IT Development Division
Trading Systems Development Department

OASIS

Athex Gateway Architecture & Application Development Guidelines

Version 3.0.2

Athens, September 2017

Revision List

Version	Date	Description
1.0.0.0	09/04/2010	- The ATHEX Gateway is a suite of services that includes the ODL Gateway service with the additional support for FIX protocol interface
1.0.0.1	20/12/2010	- Changes in the format of price-type fields in the messages from/to the XOrder Server.
1.0.0.2	21/02/2011	- Changes in the valid values for the OrdType and TimeInforce tags for the FIX interface. - Addition to the FIX interface of a notification when a communication problem is detected between ATHEX Server and Comms Server.
1.1.0.0	30/05/2011	- Change in the format of the price fields - Translation of the TradeStatus (TG) message to a FIX Execution Report message - Translation of the CreditLimitInfo (TL) message to a FIX Security Status message - Translation of the MarketStatus (CC) message to a FIX Trading Session Status message
1.2.0.0	17/04/2013	- New DSS (Dematerialized Securities System) interface - New messages (MS, TS, TU and CS) specific for the new DSS interface - Replaced all references to XOrder with XNet - Grouped Comms, XNet and DSS servers together as ATHEX Servers.
2.0.0.0	24/04/2013	- Common ODL API and FIX Protocol interface is now used for all instruments of Greek and Cyprus Stock, Fixed Income and Derivatives markets. - Upgrade of FIX protocol interface from version 4.2 to version 4.4 - Addition of quotation negotiation handling to both ODL API and FIX protocol - Addition of Trade Reporting (Pre-agreed price trading) handling to FIX protocol interface. The pattern for submitting trade reports through ODL API and FIX protocol is now more simplified. See paragraphs 4.2.3.4, 0 and 6.2.4. - In Trade Reporting TF messages, order-related fields are now unused. - Addition of combined order type. This order type concerns the Time Spread Strategies in the Derivatives market - Addition of FIX protocol interface to XNET services. - A number of existing ODL API Requests Objects have been updated (updates have been marked with red color in paragraph 4.2.3) - A number of existing FIX messages have been updated (updates have been marked with red color in chapter 5) - Trade Status (TG) message has been removed. This information is now sent through Trade Info (TF) message. - Addition of ExchangeNotes object

		▪ Added the relevant message type to the objects described in 4.2.3 ▪ Section 7.2 was renamed to Best practices for the development of ODL client
2.0.0.2	16/10/2013	▪ Changes in SecurityInfo message - changed size and values in UnderlyingProduct field - changed values in Product field - changed values in SecurityType field ▪ Added FIX support for DSS & Clearing transactions. ▪ Revised ODL & FIX messages and fields (including Appendix B) ▪ Added tag 198 (SecondaryOrderID) to all outgoing messages (FIX) ▪ Added field PositionEffect to message MI (ODL) ▪ Added tag 77 (Position Effect) to TCR/TCRA (FIX) ▪ Added tag 880 (TradeMatchId) to ExecutionReport (FIX) ▪ User-defined FIX tag 5534 (EditType) has been removed ▪ FIX tag 210 (MaxShow) has been removed from the user-defined tags table of the Order Cancel Request message ▪ FIX tag 207 (SecurityExchange) is used instead of FIX tag 100 (ExDestination) in New Order Single, Order Cancel/Replace Request and Execution Report messages ▪ Changed TJ message to QuoteRequestConfirmation ▪ Added QuoteRequestExecution (TP) message ▪ Changed Quote Request sequence diagram ▪ Added NoteType field to message TO (ODL). ▪ Renamed SettlementType to SettlType in all ODL application request objects. ▪ Renamed ChangedSettlementType to ChangedSettlType in all ODL application request objects.
2.0.0.3	22/01/2014	▪ PositionEffect moved to within “Side” component in TCR/TCRA FIX messages ▪ Added ODL “CG” message and FIX equivalent ▪ Added SecuritySubType field in “CF” message and FIX equivalent ▪ Split Trade Confirmation is supported in FIX protocol through the ExecutionReport message ▪ Added “Trade Correct” value in ExecType(150) FIX tag in ExecutionReport message ▪ Changed price field lower limit to -1000. ▪ Added FIX user defined tag CancelReasonCode(5508) in the ExecutionReport message ▪ Added FIX user defined tag CurrentCreditLimit(5545) in QuoteStatusReport message ▪ Valid values for SettlType field in ODL messages have changed ▪ Updated TCRA FIX message.
2.0.0.4	17/02/2014	▪ Removed VenueID and Reserved field from message TL (ODL). ▪ The FIX message News (35=”B”) is used to send Credit Limit Info (TL).

		▪ Added field OrigClientOrdID to TF message.
2.0.0.5	28/02/2014	▪ QuoteRequestStatus(5576) FIX tag has been substituted by ODLMsgType(5574) tag in QuoteRequest and QuoteRequestReject messages. ▪ Default value for NoPartyIDs(453) FIX tag is 2 in QuoteRequest and QuoteRequestReject messages. ▪ Account(1) FIX tag is not required in OrderCancelReplaceRequest message.
2.0.0.6	04/03/2014	▪ Price(44) FIX tag is not required in NewOrderSingle message. ▪ Added two combinations-related error codes to field RejectReasonCode. ▪ The signature of Administrator.NewDataMessage has changed. ▪ The signature of Broker.NewETSMessage has changed. ▪ The Serial interface has been removed. From this version, multiple Admin connections are possible. ▪ Group Of Investors feature is removed including messages MX, TX and TH. ▪ Updated valid values list for Product and UnderlyingProduct fields.
2.0.0.7	12/05/2014	▪ ExecRestatementReason(378) FIX tag has been added in the ExecutionReport message ▪ Add "OOF" (Option on Future) value in SecurityType field (for FIX interface and ODL interface) ▪ Substitute ' ' value with 'D' for FIX tag PhaseID(5511)
2.0.0.8	12/07/2014	▪ Added note on field OrdRefId ▪ The Associated Product of the Security Type = WAR is now 12 (OTHER) ▪ The user defined field SecurityISIN (5578) added to the FIX message f with ODLMsgType (5574) = CF ▪ As long as a life order (GTD or GTC) is valid, an ExecutionReport message is sent at the beginning of each trading session with ExecType(150) equal to D (restated) instead of 0 (New) ▪ LastPx(31) and LastQty(32) FIX tags in ExecutionReport message have been marked as not required.
2.0.0.9	06/08/2014	▪ Trade report types "RC" & "RG" added. ▪ ExecID(17) FIX tag's description has been updated
2.0.0.10	30/10/2014	▪ Added comment on excluded zero price on combinations. ▪ Added section on restated orders (ExecType/OrdStatus) ▪ Added section on TF handling (OrderRefID) ▪ Added new Halt Reason (13) for Series Expiration Message
2.0.1.0	April 2015	▪ Updated FIX session recovery section (12.3) ▪ Added comment on excluded zero price on combinations. ▪ Added TradeType value "MM" ▪ Added OrderStatus value "A "
3.0.0.0	May 2017	▪ Added new DateTime field data type. ▪ Added section "Appendix J. Production of MIFID II field 21 of RTS24" ▪ Corrected description of parameter ldStartMsgID of Broker.GetLostMessages(). ▪ Added HaltReasonCode "13". ▪ Added values for the RejectReasonCode field

		▪ Added value for the CancelReasonCode field ▪ Added details on conditionally unavailable fields of FIX Execution Report (OrderType, Price, ClearingFirm) ▪ Changed default value for PositionEffect to 'Opening' (FIX) ▪ Added value 'N' to Status field of CC message, Added the same value in FIX TradingSessionID(336) field. This field is used only by XNET interface. ▪ Added possible value 0 'Normal' on FIX tag SettleType. ▪ The value 'U' – Underlying is added in the FIX user defined field QuotationResponsibilityLevel (5563) ▪ Added max length for FIX tag ClOrdId and OrigClOrdId. ▪ ▪ Updated section "3.1.4 FIX Applications" ▪ Updated section "3.7.2 Protection Valves" (Oustanding Messages) ▪ Updated section "5.8 Unsolicited order changes" (Execution Reports) ▪ Updated section "6.2.2.1 Order Edit - Initial Verifications" ▪ Updated section "6.2.3.1 Order Change - Initial Verifications" ▪ Updated section "6.2.3.1 Order Change - ChangedPrice" ▪ Added section "7.4 FIX Client Application Development" ▪ Updated section "7.5 Guidelines summary" ▪ Updated section "12.3.1 Resetting FIX sequence number counters on the client side" ▪ Updated section "12.3.2.1 Automated FIX session recovery" ▪ Updated section "12.3.2.2 Manual FIX session recovery" ▪ The following ODL classes/messages have been removed: - SecurityInfo (CF) - BondsInformation (CE) - CombinationInfo (CG) ▪ The following FIX messages have been removed: - SecurityInfo (ODL CF Message) - SecurityInfo (ODL CE Message) - CombinationInfo (ODL CG Message)
3.0.1	August 2017	▪ Added PartyRoleQualifier field (tag 2376) to the Parties repeating group (FIX) ▪ Removed NoOrderAttributes, OrderAttributeType and OrderAttributeValue from TradeCaptureReport (FIX) ▪ Deleted incorrect restriction of one Admin connection per GW. Since version 2.0, more than one Admin connections are possible. ▪ Added field "SpecialInstruction" in messages MB, TB, MD, TD for use with XNET and the Electronic Book Building ▪ Added special tag 5999 ("SpecialInstruction") in FIX messages NewOrder – Single, Order Cancel/Replace Request and Execution Report. For use with XNET and the Electronic Book Building. ▪ All new fields have been moved to the end of ODL API messages ▪ Added OrderOrigination field in Quote (FIX) ▪ Added OrigTime field in News (ODL TM message) ▪ Added GrossTradeAmt field in ExecutionReport (FIX) ▪ Removed OrderOrigination and OrderCapacity fileds from OrderCancelReplaceRequest (FIX)
3.0.2	September 2017	▪ Formating of document ▪ Added values (147 -158) for the RejectReasonCode field ▪ Waiver Indicator field has been corrected

		▪ ClientIDQualifier, InvestmentDecisionIDQualifier & ExecutionWithinFirmIDQualifier may take the vaule od space " " in the ODL API ▪ Filed SpecialInstructions renamed to ChangedSpecialInstructionsIn in TD Message

Table of Contents

List of Figures

Copyright

Important Note

The Athens Exchange agrees that the information contained herein is accurate as of the date of publication. This document may be altered without any prior notice. The Athens Exchange shall not be held liable of any direct or indirect consequences arising from any errors herein.

This document is the updated version of older documents containing corresponding information.

1. Introduction

This is the basic documentation of the ATHEX Gateway version 3.0; this is the Athens Exchange (ATHEX) in-house implementation of the API of OASIS Trading System (OASIS), ATHEX Order Routing (XNet) and DSS (Dematerialized Securities System). It is actually a suite of services that includes the following:

- ODL service version 5.0 (Athex proprietary API)
- ASE service version 5.0 (FIX protocol[1])

The ATHEX Gateway provides the ATHEX Members with the capability to develop their own applications with double connectivity to the central exchange trading application exploiting the specific functionality of OASIS or to the ATHEX Order Routing infrastructure [2] (XNET) for routing order information to multiple venues/exchanges as well as connectivity to DSS for Clearing. An extensive description of the Xorder service is out of the scope of this document. Moreover, this document mainly respects and references the ATHEX Gateway route to the ATHEX OASIS platform; the reader may however presume the validity of extensions of development guidelines to the new Xnet interface [3] as well as the new DSS interface.

Another functionality introduced in ATHEX Gateway (since vesion 1.0) is the capability of the ATHEX Gateway to communicate with the ATHEX Members applications using two types of interfaces: the ODL API interface and the FIX protocol interface.

With the ATHEX Gateway, the Athens Exchange allows Members to add new possibilities and characteristics to their applications.

The basic equipment configuration (hardware and applications) for the usage of the ATHEX Gateway at the ATHEX Member's office and the central systems at the Stock Exchange is shown in following figure:

[1] The Financial Information eXchange (FIX) Protocol is a messaging standard developed specifically for the real-time electronic exchange of securities transactions. FIX is a public-domain specification owned and maintained by FIX Protocol, Ltd. For more information, please visit http://www.fixprotocol.org.

[2] The ATHEX order routing infrastructure is implemented by ATHEX XOrder Server system.

[3] Although client applications may communicate transparently to all available interfaces (ETS/XNet/DSS), differences may apply in supported functionality through different exchanges supported by XNet

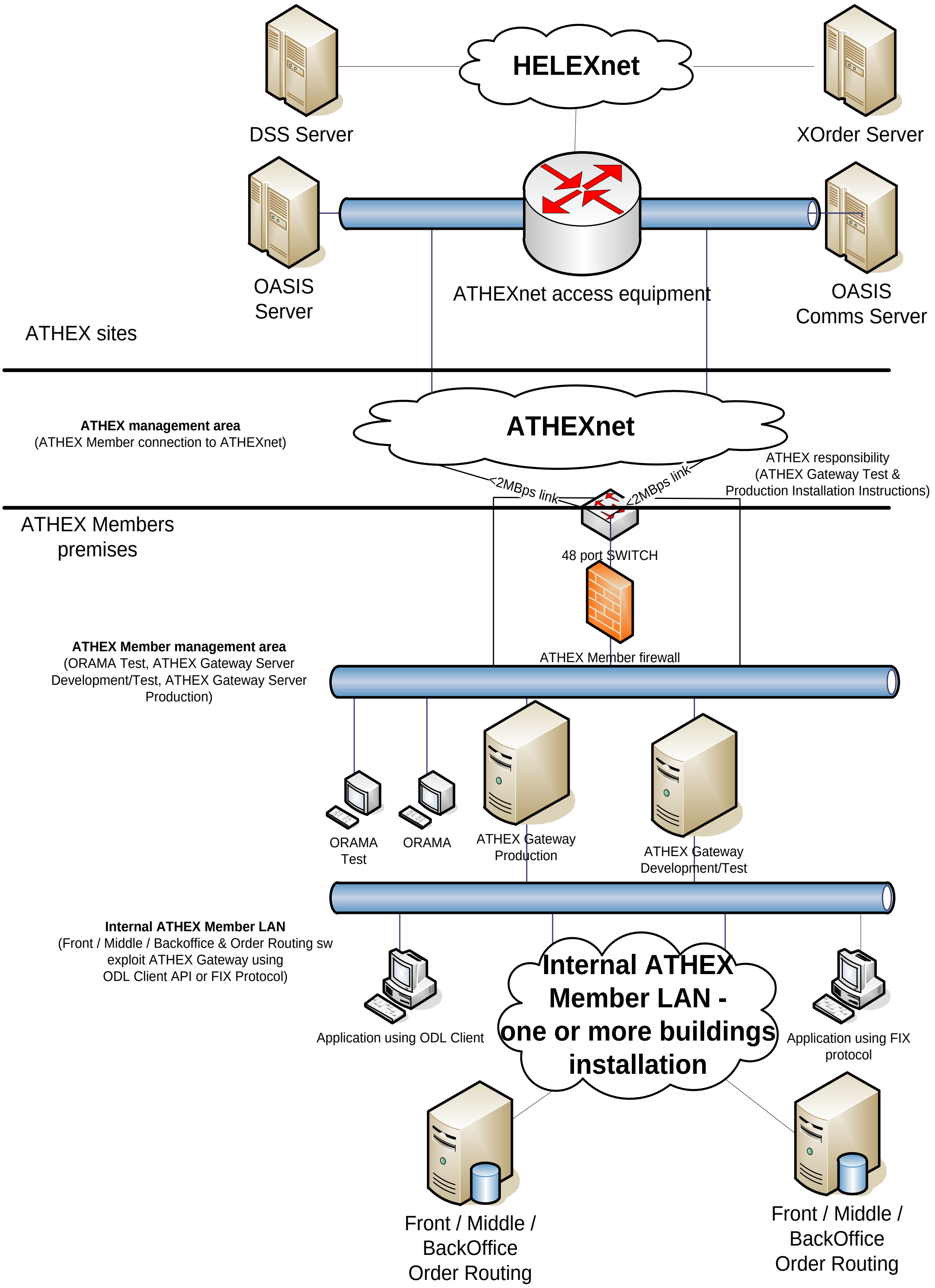

HELEXnet
DSS Server
XOrder Server
OASIS Server
ATHEXnet access equipment
OASIS Comms Server
ATHEX sites
ATHEX management area
(ATHEX Member connection to ATHEXnet)
ATHEXnet
<2MBps link
<2MBps link
ATHEX responsibility
(ATHEX Gateway Test & Production Installation Instructions)
ATHEX Members premises
48 port SWITCH
ATHEX Member firewall
ATHEX Member management area
(ORAMA Test, ATHEX Gateway Server Development/Test, ATHEX Gateway Server Production)
ORAMA Test
ORAMA
ATHEX Gateway Production
ATHEX Gateway Development/Test
Internal ATHEX Member LAN
(Front / Middle / Backoffice & Order Routing sw exploit ATHEX Gateway using ODL Client API or FIX Protocol)
Application using ODL Client
Internal ATHEX Member LAN - one or more buildings installation
Application using FIX protocol
Front / Middle / BackOffice Order Routing
Front / Middle / BackOffice Order Routing

Figure 1-1 Equipment configuration required (hardware and applications) in the ATHEX Member sites and the ATHEX site for the ATHEX Gateway

The following different equipment areas as far as the installation and usage of the ATHEX Gateway from a Member's site is concerned, are shown:

- ATHEX Servers

- Equipment at the premises of an ATHEX Member

The network connection and the communication of the equipment in these areas are made using exclusively the Athens Exchange Trading Network, thereafter "ATHEXnet" or "Network".

