

**Document title****UNIVERSAL TRADING PLATFORM FOR DERIVATIVES****Document type or subject****Developer guidelines****Revision number/ Version number**

Revision Number: 1.2

**Date**

21 Feb 2018

**Number of pages**

132

**Author**

Euronext

This document is for information purposes only and is not a recommendation to engage in investment activities. The information and materials contained in this document are provided 'as is' and Euronext does not warrant the accuracy, adequacy or completeness of the information and materials and expressly disclaims liability for any errors or omissions. This document is not intended to be, and shall not constitute in any way a binding or legal agreement, or impose any legal obligation on Euronext. This document and any contents thereof, as well as any prior or subsequent information exchanged with Euronext in relation to the subject matter of this document, are confidential and are for the sole attention of the intended recipient. All proprietary rights and interest in or connected with this publication shall vest in Euronext. No part of it may be redistributed or reproduced without the prior written permission of Euronext.

Euronext refers to Euronext N.V. and its affiliates. Information regarding trademarks and intellectual property rights of Euronext is located at <https://www.euronext.com/terms-use>.

## PREFACE

---

### PURPOSE

This document provides guidelines on how to write effective and efficient applications using the UTP protocols for Order Entry (Binary or FIX 5.0) and Market Data (XDP).

It is important to be aware that this document is not a replacement for, but rather a complementary document to the UTP Reference Manuals (see related documentation below).

Most of the points raised in this document are concerned with the most efficient, or Exchange friendly way of performing various tasks, and the Exchange requires application developers to adopt the recommendations given where relevant.

---

### TARGET AUDIENCE

This document is targeted at application developers building trading or market information systems that use those protocols.

---

### WHAT'S NEW?

The following lists only the most recent modification made to this revision/version. For the Document History table, see the Appendix.

REVISION NO.	DATE	CHANGE DESCRIPTION
1.1	4 Apr 2017	<ul style="list-style-type: none"><li>Added specific kinematics for Total Return Futures</li></ul>
1.2	21 Feb 2018	<ul style="list-style-type: none"><li>Update Total Return Future TAIC kinematics</li></ul>

---

### ASSOCIATED DOCUMENTS

The following lists the associated documents, which either should be read in conjunction with this document or which provide other relevant information for the user:

REF #	NAME	REF #	NAME
[1]	XDP Client specifications	[4]	Binary Protocol – C Header File
[2]	FTP Client Specifications	[5]	CCG FIX 5.0 Interface Specifications
[3]	CCG Binary Interface Specifications	[6]	Input Order Throttling Per Contract

All the IT documentation of Euronext can be found on a dedicated page of the Euronext website: [www.euronext.com/en/it-documentation/](http://www.euronext.com/en/it-documentation/)

## CONTENTS

<b>1.</b>	<b>THE UTP-DE INFRASTRUCTURE .....</b>	<b>6</b>
1.1	Introduction .....	6
1.2	Global architecture .....	6
1.3	Overview of market data and order entry protocols.....	7
<b>2.</b>	<b>DISTRIBUTION TECHNOLOGY.....</b>	<b>8</b>
2.1	SFTI® network .....	8
2.1.1	General overview.....	8
2.1.2	Which type of connectivity for which members .....	10
<b>3.</b>	<b>INSTALLATION GUIDELINES FOR A UTP-DE INFRASTRUCTURE.....</b>	<b>11</b>
3.1	Capacity and bandwidth usage .....	11
3.1.1	The XDP Market Data services for UTP-DE .....	11
3.1.2	How have the bandwidths for each Market Data stream determined?.....	12
3.1.3	What are the main differences between a ‘single stream’ and an ‘XDP Grouped’ service? .....	13
3.1.4	The logical layout of XDP services in EUA Test environment .....	14
3.1.5	The Service ID configuration file.....	14
3.2	The member website .....	17
<b>4.</b>	<b>ORDER TRAFFIC MANAGEMENT .....</b>	<b>18</b>
4.1	Architecture overview.....	18
4.1.1	Architecture overview of the Common Customer Gateway .....	18
4.1.2	The Router .....	20
4.1.3	UTP Matching Engine .....	20
4.1.4	The Drop Copy server .....	20
4.1.5	Pre Trade Risk Management (PTRM).....	21
4.2	Managing the CCG logons .....	22
4.3	The CCG security model .....	22
4.4	Input order throttling.....	23
4.5	System Busy .....	23
<b>5.</b>	<b>GENERAL OVERVIEW OF ORDER ENTRY PROTOCOL.....</b>	<b>26</b>
5.1	General overview of the CCG binary interface .....	26
5.1.1	Message format.....	26
5.1.2	Header .....	26
5.1.3	Order Acks and Execution reports.....	27
5.1.4	ReturnCode and reject codes .....	27
5.1.5	Message Sequence Number .....	27
<b>6.</b>	<b>MARKET ACCESS .....</b>	<b>28</b>
6.1	Logon to the Common Customer Gateway .....	28
6.1.1	ITM connection.....	28
6.1.2	Heading logon failures and automatic logons .....	29
6.2	Logout .....	30
6.3	System failure handling.....	30

6.3.1	Common Customer Gateway Failure.....	31
6.3.2	Trading Engine Failure .....	31
6.3.3	Router Failure .....	32
<b>7.</b>	<b>MARKET DATA .....</b>	<b>33</b>
7.1	Standing data .....	33
7.1.1	Static versus dynamic static data .....	33
7.1.2	How standing data are organised within the Trading Host .....	34
7.1.3	Identification of instruments .....	35
7.1.4	What is the link between the FIXML file and a Market Data Service? .....	37
7.1.5	What is the content of the FIXML files? .....	38
7.1.6	XDP standing data streams.....	38
7.2	Link between an option series and its underlying .....	39
7.2.1	Individual Equity Options.....	39
7.2.2	Index Options .....	40
7.2.3	Options on Futures .....	41
7.3	Determination of exercise and trade prices .....	42
7.3.1	How to determine exercise prices.....	42
7.3.2	How to determine trade prices .....	44
7.4	Premium based tick sizes .....	44
7.5	Create and handle strategies .....	47
7.5.1	Examples of Security Definition Request messages .....	48
7.5.2	Specific rules applying to Cash Underlying contracts and delta neutral trades .....	52
<b>8.</b>	<b>MARKET INFORMATION .....</b>	<b>54</b>
8.1	Implied pricing .....	54
8.1.1	Implied in .....	54
8.1.2	Implied out .....	55
8.2	Pack and bundle strategies .....	55
8.2.1	Pack & Bundle pricing and quoting.....	56
8.2.2	Pack & Bundle Implied In pricing .....	57
<b>9.</b>	<b>TRADING AND ORDER HANDLING.....</b>	<b>58</b>
9.1	Retrieve orders and trades .....	58
9.2	Submit orders.....	68
9.2.1	The COrderID field .....	69
9.2.2	Description of the 'New Order Single' message .....	69
9.2.3	What are the mandatory clearing fields per Euronext markets? .....	70
9.3	Pull orders .....	72
9.4	Revise orders.....	77
9.5	Wholesale trading.....	78
9.6	Stock order routing .....	81
9.7	Trade notification.....	82
9.7.1	Quantity fields .....	82
9.7.2	Underlying traded volume algorithm .....	86
9.7.3	Specific case of EFP trades.....	89
9.8	Market making functionality.....	89
9.8.1	Mass quotes .....	89

9.8.2	Underlying traded volume algorithm .....	91
9.9	Batching of orders.....	92
<b>10.</b>	<b>MARKET CONTROL.....</b>	<b>96</b>
10.1	Market status .....	96
10.2	Forced logout / locKout .....	98
10.3	Trader exchange suspend .....	98
10.3.1	Market Control Messages.....	98

## 1. THE UTP-DE INFRASTRUCTURE

### 1.1 INTRODUCTION

Euronext is the primary exchange in the Euro zone with more than 1 300 issuers worth €3.2 trillion in market capitalisation, an unmatched blue-chip franchise consisting of 24 issuers in the EURO STOXX 50® benchmark and a strong, diverse domestic and international client base. Euronext operates regulated and transparent equity and derivatives markets. Its total product offering includes Equities, Exchange Traded Funds, Warrants & Certificates, Bonds, Derivatives, Commodities and Indices.

Euronext Derivatives that comprises the Paris, Amsterdam, Brussels and Lisbon markets offer a wide range of Equity Derivatives and Commodity products:

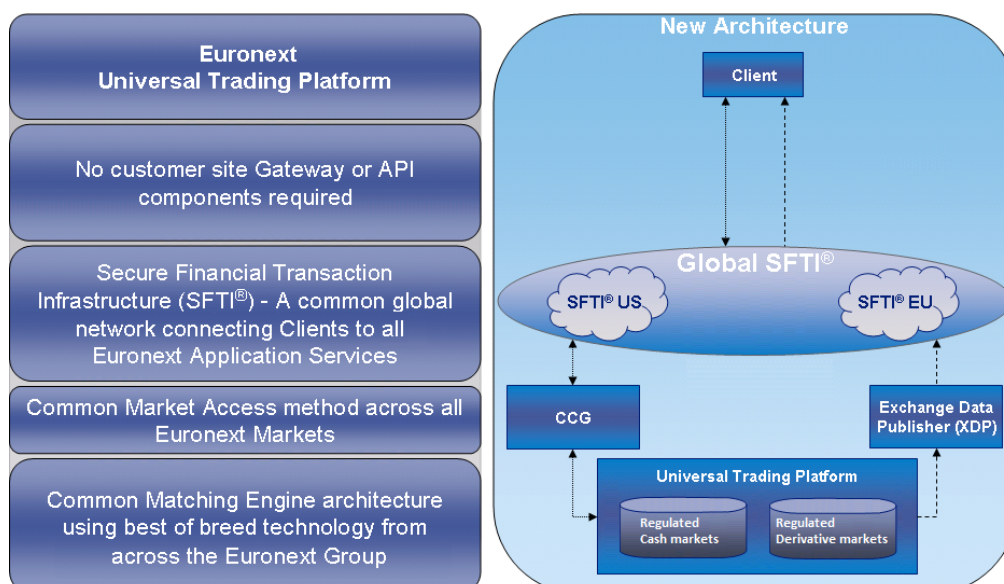
- Stock Indices Futures & Options
- Single Stock & Single Stock Dividend Futures
- Single Stock Options
- ETF options
- Soft and Agricultural Futures & Options

They operate on Universal Trading Platform Derivatives technology (UTP-DE) that is comprised on three main components:

- The Common Customer Gateway (CCG), providing members with multi-format order entry
- The UTP Matching Engine, which supports the trading functionalities of all different markets by providing ultra-low latency and resiliency
- The multicast Market Data Dissemination system (XDP).

### 1.2 GLOBAL ARCHITECTURE

The UTP target architecture for UTP-DE is summarised in the picture below.



More detailed diagrams of the components of this architecture are presented later in this document.

### 1.3 OVERVIEW OF MARKET DATA AND ORDER ENTRY PROTOCOLS

In UTP-DE, Order Entry and Market Data protocols are common across all Euronext derivative markets.

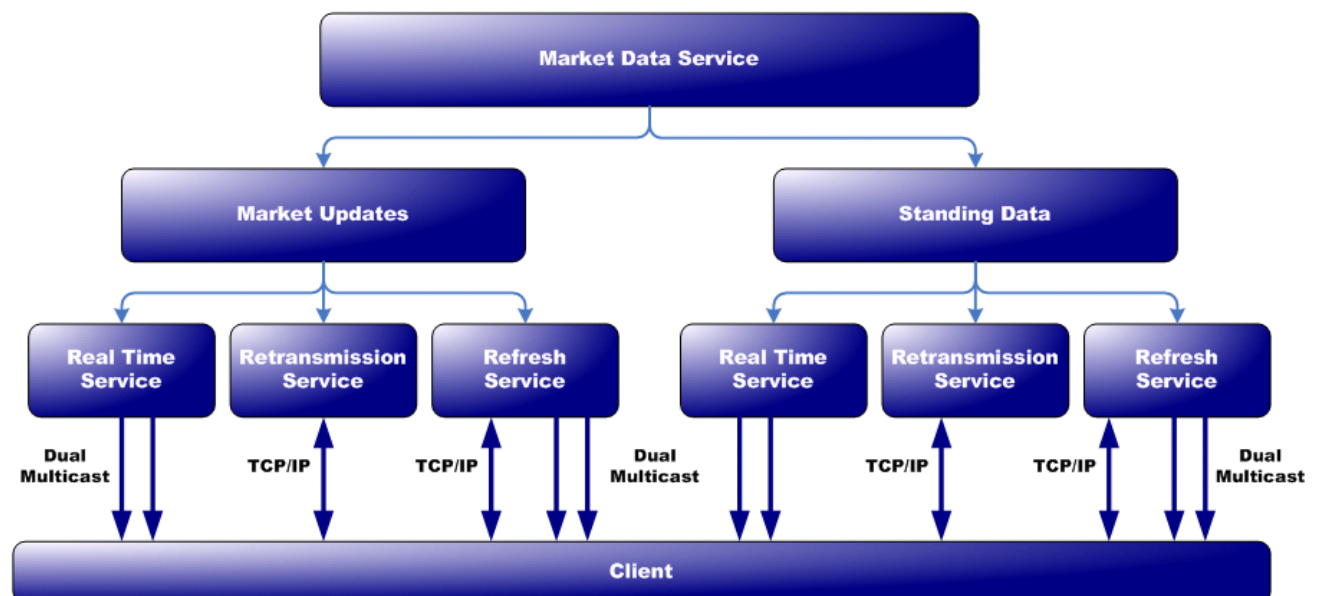
Order Entry and Management are via the Common Customer Gateway (CCG). The CCG supports point to point communication (TCP/IP) via the following protocols:

- FIX (5.0): Industry standard protocol allowing seamless integration with Member's straight through processing systems
- Binary: a proprietary messaging protocol requiring little translation meaning it is the fastest Order Entry method available. Market Making functionality will be supported via this interface.

A multicast feed, XDP (Exchange Data Publisher) is composed of four (4) main following modules:

- Market Data: Real time data broadcast via dual multicast streams (A and B)
  - FAST optimised
  - divided into product groups serviced by separate multicast streams
  - separate multicast streams for intraday standing data updates
- Retransmission: for the re-transmission of a limited number of missed data packets. This is provided via TCP/IP
- Refresh: to provide a Snapshot of the order book at the time of request - would be used for starting your application intraday. This is provided in dual multicast streams
- FTP Standing Data: static and daily data are available to download from an FTP site each morning, prior to the opening of the market

The logical structure of XDP, presented into details in the XDP Client Specifications, is as follows:



## 2. DISTRIBUTION TECHNOLOGY

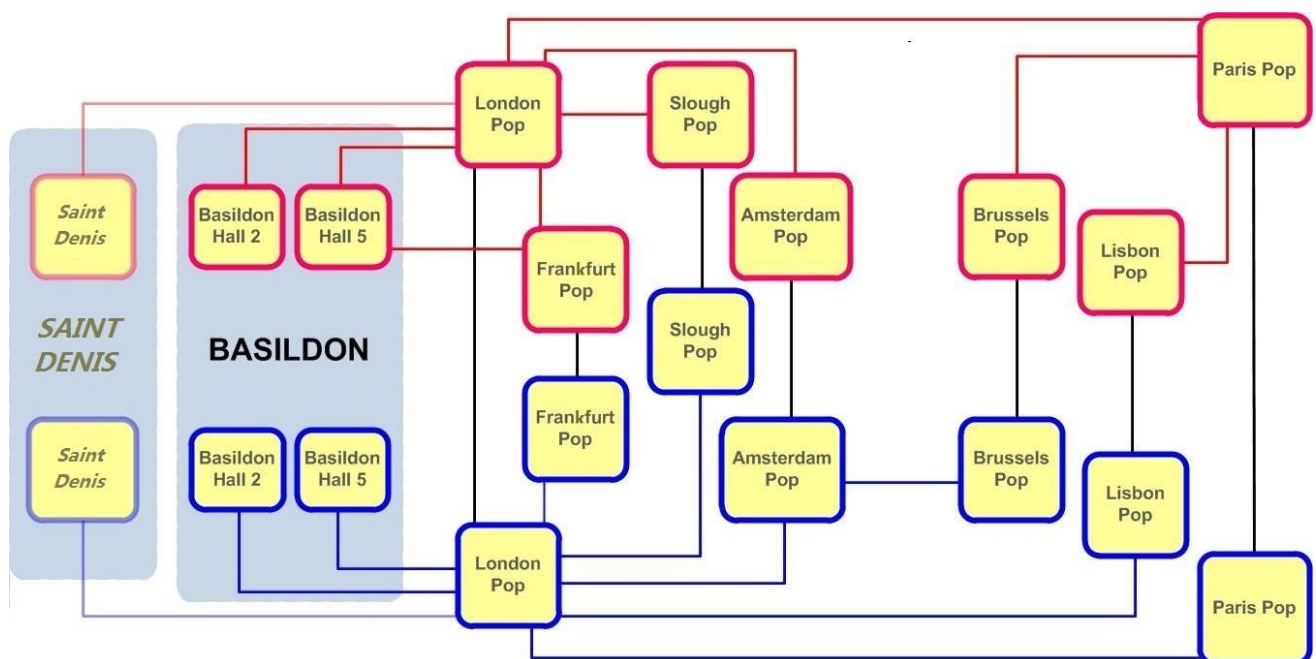
### 2.1 SFTI® NETWORK

#### 2.1.1 General overview

SFTI® (Secure Financial Transaction Infrastructure) is the ICE NYSE global network aimed at connecting members to all Euronext Cash and Derivatives markets.

It has been designed using state-of-the-art networking technologies:

- Dedicated fiber routes with multiple carriers
- Access Centers (POPs) per city for backbone and client redundancy
- Two separate logical networks for complete redundancy
- Utilizes MPLS Layer 3 VPN technology
- MPLS network engineered for lowest latency paths





There are six (6) different SFTI® PoPs (Points of Presence) in Europe: London, Paris, Amsterdam, Brussels, Frankfurt and Lisbon, with two Access Centres in each of them.

SFTI® supports the following connectivity methods (see diagrams in the [SFTI® Connectivity Access Solutions](#) appendix):

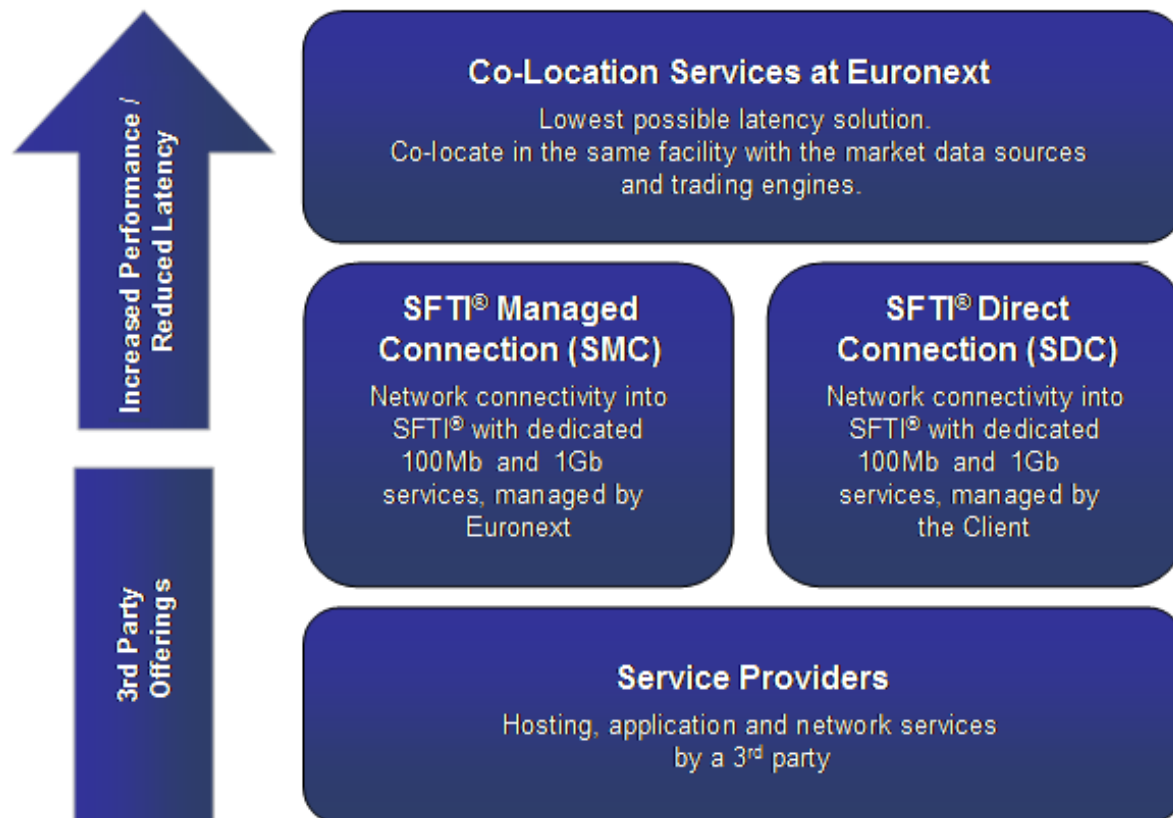
- 1- **[SFTI® Managed Connection \(SMC\)](#)** – SFTI® provides the local access link and customer premises equipment (CPE), supporting a fully managed end-to-end solution from the SFTI® network to the customer's location with a local Ethernet handoff.
- 2- **[SFTI® Direct Connection \(SDC\)](#)** – Customers order their own Ethernet links and configure their own CPE directly connected to SFTI®.
- 3- **[Extranet Service Providers \(ESP\)](#)** – ESPs aggregate clients with layer 3 routing to the SFTI® backbone to some Euronext applications.
- 4- **Cross-connect in a SFTI® data centre** – Customers locate their equipment in the same data centre as the SFTI® PoP. No circuit is required, but only a cross connect. A cross connect into a SFTI® data centre is considered as a SDC connection.
- 5- **Co-location** – The co-location service allows the lowest possible latency to the UTP-DE trading engine, by letting members co-locate their servers within Euronext's data centre.

In Europe, SFTI® went live with the NYSE Liffe new connectivity solutions implemented on July 2008: 2\*100Mbps infrastructures in Basildon. Along with the UTP migration of the European Cash markets, SFTI® is now live for the European Regulated Cash and Derivative Markets and Smartpool. Since the UTP migration of NYSE Liffe, members are able to consolidate further their connectivity, whilst increasing performance.

For further information about the SFTI® network, please contact Client Coverage Center ([ccc@euronext.com](mailto:ccc@euronext.com)).

### 2.1.2 Which type of connectivity for which members

SFTI® and the removal of on-site customer gateways provides members with greater flexibility to choose their access method and their resilience model. The choice between the different connectivity options is summarized in the diagram below:



- **Bandwidth** is a key aspect when connecting to the SFTI® network. Members' requirements depend on which Euronext XDP market data streams they want to subscribe to and also if they wish to consolidate connectivity with Cash services. The different bandwidths available today on the SFTI® network are 100Mbps and 1Gbps. 10Gbps over DWDM will also be available for Basildon co-location facility. More details about XDP data services, capacity and bandwidth usage are provided in the [Market Data chapter](#).
- **Single circuit** – For full network redundancy, members are strongly recommended to have two (2) circuits to connect to the SFTI® network. For resilience purposes, the Exchange also highly recommends to process the A and B multicast feeds on separate lines. From the removal of on-site gateways, Euronext does not require members to have circuits arriving in the same premises (this was a requirement on the legacy system). Members willing to access Euronext derivatives and additional services via a single solution will be required to sign a specific agreement recognizing that they understand and accept the risk of a single connection, without any resilience (note that the SFTI® network is fully resilient).

### 3. INSTALLATION GUIDELINES FOR A UTP-DE INFRASTRUCTURE

This section provides a list of those areas that members must take into account when ordering an infrastructure with Euronext.

All members should liaise with ISVs, internal IT and network staff, prior to implementing UTP to ensure that their hardware, software and network capability is sufficient for the Euronext market models.

#### 3.1 CAPACITY AND BANDWIDTH USAGE

In recent years, the bandwidth requirements for Derivatives markets has significantly increased to levels where the market data needs to be compressed before it can be disseminated. In order to optimise bandwidth usage and increase performance further, the XDP multicast feed is FAST compressed using FAST inline message techniques which reduces the amount of data to be disseminated and written to and read from Network Interface cards.

The legacy system was based on a subscription model. However, over subscriptions could lead to degradation of performance. The XDP feed has been designed so that members can choose their Market Data Services based on their requirements and trading profile. This also allows them to optimise the bandwidth usage on their circuits,

This chapter describes the Euronext Derivatives XDP Market Data streams, how they have been determined based upon the current figures on the system, and what the main differences between them are.

##### 3.1.1 The XDP Market Data services for UTP-DE

The table below lists all the different XDP Market Data Services along with their respective multicast bandwidth requirements. Note that figures are per circuit.

Level 1 means the dissemination of Best Bid and Offer only. Level 1&2 means the dissemination of the whole market depth (no limit).

Product Group	Service Type	Multicast Bandwidth (Mbps)
Equity & Index Derivatives	Level 1	10
Equity & Index Derivatives	Level 1+2	17
Currency Derivatives	Level 1	4
Currency Derivatives	Level 1+2	7
Commodity Derivatives	Level 1	2
Commodity Derivatives	Level 1+2	3
Amsterdam - Equity Derivatives	Level 1+2	13
Amsterdam - Index Futures	Level 1+2	5
Amsterdam - Index Options	Level 1+2	9
Paris - Equity Derivatives	Level 1+2	7
Paris - Index Futures	Level 1+2	5

Paris - Index Options	Level 1+2	3
Brussels - Equity & Index Derivatives	Level 1+2	3
Lisbon - Equity & Index Derivatives	Level 1+2	1
Currency Derivatives	Level 1+2	4
Euronext Equity & Index Derivatives	VAP + Open Interest	2
Euronext Currency Derivatives	VAP + Open Interest	1
Euronext Commodity Derivatives	VAP + Open Interest	1

These services are for market data clients only.

In addition to these multicast streams, members need an additional 5Mbps bandwidth included into the UTP Base services in order to access the Common Customer Gateway, the Refresh and the Retransmission server.

Note that Market Makers can benefit from an additional 10Mbps bandwidth of unicast service.

---

### 3.1.2 How have the bandwidths for each Market Data stream determined?

A complex algorithm has been used to determine the bandwidth of each multicast stream. The calculation is based on the actual inbound traffic on which a number of market updates per order has been applied. A different ratio has been used depending on the type of products (Options or Futures contracts). These figures have then been converted into bandwidth based upon the XDP message length and FAST compression ratio (for the part of the message that is compressed).

For the standing data, the dissemination of standing data prior to the opening of the market has been taken as the basis to determine the appropriate bandwidth.

As calculations were based on a calculation of the maximum output of the matching engines and the peak per second observed over a period of time, bandwidth throttles have been set so that the majority of the time they will not be used. There may however be some peak seconds where the data is throttled back.

Bandwidths for all Cash and Derivatives markets are reviewed on a regular basis and can be adjusted if necessary to guarantee a suitable throttling.

---

### 3.1.3 What are the main differences between a 'single stream' and an 'XDP Grouped' service?

A **single stream** is an XDP multicast feed providing data for a group of products. An application throttle is set to ensure the allocated bandwidth is not oversubscribed. If this limit is reached the data will be throttled back to the specified rate.

The **Grouped service** is purely a short cut for the SFTI® order form. It represents the selection of all the single streams below it.

Unless customers order a grouped stream (e.g. the Equity & Index Derivatives – Grouped) and a single stream for the same group of products (e.g. the Equity & Index Derivatives – Level 1), there will be no overlap between the services.

For a given set of products, the grouped service requires more bandwidth than the single stream. To ensure that delays are not introduced the XDP system does not purely wait for a packet to be full before sending it. It instead uses the combination of the maximum packet size and a timer. For each packet there is also a packet header and a UDP frame which add to the size on the network. This means that although we are minimising delays by invoking a timer, we are utilising packets less efficiently, meaning that more are sent and hence more bandwidth is used on the network. When products are split over several streams it means that packets are less likely to be fully utilised and hence more packets will be used than if they were sent over a single stream. This means that more bandwidth will be required for the same amount of data.

Customers might be concerned by performance differences between a single stream and a grouped service for a given set of products. For the majority of the time the two services will have comparative performance. There are however some circumstances where there could be a difference:

If there is an exceptional peak in a single product whilst the rest of the market remains quieter. In this case the single stream containing the product may be throttled back even though the remaining streams in the group are not fully utilised. The larger single stream may not be throttled as more bandwidth will be available for the individual product. In this scenario the single stream will be faster than the individual stream of the group until the message rate falls below the throttle rate once more.

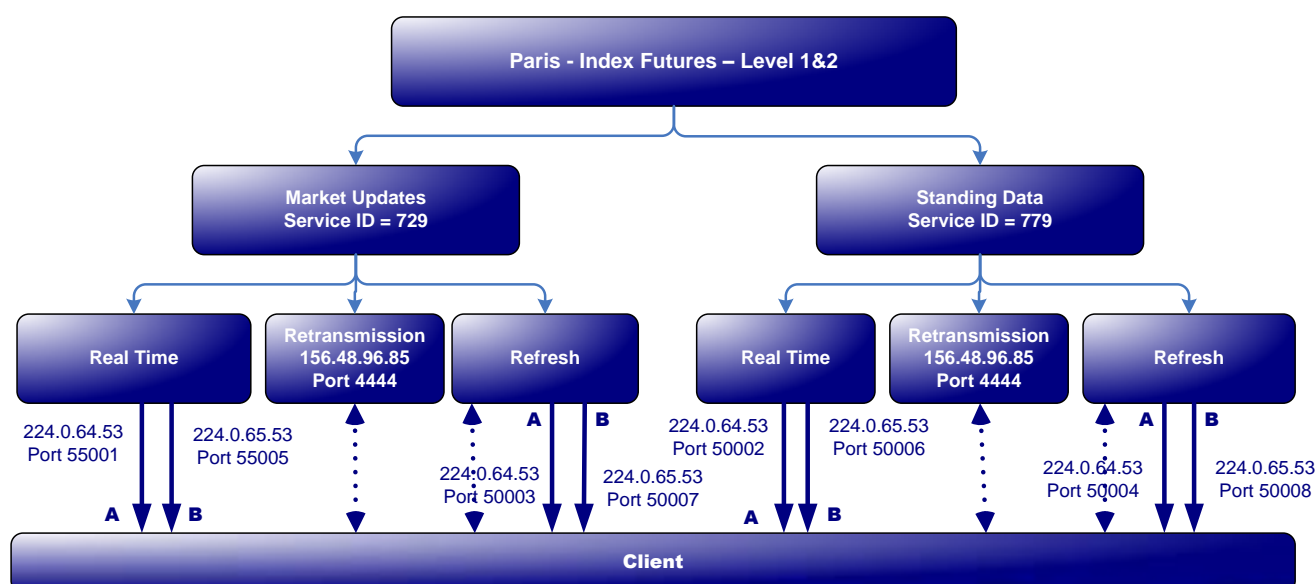
If multiple products spread across the individual streams of the grouped service become busy simultaneously but do not reach the individual limits it is still possible that the single stream will reach its overall throttle setting. In this case a product in the individual stream of a group could be faster than the single stream service until the peak has subsided.

### 3.1.4 The logical layout of XDP services in EUA Test environment

XDP Market Data services cover both Production and EUA test environments (Current and Next). For a given environment, the XDP Market Data Services is logically broken down into Market Updates and Standing Data Updates, then broken down into real time updates, refresh and retransmission services as presented in chapter 1.3.

Prior to accessing the test environment, developers should liaise with their network team to know which Market Data Services have been provisioned on their SFTI® circuits. This will help them to know what Services ID they should connect their feed handler to. They must then refer to the CTSG XDP Configuration document available on the Euronext website.

The example below shows the Market Data services for a single stream, the Paris Index Futures – Level 1&2, in the EUA Next Release Test environment. Note that the values have been taken at the time of writing and might be subject to changes.



### 3.1.5 The Service ID configuration file

IP addresses for the different Market Data services are available to members in a XML configuration file. Developers should note that the file also contains a reference to the name of the FIXML standing data files associated to the contracts listed on each product group.

Files are available for the EUA Test Environments as well as for Production. Developers are encouraged to parse this file in order to configure the IP addresses to the different services they have been provisioned with.

The structure of the file is explained in the following table.

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	Description
<euronextderivativesxdpconfig>						
<productgroup>						
					name	Name of the Product group (note that a product group contains multiple services) Eg: Equity and Index Der - L1
→	<exhosts>					
	→	<exhost>				
					exchange	Exchange Code used to reference the FIXML standing data file
					host	Reference to the Trading Environment for the FIXML standing data file. Possible values: . <b>EQT</b> : Equities
	→	</exhost>				
→	</exhosts>					
→	<services>					
	→	<service>				
					id	Service ID
					description	Long name of the service ID as ordered and provisioned on the member's network
					level	Indicates the type of data distributed over that service. Possible values include: . <b>L1+2</b> : Full market depth . <b>L1</b> : Top of the Book . <b>SD</b> : Standing data . <b>VAPOI</b> : VAP & Open Interest . <b>AtomX</b> : AtomX trade reports, settlements
		→	<multicast>			
					type	Type of multicast data. Possible values include: . <b>RT</b> : Real Time . <b>RFS</b> : Refresh Server
			→	<multicasta>		
					primarysource	IP address of the primary source for feed A
					secondarysource	IP address of the secondary source for feed A
					multicastgroup	IP address of the multicast group for feed A
					port	Port number for feed A
			→	</multicasta>		
			→	<multicastb>		
					primarysource	IP address of the primary source for feed B
					secondarysource	IP address of the secondary source for feed B
					multicastgroup	IP address of the multicast group for feed B

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	Description
					port	Port number for feed B
			→	</multicastb>		
		→	</multicast>			
		→	<tcp>			
					type	Type of the unicast service. Possible values include: . <b>RTX</b> : Retransmission server . <b>RFR</b> : Refresh request
			→	<tcpa>		
					ipaddress	IP address of the unicast service (primary server)
					port	Port number (primary server)
			→	</tcpa>		
			→	<tcpb>		
					ipaddress	IP address of the unicast service (secondary server)
					port	Port number (secondary server)
			→	<tcpb>		
			→	</tcpb>		
		→	</tcp>			
	→	</service>				
→	</services>					
</productgroup>						
</euronextderivativesxdpconfig>						



---

## 3.2 THE MEMBER WEBSITE

Euronext offers an extranet Service aimed at providing members with gateway loading figures. This secure website provides members with the following data:

- **Infrastructure Metrics Files** - Member's own secure infrastructure performance metrics, including:
  - CCG IP addresses
  - Details of logon attempts
  - Round trip times (detailed/summarized) from the gateway through the Matching Engine and back to the gateway
- **Audit Files** - Member's own secure audit and transaction files related to activity on UTP-DE platform
  - Audit Files listed by file, Member's mnemonic and market place
  - Files have been designed to provide a fixed width layout with CRLF line termination

Note that this service is free of charge.

To register to it, customers should contact the CAS (Customer Access Services) at [cas@euronext.com](mailto:cas@euronext.com).

---

## 4. ORDER TRAFFIC MANAGEMENT

---

### 4.1 ARCHITECTURE OVERVIEW

---

The following trading components are part of the UTP-DE architecture:

- 1) [The Common Customer Gateway \(CCG\)](#)
- 2) [The Router](#)
- 3) [UTP Matching Engine](#)
- 4) [The Drop Copy Server](#)
- 5) [Pre Trade Risk Management \(PTRM\)](#)

---

#### 4.1.1 Architecture overview of the Common Customer Gateway

The Common Customer Gateway provides the UTP connectivity solution for members to interact with the Matching Engine, submit and manage their orders and quotes. Developers should pay attention that **NO** market data is provided via the CCG, this includes standing data, market states and RFQs.

All the CCGs are hosted within the Euronext Data Centers (Basildon as the primary site).

