdds_SunOS_5.8_sparc -conf <conf-file> <Active-Section>`, where

- **<conf-file>**: path to configuration file to read. Path can be either absolute or relative to the directory the application is started from.
- **<Active-Section>**: Primary function of this parameter is to identify top-level section in the configuration file that should be used for instance configuration. To simplify configuration management, the same file can hold the parameters for different instances or configurations to switch between (ex: Test vs. Live configuration). Additionally this parameter is used for naming of instance-specific files such as log, FIX-related persistent and pid files.

On startup, the server loads requested configuration and performs consistency checks. If an error is found in the configuration, it is reported to console/log file and application exits. Otherwise the server is detached from controlling terminal and regular processing starts with listening for FIX client connections and simultaneously opening requested relay terminal connections. NOTE: If the server has been started outside configured online hours, it will not initiate relay terminal connections until online start time.

4.2 Termination

Upon successful startup, pid file named **<Active-Section>.pid** is created in execution directory. It holds textual representation of instance's process identifier and can be used to communicate asynchronous interrupts to running instance. Similarly SIGINT and SIGTERM signals are used to terminate the instance. This is preferred way to stop the server.

5. Configuration management

5.1 Configuration file format overview

Configuration file is comprised of a number of sections, where a section can contain other sections as well as configuration parameters. Section is represented by its name followed by figure brackets identifying section content. Parameters always appear within a section (parameter's context) in form *Parameter_Name=Value*, where the *Value* is parameter-specific and can be identifier, string, or a list.

In order to avoid information duplication, parameters together with their context can be moved from inner level into outer level to extend their scope. Parameter search is performed starting from inner and expanding to outer contexts. Search is stopped with successful status if all required parameters are found; otherwise missed parameter is reported to the log file accompanied with last searched context information. All parameters are considered mandatory unless stated otherwise.

5.2 Active service section

As mentioned earlier, the second command-line parameter for starting an instance is a name of top-level section in configuration file that defines all parameters for particular instance. Once configuration tree is loaded, top-level context is searched for the name of that section. If not found, error is reported and application terminates, otherwise configuration parsing and consistency checking is performed. Instance-specific top-level section is referred to as *startup section*.

5.3 Global parameters context

This section describes global instance parameters. Startup section is searched to collect service-global parameters. Corresponding parameters reference is presented in following table:

Parameter Name	Value type	Description
Log_Path	String	Directory, where log files are to be collected. If relative path is specified, it is interpreted as starting from application startup directory.
Keep_Old_Data_Days	Number	Number of days the server will preserve old log and data files on hard disk.
Category_List	List of identifiers	Parameter controls logging depth. Every specified identifier designates single category of log messages to be output. Although the amount of information logged is configurable, it is suggested to collect maximum information unless it has negative effects such as performance or disk space issues. Following values are supported:
		<table border="1"> <tr> <td>FIX_SentData</td> <td>All FIX messages originated from the server.</td> </tr> </table>
FIX_SentData	All FIX messages originated from the server.	

Parameter Name	Value type	Description	
		FIX_ReceivedData	All FIX messages received from member system.
		FIX_API	Status information from the FIX engine.
		TOCOM_Data_Sent	Hexadecimal dump of the data, sent to Relay Terminal.
		TOCOM_Data_Received	Hexadecimal dump of the data, received from Relay Terminal.
		Debug	Extra information about low-level server operation.
		Info	Informative messages.
		Warning	System warnings.
		Fatal	Abnormal events.

5.4 FIX connection settings context

This section describes FIX connection-specific parameters. Startup section followed by surrounded contexts is searched for **FIX_Settings** section to collect required parameters. Corresponding parameters reference is presented in following table:

Parameter Name	Value type	Description
Cache_Size	Number	Total number of instruments the server can subscribe simultaneously.
FIX_Data_Path	String	Directory, where FIX engine data files are to be stored. If relative path is specified, it is interpreted as starting from application startup directory.
Local_Comp_ID	String	FIX-specific. Local company identifier to be used in FIX traffic.
Local_Sub_ID	String, Optional	FIX-specific
Local_Location_ID	String, Optional	FIX-specific
Port	Number	TCP port to be used to accept FIX connection from member's system.

Parameter Name	Value type	Description	
Max_Recover_Buffers	Number	Number of messages FIX engine will store in internal queue during sequence number mismatch recovery. Stored messages will minimize number of resend requests that follows the recovery.	
Logon_Timeout	Number	Time interval in seconds during which normal FIX logon procedure should be completed.	
Log_Raw_FIX_Data	Identifier	Flag specifying whether the server should dump raw data on FIX connection to a separate file:	
		yes	All messages going through FIX connection will be dumped as is to a file.
		no	No raw FIX messages will be dumped.