More specifically:

1. **ATHEX Servers centrally located at the ATHEX sites are:**

    1.1. Server systems running the *Trading Subsystem of OASIS*.

    1.2. Communications Server systems (Comm Server) running the *CTCI (Computer To Computer Interface*) application, which implements the connection of the Members' applications running inside their local network environment to the trading application on the OASIS Server (1.1 above).

    1.3. Xnet Server for routing orders to multiple venues/exchanges.

    1.4. DSS Server for clearing and settlement of trades

2. **Located at the ATHEX Member's sites are:**

    2.1. The network access equipment, connecting the Member to the ATHEXnet. It is inside the **ATHEX** management area and consists of:

    2.1.1. One router.

    2.1.2. One 48 port Switch.

    These are used for the installation of the development/test and production environments of the ATHEX Gateway in the Member workplace.

    2.2. The equipment of the ATHEX Gateway development/test and production environments. It is inside the **ATHEX Member's** management area and consists of:

    2.2.1. The ATHEX Gateway **development/test** environment:

    2.2.1.1. Server – PRIMARY GATEWAY 1, running the ATHEX Gateway –Server software (namely the ODL Server software and on demand the ASE _Service Server Software), configured to connect either to the development/test environment of the OASIS or to the shadow environment of the OASIS used for mock trading. This system is physically connected **both** to the ATHEXnet and the internal local network (LAN) of the ATHEX Member.

    Should the Member select to use a primary – backup ATHEX Gateway configuration, a second identical server – BACKUP GATEWAY 1 must be configured to connect to the same OASIS environment as the primary one. This second server increases the fault tolerance capability and decreases the time needed for recovery in case of operation failure of the GATEWAY 1.

A further alternative is a multiple gateway configuration (i.e. PRIMARY GATEWAY ½), each running on identical servers, set to primary and active concurrently. This solution also covers fault tolerance requirements and additionally increases order throughput while eliminating time and error margin in failure scenarios.

2.2.1.2. One Workstation PC, running the OASIS/ETW-ORAMA application configured to connect exclusively to the development environment of OASIS. This system is physically connected only to the ATHEXnet.

2.2.2. The ATHEX Gateway **production** environment:

2.2.2.1. Server – PRIMARY GATEWAY 2, running the ATHEX Gateway –Server software (namely the ODL Server software and on demand the ASE _Service Server Software), configured to connect exclusively to the production environment of the OASIS. This system is physically connected **both** to the ATHEXnet and the internal local network (LAN) of the ATHEX Member.

Should the Member select to use a primary – backup ATHEX Gateway configuration, a second identical server BACKUP GATEWAY 2, must be configured. This second server increases the fault tolerance capability and decreases the time needed for recovery in case of operation failure of the GATEWAY 2. Both primary-backup and multiple gateway schemes as described in 2.2.1.1 above also apply.

2.2.3. The ODL-Client-API application, installed with Member's responsibility on computer systems (called ODL Clients henceforth) in the internal ATHEX Member local network, without direct ATHEXnet connection but capable to communicate either with the ATHEX Servers through either the ATHEX Gateway development/test or production environments (using GATEWAY 1 or GATEWAY 2 respectively).

Note: The ATHEX Member makes the procurement of the above equipment (hardware and OS software) following ATHEX recommendations.

2.3. The equipment in the ATHEX Member workplace consists of:

2.3.1. The internal ATHEX Member networking infrastructure. It may expand to more than one building, including the one hosting the ATHEXnet access equipment (2.1 above).

2.3.2. The specially developed ATHEX Member application(s) running in the Member's ODL Clients that use the ODL-Client-API of ATHEX Gateway, connecting other applications in Member's LAN (Front/Middle/Back Office type, Order Routing type etc) with ATHEX Servers.

2.3.3. The specially developed ATHEX Member application(s) running in the Member's Clients that use the FIX protocol interface, connecting with ATHEX Servers.

2.3.4. ATHEX Member's Front/Middle/Back Office, Order Routing applications inside the ATHEX Member local network exploit the ATHEX

Gateway through the proper communication with the applications referred to the 2.3.2 above.

1.1. General Characteristics of ATHEX Gateway

The general characteristics of the ATHEX Gateway are:

- It assumes the existence of a particular equipment configuration at the Member's offices. Specifically, a Microsoft Windows Server system must be installed in order to become the ATHEX Gateway. This system will be simultaneously connected to the ATHEXnet and to the Member's internal network.

  The applications using the ATHEX Gateway are installed in one or more systems (clients), connected to the Member's internal network and communicating with the ATHEX Gateway(s) (and therefore with OASIS), using the ODL API. Alternatively, the ATHEX Gateway communicates with the Clients using the FIX protocol interface.

  Two types of application message interfaces between ATHEX Gateway and Client. The clients communicate with the ATHEX Gateway either by using ODL API and the ODL application message interface (ODL Clients) or without the use of the ODL API by using the FIX protocol interface (FIX Clients).

- Bi-directional communication. The applications deliver orders, alter or cancel orders and receive confirmation messages and messages for the system's status. It should be noted that the member shall use only these confirmation messages in order to be sure that these order entries, alterations and/or cancellations have been accepted by ATHEX Servers. It should also be noted that no security price information is returned.

- Communications security. The security of the ODLservice is based on the security provided by the Microsoft Windows operating system and requires the harmonization of the Member's security policy with the Athens Exchange security rules. The design and implementation of the ATHEX Gateway guarantees the avoidance of security problems. The falsification or interception of messages outside the Member's scope is not possible since the basic security rules in the Member's internal network are obeyed.

- Recovery capability after the occurrence of an error. Even in the cases where a technical problem impedes the transfer of messages between the OASIS and the ATHEX Gateway or between the ATHEX Gateway and the ODL or FIX Client, these messages are saved and can be recovered when the technical problem is solved.

- The ATHEX Gateway software (since version 1.3) introduces the concept of the backup ATHEX Gateway. This is a server similar to the primary, but possibly physically located in another building. It can be always synchronized with the primary ATHEX Gateway. This is shown in the following figure:

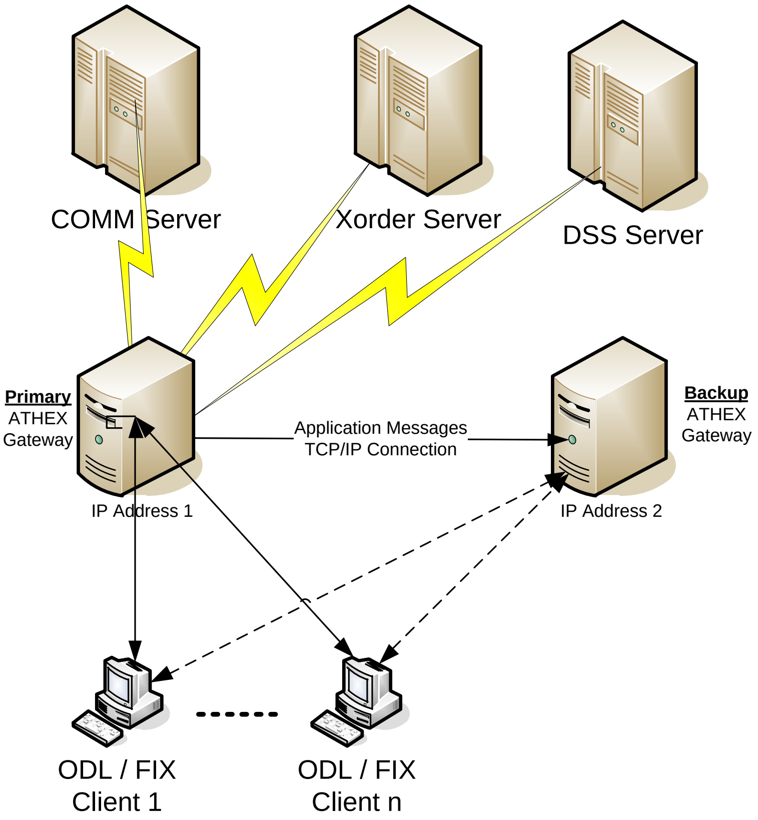


Figure 1-2: Primary – Backup Gateway configuration

- Fault tolerance capability. If the ATHEX Member uses a second server operating as backup ATHEX Gateway, then this server will always be aware of the status of the execution in the primary ATHEX Gateway So, in case of an operation failure of the primary ATHEX Gateway, this second server can continue the operation of the service without any significant time delay; just by switching its operating mode to primary. This is achieved by using the architecture described in Figure 1-2.

- Use of multiple concurrently active gateways (available as of version ODL service 3.0) transposes the primary-backup gateway principal and offers fault tolerance while elaborating on seamless failover, increased order throughput and load balancing. Again, this is achieved by ensuring that all available gateways are in-synchro with respect to incoming messages. Effectively, each outgoing message spawns one identical response per available gateway. Hence, at any point in time all gateways are readily functional; a client can transmit orders through any given gateway at his own initiative, while expecting that all gateways will be kept informed of the outcome. An example setup based on this model is illustrated in the figure below:

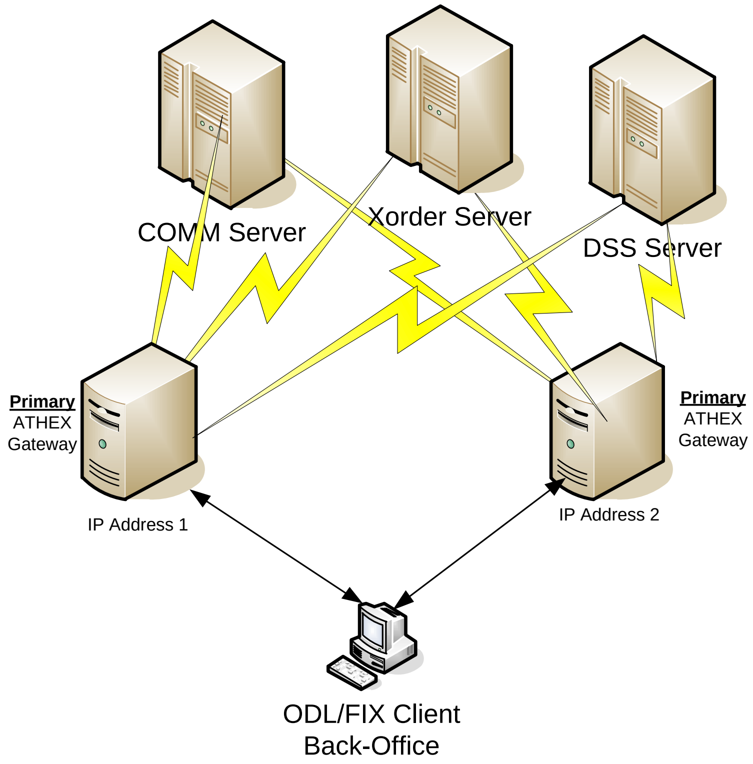


Figure 1-3 Double concurrent Gateway configuration

- Alternative route for receiving messages. All messages in relation to the Member's activity can be exported from the ATHEX Gateway's Admin interface.
- Finally, in case of ODL Clients, the use of DCOM guarantees the efficient operation of the ATHEX Gateway.

1.1.1. *Compatibility with ATHEX Gateway (1.2)*

The version 2.0 of the ATHEX Gateway is **not** backwards compatible to the previous version 1.2 of ATHEX Gateway. ATHEX Members, using the previous version 1.2 of the ATHEX Gateway, in order to connect to OASIS, **are** obliged to switch to the ATHEX Gateway version 2.0.

The ODL Client applications developed for use with ATHEX Gateway 1.2 can **not** use the ATHEX Gateway version 2.0. This means that they must be updated. Moreover, should an ATHEX Member wish to use the FIX protocol interface (FIX 4.4) implemented in this version (2.0), ATHEX recommends that it shall develop or use a FIX Client application capable to communicate with ATHEX Gateway using FIX protocol version 4.4.

1.1.2. *New features of the ATHEX Gateway version 2.0*

Comparing to the ATHEX Gateway 1.2, ATHEX Gateway version 2.0 supports the following:

- Common ODL API and FIX Protocol interface for all instruments of Greek and Cyprus Stock, Fixed Income and Derivatibes markets.

- Quotation negotiation handling.FIX protocol interface support for Xorder Server (XNET)
- Upgrade of FIX protocol interface from version 4.2 to version 4.4
- Addition of quotation negotiation handling to both ODL API and FIX protocol
- Addition of Trade Reporting (Pre-agreed price trading) handling to FIX protocol interface. Besides, the pattern for submitting trade reports through ODL API and FIX protocol is now simplified. See paragraphs 4.2.3.4, 0 and 6.2.4 and Appendix I. Sequence Diagrams for Trade Report procedure Examples.
- Addition of combined order type This type concerns the Time Spread Strategies in the Derivatives market
- A number of existing ODL API Requests Objects have been updated (updates have been marked with red color in paragraph 4.2.3)
- A number of existing FIX messages have been updated (updates have been marked with red color in chapter 5)
- Trade Status (TG) message has been removed. This information is now sent through Trade Info (TF) message.

1.2. Next Chapters Structure

In the following chapters the design characteristics of the ATHEX Gateway are described followed by application development guidelines and the essential installation and usage instructions. Specifically:

Chapter 2: The design characteristics of the ATHEX Gateway are described (ATHEX Gateway –Server and ODL-Client-API applications).
Chapter 3: General description of application development using ATHEX Gateway.
Chapter 4: The usage of the ATHEX Gateway through the ODL-Client-API application is described.
Chapter 5: The usage of the ATHEX Gateway through the FIX protocol interface is described.
Chapter 6: This chapter describes the Business Logic that the Member's applications should implement, in order to avoid the sending of incorrect messages to the OASIS Server which are both rejected and cause undue burden to the infrastructure.
Chapter 7: Contains guidelines and clarifications which will facilitate the development of more efficient applications for the use of the ATHEX Gateway.

Appendix A, B: Information to understand concepts described in Chapters 3 and 4.
Appendix C, D: Description of the application error codes.
Appendix E: Visual Basic programming example using ETS_Broker Object.
Appendix F: ATHEX Gateway Failover Mechanism when a primary – backup scheme is used
Appendix G: OASIS system's Parameters and some General Remarks and References to relevant documents.
Appendix H: Sequence diagrams for Quotation Negotiation.
Appendix I: Sequence diagrams for Trade Report procedure Examples.
Appendix J: The glossary of terms related to the OASIS and the ATHEX Gateway is provided.

The Members and software developers may address:

Technological Systems & Services Directorate

110, Athinon Ave., 10442 Athens GREECE,

Tel. +30 210 33 66 800,

Fax. +30 210 33 66 303

0) For issues in relation to the information contained in this document and generally issues concerning the Stock Market operation:

Attention: Members-Support@helex.gr

b) For issues in relation to the information contained in this document and generally issues concerning the development of applications by Members through the ATHEX Gateway:

Attention: TSDD-API@helex.gr

2. ATHEX Gateway Design Characteristics

The ATHEX application for OASIS trading and clearing system, as far as the architecture of the ATHEX Gateway is concerned, comprises the following modules:

1. *OASIS Server* is the central ATHEX Trading Server. It carries out all the daily transactions of the equities & derivatives trading in ATHEX. The *Communications Server* (running the CTCI application) handles the communication between OASIS and ATHEX Gateways,,

2. *Xnet Server* for routing orders to multiple venues/exchanges enabling Direct Market Access (DMA),

3. *DSS Server* for clearing and settlement of trades

4. *ATHEX Gateway* functioning as an order concentrator from the ATHEX Member applications sending the orders to the *OASIS* Server, and as a distributor of the results of the order placement from the central system to the original sender of the order. Pricing information is not sent. *ODL Clients* running the ATHEX Member applications sending orders to the central system and getting the results of the order placement and/or instruction for clearing as well as the system notifications to the ATHEX Members. These applications make use of the ODL-Client-API and the ODL application message interface (Microsoft Windows systems).

5. *FIX Clients* are systems that send orders (to OASIS Server or Xnet Server) and get the results of the order placement as well as the system notifications using the FIX protocol interface.

The procurement of the necessary equipment (hardware and system software) is made by the ATHEX Member. The configuration of the development/test or production environment of the ATHEX Gateway in Member's office is realized following the technical specification and guideline documents issued by ATHEX.