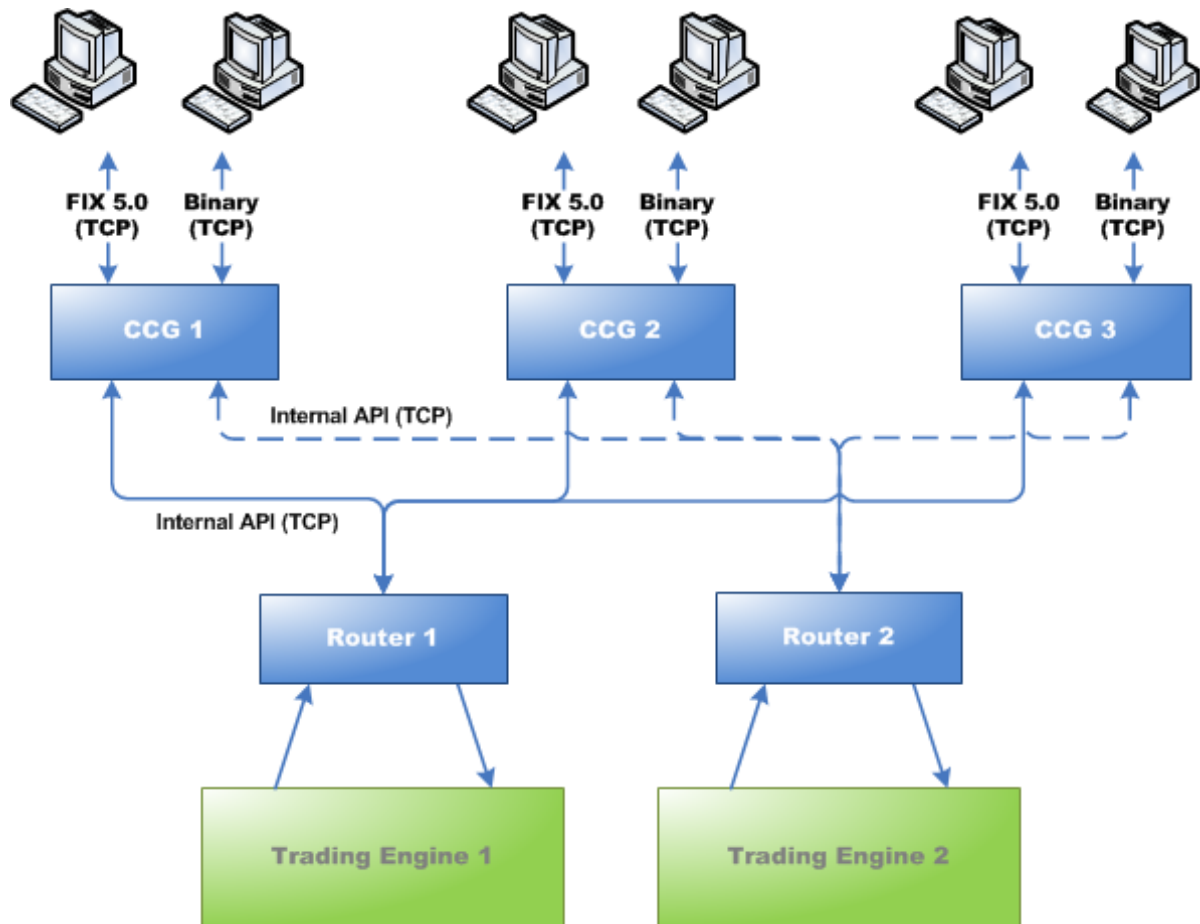
The Common Customer Gateway provides two different interfaces to which members' trading applications can connect to via TCP:

- **A FIX 5.0 based protocol.** This interface provides a seamless integration with members' Front to Back Office systems.
- **A proprietary Binary protocol.** The Binary interface is intended to offer a more efficient interface for latency sensitive customers and Market Makers. It also supports a wider range of features. Indeed, Market Making functionality is only available in Binary. Despite being proprietary the general structure of the binary messages is very FIX "like" and uses the same FIX fields.

Each CCG is connected over TCP to all available Routers and Trading Engines. It then routes all the incoming orders to the appropriate Trading Engine via the appropriate Router. Outgoing messages sent by the Trading Engine and received by the Router are routed to the appropriate session on the CCG.

All the messages sent by the Router are stored by the CCG. Note that this is true for a trading day; however messages do not persist overnight. This allows the CCG to resend any messages that have been missed and are re-requested by either the Router or by client applications. Therefore, if a trader is not connected at the point in time when the CCG receives a message for that trader then the message will be available if the trader does connect later on during the same trading day.

The following diagram provides an overview of the connections between the CCG, the Routers and the Trading Engines. In this example, there are three CCG instances and two Trading Engines.



Each CCG instance has a backup in case of failure.

Each CCG can support up to 600 client sessions with 200 actively submitting orders in any one second.

The CCG maintains a mapping of the **SenderCompID** to the TCP session. This mapping is used to direct responses received from the Router to the appropriate TCP session, using the **TargetCompID** on these messages. These two fields are contained within the standard header of FIX messages.

When a user connects to a CCG the associated checking of last sent and received sequence numbers ensures that any missed messages can be re-transmitted. The same mechanism is used to ensure that any gaps in message sequence numbers during connection are also recovered.

There is only one pool of CCGs to provide trading facilities for all traders via FIX or Binary, i.e. there is not a pool of CCGs for Binary users and another for FIX users.

Order messages from CCG users are routed by the CCG to the appropriate Router instance according to the **SecurityID** field of the incoming message while non trading messages i.e. logon and logoff, will be routed to all Router instances.

#### 4.1.2 The Router

The Router provides an interface between the CCGs and a Trading Engine.

The Router is responsible for passing incoming messages from the CCGs to the Trading Engine and routing responses from that Trading Engine back to the appropriate CCG instance. All interaction between the Trading Engines and the users of the CCG is therefore carried out via the Routers.

Each Router is associated with a single Trading Engine so can only route incoming messages onto that Trading Engine instance. However, each Router is connected to all available CCGs and thus can route outgoing messages to any CCG.

The Router is also responsible for sending messages to the Drop Copy server.

The Router stores all messages that it sends to the CCGs. This is used to recover messages that are re-requested by a CCG.

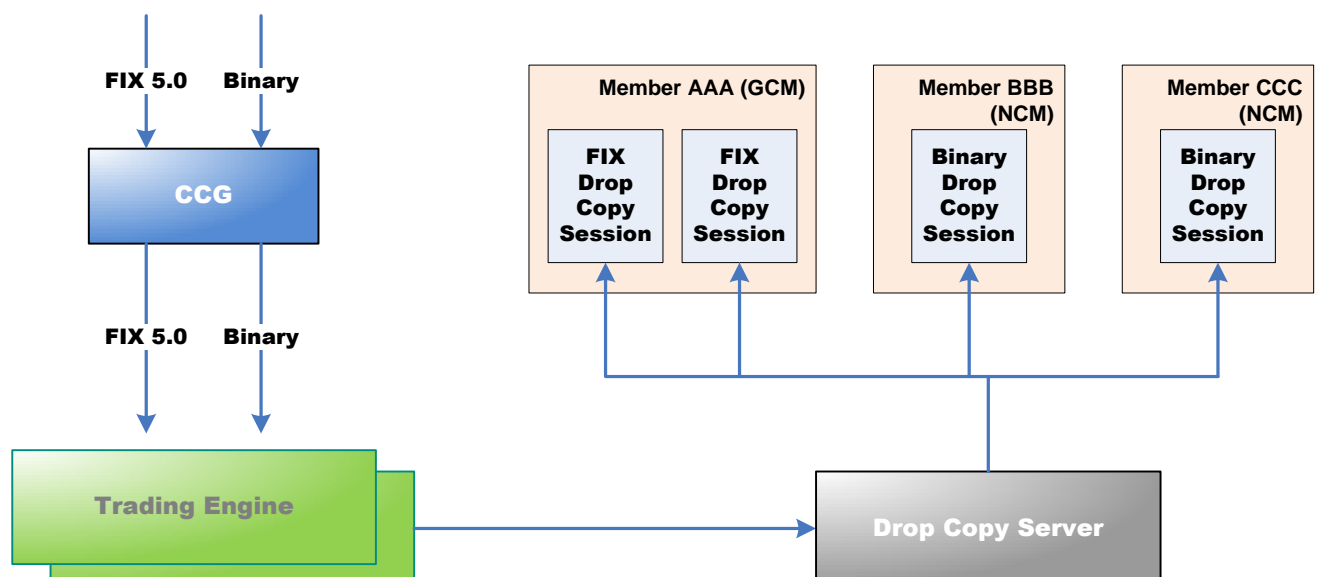
#### 4.1.3 UTP Matching Engine

The Matching Engine of UTP-DE is the heart of the Euronext Information System. This component receives all incoming orders and performs order matching, trade reporting and price dissemination.

#### 4.1.4 The Drop Copy server

The Drop Copy Server provides registered users with a copy of **Execution Reports** from monitored sessions. The Drop Copy Server provides the external interface for users to receive **Execution Reports**. FIX and Binary interfaces are provided. A specific process running on the same machine as the Router reads **Execution Reports** from the Router and forwards these messages on to every available Drop Copy Server. Note that both orders and trades are provided.

The following diagram illustrates the flow of **Execution Reports** for Drop Copy.



A Drop Copy user can be configured to receive the same response messages as sent to one or more CCG users. There may be a N to M relationship between users. One Drop Copy user can receive messages for multiple ITMs and each ITM can be monitored by multiple Drop Copy users.

Note that Drop Copy can only be used to receive messages. A Drop Copy user cannot use the Drop Copy service to perform any other operation.

When an ITM logs on to a CCG the message that the CCG sends to the Routers indicates whether the ITM has some Drop Copy recipients configured.

Every Drop Copy Server instance receives all **Execution Reports**. Users of Drop Copy will be allocated access to these Drop Copy Servers in similar way to the way users are allocated access to CCGs.

Drop Copy might require members to order extra unicast bandwidth on their SFTI® infrastructure. Members willing to set up one or more Drop Copy users should therefore liaise with CCC (Client Coverage Center) at [ccc@euronext.com](mailto:ccc@euronext.com) in order to validate that they have sufficient bandwidth.

All **Execution Reports** are copied for traders connecting via the CCGs. Note that **Execution Reports** are not generated for Mass Quote submissions but are generated when quotes trade.

The Drop Copy server stores all the messages that it receives throughout the trading day. Therefore it can be used by a Drop Copy client to retrieve any messages that may have been missed while that client was not connected.

Drop Copy users connect to the Drop Copy server using TCP.

It should only be possible for a Drop Copy user to be configured to monitor at ITM which:

- Has a clearing relationship with the monitoring ITM for the monitored contracts
- Is within the same membership as the monitoring ITM

**Note**

All these new components as well as the Trading Engines run on machines using the 64 bit Red Hat Enterprise Linux Operating System.

---

#### 4.1.5 Pre Trade Risk Management (PTRM)

PTRM is a group of functionalities destined to Risk Managers (either from Trading Firms or Clearing members) that offers the possibility to automatically block or cancel orders (that do not meet set prices or size parameters, that is based on a financial instrument that a trader does not have permission to trade or that compromise the firm's own risk management thresholds).

The 3 functionalities are the following:

- A Kill Switch Button
- Order Size Limits
- Market Exposure Position Limits

---

## 4.2 MANAGING THE CCG LOGONS

When a user sends a **Logon** (A) message to a CCG, the CCG forwards that logon to every available Router instance. Only once a successful logon status has been received by the CCG from a Router instance will the CCG forward messages onto that Router.

This approach ensures that a Trading Engine will not receive messages from a trader until he has successfully logged on to that Trading Engine instance.

Until a CCG receives a successful logon status from a Router any incoming messages for contracts handled by that Router will be rejected and the submitting trader will receive a **Reject** (3) message with the `RejectReason` set to 'Logon problem'.

Note that as each Trading Engine responds independently to the logon message. The CCG may therefore reject orders for some products while at the same time accepting orders for other products.

---

## 4.3 THE CCG SECURITY MODEL

It is important that members understand the CCG security model. As mentioned in the previous paragraph, each ITM needs to be provisioned for use on a CCG before it may be used to submit orders via the CCG. The following information must be provided to the Exchange before an ITM can be configured:

- **Protocol (Binary / FIX).** For use via a CCG each ITM needs to be configured to use either the FIX or Binary protocol. Specifying the protocol for an ITM as either "FIX" or "Binary" will not affect access to UTP-DE however an individual ITM may only log on into one interface at any one time.
- **Valid Site IDs.** A valid list of sites from which an ITM may connect to a CCG must be specified. For each valid site a Site ID must be quoted. This is the Site ID that would have been provided when the member's SFTI® connection and / or application services were provided (e.g MNE01). When specifying the Site IDs members should ensure that any secondary or disaster recovery sites are included. Should a member be connecting via a third party such as an ASP or an ESP, members should contact them to obtain all their Site IDs for inclusion.

The Site ID and the CCG protocol information is set at a Member level and is then automatically inherited by each member which sits under that member. It is also possible to override the protocol and / or the Site ID(s) at the ITM level.

#### 4.4 INPUT ORDER THROTTLING

In addition, the Trading Host has a protection mechanism that restricts the total number of messages (of all types) that a trading session can submit per second.

**In UTP-DE, each session on the CCG has a session limit set at 100 msgs/s.**

An inbound order throttle at contract level can be set for some contracts. When the inbound throttling is enabled for a specific contract (and a certain list of message types), the Trading Host checks the number of messages sent

- Per second
- Per ITM
- Per contract
- Per type of message

and will reject any n+1 messages submitted in the current second.

Most of the time, input order throttling currently applies for markets and commodities on which Mass Quotes have been implemented. Therefore, for those contracts, client applications are recommended to submit homogeneous batch orders / revises calls.

Current values of the throttling applying to the different Euronext markets can be found in the *'Input Order Throttling Per Contract'* document on the [Euronext / order entry](#) section of the Euronext "IT documentation" website.

The value of the throttling per contract can be retrieved in the FIXML standing data files, at the <Contract> level, in `NestAttr = "121"` (throttle for incoming orders).

```
<Contract SecGrp="POCA1" LogSym="CA1" Desc="Carrefour (100)" Exch="XMON"
RndLot="100" Ccy="EUR" MinPxIncr=".01" PxQteMeth="STD" SettlMeth="P" ExerStyle="1"
FlexInd="N" CFI="XXXXXX" Src="4" ID="FR0000120172">
  <NestAttr Typ="25" Val="100" />
  <NestAttr Typ="101" Val="1" />
  <NestAttr Typ="102" Val="1" />
  <NestAttr Typ="103" Val="100" />
  <NestAttr Typ="104" Val="2" />
  <NestAttr Typ="110" Val="C" />
  <NestAttr Typ="120" Val="T" />
  <NestAttr Typ="121" Val="4" />
```

Note that when no throttling applies, `NestAttr = "121"` is equal to 0.

#### 4.5 SYSTEM BUSY

When the client application submits a new order, a revision request or a new Mass Quote, it must pay attention that the request is not rejected because the ITM session has exceeded the maximum number of requests permitted within each second (determined dynamically by the Trading Host and globally applied to all sessions).

In the CCG, the appropriate response message will be sent with the **RejectReason** (or **QuoteRejectReason** in case of a Mass Quote) set to **99 = Other** and the **Text** field populated with **'SYSTEM BUSY'**.

```

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 05 A2 : 1442
OrderID         : 00 00 00 00 00 00 00 00 : 0
TradeID         : 00 00 00 00 00 00 00 00 : 0
ExecID          : 00 00 00 00 00 00 00 00 : 0
ClOrdID        : 0F AD 86 A9 : 263030441
OrigClOrdID     : 00 00 00 00 : 0
CrossID         : 00 00 00 00 : 0
ListID          : 00 00 00 00 : 0
TransactTime    : 01 E0 23 63 : 31466339
ClOrdLinkID     : 00 00 00 00 : 0
LastPx          : 00 00 00 00 : 0
Price           : 00 00 00 00 : 0
StopPx          : 00 00 00 00 : 0
ReturnCode      : 00 00 00 00 : 0
ExpireDate      : 00 00 00 00 : 0
CumQty          : 00 98 96 7F : 9999999
LastQty         : 00 00 00 00 : 0
OrderQty        : 00 00 00 00 : 0
MinQty          : 00 00 00 00 : 0
LeavesQty       : 00 98 96 7F : 9999999
QtyDelta        : 00 00 00 00 : 0
MassStatusReqID : 00 00 00 00 : 0
OtherLegLastPx  : 00 00 00 00 : 0
OrdRejReason    : 00 63           : 99
ClearingInstruction : 00 00           : 0
TradingSessionID : 00              : 0
CrossType       : 00              : 0
CustOrderCapacity : 00              : 0
PartyRole       : 00              : 0
ExecRefID       :                  : ()
OrigCompID      :                  : ()
LastRptRequested : 00              : 
Text            :                  : (System Busy)
SecondaryClOrdID :                  : ()
SecurityIDSource : 38              : 8
SecurityID       :                  : (AOASL121104300C)
Account          :                  : ()
PackageID        :                  : ()
SecondaryOrderID :                  : ()
RiskID           :                  : ()
MatchingCode     :                  : ()
TradeInputDevice :                  : ()
OtherParty       :                  : ()
OrdStatus        : 38              : 8 ** Rejected **
OrdType          : 00              : 
Side            : 32              : 2
TimeInForce      : 00              : 
ExecType         : 38              : 8 ** Rejected **
OrderOrigin      : 00              : 
OrderCapacity    : 00              : 
TradeInputSource : 00              : 
AccountCode      : 00              : 
ClientInfo       :                  : ()
PartyID          :                  : ()
PostingAction    :                  : ()
OtherLegSecurityIDSource: 00          : 

```



```
OtherLegSecurityID : ()
OtherLegReferenceNo : ()
NoLegs : 00 : 0
```

“SYSTEM BUSY” messages can be sent either as a result of the key breaching the throttling limitation, at the ITM or at the contract level. Members should therefore pay attention to the exact cause of the messages.

When received, “SYSTEM BUSY” needs be handled sympathetically by the calling application and, in the simplest model; the request should be retried after a short pause. This is **only** returned when the session has been throttled (i.e. the maximum number of requests for the current second has been exceeded). This means that **no** requests will succeed until the throttle is lifted at the end of the current second. The precise length of time an application needs to pause will, therefore, depend on where the session is within the current second's cycle. A guaranteed response would be forthcoming with a 1 second pause, but a more reasonable pause should be considered of approximately 200 milliseconds.

---

## 5. GENERAL OVERVIEW OF ORDER ENTRY PROTOCOL

---

### 5.1 GENERAL OVERVIEW OF THE CCG BINARY INTERFACE

---

#### 5.1.1 Message format

Messages in the Binary protocol are either fixed or variable length depending on whether they include repeating data entries. Fields within the messages have a defined number of fixed-length fields containing both binary and ASCII data:

Binary data must be sent in **Big Endian** format.

ASCII data must be **left justified** and **null padded**.

When using the CCG Binary interface, developers must pay attention that ALL fields within the messages should be populated. As a general principle, all fields should be populated with a null value if they are not required.

Spaces can be used if the fields are of ASCII datatype

0 can be used if the fields are of Binary datatype. However, for some fields such as Price, 0 might be a valid value. In this case, the CCG specifications indicate specific values such as LIFFE\_NO\_PRICE (999999999, i.e. 9 nines) or LIFFE\_NO\_VOLUME (999999999, i.e. 9 nines).

---

#### 5.1.2 Header

When calculating the length of the message, developers must note that the message length should include the 16 bytes of the header. They should also pay attention that some messages have variable sizes depending on the number of repeating legs.

- **Message Version Profile**

The Message Version Profile defined in the CCG Binary interface is used for response only and not for application messages. Because there are no message variants, the field is completely ignored at the moment.

The hex number against the message should just be converted to an unsigned 16 bit Integer. It represents both the message type and version number. If there is a new version of a message then the new version will have a different value.

---

### 5.1.3 Order Acks and Execution reports

Client apps are notified that a new order has been successfully accepted by the Trading Host via an **Order Ack (a)** message in Binary or an **Execution Report (8)** in FIX with `ExecType = 0 (New)` and `OrderStatus = 0 (New)`.

Further to that, all changes to the order are notified via **Execution Report** messages. Client apps should pay attention to the values of the *ExecType* and *OrdStatus* fields:

- The `ExecType` field contains the event that triggered the Execution Report, i.e. '5' as a result of an order revision
- The `OrdStatus` field gives the current status of the order as the result of that event.

---

### 5.1.4 ReturnCode and reject codes

When an order message is being rejected, application developers should pay attention to the reject code returned. The `ReturnCode` is populated by the Trading Engine where reject codes including `OrdRejReason`, `CxlRejReason`, `MassCancelRejReason`, `QuoteRequestRejReason`, `RejectReasonCode`, `ListRejReason`, `QuoteRejReason` are populated by the CCG. Therefore, if there is no `ReturnCode` but there is a reject code then this means that the order message has been rejected by the CCG.

Note that the `ReturnCode` is always populated in the Execution Report, i.e. when an order is successfully revised, `ReturnCode = '16252931' (TRADE_STATUS_REVISED)`.

---

### 5.1.5 Message Sequence Number

The Message Sequence Number should be contiguous and reset to 1 at the start of each day.

The message sequence number is on per session basis and applies to both inbound and outbound messages.

If messages are received in an incorrect sequence, they are rejected with a **303** reject code (Invalid Sequence Number). Developers must pay attention that messages are sent with a correct sequence.

## 6. MARKET ACCESS

### 6.1 LOGON TO THE COMMON CUSTOMER GATEWAY

#### 6.1.1 ITM connection

The **Logon (A)** message is used to connect (i.e. to establish a session) an Individual Trader Mnemonic (ITM) to the Trading Host.

```
MsgId           : 00 21           : 33 [A]
iLen            : 00 42           : 66
lSeqNum         : 00 00 00 00 : 0
LastSeqNum      : 00 00 00 00 : 0
HeartBtInt      : 00 5A           : 90
SenderCompId    :                 : (XYZ)
SenderSubId     :                 : (DTS)
Versions        :                 : (A 15 10 11 13 18 1a 19 1D 1E 1F 1G 1N 1R 1)
CancelOnDisconnect : 31           : 1
```

```
MsgId           : 00 21           : 33 [A]
iLen            : 00 42           : 66
lSeqNum         : 00 00 00 00 : 0
LastSeqNum      : 00 00 00 00 : 0
HeartBtInt      : 00 5A           : 90
SenderCompId    :                 : (EXCHG)
SenderSubId     :                 : ()
Versions        :                 : ()
CancelOnDisconnect : 00           :
```

```
MsgId           : 02 51           : 593 [UC]
iLen            : 0B C8           : 3016
lSeqNum         : 00 00 00 01 : 1
ContractAvailabilityID: 00 00 00 01 : 1
SecurityIDSource : 50             : P
AvailabilityStatus : 31           : 1
LastRptRequested : 4E             : N
NoContracts      : C8             : 200
SecurityID       :                 : (JFFCE)
SecurityID       :                 : (JFXFC)
```

.....

```
MsgId           : 02 51           : 593 [UC]
iLen            : 01 78           : 376
lSeqNum         : 00 00 00 02 : 2
ContractAvailabilityID: 00 00 00 02 : 2
SecurityIDSource : 50             : P
AvailabilityStatus : 31           : 1
LastRptRequested : 59             : Y
NoContracts      : 18             : 24
SecurityID       :                 : (AFHEI)
SecurityID       :                 : (AFAD6)
SecurityID       :                 : (AFBY6)
SecurityID       :                 : (AFCIO)
SecurityID       :                 : (AFGM6)
SecurityID       :                 : (AFTTM)
SecurityID       :                 : (AFIM )
```

.....

When logging on, the CCG validates that the session attempt comes from a valid Member Mnemonic specified in the `SenderSubID` field, ITM specified in the `SenderCompID` field and site ID.

One ITM can only establish one single session on a given Trading Engine. This is true in the following situations: The ITM tries to connect twice to the same CCG. In this scenario, any further attempts will be rejected by a **Logon Reject (L)** message. Developers should look at the `Text [58]` field for further information. If the ITM attempts to connect twice to the same CCG, the second attempt will be rejected by a 'ITM already logged on'.

- **The Heartbeat interval**

When connecting, client applications should specify a heartbeat interval. If no heartbeat is specified in Binary, the Logon will be rejected and the member disconnected from the CCG. In FIX, if the field is left blank, then a default value will be applied. It is up to the member to decide the heartbeat interval. However, a value of 30 seconds is recommended.

- **The CancelOnDisconnect field**

If set, a mass cancellation of non-GTC orders will be triggered on any type of logoff (ie logoff request, disconnection on failure, forced disconnection). This is effective on all connections, only the default value can be set.

- **The LastSeqNum field**

Whatever the value of the `LastSeqNum` in the `Logon` message, the CCG always provides as a response to a successful logon the last sequence number of the message that was last processed by the server. However, because the CCG records all outbound messages for a user in memory during the day, members must pay attention to this value.

- **LastSeqNum = 0.** In this case, **ALL** outbound messages since the beginning of the day will be sent after the logon has been successfully accepted. Developers should pay attention to the potential traffic that such a setup can generate when logging on during the day. However, developers are highly recommended to use 0 for the first connection at start of day.
- **LastSeqNum = N**, N being any positive value. The CCG then rewinds to this number plus 1 and replays all outbound messages.
- **LastSeqNum = -1**, The CCG does not replay any message but continues from the last known transmitted sequence plus one.

---

### 6.1.2 Heading logon failures and automatic logons

Developers should not simply allow the Exchange to lock them out should a logon error occurs (see [System Failure Handling](#)). The `Logon Reject (L)` message may return a number of values upon failure in the `Text [58]` field. Developers should ensure that their application checks these return values, takes the appropriate action and informs the application user accordingly.

For example, a `Text` field of "**CompID problem**" is returned either when no ITM is specified or if the ITM is not recognised as belonging to the Member Mnemonic specified in the `SenderSubId` field.

ISVs/Members are reminded that the use of automatic logon facilities may impact the performance of the market unless the following guidelines are adhered to:

1. Once the Logon is called, then the software interfacing with the CCG should wait for a response from the Trading Host, before attempting any further action, certainly before attempting to logon again.
2. Occasionally there is a delay at the trading host when processing logon requests (at peak times for example). When this occurs users should be patient and not disconnect and try again. Doing so will only put the new request at the end of the queue to be processed, add to the queue and potentially add to the delay. That is, it will take longer to logon.
3. However, many Members have automatic logon facilities, which are waiting for a very short period of time before disconnecting and trying to logon again. Members should consult with their internal IT departments and / or ISV software supplier to ensure that the automatic logon facility can be adjusted to wait for a minimum period before attempting to logon again. Whilst the Exchange advises that automatic logons to the Trading Host wait for a response in all cases (due to the reasons above), we accept that automatic logon procedures may be different for different markets. We therefore suggest that a minimum response time of 30 seconds is set prior to retrying to logon.

## 6.2 LOGOUT

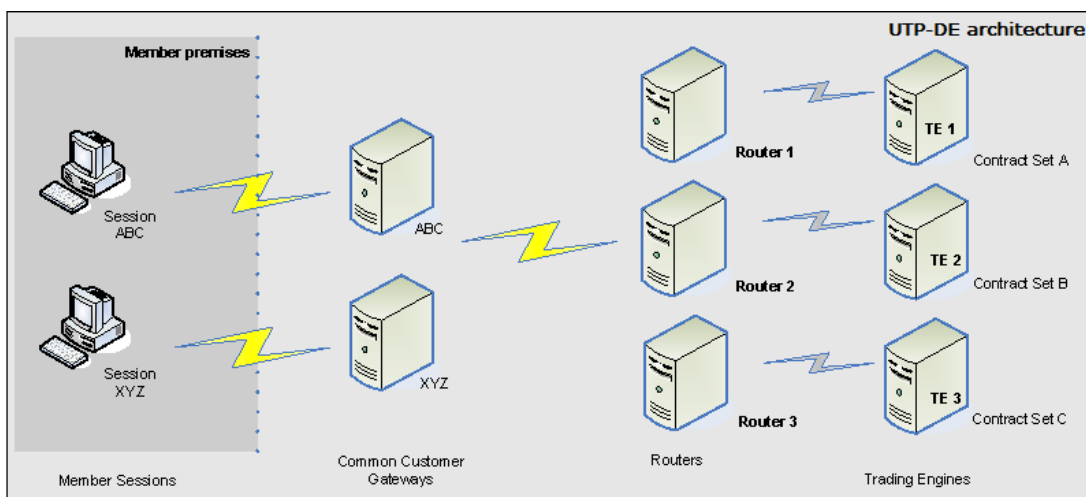
The **Logout** message is used to close a trader's session to the Host. The application developer should always ensure that the trader knows his logon state with respect to the Trading Host at all times.

## 6.3 SYSTEM FAILURE HANDLING

When a client application detects a failure, it should intelligently interpret the type of failure, differentiate between them and inform the application user in the appropriate manner.

Indeed, failures can stand at different points and therefore applications must differentiate between a network / connection error, a Common Customer Gateway failure, a Trading Engine failure or a Router failure.

The logical diagrams below outline these different points of failure in UTP-DE. This is aimed at helping developers familiar with the management of failure.



---

### 6.3.1 Common Customer Gateway Failure

If the primary Common Customer Gateway fails, members will be able to connect to the back-up CCG. Virtual IP addresses are used for the back-up CCG so members can still try to attempt to connect to the same IP address. They are recommended to keep polling this until they regain connection.

There should be no gap in sequence numbers between the primary CCG and its backup.

---

### 6.3.2 Trading Engine Failure

A Trading Engine Failure does not result in a disconnection from the market. It only affects trading in the contracts listed on the failed instance. The sequence of events characterising a Trading Engine Failure is as follows.

The Router detects that a Trading Engine has failed. It disconnects from all CCGs and rejects any orders in its queue.

The CCGs detect that the Router has disconnected. **Contract Availability (UC)** messages for the affected contracts with `AvailabilityStatus = 2` are generated by the CCGs and send to traders attached to each CCG. **Product Availability (741)** messages with `TradingAvailableFlag = 0` are also sent out over XDP indicating that certain contracts are not available for trading.

When the Trading Engine re-starts all non-GTC orders have been lost. Execution Reports for such lost/pulled orders are sent to client apps by the CCGs.

`Product Availability (741)` messages with `TradingAvailableFlag = 1` are sent out over XDP indicating that contracts are available again. The CCG also notify traders via `Contract Availability (UC)` messages with the `AvailabilityStatus = 1`.

Following the reception of `Contract Availability` messages with `AvailabilityStatus = 1`, members can retrieve a proper view of their book by calling `Order Mass Status Request (AF)` messages.

---

### 6.3.3 Router Failure

A Router Failure might look similar in the sequence of events to a Trading Engine Failure especially from a CCG point of view. However, developers must make sure that their client applications distinguish between the two types of failure and alert the user in the appropriate manner.

A Router failure like a Trading Engine Failure does not result in a disconnection from the market. It only affects trading in certain contracts.

If a Router fails, the connections between that Router and the CCGs will be lost. The CCGs will detect the disconnection and send some Contract Availability (UC) messages with `AvailabilityStatus = 2`.

As a Router is attached to a Trading Engine, only contracts listed on that Trading Engine will become unavailable for trading. The Trading Engine also detects the Router failure and automatically pulls all Day orders to traders connected to CCGs (during the backward compatibility period, a trader connected to the same Trading Engine will not be affected by the Router failure). In case of a Router Failure, Execution Reports are generated for these pulled orders, however they can't be sent to the CCG and remain in the trade out queue of the Router. These Execution Reports will not be sent out either to the Drop Copy Server as this is the responsibility to the Router to send such messages.

When the Router restarts, it reconnects to the CCGs and the Drop Copy Servers. The CCGs will send:

- Contract Availability (UC) messages to client apps indicating that the affected contracts are available again (**AvailabilityStatus = 1**).
- Product Availability ('741') messages with `TradingAvailableFlag = '0'` will be sent out via XDP to notify client apps that some contracts are not available for trading.
- Product Availability ('741') messages with `TradingAvailableFlag = '1'` will be sent out via XDP to notify client apps that contract have become available.

The Router sends to all the CCGs and the Drop Copy Servers the outstanding reports for the pulled orders.

In case of a Router Failure, no message is sent out over XDP. ISVs and Member Developers are therefore recommended to pull their non GTC orders upon receiving the Contract Availability messages notifying them that some contracts are unavailable for trading. This way, they are not dependent upon receiving the Execution Reports – which they will only get after the Router has been restarted.

Following the reception of Contract Availability messages with `AvailabilityStatus = 1`, members can retrieve a proper view of their book by calling Order Mass Status Request (AF) messages.



---

## 7. MARKET DATA

This section describes how standing data is organised as well as what client applications should pay attention to when downloading standing data at the start of the trading day and within the day for intraday adds.

---

### 7.1 STANDING DATA

---

#### 7.1.1 Static versus dynamic static data

In UTP, instrument characteristics can be divided into two categories:

**Static data** which refers to the characteristics of an instrument that are not subject to change intraday i.e. Tick Size Numerator, Exchange Code, round lot.

**Dynamic data.** This is relative to intraday adds, i.e. creation of new strikes and new strategies.

Static data is provided via a set of FIXML files on an FTP site available either via the Internet or the SFTI® network. The FTP files contain the full set of instrument characteristics.

Because the list of listed and tradable instruments and their characteristics are subject to changes on a daily basis, it is highly recommended to download static standing data from the FTP server every day before the market opens.

In addition to the FIXML files described in detail in the following paragraph, a subset of static standing data is disseminated in the XDP multicast standing data streams before the market opens. Note that only a subset is disseminated. It does not provide client applications with the required fields for submitting orders. For example the Tick Size Denominator and the Price Decimal Locator are not disseminated via XDP.

Notification of intraday created strategies and strikes are only available via the XDP multicast standing data streams. Developers must pay attention that there is no update to the FIXML files intraday.

Therefore, to ensure that their application is using the latest information, developers should ensure that the standing data is re-loaded immediately after joining a Market Service and before processing real-time market updates on the multicast channels. Developers are highly recommended to do a request of standing data on the Refresh server. This allows the application to get the updated list of intraday created instruments. Developers should **NOT** attempt to speed up their applications by trying to automatically generate AMRs (for explanation of AMR see section [7.1.3](#)). Similarly, developers should **NOT** attempt to speed up their applications by using previously stored Standing Data.

#### Specific note on strategies

There is a mechanism within the Matching Engine where any strategy on which there is no resting GTC / GTD orders will be “hidden” the following day. As a consequence, only the non-hidden strategies will be available in the FIXML files. As any persistent orders on delta-neutral strategies are automatically pulled at the end of the day by the Trading Host, there will be no delta-neutral strategies in the FIXML files. Therefore, it is highly recommended to download the list of valid strategies per Commodity code at the beginning of each trading day via the FIXML files, and / or via the standing data multicast service. This will prevent users submitting orders on non-existing strategies. If a client application wants to trade a hidden strategy the day after, it will be necessary to recreate it before submitting an order. Note that this “re-creation” will return the same AMR as before, as it will merely unhide the strategy.

### 7.1.2 How standing data are organised within the Trading Host

Instruments on Euronext derivatives are, as a first criteria, identified by an **Exchange Code**. Exchange Codes are used to differentiate contracts between instrument types and Euronext derivatives markets.

The list of defined Exchange Codes within the Trading Host is as follows:

Exchange Code	Name	Description
<b>A</b>	Euronext Amsterdam - Equity Derivatives	Amsterdam Individual Equity Option Contracts
<b>B</b>	Euronext Brussels - Equity Derivatives	Brussels Individual Equity Option contracts
<b>C</b>	Euronext Paris - Cash Underlying	Underlying Cash products for French IEOs
<b>D</b>	Euronext Brussels - Cash Underlying	Underlying cash instruments for Brussels IEOs
<b>F</b>	Euronext Brussels - Index Derivatives	Brussels Index Futures and Option contracts
<b>G</b>	Euronext Amsterdam - Cash Underlying	Underlying Cash instruments for Dutch IEOs
<b>H</b>	Euronext Lisbon - Cash Underlying	Underlying cash instruments for Lisbon Stock Futures contracts
<b>J</b>	Euronext Paris - Index Derivatives	French Futures and Option Index contracts
<b>K</b>	Euronext Amsterdam - Index Derivatives	Amsterdam Index Futures and Option contracts
<b>M</b>	Euronext Lisbon - Index Derivatives	Lisbon Index Futures Contracts
<b>P</b>	Euronext Paris - Equity Derivatives	French Individual Equity Option Contracts
<b>R</b>	Euronext Amsterdam - Commodity Derivatives	Amsterdam Commodity Products
<b>S</b>	Euronext Lisbon- Equity Derivatives	Lisbon Individual Stock Futures contracts
<b>Y</b>	Euronext Paris - Commodity Derivatives	Paris Futures and Options Commodity contracts
<b>Z</b>	Euronext Amsterdam – Currency Derivatives	Amsterdam Currency Futures and Options Contracts

### 7.1.3 Identification of instruments

All instruments on Euronext Derivatives, outright contracts as well as strategies, are given a unique reference: the **Automated Market Reference**, so called AMR. AMRs refer to a tradable or listed instrument, i.e. a Futures outright contract, a strategy, a Cash underlying leg.

#### Note

Symbol Index are not used in Euronext Derivatives markets.

The AMR is a 15-character alpha-numeric string where the Symbol index is a numeric value between 1 and 4,294,967,295.

The AMR is the responsibility of Euronext and is subject to change with minimal notice. Application developers are strongly advised **NOT** to use this field as anything but a unique identifier for each possible outright contract or strategy. Developers should **NOT** try to recreate an AMR upon instrument characteristics.

In the FIXML file, the AMR is provided in the ID field at the `<Series>` level. In the Binary and FIX interfaces, the AMR is contained in the `SecurityId` field.

