Rithmic™

R | Protocol API™

Reference Guide
DRAFT

0.29.0.0

January 2020

Table of Contents

Document Information

Introduction

Messages transmitted between Server and Client is in binary format using Google Protocol Buffers API.
Server uses Big Endian format for binary data.

Web Clients should download a copy of Javascript implementation of Protocol Buffers, which can be found at:

https://github.com/dcodeIO/ProtoBuf.js/tree/master/dist

General structure of messages transmitted back and forth between client and server

4 byte message length + R | Protocol API message

R | Protocol API message

Template Id (always required) and

Fields specific to that template

Messages sent/received to/from the server should always be prefixed with a 4 byte message length. Whenever the server doesn't get the 4 byte message length, it will send a reject message with 'message length is invalid' text.

Eg : Suppose the request message "RequestRithmicSystemInfo" after encoding results to say 50 bytes...the buffer that gets sent to the server should be first prefixed with a 4 byte integer with value "50" followed by the actual message. Similarly while parsing messages received from the server, client application should first read the 4 bytes to know the message length and then extract so many bytes to process the actual business message.

Support

The Rithmic Operations team can be contacted by phone at **(877) 408-0008** or by email to **operations@rithmic.com**. Consult your support team for after-hours contact details.

1.

TEMPLATES

1.1 Templates Shared across Infrastructure Plants

TEMPLATE NAME	TEMPLATE ID	MESSAGE DIRECTION
Login Request	10	From Client
Login Response	11	From Server
Logout Request	12	From Client
Logout Response	13	From Server
Reference Data Request	14	From Client
Reference Data Response	15	From Server
Rithmic System Info Request	16	From Client
Rithmic System Info Response	17	From Server
Request Heartbeat	18	From Client
Response Heartbeat	19	From Server
Reject	75	From Server
User Account Update	76	From Server
Forced Logout	77	From Server

1.2 Templates Specific to Market Data Infrastructure

The templates mentioned in this section are serviced on ‘Ticker Plant’. Clients should make sure ‘infra_type’ in the login request is set to ‘Ticker Plant’ in order to run these templates.

TEMPLATE NAME	TEMPLATE ID	MESSAGE DIRECTION
Market Data Update Request	100	From Client
Market Data Update Response	101	From Server
Get Instrument by Underlying Request	102	From Client
Get Instrument by Underlying Response	103	From Server
Get Instrument by Underlying Keys Response	104	From Server
Market Data Update by Underlying Request	105	From Client
Market Data Update by Underlying Response	106	From Server
Give Tick Size Type Table Request	107	From Client
Give Tick Size Type Table Response	108	From Server
Search Symbols Request	109	From Client
Search Symbols Response	110	From Server
Product Codes Request	111	From Client
Product Codes Response	112	From Server
Front Month Contract Request	113	From Client
Front Month Contract Response	114	From Server
Depth By Order Snapshot Request	115	From Client

Depth By Order Snapshot Response	116	From Server
Depth By Order Updates Request	117	From Client
Depth By Order Updates Response	118	From Server
Last Trade	150	From Server
Best Bid Offer	151	From Server
Trade Statistics	152	From Server
Quote Statistics	153	From Server
Indicator Prices	154	From Server
End Of Day Prices	155	From Server
Order Book	156	From Server
Market Mode	157	From Server
Open Interest	158	From Server
Front Month Contract Update	159	From Server
Depth By Order	160	From Server
Depth By Order End Event	161	From Server
Symbol Margin Rate	162	From Server
Order Price Limits	163	From Server

1.3 Templates Specific to Order Plant Infrastructure

TEMPLATE NAME	TEMPLATE ID	MESSAGE DIRECTION
Login Info Request	300	From Client

Login Info Response	301	From Server
Account List Request	302	From Client
Account List Response	303	From Server
Account RMS Info Request	304	From Client
Account RMS Info Response	305	From Server
Product RMS Info Request	306	From Client
Product RMS Info Response	307	From Server
Subscribe For Order Updates Request	308	From Client
Subscribe For Order Updates Response	309	From Server
Trade Routes Request	310	From Client
Trade Routes Response	311	From Server
New Order Request	312	From Client
New Order Response	313	From Server
Modify Order Request	314	From Client
Modify Order Response	315	From Server
Cancel Order Request	316	From Client
Cancel Order Response	317	From Server
Show Order History Dates Request	318	From Client
Show Order History Dates Response	319	From Server
Show Orders Request	320	From Client