Concerning the ATHEX Gateway Server software that is installed on the ATHEX Gateway(s):

1. It is installed and configured in the development/test or production environment of the ATHEX Gateway by the ATHEX Member technicians.
2. It receives all the information relevant to the ATHEX Member transactions posted either through terminals running the OASIS/ETW–ORAMA applications or through ODL Clients. This information is distributed to the appropriate ODL Clients. Additionally the ATHEX Member may also use a FIX Client to receive and send information using the FIX protocol.
3. It provides export capabilities of all the messages exchanged through the ETS_Service module to the ATHEX Gateway Admin interface. These messages constitute a superset (e.g. orders are included) of the records in the Members Transaction File downloaded by the ATHEX Members after the trading session ends using the OASIS/ETW – ORAMA application.
4. It can operate under the following operating modes:

- Primary server with backup functionality.

  Under this mode, it provides all the functionality described above and additionally forwards all the input and output application messages to the backup server, so that the latter will be ready to operate as primary if needed.

- Primary standalone server without backup functionality.

  It is identical to the above mode with the only exception that it doesn't communicate with any backup server.

- Backup server.

  The only functionality provided under this mode is the receipt of all the application messages from the primary server and their storage in a local disk. It cannot connect to the ATHEX Servers and cannot accept any requests from a client.

- Multiple servers. (i.e. setup of each is identical to a primary standalone server setup)

  Under this mode, any number of gateways runs concurrently. *All* available gateway servers are recipient of *all incoming* (member-wise) information (messages) and are therefore equally 'informed' at any point in time. This not only maintains all advantages of the primary-backup but also achieves seamless failover and increased order throughput.

5. It provides service administration capabilities. First it can be used for monitoring and changing ATHEX Gateway's operating mode (if it is the primary or the backup server). It can also be used to manipulate the list of ODL Clients that can be connected to the ATHEX Gateway.

Concerning the ODL-Client-API software installed on the ODL-Client(s):

1. It is a distributed application using Microsoft COM/DCOM technologies.
2. It is installed and configured in the test or production environment of the ATHEX Gateway by the ATHEX Member staff (see guidelines in "ATHEX Gateway System Preparation Guide" and "ATHEX Gateway Installation Guide" documents).
3. Its basic structural element is the ETS_Broker.dll module, providing in short the following basic services: (for more detail see Chapter 4):
   - 3.1. Activation – deactivation of the ATHEX Member applications connection to the ATHEX Servers. (ETS_Broker.Administrator)
   - 3.2. Transaction data retrieval during a session for the current or previous dates sessions. (ETS_Broker.Administrator)
   - 3.3. Registration – unregistration of a user (ETS_Broker.Broker, ETS_Broker.Administrator)
   - 3.4. Synchronization mechanisms for the communication with the ATHEX Servers after the appearance of technical problem (e.g. disconnection) (ETS_Broker.Broker, ETS_Broker.Administrator)
   - 3.5. OASIS-CTCI messages formatting and decomposition (ETS_Broker. Broker)
   - 3.6. ATHEX Gateway administration capabilities which include the following:
     - monitoring and changing of the ATHEX Gateway's operating mode
     - manipulation of the list of valid ODL clients that can be connected to the specific ATHEX Gateway

The following figure presents the architecture of the ATHEX Gateway with the basic modules it consists of.

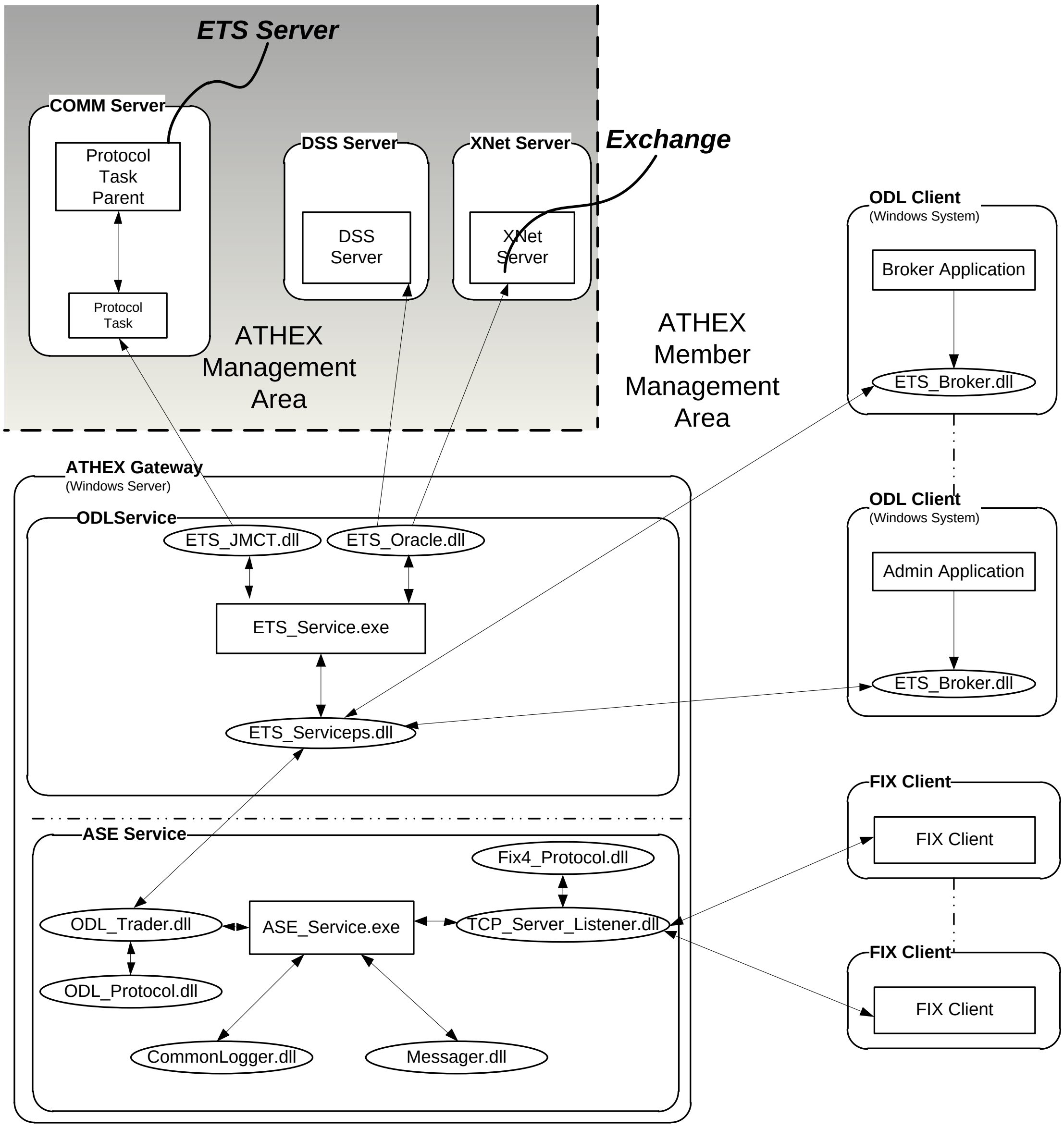


Figure 2-1 ATHEX Gateway architecture

Figure 2-1 also depicts the two different kinds of applications on the ODL Clients into the ATHEX Member management area:

- the Admin Application (AA) and
- the Broker Application (BA).

The fundamental difference in programming terms lies in the fact that the AA uses the Iαdmin interface whereas the BA uses the IBrokerETS interface. A more detailed description of these objects will be done in a following chapter.

Figure 2-1 also depicts the capability of using one or more FIX Clients to communicate with ATHEX Gateway.

A brief description of the modules (ETS_JMCT, ETS_Oracle, ETS_Service, ASE_Service, ODL_Trader, ODL_Protocol, CommonLogger, Messager, Fix4_Protocol, TCP_Server_Listener) of the ATHEX Gateway-Server follows here below:

- ODL Service module

ETS_JMCT	A module in DLL form, handling the communications with the ATHEX Comm Server. Its role is the administration of the communication part of the protocol; not of the protocol as such. More specific, first it creates the sockets (control, data) and, in sequence it makes the connection to the Comm server through the control socket and waits for the Comm Server to connect to the data socket. When this happens, it notifies its creator (ETS_Service) that the connection has been created and is ready to send messages either through the control or the data channel. Sending and receiving messages happens in a completely asynchronous way. This is realized using threads specifically created by the ETS_JMCT module.
ETS_Service	It is a Microsoft Windows Server service that implements the IBrokerETS and Iadmin interfaces described later on. It also accepts, as notification interfaces, the Ireport for IBrokerETS and Imonitor for Iadmin.
ETS_ORACLE	A module in DLL form, handling the communications with the Xorder Server (Oracle). This library establishes connection to the remote Oracle database through an oracle client and manages message transmissions between ATHEX Gateway and the Xnet server.

- ASE service module

ASE_Service	It is a Microsoft Windows Server service that starts and controls the FIX Protocol communication. It routes messages from TCP_Server_Listener to ODL_Trader and vice versa.

	It also checks the state of all components bellow applying recovery mechanisms in case of malfunction.
ODL_Trader	A module in DLL form, using the ODL_Protocol module to translate the message objects to ODL text message.
ODL_Protocol	A module in DLL form, implementing the translation of the ODL application messages received from Comms Server or Xnet Server to internal message objects directed to ASE_Service. Additionally, it translates the internal message objects received from ASE_Service to ODL application messages directed to Comms Server or Xnet Server.
CommonLogger	A module in DLL form, responsible for logging and configuring the FIX Protocol communication
Messager	A module in DLL form, responsible to create and handle message objects for the FIX Protocol communication.
Fix4_Protocol	A module in DLL form, implementing the translation of the stream of characters received from FIX Client to an internal message object which is sent to ASE_Service. Additionally, it translates the internal message object received from ASE_Service to stream of characters to be directed to FIX Client.
TCP_Server_Listener	A module in DLL form implementing the TCP transport layer of the communication with the FIX Client. It operates as a server accepting connections from remote FIX Clients.

A brief description of the ETS_Broker module of the ODL-Client-API application follows:

ETS_Broker	It is a COM module (DLL) in which objects for the order formatting, order decomposition and message exchange with the central system are implemented. More specifically the role of the ETS_Broker module is to provide the user the ability: - to create business logic objects without caring about the formatting process of the message. - to extract the message type that either has been stored into a database or received by the Comm Server.

	- to create a business object from a message (character stream) decomposing it to its components and check or modify one of them. - to create from a business object a message in order either to send or to store it as a message. - to send a message either asynchronously (non-blocking mode) or synchronously (blocking mode)

2.1. Transaction and connection security

Security is one of the major points in the ATHEX Gateway design and implementation. The ATHEX Gateway provides full security at the connection creation level.

2.1.1. *Use of ODL Clients*

When ODL clients are used, ATHEX Gateway (through the ODL service part) provides full security at the level of a remote function call from an ATHEX Member application, as well. It is based upon the security characteristics offered by the COM/DCOM technology, the Microsoft Windows Operating System and the validation of the machines used to run the ODL clients. In order for the ATHEX Gateway and the ODL clients systems to be able to function, the creation of at least one Microsoft Windows domain at the ATHEX Member site is required, so that the authentication and authorization features provided by the LAN Manager, can be used.

The ATHEX Gateway receives calls only from users belonging to a specific user group, depending on the notification interfaces they will use. The local network administrator must define two groups: ETS_Admins and ETS_Brokers containing users that will use the Iadmin and IBrokerETS interface respectively. The ETS_Service module authenticates every connection attempt (initiated by an application) by checking if the attempting user belongs to the appropriate user group. Furthermore, all the calls to ETS_Service are encoded and authenticatedotherwise they are rejected and not handled by the ETS_Service. This happens because the ETS_Broker dll modifies the security level at the maximum level provided by Microsoft Windows and ATHEX Gateway checks if calls belong in this security level. The ETS_Service must be executing under a specific account from which all the permissions are inherited. These settings are made using the *dcomcnfg* application and the *Services* applet.

For example, in a system with defined groups of ETS_Admins and ETS_Brokers, if an application wants to connect and use the IBrokerETS interface, the steps it has to follow are:

1. The application (in ODL Client) uses the Username, Password and Domain of the member to call the ***Register*** function of the Broker object.
2. The Security ID of the user will then be checked in the Domain Controller of the domain entered in connection form by the user.
3. Attempt to connect with the ATHEX Gateway.
4. If the Security ID does not belong in the two groups (ETS_Admins or ETS_Brokers) or is invalid, then the LAN Manager rejects the connection.

5. If the Security ID belongs in one of the two groups (ETS_Admins or ETS_Brokers), then the LAN Manager accepts the connection and the ***Register*** function of the ATHEX Gateway is called.
6. The ATHEX Gateway checks if the computer that originates this call is a valid one. This check is performed against a list of valid computers that is kept locally in the ATHEX Gateway. If the specific computer is not a valid one, the logical connection fails, otherwise it proceeds to the next step.
7. The Registered user is then checked, if he belongs to the appropriate user group and if not a failure attempt is logged (hacker attack) and the logical connection fails.

The same of course stands for the case of the Iadmin interface validations.

As far as the ODL client system(s) side is concerned, the administrator must define that the ETS_Service user has access permission in order to be able to use the notification interfaces. The *dcomcnfg utility* is used in this case, too.

Consequently, the order entry facility and the rest of the functionality provided by the ATHEX Gateway are possible only after the authentication of the corresponding user who must belong to the ETS_Admins or ETS_Brokers user groups. Thus, the ATHEX Member application is wholly responsible for every requirement for extra checks before sending requests to the ATHEX Servers; all the checks and the selection criteria constitute a part of the ATHEX Member application and not of the ATHEX Gateway. All the transactions forwarded and finally received by the OASIS central application or target exchanges are considered actual, real and binding for the ATHEX Member, regardless of the possibility of a human error or a malfunction of the ATHEX Member application.

2.1.2. *Use of FIX Clients*

The ATHEX Gateway (through the ASE service) will authenticate the identity of a FIX Client by examining the credentials submitted in the login process. If the FIX Client is successfully authenticated, the ATHEX Gateway responds with a confirmation message. If authentication fails, the ATHEX Gateway shuts down the connection. The FIX Client must wait for the confirmation message from the ATHEX Gateway before declaring the session fully established.

3. Application Development for ATHEX Gateway

With the ATHEX Gateway, ATHEX members are given the opportunity to develop applications that will communicate directly with the trading system OASIS or the Xorder Server. These applications will be responsible for entering orders, receiving confirmations and other information that will allow them to correctly manipulate these orders.

3.1. Application Types

Throughout this document, the usage of the term "application" is done to denote the following categories:

- The applications that compose the ATHEX Gateway itself.

- The ODL applications:

    - The elementary ODL applications that are the simplest possible applications that use the ATHEX Gateway.

    - The ATHEX members' integrated ODL applications that use the ATHEX Gateway.

- The FIX Client applications which do not make use of the ODL-client-API and communicate with ATHEX Gateway using the FIX Protocol interface.

3.1.1. *ODL Applications*

In order to build the integrated ODL applications that use the ATHEX Gateway, portions of software provided by the ATHEX must be used. This is special 'middleware' software that composes the ATHEX Gateway that is divided in the following two parts:

1. The ATHEX Gateway Server application, running as a service on the ATHEX Gateway machine.

2. The ODL-client-API. This is the only software part directly used by the member's applications. The ODL application programmers should be well aware of this part. In order to aid the application programmers in their task, the ODL-client-API has been designed in an object oriented programming manner. The functionality provided is exploited in the form of objects the application has to create or manipulate.

3.1.2. *Elementary ODL Applications*

These are the simplest applications that can be constructed, making use of the ODL-client-API and are categorized by their primary functionality in terms of the ATHEX Gateway as follows:

- **Administrative Application**, which uses the Administrator object for:

    1. the manipulation of the ATHEX Gateway

2. the connection of the ATHEX Gateway with the ATHEX Servers
3. the retrieval of the entire set of information (even from previous dates), regarding the ATHEX Member.

An administrative application has no way to send messages to the system. It must be noted that this application can also use the ATHEX Gateway that operates in backup mode. Under this mode, the connections of the ATHEX Gateway with the ATHEX Servers are not possible.

- **Order Transmitting Application(s)**, using a Broker object to enter, modify, activate, deactivate, cancel and monitor orders and to receive Member's credit limit information. The simultaneous use of multiple Order Transmitting Applications from the same ODL Client system of an ATHEX member is not possible. It must be noted that this kind of applications cannot use the ATHEX Gateway that operates in backup mode.

- **Clearing & Settlement Application(s)**, using a Broker object to request registry, settlement and clearing services as well as to receive information on subscription basis.

3.1.3. *Integrated ODL Applications*

An integrated ODL application, which uses the ATHEX Gateway and more particular the ODL-Client-API application, fully implements the business logic, offering the entire set of functionality its users need. It always includes an order transmitting part (which logically corresponds to an order transmitting application), optionally an administrative part and a clearing/settlement part. A logical correspondence is stated below:

1 (one) integrated application → 1 (one) order transmitting application → 1 (one) clearing/settlement application → 1 (one) ODL-Client-API application functioning in 1 (one) ODL-Client system.

3.1.4. *FIX client Applications*

A FIX client application can be any FIX client that implements the FIX messages as they are decribed in Chapter 5: FIX client application.