- **The Security Group**

In addition to the AMR, developers must pay attention to the **SecurityGroup** that must be used in place of the AMR in some order entry messages. The Security Group that is a 5 char string is returned in the FIXML files, at the `<Contract>` level, in the `SecGrp` field.

The Security Group is the concatenation of the `ExchangeCode`, the `GenericContract` and the `PhysicalCommodityCode`. As an example, the Security Group for the AEX Index Future contract is "FTI". The corresponding output in the FIXML file is shown below:

```
<Contract SecGrp="FTI" LogSym="FTI" Desc="FUTURE AEX INDEX" Exch="XEUE"
RndLot="200" Ccy="EUR" MinPxIncr=".01" PxQteMeth="STD" SettlMeth="P" FlexInd="N">
.....
<Expiry ExpiryDt="20140300" MMYFmt="0" MMY="201403" FirstTrdDt="20130318"
LastTrdDt="20140321">
  <NestedAttr Typ="26" Val="5" />
  <NestedAttr Typ="109" Val="100" />
  <Series CFI="FFXPSX" RndLot="200" Src="8" ID="KFFTI140300000F">
    <InstrCd AltIDSrc="X" AltID="00000000000000000000" />
  </Series>
```

#### Important note relative to the Security Group:

The Security Group is not disseminated in XDP messages 722 and 732. Outright standing data and strategies standing data messages contain the `ExchangeCode`, the `ProductCode` and the `ContractType` fields that developers are recommended to use to determine the SecurityGroup of the instrument.

- **The CFI Code**

Euronext Derivatives instruments are also classified using the CFI Code that is a Classification of Financial instruments using ISO 10962 norm. The CFI Code is disseminated in the FIXML files at the `<Series>` level in the `CFI` field. Taking the example above, the CFI code for the CAC40 Index Future contract is "FFXPSX".

The table on the following page explains how is built the CFI code for the Euronext derivative instruments.

		COL1	COL2	COL3	COL4	COL5	COL6
<b>CFI (at either Series or Underlying level)</b>	Futures	F (future)	Euronext derivatives - Exchange Codes R/Y: C (commodities) Euronext derivatives - Other Exchange Codes: F (non commodities)	Underlying asset: X	Settlement method. C (cash) P (physical) N (non-delivery) else X	S (standardized)	X
	Options	O (option)	Contract type. C (call) P (put) else X NB: at Underlying level will always be X	Exercise type. A (American) E (European) else X	Underlying asset: X	Settlement method. C (cash) P (physical) N (non-delivery) else X	S (standardized)
	Others	X	X	X	X	X	X
<b>Strategy leg CFI</b>	Futures	F (future)	Euronext derivatives - Exchange Codes R/Y: C (commodities) Euronext derivatives - Other Exchange Codes: F (non commodities)	Underlying asset: X	Settlement method. C (cash) P (physical) N (non-delivery) else X	N (non standardized)	X
	Options	O (option)	M (misc)	X	Underlying asset: X	Settlement method. C (cash) P (physical) N (non-delivery) else X	N (non standardized)
	Others	M (misc)	M (misc)	M (misc)	X	X	X

#### 7.1.4 What is the link between the FIXML file and a Market Data Service?

There is one FIXML file per Exchange Code. Thus, the file `nyseliffe_stddata_eqt_P_090315.xml` corresponds to the FIXML file for Exchange Code P (French IEOs) for March 9, 2015.

Developers must pay attention that while XDP standing data are provided per Market Data services, FIXML files are provided by Exchange Code. Thus, developers must pay attention to which FIXML files they need to download depending on the Market Data Services they join. The table below indicates which Exchange Codes are included in each service.

Product Group	Service Type	Exchange Codes
Equity & Index Derivatives	Level 1	A, B, F, J, K, M, P, S
Equity & Index Derivatives	Level 1+2	A, B, F, J, K, M, P, S
Currency Derivatives	Level 1	Z
Currency Derivatives	Level 1+2	Z
Commodity Derivatives	Level 1	R, Y
Commodity Derivatives	Level 1+2	R, Y
Amsterdam - Equity Derivatives	Level 1+2	A
Amsterdam - Index Futures	Level 1+2	K
Amsterdam - Index Options	Level 1+2	K
Paris - Equity Derivatives	Level 1+2	P
Paris - Index Futures	Level 1+2	J
Paris - Index Options	Level 1+2	J
Brussels - Equity & Index Derivatives	Level 1+2	B, F
Lisbon - Equity & Index Derivatives	Level 1+2	M, S
Euronext Equity & Index Derivatives	VAP + Open Interest	A, B, F, J, K, M, P, S
Euronext Currency Derivatives	VAP + Open Interest	Z
Euronext Commodity Derivatives	VAP + Open Interest	R, Y

---

### 7.1.5 What is the content of the FIXML files?

The FIXML files contain the full static standing data for all Euronext instruments. Data are organised from the highest level, i.e. <Exchange Code> to the finest one <Series>.

- **<Exchange Code>** - Name and description of the Euronext Exchange Code.
- **<Contract>** - Characteristics of the contract which are common to all instruments of the same contract (all maturities, all strikes).
- **<Expiry>** - At this level are listed details of all the available expiry months. For Option contracts, standing data provided at this level are common to all series within the same expiry.
- **<Series>** - At this level are provided standing data for a specific instrument.
  - In case of a Futures contract, the call returns standing data about the contract itself.
  - For an Option contract, the function returns information about the whole series, i.e. all Call and Put options listed in the specified expiry month.
- **<Strategies>** - This contains the list and details of the strategies available for the contract (remind that only the non-hidden ones are available in the FIXML file).

---

### 7.1.6 XDP standing data streams

This paragraph contains a set of useful information about standing data messages sent on XDP multicast streams.

- At start of the day, outright standing data messages 722 are sent before strategy standing data messages 732.
- The Security Group is not disseminated in the messages 722 and 732. Instead are disseminated the ExchangeCode, the ProductCode (i.e. the Physical Commodity Code), and the ContractType. Developers are strongly recommended to concatenate those 3 fields to determine to which Security Group the instrument is attached to. Fields must be concatenated as follows:  
`SecurityGroup = ExchangeCode & ContractType & ProductCode`  
 Example: the CAC40 future belongs to the Paris Index exchange 'J' and its contract code is 'FCE'. So, its SecurityGroup is: JFFCE.
- The field NoSecurityIDs field is not used but always equal to 1 meaning that an instrument has only one identifier.
- For all instruments listed in the Central Order Book, in the message 722:
  - SymbolIndex = 0
  - NoSecurityIDs = 1
  - SecurityIDSource = 8
  - SecurityID contains the AMR of the product.

## 7.2 LINK BETWEEN AN OPTION SERIES AND ITS UNDERLYING

The FIXML files provide client applications with a way to link an option with its underlying contract.

### 7.2.1 Individual Equity Options

For Individual Equity Options, the link between an option series and its underlying contract is the ISIN Code. In this case, at the <Contract> level:

- The `Src` field at the contract level is equal to 'P'
- The `ID` field contains the SecurityGroup of the underlying cash instrument
- The `CFI` field contains the CFI code of the underlying stock. Note that it is always equal to "XXXXXX".

**Example:** AIR FRANCE – KLM Equity Option series listed in Amsterdam (AFA class)

AIR FRANCE – KLM stock in the Amsterdam Cash market is defined as follows

- ISIN Code : FR0000031122
- Mnemonic : AFA

As shown below, the FIXML file for AFA Equity Option returns the SecurityGroup code of the stock underlying instrument at the contract level

```
<Contract  SecGrp="AOAFA"  Sym="AFA"  LogSym="AFA"  Desc="AIR  FRANCE"
Exch="XEUE" RndLot="100" Ccy="EUR" MinPxIncr=".01" PxQteMeth="STD" SettlMeth="P"
ExerStyle="1" FlexInd="N" CFI="XXXXXX" Src="P" ID="GSAFA">
  <NestedAttr Typ="25" Val="100" />
  <NestedAttr Typ="101" Val="1" />
  <NestedAttr Typ="102" Val="1" />
  <NestedAttr Typ="103" Val="100" />
  <NestedAttr Typ="104" Val="2" />
  <NestedAttr Typ="110" Val="C" />
  <NestedAttr Typ="120" Val="T" />
  <NestedAttr Typ="121" Val="4" />
```

Client applications can then retrieve standing data about the underlying Cash instrument by looking at the appropriate FIXML standing data files. With the exception of the UK market, underlying Cash instruments are listed in the following Exchange Codes:

- Paris Cash Market: 'C'
- Amsterdam Cash Market: 'G'
- Brussels Cash Market: 'D'
- Lisbon Cash Market: 'H'

Taking the example above, information about the Air France stock can be retrieved in the nyseliffe\_stddata\_eqt\_G FIXML file:

```

Contract SecGrp="GSAFA" Sym="AFA" LogSym="AFA" Desc="AIR FRANCE" RndLot="1"
Ccy="EUR" MinPxIncr=".001" PxQteMeth="STD" SettlMeth="P" FlexInd="N" Src="4"
ID="FR0000031122">
  <NestedAttr Typ="25" Val="10000" />
  <NestedAttr Typ="26" Val="1" />
  <NestedAttr Typ="101" Val="1" />
  <NestedAttr Typ="102" Val="1" />
  <NestedAttr Typ="120" Val="L" />
  <NestedAttr Typ="121" Val="0" />
<Expiry>
  <Series CFI="XXXXXX" RndLot="1" Src="8" ID="GSAFA000000000N" />
</Expiry>
</Contract>

```

The Src and ID fields at the <Contract> level provide client apps with the ISIN code of the underlying stock of the Individual Equity Options, “FR0000031122” in the example above.

## 7.2.2 Index Options

For Index Options, the link between an option series and its underlying contract is the SecGrp of the Index Cash product.

At the <Contract> level:

- The Src field at the contract level is equal to ‘P’
- The ID field contains the Security Group of the Cash Index product
- The CFI code is always equal to ‘XXXXXX’

See below the output for the Option contract on the AEX Index. In the FIXML file for Exchange Code ‘K’:

```

<Contract SecGrp="KOAEX" LogSym="AEX" Desc="AEX-INDEX" Exch="XEUE"
RndLot="100" Ccy="EUR" MinPxIncr=".05" PxQteMeth="STD" SettlMeth="C" ExerStyle="0"
FlexInd="N" CFI="XXXXXX" Src="P" ID="GIAEX">
  <NestedAttr Typ="25" Val="100" />
  <NestedAttr Typ="101" Val="5" />
  <NestedAttr Typ="102" Val="5" />
  <NestedAttr Typ="103" Val="10" />
  <NestedAttr Typ="104" Val="1" />
  <NestedAttr Typ="110" Val="C" />
  <NestedAttr Typ="120" Val="T" />
  <NestedAttr Typ="121" Val="4" />
  <OrdTypeRules OrdTyp="1" />
  .....
  <WTradeTypeRules WTrdTyp="6" />
  <WTradeTypeRules WTrdTyp="5" />
  <SecSubTypes SubTyp="A" />
  .....

```



```

<Expiry ExpiryDt="20170700" MMYFmt="0" MMY="201707" FirstTrdDt="20170419"
LastTrdDt="20170716">
  <NestedAttr Typ="26" Val="5" />
  <NestedAttr Typ="109" Val="500" />
  <Series CFI="OCEXCS" StrkPx="2000" RndLot="100" Src="8"
ID="KOAEX170702000C">
    <InstrCd AltIDSrc="X" AltID="0000000000000000227418" />
    <InstrCd AltIDSrc="Y" AltID="EUNL02274187" />
  </Series>

```

As for IEOs, client applications can retrieve or refer to standing data of the underlying instrument identified by its Security Group by looking at the appropriate FIXML file. In the example above, developers must refer to the FIXML file for Exchange Code 'G'.

```

<Contract SecGrp="GIAEX" Sym="AEX" LogSym="AEX" Desc="AEX-INDEX" RndLot="1"
Ccy="EUR" MinPxIncr=".5" PxQteMeth="STD" SettlMeth="P" FlexInd="N" Src="4"
ID="NL00000000107">
  <NestedAttr Typ="25" Val="100" />
  <NestedAttr Typ="26" Val="1" />
  <NestedAttr Typ="101" Val="1" />
  <NestedAttr Typ="102" Val="1" />
  <NestedAttr Typ="120" Val="L" />
  <NestedAttr Typ="121" Val="0" />
  <Expiry>
    <Series CFI="XXXXXX" RndLot="1" Src="8" ID="GIAEX0000000000N" />
  </Expiry>
</Contract>

```

### 7.2.3 Options on Futures

For Options on Futures contracts, the link between an option series and its underlying contract is the associated futures expiry month. In this case, at the <Contract> level:

- The `Src` field at the contract level is equal to 'P',
- The `ID` field contains the Security Group of the underlying Future contract
- The `CFI` field contains the CFI code of the Underlying Future contract

Each Option series of the same expiry has an associated underlying Future expiry month that is specified in the `UndMMYFmt` and `UndMMY` fields at the <Expiry> level.

The example below shows the underlying Futures expiry associated to the “Rapeseed / Colza” AUG15 Option contract.

```
<Contract SecGrp="ECO" LogSym="ECO" Desc="Rapeseed / Colza" Exch="XMAT"
RndLot="1" Ccy="EUR" MinPxIncr="12.5" PxQteMeth="STD" SettlMeth="P" ExerStyle=""
FlexInd="N" CFI="FCXPSX" Src="P" ID="">
  <NestedAttr Typ="25" Val="100" />
  <NestedAttr Typ="101" Val="25" />
  <NestedAttr Typ="102" Val="25" />
  <NestedAttr Typ="120" Val="T" />
  <NestedAttr Typ="121" Val="0" />

  <Expiry ExpiryDt="20150800" MMYFmt="0" MMY="201508" FirstTrdDt="20130201"
LastTrdDt="20150731" UndMMYFmt="0" UndMMY="201508">
  <NestedAttr Typ="26" Val="25" />
```

### 7.3 DETERMINATION OF EXERCISE AND TRADE PRICES

All prices within the Trading Host are in system ticks and integer values. Members must therefore apply some calculations to convert them into decimal values.

#### 7.3.1 How to determine exercise prices

Exercise prices are disseminated in the standing data as integers, with no decimal locator. For some contracts, exercise prices could also be disseminated as a combination of real values and ticks.

In order to calculate exercise prices, developers must use the two following fields included in the FIXML files, at the <Contract> level:

- 1- The **Strike Price Denominator** provided in the `NestedAttr = "103"` field
- 2- The **Strike Price Decimal Locator** provided in the `NestedAttrib = "104"` field

The Decimal Locator splits the exercise price into two parts: the "points" and "ticks" parts of the price. This is to handle potentially, if required, fractional prices.

- The “Points” part is the integer part of the strike;
- The “Ticks” part is divided by the strike denominator to get the fractional part.

Therefore, the general formula to determine the exercise prices is the following one:

$$\text{Strike Price} = P + \frac{T}{D}$$

$$P = \text{int}\left(\frac{E}{10^L}\right)$$

$$T = E \bmod(10^L)$$

- E = Exercise Price
- L = Decimal Locator
- D = Exercise Price Denominator
- P = “Points” value of the Exercise Price (truncated, not rounded)
- T = “Ticks” value of the Exercise Price

This formula is illustrated in three examples below.

#### **Example 1**

- Exercise Price = 649
  - Strike Denominator = 10
  - Decimal Locator = 1
  - $P = \text{int}(649 / 10^1) = 64$
  - $T = 649 \bmod(10^1) = 9$
- ➔ Exercise Price =  $64 + 9/10 = \mathbf{64.9}$

#### **Example 2**

- Exercise Price = 641
  - Strike Denominator = 4
  - Decimal Locator = 1
  - $P = \text{int}(641 / 10^1) = 64$
  - $T = 641 \bmod(10^1) = 1$
- ➔ Exercise Price =  $64 + 1/4 = \mathbf{64.25}$

#### **Example 3**

- Exercise Price = 6425
  - Strike Denominator = 100
  - Decimal Locator = 2
  - $P = \text{int}(6425 / 10^2) = 64$
  - $T = 6425 \bmod(10^2) = 25$
- ➔ Exercise Price =  $64 + 25/100 = \mathbf{64.25}$

#### **Note**

The examples above outline that where Exercise Price Denominator =  $10^{\text{Decimal Locator}}$ , the result is the same as if the price had been simply divided by the strike denominator.

### 7.3.2 How to determine trade prices

All order and trade prices in UTP must be sent or are broadcast as Integer values. In order to determine the exact prices, developers must use the following fields disseminated in the FIXML files:

- The **Instrument Denominator** provided at the <Contract> level, in the NestedAttr = "25" field
- The **Instrument Numerator** provided at the <Expiry> level, in the NestedAttr = "26" field

**Note:**

Developers should pay attention that the Instrument Numerator (NestedAttr = "26") at the <Contract> level will be never be populated.

### 7.4 PREMIUM BASED TICK SIZES

The Premium Based Tick Size functionality allows orders to be entered at smaller minimum price movements when the option premium moves below a specified threshold. The tick size below the premium threshold must always be smaller than that above, and the tick size above must always be a multiple of that below.

The different finer and greater tick sizes as well as the premium threshold are included in the FIXML standing data files.

- The finer tick size numerator corresponds to the Instrument Numerator provided at the <Expiry> level in the NestedAttr = "26" field
- The greater tick size numerator to apply above the threshold is given at the <Expiry> level in the NestedAttr = "105" field
- The Premium threshold numerator is provided at the <Expiry> level in the NestedAttr = "106" field
- The Instrument Denominator to apply is provided at the <Contract> level in the NestAttr = "25" field.

The formula to calculate the premium threshold, the finer and greater tick sizes are given below:

$$Finer\_Tick = \frac{Instrument\_Numerator}{Instrument\_Denominator}$$

$$Greater\_Tick = \frac{Premium\_Pricing\_Numerator}{Instrument\_Denominator}$$

$$Premium\_Threshold = \frac{Premium\_Pricing\_Threshold}{Instrument\_Denominator}$$

- **Example**

The example is based on the SEP15 PHILIPS option series (SecurityGroup = AOPHI).

The Tick Size Denominator is provided in the NestedAttr "25" field within the <Contract> sub-block while remaining data is provided at the <Expiry> level.

```
<Contract SecGrp="AOPHI" Sym="PHI" LogSym="PHI" Desc="PHILIPS" Exch="XEUE"
  RndLot="100" Ccy="EUR" MinPxIncr=".01" PxQteMeth="STD" SettlMeth="P"
  ExerStyle="1" FlexInd="N" CFI="XXXXXX" Src="P" ID="GSPHI">
  <NestedAttr Typ="25" Val="100" />
  <NestedAttr Typ="101" Val="1" />
  <NestedAttr Typ="102" Val="1" />
  <NestedAttr Typ="103" Val="100" />
  <NestedAttr Typ="104" Val="2" />
  <NestedAttr Typ="110" Val="C" />
  <NestedAttr Typ="120" Val="T" />
  <NestedAttr Typ="121" Val="4" />
  <OrdTypeRules OrdTyp="1" />
  <OrdTypeRules OrdTyp="2" />
  <OrdTypeRules OrdTyp="3" />
  <OrdTypeRules OrdTyp="4" />
  <OrdTypeRules OrdTyp="W" />
  <WTradeTypeRules WTrdTyp="6" />
  <WTradeTypeRules WTrdTyp="5" />
  <SecSubTypes SubTyp="A" />
  <SecSubTypes SubTyp="B" />
  .....
  <Expiry ExpireDt="20150900" MMYFmt="0" MMY="201509" FirstTrdDt="20140922"
    LastTrdDt="20150918">
    <NestedAttr Typ="26" Val="1" />
    <NestedAttr Typ="105" Val="5" />
    <NestedAttr Typ="106" Val="20" />
    <NestedAttr Typ="109" Val="500" />
    .....
    <Series CFI="OCAXPS" StrkPx="2800" RndLot="100" Src="8"
      ID="AOPHI150902800C">
      <InstrCd AltIDSrc="X" AltID="00000000000000314728" />
      <InstrCd AltIDSrc="Y" AltID="EUNL03147283" />
    </Series>
    <Series CFI="OPAXPS" StrkPx="2800" RndLot="100" Src="8"
      ID="AOPHI150902800P">
      <InstrCd AltIDSrc="X" AltID="00000000000000314729" />
      <InstrCd AltIDSrc="Y" AltID="EUNL03147291" />
    </Series>
```

From the standing data above, the premium threshold, finer and greater tick sizes are equal to:

- ➔ Finer Tick =  $1/100=0.01$
- ➔ Greater Tick =  $5/100=0.05$
- ➔ Premium Threshold =  $20/100=0.2$

Therefore, any orders below an option premium of 0.2 can be submitted with a 0.01 tick size granularity, and any orders above that premium can only be submitted with a 0.05 tick size.

This is illustrated below. Let us assume that the price for the Call 2800 PHI SEP15 is about 0.20 and is about 1.50 for the Put 2800 PHI SEP2015.

A trader submits an order for the Call at 0.19.

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 0B : 11
ClOrdID         : 00 21 A9 31 : 2206001
Price          : 00 00 00 13 : 19
OrderQty        : 00 00 00 64 : 100
.....
SecurityIDSource : 38             : 8
SecurityID       :                : (AOPHI150902800C)

```

The order at 0.19 is successfully accepted by the Trading Host:

```

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 04 : 4
OrderID         : 01 04 19 00 00 00 70 E4 : 73210981735493860
ClOrdID         : 00 21 A9 31 : 2206001

```

Then, the trader submits two orders for the Put: the first at 1.55 and the second one at 1.52. As shown below, the first order is accepted while the second one is rejected with a Invalid Price (318) OrdRejReason and a ORDER\_PRICE\_TICK\_SIZE\_INVALID (4194462) ReturnCode.

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 0D : 13
ClOrdID         : 00 21 A9 35 : 2206005
Price          : 00 00 00 9B : 155
OrderQty        : 00 00 00 64 : 100
.....
SecurityIDSource : 38             : 8
SecurityID       :                : (AOPHI150902800P)

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 06 : 6
OrderID         : 01 04 19 00 00 00 70 E5 : 73210981735493861
ClOrdID         : 00 21 A9 35 : 2206005

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 0E : 14
ClOrdID         : 00 21 A9 37 : 2206007
Price          : 00 00 00 98 : 152
OrderQty        : 00 00 00 64 : 100
.....
SecurityIDSource : 38             : 8
SecurityID       :                : (AOPHI150902800P)

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 07 : 7
OrderID         : 00 00 00 00 00 00 00 00 : 0
TradeID         : 00 00 00 00 00 00 00 00 : 0

```

```

ExecID           : 01 00 00 00 00 01 0B 34 : 72057594037996340
ClOrdID          : 00 21 A9 37 : 2206007
OrigClOrdID      : 00 00 00 00 : 0
TransactTime     : 01 BF EC A0 : 29355168
Price            : 00 00 00 98 : 152
ReturnCode       : 00 40 00 9E : 4194462
OrderQty         : 00 00 00 64 : 100
LeavesQty        : 00 00 00 64 : 100
OrdRejReason     : 01 3E : 318
LastRptRequested : 59 : Y
Text             : : ()
SecurityIDSource : 38 : 8
SecurityID       : : (AOPHI150902800P)
OrdStatus        : 38 : 8
ExecType         : 38 : 8

```

## 7.5 CREATE AND HANDLE STRATEGIES

When creating a new strategy using the Security Definition Request ('c') message, client applications must avoid providing a front-end functionality that will allow a trader to create a number of strategies simultaneously, more easily than creating a single one. This ensures that no more strategies are created than are required.

For example, if the application incorporates a 'spread matrix' facility, then care should be taken not to provide a 'create all' button that will allow creation of a number of strategies when the trader is interested in only a couple of these strategies.

Also, client applications should avoid allowing other operations such as Quote Request message ('R') to be 'bundled' with a Security Definition Request. This ensures that other participants have enough time to subscribe to the newly created instrument.

If the strategy is successfully created:

- Client applications receive a `Security Definition (d)` message that contains the AMR of the strategy. Note that if a client app submits a creation for an already existing strategy, then the Trading Engine returns the AMR of the previously created one. This AMR must then be used to invoke any trading or order handling for the new strategy.
- All market participants receive a `Strategy Standing Data update (732)` via the XDP multicast standing data streams. This notification is followed by a `Market Status (752)` message in XDP indicating that the strategy is open and available for trading.

Developers must pay attention to the following. The 732 and 752 messages are sent to separate multicast streams, respectively the real-time **standing data** streams and the real-time **market data** streams. The two streams being not synchronized, it might be the case the market status message is received by the client application before the standing data update message. Developers must therefore listen to the two different channels but make sure their application processes the strategy standing data update before the market status one.

If the strategy is not successfully created, client applications should pay attention to the `ReturnCode` returned. There are a number of codes related to strategy creation. If no specific error code is returned, then a `STATUS_ERROR` is returned.

The leg details for defining the strategy must reflect a long position otherwise the Trading Host will return an error. The `Security Definition Request` accepts only one strategy instrument per message. When any leg of a strategy is in an outright market which has been newly created and does not yet have any prices, the request will be rejected with a `NO_PRICE_IN_OUTRIGHT_MARKET` status.

### 7.5.1 Examples of Security Definition Request messages

- **Creation of a strategy on a Futures contract**

The example below corresponds to the creation of an AEX Index futures MAR15 / JUN15 / SEP15 Butterfly. Such a strategy is defined as buying 1 Future in the first expiry, selling 2 Futures in the far expiry, buying 1 Futures in a far more expiry. Expiries must be quarterly consecutive ones.

```

MsgId           : 01 F1           : 497 [c]
iLen            : 00 98           : 152
lSeqNum         : 00 00 00 0B     : 11
SecurityReqID   : 00 00 04 57     : 1111
SecurityRequestType : 04           : 4
SecurityIDSource : 50             : P
SecurityID      :                 : (KFFFTI)
SecuritySubType :                 : (B)
NoLegs          : 03             : 3
LegRatioQty     : 00 00 00 01     : 1
LegPrice        : 00 00 00 00     : 0
LegStrikePrice  : 00 00 00 00     : 0
LegMaturityMonthYear: 00 03 11 8F : 201503
LegSecurityIDSource : 50           : P
LegSecurityID    :                 : (KFFFTI)
LegSecurityType  :                 : (FUT)
LegPutOrCall    : 00             :
LegSide         : 31             : 1
Filler[2]       :                :
LegRatioQty     : 00 00 00 02     : 2
LegPrice        : 00 00 00 00     : 0
LegStrikePrice  : 00 00 00 00     : 0
LegMaturityMonthYear: 00 03 11 92 : 201506
LegSecurityIDSource : 50           : P
LegSecurityID    :                 : (KFFFTI)
LegSecurityType  :                 : (FUT)
LegPutOrCall    : 00             :
LegSide         : 32             : 2
Filler[2]       :                :
LegRatioQty     : 00 00 00 01     : 1
LegPrice        : 00 00 00 00     : 0
LegStrikePrice  : 00 00 00 00     : 0
LegMaturityMonthYear: 00 03 11 95 : 201509
LegSecurityIDSource : 50           : P
LegSecurityID    :                 : (KFFFTI)
LegSecurityType  :                 : (FUT)
LegPutOrCall    : 00             :
LegSide         : 31             : 1
Filler[2]       :                :

```



```

MsgId           : 01 E1           : 481 [d]
iLen            : 00 4A           : 74
lSeqNum         : 00 00 00 06 : 6
SecurityReqID   : 00 00 04 57 : 1111
ReturnCode      : 00 40 00 01 : 4194305
RejectReasonCode : 00 00           : 0
Text            :                  : ()
SecurityIDSource : 38             : 8
SecurityID      :                  : (KFFTI03B006977S)

```

On the real time standing data stream over XDP a **Strategy Standing Data (732)** message indicating that the strategy has been created is sent:

PKT,799, 1795,210715756,102,8,1

```

MSG,MsgType 732, SourceTime 288545139, SymbolIndex 0, ExchangeCode 'L', ProductCode 'I ',
ExpiryDate 20150300, ContractType 'F', StrategyCode 'B', NoSecurityIDs 1, SecurityIDSource
8, SecurityID 'KFFTI03B006977S', NumLegs 3,

LegSymbolIndex_1 0, LegPrice_1 0, LegRatio_1 2, LegRatioScaleCode_1 0, LegBuySell_1 'B',
LegNoSecurityIDs_1 1, LegSecurityIDSource_1 8, LegSecurityID_1 'KFFTI150300000F',

LegSecurityIndex_2 0, LegPrice_2 0, LegRatio_2 2, LegRatioScaleCode_2 0, LegBuySell_2 'S',
LegNoSecurityIDs_2 1, LegSecurityIDSource_2 8, LegSecurityID_2 'KFFTI150600000F'

LegSecurityIndex_3 0, LegPrice_3 0, LegRatio_3 1, LegRatioScaleCode_3 0, LegBuySell_3 'B',
LegNoSecurityIDs_3 1, LegSecurityIDSource_3 8, LegSecurityID_3 'KFFTI150900000F'

```

**Note:**

Strategies should always be created in the **Buy** order. Any strategy created in the **Sell** order will be rejected with a STATUS ERROR reject message.

- **Creation of a delta neutral strategy**

When creating a delta neutral strategy, client applications should pay attention to the following rules:

- 1- The delta should be expressed as an Integer value between 0 and 1000 and specified in the `LegRatio` field of the hedge leg. Therefore, a delta of 50% must be entered as 500.
- 2- The price of the hedge instrument must be specified in the `LegPrice` field.

The example below corresponds to the creation of a 3400 Call Volatility AEX AUG15 hedged against the AEX Futures contract at a price of 338.5 with a Delta of 10%.

```

MsgId           : 01 F1           : 497 [c]
iLen            : 00 70           : 112
lSeqNum         : 00 00 00 F4 : 244
SecurityReqID   : 00 0F 59 B2 : 1006002
SecurityRequestType : 04           : 4
SecurityIDSource : 50             : P
SecurityID      :                  : (KOAEX)
SecuritySubType  :                  : (V)
NoLegs          : 02             : 2
LegRatioQty     : 00 00 00 01 : 1
LegPrice        : 00 00 00 00 : 0
LegStrikePrice  : 00 00 0D 48 : 3400

```

```

LegMaturityMonthYear: 00 03 11 30 : 201508
LegSecurityIDSource : 50 : P
LegSecurityID : : (KOAEX)
LegSecurityType : : (OPT)
LegPutOrCall : 31 : 1
LegSide : 31 : 1
Filler[2] : :
LegRatioQty : 00 00 00 64 : 100
LegPrice : 00 00 0D 39 : 3385
LegStrikePrice : 00 00 00 00 : 0
LegMaturityMonthYear: 00 03 11 30 : 201508
LegSecurityIDSource : 50 : P
LegSecurityID : : (KFFFTI)
LegSecurityType : : (FUT)
LegPutOrCall : 00 :
LegSide : 32 : 2
Filler[2] : :

MsgId : 01 E1 : 481 [d]
iLen : 00 4A : 74
lSeqNum : 00 00 00 2F : 47
SecurityReqID : 00 0F 59 B2 : 1006002
ReturnCode : 00 40 00 01 : 4194305
RejectReasonCode : 00 00 : 0
Text : : ()
SecurityIDSource : 38 : 8
SecurityID : : (KOAEX02V000169S)

```

The following messages are then sent over XDP:

On the real time Market Data stream, a Market Status (752) message notifying that the strategy is in open mode and available:

```

PKT,799,4454,210715756,2,8,1
MSG,MsgType 752, SourceTime 210689552, SymbolIndex 0,
SecurityIDSource 8, SecurityID 'KOAEX02V000169S', NoUpdates 7, MarketMode
32,11,7,23,4,42,14

```

On the real time standing data stream a Strategy Standing Data (732) message indicating that the strategy has been created:

```

PKT,799, 1795,210715756,102,8,1
MSG,MsgType 732, SourceTime 210689552, SymbolIndex 0, ExchangeCode 'K',
ProductCode 'AEX', ExpiryDate 20170800, ContractType 0, StrategyCode 'V',
NoSecurityIDs 1, SecurityIDSource 8, SecurityID 'KOAEX02V000169S', NumLegs 2,
LegSymbolIndex_1 0, LegPrice_1 3400, LegRatio_1 1, LegRatioScaleCode_1 0,
LegBuySell_1 'B', LegNoSecurityIDs_1 1, LegSecurityIDSource_1 8, LegSecurityID_1
'KOAEX170803400C',
LegSecurityIndex_2 0, LegPrice_2 3385, LegRatio_2 100, LegRatioScaleCode_2 0,
LegBuySell_2 'S', LegNoSecurityIDs_2 1, LegSecurityIDSource_2 8, LegSecurityID_2
'KFFFTI170800000F'

```

From this output, developers should pay attention that when the LegRatioScaleCode field is equal to 0. This field is reserved for future use.

- **Creation of an EFP strategy**

An Exchange For Physicals (EFP) strategy is constituted of two “legs” of opposite direction. The first leg is a future on an index (CAC40 – French top 40 companies or AEX - Dutch top 25 companies) and the second “leg” is in reality constituted of all the components of the index. For example an EFP on the CAC40 will have 41 legs– one leg corresponding to the CAC40 future and the other legs are the 40 components of the CAC40 called “Cash legs”.

Members are able to enter an EFP order directly on the UTP-DE Central Order Book using the standard order entry message with the dedicated EFP strategy AMR (Automated Market Reference). The order information required must include:

- The price of the strategy: the EFP Price is to be expressed in index points and is the difference between the index futures price and the index spot price. This price can be either positive, negative or zero. The number of decimal places will be 1 for the CAC40 EFP and 2 for the AEX EFP.
- The quantity of the EFP: the quantity is in units and corresponds to the quantity of the EFP to be traded. It will be in the same units as the index future quantity.

**Note:**

An example of EFP strategy creation will be provided in a future version of this document.

- **Creation of an ICS strategy**

An Inter Commodity Spread (ICS) strategy is constituted of two “legs” that are two different contracts (for example the Premium Milling Wheat future and the Milling Wheat Future or the CAC40 week #2 future and the CAC40 week #4 future).

The order entry process is the same as a classical strategy, except for the Weekly Futures (week #x – week #y of CAC40 or AEX indices futures) where a contract is associated to each combination (week #1 – week #2 of CAC40 on the same month, week #4 – week #1 of AEX on different months...).

**Note:**

Examples of ICS (including Weekly Futures) strategy creation will be provided in a future version of this document.

## 7.5.2 Specific rules applying to Cash Underlying contracts and delta neutral trades

### Note:

Developers must pay attention to the rules presented in this paragraph that are relative to the way prices are disseminated in the Euronext Cash markets and the implications for developers when creating delta-neutral strategies.

Prices for cash underlying contracts in all the Continental Derivatives markets of Euronext, i.e. Amsterdam, Paris, Brussels and Lisbon are provided with three (3) decimal prices. In the FIXML standing data files, prices for underlying stocks for the Individual Equity Options listed in the Amsterdam, Paris and Brussels markets are also represented using 3 decimal places.

This covers both underlying price dissemination for Amsterdam Cash products but also the delta neutral leg definition.

In terms of standing data, the tick size denominator, `NestedAttr = "25"`, returned for instruments on Cash Underlying Exchange Codes , i.e. G, C and D are equal to 10000. See below an example from the Amsterdam market.

```
<Contract SecGrp="GSING" LogSym="ING" Desc="ING Groep" RndLot="1" Ccy="EUR"
MinPxIncr=".001" PxQteMeth="STD" SettlMeth="P" FlexInd="N">
  <NestedAttr Typ="25" Val="10000" />
  <NestedAttr Typ="26" Val="1" />
  <NestedAttr Typ="101" Val="1" />
  <NestedAttr Typ="102" Val="1" />
  <NestedAttr Typ="120" Val="T" />
  <NestedAttr Typ="121" Val="0" />
<Expiry>
  <Series CFI="XXXXXX" RndLot="1" Src="8" ID="GSING000000000N" />
</Expiry>
</Contract>
```

### Note:

Developers must pay attention that the fact that Cash prices are represented with 3 decimals. This has no impact on the granularity of option exercise / strike prices.

- **Rule for creating delta neutral trades**

When creating a delta neutral trade in the Euronext Derivatives markets, the granularity of the underlying leg must be 3 decimal places (€0.001). Therefore, the price of the underlying specified in the `LegPrice` field should be expressed using the Tick Size Denominator of the underlying instrument.