5.5 Exchange connectivity

Single server instance can handle up to two relay terminal connections. To describe the available connections, following scheme is used. Exchange common section is required to contain a parameter **Session_List**, which value should be a list of section identifiers, one per available relay terminal. These identifiers are primarily used to collect configuration information about particular connection, but also serve as marks in the log file to identify connection-specific information. For every section name listed, corresponding context (relay terminal description section) should be presented within startup section.

5.5.1 Exchange common settings context

This section describes TOCOM connection-specific parameters. Startup section followed by surrounded contexts is searched for **TOCOM_Common_Settings** section to collect required parameters. Corresponding parameters reference is presented in following table:

Parameter Name	Value type	Description
User_ID	String	User ID to be used to connect to relay terminal.
Password	String	Password to be used to connect to relay terminal.
Reconnection_Timeout	Number	Timeout interval in seconds the server will wait before attempting to recover failed relay terminal connection.
Raw_Data_Path	String	Directory, where the server will store raw data from the exchange. If relative path is specified, it is interpreted as starting from application startup directory.

Parameter Name	Value type	Description	
Data_Recording	Identifier	Flag specifying whether the server should record raw data from relay terminal connection to a separate file:	
		Enabled	All messages coming from relay terminal connection will be dumped as is to a file.
		Disabled	Message recording on relay terminal connection will be disabled.
Server_Online_Time	String	Time in "HH:MM" format when the server will initiate logon on relay terminal connections.	
Server_Offline_Time	String	Time in "HH:MM" format when the server will log off from connected relay terminals.	
Session_List	List of identifiers	The list of section identifiers in the configuration file that define each relay terminal connection (session).	

5.5.2 Relay Terminal settings context

This section describes relay terminal-specific connection parameters. Relay terminal description section followed by surrounded contexts is searched for a section with identifiers defined in **Session_List** parameter of exchange common settings context to collect required parameters. Corresponding parameters reference is presented in following table:

Parameter Name	Value type	Description
Relay_Terminal_IP	String	IP address of the target relay terminal.
Relay_Terminal_Port	Number	TCP port number on relay terminal side to connect to establish a session.
Subscription_List	List of identifiers	The list of exchange specific instrument codes defining subscription on target relay terminal.
Backup_Session	String, Optional	Section identifier of another relay terminal connection on which the server will perform failover in case of current session's failure.
Local_IP	String, Optional	IP address of the price server, in case TOCOM relay terminal requires fixed IP address binding.
Local_Port	Number, Optional	TCP port number of the price server, in case TOCOM relay terminal requires fixed port number binding.

6. Client side integration

FIX implementation is expected to be compliant with industry-standard FIX protocol version 4.2. Implementation-specific issues are highlighted in the following paragraphs.

6.1 Sequence number management

Sequence numbers are naturally reset at the beginning of new trading day and preserved during the same day. Latest sequence number pair is stored in instance-specific file in persistent way. New file is created for every business day. Similarly, all outgoing FIX messages are stored to fulfil possible resend requests on FIX session.

6.2 Implemented message types

The server recognises following FIX messages from a member's system:

- Security Definition Request;
- Market Data Request.

The server will send following FIX messages to a member's system:

- Market Data Snapshot / Full Refresh;
- Market Data Request Reject;
- Security Definition.

6.3 Market Data Request implementation

The implementation of Market Data Request handler has following features:

1. FIX field SubscriptionRequestType (tag 263) can only have Snapshot+Updates ("1") and CancelSubscription ("2") values. Value Snapshot (0) is not supported.
2. Only one instrument per request is supported. FIX field NoRelatedSym (tag 146) cannot have values other than "1".
3. Requested instrument is identified by FIX fields Symbol (tag 55) and, in case of options, StrikePx (tag 202). FIX fields MaturityMonthYear (tag 200) and PutOrCall (tag 201) can be optionally resolved, but are not used for identification purposes.
4. Format of FIX field Symbol (tag 55) has format: "*TCCYYX[yyx]*", where *T* is instrument type; *CC* is commodity type; *YYX* is near delivery month (*YY* – year; *X* – month as hexadecimal number); *yyx* is far delivery month (in case of spread contracts only). Codes used for *TCC* are the same as in standard TOCOM issue code.

6.4 Implemented instrument types

Currently only futures and options are supported. Spread contracts are not supported as being still unimplemented on exchange side.

7. Reference documentation

1. FIX protocol version 4.2, available from www.fixprotocol.org
2. TOCOM market data system reference documentation