Additionally, a FIX client connection can be properly configured to receive the entire set of information, regarding the ATHEX Member. Such a FIX client connection has no way to send application messages to the system.

3.2. Information Flow

Regarding the information flow among an integrated application from one ATHEX member and the ATHEX Servers the following terms apply:

The connection (and the disconnection) of the ATHEX Gateway with the ATHEX Server is in the task sphere of an administrative application. It is a required step for Information Flow to exist.

The order transmitting part of an integrated application transforms the objects to messages and passes the messages to the ATHEX Gateway. By using the "ETS" connection, the ATHEX Gateway passes the messages to the OASIS –CTCI Comm Server, which finally sends them to the OASIS Server. By using the "ORA" connection, the ATHEX Gateway passes the messages to the Xnet server, which finally sends them to the target exchange. By using the "DSS" connection, the ATHEX Gateway passes the messages to the DSS server.

The opposite information flow is performed as follows: When an ATHEX Server has to send a message to an ATHEX member it sends it to the member's ATHEX Gateway(s). Each ATHEX Gateway has the ability to determine the application that must receive the message and notifies it to pick the message up.

This mechanism will be explained in greater detail at other parts of this document.

A representation of the above can be viewed in the following figure:

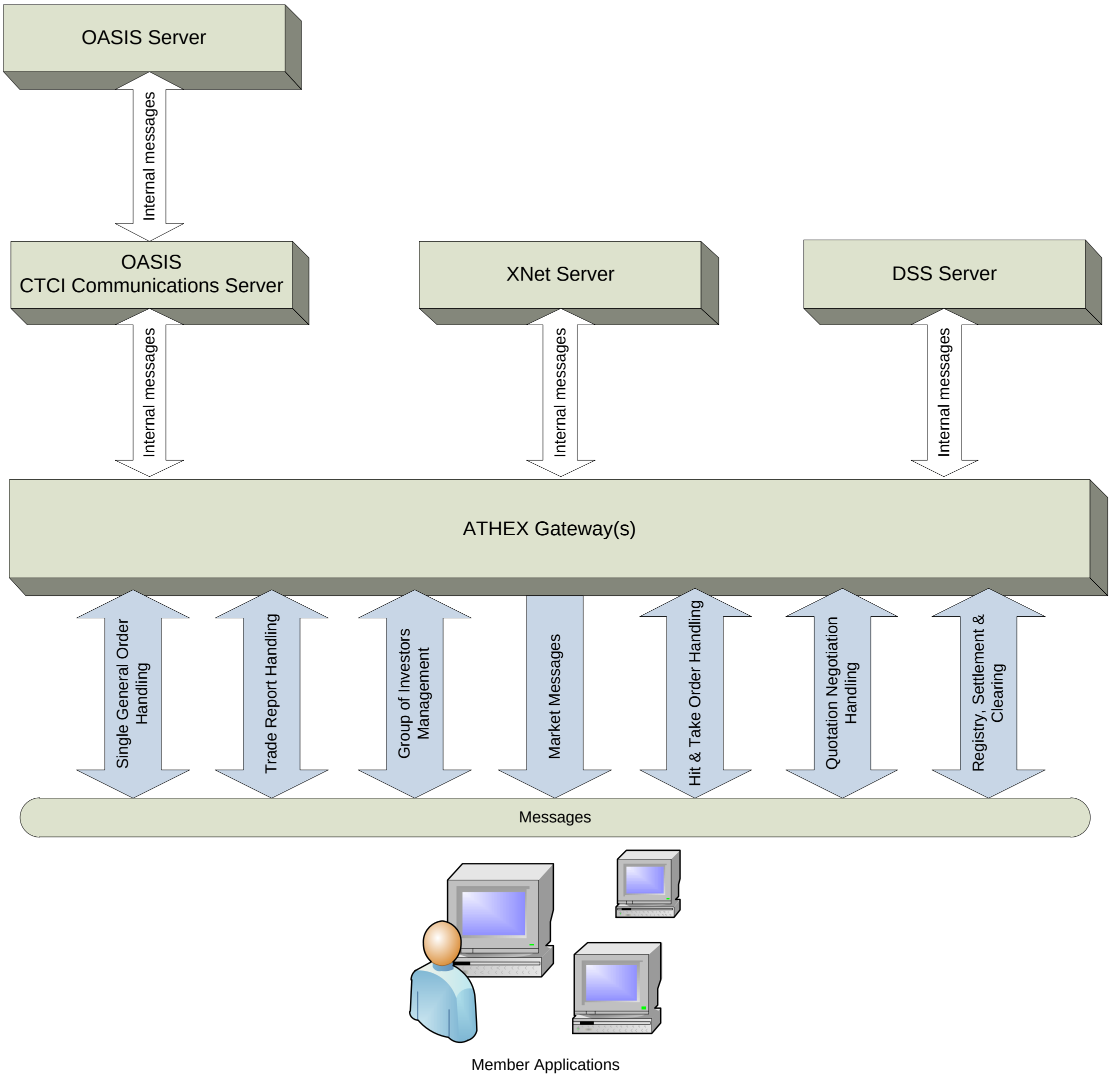


Figure 3-1, Message Path from the Members' Applications to the ATHEX Servers

The transfer (entry, modification, cancellation) of orders, quotesand trade reports on behalf of an ATHEX member to the central system OASIS Server can be performed by:

- The OASIS-ORAMA terminal, or
- The application EMRW, used in the ATHEX Supervision, or
- The ATHEX member's order transmitting application

Regardless the case, a proper confirmation of the transmission of the message is being recorded in the ATHEX Gateway. In more detail:

1. It is available to the administrative application, which is connected to the ATHEX Gateway system

Regarding the information from the central OASIS Server:

1. If it is an order transmission confirmation or notification for a new trade, then it is sent to:
    a. the terminal or the order transmission application that originated the order transmission, and
    b. the administrative application(s) that is connected to the ATHEX member's ATHEX Gateway machine(s)
2. If it is information regarding the system status, the market status, the securities status then:
    a. the system sends notification to all the connected terminals (OASIS-ORAMA and EMRW), and
    b. the information is being transmitted (as a message) to all ATHEX Gateway systems of all the members, and more specific to the following destinations:
        i. the administrative application(s) that are connected to the ATHEX Gateway system(s)
        ii. the order transmitting application of the member
3. If it is information regarding the Member's credit limit or information regarding the available orders on which Hit and Take orders can be entered, then it is being transmitted (as a message) to the ATHEX Gateway systems of all the members, and more specific to the following destinations:
    a. the ODL administrative application(s) that are connected to the ATHEX Gateway system(s)
    b. the ODL order transmitting applications of the member

Regarding the information from the Xorder server, if it is an order transmission confirmation or notification for a new trade, then it is sent to:

   a. the terminal or the order transmission application that originated the order transmission, and

   b. the administrative application that is connected to the ATHEX member's ATHEX Gateway machine

Regarding the information from the DSS server:

1. if it is a confirmation on a request submission or notification for results of requests then it is sent to:

   a. the terminal or the Clearing & Settlement application that originated the request transmission, and

   b. the administrative application that is connected to the ATHEX member's ATHEX Gateway machine

2. if it includes information about events on DSS then it is sent to all ATHEX Gateway systems of all the members, and more specific to the following destinations it:

   a. the terminal or the Clearing & Settlement application, and

   b. the administrative application that is connected to the ATHEX member's ATHEX Gateway machine

As a consequence, the integrated applications of the members that are making use of the ATHEX Gateway have the option to use the ODL administrative application in order to receive messages for:

- Transmission of their orders
- Confirmations of transmissions of their orders
- Notifications for trades performed by one of their orders
- System status, market status, securities status
- Orders for which a Hit and Take order can be entered
- ATHEX Member's credit limit information
- Transmission of requests for service about registry, clearing or settlement process
- Results of requests for service about registry, clearing or settlement process
- Information about events on DSS

The FIX Client applications of the members that are making use of the ATHEX Gateway may receive messages for:

- Transmission of their orders
- Confirmations of transmissions of their orders
- Notifications for trades performed by one of their orders
- System status, market status, securities status
- ATHEX Member's credit limit information
- Quotation negotiation handling messages
- Trade report handling messages
- Exchange notes

3.2.1. *OASIS Server*

The OASIS Server, in relation to messages, is responsible for:

- the Verification of incoming messages, that is checking whether the messages have the correct format (length, type of fields, etc)
- The Validation of incoming messages, that is checking whether the various fields include prices that correspond both to the Market Business Logic (correct market, security, member symbol, etc) and the market conditions on that particular moment (not suspended security, not active member, etc).
- The processing of incoming messages
- The creation of outgoing messages to the Member's applications.

3.2.2. *OASIS-CTCI Comm Server*

OASIS-CTCI Comm Server is responsible for:

- The implementation of communication between the ATHEX Gateway and the OASIS Server.
- The Authentication of Messages from Members, which is the identity confirmation of the person that sent the message.
- The implementation of the Recovery Mechanism, that is the mechanism ensuring the smooth continuation of operation following a temporary interruption of communication due to technical problems either at the ATHEX Gateway or the OASIS Server.

3.2.3. *Xnet Server*

Xnet Server allows the market participants to have indirect market access to the trading servers of the Exchanges connected to Xnet. Xnet connects the Xnet Member who is the source of the order to the Xnet agents who have the responsibility to enter the order in the trading platform of the target Exchange. This service frees the user from the specifics of inter-Exchange access/connection, order-routing and transaction security, and eases the user's access to those markets via a simple FIX-based API.

3.2.4. *DSS Server*

The DSS Server is the electronic system for the book entry and monitoring of transferable securities, for the monitoring of their changes and the overall monitoring of the procedures provided for by the DSS Rules and Operation Regulation.

3.2.5. *ATHEX Gateway Server*

The ATHEX Gateway Server is responsible for:

- The implementation of communication among the Member's applications (through the ODL and/or FIX Clients), the trading system and the forwarding of relevant messages through the ATHEXnet.
- The verification of the length of messages received from the Member's applications (no data verification is carried out).
- The completion of the MemberID field with the Member's ID on the Header of the message.

3.2.6. *Member's Application*

The Member applications through the ATHEX Gateway have the ability to connect and exchange messages with OASIS Server and/or target exchanges connected to Xnet by making use of the objects implemented by the ODL-Client-API (ODL Clients) or by making use of the FIX protocol (FIX Clients). The message exchange with the DSS server can only be made using the ODL-Client-API.

3.3. Objects of the ODL-Client-API application

The ODL-Client-API application was designed and developed in an object oriented environment. The objects used on the ODL-Client-API can be distinguished in two major categories:

1. **Call Interface:** Objects responsible for the communication of the Member application with the OASIS system, sending and receiving messages and connection management:

    Administrator
    Broker

2. **Application Requests:** Objects responsible for request/reply action between the Member application and the OASIS system:

    OrderEntry
    OrderEdit
    OrderChange
    …

All object types are documented in Chapter 4.

3.4. Connection – Disconnection Procedure

In order for an ODL or FIX client application to be able to connect with the OASIS Server and to participate in the current day's trading session, the following sequence of actions should be completed:

- The OASIS Server should be operating.
- The Comm Servers should be operating.
- The ATHEX Gateway should be operating and the ODL Gateway Service (ETS_Service) should be started.
- In case of ODL Client, it must be contained in the list of valid clients maintained by the ATHEX Gateway.
- In case of FIX Client, the credentials for the login procedure must have been previously agreed by the ATHEX and set in ATHEX Gateway.
- The Client should certify its existence in the service.

In order for an ODL or FIX client application to be able to connect with the Xnet Server, the following sequence of actions should be completed:

- The Xnet Server should be operating.
- The ATHEX Gateway should be operating and the ODL Gateway Service (ETS_Service) should be started.
- In case of ODL Client, it must be contained in the list of valid clients maintained by the ATHEX Gateway.
- The Client should certify its existence in the service.

In order for an ODL client application to be able to connect with the DSS Server, the following sequence of actions should be completed:

- The DSS Server should be operating.
- The ATHEX Gateway should be on and the ETS_Service should be activated.
- In case of ODL Client, it must participate to the list of valid clients maintained by the ATHEX Gateway.
- The Client should certify its existence in the service.

Depending on the configuration of the Members installation, there are two cases with regard to the connection of the ATHEX Gateway (operating as primary ATHEX Gateway) to the Comm Server and/or the Xnet server and/or the DSS Server.

1. **AutoConnect Procedure**. Upon the launch of the ETS_Service (performed automatically through the Windows AT Scheduler), the ATHEX Gateway is automatically connected to the ATHEX Server(s). If only FIX client applications are to be used for communication with ATHEX Gateway, then this connection procedure is the one recommended.

2. **Non Automated Procedure**. Upon the launch of the ETS_Service, an ODL Administrative application (which uses the Administrator object) should certify its existence in the ATHEX Gateway and the ETS_Service will confirm the entry

parameters (name, code, network domain) in order to verify whether the user that attempts to login is valid and belongs in the Administrators group. If the information is wrong, then the user should go through the certification procedure again. Next, the return value of IsConnected function is checked and if it is false (no connection), the ATHEX Gateway connects to the ATHEX Server(s) using the Connect function. It must be noted that the first connection of the ATHEX Gateway in a day is not instant. The application checks the return value of IsConnected function to verify that connection established.

The AutoConnect procedure is provided for the member's convenience and is optional. A member can select to activate it or not by declaring it during installation time of ATHEX Gateway Server application. For detailed information on this option refer to "ATHEX Gateway Installation Guide".

The confirmation mechanism of the ATHEX Server(s) is based on network parameters during the connection of the ATHEX Gateway, as well as to the verification of the name and code of the securities company that attempts to connect. If the information regarding the connection request of an ATHEX Gateway is incorrect, then it is given the opportunity to try again (however, there is a limit to the number of attempts).

After the connection with the ATHEX Servers has been established, the ODL Client applications, by using an Ibroker type object, should certify their existence in the ETS_Service in order to be able to send/receive messages to/from the ATHEX Server(s). The login parameters' validation is made in the same way as has already been mentioned for the case of the Administrator, with the only difference that in this case the user should belong to the Brokers group. After the above procedure has been followed, the ODL Client will be able to successfully connect and participate in the current day's trading session.

In case of a FIX client application, the preagreed credentials should be submitted during the login procedure. After a session between the FIX Client and the ATHEX Gateway has been established, the FIX client will be able to participate in the current day's trading session.

When a communication problem with the Comm Server is detected by the ATHEX Gateway for a specified time period (default 10 sec), then the ATHEX Gateway will send a logout message to the FIX clients and will terminate the FIX sessions. The FIX clients should try to reconnect to the primary ATHEX Gateway at least 3 times and if they fail then they shall try to connect to the backup ATHEX Gateway (if exists) after the failover procedure has finished (see chapter **Error! Reference source not found.** REF _Ref286052090 \h * MERGEFORMAT **Error! Reference source not found.**).

Finally, the proposed disconnection procedure at the end of the session is as follows:

1. **The ODL Client disconnects from the ETS_Service** using the Unregister method of the Broker object. Alternatively, the FIX client terminates the session by sending a Logout message.

2. **The ATHEX Gateway is disconnected from the ATHEX Server(s)** using the Disconnect method of the Administrator object.

3. **The Member's ODL Administration application disconnects from the ETS_Service** using the Unregister method of the Administrator object.

3.5. Registration Procedure for ODL Clients

As mentioned in paragraph 2.1, in case of ODL clients, the existence of a Domain Controller with two groups (Admins, Brokers) is required.

Every ODL application (basic or integrated), which uses the ATHEX Gateway, is required to be certified to the ATHEX Gateway. The ATHEX Gateway, in order to service the sending and receiving of messages, has to be connected with the ATHEX Server(s).

Regarding the certification of a given ODL client application to the ATHEX Gateway, the way in which it is being established will be described below. To simplify the description, basic application terms will be used

1. if the ODL client application has to do with administration, it first creates an object of type Administrator and then calls the Register method of this object using the username, the password and the domain name of a user that has been declared in the Administrators group inside the member firm's Domain. The ATHEX Gateway first checks if the above call originates from a valid ODL Client, which means a client included in the list maintained by the ATHEX Gateway. If it is included, then the ATHEX Gateway checks the above credentials and if they are found valid, it returns a number that certifies the application at an ODL Client machine level. This number (which symbolizes the certification itself) is unique and is valid for all the current day. This means that regardless the number of times the Register method of an Administrator object is called throughout the same day from the same ODL Client machine (assuming correct parameters), the ATHEX Gateway will be returning the same number, thus recognizing the same application. From the time a successful certification is completed, the Administrator object that has been used is functioning as a unique representative of the administration application for its communication with the ATHEX Gateway. If another Administrator type object in the same or in a different ODL Client system inside the member firm calls the Register method gets itself certified, then the ATHEX Gateway recognizes the second object as the representative on behalf of the administration application. The first Administrator type object, which had been certified first, does not communicate with the ATHEX Gateway any more. This is the reason why a single ATHEX Gateway does not support the concurrent operation for more than one administrative application.