This is represented in the following example. A user wants to create a Call Volatility on ING / JUL15 / 7.60 hedged against the stock at 7.245 with a delta of 95%. The `Security Definition Request` message should be built the following way:

```
ThreadID          :          : 1242605888
MsgId             : 01 F1     : 497 [c]
iLen              : 00 70     : 112
lSeqNum           : 00 00 00 03 : 3
SecurityReqID     : 00 00 17 73 : 6003
SecurityRequestType : 04         : 4
```

```

SecurityIDSource      : 50          : P
SecurityID            :              : (AOING)
SecuritySubType       :              : (V)
NoLegs                : 02          : 2
LegRatioQty           : 00 00 00 01 : 1
LegPrice              : 00 00 00 00 : 0
LegStrikePrice        : 00 00 02 F8  : 760
LegMaturityMonthYear : 00 03 11 2F  : 201507
LegSecurityIDSource   : 50          : P
LegSecurityID         :              : (AOING)
LegSecurityType       :              : (OPT)
LegPutOrCall          : 31          : 1
LegSide               : 31          : 1
Filler[2]             :              :
LegRatioQty           : 00 00 03 B6  : 950
LegPrice              : 00 00 1C 4D : 72450
LegStrikePrice        : 00 00 00 00 : 0
LegMaturityMonthYear : 00 00 00 00 : 0
LegSecurityIDSource   : 50          : P
LegSecurityID         :              : (GSING)
LegSecurityType       :              : (CASH)
LegPutOrCall          : 00          :
LegSide               : 32          : 2
Filler[2]             :              :

MsgId                 : 01 E1        : 481 [d]
iLen                  : 00 4A        : 74
lSeqNum               : 00 00 00 05  : 5
SecurityReqID         : 00 00 17 73  : 6003
ReturnCode            : 00 40 00 01  : 4194305
RejectReasonCode      : 00 00        : 0
Text                  :              : ()
SecurityIDSource      : 38          : 8
SecurityID            :              : (AOING02V000140S)
ThreadID              :              : 1242605888

```

PKT,799,143,118184863,121,8,1

```

MSG, MsgType 732, SourceTime 118188557, SymbolIndex 0, ExchangeCode 'A', ProductCode
'ING', ExpiryDate 20170700, ContractType 0, StrategyCode 'V', NoSecurityIDs 1,
SecurityIDSource 8, SecurityID 'AOING02V000140S', NumLegs 2,
LegSymbolIndex_1 0, LegPrice_1 760, LegRatio_1 1, LegRatioScaleCode_1 0, LegBuySell_1
'B', LegNoSecurityIDs_1 1, LegSecurityIDSource_1 8, LegSecurityID_1 'AOING170700760C',
LegSecurityIndex_2 0, LegPrice_2 7245, LegRatio_2 950, LegRatioScaleCode_2 0,
LegBuySell_2 'S', LegNoSecurityIDs_2 1, LegSecurityIDSource_2 8, LegSecurityID_2
'GSING000000000N'

```

PKT,799,284,118184863,21,8,1

```

MSG, MsgType 752, SourceTime 118188557, SymbolIndex 0, SecurityIDSource 8, SecurityID
'AOING02V000140S', NoUpdates 7, MarketMode 32,11,7,23,4,42,14

```

## 8. MARKET INFORMATION

### 8.1 IMPLIED PRICING

An implied price is a price for a market calculated by the Trading Host using price data from other markets. UTP-DE differentiates 'Implied In' from 'Implied Out' prices.

#### 8.1.1 Implied in

'Implied In' are prices in a strategy market derived from prices in the associated outright markets. The example below shows Implied In prices in a Butterfly contract.

DEC15		MAR16		JUN16	
BID	ASK	BID	ASK	BID	ASK
95.750 [100]	95.800 [70]	96.015 [150]	96.115 [80]	96.850 [20]	96.875 [200]

Butterfly DEC15 / MAR16 / JUN16	
BID	ASK
0.370 [20]	0.645 [70]

The DEC bid can combine with the MAR offer and the JUN bid to generate an implied bid 'in' the butterfly; The DEC offer can combine with the MAR bid and the JUN offer to generate an implied offer 'in' the butterfly.

Implied In prices are calculated by the Trading Host but not disseminated via XDP. Hence, the application developer should ensure that their application calculates the 'implied in' prices and displays them to the front-end user in the appropriate manner.

Implied In trading is supported on the following strategies on Futures contracts:

- Calendar Spread
- Butterfly
- Condor
- Strips - where the volume is quoted in terms of a 1 lot on each leg. Any other volume ratio will not generate implied prices
- Packs
- Bundles

And the following strategies on Options :

- Call and Put spread
- Call and Put Calendar spread
- Call and Put Diagonal Calendar spread
- Strangle
- Straddle
- Call and Put Butterfly
- Call and Put Ladder
- 2:1 Ratio Call and Put Spread

### 8.1.2 Implied out

'Implied Out' are prices in an outright market derived from prices in one strategy and prices in associated legs of the strategy.

DEC15		MAR16	
BID	ASK	BID	ASK
95.650 [100]		95.445 [50]	

Spread DEC15 / MAR16	
BID	ASK
0.200 [75]	0.205 [50]

The DEC bid can combine with the DEC/MAR ask spread to generate an implied 'out' MAR bid.

Although the Trading Host calculates 'implied out' prices, it only transmits the best 'implied out' prices along with full explicit pricing.

Implied Out trading is supported on the following strategies on Futures contracts:

- Calendar Spread

And the following strategies on Options :

- Call and Put spread
- Call and Put Calendar spread
- Call and Put Diagonal Calendar spread
- Strangle
- Straddle

## 8.2 PACK AND BUNDLE STRATEGIES

A **Pack** comprises four consecutive quarterly delivery months in the same colour grouping. As with a Strip, the trader would buy each leg of the strategy or sell each leg of the strategy to go long or short of the Pack. Packs are more standardised than Strips in that each constituent leg must be for the same volume, e.g. a long 5 lots White Pack position consists of 5 lots long White Dec, 5 lots long White Mar, 5 lots long White Jun and 5 lots long White Sep.

A **Bundle** comprises consecutive quarterly months (starting from the front month) at equal volume per leg, covering two or more years, e.g. a 2 Year Bundle would consist of an equal number of lots (all long or all short) in each White Dec, White March, White Sep, Red Dec, Red Mar, Red Jun and Red Sep.

### 8.2.1 Pack & Bundle pricing and quoting

The Euronext convention for quoting Packs and Bundles is based on the net total change in price between the current trading session's market price and the previous day's settlement price (i.e. the YDSP broadcasted at start of day).

$$Pack\_Bid\_Price = \sum_{i=1}^{1=4} (Bid\ Price_i - YDSP_i)$$

$$Pack\_Ask\_Price = \sum_{i=1}^{1=4} (Ask\ Price_i - YDSP_i)$$

Where i represents the quarterly month.

$$Bundle\_Bid\_Price = \sum_{j=1}^{j=(4 \times N)} (Bid\ Price_j - YDSP_j)$$

$$Bundle\_Ask\_Price = \sum_{j=1}^{j=(4 \times N)} (Ask\ Price_j - YDSP_j)$$

Where j represents the quarterly months and N the number of years of the Bundle.

- **Example: we consider the following order book**

Expiry Month	Bid	Offer	YDSP
MAR	98.865	98.870	98.870
JUN	98.750	98.755	98.760
SEP	98.545	98.550	98.555
DEC	98.210	98.215	98.215

The Bid and Ask prices of the Pack are as follows:

$$PackBid\ Price = (98.865 - 98.870) + (98.750 - 98.760) + (98.545 - 98.555) + (98.210 - 98.215)$$

$$PackBid\ Price = (-0.005) + (-0.010) + (-0.010) + (-0.005)$$

$$PackBid\ Price = -0.030$$

$$PackAsk\ Price = (98.870 - 98.870) + (98.755 - 98.760) + (98.550 - 98.555) + (98.215 - 98.215)$$

$$PackAsk\ Price = (0.000) + (-0.005) + (-0.005) + (0.000)$$

$$PackAsk\ Price = -0.010$$



### 8.2.2 Pack & Bundle Implied In pricing

For contracts supporting Implied In trading in Packs and Bundles, prices in the constituent outright months can feed into the relevant Pack and Bundle market and show as tradeable prices by applying the general pricing formula presented in the previous paragraph.

Let us suppose the following orders in the White (front four) and Red (following four) quarterly months:

Expiry Month	Bid Size	Bid Price	Ask Price	Ask Size	YDSP
MAR	1 200	98.605	98.610	1 000	98.610
JUN	600	98.490	98.495	750	98.500
SEP	550	98.340	98.345	500	98.350
DEC	400	98.140	98.145	200	98.150
MAR	250	98.005	98.010	100	98.010
JUN	180	97.990	97.995	90	98.000
SEP	120	97.740	97.745	80	97.750
DEC	75	97.680	97.685	75	97.690

- The Implied In Bid price and size of the White Pack are equal to:

$$\text{WhitePack Implied Bid Price} = (98.605 - 98.610) + (98.490 - 98.500)$$

$$(98.340 - 98.350) + (98.140 - 98.150)$$

$$\text{WhitePack Implied Bid Price} = (-0.005) + (-0.010) + (-0.010) + (-0.010)$$

$$\text{WhitePack Implied Bid Price} = -0.035$$

$$\text{WhitePack Implied Bid Size} = \min(\text{BidSize}_i)$$

$$\text{WhitePack Implied Bid Size} = 400$$

- The Implied In Ask price and size of the 2 Year Bundle are equal to:

$$\text{2YBundle Implied Ask Price} = (98.610 - 98.610) + (98.495 - 98.500) +$$

$$(98.345 - 98.350) + (98.145 - 98.150) + (98.010 - 98.010) + (97.995 - 98.000) +$$

$$(97.745 - 97.750) + (97.685 - 97.690)$$

$$\text{2YBundle Implied Ask Price} = (0.000) + (-0.005) + (-0.005) + (-0.005) + (0.000) +$$

$$(-0.005) + (-0.005) + (-0.005)$$

$$\text{2YBundle Implied Ask Price} = -0.030$$

$$\text{2YBundle Implied Ask Size} = \min(\text{AskSize}_j)$$

$$\text{2YBundle Implied Ask Size} = 75$$

## 9. TRADING AND ORDER HANDLING

### 9.1 RETRIEVE ORDERS AND TRADES

With UTP-DE, the CCG receives all order and trade notifications from the Router regardless of whether the user is connected or not. Also, the CCG keeps in memory all `Execution Reports` for the day for all its users.

Whatever the value of the `LastSeqNum` sent in the `Logon` message, the CCG checks if there are outstanding `Execution Reports` that was sent for the user while he was not connected and automatically sends them to the client application. This is outlined in the example below.

Client applications can then get details of their GTCs by calling the **Order Mass Status Request (AF)** message. When using this message to retrieve GTC order details for a trader, the application developer should ensure that this request is called only once, ideally immediately upon logging on.

The same message can also be used in order to get details of ex-pit or cross trades still awaiting for Exchange Validation. This function must be called once, after logging on.

The following example illustrates how can members applications retrieve their order book from the time they logon and get outstanding `Execution Reports` to the use of the `Order Mass Status Request` message.

**Note:**

for ease of reading only the fields relevant to the example in the different messages are provided. Clearing information for example has been removed from the `New Order Single` and `Execution Report` messages.

Several orders are submitted, as shown below.

- **Two GTC Orders are submitted in SecurityGroup POFT1 (France Telecom American option class)**

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 03 : 3
ClOrdID        : 01 6F 24 49 : 24061001
Price          : 00 00 00 19 : 25
ExpireDate     : 01 32 B6 11 : 20170625
OrderQty       : 00 00 00 64 : 100
SecurityIDSource : 38           : 8
SecurityID      :                : (POFT1170601600C)

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 04 : 4
OrderID        : 01 04 64 00 00 00 8A 48 : 73293445107583560
ClOrdID        : 01 6F 24 49 : 24061001
Side           : 31             : 1

```

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 05 : 5
ClOrdID         : 01 6F 24 4A : 24061002
Price           : 00 00 00 28 : 40
ExpireDate      : 01 32 B6 11 : 20170625
OrderQty        : 00 00 00 64 : 100
SecurityIDSource : 38           : 8
SecurityID      :               : (POFT1170601550C)
Side            : 31           : 1

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 05 : 5
OrderID         : 01 04 64 00 00 00 8A 49 : 73293445107583561
ClOrdID         : 01 6F 24 4A : 24061002

```

- **Two GTC orders are submitted in the Paris Individual Equity Option Market (Carrefour and L'Oreal option classes)**

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 08 : 8
ClOrdID         : 01 6F 24 4B : 24061003
Price           : 00 00 07 D0 : 1000
ExpireDate      : 01 32 B6 11 : 20150619
OrderQty        : 00 00 00 96 : 150
SecurityIDSource : 38           : 8
SecurityID      :               : (POCA1150602000P)
Side            : 31           : 1

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 06 : 6
OrderID         : 02 00 65 00 00 02 08 B4 : 144226238750394548
ClOrdID         : 01 6F 24 4B : 24061003

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 09 : 9
ClOrdID         : 01 6F 24 4D : 24061005
Price           : 00 00 05 DC : 1500
ExpireDate      : 01 32 B6 11 : 20150619
OrderQty        : 00 00 00 96 : 150
SecurityIDSource : 38           : 8
SecurityID      :               : (POOR1150606000P)
Side            : 31           : 1

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 07 : 7
OrderID         : 02 00 0A 00 00 02 08 52 : 144126183192266834
ClOrdID         : 01 6F 24 4D : 24061005

```

- **Two Day orders are submitted in the Ageas option class on the Brussels market**

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 0E : 14
ClOrdID        : 01 6F 24 50 : 24061008
Price           : 00 00 00 0F : 15
ExpireDate      : 00 00 00 00 : 0
OrderQty        : 00 00 00 96 : 150
SecurityIDSource : 38           : 8
SecurityID      :               : (BOAGB161203400C)
Side            : 31           : 1

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 0A : 10
OrderID         : 01 01 30 00 00 00 8A AC : 72391845572807340
ClOrdID        : 01 6F 24 50 : 24061008

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 0F : 15
ClOrdID        : 01 6F 24 51 : 24061009
Price           : 00 00 00 14 : 20
ExpireDate      : 00 00 00 00 : 0
OrderQty        : 00 00 00 96 : 150
SecurityIDSource : 38           : 8
SecurityID      :               : (BOAGB161203400P)
Side            : 31           : 1

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 0B : 11
OrderID         : 01 01 30 00 00 00 8A AD : 72391845572807341
ClOrdID        : 01 6F 24 51 : 24061009

```

- **One Market order is submitted on an AEX Option. The order automatically matches with a resting order in the market.**

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 12 : 18
ClOrdID        : 01 6F 24 52 : 24061010
Price           : 00 00 00 00 : 0
OrderQty        : 00 00 00 C8 : 200
SecurityIDSource : 38           : 8
SecurityID      :               : (KOAEX100703400P)
OrdType         : 31           : 1
Side            : 31           : 1

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 0C : 12
OrderID         : 02 04 2E 00 00 02 0D 60 : 145291665517710688
ClOrdID        : 01 6F 24 52 : 24061010

```

```

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 0D : 13
OrderID         : 02 04 2E 00 00 02 0D 60 : 145291665517710688
TradeID         : 00 00 00 00 00 00 11 94 : 4500
ExecID          : 02 00 00 00 00 01 20 AE : 144115188075929774
ClOrdID         : 01 6F 24 52 : 24061010
OrigClOrdID     : 00 00 00 00 : 0
TransactTime    : 01 D8 52 E7 : 30954215
LastPx          : 00 00 05 DC : 1500
Price           : 00 00 00 00 : 0
ReturnCode      : 00 F8 00 01 : 16252929
CumQty          : 00 00 00 C8 : 200
LastQty         : 00 00 00 3E : 62
OrderQty        : 00 00 00 C8 : 200
LastRptRequested : 59           : Y
Text            :                : ()
SecurityIDSource : 38           : 8
SecurityID       :                : (KOAEX100703400P)
OrdStatus        : 32           : 2
Side             : 31           : 1
ExecType         : 46           : F

```

The trader logouts from the system then logs back on again with LastSeqNum = -1. In that case, the CCG returns in the LastSeqNumber field what it believes is the last sequence number.

```

MsgId           : 00 21           : 33 [A]
iLen            : 00 42           : 66
lSeqNum         : 00 00 00 0E : 14
LastSeqNum      : FF FF FF FF : -1
HeartBtInt      : 00 3C           : 60
SenderCompId    :                : (XYZ)
SenderSubId     :                : (DTS)
Versions        :                : (A 15 10 11 13 18 1a 19 1D 1E 1F 1G 1N 1R 1)
CancelOnDisconnect : 31           : 1

MsgId           : 00 21           : 33 [A]
iLen            : 00 42           : 66
lSeqNum         : 00 00 00 00 : 0
LastSeqNum      : 00 00 00 12 : 18
HeartBtInt      : 00 3C           : 60
SenderCompId    :                : (EXCHG)
SenderSubId     :                : ()
Versions        :                : ()
CancelOnDisconnect : 00           :

```

From the logon, it appears that the last outbound sequence number is 18 while the last sequence number known by the client application is 13. As LastSeqNum in the logon message has been set to one, the CCG checks if there are any outstanding reports for the session. Such messages correspond to Execution Reports that were generated while the user was not connected and that the CCG kept in memory. In the present example, as LastSeqNum sent by the CCG is 18, there are 4 outstanding Execution Reports in memory for user XYZ.

The first two Execution Reports correspond to the Day orders that were submitted prior to the disconnection and that have been automatically cancelled.

```

MsgId          : 00 A1          : 161 [8]
iLen           : 01 44          : 324
lSeqNum        : 00 00 00 0E : 14
OrderID        : 01 01 30 00 00 00 8A AC : 72391845572807340
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 01 00 00 00 00 01 A9 68 : 72057594038036840
ClOrdID        : 00 00 00 00 : 0
OrigClOrdID    : 01 6F 24 50 : 24061008

```

→ The reference of the order appears in the OrigClOrderID field while

ClOrderID has been set to 0

```

TransactTime   : 01 E3 03 54 : 31654740
LastPx         : 00 00 00 00 : 0
Price          : 00 00 00 0F : 15
ReturnCode     : 00 40 00 01 : 4194305
ExpireDate     : 00 00 00 00 : 0
CumQty         : 00 00 00 00 : 0
LastQty        : 00 00 00 00 : 0
OrderQty       : 00 00 00 96 : 150
LeavesQty      : 00 00 00 96 : 150
LastRptRequested : 59          : Y
Text           :              : ()
SecurityIDSource : 38          : 8
SecurityID      :              : (BODXB100700300C)
OrdStatus       : 34           : 4
OrdType         : 32           : 2
Side            : 31           : 1
ExecType        : 34           : 4

```

```

MsgId          : 00 A1          : 161 [8]
iLen           : 01 44          : 324
lSeqNum        : 00 00 00 0F : 15
OrderID        : 01 01 30 00 00 00 8A AD : 72391845572807341
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 01 00 00 00 00 01 A9 69 : 72057594038036841
ClOrdID        : 00 00 00 00 : 0
OrigClOrdID    : 01 6F 24 51 : 24061009
TransactTime   : 01 E3 03 54 : 31654740
LastPx         : 00 00 00 00 : 0
Price          : 00 00 00 14 : 20
ReturnCode     : 00 40 00 01 : 4194305
ExpireDate     : 00 00 00 00 : 0
CumQty         : 00 00 00 00 : 0
LastQty        : 00 00 00 00 : 0
OrderQty       : 00 00 00 96 : 150
MinQty         : 00 00 00 00 : 0
LeavesQty      : 00 00 00 96 : 150
LastRptRequested : 59          : Y
Text           :              : ()
SecurityIDSource : 38          : 8
SecurityID      :              : (BODXB100700350P)
OrdStatus       : 34           : 4
OrdType         : 32           : 2
Side            : 31           : 1
TimeInForce     : 30           : 0
ExecType        : 34           : 4

```

The last two Execution Reports correspond to GTC orders that have partially traded when the user was not connected to the market.

#### 1- A partial trade occurred on the Buy order on the BNP Paribas option for 50 lots

```

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 10 : 16
OrderID         : 02 00 65 00 00 02 08 B4 : 144226238750394548
TradeID        : 00 00 00 00 00 00 11 F8 : 4600
ExecID         : 02 00 00 00 00 01 25 5E : 144115188075930974
ClOrdID        : 01 6F 24 4B : 24061003
OrigClOrdID    : 00 00 00 00 : 0
TransactTime   : 01 E3 02 D0 : 31654608
LastPx         : 00 00 07 D0 : 2000
Price          : 00 00 07 D0 : 2000
ReturnCode     : 00 F8 00 01 : 16252929
ExpireDate     : 01 32 B6 11 : 20170625
CumQty         : 00 00 00 32 : 50
LastQty        : 00 00 00 32 : 50
OrderQty       : 00 00 00 96 : 150
LeavesQty      : 00 00 00 64 : 100
LastRptRequested : 59           : Y
Text           :                : ()
SecurityIDSource : 38           : 8
SecurityID      :                : (POBN1171205000C)
OrdStatus       : 31           : 1
OrdType         : 32           : 2
Side           : 31           : 1
TimeInForce     : 36           : 6
ExecType        : 46           : F

```

#### 1- A complete trade occurred on the Buy order on the Orange option for 50 lots

```

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 11 : 17
OrderID         : 01 04 64 00 00 00 8A 49 : 73293445107583561
TradeID        : 00 00 00 00 00 00 0A 28 : 2600
ExecID         : 01 00 00 00 00 01 AB BF : 72057594038037439
ClOrdID        : 01 6F 24 4A : 24061002
OrigClOrdID    : 00 00 00 00 : 0
TransactTime   : 01 E3 02 58 : 31654488
LastPx         : 00 00 00 28 : 40
Price          : 00 00 00 28 : 40
ReturnCode     : 00 F8 00 01 : 16252929
ExpireDate     : 01 32 B6 11 : 20170625
CumQty         : 00 00 00 64 : 100
LastQty        : 00 00 00 64 : 100
OrderQty       : 00 00 00 64 : 100
LeavesQty      : 00 00 00 00 : 0
LastRptRequested : 59           : Y
Text           :                : ()
SecondaryClOrdID :                : (NAG)
SecurityIDSource : 38           : 8
SecurityID      :                : (POFT1170601550C)
OrdStatus       : 32           : 2
OrdType         : 32           : 2
Side           : 31           : 1
TimeInForce     : 36           : 6
ExecType        : 46           : F

```

Developers should note that these outstanding reports are sent after the logon has been successfully accepted and before the Contract Availability ('UC') messages notifying users whether contracts listed on the different Trading Engines are available for trading.

```

MsgId           : 02 51           : 593 [UC]
iLen            : 0B C8           : 3016
lSeqNum         : 00 00 00 12 : 18
ContractAvailabilityID: 00 00 00 04 : 4
SecurityIDSource : 50             : P
AvailabilityStatus : 31           : 1
LastRptRequested : 4E            : N
NoContracts      : C8            : 200
SecurityID       :                : (POVA2)
SecurityID       :                : (POLG1)
SecurityID       :                : (POVI1)
SecurityID       :                : (POAF2)
SecurityID       :                : (POVI3)
SecurityID       :                : (POLG2)
SecurityID       :                : (RFSMP)
SecurityID       :                : (GSASL)
SecurityID       :                : (ZFFDE)
SecurityID       :                : (AONUO)
SecurityID       :                : (POUL1)
SecurityID       :                : (PODF1)
SecurityID       :                : (GSHEH)
SecurityID       :                : (PODF3)
SecurityID       :                : (CSPP )

```

→ Developers should note that the SecurityGroup are not sent sequentially per

#### Exchange Code

```

MsgId           : 02 51           : 593 [UC]
iLen            : 0B C8           : 3016
lSeqNum         : 00 00 00 13 : 19
ContractAvailabilityID: 00 00 00 05 : 5
SecurityIDSource : 50             : P
AvailabilityStatus : 31           : 1
LastRptRequested : 4E            : N
NoContracts      : C8            : 200
SecurityID       :                : (POAC2)
SecurityID       :                : (AOASO)
SecurityID       :                : (POAC3)
SecurityID       :                : (CSVK )
SecurityID       :                : (BOSOL)

```

```

.....

MsgId           : 02 51           : 593 [UC]
iLen            : 08 8F           : 2191
lSeqNum         : 00 00 00 14 : 20
ContractAvailabilityID: 00 00 00 06 : 6
SecurityIDSource : 50             : P
AvailabilityStatus : 31           : 1
LastRptRequested : 59            : Y
0NoContracts     : 91            : 145
SecurityID       :                : (POCA1)
SecurityID       :                : (POCS1)
SecurityID       :                : (POGL1)
SecurityID       :                : (POKN1)
SecurityID       :                : (POCR1)

```



The client application then submits **Order Mass Status Request (AF)** messages to get the latest status of open orders. Only GTC orders still active in the market will be returned. Developers must pay attention that the Order Mass Status Request must be called by SecurityGroup. If no SecurityGroup is specified, an Execution Report will be returned by the CCG with an OrdRejReason '313' and a 'No Router For Security Group' Text field. To retrieve the correct status of all orders, the function is called for every SecurityGroup in which the trader submitted orders prior to logging off.

1- The Order Mass Status Request message is called for SecurityGroup POFT1. Only one Execution Report is sent as a response as one of the two orders has been filled.

```

MsgId           : 01 61           : 353 [AF]
iLen            : 00 1D           : 29
lSeqNum         : 00 00 00 16 : 22
MassStatusReqID : 00 00 04 57 : 1111
MassStatusReqType : 07           : 7
SecurityIDSource : 50            : P
SecurityID      :                : (POFT1)

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 15 : 21
OrderID         : 01 04 64 00 00 00 8A 48 : 73293445107583560
TradeID         : 00 00 00 00 00 00 00 00 : 0
ExecID         : 01 00 00 00 00 01 AB C0 : 72057594038037440
ClOrdID        : 01 6F 24 49 : 24061001
OrigClOrdID    : 00 00 00 00 : 0
TransactTime   : 01 E6 15 12 : 31855890
LastPx         : 00 00 00 00 : 0
Price          : 00 00 00 19 : 25
ReturnCode     : 00 40 00 01 : 4194305
ExpireDate     : 01 32 B6 11 : 20170625
CumQty         : 00 00 00 00 : 0
LastQty        : 00 00 00 00 : 0
OrderQty       : 00 00 00 64 : 100
LastRptRequested : 59          : Y
Text           :                : ()
SecurityIDSource : 38          : 8
SecurityID      :                : (POFT1170601600C)
OrdStatus      : 30          : 0
OrdType        : 32           : 2
Side           : 32           : 2
TimeInForce    : 36           : 6
ExecType      : 49          : I

```

→ ExecType = 'I' as an response to an OrderMassStatusRequest

2- The Order Mass Status Request message is called for SecurityGroup AOASL. The CumQty, OrderQty and LeavesQty fields indicate that the order has been partially filled for 50 lots.

```

MsgId           : 01 61           : 353 [AF]
iLen            : 00 1D           : 29
lSeqNum         : 00 00 00 17 : 23
MassStatusReqID : 00 00 04 57 : 1111
MassStatusReqType : 07           : 7
SecurityIDSource : 50            : P
SecurityID      :                : (AOASL)
ThreadID       :                : 1253816640

```

```

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 16 : 22
OrderID         : 02 00 65 00 00 02 08 B4 : 144226238750394548
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 02 00 00 00 00 01 25 5F : 144115188075930975
ClOrdID        : 01 6F 24 4B : 24061003
OrigClOrdID    : 00 00 00 00 : 0
TransactTime   : 01 E6 8C B5 : 31886517
LastPx         : 00 00 00 00 : 0
Price          : 00 00 07 D0 : 2000
ReturnCode     : 00 40 00 01 : 4194305
ExpireDate     : 01 32 B6 11 : 20170625
CumQty         : 00 00 00 32 : 50
LastQty        : 00 00 00 00 : 0
OrderQty       : 00 00 00 96 : 150
LeavesQty      : 00 00 00 64 : 100
LastRptRequested : 59           : Y
Text           :                : ( )
SecurityIDSource : 38           : 8
SecurityID      :                : (AOASL170610000C)
OrdStatus       : 31           : 1
OrdType         : 32           : 2
Side            : 31           : 1
TimeInForce     : 36           : 6
ExecType        : 49           : I

```

### 3- The Order Mass Status Request message is called for SecurityGroup PODA1

```

MsgId           : 01 61           : 353 [AF]
iLen            : 00 1D           : 29
lSeqNum         : 00 00 00 18 : 24
MassStatusReqID : 00 00 04 57 : 1111
MassStatusReqType : 07           : 7
SecurityIDSource : 50           : P
SecurityID      :                : (PODA1)

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 17 : 23
OrderID         : 02 00 0A 00 00 02 08 52 : 144126183192266834
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 02 00 00 00 00 01 31 79 : 144115188075934073
ClOrdID        : 01 6F 24 4D : 24061005
OrigClOrdID    : 00 00 00 00 : 0
TransactTime   : 01 E6 AD A6 : 31894950
LastPx         : 00 00 00 00 : 0
Price          : 00 00 05 DC : 1500
ReturnCode     : 00 40 00 01 : 4194305
ExpireDate     : 01 32 B6 11 : 20170625
CumQty         : 00 00 00 00 : 0
LastQty        : 00 00 00 00 : 0
OrderQty       : 00 00 00 96 : 150
LeavesQty      : 00 00 00 96 : 150
LastRptRequested : 59           : Y
Text           :                : ( )
SecurityIDSource : 38           : 8
SecurityID      :                : (PODA1170606200P)
OrdStatus       : 30           : 0
OrdType         : 32           : 2
Side            : 31           : 1
TimeInForce     : 36           : 6
ExecType        : 49           : I

```

#### 4- The Order Mass Status Request message is called for SecurityGroup KOAEX

```

MsgId           : 01 61           : 353 [AF]
iLen            : 00 1D           : 29
lSeqNum         : 00 00 00 1A : 26
MassStatusReqID : 00 00 04 57 : 1111
MassStatusReqType : 07           : 7
SecurityIDSource : 50            : P
SecurityID      :                : (KOAEX)

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 19 : 25
OrderID         : 00 00 00 00 00 00 00 00 : 0
TradeID         : 00 00 00 00 00 00 00 00 : 0
ExecID          : 02 00 00 00 00 01 20 AF : 144115188075929775
ClOrdID         : 00 00 00 00 : 0
OrigClOrdID     : 00 00 00 00 : 0
TransactTime    : 01 E7 3C 5A : 31931482
LastPx          : 00 00 00 00 : 0
Price           : 00 00 00 00 : 0
ReturnCode      : 00 40 00 01 : 4194305
ExpireDate      : 00 00 00 00 : 0
CumQty          : 00 00 00 00 : 0
LastQty         : 00 00 00 00 : 0
OrderQty        : 00 00 00 00 : 0
LeavesQty       : 00 00 00 00 : 0
LastRptRequested : 59           : Y

```

→ LastRptRequested = 'Y' indicates that this is the last Execution Report sent as the response to the Order Mass Status Request

```

Text           :                : ()
SecurityIDSource : 50            : P
SecurityID      :                : (KOAEX)
OrdStatus      : 38           : 8
OrdType         : 00            :
Side            : 00            :
ExecType      : 38           : 8

```

→ Developers should pay attention to the value of the ExecType returned in the Execution Report: when no order is active in the SecurityGroup, ExecType = '8' (whereas ExecType = 'I' for open active orders as a response of an OrderMassStatus Request).

#### 5- The Order Mass Status Request message is called for SecurityGroup BOAGB

```

MsgId           : 01 61           : 353 [AF]
iLen            : 00 1D           : 29
lSeqNum         : 00 00 00 1B : 27
MassStatusReqID : 00 00 04 57 : 1111
MassStatusReqType : 07           : 7
SecurityIDSource : 50            : P
SecurityID      :                : (BOAGB)

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 1A : 26
OrderID         : 00 00 00 00 00 00 00 00 : 0
TradeID         : 00 00 00 00 00 00 00 00 : 0
ExecID          : 01 00 00 00 00 01 A9 6A : 72057594038036842
ClOrdID         : 00 00 00 00 : 0
OrigClOrdID     : 00 00 00 00 : 0

```

```

TransactTime      : 01 E7 9D 0A : 31956234
LastPx            : 00 00 00 00 : 0
Price             : 00 00 00 00 : 0
ReturnCode        : 00 40 00 01 : 4194305
ExpireDate        : 00 00 00 00 : 0
CumQty            : 00 00 00 00 : 0
LastQty           : 00 00 00 00 : 0
OrderQty          : 00 00 00 00 : 0
LeavesQty         : 00 00 00 00 : 0
LastRptRequested  : 59           : Y
Text              :                : ( )
SecurityIDSource  : 50           : P
SecurityID        :                : (BODXB)
OrdStatus         : 38           : 8
OrdType           : 00           :
Side              : 00           :
TimeInForce       : 00           :
ExecType          : 38           : 8

```

## 9.2 SUBMIT ORDERS

All types of orders available on Euronext markets, with the exception of Mass Quotes, Wholesale trades and List orders on outright as well strategies can be submitted using the **New Order Single(D)** message.

### Note:

Members using the FIX application must pay attention to the FIX required fields.

For more detailed information about the order types, please refers to the Appendix A of the Binary and FIX specifications.

Order types may vary between contracts. The FIXML standing data files provide at the <Contract> level, in the <OrderTypeRules> block, the list of supported orders. See the example below for the AEX Index Futures.