Show Orders Response	321	From Server
Show Order History Request	322	From Client
Show Order History Response	323	From Server
Show Order History Summary Request	324	From Client
Show Order History Summary Response	325	From Server
Show Order History Detail Request	326	From Client
Show Order History Detail Response	327	From Server
OCO Order Request	328	From Client
OCO Order Response	329	From Server
Bracket Order Request	330	From Client
Bracket Order Response	331	From Server
Update Target Bracket Level Request	332	From Client
Update Target Bracket Level Response	333	From Server
Update Stop Bracket Level Request	334	From Client
Update Stop Bracket Level Response	335	From Server
Subscribe To Bracket Updates Request	336	From Client
Subscribe To Bracket Updates	337	From Server

Response		
Show Brackets Request	338	From Client
Show Brackets Response	339	From Server
Show Bracket Stops Request	340	From Client
Show Bracket Stops Response	341	From Server
List Exchange Permissions Request	342	From Client
List Exchange Permissions Response	343	From Server
Link Orders Request	344	From Client
Link Orders Response	345	From Server
Cancel All Orders Request	346	From Client
Cancel All Orders Response	347	From Server
Trade Route	350	From Server
Rithmic Order Notification	351	From Server
Exchange Order Notification	352	From Server
Bracket Updates	353	From Server

1.4 Templates Specific to History Plant Infrastructure

TEMPLATE NAME	TEMPLATE ID	MESSAGE DIRECTION
Time Bar Update Request	200	From Client
Time Bar Update Response	201	From Server

Time Bar Replay Request	202	From Client
Time Bar Replay Response	203	From Server
Tick Bar Update Request	204	From Client
Tick Bar Update Response	205	From Server
Tick Bar Replay Request	206	From Client
Tick Bar Replay Response	207	From Server
Time Bar	250	From Server
Tick Bar	251	From Server

1.5 Templates Specific to PnL Plant

TEMPLATE NAME	TEMPLATE ID	MESSAGE DIRECTION
PnL Position Updates Request	400	From Client
PnL Position Updates Response	401	From Server
PnL Position Snapshot Request	402	From Client
PnL Position Snapshot Response	403	From Server
Instrument PnL Position Update	450	From Server
Account PnL Position Update	451	From Server

1.6 Templates Specific to Repository Plant

TEMPLATE NAME	TEMPLATE ID	MESSAGE DIRECTION
List Unaccepted Agreements Request	500	From Client

List Unaccepted Agreements Response	501	From Server

2. Examples

2.1 Login To Rithmic Trading Platform

2.1.a Login Request :

Before sending login request, clients should retrieve Rithmic System Name to which they would like to connect. This information can be retrieved by sending ‘Rithmic System Info’ request.

Clients should create a new object of type “RequestLogin” and populate fields specific to this template which is defined as below.

Field Name	Data Type	Required	Description
template_id	int32	yes	Refer to templates table in section 1.1
template_version	string	yes	Refer to otps_proto_pool.proto Copy the TEMPLATE VERSION string defined
user_msg	String array	optional	Data set in this field will be returned back to client in the response message. More than one data item can be set.
user	string	yes	Username to login
password	string	yes	Password in plain text format.

app_name	string	yes	Name of the client application
app_version	string	yes	Version of the client application
system_name	string	yes	Rithmic System Name as received from Rithmic System Info response
infra_type	enum	yes	Refer to the enumeration block defined in request_login.proto file.

2.1.b Login Response :

This message is sent by the server. Clients should first evaluate 'rp_code' field to determine if the response is GOOD (login success) or BAD (login failed). If the field has one data item with value '0', it indicates the login is successful, if the value is a number other than "0" and has 2 data items, it indicates the login is unsuccessful. The values represent error code and error text respectively. Only if the login is successful, clients should read other fields defined in the response Message.