2. if the ODL client application has to do with transmitting orders, then it creates an object of type Broker and calls the Register method of this object using the username, the password and the domain name of a user that has been declared in the Brokers group inside the member firm's Domain. The ATHEX Gateway first checks if the above call originates from a valid ODL Client, which means a client included in the list maintained by the ATHEX Gateway. If it is included, then the ATHEX Gateway checks the above credentials and if they are found valid, it returns a number that certifies the order transmitting application at an ODL Client machine level. This number (which symbolizes the certification itself) is unique and is valid

for all the current day. This means that regardless of the number of times the Register method of a Broker type object is called throughout the same day from the same ODL Client machine (assuming correct parameters), the ATHEX Gateway will be returning the same number, thus recognizing the same application. From the time a successful certification has been completed, the specified Broker type object functions as a unique representative of the order transmitting application at the ATHEX Gateway. If some other Broker type object inside the same ODL Client system calls the Register method and gets itself certified, then the ATHEX Gateway recognizes the second object as the representative on behalf of the order transmitting application. The first Broker type object, which had been certified first, does not communicate with the ATHEX Gateway any more. This is the reason why a single ODL Client system cannot support the concurrent operation for more than one order transmitting applications.

Note:

a. Certification of one administration application and one order transmitting application from the same ODL Client system is permitted.

b. Repetitive calls of the Register method from different objects of the same type (or switching 'representatives') have no point and should be avoided.

3.6. Relation between Messages – Objects for ODL application development

The OASIS system has been designed in a way that all the functions at the various layers are based in the exchange of messages (message passing).

As already mentioned before, in order to assist the ODL application programmers, the ODL client API provides objects, which are responsible for:

1. The implementation of the connection with the ATHEX Servers.

2. The manipulation of the messages (Application Requests).

More specific, the objects of the category *Application Requests* correspond on a one to one basis to messages that are being sent from the members' integrated applications to the ATHEX Servers and vice versa. It is obvious that these objects are a great aid in formatting information.

The properties of the objects have the same names and the same meaning with the fields in the corresponding messages. Appendix B. ODL-Client-API: ETS_Broker properties, lists the properties of the objects that are used by the programmer and the types of the corresponding fields in the relative messages.

The ATHEX Gateway system exclusively manipulates messages. The mechanism used to associate messages with objects is implemented by the ODL –Client – API.

3.6.1. *Formatting and transmission*

To describe the way messages are formatted and transmitted to the ATHEX Gateway the following should be first mentioned:

The Member’s application has already been connected and is in a state ready for order transmission. This is accomplished by having created an object of type Broker (Call interface) and having checked the return value of the IsConnected function of this object. Furthermore, an object from the application requests category is required. For the scope of the current example, we will refer to an Order type object. **All** properties of these objects must be set to valid values. The conversion to message form and the transmission of the message are performed in the series of following steps:

1. To convert the Order type object to string representation, the FormatMessage method of this object must be called. The return value is the requested string.

2. The return value of the CanSend function of the Broker type object is checked to determine whether the ATHEX Gateway will be able to accept the message.

3. Finally, the SendMessage method of the Broker Type object is called, using the string produced in step 1. The message arrives to the ATHEX Gateway, which in turn returns to the member’s application two values. The first value is the incremental sequence number of the message at the context of the current connection phase. The second value is a code, which can be used in order to seek the current message at the ATHEX Gateway, either in the current day or even in future days.

Note: For optimal performance, the ATHEX Gateway performs only sanity checks on the message being transmitted. It passes the message to the appropriate ATHEX Server. **It is the members’ applications responsibility to correctly fill the required information before sending the message.**

The described data flow is depicted in Figure 3-2.

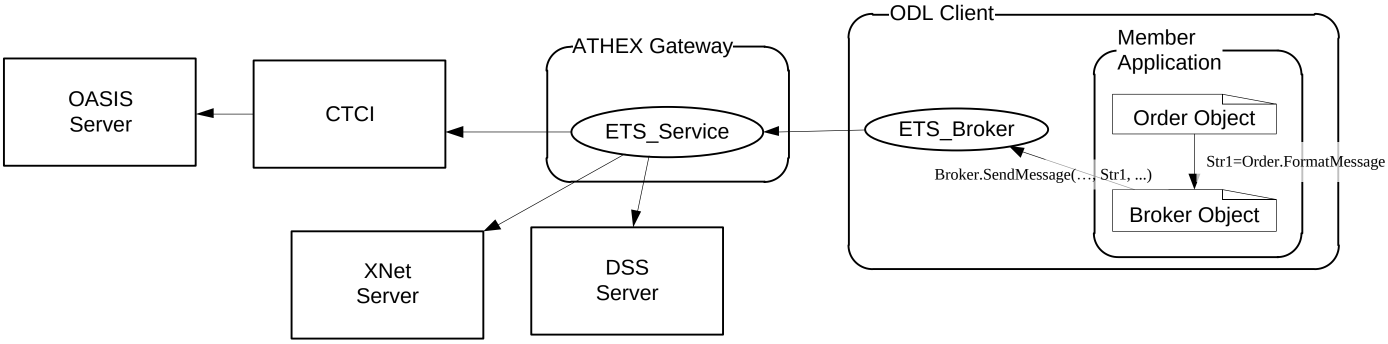


Figure 3-2. From Object to Message

3.6.2. *Receipt of incoming Message and Creation of Object*

In order to receive an incoming message, a mechanism called Notification Interface is used. This mechanism is a feature provided by DCOM and allows the ATHEX Gateway to call functions of the Broker or Administrator objects. The creation of a Broker or Administrator object is required. The object has to be connected to the ATHEX

Gateway and there must also be a message, which is directed from the ATHEX Server towards the member's application. The required steps are presented in more detail below:

1. The ATHEX Gateway and specifically the ODL-Server application receives the message and finds out which application that is making usage of the ODL-client-API the message is directed to. It must be reminded that a single ATHEX Gateway has the ability to service more than one application using the ODL-client-API (one application per ODL Client machine).

2. By using the notification interface, the ETS_Service module from the ODL-Server application, notifies the ETS_Broker module from the ODL-client-API, and transfers the message. Inside the member's application using the ODL-Client-API, inside the Broker type object, an incoming NewETSMessage event arrives, containing the incoming message.

3. The GetMessageObject method of the Broker type object is called. This call results in the creation of an object in the Application Requests category and in the filling of the fields of that object from the products of the decomposition of the initial incoming message. This method also provides information for the type of the newly created object.

As aforementioned, a multiple concurrently active gateway scheme may be applied by a member, in which case all deployed gateways are set to receive copies of incoming of each messages to support fault tolerance. Consequently, any *single* application that is connected to more than one gateway will be *receiving*, by definition, *the same message as many times as the number of gateways it is connected to*. It is hence in the member's application discretion to properly handle the duplicated information.

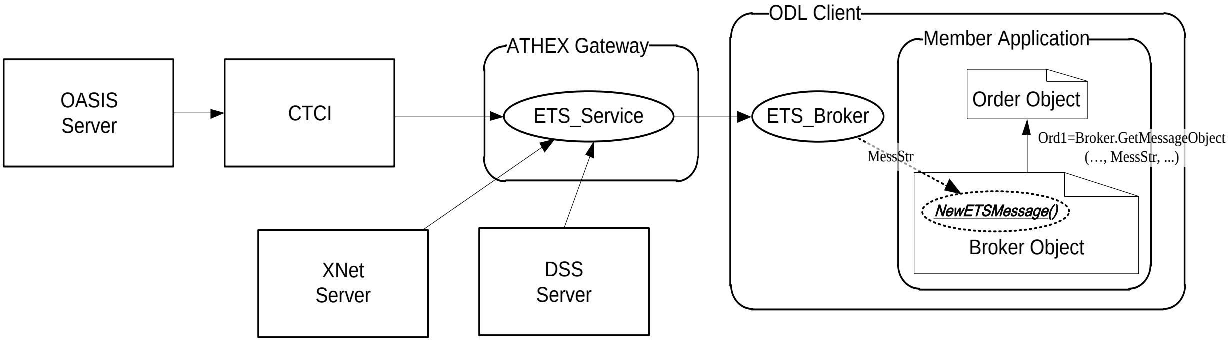


Figure 3-3 From Message to Object

3.7. Protection Valves

In order to limit the risk of overload of both the ATHEXnet and the ATHEX Servers, a number of protection valves has been set, which control the inflow rate of messages.

In the system there are 3 valves that concern the transmission of messages:

1. In ODL Client for messages it sends to each ATHEX Gateway.

In accordance to the current settings, the ODL Client does not allow the transmission of a new message if it has not received a response (confirmation or rejection) to a predefined number of previous messages it has sent. This number is configurable and its default value is 3. It must be noted that the ODL Client shall use only this response in order to be sure that the sent message has reached its destination.

2. **In ATHEX Gateway for outgoing messages to the Comm Server.**

   An ATHEX Gateway may send the Comm Server up to five (5) messages by default (9 maximum) without receiving a response. Obviously, on a member level, this is multiplied by the overall number of available gateways in a multiple gateway scheme (excluding backup servers)

3. **In the Comm Server machine for messages coming from the ATHEX Gateway.**

   The Comm Server checks the messages it receives. In the present status this limit has been set to ten (10) unanswered messages per gateway. If the ATHEX Gateway is used and operates correctly this limit will never be exceeded (because the valve at point (2) will be activated). In the event, though, that this limit of ten messages is exceeded, then that specific ATHEX members ODL will be automatically deactivated and, in order for it to be restored, a procedure should be launched through the ATHEX.

3.8. Recovery after failure

3.8.1. *Recovery between Comm Server and ATHEX Gateway*

Comm Servers employ complicated mechanisms to regain lost information. This happens due to the fact that they must be able to do a double recovery if needed; a recovery from the OASIS server part and a recovery from the member ATHEX Gateway part.

A simple description of a recovery from the Member ATHEX Gateway follows:

1. ATHEX Gateway logs in the Comm Server.

2. Comm Server sends to ATHEX Gateway the last messages it received.

3. ATHEX Gateway sends to Comm Server the last message it received.

4. Comm Server will then check if the results match otherwise it will look in the spool files to find the remaining messages that the Gateway needs and will post them.

5. ATHEX Gateway stores all the messages it receives from the Comm Server to the local disk. In case that these messages are totally lost, the ATHEX Gateway will request them again from the Comm Server. **If the number of these lost messages is outside limits that are defined as secure, then the Comm Server will not allow the connection to be established.**

Other parameters that can lead to a connection rejection are the number of **login attempts** as well as the number of **recovery requests** to obtain lost messages. In such case, please contact ATHEX (refer to Chapter 1: Introduction).

In normal operation the ATHEX Gateway does not use the recover mechanism at all.

3.8.2. *Recovery between ATHEX Gateway and ODL Client*

This mechanism is transparent to the Comm Server because it is done completely in the member's side and only the ODL Gateway Service is responsible for this task. In cases that the recovery is the only way to regain lost information (e.g. disk failure) then the procedure is quite simple and work as follows:

1. Member ODL Client → ATHEX Gateway Request messages from number XXXXXX

2. ATHEX Gateway --> Member ODL Client Replies with the messages from number XXXXXX and all the later messages.

The request for retransmitting messages from the Client side is done with the use of the function GetLostMessages or the function SendLostMessages, which are described in chapter 4.

Finally, the administrative ODL application (making use of the IAdmin interface) has the ability to retrieve from the ETS_Service all the messages from a previous date, in order to analyze them or store them somewhere else. The number of the past days for which data is kept, is limited only by the storage resources on the ATHEX Gateway machine and is an element of the administrative policy set by ATHEX Member for the ATHEX Gateway.

3.8.3. *Recovery between ATHEX Gateway and FIX Client*

Fault tolerance is handled with two complementary strategies.

The first strategy is the way the software is designed and written. "Defensive programming", exception handling and various fault tolerance patterns have been applied in order to make the software resistant and reliable.

The second strategy is achieved with the complete implementation of FIX session recovery semantics. Network disconnections and failures are handled by the session management recovery functionality. Every network malfunction in the FIX communication part drives the ATHEX Gateway in a clean startup state ready to accept the next connection from remote client. Afterwards a synchronization mechanism with the client is activated, permitting a full recovery of the previous lost Fix session. This mechanism is provided by the ASE_Service. In cases that the recovery is the only way to regain lost information (e.g. disk failure) then the procedure is quite simple and work as follows:

1. Member FIX Client --> ATHEX Gateway Request messages from number XXXXXX to number YYYYYY

2. ATHEX Gateway --> Member FIX Client Replies with the messages from number XXXXXX to number YYYYYY.

The request for retransmitting messages from the Client side is done with the use of the FIX session *Resend Request* message, which is described by the FIX Protocol[1].

3.9. Fast failover

3.9.1. *Fast failover in case of primary – backup ATHEX Gateway configuration*

If the Member uses a primary – backup ATHEX Gateway configuration, then the primary ATHEX Gateway forwards to the backup ATHEX Gateway through TCP/IP all the information that is logged in disk file. The backup ATHEX Gateway in its turn logs this information locally. If some problem occurs and the ETS_Service in the primary ATHEX Gateway cannot be restarted, then the backup ATHEX Gateway (and, subsequently, its ETS_Service) can be restarted operating as primary server. In this case the state can be regained from the logging information stored locally, which will lead in fast recovery. Even in the case the logging information is incomplete or missing, there is always the capability to recover its state from the ATHEX Server(s).

3.9.2. *Fast failover of data in case of multiple ATHEX Gateway configuration*

It is worth underlining here a notable feature offered by a multiple gateway scheme. In the unlikely event of a gateway entering an unrecoverable state, the member can resume seamlessly through a different gateway, under the safe assumption that all gateways are equally informed of all incoming (member-wise) messages. This scheme offers a solution similar to a primary-backup model, but improves on switching over overhead.

Both failover mechanisms are described in more detail in appendix F.

3.10. Error Management

During the operation of an integrated member's application, some errors may occur. These errors can be classified in one of the following categories:

1. Errors in the content (syntax) or the timing of the messages.

2. Errors in the calls that are provided by the various objects of the ODL-Client-API application.

3. Errors that originate from the rest of the subsystems inside the Microsoft Windows Operating System, which are used by the member's application.

[1] For information on FIX Protocol, please visit the following site http://www.fixprotocol.org

3.10.1. *Errors in the content of the message*

The equities trading part of the OASIS provides a mechanism that recognizes errors and returns the proper indications to the user, by the usage of the Reject type objects, so that the user should be in an appropriate position **to understand the error condition and correct the problem**. The rejections arrive as the result of a previous user action and never in any other case. Usual actions that produce rejections are the insertion, modification of orders using one or more incorrect parameters, the cancellation of orders that have already been cancelled, attempts that should result in the member's credit limit violation, etc.

The ATHEX Gateway transfers certain types of messages, which by itself is a natural limiting factor in the number of the reject messages that can appear at the member's application side. The primary field inside every rejection message is the error code.

The error code is a number, rating from 1 to 999 and denotes the type of the error that has been intercepted. It must be pointed out that only one reject message is issued per user action (entry/change/activation/deactivation/cancellation of an order and entry/approval/disapproval/cancellation of a trade report (preagreed price trade)). **If in the same action there are more than one errors, then the system will issue a single error message with the error code of the first error condition that occurred during the message's inspection.**

The developer of the Member's Client Application should follow a defensive approach to minimize rejection messages.In order for something like that to be accomplished there must exist:

1. An **error trapping and monitoring mechanism** at members' application layer. This mechanism will have the objective to be constantly ready to accept a rejection message and be able to signal it via a high priority mechanism to the member's application. It is vital that the reject messages should be immediately processed and not get delayed by various program structures (lists, queues, files), which would leave potential margin for additional reject messages in the mean processing time.

2. A mechanism that would be able to **combine the reject message with the action that caused it**. Without such a facility, many of the reject messages are practically rendered useless because it would not be clear any more which part from which action caused the rejection. A valuable aid in the task of determining the faulty action is the encapsulation of the original message (the one that caused the rejection) inside the reject message itself.

3. Knowledge of what has to be done when a reject message is intercepted. There are the following categories of actions:

    a. **Correction of the faulty action and immediate retry**. In this case the reject message usually informs for some invalid value, which has been provided by mistake, therefore after the correction has taken place the transmission can be immediately retried without affecting any other element inside the member's application.

b. **Correction of the faulty action, wait for retry or cancel the action**. In this case the rejection has originated from some reason with a temporary time scope, such as insufficient member credit limit value, security in temporary halting, etc. The member's application must first be notified of the condition and not proceed to further actions which may cause a second rejection (in the case of the insufficient credit limit above, the application should wait until either the member's credit limit is raised or some of the member's orders are cancelled). The same applies to all actions that result to a rejection message but the member decides not to reintroduce them any more. In many cases, the factor to determine the proper timing to repeat an action (triggering event) can be a message that will arrive to the member's application via the ATHEX Gateway, i.e. market opening, ceasing of the temporary halting for a single security, etc.

c. **Cancellation of the action and no future attempt to repeat it**. In this case, the trading system informs the member's application for something permanent and not reversible (at least in the time scope of the current day). The action that caused the initial rejection must not be reattempted, neither itself, nor any other action that it is now known that will bring the same rejection message. For example, the reject code 013 (Invalid Security) signals that the security code that had been used was invalid, therefore this security code should not be furthermore used, at least in the time scope of the current day.