```

<Contract SecGrp="KFFTI" LogSym="FTI" Desc="FUTURE AEX INDEX" Exch="XEUE"
RndLot="200" Ccy="EUR" MinPxIncr=".01" PxQteMeth="STD" SettlMeth="P" FlexInd="N"
CFI="XXXXXX" Src="P" ID="GIAEX">
  <NestedAttr Typ="25" Val="100" />
  <NestedAttr Typ="101" Val="1" />
  <NestedAttr Typ="102" Val="1" />
  <NestedAttr Typ="120" Val="T" />
  <NestedAttr Typ="121" Val="4" />
  <OrdTypeRules OrdTyp="1" /> → Market Order
  <OrdTypeRules OrdTyp="2" /> → Limit Order
  <OrdTypeRules OrdTyp="W" /> → Market On Open (MOO)
  <WTradeTypeRules WTrdTyp="6" />
  <WTradeTypeRules WTrdTyp="5" />

```

---

### 9.2.1 The ClOrderID field

Every outright or strategy order must contain a unique client reference in the `ClOrdId` field. There is no imposed format; it is up to customers to choose. Nevertheless, the implementation should follow some rules:

- Uniqueness of the order reference must be guaranteed across multiple days for GTCs. The reasons for such uniqueness are outlined below. The FIX Protocol recommends prefixing the `ClOrdId` by a date.
- Uniqueness must be guaranteed within a trading day by session
- When replacing, cancelling orders, client applications should either specify the `ClOrdId` and / or the `OrderId` provided by the Trading Host to identify the original order. The `ClOrdId` has to be specified in the `OrigClOrderId` field. Developers are therefore recommended to implement a chaining between the `ClOrdId` and the `OrigClOrderId` fields.

During the day, the CCG maintains a list of all the `ClOrdIDs` of orders submitted by an ITM. Any new order submission is validated to ensure that it has a unique `ClOrdID` for that user. The Trading Engine does not validate `ClOrdIDs` – it assumes that such validation has already been done by the CCG.

Developers should pay attention to the following when implementing their `ClOrderId` sequencing to handle GTC orders. When a new order is submitted, the Trading Engine maps this `ClOrdID` to reference the new order. This is done on a per contract basis. Any subsequent submissions with the same `ClOrdID` and the same contract will overwrite the reference of the `ClOrdID` to refer to the most recent order. This means that it is possible for `ClOrdIDs` to be duplicated over different trading days.

For example, one day a trader submits a GTC order with `ClOrdID` 'A'. This order gets allocated exchange `OrderID` '1'.

The next day the same trader submits an order also with `ClOrdID` 'A'. This order gets allocated exchange `OrderID` '2'.

The trader then submits a revision (same for a pull), specifying `ClOrdID` 'A' and not specifying an exchange `OrderID`.

Note that all revisions and pulls must specify the contract e.g. "AEX" (AEX option)

If both orders were for the same contract then the revision (or pull) would affect the second order – the one with exchange `OrderID` '2'.

If the orders were for different contracts then the correct order would be referenced.

---

### 9.2.2 Description of the 'New Order Single' message

For the description of the `New Order Single` message, please refer to the BIN and FIX messages specifications.

### 9.2.3 What are the mandatory clearing fields per Euronext markets?

Clearing information varies between Euronext markets. Some fields might therefore be mandatory in certain markets while optional in other ones. The table below provides the rules to follow to setup the clearing fields in the different order messages.

**Note:**

Developers must ensure that they currently display and interpret any order rejection due to incorrect clearing information.

Clearing field	Paris Brussels Lisbon	Amsterdam
<b>PartyRole</b>	Optional. The PartyRole field must be used to specify that the trade has to be given up to a Member firm identified either by its Member Mnemonic (PartyRole = 14) or by a trader mnemonic (PartyRole = 53). The AccountCode field must also be set to 'A' and the ClearingInstruction field to "4010" for Paris, Brussels, Lisbon or Amsterdam market	
<b>PartyID</b>	Mandatory only when the PartyRole field is set. It must contain the Member Mnemonic or the trader mnemonic to whom the trade is allocated to depending on the value of PartyRole field	
<b>NoPartyID</b>	FIX only field. The NoPartyID field is mandatory only when PartyRole is set. One unique value, NoPartyID = 1.	
<b>PartyIDSource</b>	FIX only field. The NoPartyIDSource field is mandatory only when PartyRole is set. One unique value, NoPartyIDSource = 'D'	
<b>AccountCode</b>	Mandatory field. The available values are: 'A' = Give up to multiple firms 'C' = Customer 'H' = House 'M' = Market Maker	
<b>ClientInfo</b>	Mandatory when ClearingInstruction is set	Mandatory field
<b>SecondaryCLOrderID</b>	Optional	Optional
<b>PostingAction</b>	Optional. The PostingAction contains the Open / Close Indicator. When submitting a strategy order, the PostingAction can be set up individually for each leg. For strategies with more than 4 legs, all legs beyond the fourth leg will take the same value as the fourth leg.	
<b>Account</b>	Optional	Mandatory only when AccountCode is set to 'C'

Clearing field	Paris Brussels Lisbon	Amsterdam
<b>ClearingInstruction</b>	Optional. The field is used to specify that the trade has to be posted in a specific account in the Clearing System. The available values are: 0: Undefined 8: Manual Posting 9: Automatic Posting (the field ClientInfo can be used to provide the account name) 4010: Give-Up (the fields PartyRole and PartyID Source must also be set)	Optional. The field is used to specify that the trade has to be posted in a specific account in the Clearing System. The available values are: 0: Undefined 8: Manual Posting 9: Automatic Posting (the field ClientInfo can be used to provide the account name) 4010: Give-Up (the fields PartyRole and PartyID Source must also be set) 4008: Manual Posting and Account Authorisation (1) 4009: Automatic Posting and Account Authorisation
<b>CustomerOrderCapacity</b>	Not used in Euronext markets	
<b>OrderOrigin</b>	Not used in Euronext markets	

(1): The Account Authorization requires a trader to have explicit pre-negotiated permission to give-up a trade to a separate member without requiring the user manual intervention.

- When Manual Posting plus Account Authorisation is used, the Give Up is automatically taken up but user intervention is required to specify into which posting position account the trade is to be allocated.
- When Automatic Posting plus Account Authorisation is used, the Give Up is automatically taken up and is automatically allocated to the position account specified in the ClientInfo field

**Note:**

An example showing the differences between an order creation message sent with the FIX or the Binarymessage format will be provided in a future version of this document.

### 9.3 PULL ORDERS

The CCG provides client apps with 3 different messages to pull open orders:

- The **Order Cancel Request (F)** message can be used to pull a single order. The order might be cancel either by its **ClOrderID** as assigned by the trader or by the **Exchange OrderID**.
- The **Order Mass Cancel Request (q)** message must be used to cancel multiple orders. When using this message, client application should ensure that the **MassCancelRequestType** is specified at the highest level. This ensures that orders are pulled from the market as quickly as possible. The hierarchy of levels is:
  - **'A'**: By SecurityGroup (highest)
  - **'W'**: By maturity Date (of a same SecurityGroup)
  - **'1'**: By instrument (using its AMR)
- Members wishing to remove a series of individual orders (up to 32) representing a trading strategy for example can use the **Order Cancel List (UB)** message.

**Note:**

The **Order Cancel List** message is only available in the Binary interface

#### Example of Mass Order Cancel Request

In the example below, five orders are entered into the POCA1 (Carrefour American Paris Equity Option).

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 68 : 104
ClOrdID        : 01 6F 24 66 : 24061030
Price           : 00 00 00 73 : 115
OrderQty        : 00 00 00 64 : 100
SecurityIDSource : 38           : 8
SecurityID      :               : (POCA1100803600C)
Side            : 31            : 1

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 2B : 43
OrderID         : 01 04 54 00 00 00 8B 78 : 73275852921539448
ClOrdID         : 01 6F 24 66 : 24061030

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 69 : 105
ClOrdID        : 01 6F 24 68 : 24061032
Price           : 00 00 00 78 : 120
OrderQty        : 00 00 00 64 : 100
SecurityIDSource : 38           : 8
SecurityID      :               : (POCA1100803500C)
Side            : 31            : 1

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 2C : 44
OrderID         : 01 04 54 00 00 00 8B 79 : 73275852921539449
ClOrdID         : 01 6F 24 68 : 24061032

```



```

MsgId          : 00 41          : 65 [D]
iLen           : 00 72          : 114
lSeqNum        : 00 00 00 6A : 106
ClOrdID       : 01 6F 24 6A : 24061034
Price          : 00 00 00 96 : 150
OrderQty       : 00 00 00 64 : 100
SecurityIDSource : 38          : 8
SecurityID     :              : (POCA1100803400C)
Side           : 31           : 1

MsgId          : 00 91          : 145 [a]
iLen           : 00 14          : 20
lSeqNum        : 00 00 00 2D : 45
OrderID        : 01 04 54 00 00 00 8B 7A : 73275852921539450
ClOrdID        : 01 6F 24 6A : 24061034

MsgId          : 00 41          : 65 [D]
iLen           : 00 72          : 114
lSeqNum        : 00 00 00 76 : 118
ClOrdID       : 01 6F 24 79 : 24061049
Price          : 00 00 00 82 : 130
OrderQty       : 00 00 00 64 : 100
SecurityIDSource : 38          : 8
SecurityID     :              : (POCA1100703400P)
Side           : 31           : 1

MsgId          : 00 91          : 145 [a]
iLen           : 00 14          : 20
lSeqNum        : 00 00 00 36 : 54
OrderID        : 01 04 54 00 00 00 8B 7B : 73275852921539451
ClOrdID        : 01 6F 24 79 : 24061049

MsgId          : 00 41          : 65 [D]
iLen           : 00 72          : 114
lSeqNum        : 00 00 00 77 : 119
ClOrdID       : 01 6F 24 7B : 24061051
Price          : 00 00 00 82 : 130
OrderQty       : 00 00 00 64 : 100
SecurityIDSource : 38          : 8
SecurityID     :              : (POCA1100703300C)
Side           : 32           : 2

MsgId          : 00 91          : 145 [a]
iLen           : 00 14          : 20
lSeqNum        : 00 00 00 37 : 55
OrderID        : 01 04 54 00 00 00 8B 7C : 73275852921539452
ClOrdID        : 01 6F 24 7B : 24061051
ThreadID       :              : 1243326784

```

The message is called at the SecurityGroup level, i.e. MassCancelRequestType = 'A'

```

MsgId          : 01 11          : 273 [q]
iLen           : 00 26          : 38
lSeqNum        : 00 00 00 78 : 120
ClOrdID       : 01 6F 24 7D : 24061053
MassCancelRequestType: 41 : A
SecurityIDSource : 50          : P
SecurityID     :              : (POCA1)
RiskID         :              : ()

```

→ The ClOrderId is a mandatory field of the Order Mass Cancel Request message. However client apps should make sure that the ClOrdID assigned to the request is unique amongst the ClOrdID assigned to order submission, revision, single order cancel requests and other order mass cancel requests.

A series of Execution Reports are sent for each order cancelled meeting the criteria entered.

```

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 38 : 56
OrderID         : 01 04 54 00 00 00 8B 7C : 73275852921539452
TradeID         : 00 00 00 00 00 00 00 00 : 0
ExecID          : 01 00 00 00 00 01 90 70 : 72057594038030448
ClOrdID         : 01 6F 24 7D : 24061053
OrigClOrdID    : 01 6F 24 7B : 24061051
TransactTime    : 02 4B 4A 71 : 38488689
Price           : 00 00 00 82 : 130
ReturnCode      : 00 40 00 01 : 4194305
OrderQty        : 00 00 00 64 : 100
LeavesQty       : 00 00 00 64 : 100
LastRptRequested : 4E           : N
Text            :                : ( )
SecurityIDSource : 38           : 8
SecurityID      :                : (POCA1100703300C)
OrdStatus      : 34           : 4
Side            : 32           : 2
ExecType      : 34           : 4

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 39 : 57
OrderID         : 01 04 54 00 00 00 8B 78 : 73275852921539448
TradeID         : 00 00 00 00 00 00 00 00 : 0
ExecID          : 01 00 00 00 00 01 90 71 : 72057594038030449
ClOrdID         : 01 6F 24 7D : 24061053
OrigClOrdID    : 01 6F 24 66 : 24061030
TransactTime    : 02 4B 4A 71 : 38488689
Price           : 00 00 00 73 : 115
ReturnCode      : 00 40 00 01 : 4194305
OrderQty        : 00 00 00 64 : 100
LeavesQty       : 00 00 00 64 : 100
LastRptRequested : 4E           : N
Text            :                : ( )
SecurityIDSource : 38           : 8
SecurityID      :                : (POCA1100803600C)
OrdStatus      : 34           : 4
Side            : 31           : 1
ExecType      : 34           : 4

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 3A : 58
OrderID         : 01 04 54 00 00 00 8B 79 : 73275852921539449
TradeID         : 00 00 00 00 00 00 00 00 : 0
ExecID          : 01 00 00 00 00 01 90 72 : 72057594038030450
ClOrdID         : 01 6F 24 7D : 24061053
OrigClOrdID    : 01 6F 24 68 : 24061032
TransactTime    : 02 4B 4A 71 : 38488689
Price           : 00 00 00 78 : 120
ReturnCode      : 00 40 00 01 : 4194305
OrderQty        : 00 00 00 64 : 100
LeavesQty       : 00 00 00 64 : 100
LastRptRequested : 4E           : N
Text            :                : ( )
SecurityIDSource : 38           : 8
SecurityID      :                : (POCA1100803500C)
OrdStatus      : 34           : 4
Side            : 31           : 1
ExecType      : 34           : 4

```

```

MsgId          : 00 A1          : 161 [8]
iLen           : 01 44          : 324
lSeqNum        : 00 00 00 3B : 59
OrderID        : 01 04 54 00 00 00 8B 7A : 73275852921539450
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 01 00 00 00 00 01 90 73 : 72057594038030451
ClOrdID        : 01 6F 24 7D : 24061053
OrigClOrdID    : 01 6F 24 6A : 24061034
TransactTime   : 02 4B 4A 71 : 38488689
Price          : 00 00 00 96 : 150
ReturnCode     : 00 40 00 01 : 4194305
OrderQty       : 00 00 00 64 : 100
LastRptRequested : 4E          : N
Text           :               : ()
SecurityIDSource : 38          : 8
SecurityID     :               : (POCA1100803400C)
OrdStatus      : 34          : 4
Side           : 31           : 1
ExecType       : 34          : 4

```

```

MsgId          : 00 A1          : 161 [8]
iLen           : 01 44          : 324
lSeqNum        : 00 00 00 3C : 60
OrderID        : 01 04 54 00 00 00 8B 7B : 73275852921539451
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 01 00 00 00 00 01 90 74 : 72057594038030452
ClOrdID        : 01 6F 24 7D : 24061053
OrigClOrdID    : 01 6F 24 79 : 24061049
TransactTime   : 02 4B 4A 71 : 38488689
Price          : 00 00 00 82 : 130
ReturnCode     : 00 40 00 01 : 4194305
OrderQty       : 00 00 00 64 : 100
LastRptRequested : 59          : Y

```

→ LastRptRequested = 'Y' indicates that this Execution Report is the last one received as the response to the Order Mass Cancel Request message.

```

Text           :               : ()
SecondaryClOrdID :               : (NAG)
SecurityIDSource : 38          : 8
SecurityID     :               : (POCA1100703400P)
OrdStatus      : 34          : 4
Side           : 31           : 1
ExecType       : 34          : 4

```

The message is then acknowledged by an **Order Mass Cancel Report** message.

```

MsgId          : 01 51          : 337 [r]
iLen           : 00 41          : 65
lSeqNum        : 00 00 00 3D : 61
ClOrdID        : 01 6F 24 7D : 24061053
TotalAffectedOrders : 00 00 00 05 : 5
MassCancelRejectReason: 00 00          : 0
MassCancelRequestType: 41             : A
Text           :               : ()
RiskID         :               : ()
MassCancelResponse : 41             : A

```

**Example of an Order Cancel List message**

In the example below, let us assume that a few orders are submitted into the POFT1 (France Telecom American Paris Equity Option) in different expiries and different strikes. This example outlines that within the list, orders can individually be cancelled by their ClOrderID or their Exchange OrderID.

```

MsgId      : 00 41      : 65 [D]
iLen       : 00 72      : 114
lSeqNum    : 00 00 00 7A : 122
ClOrdID   : 01 6F 24 84 : 24061060
Price      : 00 00 00 0F : 15
OrderQty   : 00 00 00 64 : 100
SecurityIDSource : 38      : 8
SecurityID :              : (POFT1100801550C)
Side       : 32          : 2

```

```

MsgId      : 00 91      : 145 [a]
iLen       : 00 14      : 20
lSeqNum    : 00 00 00 3E : 62
OrderID   : 01 04 64 00 00 00 8A 4B : 73293445107583563
ClOrdID    : 01 6F 24 84 : 24061060

```

```

MsgId      : 00 41      : 65 [D]
iLen       : 00 72      : 114
lSeqNum    : 00 00 00 7C : 124
ClOrdID   : 01 6F 24 88 : 24061064
Price      : 00 00 00 B4 : 180
OrderQty   : 00 00 00 64 : 100
SecurityIDSource : 38      : 8
SecurityID :              : (POFT1100901600P)
Side       : 31          : 1

```

```

MsgId      : 00 91      : 145 [a]
iLen       : 00 14      : 20
lSeqNum    : 00 00 00 40 : 64
OrderID   : 01 04 64 00 00 00 8A 4D : 73293445107583565
ClOrdID    : 01 6F 24 88 : 24061064

```

```

MsgId      : 01 A1      : 417 [UB]
iLen       : 00 40      : 64
lSeqNum    : 00 00 00 81 : 129
ListID     : 00 00 01 4D : 333
SecurityIDSource : 50      : P
SecurityID :              : (POFT1)
NoOrders   : 02          : 2
Filler1[3] :              :
OrderID    : 00 00 00 00 00 00 00 00 : 0
OrigClOrdID : 01 6F 24 84 : 24061060

```

➔ the first order is cancelled using the ClOrderID

```

Filler[4]   :              :
OrderID   : 01 04 64 00 00 00 8A 4D : 73293445107583565

```

➔ the second order is cancelled using the OrderID

```

OrigClOrdID : 00 00 00 00 : 0
Filler[4]   :              :

```

```

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 42 : 66
OrderID         : 01 04 64 00 00 00 8A 4B : 73293445107583563
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 01 00 00 00 00 01 AB C5 : 72057594038037445
ClOrdID        : 00 00 00 00 : 0
OrigClOrdID    : 01 6F 24 84 : 24061060
TransactTime    : 02 50 AD 04 : 38841604
Price           : 00 00 00 0F : 15
ReturnCode      : 00 40 00 01 : 4194305
OrderQty        : 00 00 00 64 : 100
LastRptRequested : 4E           : N
Text            :                : ( )
SecurityIDSource : 38           : 8
SecurityID      :                : (POFT1100801550C)
OrdStatus      : 34           : 4
Side            : 32           : 2
ExecType      : 34           : 4

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 43 : 67
OrderID        : 01 04 64 00 00 00 8A 4D : 73293445107583565
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 01 00 00 00 00 01 AB C6 : 72057594038037446
ClOrdID        : 00 00 00 00 : 0
OrigClOrdID    : 01 6F 24 88 : 24061064
TransactTime    : 02 50 AD 04 : 38841604
Price           : 00 00 00 B4 : 180
ReturnCode      : 00 40 00 01 : 4194305
OrderQty        : 00 00 00 64 : 100
LastRptRequested : 59           : Y
Text            :                : ( )
SecurityIDSource : 38           : 8
SecurityID      :                : (POFT1100901600P)
OrdStatus      : 34           : 4
Side            : 31           : 1
ExecType      : 34           : 4

```

In the event that some order(s) within the list cannot be cancelled, i.e. they are no longer active, client apps will be notified via Order Cancel Reject messages.

## 9.4 REVISE ORDERS

When changing the price or quantity of an order, members should ensure they use the **Order Revision Request (G)** message rather than pulling and re-submitting the order.

**In UTP, only the OrderQty, meaning the original volume of the order can be revised.**

**Notes:**

- The time stamp of the order will be updated only if the price is changed or the volume is increased.
- The system validates all parameters, irrespective of revision type. Ensure unchanged fields are set to LIFFE\_NO\_PRICE (999999999 – nine 9), LIFFE\_MAX\_VOLUME+1 (999999999 – nine 9).
- Cross orders cannot be revised once they have been submitted.
- If a crossing violation could occur because part of the revised order could match with an earlier order from the same trader, the system would match the volume that could be traded without causing a crossing violation, and the residual volume would be rejected.

UTP offers two different ways to revise orders i.e. using the Exchange OrderID or the ClOrdID. Developers can use either the OrderID or the ClOrdID, but **NOT** both.

When the ClOrdID option is selected, developers must pay attention that the Trading Engine does **NOT** change the OrigClOrderID after each revision request. The OrigClOrderID remains constant and is set to the ClOrdID relating to the ClOrdID of the original order submitted.

Let us suppose that an order is submitted with ClOrdID = 2010001. Further revisions of the order using the ClOrdID must be submitted with the following combinations:

- Revision 1: ClOrderID = 2010002 - OrigClOrderID = 2010001
- Revision 2: ClOrderID = 2010003 - OrigClOrderID = 2010001
- Revision 3: ClOrderID = 2010004 - OrigClOrderID = 2010001

When the OrderID option is selected, developers must pay attention that the OrigClOrderID will be left blank in the Execution Report.

---

## 9.5 WHOLESALE TRADING

The Euronext derivative Exchange offers a number of different pre negotiated trades to answer the needs of the wholesale market. All these trading facilities must be first pre negotiated bilaterally by traders, but could then be reported and executed through UTP.

Wholesale trading facilities can be submitted using the **New Order Cross ('s')** message. This includes the following types:

- **Large in Scale (LIS) Trades:** This facility allows traders to transact business of significant size as bilateral agreed transactions on-exchange. Trading volume must be above a pre-defined volume in a number of Euronext Equity contracts, and when permitted, as well as on strategies.
- **Basis Trades:** This facility allows a trader to enter into a conditional transaction involving both a Euronext Futures contract and a corresponding Cash instrument. The executing member must assign the price(s) to the futures leg(s) of the trade. Reference information about the cash leg must also be provided.
- **Against Actuals:** This facility is dedicated to the Commodities market. As basis trades, it incorporates a Futures leg and an underlying commodity leg. Trade notification must contain the price and volume of the future leg as well as information about the commodity leg.
- **Exchange for Swap:** This facility allows holders of OTC swap positions to register them with the equivalent in commodity futures contracts

- **Guaranteed Crosses:** This is a pre negotiated order that, subject to certain conditions, will be automatically matched by the Trading Host. It allows a trader to enter an order for both sides. Guaranteed crosses must be for a minimum volume and follow contract specific price restrictions relative to the BBO. For certain contracts, a Request For Quote (RFQ) must be submitted to the market prior to the execution of the Guaranteed Cross.
- **Large in Scale (LiS) Packages:** They are pre negotiated trades, that are subject to a minimum volume threshold, and that take place outside of the central order book. A Large in Scale (LiS) Package results from the matching of two Large in Scale (LiS) Package **'intentions'**, in the same way as a normal trade results from the matching of two orders. Each counterparty should enter his intentions, including price and volume details, as well as the identification of the 'intended' counterparty. Then, intentions match only if both parties have submitted the correct matching trade details.

All the wholesale trading facilities above are automatically validated by the Trading Host.

The availability of these trades is defined at a contract level, per market. For more information, please refer to the Rule Book available on the [Euronext / market rules](#) section of the Euronext "Regulation" website. Developers can also refer to the **WTradeTypeRules** subblock of the FIXML standing data files to find the list of available wholesale trades per contract.

Minimum volume thresholds for outright and strategies Large in Scale (LiS) trades and Guaranteed Crosses also vary between Exchanges and contracts. These thresholds are provided in the FIXML standing data files per contract at the expiry level in the following fields:

- NestedAttribute "107" : Outright Large in Scale (LiS) Trade Threshold
- NestedAttribute "108": Strategy Large in Scale (LiS) Trade Threshold
- NestedAttribute "109": Outright Guaranteed Cross Threshold
- Nested Attribute "110": Strategy Guaranteed Cross Threshold

See the example below for the CAC40 Option contract.

```
<Contract SecGrp="PXA" Sym="PXA" LogSym="PXA" Desc="CAC 40 Index Option* (10
Euro)" Exch="XMON" RndLot="10" Ccy="EUR" MinPxIncr=".01" PxQteMeth="STD"
SettlMeth="C" ExerStyle="0" FlexInd="N" CFI="OPEXCS" Src="P" ID="CIPX1">
  <NestedAttr Typ="25" Val="100" />
  <NestedAttr Typ="101" Val="1" />
  <NestedAttr Typ="102" Val="1" />
  <NestedAttr Typ="103" Val="1" />
  <NestedAttr Typ="104" Val="0" />
  <NestedAttr Typ="110" Val="C" />
  <NestedAttr Typ="120" Val="T" />
  <NestedAttr Typ="121" Val="8" />
  <OrdTypeRules OrdTyp="1" />
  <OrdTypeRules OrdTyp="2" />
  <WTradeTypeRules WTrdTyp="1" />
  <WTradeTypeRules WTrdTyp="6" />
  <SecSubTypes SubTyp="A" />
  <SecSubTypes SubTyp="B" />
  .....
  <Expiry ExpiryDt="20170700" MMYFmt="0" MMY="201509" FirstTrdDt="20130923"
LastTrdDt="20170719" UndMMYFmt="0" UndMMY="201711">
    <NestedAttr Typ="26" Val="10" />
    <NestedAttr Typ="107" Val="500" />
    <NestedAttr Typ="109" Val="1000" />
```

Some Wholesale Trades can be submitted in Open and Pre-Close periods. However, for a few individual contracts, their submission is also allowed in Pre-Open and after Market Close. Notification of this feature is disseminated in XDP via the **Market Status (752) message - Market Mode = 4 (ExPit Extend Open)**.

All types of Ex-Pit Trades can be submitted using the **New Order Cross (s)** message. The explicit type must be specified in the `WholesaleTradeType` field.

For more information about the `New Order Cross` message, please refer to the “CCG Functional Descriptions” document.

Client applications should pay attention:

- to the specific fields to fill for each type of Wholesale Trade. For example, the `OrderCapacity` field is specific to Large in Scale (LiS) Package and must not be filled for any other types of trades.
- to the content of the two Buy and Sell repeating legs.

For a description of the `New Order Cross` message, please refer to the FIX and BIN messages specifications.

**Note:**

An example of a Wholesale order entry will be provided in a future version of this document.



## 9.6 STOCK ORDER ROUTING

Stock Order Routing is a facility whereby a trader or market maker may send a stock order to their clearing member for execution. The trader or market maker sends a request to buy or sell stock to another ITM at their clearing member. The ITM to which the request is routed receives the request, then attempts to execute the trade in the relevant stock market, and returns a response indicating the result of the trade to the originating trader.

**Note:**

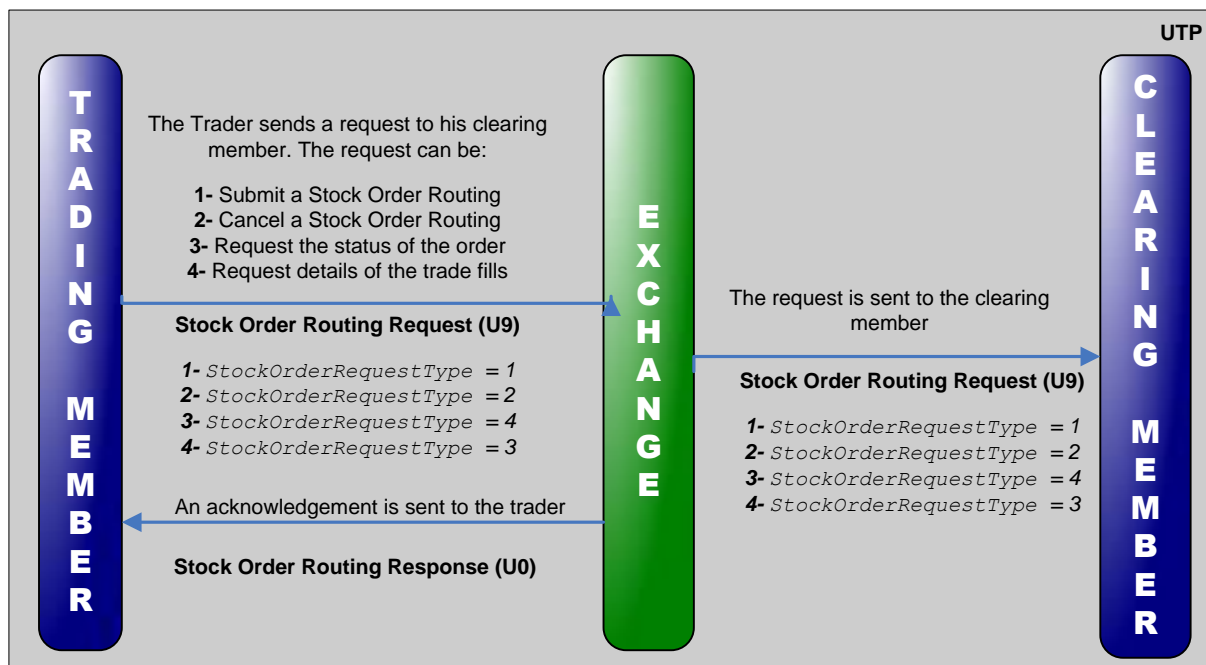
The Trading Engine routes messages to the recipients only if they are connected. During the backward compatibility period, this is the case whether the target ITM is logged on via the CCG.

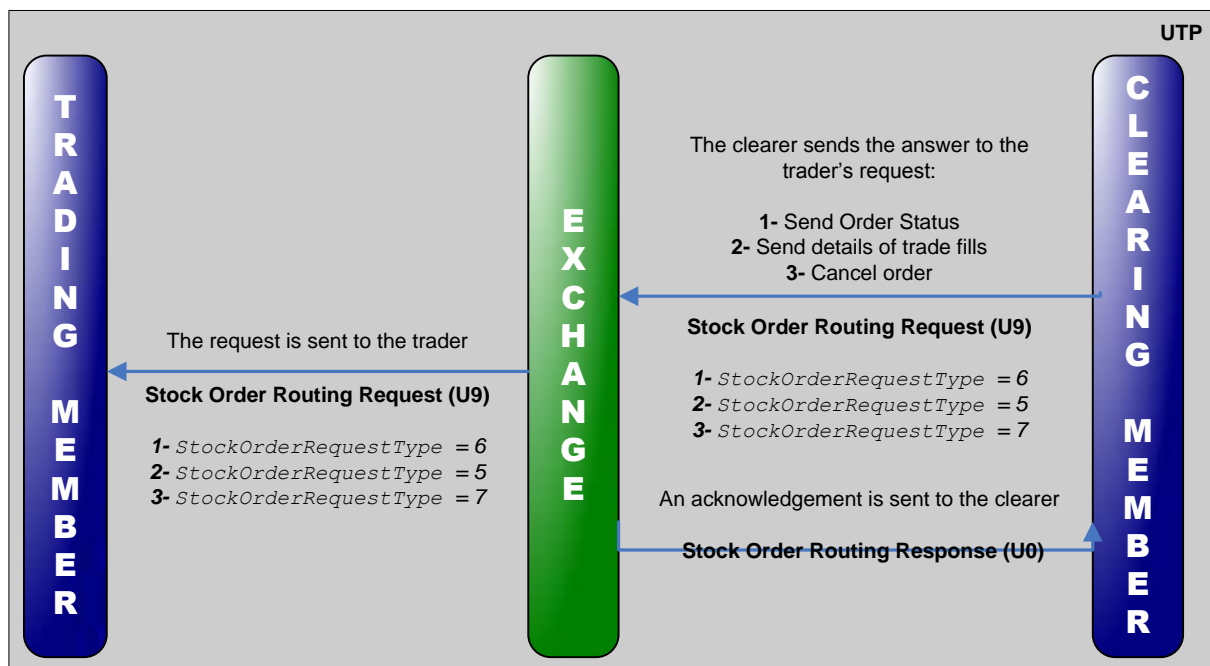
It is therefore important to distinguish the two types of messages exchanged between the trader or Market Maker and the clearing member: the request sent by the trader and the answer to the request sent by the counterparty.

The actual kinematics of the Stock Order Routing in UTP-DE are the following two messages:

- The **Stock Order Routing Request** ('U9')
- The **Stock Order Routing Response** ('U0')

The diagrams below shows how messages are routed in UTP-DE:





## 9.7 TRADE NOTIFICATION

### 9.7.1 Quantity fields

The order quantities and trade fills fields are defined by the FIX protocol as follows:

- **OrderQty (38)** : Initial order quantity.
- **CumQty (14)** : Total quantity filled, i.e. the sum of all *LastQty* from previous Execution Reports
- **LeavesQty (151)** : Quantity open for further execution, i.e. the residual active volume of the order.

In addition to these fields, an additional one has been introduced: **QtyDelta (8011)**.

Developers should apply the following rules when receiving Execution Reports to update quantity of orders and trade fills:

1. The absolute working volume desired for the order after the amendment. (Note that this is possible, if the working volume is reduced in this way, and the order trades or partially trades at the same time as the revision is submitted. The working volume left after the trade will not be reduced by the traded volume and this volume will, therefore, be subject to trading twice. This method was not recommended.
2. Trade deletions have NO effect on updating any volume fields for the related order.
3. *QtyDelta* must be used to track the change in volume of an order as a result of a revision. If the original volume is increased as a result of the revision, *QtyDelta* will be positive, other it will be negative.

The following example outlines the order quantity and trade fills after an order is submitted, partially trades. The order quantity is revised after the partial fill occurred.

A Buy order is submitted in a SEP2015 / MAR2016 calendar spread on the CAC40 Index Future contract at 0.65 for 50 lots.

```

MsgId           : 00 41           : 65 [D]
iLen            : 00 72           : 114
lSeqNum         : 00 00 00 05 : 5
ClOrdID         : 06 00 AA B9 : 100707001
Price           : 00 00 02 8A : 650
StopPx          : 00 00 00 00 : 0
ExpireDate      : 00 00 00 00 : 0
OrderQty       : 00 00 00 32 : 50
MinQty          : 00 00 00 00 : 0
ClearingInstruction : 00 00           : 0
TradingSessionID : 00                : 0
PartyRole       : 00                : 0
CustOrderCapacity : 00                : 0
SecondaryClOrdID :                   : ()
SecurityIDSource : 38                : 8
SecurityID       :                   : (JFFCE02E0065EES)
Account          :                   : ()
TradeInputSource : 00                : 
TradeInputDevice :                   : ()
OrdType          : 32                : 2
Side             : 31                : 1
TimeInForce      : 30                : 0
OrderOrigin      : 00                : 
AccountCode      : 48                : H
ClientInfo       :                   : ()
PartyID          :                   : ()
PostingAction    :                   : ()

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 04 : 4
OrderID         : 01 00 0B 01 00 0F B4 34 : 72069692961829940
ClOrdID         : 06 00 AA B9 : 100707001

```

The order is partially matched for 10 lots.

```

MsgId           : 00 A1           : 161 [8]
iLen            : 01 74           : 372
lSeqNum         : 00 00 00 06 : 6
OrderID         : 01 00 0B 01 00 0F B4 34 : 72069692961829940
TradeID         : 00 00 00 00 00 00 EC 5D : 60509
ExecID          : 01 00 00 00 00 0C 26 EE : 72057594038724334
ClOrdID         : 06 00 AA B9 : 100707001
OrigClOrdID     : 00 00 00 00 : 0
CrossID         : 00 00 00 00 : 0
ListID          : 00 00 00 00 : 0
TransactTime    : 07 05 19 AD : 117774765
ClOrdLinkID     : 00 00 00 00 : 0

```

```

LastPx           : 00 00 02 8A : 650
Price            : 00 00 02 8A : 650
StopPx           : 00 00 00 00 : 0
ReturnCode       : 00 F8 00 01 : 16252929
ExpireDate       : 00 00 00 00 : 0
CumQty           : 00 00 00 0A : 10
LastQty          : 00 00 00 0A : 10

```

→ LastQty contains the quantity of the trade fill

```

OrderQty         : 00 00 00 32 : 50
MinQty           : 00 00 00 00 : 0
LeavesQty        : 00 00 00 28 : 40

```

→ OrderQty = CumQty + LeavesQty

```

QtyDelta         : 00 00 00 00 : 0
MassStatusReqID  : 00 00 00 00 : 0
OtherLegLastPx   : 00 00 00 00 : 0
OrdRejReason     : 00 00       : 0
ClearingInstruction : 00 00       : 0
TradingSessionID : 00           : 0
CrossType        : 00           : 0
CustOrderCapacity : 00           : 0
PartyRole        : 00           : 0
ExecRefID        :              : ()
OrigCompID       :              : ()
LastRptRequested : 59           : Y
Text             :              : ()
SecondaryClOrdID  :              : ()
SecurityIDSource  : 38           : 8
SecurityID       :              : (JFFCE02E0065EES)
Account          :              : ()
PackageID        :              : ()
SecondaryOrderID  :              : (11029172)
RiskID           :              : ()
MatchingCode     :              : ()
TradeInputDevice :              : ()
OtherParty       :              : ()
OrdStatus        : 31           : 1
OrdType          : 32           : 2
Side             : 31           : 1
TimeInForce      : 30           : 0
ExecType         : 46           : F
OrderOrigin      : 20           : 
OrderCapacity     : 00           : 
TradeInputSource  : 55           : U
AccountCode      : 48           : H
ClientInfo       :              : ()
PartyID          :              : ()
PostingAction    :              : (OO)
OtherLegSecurityIDSource: 00       : 
OtherLegSecurityID :              : ()
OtherLegReferenceNo :              : ()
NoLegs           : 02           : 2
LegLastPx        : 00 00 10 E0 : 4320
LegLastQty       : 00 00 00 0A : 10
LegSecurityIDSource : 38           : 8
LegSecurityID     :              : (JFFCE150900000F)
LegLastPx        : 00 00 0E 56 : 3670
LegLastQty       : 00 00 00 0A : 10
LegSecurityIDSource : 38           : 8
LegSecurityID     :              : (JFFCE160300000F)

```

The order quantity is revised down to 25 lots (whose 10 have been already filled)

```

MsgId           : 00 71           : 113 [G]
iLen            : 00 38           : 56
lSeqNum         : 00 00 00 3F : 63
OrderID         : 00 00 00 00 00 00 00 00 : 0
ClOrdID        : 06 00 AA DF : 100707039
OrigClOrdID     : 06 00 AA B9 : 100707001
Price           : 00 00 02 94 : 660
StopPx          : 3B 9A C9 FF : 999999999
ExpireDate      : 05 F5 E0 FF : 99999999
OrderQty       : 00 00 00 19 : 25
SecurityIDSource : 50           : P
SecurityID      :              : (JFFCE)

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 0B : 11
OrderID         : 01 00 0B 01 00 0F B4 34 : 72069692961829940
TradeID         : 00 00 00 00 00 00 00 00 : 0
ExecID          : 01 00 00 00 00 0C 27 32 : 72057594038724402
ClOrdID        : 06 00 AA DF : 100707039
OrigClOrdID     : 06 00 AA B9 : 100707001
CrossID         : 00 00 00 00 : 0
ListID          : 00 00 00 00 : 0
TransactTime    : 07 34 9F 2B : 120889131
ClOrdLinkID     : 00 00 00 00 : 0
LastPx          : 00 00 00 00 : 0
Price           : 00 00 02 94 : 660
StopPx          : 3B 9A C9 FF : 999999999
ReturnCode      : 00 40 00 01 : 4194305
ExpireDate      : 00 00 00 00 : 0
CumQty        : 00 00 00 0A : 10
LastQty         : 00 00 00 00 : 0
OrderQty       : 00 00 00 19 : 25

```

→ The OrderQty field contains the new volume of the order following the revision, i.e.