Field Name	Data Type	Present Always	Description
template_id	int32	yes	Refer to templates table in section 1.1
user_msg	String array	optional	If data set in the request, it will be returned back.
rp_code	String array	yes	If the array length is 1 and value 0 - it is a GOOD response, login success

			If the array length is 2 and value greater than 0 - it is a BAD response, login unsuccess. Error code and text will be available.
fcm_id	string	optional	Present only if response is GOOD
ib_id	string	optional	Present only if response is GOOD
country_code	string	optional	Present only if response is GOOD
state_code	string	optional	Present only if response is GOOD

2.2 Subscribe/Unsubscribe to Market Data Updates

2.2.a Market Data Update Request

Clients should use this template to subscribe or unsubscribe for market data updates for a particular symbol and exchange. It is possible to subscribe or unsubscribe for various market data updates, viz, trades, best bid or offer, order book updates etc. defined in the proto file. Below table gives the details of fields defined in this message.

Field Name	Data Type	Required	Description
template_id	int32	yes	Refer to templates table in section 1.1
user_msg	String array	optional	Data set in this field will be returned back to client in the response message. More than one data item can be set.
symbol	string	yes	The symbol for which request is sent

exchange	string	yes	The exchange for which request is sent
request	enum	yes	Type of request being sent. It can either SUBSCRIBE or UNSUBSRIBE. Refer to the enum block definition in ‘request_market_data_update.proto’ file.
update_bits	uint32	yes	A union of update bits constants defined in the enum block ‘UpdateBits’

2.2.b Market Data Response

Field Name	**Data Type**	**Present Always**	**Description**
template_id	int32	yes	Refer to templates table in section 1.1
user_msg	String array	optional	If data set in the request, it will be returned back.
rp_code	String array	yes	If the array length is 1 and value 0 - it is a GOOD response If the array length is 2 and value greater than 0 - it is a BAD response. Error code and text will be available.

2.2.c Last Trade

The client will receive ‘LastTrade’ messages whenever there is a new trade update from the exchange or as a snapshot from the database. Below table gives the details of fields defined in this Message.

Field Name	Data Type	Present Always	Description
template_id	int32	yes	Refer to templates table in section 1.1
symbol	string	yes	Symbol
exchange	string	yes	Exchange
presence_bits	uint32	yes	A union of updates available. Refer to the enum block ‘PresenceBits’ in ‘last_trade.proto’ file
clear_bits	uint32	yes	Same as presence_bits field. But for the bits enabled it means those price fields should be cleared, viz, last trade price, net_change etc..
is_snapshot	bool	optional	This field is present only if the message received is from database.
trade_price	double	optional	Last trade price for the symbol.
trade_size	int32	optional	Last traded quantity for the symbol
net_change	double	optional	Net_change for the symbol
percent_change	double	optional	Percent_change for the symbol
volume	uint64	optional	Total trade quantity for the symbol
ssboe	int32	yes	Time the server received update from exchange in seconds since EPOC
usecs	int32	yes	Time the server received update from exchange in microseconds.

3. Responses From Server

Response messages from server can be in a single message or it can span across multiple messages. Based on the ‘template_id’ received, clients should check if the message contains field ‘rq_hndlr_rp_code’, or ‘rp_code’.

A response message can have only one of these fields. Clients should use the following logic to determine end of responses in the same sequence. First, check for the presence of ‘rq_hndlr_rp_code’ which indicates there are more response messages to receive. In the absence of this field, clients should check field ‘rp_code’, presence of this field indicates there are NO more response messages to receive.

4. Time/Tick Bar Responses From Server

If the returned bars does not include data for the entire requested time period, and/or if the number of returned bars is a round number (such as 10000), then it is possible that the request was truncated. One can request the missing bars by shifting the time period of the original request to cover the truncated data. This truncation can occur when large amounts of data are requested.

Rithmic™