It should be pointed out that the best way is the **prevention** of reject messages instead of the corrective actions after a reject message has arrived. Therefore, the member's application **must be aware of system data** (securities, markets, market phases, users, etc, as well as orders that have been entered up to the current timestamp, trades, order/trade cancellations) and must have developed such a mechanism so **it can determine** (at the best grade possible) **if the action will be accepted or not without the need to actually perform the action on the real system**.

Appart from the rejection messages that are returned by the OASIS Server, the user has to be sensitive to the error codes that are returned by calls to the various methods of the ODL-Client-API objects if an ODL client application is used.

3.10.2. *Errors from Calls to Methods of ODL-Client-API Objects*

All calls to COM objects return an error code. If the call is successful then the number zero (0) is returned, all other cases denote an error. If an error occurs, an exception is always generated and a piece of the information provided to the exception handler is the error code.

Error codes can originate from various DCOM layers as well as from the object that implements the DCOM calls. The error codes that can be returned from the ODL subsystems are presented in Appendix D. Error codes. These error codes are also presented at Chapter **4**, as comments to the functions from which they may originate. They are described starting with the ETS_ prefix, and are characterized by an error code part of 0x800425__.

3.10.3. *Operating System Errors*

There are several error codes, which indicate errors produced at various subsystems of Microsoft Windows, such as COM, RPC, TCP, etc. For each one of these errors the programmers can find a small description by using the **errlook.exe** tool from Microsoft. In Chapter **4**, there are reference points for some of the messages in this category that can occur as a direct consequence of calling functions of the ODL-Client-API application.

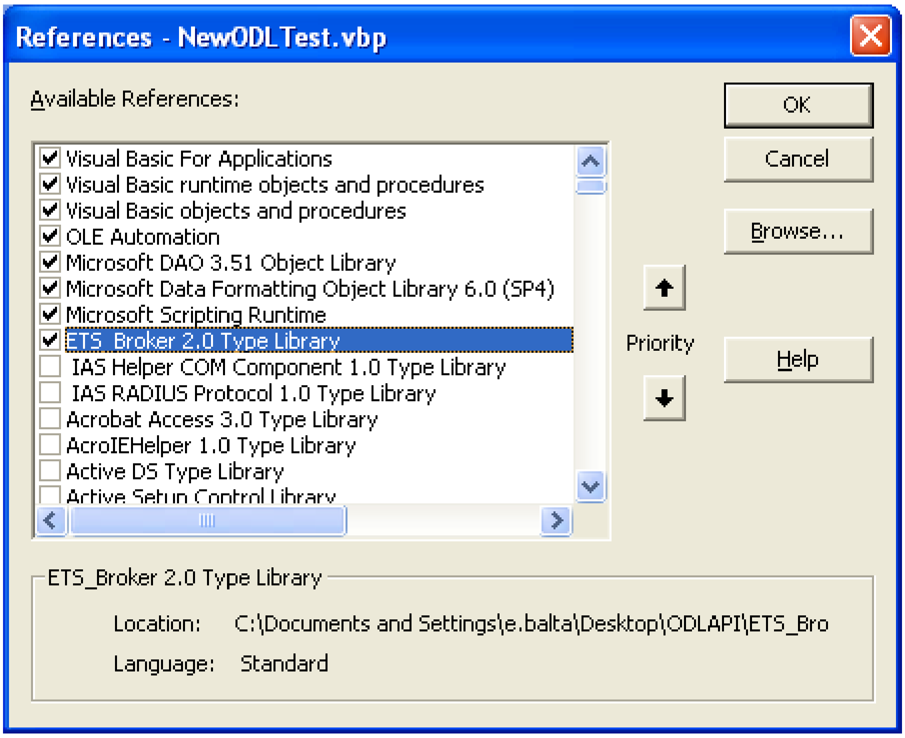
4. ODL-Client-API application

In this chapter there is a detailed description of the objects of the ODL Client API application and their proper use from the members. There is also an example in Visual Basic to help programmers understand the main concepts.

Note: In the installation suite of the ODL client (in the Samples folder) there is a sample program (ETS_Test) with the source written in Visual Basic 6 found in directory “Samples\ETS_VBTest.

4.1. Examples using Visual Basic

For the VB programmer to be able to use the objects that the ETS_Broker library contains he must first include the ETS_Broker library in the “Project/Preferences” menu of the Visual Basic. The way it is done is through the following screen:



References - NewODLTest.vbp

Available References:

☑ Visual Basic For Applications
☑ Visual Basic runtime objects and procedures
☑ Visual Basic objects and procedures
☑ OLE Automation
☑ Microsoft DAO 3.51 Object Library
☑ Microsoft Data Formatting Object Library 6.0 (SP4)
☑ Microsoft Scripting Runtime
☑ ETS_Broker 2.0 Type Library
□ IAS Helper COM Component 1.0 Type Library
□ IAS RADIUS Protocol 1.0 Type Library
□ Acrobat Access 3.0 Type Library
□ AcroIEHelper 1.0 Type Library
□ Active DS Type Library
□ Active Setup Control Library

ETS_Broker 2.0 Type Library

Location: C:\Documents and Settings\e.balta\Desktop\ODLAPI\ETS_Bro

Language: Standard

The way to declare and use an object in Visual Basic is:

Dim ord **As** Order

Set ord = **New** Order

After the developer declares and creates an object of type Order,he can then access all its fields and functions.

4.2. Objects of ETS_Broker module

The basic structural component of the ODL-Client-API application is the ETS_Broker module. This is a COM module, in DLL form, which encapsulates a number of objects which manage all the messages an application (an ODL Client), located at the member's Intranet, can either send to or receive from the OASIS Server or XOrder Server, via the ATHEX Gateway system(s). The objects and their classification into **Call Interface** and **Application Requests** are the ones that have been already mentioned in paragraph 3.3 of Chapter 3.Application Development for ATHEX Gateway.

4.2.1. *Objects usage*

4.2.1.1. Object property types and Field types

The types of the fields of the various messages can be one of the following:

- Alpha
- Numeric
- Date
- Time
- DateTime

These types are described in section 4.2.1.2.

The return types of the properties (property types) of the objects that are implemented inside the ODL API are:

- Long, which is used for the message fields having type Alpha, with a total length of 1 (byte). The programmer has to convert the long type data to a character and vice versa by utilizing the chr() and asc() functions, respectively.
- String of fixed width, which is used for the fields of all other types.

For the better understanding of the objects that are provided to the programmer, the following must be kept in mind:

1. The types of the fields (Field Types), as they are presented in the description of the objects, in sections 4.2.2 and 4.2.3. More details about the types of the fields and their formatting are listed in section 4.2.1.2
2. The Default Values and the Possible Field Values, as they are presented in the description of the objects, in section 4.2.3. The concept of the Possible Field Values is described in section 4.2.1.3
3. All properties, except the Read only ones **must** be filled.

4.2.1.2. Field Types and Formatting Rules

In the various messages the following field data types are encountered: **Alpha, Numeric, Date, Time and DateTime**.

The **numeric** fields have to be aligned to the right and in order to cover the entire field width; zeroes must be padded to the left. For example, in order to set a numeric field with a total width of 7 bytes to the value 12, the field must be filled as 0000012.

In **numeric** fields that make use of decimal digits, the field width is given in the form “X.Y”. This means that the first (from left to right) X digits are the integer part of the field and the following Y digits are the decimal part. For example, for a price field this is described as numeric 5.4. In order to set this field to the value 250, the field must be filled as 002500000 (5.4 format).

The **alpha** fields have to be aligned to the left and padded with spaces. For example, in the message used for the entry of a new order, the field Member Order Number is defined as alpha having a total width of 12 bytes. If the desired value of this field is XYZ, then we fill the field (left to right) with XYZ and then continue appending nine (9) spaces.

In all cases the **date** type is used, it will follow the “YYYYMMDD” format. For example, the 9th of March, 2001 will appear as 20010309.

In cases a time is used (**time** type), it will follow the “HHMM” format, if the field width is 4, or the “HHMMSSDD” format if the field width is 8. HH denotes the hour (ranging from 0 to 23), MM denotes the minutes (ranging from 0 to 59), SS denotes the seconds (ranging from 0 to 59), and DD denotes the hundredths of a second (ranging from 0 to 99). As an example, the time 11:40 am is filled as 1140 in the first case, and as 11400000 in the second case.

In all cases the **datetime** type is used, it will follow the “YYYYMMDDhhmmssdddddd” format. YYYY denotes the year, MM denotes the month, DD denotes the day, HH denotes the hour (ranging from 0 to 23), MM denotes the minutes (ranging from 0 to 59), SS denotes the seconds (ranging from 0 to 59), and DD denotes the hundredths of a second (ranging from 0 to 99) and dddddd denotes the microseconds.

4.2.1.3. Default and Possible Field Values

For all properties / fields, for which **default values** have been specified, the properties / fields should be filled with these default values. For example, the field MessageSource in the new order entry message, has a default value of “C”, thus it must always be filled as “C”.

Depending upon the Market parameterization as well as the Market’s state at a given time, the **possible values** that can be assigned to several fields are discreet. The members’ applications must maintain a pool of data (in a database or database like structure) regarding the Market’s parameters (e.g. Markets’ names, securities’ names, etc) as well as the present condition of the Market (e.g. if a specified market is open or closed at a given time, if a security has been put in temporary suspension), in order for the values assigned to several fields to be valid. The Athens Exchange informs its Members for changes in the Market parameterization affecting the possible values of the fields.

All the objects and their properties/methods are presented in 4.2.3.

Objects of Module ETS_Broker

Module ETS_Broker			
Objects	**Properties**	**Methods**	**Events**
Call Interface			
1. Administrator	• Server	• Function IsConnected • Function Register • Sub Unregister • Sub Connect • Sub Disconnect • Sub SendCommand • Sub GetOldMessages • Sub SetMonitorLevel	• NewCCMMessage • NewDataMessage • NewError • NewUser
2. Broker	• Server	• Function IsConnected • Function CanSend • Function Register • Sub Unregister • Function SendMessage • Function GetMessageType • Function GetMessageObject • Sub GetLostMessages • Sub SendLostMessages	• NewETSMessage • NewODLMessage
Application Requests			
1. OrderEntry	See Appendix B. ODL-Client-API: ETS_Broker properties	• Function FormatMessage • Sub ParseMessage	
2. OrderEdit	-""-	-""-	
3. OrderChange	-""-	-""-	
4. TradeReportEntry	-""-	-""-	
5. HitAndTakeOrderEntry	-""-	-""-	
6. QuoteEntryChange	-""-	-""-	
7. QuoteCancel	-""-	-""-	
8. QuoteRequest	-""-	-""-	
9. ConfirmOrderEntry	-“”-	-“”-	
10. ConfirmOrderEdit	-“”-	-“”-	
11. ConfirmOrderChange	-“”-	-“”-	
12. Trade	-“”-	-“”-	
13. Reject	-“”-	-“”-	
14. QuoteStatusReport	-""-	-""-	
15. QuoteRequestConfirmation	-""-	-""-	
16. QuoteRequestExecution	-""-	-""-	
17. QuoteRequestInfo	-""-	-""-	
18. QuoteResponsibilitySuspend/Resume	-""-	-""-	
19. QuoteAlarm	-""-	-""-	
20. CreditLimitInfo	-“”-	-“”-	
21. SecurityPrices	-“”-	-“”-	
22. SecurityStatus	-“”-	-“”-	
23. MarketStatus	-“”-	-“”-	
24. SystemStatus	-“”-	-“”-	
25. HitAndTakeOrderInfo	-""-	-""-	
26. ExchangeNotes	-""-	-""-	

Module ETS_Broker			
Objects	**Properties**	**Methods**	**Events**
27. DSSEntry	-""-	-""-	
28. DSSConfirm	-""-	-""-	
29. DSSTrade	-""-	-""-	
30. DSSBroadcast	-""-	-""-	

4.2.2. *Call Interface Objects*

Below there is a description of the two objects (Administrator, Broker) contained in the module ETS_Broker. Description of properties, methods, error codes and events of each of the above objects follows.

4.2.2.1. Administrator

This object can be used in administration applications of the ATHEX Gateway. Through this interface the application has the capability of:

1. Activating and Deactivating the connection of the ATHEX Gateway with the ATHEX Server(s),

2. Switching the operating mode of the ATHEX Gateway among the three possible ones:

    a. primary with backup functionality (the ATHEX Gateway operates as primary and sends logging information to another ATHEX Gateway acting as backup)

    b. primary without backup functionality (the ATHEX Gateway operates as primary but does not send any logging information to a possible backup server). Note this is also the operation mode of each gateway in a multiple concurrent server configuration.

    c. backup (the ATHEX Gateway operates as backup server and gets logging information from another ATHEX Gateway acting as primary)

3. Maintaining the list of ODL Clients allowed to connect to the ATHEX Gateway

4. Sending a message to all the users which are connected,

5. Reading messages which traded in a previous day

6. Monitoring (through IMonitor interface) anything which happens in the ATHEX Gateway

In multiple gateway designs where a client application intends to connect to more than one gateway, separate instances of this object should be created and maintained to achieve commumnication to each gateway.

The object has the following property:

Server	String	By making use of that property, the administrative application can set up the name (or the IP address) of the machine, where the ATHEX Gateway is located. This property can be set before the call to the Register method, thus redirecting the Register operation to target at the specified machine. Information about the default IP address of the ATHEX Gateway is also placed at the client machine's registry.

(* Read Only)

The object has the following functions:

Function IsConnected [in] bsConnection As String

Returns Bool

It is called to detect if the ATHEX Gateway is connected with a specific ATHEX Server. Variable bsConnection is used to verify the connection to the OASIS, the XNet server or the DSS server using the "ETS", "ORA" or "DSS" arguments respectively. These connections are 1) Client -> ATHEX Gateway, 2) ATHEX Gateway -> Client, 3) ATHEX Gateway <-> OASIS with respect to the OASIS and 1) Client -> ATHEX Gateway, 2) ATHEX Gateway -> Client, 3) ATHEX Gateway <-> ATHEX Server. The function returns **true** when all the above connections are valid. If, however, there is some connection problem, either among the ODL Client with the ATHEX Gateway, or among the ATHEX Gateway with the ODL Client, or at some level of the MS Windows subsystem, responsible for the connection, then it will be an error code defining the cause of the problem. In this case an exception will rise which the member application should take under consideration. The value **false** is never returned. Typical error codes are:

ETS_E_NOT_FOUND (0x80042505) The ATHEX Gateway does not recognize the application which uses the IsConnected function, because, during the current date, there had been no previous successful call to the Register method.

ETS_E_NOTACTIVE (0x80042511) The application is not currently certified to the ATHEX Gateway, despite the fact it had been certified at a previous time within the current day. A new call to the Register method is required, possibly followed by a process, which will retrieve any possibly lost messages.

ETS_E_CALLBACK_ERROR (0x80042514) There had been a problem with a previous attempt of the ATHEX Gateway to send a message to the application. For this reason, the ATHEX Gateway considers the notification interface as corrupted, therefore a new call to the Register method is required, as well as a process which will retrieve the lost messages.

E_FAIL (0x80004005) The ODL-Client-API returns that the application is not certified. There has been no previous successful call to the Register method.

ETS_E_STATE (0x80042508) This return value is supplied at the very rare case, where there is a synchronization problem between the ODL-Client-API, which considers itself certified, and the ATHEX Gateway, which considers the application not certified.

ETS_E_NOCONNECTION (0x8004250a) ATHEX Gateway is either not connected or disconnected from the ATHEX Server.

Function Register	[in] bsUserName as BSTR
Returns SHORT (user id)	[in] bsPassword as BSTR
	[in] bsDomain as BSTR

It is called to certify the presence of this application to an ATHEX Gateway. Activates the connection enabling the sending/receiving of messages. The way this is accomplished is described in paragraph 3.5 Registration Procedure. The application that is connected with the ATHEX Gateway is denoted by three parameters (username, password and Domain of the ODL Client), which are passed as arguments during the calling phase. This user must belong in the ETS_Admins user group. The Register method returns the application's certification code in the ATHEX Gateway (this code denotes the machine where the application is running). That code is unique during the trading day, regardless the number of times the method is called. The bsUserName, bsPassword and bsDomain variables contain the necessary information, as described above, in order for the ATHEX Gateway to successfully perform the certification. Typical error codes are:

ETS_E_NOTINGROUP (0x80041513) Security problem. The user does not belong in the group of the Administrators of the ATHEX Gateway.

E_INVALIDARG (0x80070057) Problem in the value passed as an argument in the function.

E_OUTOFMEMORY (0x8007000E) Memory allocation problem. In case there is not enough memory in the system where the ODL Client application is running.

Sub Unregister

Unregister connection with an ATHEX Gateway. Typical error codes are:

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration).