25 lots. After the revision, the formula  $\text{OrderQty} = \text{LeavesQty} + \text{CumQty}$  still applies

```

MinQty          : 00 00 00 00 : 0
LeavesQty      : 00 00 00 0F : 15
QtyDelta       : FF FF FF E7 : -25

```

→ The QtyDelta indicates that the original volume of the order has decreased by 25 lots

```

MassStatusReqID : 00 00 00 00 : 0
OtherLegLastPx  : 00 00 00 00 : 0
OrdRejReason    : 00 00       : 0
ClearingInstruction : 00 00       : 0
TradingSessionID : 00          : 0
CrossType       : 00          : 0
CustOrderCapacity : 00          : 0
PartyRole       : 00          : 0
ExecRefID       :             : ()
OrigCompID      :             : ()
LastRptRequested : 59         : Y
Text            :             : ()
SecondaryClOrdID :             : ()
SecurityIDSource : 38         : 8
SecurityID      :             : (JFFCE02E0065EES)
Account         :             : ()
PackageID       :             : ()
SecondaryOrderID :             : ()
RiskID          :             : ()
MatchingCode    :             : ()
TradeInputDevice :             : ()

```

```

OtherParty      :      : ()
OrdStatus      : 31    : 1
OrdType        : 32    : 2
Side           : 31    : 1
TimeInForce    : 30    : 0
ExecType       : 35    : 5
OrderOrigin    : 20    :
OrderCapacity  : 00    :
TradeInputSource : 55    : U
AccountCode    : 48    : H
ClientInfo     :      : ()
PartyID        :      : ()
PostingAction  :      : (00)
OtherLegSecurityIDSource: 00 :
OtherLegSecurityID :      : ()
OtherLegReferenceNo :      : ()
NoLegs         : 00    : 0

```

- **The TransactTime field**

Developers should note that the TransactTime field is expressed in UTC, i.e. Universal Time Coordinated also known as 'GMT', in milliseconds since midnight.

- **The ExecID field**

The ExecIDs are unique per trading environment i.e. Equities and Financials. Indeed, the Trading HostID is embedded in the ExecIDs. Developers must also note that these are not reset every day.

---

## 9.7.2 Underlying traded volume algorithm

This paragraph explains the way the Trading Engine calculates the underlying traded volume when trading delta neutral trades. The general formula is as follows:

### 1- For an option hedged against its underlying:

$$\text{UnderlyingVolume} = \text{OptionVolume} * \Delta * \text{LottingFactor}$$

$$\text{LottingFactor} = \frac{\text{OptionLotSize}}{\text{UnderlyingLotSize}}$$

$$\Delta = \text{Delta of the option expressed as a percentage}$$

### 2- For an option strategy hedged against its underlying:

$$\text{UnderlyingVolume} = \text{OptionStrategy} * \Delta * \text{LottingFactor}$$

$$\text{LottingFactor} = \frac{\text{OptionLotSize}}{\text{UnderlyingLotSize}}$$

$$\Delta = \text{Delta of the option expressed as a percentage}$$

The Lot Size of an equity option represents the number of stocks in one lot of the option. For other Derivatives instruments, it represents a multiplier against a unit of the Underlying (most commonly set to 1). For stock underlyings the lot size is always 1. Therefore:

- If UnderlyingLotSize = OptionLotSize, the lotting factor will always be 1 and the underlying volume will be the option volume multiplied by the delta.
- If OptionLotSize > UnderlyingLotSize, the lotting factor will have the effect of multiplying the number of lots allocated to the underlying position.
- If UnderlyingLotSize > OptionLotSize, the lotting factor will have the effect of reducing the number of lots allocated to the underlying to ensure a delta neutral position.

The formula above can then be extended when taking into consideration partial fills. For each partial fill, the calculation of the underlying traded volume in a delta neutral trade is:

$$\text{Underlying\_Volume} = \text{round}(\text{OldOptionOrderVolume} * \Delta * \text{Lotting\_Factor}) - (\text{NewOptionOrderVolume} * \Delta * \text{Lotting\_Factor})$$

*OldOptionOrderVolume = active order quantity before partial trade*

*NewOptionOrderVolume = active order quantity after partial trade*

The rounding in this formula is to the nearest integer.

The following examples below illustrate this calculation in different scenarios.

### **Example 1: Index Option hedged using an Index Future**

In this example, the Index Option is being hedged against the related Index Future. Let us assume that the Index Option delta neutral order for 104 lots is traded wholly with a matching Index Option delta neutral order.

The Option has a lot size of 100 (i.e. one Option is based upon €100 per index point). The underlying leg of the trade is the Index Future which has a lot size of 200 (i.e. one Future is based upon €200 per index point). This gives a Lotting Factor of 0.5 (100/200).

The value of Delta is 0.48 and all 104 Options in the Central Order Book are traded, leaving no residual volume.

Therefore, using the underlying traded volume algorithm for delta neutrals:

$$\begin{aligned} \text{Underlying Traded Volume} &= \text{round}(104 \times 0.48 \times 0.5) - \text{round}(0 \times 0.48 \times 0.5) \\ &= \text{round}(24.96) - \text{round}(0) \\ &= 25 \end{aligned}$$

So 25 Futures are traded as the underlying leg of the delta neutral trade.

**Example 2: Financial Options order matched with multiple incoming orders (1)**

Let us assume that in a Financial Option delta neutral market, a single order is matched with multiple incoming orders.

The underlying of the Financial Option is the Future based upon the same interest rate. The Option has a lot size of 1 (i.e. one Option is based upon one Future). The Future also has a lot size of 1 hence the Lotting Factor is 1.

The value of Delta is 0.56 and the sequence begins with 300 Options in the Central Order Book.

As each new incoming order is matched with the resting order, values for the required underlying volume are calculated using the underlying traded volume algorithm. Since the first incoming order is for 50 Options:

$$\begin{aligned}\text{Underlying Traded Volume} &= \text{round}(300 \times 0.56 \times 1) - \text{round}(250 \times 0.56 \times 1) \\ &= \text{round}(168) - \text{round}(140) \\ &= 28\end{aligned}$$

For this trade no rounding is required and 50 Options and 28 Futures are traded with the resting order and deducted from its remaining volume.

This sequence continues as the resting order is matched with multiple incoming delta neutral orders:

RESTING ORDER		INCOMING ORDER			(OLD OPTION ORDER VOLUME X DELTA X UNDERLYING CONTRACT SIZE)		(NEW OPTION ORDER VOLUME X DELTA X UNDERLYING CONTRACT SIZE)		
RESIDUAL ORDER VOLUME	UNDERL. VOLUME	ORDER VOLUME	TRADED VOLUME	TRADED UNDERL. VOLUME	CALCULATED	ROUNDED	CALCULATED	ROUNDED	DELTA
300	168	50	50	28	168	168	140	140	0.56
250	140	25	25	14	140	140	126	126	0.56
225	126	30	30	17	126	126	109.2	109	0.57
195	109.2	30	30	17	109.2	109	92.4	92	0.57
165	92.4	70	70	39	92.4	92	53.2	53	0.56
95	53.2	60	60	33	53.2	53	19.6	20	0.55
35	19.6	24	24	14	19.6	20	6.16	6	0.58
11	6.16	50	11	6	6.16	6	0	0	0.55
				Total = 168					

The trades above show when rounding is used in the underlying traded volume algorithm. In these cases the delta for each trade may not be exactly 0.56, but the overall delta for the sequence is:

$$\begin{aligned}\text{Overall Delta} &= \text{Overall Underlying Volume Traded} / \text{Overall Option Volume Traded} \\ &= 168 / 300 \\ &= 0.56\end{aligned}$$



**Example 3: Multiple single volume orders resting in market with delta below 0.5**

This example is an unusual example of 100 separate Financial Option delta neutral orders resting in the market, each with an Option Volume of 1. The Option is the same as that in the previous examples but the value of Delta is now 0.49.

If a single order with Option Volume of 100 is submitted, the trader who submitted it will expect to trade all the Options volume resting in the Central Order Book, and 49 underlying Futures. But as the incoming order trades with each resting order, the underlying traded volume algorithm will calculate for each:

$$\begin{aligned}
 \text{Underlying Traded Volume} &= \text{round}(1 \times 0.49 \times 1) - \text{round}(0 \times 0.49 \times 1) \\
 &= \text{round}(0.49) - \text{round}(0) \\
 &= 0
 \end{aligned}$$

Therefore, the incoming order will trade with each resting order but no underlying volume will be traded between the resting order and the incoming orders.

---

**9.7.3 Specific case of EFP trades**

The algorithm to calculate the quantity price of the cash legs belonging to an EFP trade is very specific to this functionality. It is explained in details within the documentation dedicated to EFP. Also, quantity and price are rounded with up to 4 decimals.

The way UTP-DE reports the execution of an EFP trade to both counterparties is also very specific:

- First, an Execution Report (8) message is sent to report the trade on the EFP strategy,
- Then an Execution Report (8) message is sent to report the trade on the index future,
- Then an Execution Report (8) message is sent to report the trade on the index,
- Then an Execution Report (8) message is sent to report the trade on each leg composing the index. Note that the field 'LastRptRequested' will be set to 'N' until the last execution message is received for the EFP trade.

---

**9.8 MARKET MAKING FUNCTIONALITY**

UTP offers a set of specific features for Market Makers.

For a detailed functional overview of these features, please refer to the “*Functional Overview of UTP Market-Making functionalities*” document. See also the dedicated page on the Euronext website: [www.euronext.com/membership/liquidity-providers-and-market-makers](http://www.euronext.com/membership/liquidity-providers-and-market-makers).

---

**9.8.1 Mass quotes**

**Mass Quote (I)** messages allow a Market Maker to submit simultaneously batches of two-sided orders (bids and offers) into different series of the same contract in one single transaction.

The availability of Mass Quotes is configurable by contract and must be submitted using a Market Making ITM. Only ITMs that are registered to submit mass quotes in a contract are allowed to do so. Using a Market Making ITM, a trader can only submit non-resting orders in the classes for which the key has been set up. All orders in these classes are subject to the throttling. However, he can submit orders in other classes depending on his trading rights. These orders may or may be not subject to the throttling. Example: if a Market Maker uses a Market Making key to fulfill his Market Making obligations on CAC40 options, he can submit orders on the CAC40 Futures using the same key. Any order on CAC40 Options (MMOs or non-resting orders) will be subject to the throttling, however, orders on the Futures will not be subject to the throttling as it only applies to CAC40 Options.

Mass Quote batch size varies between contracts and Market Maker obligations. Liquidity Providers can get the list of contracts for which their Market Making key has permissions as well as the associated batch size by calling the **MM Configuration Status Request (U1)** message. The **MM Configuration Status Request Ack (U5)** message returns the list of SecurityGroups on which the Market Maker can submit quotes using this key as well as the associated batch size in the **NoBatchSize** field.

**Note:**

The **NoBatchSize** returns the number of **double-sided** quotes.

### **Mass Quotes submission**

Client applications should pay attention to the following rules when submitting quotes using the **Mass Quote** message:

- Mass Quotes do not persist in the order book if the ITM logs out or is disconnected.
- All quotes in a batch should be for the same commodity. If this is not the case, all quotes for the same contract as the first quote in the batch are accepted and the remainder is rejected.
- If the batch contains more pairs than allowed, then the entire batch is rejected.
- Mass quotes can be submitted as single-sided as well as double-sided quotes. This is determined by the value of the **SideRevised** field in the repeating leg of the message.
  - **SideRevised** should be set to '1' when only providing the Buy side
  - **SideRevised** should be set to '2' when only providing the Sell side
  - **SideRevised** should be set to '0' for submitting a double-sided quote.

When **SideRevised** = 1, the **SellPx** and **SellSize** fields (respectively when **SideRevised** = 2, the **BuyPx** and **BuySize** fields) must be populated even if their content is ignored by the Trading Engine.

- Orders in a **Mass Quote** message are processed sequentially, i.e. if two pairs are for the same instrument (AMR), then the second pair supersedes the first one.

Once in the market, each bid and offer that is part of a **Mass Quote** message is given its own individual Exchange OrderId, however returned either in the **BuyOrderID** or **SellOrderID** field. When the quote will match, the **OrderID** field sent in the trade notification Execution Report will contain this original **BuyOrderID** or **SellOrderID**.

### Mass Quotes revision

There are the important rules that developers should pay attention to when revising quotes. When a Mass Quote is submitted, it replaces any existing quote by that ITM in the following manner:

- Bids replace existing bids and offers replace existing offers, regardless of price.
- If either side of the new order has a volume of zero, all volume in that side of the order is pulled.
- If either side of the new order has a null volume, any volume in that side of the order is not altered.

This means that when users amend the volume of a quote, they change the **residual** volume. The volume specified in the `BidSize` and `SellSize` fields of the `Mass Quote` message therefore correspond to the absolute working volume the Market Maker wants to display in the market after the amendment, regardless of previously executed volume in the series. This is therefore different from when a single order is revised i.e. the original volume is modified.

If a user enters a quote for 100 lots that trades 50 lots, and then resubmits the quote at a different price for a volume of 100 lots, this will result in an absolute working volume of 100 lots after the revision.

**Note:**

Mass Quotes can only be revised by entering a new batch of quotes. Using the standard `Order Revision Request` message will result in the revision attempt being rejected, i.e. the quotes remain unchanged.

### 9.8.2 Underlying traded volume algorithm

In addition to Mass Quotes, registered Liquidity Providers can use any of the Market Maker Protection facilities activated per contract:

- **Delta Protection.** This facility offers Market Makers a degree of protection against being traded on multiple quotes simultaneously.

After each MMO trade, the Trading Host recalculates the cumulative Delta Position of the Market Maker.

- In case of an options trade, it is based on the option delta calculated by the Exchange, as follows:

$$\text{NewDeltaPosition} = \text{CurrentDeltaPosition} +/- (\text{OptionDelta} * \text{NbLots} * \text{LotSize})$$

The +/- sign is >0 for Buy Call or Sell Put and <0 for Sell Call or Buy Put.

Note that if no option delta can be calculated, then a value of 0.5 will be used.

For a delta-neutral strategy trade, the delta is assumed to be zero and so no delta position update is made.

- In case of an futures trade, it is based on the following formula:

$$\text{NewDeltaPosition} = \text{CurrentDeltaPosition} +/- (\text{NbLots} * \text{LotSize})$$

The +/- sign is >0 for Buy Future and <0 for Sell Future.

For example, a Market Maker configures a Delta Protection Limit to 100. Trades executed during the uncrossing cause the Market Maker's cumulative Delta Position to be updated to 110. No breach action occurs at this point. A subsequent trade of delta -1 causes their position to be updated to 109 and at this point the breach action occurs.

- **Vega Protection.** Vega Protection is intended to protect Market Makers in scenarios where machines trade against machines resulting in many trades in a small period of time without a big change in delta.

After each MMO trade, the Trading Host recalculates the cumulative Vega Position of the Market Maker as follows:

$$\text{New N Position} = \text{Current N Position} \pm (\text{N Option} \times \text{Option Volume Traded}) \times \text{LotSize}$$

- **Volume Protection.** The Volume Protection functionality is intended to protect market makers in scenarios where multiple trades result in small net changes of delta or vega position.

After each MMO trade, the Trading Host recalculates the cumulative Volume Position of the Market Maker as follows:

- For Options:

$$\text{New V Position} = \text{Current V Position} + \text{Volume Traded} \times \text{LotSize}$$

+: for both buying and selling Call or Put options

- For Futures:

$$\text{New V Position} = \text{Current V Position} + \text{Volume Traded} \times \text{LotSize}$$

+: for both buying or selling Futures

**Note:**

For the time being, only the Delta and Volume protection facilities are available to registered Market Makers. Please refer to the Market Makers Guideline.

## 9.9 BATCHING OF ORDERS

There are a number of contracts, mainly those where Mass Quotes are not available, for which developers can take advantage of a batching facility using the **New Order List (E)** and **Order Revision List (UA)** messages. Up to 16 orders may be submitted in a New Order List message, and up to 32 orders can be revised in an Order Revision List message. The following main validation checks occur when submitting multiple orders:

- commodity homogeneity: orders within the list must be for the same SecurityGroup,
- price type – it is not possible to submit a Market order in a list,
- crossing prevention.

These are concerned primarily with the integrity of the batch submission message. Further validation is then undertaken on each individual order as they are processed.

Note that the New Order List and the Order Revision List messages are only available in the CCG Binary interface.

In the example below, an Order List is submitted on the CAC40 Index Future (SecurityGroup = FCE). There are 5 orders included in the list on five different expiries.

```

MsgId           : 00 51           : 81 [E]
iLen            : 02 2C           : 556
lSeqNum         : 00 00 00 0B : 11
ListID          : 00 00 56 CE : 22222
NoOrders      : 05           : 5
Filler1[3]     :                :

ClOrdID       : 01 CA B3 03 : 30061315 → Order 1
Price           : 00 01 7C DC : 97500
ExpireDate      : 00 00 00 00 : 0
OrderQty        : 00 00 00 0F : 15
SecurityIDSource : 38           : 8
SecurityID      :                : (JFFCE151200000F)
OrdType         : 32           : 2
Side            : 31           : 1
TimeInForce     : 30           : 0
Filler[2]       :                :

ClOrdID       : 01 CA B3 04 : 30061316 → Order 2
Price           : 00 01 7F CA : 98250
ExpireDate      : 00 00 00 00 : 0
OrderQty        : 00 00 00 14 : 20
SecurityIDSource : 38           : 8
SecurityID      :                : (JFFCE151200000F)
OrdType         : 32           : 2
Side            : 32           : 2
TimeInForce     : 30           : 0
Filler[2]       :                :

ClOrdID       : 01 CA B3 05 : 30061317 → Order 3
Price           : 00 01 7D D6 : 97750
ExpireDate      : 00 00 00 00 : 0
OrderQty        : 00 00 00 14 : 20
MinQty          : 00 00 00 00 : 0
SecurityIDSource : 38           : 8
SecurityID      :                : (JFFCE160300000F)
OrdType         : 32           : 2
Side            : 31           : 1
TimeInForce     : 30           : 0
Filler[2]       :                :

ClOrdID       : 01 CA B3 06 : 30061318 → Order 4
Price           : 00 01 81 B2 : 98738
ExpireDate      : 00 00 00 00 : 0
OrderQty        : 00 00 00 0F : 15
SecurityIDSource : 38           : 8
SecurityID      :                : (JFFCE161200000F)
OrdType         : 32           : 2
Side            : 31           : 1
TimeInForce     : 30           : 0
Filler[2]       :                :

ClOrdID       : 01 CA B3 07 : 30061319 → Order 5
Price           : 00 01 86 8C : 99980
ExpireDate      : 00 00 00 00 : 0
OrderQty        : 00 00 00 0A : 10
SecurityIDSource : 38           : 8
SecurityID      :                : (JFFCE160900000F)
OrdType         : 32           : 2

```

```

Side           : 31           : 1
TimeInForce    : 30           : 0
Filler[2]      :              :

```

Each order within the list will be acknowledged separately. The first 3 orders are successfully accepted, an Order Ack message is sent for each of them

```

MsgId          : 00 91          : 145 [a]
iLen           : 00 14          : 20
lSeqNum        : 00 00 00 03 : 3
OrderID        : 01 00 0B 00 00 0E 65 4F : 72069688666776911
ClOrdID       : 01 CA B3 03 : 30061315

```

```

MsgId          : 00 91          : 145 [a]
iLen           : 00 14          : 20
lSeqNum        : 00 00 00 04 : 4
OrderID        : 01 00 0B 00 00 0E 65 50 : 72069688666776912
ClOrdID       : 01 CA B3 04 : 30061316

```

```

MsgId          : 00 91          : 145 [a]
iLen           : 00 14          : 20
lSeqNum        : 00 00 00 05 : 5
OrderID        : 01 00 0B 00 00 0E 65 51 : 72069688666776913
ClOrdID       : 01 CA B3 05 : 30061317

```

Order 4 is rejected because of an Invalid Tick Size (ReturnCode = 4194462). The rejection is notified via an Execution Report.

```

MsgId          : 00 A1          : 161 [8]
iLen           : 01 44          : 324
lSeqNum        : 00 00 00 06 : 6
OrderID        : 00 00 00 00 00 00 00 00 : 0
TradeID        : 00 00 00 00 00 00 00 00 : 0
ExecID         : 01 00 00 00 00 0A 5E F9 : 72057594038607609
ClOrdID       : 01 CA B3 06 : 30061318
OrigClOrdID    : 00 00 00 00 : 0
TransactTime   : 07 3C 3E B1 : 121388721
Price          : 00 01 81 B2 : 98738
ReturnCode    : 00 40 00 9E : 4194462
ExpireDate     : 00 00 00 00 : 0
CumQty         : 00 00 00 00 : 0
LastQty        : 00 00 00 00 : 0
OrderQty       : 00 00 00 0F : 15
LeavesQty      : 00 00 00 0F : 15
OrdRejReason  : 01 3E          : 318
LastRptRequested : 4E          : N
Text           :              : ()
SecurityIDSource : 38          : 8
SecurityID      :              : (JFFCE161200000F)
OrdStatus     : 38          : 8
OrdType        : 32          : 2
Side           : 31          : 1
TimeInForce    : 30          : 0
ExecType      : 38          : 8

```

Order 5 is successfully accepted and immediately trades. This is notified by an Order Ack (a) message then an Execution Report.

```

MsgId           : 00 91           : 145 [a]
iLen            : 00 14           : 20
lSeqNum         : 00 00 00 07 : 7
OrderID         : 01 00 0B 00 00 0E 65 52 : 72069688666776914
ClOrdID        : 01 CA B3 07 : 30061319

MsgId           : 00 A1           : 161 [8]
iLen            : 01 44           : 324
lSeqNum         : 00 00 00 08 : 8
OrderID         : 01 00 0B 00 00 0E 65 52 : 72069688666776914
TradeID         : 00 00 00 00 00 00 B8 27 : 47143
ExecID          : 01 00 00 00 00 0A 5E FB : 72057594038607611
ClOrdID        : 01 CA B3 07 : 30061319
OrigClOrdID     : 00 00 00 00 : 0
TransactTime    : 07 3C 3E B1 : 121388721
LastPx          : 00 01 86 8C : 99980
Price           : 00 01 86 8C : 99980
ReturnCode      : 00 F8 00 01 : 16252929
CumQty          : 00 00 00 0A : 10
LastQty         : 00 00 00 0A : 10
OrderQty        : 00 00 00 0A : 10
LeavesQty       : 00 00 00 00 : 0
LastRptRequested : 59           : Y
Text            :                : ( )
SecurityIDSource : 38           : 8
SecurityID       :                : (JFFCE160900000F)
OrdStatus      : 32           : 2
OrdType          : 32           : 2
Side             : 31           : 1
TimeInForce      : 30           : 0
ExecType       : 46           : F

```

## 10. MARKET CONTROL

### 10.1 MARKET STATUS

When an application detects a change in the Market State via the XDP **Market Status** (752) message, it should intelligently interpret the change in the state of the market for the following reasons:

- If the market is transitioning from the pre-open or open state to the closed state, the Trading Engine will remove all Day orders in that market from the order book. The application developer should ensure this is replicated on the front-end application. Note that the CCG will also send **Execution Reports** for all the pulled orders with **ExecType** = 'Done for Day'.
- If the market is transitioning from any state to the Terminated state, the Trading Engine will remove all orders, including GTCs, in that market from the order book. The application developer should ensure this is replicated on the front-end application. Note that the CCG will also send **Execution Reports** for all the pulled orders with **ExecType** = 'Done for Day'.

'Terminate' status can only be un-done by an "Un-terminate" status.

The application developer should always ensure that the front-end user is aware of the current Market State for a given market.

The Trading Host will attempt to send Market State at the highest level (as defined by the **SecurityIdSource** parameter). Application developers should ensure that their applications are able to interpret Market State information at different levels and apply these state changes to their market structures.

The Market State as sent via the XDP **Market Status** (752) message is a combination of different individual Market Modes. The table below shows the most important combinations, especially the different market phases that a product may go through.

Developers must note that the Halted (6), Dark Series (28), Light Series (29), Trading Unhalt (30), Expire (39) and Pre-Expiry (40) are for future use.



		MARKET MODES																							
		Closed	Exit Extend Open	Halted	Open	Pre Closed	Pre Open	Price Limits Enabled	Price Limits Disabled	Restricted Open	Session 1	Session 2	Session 3	Quote Width Exemption 1	Quote Width Exemption 2	Quote Width Exemption 3	Dark Series	Light Series	Trading Unhalt	Terminate	Underminate	Expire	Pre-Expiry	Hold	Unhold
		1	4	6	7	8	9	10	11	12	13	14	15	23	24	25	28	29	30	31	32	39	40	41	42
MARKET STATE	Closed	✓						✓			✓			✓							✓				✓
	Pre-Open						✓	✓			✓			✓							✓				✓
	Open				✓			✓			✓			✓							✓				✓
	Open (optional session 1)				✓			✓			✓			✓							✓				✓
	Open (optional session 2)				✓			✓				✓		✓							✓				✓
	Open (optional session 3)				✓			✓					✓	✓							✓				✓
	Restricted Open (optional)				✓			✓		✓	✓			✓							✓				✓
	Pre-Close					✓		✓			✓			✓							✓				✓
	Close with Exit Extend (Optional)	✓	✓					✓			✓			✓							✓				✓
	Close with Block Extend Open (optional)	✓						✓			✓			✓							✓				✓
	Close	✓						✓			✓			✓							✓				✓
	Close with Price Limits Disabled	✓							✓		✓			✓							✓				✓
	Terminated (optional)	✓						✓			✓			✓						✓					✓
	Halted (optional)	✓						✓			✓			✓							✓			✓	

---

## 10.2 FORCED LOGOUT / LOCKOUT

The application developer should always ensure that the trader knows when an Exchange official has removed him from the market, as defined by the reception of a **User Notification (CB)** message. Exchange officials can either decide to 'logout' or 'lockout' the ITM. Developers can differentiate between the two cases by looking at the `UserStatus` field.

In the event that the trader is logout, only the Day orders will be pulled by the Trading Engine when all orders including GTCs will be pulled if the trader is locked out. If the ITM is authorised to connect to the market the same trading day, the CCG will send the Execution Reports notifying him that some (or all) of his orders have been cancelled. Developers might decide to wait until the reconnection to update the trader's order book or pull the orders accordingly upon reception of the `User Notification (UC)` message. If the first option is chosen, developers are required to implement a temporary solution to inform the trader in his front-end application, this upon reception of the `User Notification` message.

---

## 10.3 TRADER EXCHANGE SUSPEND

During the trading day, Market Control can block an ITM's access to products on the market. When a trader is suspended all his orders including GTCs, for contracts within the chosen logical exchange (example: Exchange Code "A"), will be pulled. Such changes must be properly reflected in the front-end application.

Any further orders submitted for that logical Exchange will be rejected. The trader is not logged out, but is notified of the change in status via a **User Notification (CB)** message with `UserStatus = '80'`

The suspension of an ITM can be lifted by being cancelled by Market Control, or by the suspension lapsing automatically at the end of the trading day.

As for a Forced Logout / Lockout, Execution Reports relative to the pulling of the ITM's orders will be sent to the user only when he will be authorised to reconnect and if this happens before the end of the trading day. However, developers are recommended to pull the trader's order book upon reception of the `User Notification` message or to find an alternative solution to inform the trader in his front-end application.

---

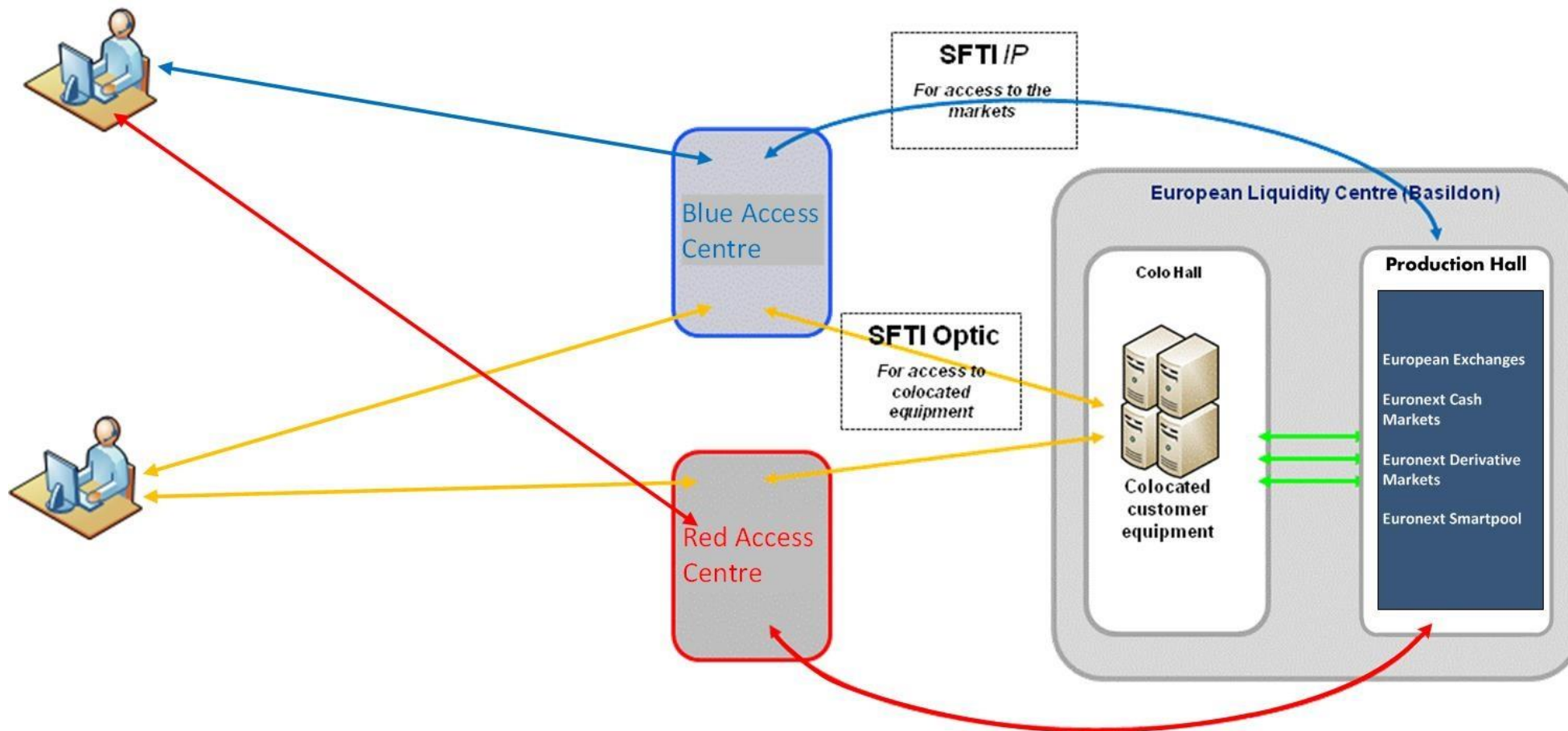
### 10.3.1 Market Control Messages

The **User Notification (CB)** message can be used by the Matching Engine to update a user regarding their connectivity. It can also be used by Market Control to send messages to market participants.

Market Control can transmit a message, containing text to client Applications. Messages can be sent in other languages, where this functionality is made available by the Exchange.

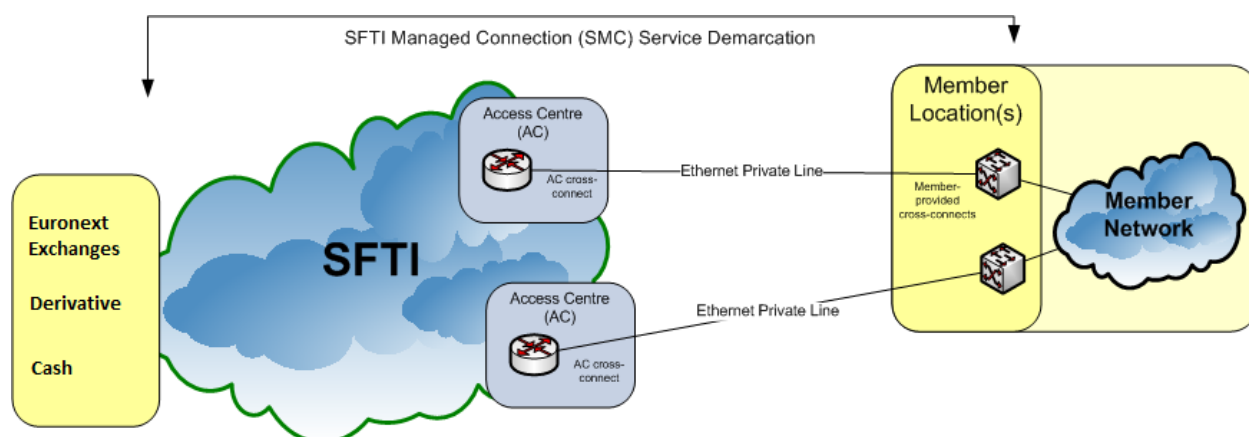
Messages can be sent to the following:

- All market participants via the XDP Exchange Message (761) or
- Individual ITMs via the **User Notification (CB)** message.

**APPENDIX A: 2\*100 MB RESILIENT INSTALLATION**

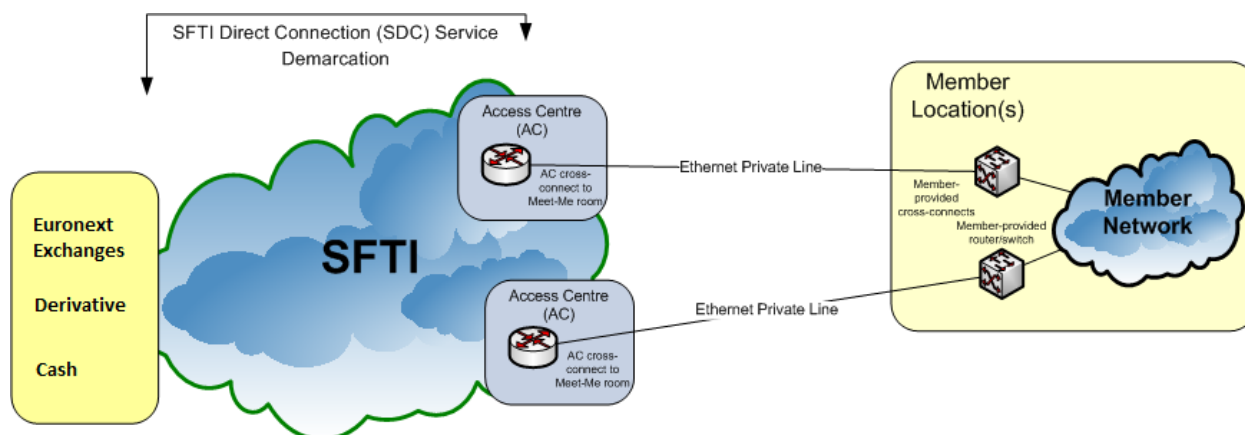
## APPENDIX B: SFTI® CONNECTIVITY ACCESS SOLUTIONS

### B.1 SFTI® MANAGED CONNECTION (SMC)



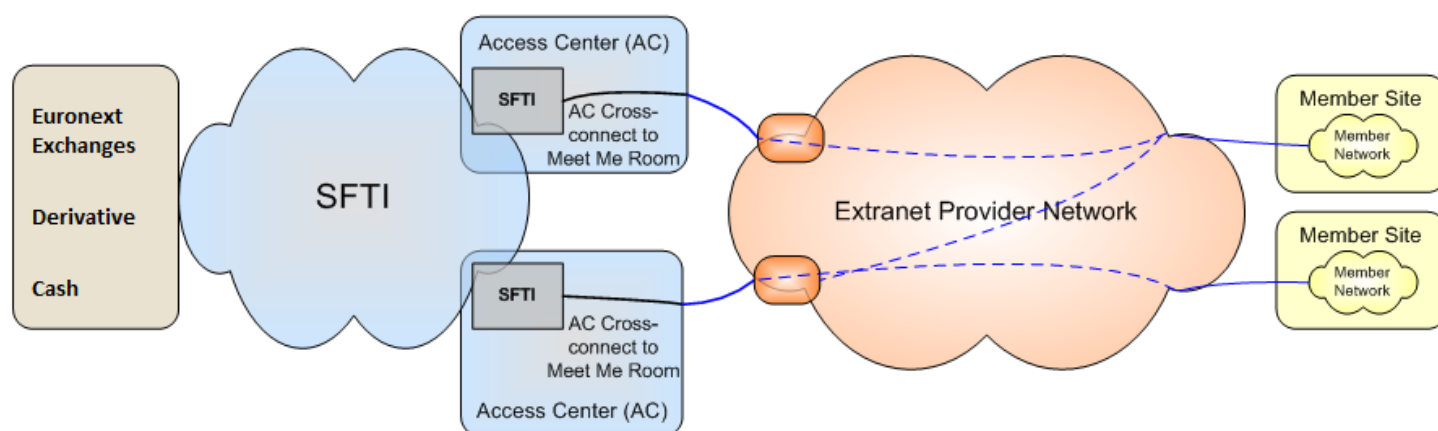
Member Model	SFTI Ports	AC Cross Connects	Ethernet circuits	Member site Cross Connects	Member Premises Equipment
<b>SFTI Managed Connection</b>	Included	Included	Included	Member Responsibility	Included

### B.2 SFTI® DIRECT CONNECTION (SDC)



Member Model	SFTI Ports	AC Cross Connects	Ethernet circuits	Member site Cross Connects	Member Premises Equipment
<b>SFTI Direct Connection</b>	Included	Member to provide to MMR	Member Responsibility	Member Responsibility	Member Responsibility

### B.3 EXTRANET SERVICE PROVIDER (ESP)

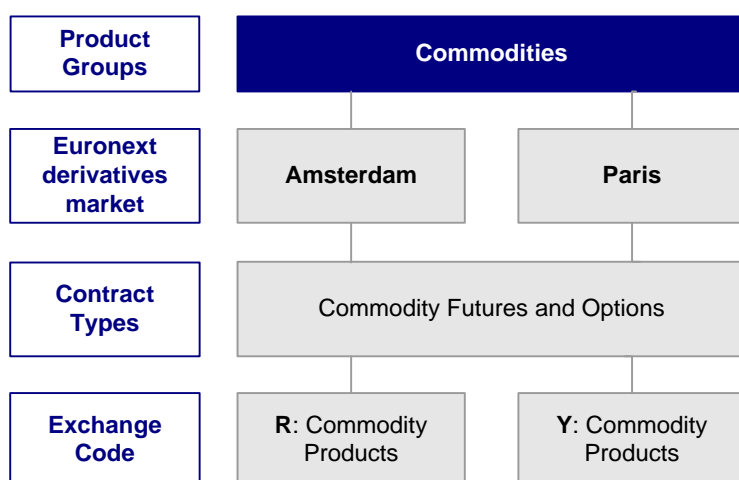


Member Model	SFTI Ports	AC Cross Connects	Ethernet circuits	Member site Cross Connects	Member Premises Equipment
Extranet Service Provider	N/A	N/A	N/A	N/A	Extranet or Member Responsibility

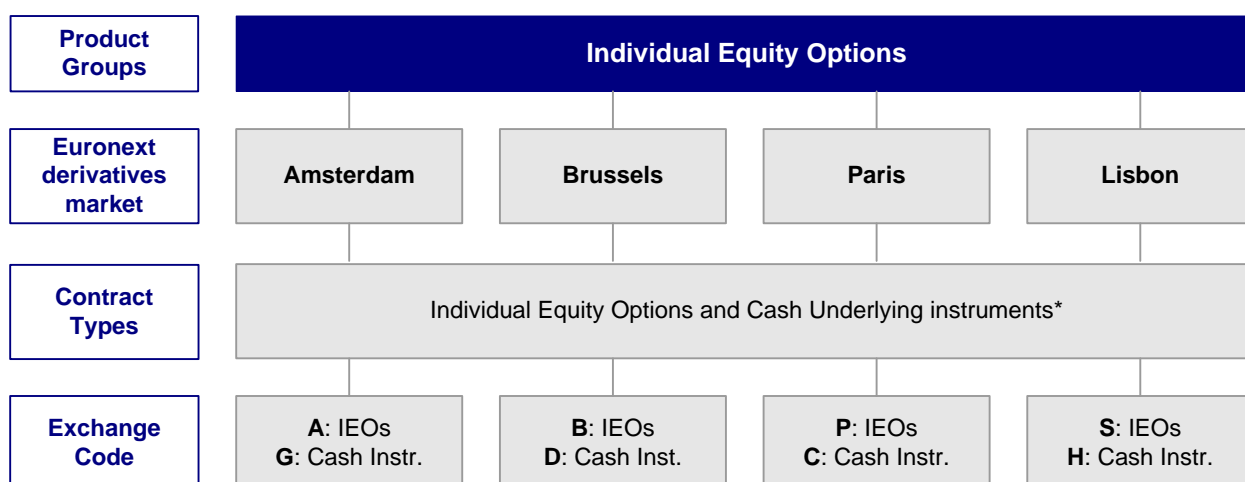
## APPENDIX C: EURONEXT INSTRUMENTS

The following tables provide a general overview of the way instruments are organised within the Trading Environment.