Sub Connect	[in] bsConnection As String,
	[in] ldTimeout As LONG,
	[out] pRejectedText As String

Gives the command to an ATHEX Gateway to connect to a specific ATHEX Server. Variable bsConnection is used to refer the action to the OASIS, the XNet server or the DSS sever usingthe "ETS", "ORA" or "DSS" arguments respectively. Variable ldTimeout

keeps the total time that ATHEX Gateway has for connecting to the OASIS in msec. In case of wrong variable the pRejectedText will contain an explanation message of the false. Typical error codes are: E_FAIL (0x80004005) ODL-Client-API returns that the application is not Registered, thus no successful call of the Register function has occurred. ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration). ETS_E_LOGONREJECTED (0x80042509) ATHEX Gateway could not contact Comm Server because of invalid password. ETS_E_MAXRETRIES (0x8004250b) ATHEX Gateway exhausted the pre-defined number of maximum attempts to connect to Comm Server successfully. NTE_BAD_DATA (0x80090005) Invalid connection parameter in Register of ATHEX Gateway.	
Sub Disconnect	[in] bsConnection As String, [in] fForce As Boolean
Gives the command to an ATHEX Gateway to disconnect from the OASIS, the XNet server or the DSS server. Variable bsConnection is used to refer the action to the OASIS, the XNet server or the DSS server using the “ETS”, “ORA” or “DSS” arguments respectively. Variable fForce forces the disconnection even if there are messages to be send or messages not yet confirmed. Typical error codes are: E_FAIL (0x80004005) Appears in two cases: a) The ODL-Client-API returns that the application is not Registered (no previous successful call of Register function occurred), b) in case there is no connection between ATHEX Gateway and the ATHEX Server. ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration).	
Sub SendCommand	[in] bsMessage As String [out] result as Variant
Sends an internal message from Administration Application to an ATHEX Gateway. This message is not send to OASIS it just remains in the member firm network. The ATHEX Gateway parses this message and runs an action. The result of this action is returned in the result output parameter. It is a communication protocol open in future expansions. In current version the following commands are implemented:	

1. The Broadcast command

   Format: BR<broadcast message>.

   Description: This kind of message sends an ODL message to all the registered clients on the specific ATHEX Gateway. The message to be sent must have maximum size of 512 bytes. Variable bsMessage contains the message body, variable result is not used for Broadcast messages.

2. The Get Operating Mode command

   Format: GM<fRegistry>

   The field fRegistry is a flag that indicates whether the command will return the mode under which the ATHEX Gateway currently operates or the operating mode stored in ATHEX Gateway's registry. It takes the following values:

   0=the command shall return current operating mode, 1=the command shall return the operating mode stored in ATHEX Gateway's registry

   Description: Asks the Gateway of the operating mode that it has been set to (current operating mode or operating mode stored in registry). The parameter result will contain a string of one character indicating this mode. The possible values of this character are: 0=PrimaryStandalone (also for multiple gateway setups), 1=PrimaryWithBackup, 2=BackupServer.

3. The Switch Operating Mode command

   Format: SM<OldMode><NewMode><RestartService>

   The fields OldMode and NewMode indicate the old mode stored in the ATHEX Gateway's registry and the new one respectively. They take the following values:

   0=PrimaryStandalone (also for multiple gateway setups),

   1=PrimaryWithBackup,

   2=BackupServer

   The field RestartService takes the following values: 0=NoRestart, 1=Restart

   Description: Changes the operating mode of the ATHEX Gateway and restarts it if the RestartService flag is set. The parameter result will contain a string of one character indicating the new mode stored in ATHEX Gateway's registry.

4. The Get Valid ODL Clients command

   Format: GU

Description: Asks the Gateway of the ODL clients that are allowed to be connected to the ATHEX Gateway. The parameter result is a string containing all valid ODL clients in the following format:

<comp_name 1>:<UserID 1>;<comp_name 2>:<UserID 2>;….;<comp_name n>:<UserID n>;

where <comp_name i> is the name of the computer and <UserID i> is a printable character unique for each ODL client

5. The Set Valid ODL Clients command

Format: SU<ValidODLClients>

The field <ValidODLClients> contains the new list of all valid ODL clients in the following format: <comp_name 1>:<UserID 1>;<comp_name 2>:<UserID 2>;….;<comp_name n>:<UserID n>;

where <comp_name i> is the name of the computer and <UserID i> is a printable character unique for each ODL client and not equal to '0' (zero), ' ' (space character), ':' or ';'

Description: Changes the ODL clients that are allowed to be connected to the ATHEX Gateway. The parameter result is a string containing the new list of valid ODL clients in the format described above

Typical error codes are:

E_FAIL (0x80004005) ODL-Client-API returns that the application is not Registered, thus no successful call of the Register function has occurred.

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration).

E_POINTER (0x80004003) Invalid parameter. The bsMessage can not be used because is nil.

E_INVALIDARG (0x80070057) Size of internal message exceeds 512 bytes.

Sub GetOldMessages	[in] bsDate As String, [in] userId As Integer [out] msgIds As Variant [out] connections As Variant [out] msgs As Variant [out] dates As Variant

[out] users As Variant

This function allows the Administration Application to read messages from specific date, which have been written to hard disk. In the field bsDate user must enter the date for which he is interested to load the messages. Date is of the form ‘yyyymmdd’ e.g. 19991003, which means 3/10/1999. userId keeps the user identification for which the look up is intended. If this ID is greater than zero then only the messages for the specific user will be returned. If it is (-1) then the messages of all the users will be returned. Variables msgIds, connections, msgs, dates, users hold the ID, the originating connection of the messages, the messages, the dates and the User Id. Please note that the msgIds are unique per connection.

Typical error codes are:

E_FAIL (0x80004005) ODL-Client-API returns that the application is not Registered, thus no successful call of the Register function has occurred.

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration).

E_POINTER (0x80004003) Invalid parameter. Some parameter cannot be used because it is nil.

ETS_E_INVALIDARG (0x8004250c) Invalid Date. Either wrong format or no data found for the specific date given. Check is done on the ATHEX Gateway

Sub SetMonitorLevel [in] bsConnection As String,

[in] dLevel As LONG

Defines the level for which the Administrator wants to accept messages. Variable bsConnection is used to refer the action to the OASIS, the XNet server or the DSS server using the “ETS”, “ORA” or “DSS” arguments respectively. Possible values of variable dLevel are 0 -> no messages accepted, 1 -> accept OASIS messages, 2 -> accept all messages.

Typical error codes are:

E_FAIL (0x80004005) ODL-Client-API returns that the application is not Registered, thus no successful call of the Register function has occurred.

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration).

The object has the following Events:

<table>
<tr><td>NewCCMMessage</td><td>[in] bsConnection As String,
[in] bsCCM As String,
[in] fSend As Boolean</td></tr>
<tr><td colspan="2">Activated when a new Control message has been sent from an ATHEX Server. Control messages contain information about connectivity between the ATHEX Gateway and OASIS system. Variable bsConnection informs the client application of the origin of the message (“ETS”: OASIS, “ORA”: XNet server or “DSS”: DSS server). Variable bsCCM holds the control message while variable fSend shows if the message was send from ATHEX Gateway (TRUE) or from ATHEX Server (FALSE).</td></tr>
<tr><td>NewDataMessage</td><td>[in] bsConnection As String,
[in] bsDATA As String,
[in] swMsgId As LONG
[in] fSend As Boolean
[in] dat As Double</td></tr>
<tr><td colspan="2">Activated when a new data message has been sent from an ATHEX Server to ATHEX Gateway. These messages come when a client connects to the ATHEX Gateway, or when a reply to client request comes. Variable bsConnection informs the client application of the origin of the message (“ETS”: OASIS, “ORA”: XOrder server or “DSS”: DSS server). Variable bsDATA holds the content of the message, variable swMsgId holds the message identification (unique per connection), variable fSend shows if the message was send from ATHEX Gateway to ATHEX Server (TRUE) or from ATHEX Server to ATHEX Gateway otherwise. Variable dat holds the arrival time of the message on the ATHEX Gateway. Note that dat is of type double but represents an OLE Automation date. It is the client’s responsibility to convert it to a more convenient date type.</td></tr>
<tr><td>NewError</td><td>[in] bsError As String,
[in] hError As LONG</td></tr>
<tr><td colspan="2">Activated when an error occurs in the ATHEX Gateway. Variable bsError contains description of the message and variable hError contains the error code.</td></tr>
<tr><td>NewUser</td><td>[in] bsMachine As String,
[in] fReg As Boolean</td></tr>
<tr><td colspan="2">Activated when a user registers/unregisters to/from the ATHEX Gateway. Variable bsMachine keeps the name of the machine that register/unregister and variable fReg shows if client registers (TRUE) or unregisters (FALSE).</td></tr>
</table>

4.2.2.2. Visual Basic example

For the developer to be able to use the object Administrator he has to use first the command:

```
Dim WithEvents adm As Administrator
```

The developer has to complete the code of the functions that follow.

```
Private Sub adm_NewCCMMessage(ByVal bsConnection As String, ByVal bsCCM As
String, ByVal fSend As Boolean)
…
End Sub
```

```
Private Sub adm_NewDataMessage(ByVal bsConnection As String, ByVal bsDATA As
String, ByVal swMsgId As Long, ByVal fSend As Boolean, ByVal dat As double)
…
End Sub

Private Sub adm_NewError(ByVal bsError As String, ByVal hError As Long)
…
End Sub

Private Sub adm_NewUser(ByVal bsMachine As String, ByVal fReg As Boolean)
…
End Sub
```

Attention: The developer has to make sure that the functions mentioned above will consume the least time possible. Complicated calculations must be avoided inside these functions. The reason for this is because every time the function is called ATHEX Gateway waits (blocks) for its completion to continue its work.

To read the message of a previous day:

```
 On Error GoTo err_lbl

 Dim omsgs As Variant
 Dim ocons As Variant
 Dim omids As Variant
 Dim omdts As Variant
 Dim omusr As Variant

   adm.GetOldMessages txtDate.text, 0, omids, ocons, omsgs, omdts, omusr

 Dim con As String
 Dim msg As String
 Dim mid As Long
 Dim dat As Date
 Dim uid As Integer

 For i = LBound(omids, 1) to UBound(omids, 1)
```

```
   msg = omsgs(i)
   con = ocons(i)
   mid = omids(i)
   dat = omdts(i)
   uid = omusr(i)

   ...............
  Next i

  Exit Sub
err_lbl:
   MsgBox Err.Description
```

There is another example of the ETS_Broker.Administrator with Visual Basic in appendix F.

4.2.2.3. Broker

The object has the following property:

NAME	TYPE	REMARKS
Server	String	With this property the member application can define the name or IP address of the ATHEX Gateway. It can be set before the call of the function Register, thus it is considered during the connection with the ATHEX Gateway. Information about the IP address of the ATHEX Gateway is also found on the clients registry. For more on this refer to the ATHEX Gateway installation manual.

(* Read Only)

The object contains the following functions:

Function IsConnected [in] bsConnection As String

Returns Boolean

It is called to detect if the ATHEX Gateway is connected with a specific ATHEX Server. bsConnection is used to verify the connection to the OASIS, the XOrder server or the DSS server using the “ETS”, “ORA” or “DSS” arguments respectively. These connections are 1) Client -> ATHEX Gateway, 2) ATHEX Gateway -> Client, 3) ATHEX Gateway <-> ATHEX Server. The function returns **true** if all the above connections are active. In other case there will be an error code defining the cause of the problem. In this case an exception will rise which the member application should take under consideration. The value **false** is never returned. Variable bsConnection is used to direct the action to the OASIS, the XOrder server or the DSS server by use of the “ETS”, “ORA” or “DSS” arguments respectively.

Typical error codes are:

ETS_E_NOT FOUND (0x80042505) Gateway does not recognize the apllication which checks the value of the property because it is not registered on the current day

ETS_E_NOTACTIVE (0x80042511) The application currently is not registered on the ATHEX Gateway, even though it was registered in some previous moment on the same date. Registration and possible recovery of data is necessary to obtain lost messages

ETS_E_CALLBACK_ERROR (0x80042514) Gateway couldn’t send a message to the application due to a problem. Thus it considers the notification interface unavailable. Registration and data recovery is needed for the lost messages

E_FAIL (0x80004005) ODL-Client-API returns that the application is not registered, thus no successful call of the Register function has occurred.

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration).

ETS_E_NOCONNECTION (0x8004250a) ATHEX Gateway is either not connected or disconnected from the ATHEX Server.

Property CanSend	[in] bsConnection As String
Returns Boolean	

This function returns **true** when a user can send a message. **No error code returned**. No call to the ATHEX Gateway is done. The ODL-Client-API checks if registration of the member application has occured, if there is a connection between ATHEX Gateway and the specified ATHEX Server, if there is no problem between ATHEX Gateway and the specified ATHEX Server and finally if there are no pending messages at that time. In case all the above conditions are true then CanSend returns **true**, otherwise it returns **false**.Variable bsConnection is used to direct the action to the OaSIS, the XNet server or the DSS server using the “ETS”, “ORA” or “DSS” arguments respectively.

Function Register	[in] bsUserName As String
Returns Integer (user id)	[in] bsPassword As String
	[in] bsDomain as BSTR

This function is called to register the Member Application to the ATHEX Gateway. It activates the connection for the application to send messages. The user that will be connected to the ATHEX Gateway will be recognized be three parameters as shown above. The user must belong to the user group of ETS_Brokers. This group is declared in the ATHEX Gateway and is used for authentication and validation of the users. The function returns the User ID, which is unique through out the daily trading and is the same no matter how many times the function is called. Variables bsUserName, bsPassword and bsDomain have the necessary information required from the ATHEX Gateway to verify the user.

Typical error codes are:

ETS_E_PENDING_MESSAGES (0x80042515) Appears when there are messages sent for which a reply is expected.

ETS_E_SECURITY (0x80042517) Security problem. The call in ATHEX Gateway is not encrypted and authenticated.

ETS_E_NOTINGROUP (0x80041513) Security problem. The user does not belong to the broker group of the ATHEX Gateway.

E_INVALIDARG (0x80070057) Problem on the value of a parameter (e.g. bsPassword is empty). General error code used from all DCOM subsystems as well as ODL subsystems.

E_OUTOFMEMORY (0x8007000E) In the rare case where there is no available memory on the system that the ODL-Client-API application is running.

Sub Unregister	[in] fForce As Boolean

Deactivates the connection. If the user has send one or more messages for which he is waiting for a confirmation then the deactivation will fail if the value of the variable fForce is false. If the variable fForce is set to TRUE, then the deactivation will occur no matter its current status.

Typical error codes are:

ETS_E_PENDING_MESSAGES (0x80042515) Appears when there are messages sent for which a reply is expected. The function cannot be called because fForce is set to **false**.

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration).

Function SendMessage Returns Long (Member Sequence Number)	[in] bsConnection As String, [in] bsMessage As String [out] pdwMsgld As Long [out] pDate As Date

Sends a message to the ATHEX Gateway directed to a specific connection.

Variable bsConnection is used to direct the action to the oASIS,the XNet server or the DSS server using the “ETS”, “ORA” or “DSS” arguments respectively. Variable bsMessage holds the content of the message to be send. This is an asynchronous call. The function returns immediatelly if the application reaches the maximum number of non-confirmed messages, otherwise it activates the sending of the message and returns. A unique (per connection) verification number is returned from ATHEX Gateway (pdwMsgld), the exact time it reached the server (pDate) and the value MemberSequenceNumber which is created from the ATHEX Gateway and is the returning value of the function. To use this function the application must first be registerd to the appropriate ATHEX Gateway.

Typical error codes are:

ETS_E_PENDING_MESSAGES (0x80042515) Appears when there are messages sent for which a reply is expected. The function cannot be called before the arrival of the expected messages.

E_FAIL (0x80004005) ODL-Client-API returns that the application is not registered, thus no successful call of the Register function has occurred.

E_POINTER (0x80004003) Invalid parameter. Some parameter cannot be used because it is null.

ETS_E_INVALIDVALUE (0x80042518) Message type is invalid (the two first characters) or it is a message which is not supposed to be send.

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration).

ETS_E_NOT FOUND (0x80042505) ATHEX Gateway does not recognize the apllication which checks the value of the property because it is not registered on the current day.

ETS_E_NOTACTIVE (0x80042511) The application currently is not registered on the ATHEX Gateway, even though it was registered in some previous moment on the same date. Registration and possible recovery of data is necessary to obtain lost messages.

ETS_E_CALLBACK_ERROR (0x80042514) ATHEX Gateway couldn’t send a message to the application due to a problem. Thus it considers the notification interface unavailable. Registration and data recovery is needed for the lost messages.

ETS_E_NOCONNECTION (0x8004250a) ATHEX Gateway is either not connected or disconnected from the Comm Server.

ETS_E_MAX_REJECTED_MSGS (0x80042518) No more messages can be sent from the application due to the fact that the maximum number of Reject messages (TR) has been reached. The only action that the member can do is to contact the appropriate ATHEX department.

ETS_E_MAX_RATE_REACHED (0x8004251B) No more messages can be sent from the application due to the fact that the maximum message tranfer rate has been reached. The only action that the member can do is to try again later.