### C.1 COMMODITY PRODUCTS

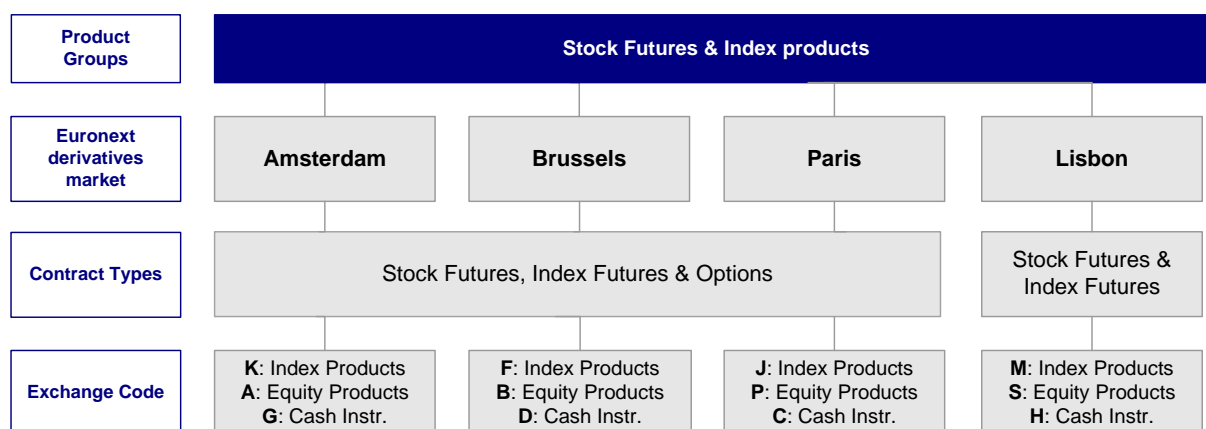


### C.2 EQUITY DERIVATIVES PRODUCTS: INDIVIDUAL EQUITY OPTIONS



\* Cash Underlying instruments are non tradable, listed only contracts

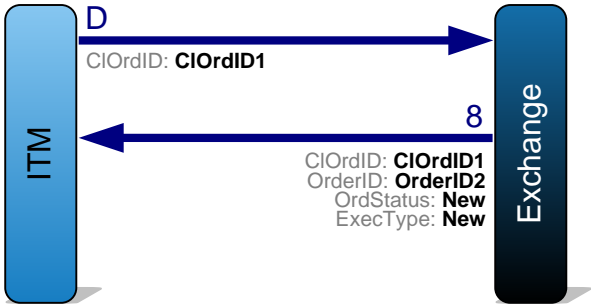
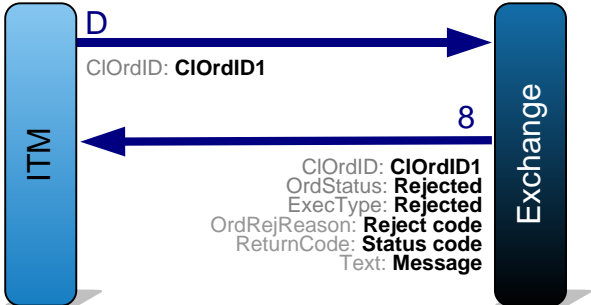
### C.3 EQUITY DERIVATIVES PRODUCTS: INDEX PRODUCTS AND STOCK FUTURES



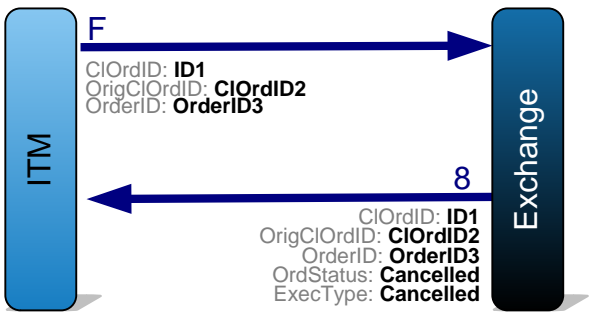
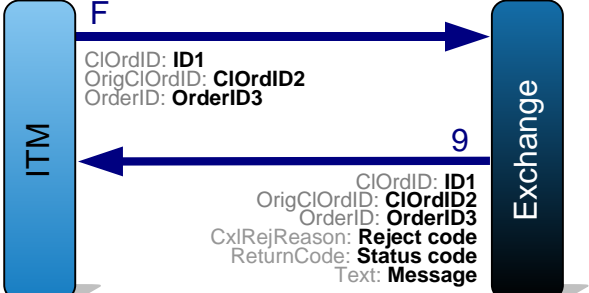
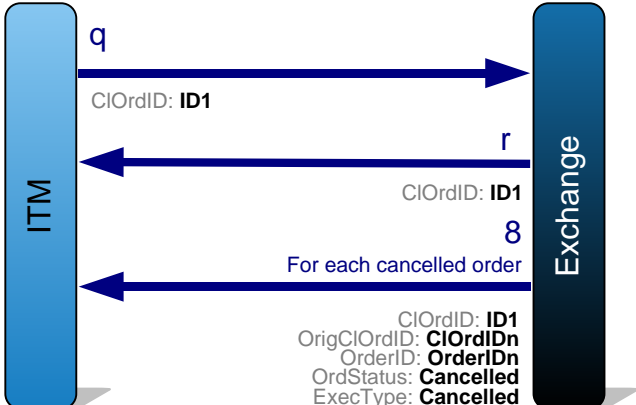
## APPENDIX D: CCG KINEMATICS

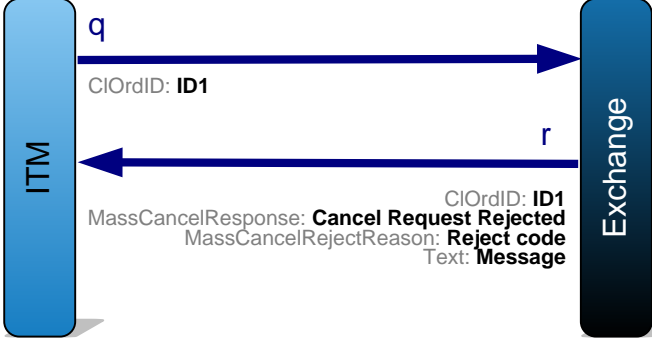
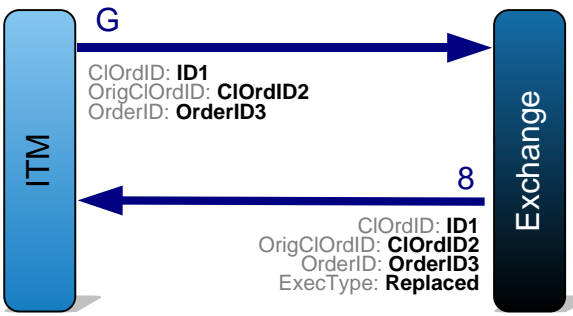
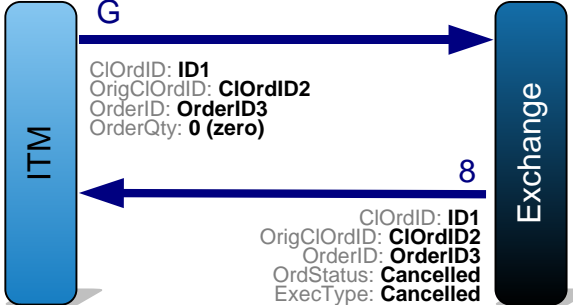
This section provides kinematics between members and the CCG.

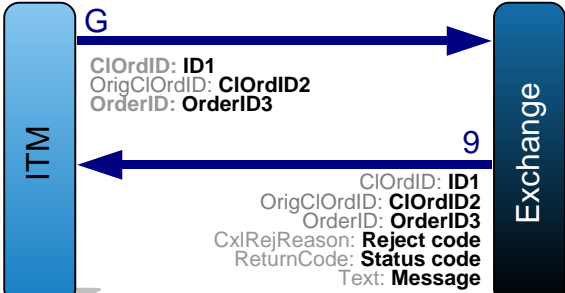
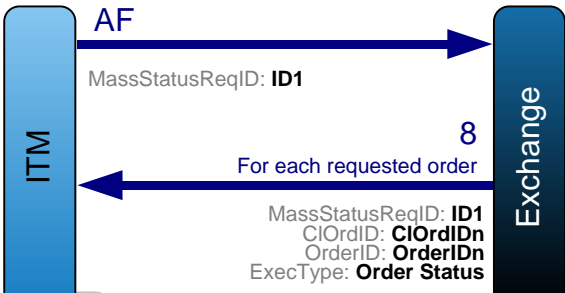
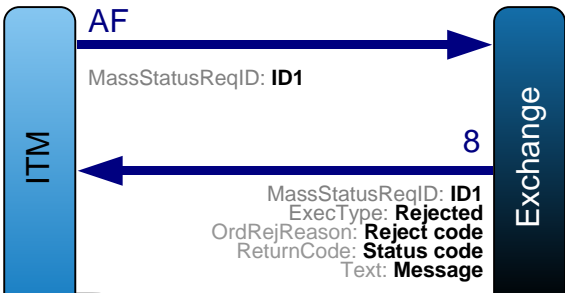
### D.1 APPLICATION MESSAGE RESPONSES

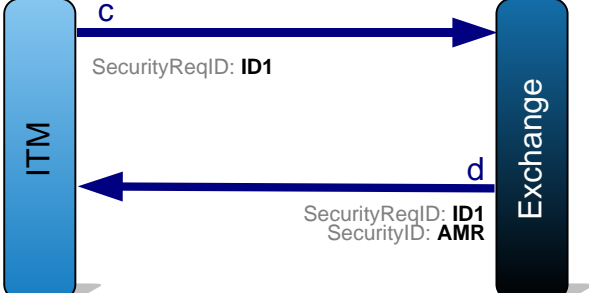
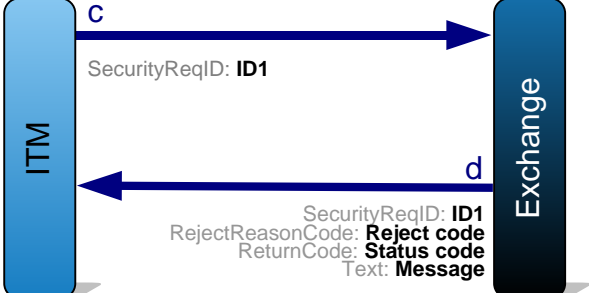
Client Message	Response Messages
New Order Single (D)	<p>If the new order is accepted then the Exchange responds with an <b>Execution Report (8)</b> which has the OrdStatus = New and ExecType = New.  The ClOrdID contains the Client's identifier for the order.  The Execution Report includes the OrderID allocated by the Exchange.  The OrderID or the ClOrdID must be used on subsequent requests regarding the order.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: D ClOrdID: ClOrdID1     Exchange--&gt;&gt;ITM: 8 ClOrdID: ClOrdID1 OrderID: OrderID2 OrdStatus: New ExecType: New </pre> <p>If the new order is rejected then the Exchange responds with an <b>Execution Report (8)</b> which has OrdStatus = Rejected and ExecType = Rejected.  The OrdRejReason contains a code for the rejection reason. If the OrdRejReason is set to 'Other', the status code is given in ReturnCode.  When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: D ClOrdID: ClOrdID1     Exchange--&gt;&gt;ITM: 8 ClOrdID: ClOrdID1 OrdStatus: Rejected ExecType: Rejected OrdRejReason: Reject code ReturnCode: Status code Text: Message </pre>

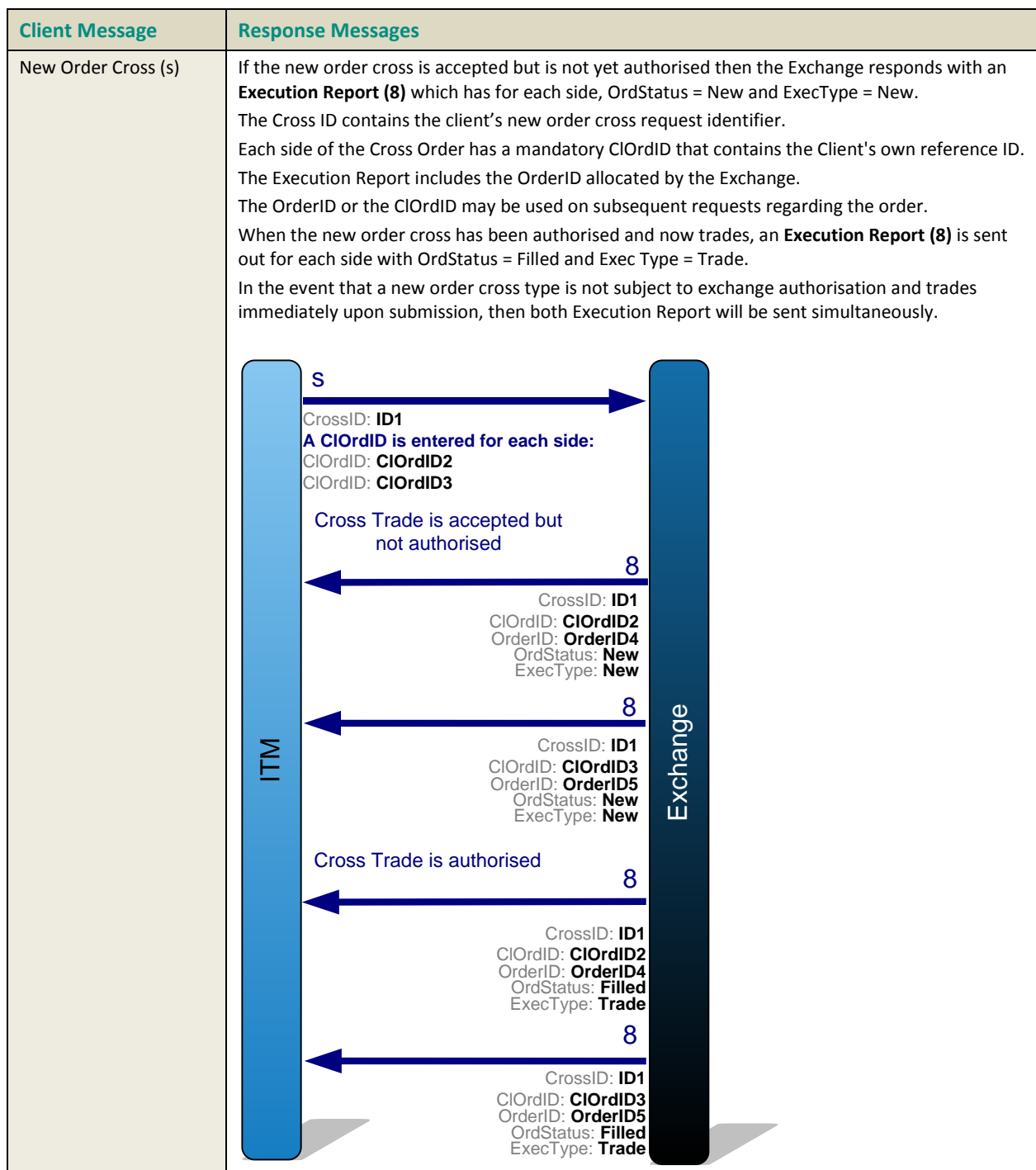


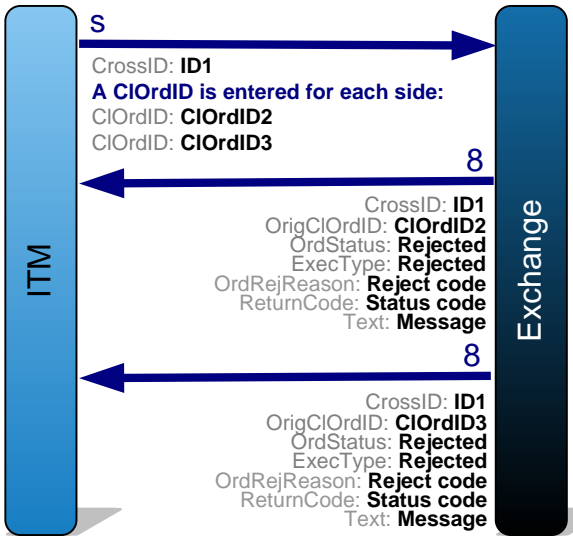
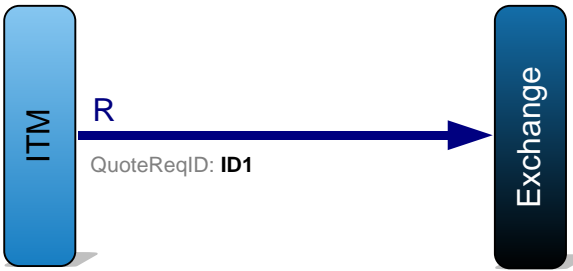
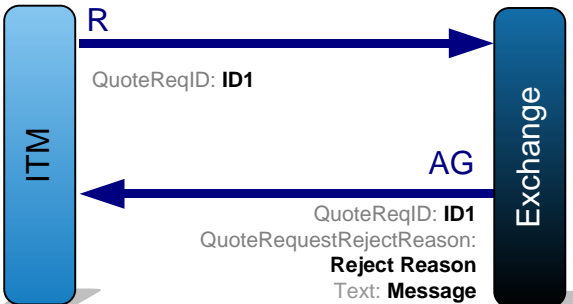
Client Message	Response Messages
<p>Order Cancel Request (F)</p>	<p>If the cancellation is accepted then the Exchange responds with an <b>Execution Report (8)</b> which has ExecType = Cancelled and OrdStatus = Cancelled.</p> <p>The OrderID or the OrigClOrdID is used to identify the order to be cancelled.</p> <p>The ClOrdID contains the client order cancel request ID.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: F ClOrdID: ID1 OrigClOrdID: ClOrdID2 OrderID: OrderID3     Exchange--&gt;&gt;ITM: 8 ClOrdID: ID1 OrigClOrdID: ClOrdID2 OrderID: OrderID3 OrdStatus: Cancelled ExecType: Cancelled </pre> <p>If the cancellation is rejected then the Exchange responds with an <b>Order Cancel Reject (9)</b> message.</p> <p>The CxlRejReason contains a code for the rejection reason. If the CxlRejReason is set to 'Other', the status code is given in ReturnCode.</p> <p>When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: F ClOrdID: ID1 OrigClOrdID: ClOrdID2 OrderID: OrderID3     Exchange--&gt;&gt;ITM: 9 ClOrdID: ID1 OrigClOrdID: ClOrdID2 OrderID: OrderID3 CxlRejReason: Reject code ReturnCode: Status code Text: Message </pre>
<p>Mass Cancel Request (q)</p>	<p>If the mass cancellation is accepted then the Exchange responds with an <b>Order Mass Cancel Report (r)</b> and an <b>Execution Report (8)</b> for each cancelled order which has an OrdStatus = Cancelled and ExecType = Cancelled.</p> <p>The ClOrdID contains the client mass cancel request ID.</p> <p>The OrigClOrdID or OrderID identifies the order.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: q ClOrdID: ID1     Exchange--&gt;&gt;ITM: r ClOrdID: ID1     Exchange--&gt;&gt;ITM: 8 For each cancelled order ClOrdID: ID1 OrigClOrdID: ClOrdIDn OrderID: OrderIDn OrdStatus: Cancelled ExecType: Cancelled </pre>

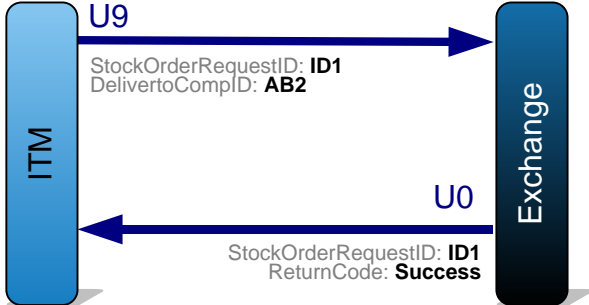
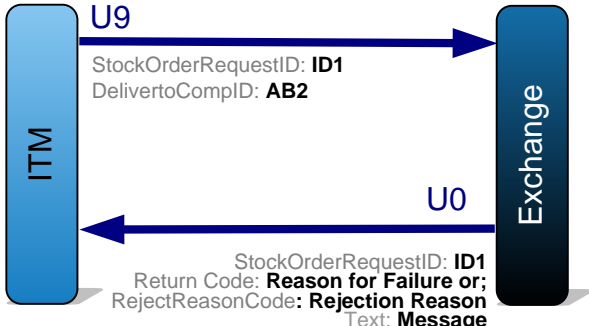
Client Message	Response Messages
	<p>If the mass cancellation is rejected then the Exchange responds with an <b>Order Mass Cancel Report (r)</b> message which has a MassCancelResponse indicating that the message has failed. The MassCancelRejectReason contains a code for the rejection reason. When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: q CIOrdID: ID1     Exchange--&gt;&gt;ITM: r CIOrdID: ID1 MassCancelResponse: Cancel Request Rejected MassCancelRejectReason: Reject code Text: Message   </pre>
Order Revision Request (G)	<p>If the revision is accepted then the Exchange responds with an <b>Execution Report (8)</b> which has ExecType = Replaced. The OrigCIOrdID or OrderID identifies each cancelled order. The CIOrdID contains the client order revision request ID.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: G CIOrdID: ID1 OrigCIOrdID: CIOrdID2 OrderID: OrderID3     Exchange--&gt;&gt;ITM: 8 CIOrdID: ID1 OrigCIOrdID: CIOrdID2 OrderID: OrderID3 ExecType: Replaced   </pre> <p>If the OrderQty is revised to 0 (zero) this is treated as a Cancellation request.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: G CIOrdID: ID1 OrigCIOrdID: CIOrdID2 OrderID: OrderID3 OrderQty: 0 (zero)     Exchange--&gt;&gt;ITM: 8 CIOrdID: ID1 OrigCIOrdID: CIOrdID2 OrderID: OrderID3 OrdStatus: Cancelled ExecType: Cancelled   </pre>

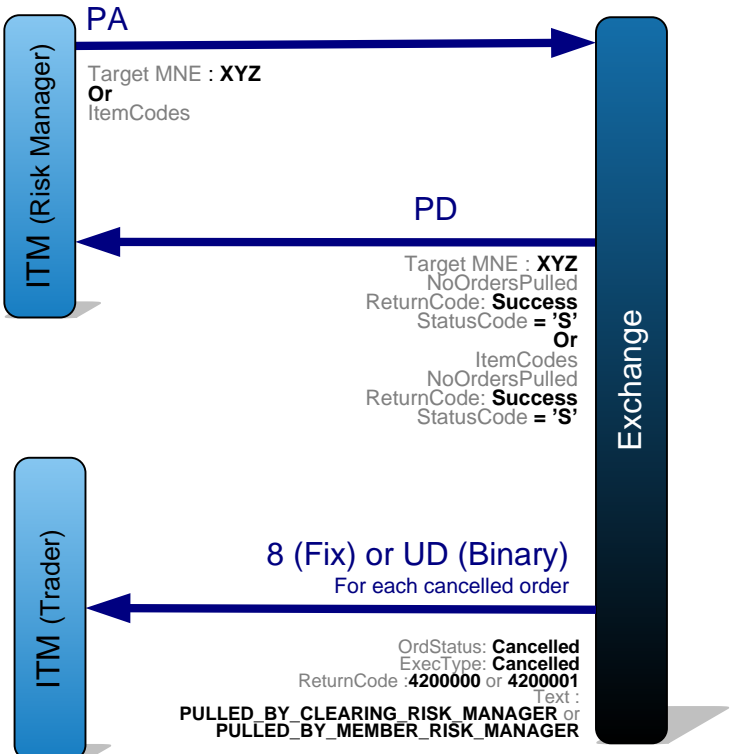
Client Message	Response Messages
	<p>If the revision is rejected then the Exchange responds with an <b>Order Cancel Reject (9)</b> message. The OrderID contains either the Order ID allocated by the Exchange or the word 'NONE'.</p> <p>The CxlRejReason contains a code for the rejection reason. If the CxlRejReason is set to 'Other', the status code is given in the ReturnCode.</p> <p>When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: G CfOrdID: ID1 OrigCfOrdID: CfOrdID2 OrderID: OrderID3     Exchange--&gt;&gt;ITM: 9 CfOrdID: ID1 OrigCfOrdID: CfOrdID2 OrderID: OrderID3 CxlRejReason: Reject code ReturnCode: Status code Text: Message   </pre>
<p>Order Mass Status Request (AF)</p>	<p>If the mass status request is accepted then the Exchange responds with an <b>Execution Report (8)</b> which has an ExecType = Order Status for each resting order. The CfOrdID or OrderID returned in the Execution Report identifies each order. The MassStatusReqID contains the client order mass status request ID.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: AF MassStatusReqID: ID1     Exchange--&gt;&gt;ITM: 8 For each requested order MassStatusReqID: ID1 CfOrdID: CfOrdIDn OrderID: OrderIDn ExecType: Order Status   </pre> <p>If the mass status request is rejected then the Exchange responds with an <b>Execution Report (8)</b> with the ExecType = Rejected. The OrdRejReason contains a code for the rejection reason. If the OrdRejReason is set to 'Other', the status code is given in the ReturnCode. When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: AF MassStatusReqID: ID1     Exchange--&gt;&gt;ITM: 8 MassStatusReqID: ID1 ExecType: Rejected OrdRejReason: Reject code ReturnCode: Status code Text: Message   </pre>

Client Message	Response Messages
Strategy Definition Request (c)	<p>If the Strategy definition request is accepted then the Exchange responds with a <b>Security Definition (d)</b></p> <p>The SecurityReqID contains the client's security definition request ID.</p> <p>The Security ID contains the AMR (Automated Market Reference) of the new instrument.</p> <p>Participants are notified of the new strategy market on the XDP market data interface.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: c SecurityReqID: ID1     Exchange--&gt;&gt;ITM: d SecurityReqID: ID1 SecurityID: AMR </pre>
	<p>If the Strategy definition request is rejected then the Exchange responds with a <b>Security Definition (d)</b></p> <p>The RejectReasonCode contains a code for the rejection reason. If the RejectReasonCode is set to 'Other', the status code is given in the ReturnCode.</p> <p>When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     ITM-&gt;&gt;Exchange: c SecurityReqID: ID1     Exchange--&gt;&gt;ITM: d SecurityReqID: ID1 RejectReasonCode: Reject code ReturnCode: Status code Text: Message </pre>

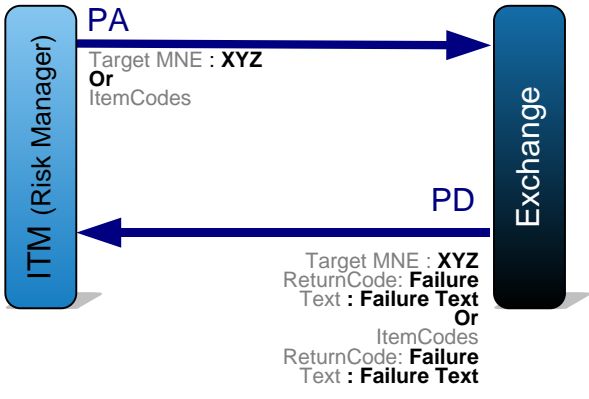
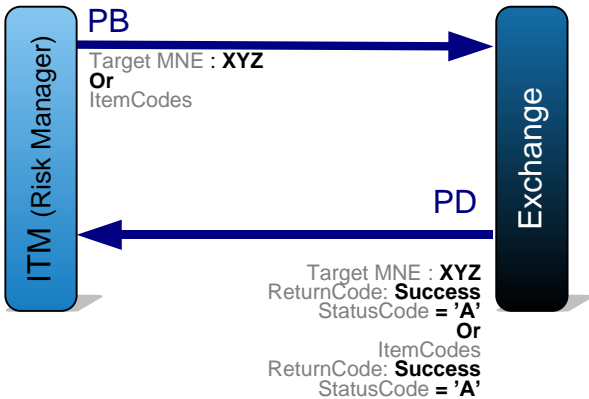


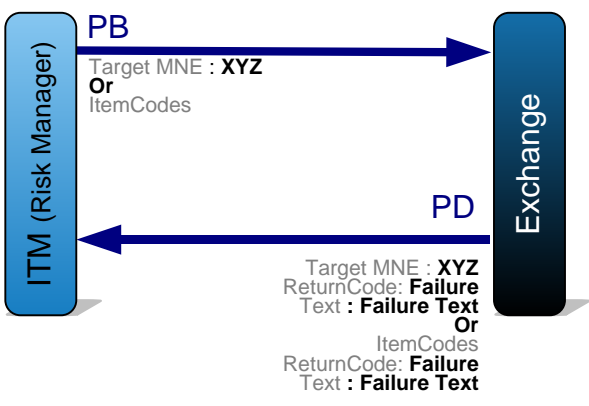
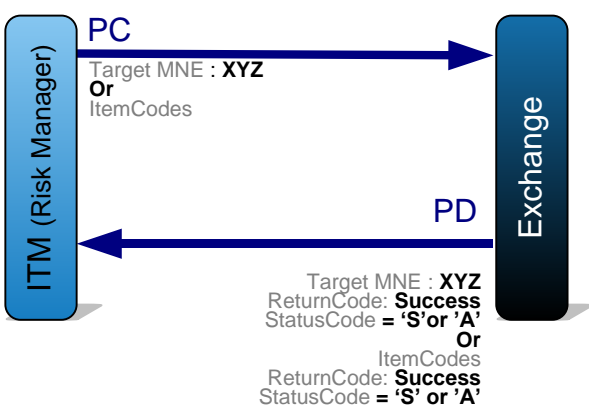
Client Message	Response Messages
<p>New Order Cross (s)</p>	<p>If the new order cross is rejected then the Exchange responds with an <b>Execution Report (8)</b> which has for each side OrdStatus = Rejected and ExecType = Rejected.</p> <p>The new order cross may be rejected because it either:</p> <ul style="list-style-type: none"> <li>failed the Matching Engine validation</li> <li>it was subject to Market Operations authorisation and was subsequently rejected.</li> </ul> <p>The OrdRejReason contains a code for the rejection reason. If the OrdRejReason is set to 'Other', the status code is given in the ReturnCode.</p> <p>When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     Note over ITM: S     ITM-&gt;&gt;Exchange: CrossID: ID1 A CIOrdID is entered for each side: CIOrdID: CIOrdID2 CIOrdID: CIOrdID3     Note over Exchange: 8     Exchange--&gt;&gt;ITM: CrossID: ID1 OrigCIOrdID: CIOrdID2 OrdStatus: Rejected ExecType: Rejected OrdRejReason: Reject code ReturnCode: Status code Text: Message     Note over Exchange: 8     Exchange--&gt;&gt;ITM: CrossID: ID1 OrigCIOrdID: CIOrdID3 OrdStatus: Rejected ExecType: Rejected OrdRejReason: Reject code ReturnCode: Status code Text: Message   </pre>
<p>Quote Request (R)</p>	<p>If the quote request is accepted then the Exchange will not reply with an acknowledgement message to the ITM.</p> <p>Market Makers will be notified of the quote request via the XDP Market Update messages. The new series will only be available on subsequent XDP Market Update messages when Market Makers have provided quotes or orders have been placed.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     Note over ITM: R     ITM-&gt;&gt;Exchange: QuoteReqID: ID1   </pre> <p>If the quote request is rejected then the Exchange responds with a <b>Quote Request Reject (AG)</b> message.</p> <p>The QuoteReqID is used to identify the rejected Quote Request.</p> <p>The QuoteRequestRejectReason contains a code for the rejection reason.</p> <p>When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     Note over ITM: R     ITM-&gt;&gt;Exchange: QuoteReqID: ID1     Note over Exchange: AG     Exchange--&gt;&gt;ITM: QuoteReqID: ID1 QuoteRequestRejectReason: Reject Reason Text: Message   </pre>

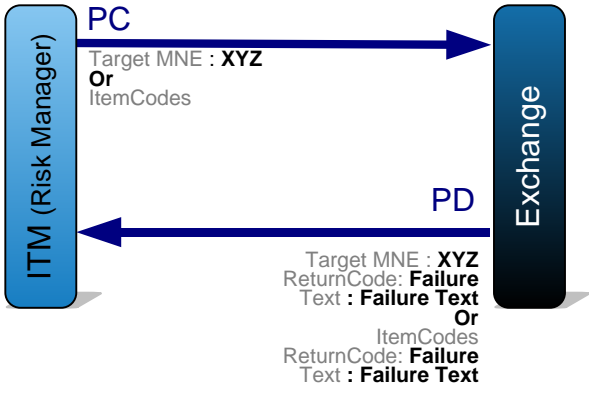
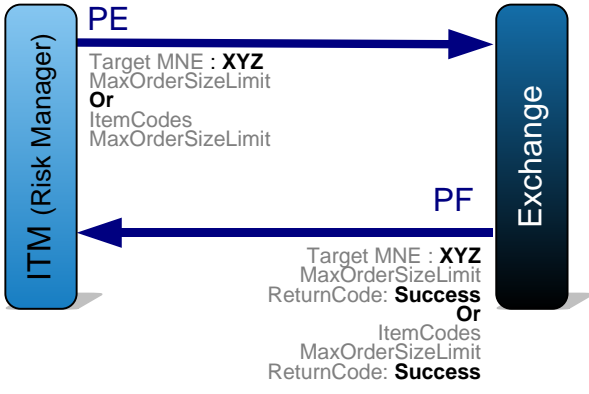
Client Message	Response Messages
Stock Order Routing Request (U9)	<p>If stock order routing request is successfully sent onto the recipient (identified by DeliverToCompID) then the Exchange responds with a <b>Stock Order Routing Response (U0)</b> with Return Code = Success.</p> <p>The StockOrderRequestID is allocated by the Client to identify the Stock Order Routing Request.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     Note over ITM: U9     ITM-&gt;&gt;Exchange: StockOrderRequestID: ID1 DeliverToCompID: AB2     Note over Exchange: U0     Exchange--&gt;&gt;ITM: StockOrderRequestID: ID1 ReturnCode: Success           </pre>
	<p>If stock order routing request is not successfully sent to the recipient then the Exchange responds with a <b>Stock Order Routing Response (U0)</b> with either the Return Code or RejectReasonCode showing the reason for the failure.</p> <p>When necessary the Exchange will also provide an explanation in the Text.</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     Note over ITM: U9     ITM-&gt;&gt;Exchange: StockOrderRequestID: ID1 DeliverToCompID: AB2     Note over Exchange: U0     Exchange--&gt;&gt;ITM: StockOrderRequestID: ID1 Return Code: Reason for Failure or; RejectReasonCode: Rejection Reason Text: Message           </pre>

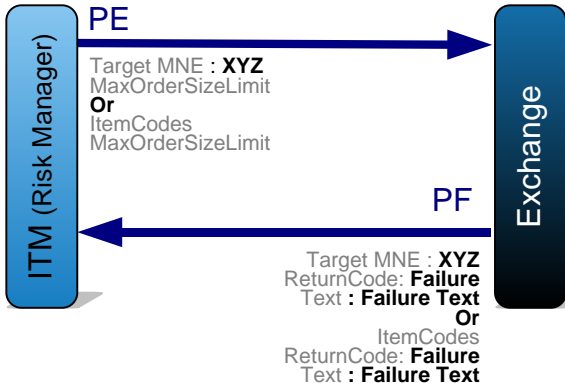
Client Message	Response Messages
PTRM Suspend (PA)	<p>If the PTRM Suspend Message (PA) is successfully received and processed, the Exchange responds with a PTRM Status Response (PD) including the status of the member or traders.</p> <p>Note: The PTRM Suspend Message (PA) can be sent either at member level (i.e. all ITMs of a Target MNE) or for a subset list of ITMs of the Target MNE. The corresponding PTRM Status Response (PD) message is returned at the same level, i.e. on Target MNE level or for the ITM list respectively.</p> <p>Please note that one PD message will be sent back per Trading Unit on which the Member or ITM is present.</p> <p>In addition, when the Exchange proceeds in pulling the orders of the Target MNE or ITMs, one cancellation message per order to the Impacted traders by the Exchange. For traders connected in the FIX protocol, they will receive Execution Report (8) messages while those in the binary protocol will receive Cancel Notification List message (UD) messages:</p>  <pre> sequenceDiagram     participant ITM_RM as ITM (Risk Manager)     participant Exchange     participant ITM_T as ITM (Trader)      ITM_RM-&gt;&gt;Exchange: PA Target MNE : XYZ Or ItemCodes     Exchange--&gt;&gt;ITM_RM: PD Target MNE : XYZ NoOrdersPulled ReturnCode: Success StatusCode = 'S' Or ItemCodes NoOrdersPulled ReturnCode: Success StatusCode = 'S'     Exchange--&gt;&gt;ITM_T: 8 (Fix) or UD (Binary) For each cancelled order OrdStatus: Cancelled ExecType: Cancelled ReturnCode: 4200000 or 4200001 Text : PULLED_BY_CLEARING_RISK_MANAGER or PULLED_BY_MEMBER_RISK_MANAGER   </pre>




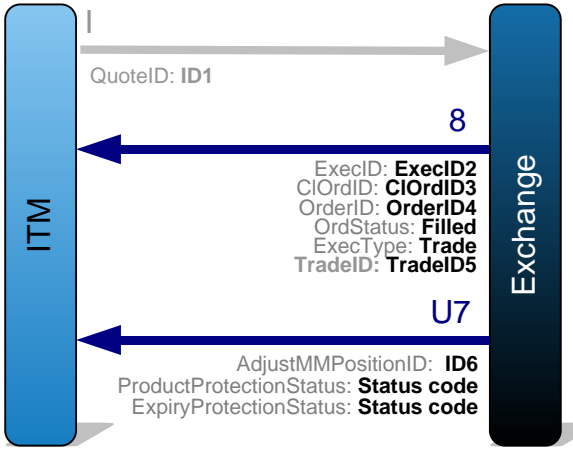

Client Message	Response Messages
	<p>If the PTRM Suspend Message (PA) is rejected by the Exchange, the Exchange replies with a PTRM Status Response (PD) message with a ReturnCode and a Text both explaining the reason of the failure.</p> <p>Note: The PTRM Suspend Message (PA) can be sent either for a given Target MNE or for a list of ITMs. The corresponding PTRM Status Response (PD) message is returned at the same level, i.e. on Target MNE level or per ITM respectively.</p> <p>Please note that one PD message will be sent back per Trading Unit on which the Member or ITM is present.</p>  <pre> sequenceDiagram     participant ITM as ITM (Risk Manager)     participant Exchange     ITM-&gt;&gt;Exchange: PA Target MNE : XYZ Or ItemCodes     Exchange--&gt;&gt;ITM: PD Target MNE : XYZ ReturnCode: Failure Text: Failure Text Or ItemCodes ReturnCode: Failure Text: Failure Text </pre>
PTRM Unsuspend (PB)	<p>If the PTRM Unsuspend Message (PB) is successfully received and processed, the Exchange responds with a PTRM Status Response (PD) including the status of the member or traders.</p> <p>Note: The PTRM Unsuspend Message (PB) can be sent either at the Member Level (i.e. all ITMs of a Target MNE) or for a subset list of ITMs of the Target . The corresponding PTRM Status Response (PD) message is returned at the same level, i.e. on Target MNE level or per ITM respectively.</p> <p>Please note that one PD message will be sent back per Trading Unit on which the Member or ITM is present.</p>  <pre> sequenceDiagram     participant ITM as ITM (Risk Manager)     participant Exchange     ITM-&gt;&gt;Exchange: PB Target MNE : XYZ Or ItemCodes     Exchange--&gt;&gt;ITM: PD Target MNE : XYZ ReturnCode: Success StatusCode = 'A' Or ItemCodes ReturnCode: Success StatusCode = 'A' </pre>


Client Message	Response Messages
	<p>If the PTRM Get Status Message (PC) is successfully received and processed, the Exchange responds with a PTRM Status Response (PD) including the status of the member or traders.</p> <p>Note: The PTRM Get Status Message (PB) can be sent either at the Member Level (i.e. all ITMs of a Target MNE) or for a subset list of ITMs of the Target MNE.. The corresponding PTRM Status Response (PD) message is returned at the same level, i.e. on Target MNE level or per ITM respectively.</p> <p>Please note that one PD message will be sent back per Trading Unit on which the Member or ITM is present.</p>  <pre> sequenceDiagram     participant ITM as ITM (Risk Manager)     participant Exchange     ITM-&gt;&gt;Exchange: PB Target MNE : XYZ Or ItemCodes     Exchange--&gt;&gt;ITM: PD Target MNE : XYZ ReturnCode: Failure Text : Failure Text Or ItemCodes ReturnCode: Failure Text : Failure Text   </pre>
PTRM Get Status (PC)	<p>If the PTRM Get Status Message (PC) is successfully received and treated, the Exchange responds with a PTRM Status Response (PD) including the status of the member or traders (S(uspended) or A(ctive)).</p> <p>Note: The PTRM Get Status Message (PB) can be sent either at for a given Target MNE or the Risk Manager can specify a list of ITMs. The corresponding PTRM Status Response (PD) message is returned at the same level, i.e. on Target MNE level or per ITM respectively.</p> <p>Please note that one PD message will be sent back per Trading Unit on which the Member or ITM is present.</p>  <pre> sequenceDiagram     participant ITM as ITM (Risk Manager)     participant Exchange     ITM-&gt;&gt;Exchange: PC Target MNE : XYZ Or ItemCodes     Exchange--&gt;&gt;ITM: PD Target MNE : XYZ ReturnCode: Success StatusCode = 'S' or 'A' Or ItemCodes ReturnCode: Success StatusCode = 'S' or 'A'   </pre>

Client Message	Response Messages
	<p>If the PTRM Get Status Message (PC) is rejected by the Exchange, the Exchange responds with a PTRM Status Response (PD).</p> <p>Note: The PTRM Get Status Message (PB) can be sent either at for a given Target MNE or the Risk Manager can specify a list of ITMs. The corresponding PTRM Status Response (PD) message is returned at the same level, i.e. on Target MNE level or per ITM respectively.</p> <p>Please note that one PD message will be sent back per Trading Unit on which the Member or ITM is present.</p>  <pre> sequenceDiagram     participant ITM as ITM (Risk Manager)     participant Exchange     ITM-&gt;&gt;Exchange: PC Target MNE : XYZ Or ItemCodes     Exchange--&gt;&gt;ITM: PD Target MNE : XYZ ReturnCode: Failure Text : Failure Text Or ItemCodes ReturnCode: Failure Text : Failure Text   </pre>
PTRM Set Order Size Limit (PE)	<p>If the PTRM Set Order Size Limit Message (PE) is successfully received and processed, the Exchange responds with a PTRM Set Order Size Limit Response (PF) including the Order Size Limit for the member or traders at the Exchange Code, Contract Type or contract level.</p> <p>Note: The PTRM Set Order Size Limit Message (PE) can be sent either at the Member Level (i.e. all ITMs of a Target MNE) or for a subset list of ITMs of the Target MNE. The corresponding PTRM Set Order Size Limit Response (PF) message is returned at the same level, i.e. on Target MNE level or per ITM respectively.</p> <p>Please note that one PF message will be sent back per Trading Unit on which the Member or ITM is present.</p>  <pre> sequenceDiagram     participant ITM as ITM (Risk Manager)     participant Exchange     ITM-&gt;&gt;Exchange: PE Target MNE : XYZ MaxOrderSizeLimit Or ItemCodes MaxOrderSizeLimit     Exchange--&gt;&gt;ITM: PF Target MNE : XYZ MaxOrderSizeLimit ReturnCode: Success Or ItemCodes MaxOrderSizeLimit ReturnCode: Success   </pre>

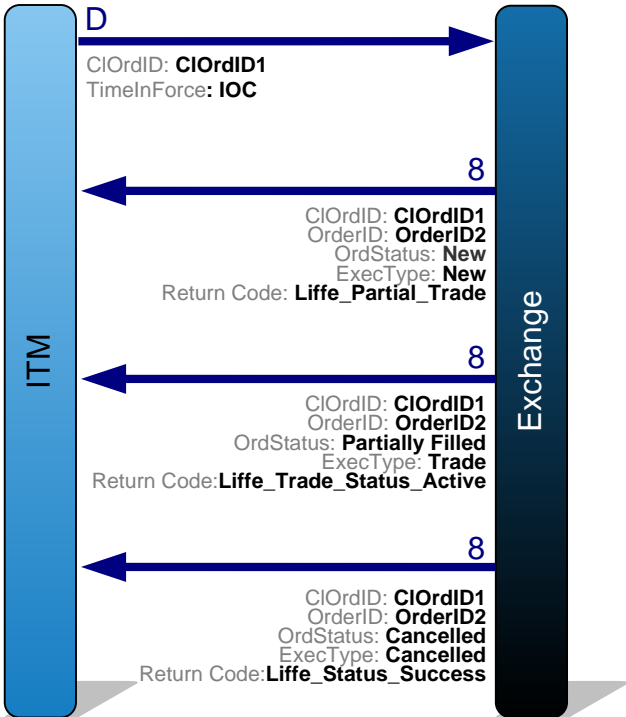
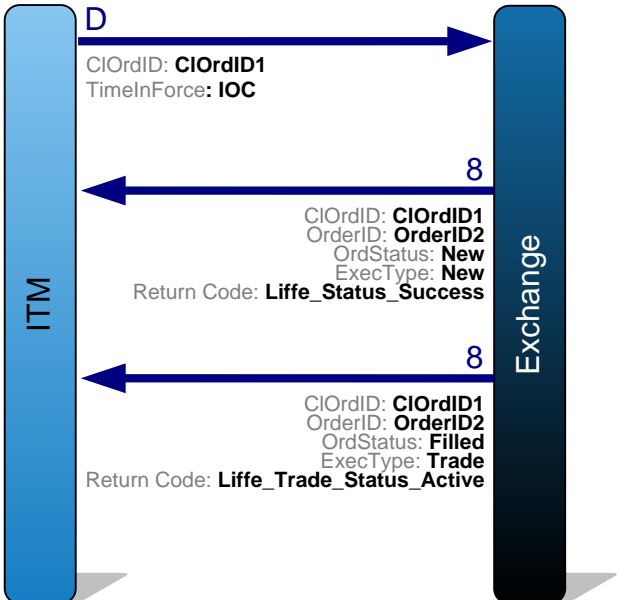
Client Message	Response Messages
	<p>If the PTRM Set Order Size Limit (PE) is rejected by the Exchange, the Exchange responds with a PTRM Set Order Size Limit Response (PF).</p> <p>Note: The PTRM Set Order Size Limit (PE) can be sent either at for a given Target MNE or the Risk Manager can specify a list of ITMs. The corresponding PTRM Set Order Size Limit Response (PF) message is returned at the same level, i.e. on Target MNE level or per ITM respectively.</p> <p>Please note that one PF message will be sent back per Trading Unit on which the Member or ITM is present.</p>  <pre> sequenceDiagram     participant ITM as ITM (Risk Manager)     participant Exchange     ITM-&gt;&gt;Exchange: PE Target MNE : XYZ MaxOrderSizeLimit Or ItemCodes MaxOrderSizeLimit     Exchange--&gt;&gt;ITM: PF Target MNE : XYZ ReturnCode: Failure Text : Failure Text Or ItemCodes ReturnCode: Failure Text : Failure Text   </pre>

## D.2 TRADE NOTIFICATION

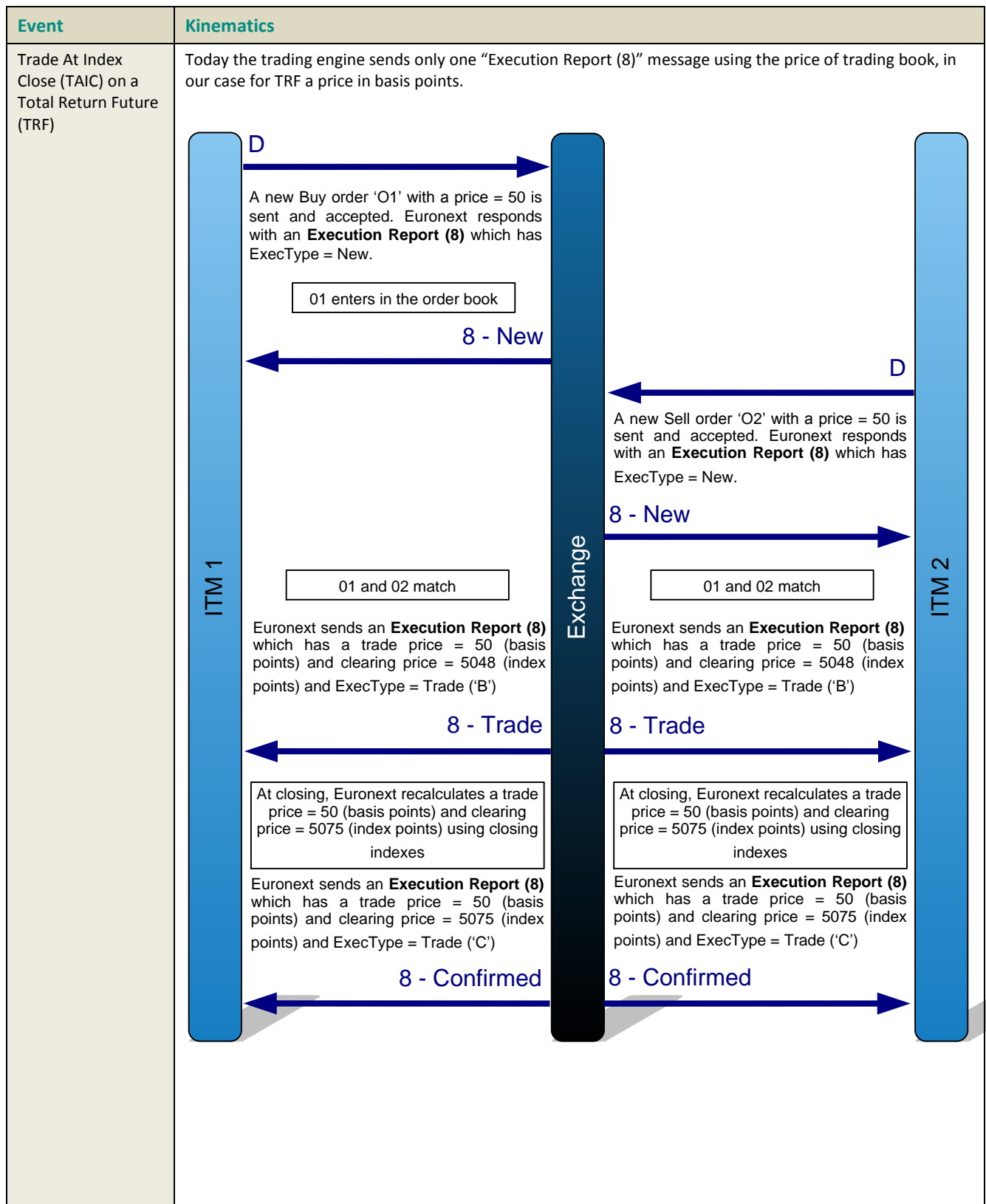
Client Message	Response Messages
A Trade occurs	<p>When an outright or strategy trade occurs or a Cross Order is authorised then the Exchange sends an <b>Execution Report (8)</b> for each trade with the OrdStatus set to Partially Filled or Filled.</p> <p>The ExecID identifies the Execution Report.</p> <p>The TradeID contains the trade identifier.</p> <p>The ExecType identifies that the execution report is for a Trade.</p> <p>The ClOrdID or OrderID fields identify the order.</p> 
A Trade occurs with a quote where Market Maker Protection is in place	<p>When an outright or strategy trade occurs with a quote where Market Maker Protection is in place then the <b>Execution Report (8)</b> would be followed by an <b>Adjust MM Position Ack (U7)</b> message. Whether the market maker protection is successfully or not successfully adjusted, is reported in the ProductProtectionStatus and ExpiryProtectionStatus.</p> 
A Trade is corrected	<p>When an outright or strategy trade is corrected then the Exchange sends an <b>Execution Report (8)</b> for each trade with the OrdStatus indicating the status of the order.</p> <p>The ExecID identifies the Execution Report.</p> <p>The TradeID contains the trade identifier.</p> <p>The ExecType indicates that the Execution Report is for a trade correction.</p> <p>The ClOrdID or OrderID fields identify the originating order.</p> 

Client Message	Response Messages
A Trade is cancelled	<p>When Market Control cancels an outright or strategy trade or rejects a Cross Order then the Exchange sends an <b>Execution Report (8)</b> for each trade with the OrdStatus indicating the status of the order. The OrdStatus = Rejected for a rejected Cross Trade and OrdStatus = Canceled for a deleted matched trade. The ExecType indicates that the execution report is for a trade cancellation. The LastQty contains the cancelled quantity.</p>  <p>The diagram illustrates the flow of an Execution Report (8) message. A blue vertical bar on the left is labeled 'ITM' and a dark blue vertical bar on the right is labeled 'Exchange'. A blue arrow points from the Exchange to the ITM. Above the arrow is the number '8'. To the right of the arrow, the following fields are listed: ExecID: ExecID1, ClOrdID: ClOrdID2, OrderID: OrderID3, OrdStatus: Cancelled/Rejected, ExecType: Trade Cancel, TradeID: TradeID4, and LastQty: 100.</p>

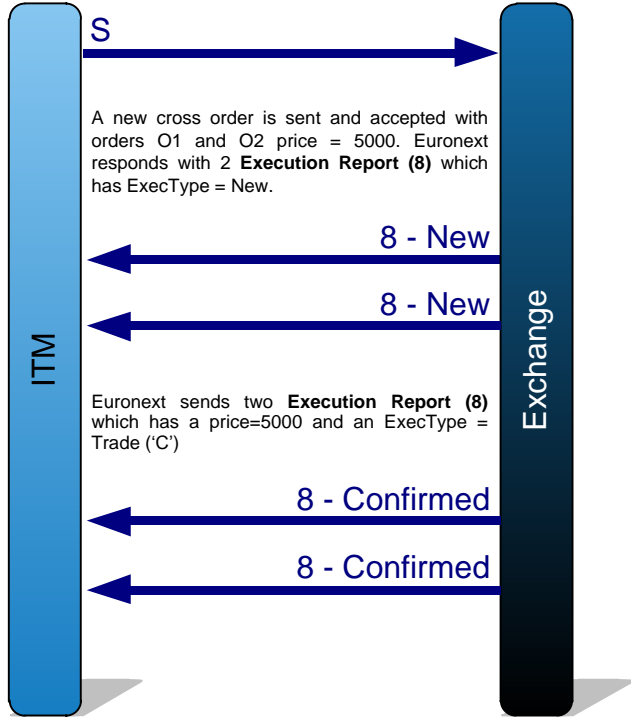
### D.3 IMMEDIATE OR CANCEL (IOC) ORDERS

Event	Kinematics
An Immediate or Cancel order trades	<p>Immediate or Cancel (IOC) orders are executed against any existing orders at the stated price or better, up to the volume of the IC order. Any residual volume from the IC order is cancelled.</p> <p>If an IOC order is partially filled then the following messages are returned:</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     Note over ITM: D     ITM-&gt;&gt;Exchange: CIOrdID: CIOrdID1 TimeInForce: IOC     Note over Exchange: 8     Exchange--&gt;&gt;ITM: CIOrdID: CIOrdID1 OrderID: OrderID2 OrdStatus: New ExecType: New Return Code: Liffe_Partial_Trade     Note over Exchange: 8     Exchange--&gt;&gt;ITM: CIOrdID: CIOrdID1 OrderID: OrderID2 OrdStatus: Partially Filled ExecType: Trade Return Code: Liffe_Trade_Status_Active     Note over Exchange: 8     Exchange--&gt;&gt;ITM: CIOrdID: CIOrdID1 OrderID: OrderID2 OrdStatus: Cancelled ExecType: Cancelled Return Code: Liffe_Status_Success     </pre> <p>If an IOC order is completely filled then the following messages are returned:</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     Note over ITM: D     ITM-&gt;&gt;Exchange: CIOrdID: CIOrdID1 TimeInForce: IOC     Note over Exchange: 8     Exchange--&gt;&gt;ITM: CIOrdID: CIOrdID1 OrderID: OrderID2 OrdStatus: New ExecType: New Return Code: Liffe_Status_Success     Note over Exchange: 8     Exchange--&gt;&gt;ITM: CIOrdID: CIOrdID1 OrderID: OrderID2 OrdStatus: Filled ExecType: Trade Return Code: Liffe_Trade_Status_Active     </pre>

#### D.4 SPECIFIC KINEMATICS FOR A TRADE ON A TOTAL RETURN FUTURE





Event	Kinematics
Trade At Market (TAM) on a Total Return Future (TRF)	<p>TAM is only possible with wholesale commands. In this case ("New Order Cross (s)" message), members will use the field 'OrderOrigin' (that is mandatory in the message when the instrument is a TRF) to indicate whether the cross trade is a Trade At Market (TAM) (in this case, the value 'M' will be used) or a Trade At Index Close (TAIC) (in this case, the value 'C' will be used).</p> <p>TAM uses directly the clearing notation. The kinematic of "Execution Report (8)" for TAM is:</p>  <pre> sequenceDiagram     participant ITM     participant Exchange     Note over ITM: S     ITM-&gt;&gt;Exchange: S     Note over Exchange: A new cross order is sent and accepted with orders O1 and O2 price = 5000. Euronext responds with 2 Execution Report (8) which has ExecType = New.     Exchange--&gt;&gt;ITM: 8 - New     Exchange--&gt;&gt;ITM: 8 - New     Note over Exchange: Euronext sends two Execution Report (8) which has a price=5000 and an ExecType = Trade ('C')     ITM--&gt;&gt;Exchange: 8 - Confirmed     ITM--&gt;&gt;Exchange: 8 - Confirmed </pre> <p>No "Execution Report (8)" message is sent with the price in trading notation.</p>

## APPENDIX E: FIXML STANDING DATA

This section provides some descriptions of the FIXML standing data file.

### E.1 EXAMPLE OF A FUTURE CONTRACT

The example below is for the JUN10 CAC40 Futures contract:

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	FIXML Values
<Exchange						
					MktSegID	MktSegID="J"
					MarketSegment Desc	MarketSegmentDesc="Euronext Paris - Index Products">
→	</Contract>					
					SecGrp	<Contract SecGrp="JFFCE"
					LogSym	LogSym="FCE"
					Desc	Desc="CAC 40 Index"
					Exch	Exch="XMON"
					RndLot	RndLot="10"
					Ccy	Ccy="EUR"
					MinPxIncr	MinPxIncr=".1"
					PxQteMeth	PxQteMeth="STD"
					ExerStyle	
					SettlMeth	SettlMeth="P"
					FlexInd	FlexInd="N"
					CFI	CFI="XXXXXX"
					Src	Src="P"
					ID	ID="CIPX1"> (CFI Code for an Index Cash Underlying)
	→	NestedAttr				
					Typ / Val	<NestedAttr Typ="25" Val="10" />
					Typ / Val	<NestedAttr Typ="101" Val="5" />
					Typ / Val	<NestedAttr Typ="102" Val="1" />
					Typ / Val	NestedAttr Type = "103" not populated
					Typ / Val	NestedAttr Type = "104" not populated

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	FIXML Values
					Typ / Val	NestedAttr Type = "110" not populated
					Typ / Val	NestedAttr Type = "111" not populated
					Typ / Val	NestedAttr Type = "111" not populated
					Typ / Val	<NestedAttr Typ="120" Val="T" />
					Typ / Val	<NestedAttr Typ="121" Val="0" />
	→	OrderTypeRules				
						<OrdTypeRules OrdTyp="1" /> <OrdTypeRules OrdTyp="2" /> <OrdTypeRules OrdTyp="W" />
	→	SecSubTypes				
						<SecSubTypes SubTyp="B" /> <SecSubTypes SubTyp="E" /> <SecSubTypes SubTyp="W" />
	→	<Expiry				
					ExpireDt	ExpireDt="20170600"
					MMYFmt	MMYFmt="0"
					MMY	MMY="201706"
					FirstTrdDt	FirstTrdDt="20020321"
					LastTrdDt	LastTrdDt="20170618">
					UndMMYFmt	
					UndMMY	
		→	NestedAttr			
					Typ / Val	<NestedAttr Typ="26" Val="5" />
					Typ / Val	<NestedAttr Typ="109" Val="500" />
					Typ / Val	NestedAttr Type = "110" not populated
					Typ / Val	NestedAttr Type = "105" not populated
					Typ / Val	NestedAttr Type = "106" not populated
					Typ / Val	NestedAttr Type = "107" not populated
					Typ / Val	NestedAttr Type = "108" not populated

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	FIXML Values
		→	<Series			
					CFI	CFI="FXIPSX"
					StrkPx	Not populated for a Future
					RndLot	RndLot="10"
					Src	Src="8"
					ID	ID="JFFCE100600000F">
			→	InstrCd		
					AltIDSrc / AltID	<InstrCd AltIDSrc="X" AltID="00000000000000789968" />
					AltIDSrc / AltID	<InstrCd AltIDSrc="Y" AltID="EUFR07899685" />
		→	</Series>			
	→	</Expiry>				
→	</Contract>					
</Exchange>						

## E.2 EXAMPLE OF AN OPTION CONTRACT

The example below is for the JUN15 BNP Paribas Option contract (note that only the first Call and Put are presented):

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	FIXML Values
<Exchange						
					MktSegID	MktSegID="P"
					MarketSegment Desc	MarketSegmentDesc="Euronext Paris - Equity Products">
→	</Contract>					
					SecGrp	<Contract SecGrp="POBN1"
					LogSym	LogSym="BN1"
					Desc	Desc="BNP Paribas (100)"
					Exch	Exch="XMON"
					RndLot	RndLot="100"
					Ccy	Ccy="EUR"
					MinPxIncr	MinPxIncr=".01"
					PxQteMeth	PxQteMeth="STD"
					SettlMeth	SettlMeth="P"
					ExerStyle	ExerStyle="1"
					FlexInd	FlexInd="N"
					CFI	CFI="XXXXXX"
					Src	Src="P"
					ID	ID="CSBNP"> (Security Group of the underlying contract)
	→	NestedAttr				
					Typ / Val	<NestedAttr Typ="25" Val="100" />
					Typ / Val	<NestedAttr Typ="101" Val="1" />
					Typ / Val	<NestedAttr Typ="102" Val="1" />
					Typ / Val	<NestedAttr Typ="103" Val="100" />
					Typ / Val	<NestedAttr Typ="104" Val="2" />
					Typ / Val	
					Typ / Val	
					Typ / Val	
					Typ / Val	<NestedAttr Typ="120" Val="T" />
					Typ / Val	<NestedAttr Typ="121" Val="4" />
	→	OrderTypeRules				

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	FIXML Values
						<OrdTypeRules OrdTyp="1" /> <OrdTypeRules OrdTyp="6" />
	→	SecSubTypes				
						<SecSubTypes SubTyp="A" /> <SecSubTypes SubTyp="B" /> <SecSubTypes SubTyp="D" /> <SecSubTypes SubTyp="E" /> <SecSubTypes SubTyp="F" /> <SecSubTypes SubTyp="G" /> <SecSubTypes SubTyp="H" /> <SecSubTypes SubTyp="I" /> <SecSubTypes SubTyp="J" /> <SecSubTypes SubTyp="K" /> <SecSubTypes SubTyp="L" /> <SecSubTypes SubTyp="M" /> <SecSubTypes SubTyp="N" /> <SecSubTypes SubTyp="P" /> <SecSubTypes SubTyp="R" /> <SecSubTypes SubTyp="S" /> <SecSubTypes SubTyp="V" /> <SecSubTypes SubTyp="W" /> <SecSubTypes SubTyp="X" /> <SecSubTypes SubTyp="a" /> <SecSubTypes SubTyp="b" /> <SecSubTypes SubTyp="c" /> <SecSubTypes SubTyp="d" /> <SecSubTypes SubTyp="e" /> <SecSubTypes SubTyp="f" /> <SecSubTypes SubTyp="g" /> <SecSubTypes SubTyp="h" /> <SecSubTypes SubTyp="i" /> <SecSubTypes SubTyp="j" /> <SecSubTypes SubTyp="k" /> <SecSubTypes SubTyp="n" /> <SecSubTypes SubTyp="p" /> <SecSubTypes SubTyp="q" /> <SecSubTypes SubTyp="r" /> <SecSubTypes SubTyp="s" /> <SecSubTypes SubTyp="t" /> <SecSubTypes SubTyp="v" /> <SecSubTypes SubTyp="w" /> <SecSubTypes SubTyp="x" /> <SecSubTypes SubTyp="y" /> <SecSubTypes SubTyp="z" />
	→	<Expiry				
					ExpireDt	<Expiry ExpireDt="20150600"
					MMYFmt	MMYFmt="0"
					MMY	MMY="201506"
					FirstTrdDt	FirstTrdDt="20120618"
					LastTrdDt	LastTrdDt="20150619"
					UndMMYFmt	UndMMYFmt="0"

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	FIXML Values
					UndMMY	UndMMY="201506">
		→	NestedAttr			
					Typ / Val	<NestedAttr Typ="26" Val="1" />
					Typ / Val	
					Typ / Val	
					Typ / Val	
					Typ / Val	
					Typ / Val	
					Typ / Val	
		→	<Series			
					CFI	<Series CFI="OCAXPS"
					StrkPx	StrkPx="2000"
					RndLot	RndLot="100"
					Src	Src="8"
					ID	ID="POBN1150602000C">
			→	InstrCd		
					AltIDSrc / AltID	<InstrCd AltIDSrc="X" AltID="000000000000000167108" />
					AltIDSrc / AltID	
		→	<Series			
					CFI	CFI="OPAXPS"
					StrkPx	StrkPx="2000"
					RndLot	RndLot="100"
					Src	Src="8"
					ID	ID="POBN1150602000P">
			→	InstrCd		
					AltIDSrc / AltID	<InstrCd AltIDSrc="X" AltID="000000000000000167109" />
					AltIDSrc / AltID	
		→	</Series>			
	→	</Expiry>				
→	</Contract>					
</Exchange>						

### E.3 EXAMPLE OF A STRATEGY

The example below is for a Calendar Spread MAY15 / JUL15 on the BNP Paribas Future contract. For the ease of the presentation, the Contract information is not copied:

Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	FIXML Values
<Exchange						
					MktSegID	MktSegID="P"
					MarketSegment Desc	MarketSegmentDesc="Euronext Paris - Equity Products">
→	</Contract> .....					
	→	<Expiry> .....				
		→	<Series> .....			
	→	</Expiry>				
	→	<Strategy				
					SubType	SubType="E"
					CFI	CFI="FFXCSX"
					Src	Src="8"
					ID	ID="PFBN602E0087CDS">
		→	<Leg			
					SecGrp	SecGrp="PFBN6"
					Src	Src="8"
					ID	ID="PFBN6150500000F"
					CFI	CFI="FFXCNX"
					MMYFmt	MMYFmt="0"
					MMY	MMY="201505"
					Strike	
					Side	Side="1"
					RatioQty	RatioQty="1" />
					Px	
		→	<Leg			
					SecGrp	SecGrp="PFBN6"
					Src	Src="8"
					ID	ID="PFBN6150700000F"
					CFI	CFI="FFXCNX"
					MMYFmt	MMYFmt="0"
					MMY	MMY="201507"



Block	Sub Block 1	Sub Block 2	Sub Block 3	Sub Block 4	Field	FIXML Values
					Strike	
					Side	Side="2"
					RatioQty	RatioQty="-1" />
					Px	
	→	</Strategy>				
→	</Contract>					
</Exchange>						

## APPENDIX F: REVIEW LOG, DOCUMENT HISTORY, SIGN-OFF

### REVIEW LOG

DOCUMENT NAME	UNIVERSAL TRADING PLATFORM FOR DERIVATIVES
REVISION VERSION	Revision Number: 1.2

### DOCUMENT HISTORY

REVISION NO./ VERSION NO.	DATE	CHANGE DESCRIPTION
0.0	1 Apr 2010	Initial version
0.1	19 Jul 2010	New version including: - Chapter 6: Order Traffic Management - Chapter 7: General Overview of Order Entry Protocol - Chapter 8: Market Access - Chapter 10: Market Information - Chapter 11: Trading and Order Handling - Chapter 12: Market Control
0.2	1 Mar 2011	4.3.1 : note about access to NYSE Liffe U.S. from Europe 9.4.1: Change on table Chapter 13: New paragraph about fractional pricing, CTI Code and addition of examples (fractional pricing, Reduced Tick Size Spread) Appendix 5: NYSE Liffe U.S. Contract Specifications
0.3	3 Aug 2011	9.1.4: Change in table following removal of Exchange Code 11.2.3: Change in table
0.4	31 Oct 2011	14.5: Change in table (NYSE Liffe U.S. Instruments) 10.2: Addition of a new paragraph (Pack and Bundle strategies)
0.5	2 Nov 2012	6.4: Change in the rejection code in case of System Busy
0.6	17 Jun 2015	Updated version since latest projects.
0.7	22 Jul 2015	Updated version since review.
0.8	04 Aug 2015	Updated version since review by EMS.
0.9	01 Oct 2015	<ul style="list-style-type: none"> <li>Updated version since changes in the Wholesale facility,</li> <li>"ISIN" changed to "SecurityGroup" in part <a href="#">7.2.1</a>.</li> <li>Addition of kinematics between members and CCG in a dedicated appendix.</li> </ul>
1.0	16 Feb 2016	<ul style="list-style-type: none"> <li>Clarification and correction about implied pricing, section <a href="#">8.1</a>,</li> <li>Amendment in the name of a Wholesale facility: "LiS package" (instead of "LiS Package trade"),</li> <li>"Account authorization" functionality and LiS Package facility are not Amsterdam specific anymore,</li> <li>Changes in the Market Makers protections,</li> <li>Updates since feedback from CTSG.</li> </ul>
1.1	4 Apr 2017	<ul style="list-style-type: none"> <li>Added specific kinematics for Total Return Futures</li> </ul>

REVISION NO./ VERSION NO.	DATE	CHANGE DESCRIPTION
1.2	21 Feb 2018	<ul style="list-style-type: none"><li>• Update Total Return Future TAIC kinematics</li></ul>

---

## CONTACT INFORMATION

For any questions about this document please contact

- Customer Technical Support Group (CTSG)
  - Email: [CTSG@euronext.com](mailto:CTSG@euronext.com)
  - Tel.: +33 (0)1 8514 8588