Function GetMessageType [in] bsMessage As String
Returns Long

The function returns the message type, thus the object that can be constructed with this message. The possible values are :

0 = Unknown Type,

1 = Order Entry,

2 = Order Edit,

3 = Order Change,

4 = Trade Report Entry(Preagreed Price Trade),

6 = Hit & Take Order Entry,

7 = Reserved,

8 = Order Entry Confirmation,

9 = Order Edit Confirmation,

10 = Order Change Confirmation,

11 = New Trade Confirmation ,

12 = Reserved,

13 = Reserved,

14 = Rejection,

16 = Credit Limit Information,

17 = Security Status,

18 = Security Price,

19 = Market Status,

20 = System Status,

21 = Hit & Take Order Information

22 = Bond Information

23 = Security Info

24 = DSS Entry,

25 = DSS Entry Confirmation,

26 = DSS Trade,

27 = DSS Broadcast

28 = Quote Entry / Change

29 = Quote Cancel

30 = Quote Status Report

31 = Quote Request

32 = Quote Request Confirmation

33 = Quote Request Execution

34 = Quote Request Info

35 = Quote Responsibility Suspend / Resume

36 = Quote Alarm

37 = Exchange Notes

38 = Combination Info

Typical error codes are:

E_POINTER (0x80004003) Invalid parameter. Some parameter can not be used because it is null.

E_OUTOFMEMORY (0x8007000E) In the rare case where there is no available memory on the system that the ODL-Client-API application is running.

E_INVALIDARG (0x80070057) Message is unrecognizable (unknown type of message) or invalid message length.

Function GetMessageObject [in] bsMessage As String
Returns Unknown
[out] pdMessageType As Long

The function returns except from the message type (pdMessageType), an object which coresponds to this message. These objects are introduced in next sessions.

Typical error codes are:

E_POINTER (0x80004003) Invalid parameter. Some parameter can not be used because it is null.

E_OUTOFMEMORY (0x8007000E) In the rare case where there is no available memory on the system that the ODL-Client-API application is running.

E_INVALIDARG (0x80070057) Message is unrecognizable (unknown type of message) or invalid message length.

Sub GetLostMessages [in] bsConnection As String,

[in] ldStartMsgId As Long

[out] pvMsgIds As Variant

[out] pvMsgs As Variant

[out] pvDates As Variant

In case that ODL client exits abnormally or has a communication problem, ODL client can ask for the lost messages during the disconnection from the ATHEX Gateway. Variable bsConnection is used to direct the action to the OaSIS, the XNet server or the DSS server using the “ETS”, “ORA” or “DSS” arguments respectively. Variable ldStartMsgId should hold the ID of the last message that the client received from the ATHEX Gateway. If it is zero then all the messages of the application will be returned. The information returned from the function is splitted to the three variables of VARIANT type. These three variables are arrays with size the number of lost messages. Variable pvMsgIds keeps the verification ID (unique per connection) of the lost messages, variable pvMsgs keeps the whole message and variable pvDates keeps the exact time the message reached ATHEX Gateway.

Typical error codes are:

E_FAIL (0x80004005) ODL-Client-API returns that the application is not registered. That means no successful call of Register has been done.

E_POINTER (0x80004003) Some of the input parameters is empty (NULL).

E_INVALIDARG (0x80070057) ATHEX Gateway didn’t find recorded message with serial number = ldStartMsgId.

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration). The same code returns from ATHEX Gateway when ldStartMsgId has invalid value.

Sub SendLostMessages [in] bsConnection As String,

[in] ldStartMsgId As Long

The functionality of this function is similar to the GetLostMessages with the difference that the messages are send asynchronous from ATHEX Gateway. ODL client sends the last verification ID (ldStartMsgId) which has been received from specified connection and ATHEX Gateway will send all the messages that he might have lost through the notification interface that the client has implement. Variable bsConnection is used to direct the action to the OaSIS, the XOrder server or the DSS server using the “ETS”, “ORA” or “DSS” arguments respectively.

Typical error codes are:

E_FAIL (0x80004005) ODL-Client-API returns that the application is not registered. That means no successful call of Register has been done.

E_POINTER (0x80004003) Some of the input parameters is empty (NULL).

E_ INVALIDARG (0x80070057) ATHEX Gateway didn’t find recorded message with serial number = ldStartMsgId.

ETS_E_STATE (0x80042508) In the rare case where ATHEX Gateway and ODL Client have consistency problems (Client thinks registered, Gateway has no client registration). The same code returns from ATHEX Gateway when ldStartMsgId has invalid value.

E_OUTOFMEMORY (0x8007000E) In the rare case where there is no available memory on the system that the ODL-Client-API application is running.

The object requires a notification interface from the ODL client. The events of this interface are the following:

NewETSMessage	[in] bsConnection As String, [in] ldMsgId As Long [in] bsMessage As String [in] dat As double
Activated when there is a new message from the OaSIS, the XOrder server or the DSS server for the specific client. These messages are either public ones (like market status, security status) or messages for a specific client (confirmations, rejections). The information sent for every message is placed on the three variables. Variable bsConnection is used to direct the action to the OaSIS, the XOrder server or the DSS server using the “ETS”, “ORA” or “DSS” arguments respectively. Variable ldMsgId keeps the verification ID (unique per connection) of the message, the actual message is kept in bsMessage and the exact time that the message reached the ATHEX Gateway is kept in dat. Note that dat is of type double but represents an OLE Automation date. It is the client’s responsibility to convert it to a more convenient date type. Note that in a multiple concurrently active Gateway scheme, an application that is connected to more than one gateways simultaneosuly should expect this callback to be executed for the same message as many times as the number of gateways the application is connected to.	
NewODLMessage	[in] bsMessage As String
ODL messages are messages exchanged in the internal network of the Member firm and are not send to ATHEX. These messages can be sent from the object Administrator.	

4.2.2.4. Visual Basic example

If the developer wants to use the object of type Broker, he has, first, to use the command:

```
Dim brk As Broker
```

In multiple gateway designs where a client application intends to connect to more than one gateway, separate instances of this object should be created and maintained to achieve commumnication to each gateway. i.e.:

```
Dim brk1 As Broker

Dim brk2 As Broker
```

Due to the fact that a notification interface is required from the client application to “listen” for events the developer has to use the next declaration, which is supported in the Visual Basic:

```
Dim WithEvents brk As Broker
```

This way Visual Basic automates the procedure and constructs the framework of the functions making it easier for the developer to complete his code.

```
Private Sub brk_NewETSMessage(ByVal bsConn As String, ByVal ldMsgId As Long, ByVal bsMessage As String, ByVal dDate As double)

End Sub

Private Sub brk_NewODLMessage(ByVal bsMessage As String)

End Sub
```

An important issue is that a slow ODL client will face communication disconnections from the ATHEX Gateway. In this case the client has to register again to the ATHEX Gateway and ask for the lost messages (GetLostMessages/SendLostMessages).

For the client to be able to send/receive messages he must first register to the ATHEX Gateway through the Register function. A typical procedure for this might be a login window where the user will enter his specific information.

Creating an object from a string can be done as follows:

```
Dim obj As Variant

Dim ord As Order

Set obj = brk.GetMessageObject(txtFormat.text, typ)

If typ = 1 Then
  Set ord = obj
```

End If

4.2.3. *Application Requests objects*

All the properties of the objects that correspond to messages are described on Appendix B. ODL-Client-API: ETS_Broker properties. For string type properties, the programmer must carefully check the length of the string. The Application Request messages correspond to messages sent and received from the members' application to OASIS, XNet server or DSS Server In the tables describing the properties of the objects there is also information about the form of the corresponding message.

The Application Request Objects provided through ODL API are divided into the following general categories:

1. Single General Order Handling
2. Quotation Negotiation Handling
3. Hit And Take Orders Hadling
4. Trading Report (Pre-agreed Trades) Handling
5. Reference & Market Data Information
6. Clearing Data Handling

4.2.3.1. Single General Order Handling objects

Single general order handling objects are divided in the following two categories:

- Objects that correspond to messages send to OASIS or XNet server through ATHEX Gateway from Applications using ATHEX Gateway. These objects are:
  - ➢ OrderEntry
  - ➢ OrderEdit
  - ➢ OrderChange
- Objects that correspond to messages sent from the target exchange through ATHEX Gateway to Applications using ATHEX Gateway. These objects are:
  - ➢ ConfirmOrderEntry
  - ➢ ConfirmOrderEdit
  - ➢ ConfirmOrderChange

- Trade Info

It must be noted that the member shall use only the ConfirmOrderEntry, ConfirmOrderEdit, ConfirmOrderChange and Trade messages sent from the target exchange in order to be sure that these order entries, alterations and/or cancellations have been accepted by ATHEX (or other venue as appropriate) and that a trade has possibly taken place.

4.2.3.1.1. OrderEntry (MB)

This object corresponds to the message New Order (MB). In the table that follows, all the OrderEntry object properties are described as well as the corresponding fields of the New Order message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	"MB" (Read Only, Filled by ODL)
Reserved			Alpha	4	" " (Filled by ODL), reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	'C' = CTCI (Default, Filled by ODL)
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	"000000" (Read Only, Filled by ODL G/W)
VenueID	String	4	Alpha	4	
BoardID	Long		Alpha	1	
OrderType	Long		Alpha	1	
Side	Long		Alpha	1	
CSDAccountID	String	12	Alpha	12	
GOIFlag	String	1	Alpha	1	Not used.
ShortSellFlag	String	1	Alpha	1	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
Price	String	9	Numeric	5.4	
Volume	String	12	Numeric	12	
DisclosedVolume	String	12	Numeric	12	
AutoDisclosedVolume	String	12	Numeric	12	“000000000000” (Default) For Future Use
ConditionVolume	String	12	Numeric	12	“000000000000” (Default) For Future Use (for condition MF and MOF)
OrderLifetime	Long		Alpha	1	
SpecialConditions	Long		Alpha	1	
OriginalPriceType	Long		Alpha	1	
StopSecurityID	String	15	Alpha	15	
StopPrice	String	9	Numeric	5.4	
ExpirationDate	String	8	Date	8	
ClientOrderID	String	16	Alpha	16	
OrderNote	String	25	Alpha	25	
ListID	String	6	Alpha	6	
ClearingMemberID	String	4	Alpha	4	
PositionEffect	String	1	Alpha	1	
SettlType	String	1	Alpha	1	
DirectElectronicAccess	String	1	Alpha	1	
ClientID	String	10	Numeric	10	
ClientIDQualifier	String	1	Alpha	1	
InvestmentDecisionID	String	10	Numeric	10	
InvestmentDecisionIDQualifier	String	1	Alpha	1	
ExecutionWithinFirmID	String	10	Numeric	10	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
ExecutionWithinFirmIDQualifier	String	1	Alpha	1	
NonExecutingBrokerID	String	10	Numeric	10	
TradingCapacity	String	1	Alpha	1	
LiquidityProvision	String	1	Alpha	1	
SpecialInstructions	String	120	Alpha	120	Only used for XNET. This field is omitted altogether for messages directed to the ETS interface.

This object is used by a member firm to enter a new order. The target exchange accepts the order if the data is valid. (i.e., the data is readable and the values are allowable) and a Confirm of a New Order Message (TB) (paragraph 4.2.3.1.4) is sent to the member firm. Otherwise, a Reject Message (TR) (paragraph 4.2.3.1.8) is sent to the member firm with a code specifying the reason why the order was rejected (RejectReasonCode).

A unique order number is then assigned to the order by OASIS Server in the specified date.

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object. Objects with a null value in the “SpecialInstructions” field will result in messages that are 120-characters shorter (i.e. the SpecialInstructions field is omitted). Therefore, objects representing messages directed to the ETS interface should always contain null value in the “SpecialInstructions” field.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

The above two methods are unique methods of all the objects corresponding to messages.

4.2.3.1.2. OrderEdit (Cancel /Activate/Deactivate) (MC)

This object corresponds to the message Order Edit (MC). In the table that follows all the OrderEdit object properties are described as well as the corresponding fields of the Order Edit message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	"MC" (Read Only, Filled by ODL)
Reserved		4	Alpha	4	" " (Filled by ODL), reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	'C' = CTCI (Default, Filled by ODL)
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	"000000" (Read Only, Filled by ODL G/W)
VenueID	String	4	Alpha	4	
BoardID	Long		Alpha	1	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
OrderNumber	String	8	Numeric	8	
OrderDate	String	8	Date	8	
OrigClientOrderID	String	16	Alpha	16	
ClientOrderID	String	16	Alpha	16	
EditType	Long		Alpha	1	
EditedDisclosedVolume	String	12	Numeric	12	
OrderNote	String	25	Alpha	25	
ListID	String	6	Alpha	6	

This object is used to cancel/activate/deactivate an order identified by the Order number and Order entry date. A Confirm of Order Edit Message (TC) is returned to the member firm by the target exchange if the order has been edited (canceled/deactivated/activated). Otherwise, a Reject Message (TR) is sent to the member firm with a reason code explaining why the request was not accepted.

The object has the following methods:

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function: E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.1.3. OrderChange (MD)

This object corresponds to the message Order Change (MD). In the table that follows, all the OrderChange object properties are described as well as the corresponding fields of the Order Change message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“MD” (Read Only, Filled by ODL)
Reserved	String	4	Alpha	4	“ “ (Filled by ODL), reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘C’ = CTCI (Default, Filled by ODL)
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000” (Read Only, Filled by ODL G/W)
VenueID	String	4	Alpha	4	
BoardID	Long		Alpha	1	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
OrderNumber	String	8	Numeric	8	
OrderDate	String	8	Date	8	
ChangedPrice	String	9	Numeric	5.4	
ChangedVolume	String	12	Numeric	12	
ChangedDisclosedVolume	String	12	Numeric	12	
ChangedAutoDisclosedVolume	String	12	Numeric	12	“000000000000” (Default) For Future Use
ChangedCSDAccountID	String	12	Alpha	12	
ChangedGOIFlag	String	1	Alpha	1	Not used.
ChangedShortSellFlag	String	1	Alpha	1	
ChangedOriginalPriceType	Long		Alpha	1	
ChangedLife	Long		Alpha	1	
ChangedExpirationDate	String	8	Date	8	
OrigClientOrderID	String	16	Aplha	16	
ClientOrderID	String	16	Alpha	16	
ChangedOrderNote	String	25	Alpha	25	
ListID	String	6	Alpha	6	
ChangedClearingMemberID	String	4	Alpha	4	
ChangedPositionEffect	String	1	Alpha	1	
ChangedSettlType	String	1	Alpha	1	
ChangedDirectElectronicAccess	String	1	Alpha	1	
ChangedClientID	String	10	Numeric	10	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
ChangedClientIDQualifier	String	1	Alpha	1	
ChangedInvestmentDecisionID	String	10	Numeric	10	
ChangedInvestmentDecisionIDQualifier	String	1	Alpha	1	
ChangedExecutionWithinFirmID	String	10	Numeric	10	
ChangedExecutionWithinFirmID Qualifier	String	1	Alpha	1	
ChangedNonExecutingBrokerID	String	10	Numeric	10	
ChangedSpecialInstructions	String	120	Alpha	120	Only used for XNET. This field is omitted altogether for messages directed to the ETS interface.

This object is used to **change an order**, identified by the **Order number and Order entry date**. A Confirm of Order Change Message (TD) is returned to the member firm by the target exchange if the order has been changed. Otherwise, a Reject Message (TR) is sent to the member firm with a reason code explaining why the request was not accepted.

The object has the following methods :

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Objects with a null value in the “ChangedSpecialInstructions” field will result in messages that are 120-characters shorter (i.e. the SpecialInstructions filed is omitted). Therefore, objects representing messages directed to the ETS interface should always contain null value in the “ChangedSpecialInstructions” field.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.1.4. ConfirmOrderEntry (TB)

This object corresponds to the message Confirm of a New Order (TB). In the table that follows, all the object properties are described as well as the corresponding fields of the Confirm of a New Order message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	"TB" (Read Only)
Reserved	String	4	Alpha	4	" " Reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	' ' = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	(Read Only)
VenueID	String	4	Alpha	4	
OrderType	Long		Alpha	1	
BoardID	Long		Alpha	1	
Side	Long		Alpha	1	
CSDAccountID	String	12	Alpha	12	
GOIFlag	String	1	Alpha	1	Not used.
ShortSellFlag	String	1	Alpha	1	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
Price	String	9	Numeric	5.4	
Volume	String	12	Numeric	12	
DisclosedVolume	String	12	Numeric	12	
AutoDisclosedVolume	String	12	Numeric	12	"000000000000" (Default) For Future Use
LeavesQuantity	String	12	Numeric	12	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
AveragePrice	String	9	Numeric	5.4	
ConditionVolume	String	12	Numeric	12	“000000000000” (Default) For future Use, for condition MF and MOF
OrderLifetime	Long		Alpha	1	
SpecialConditions	Long		Alpha	1	
OriginalPriceType	Long		Alpha	1	
StopSecurityID	String	15	Alpha	15	
StopPrice	String	9	Numeric	5.4	
ExpirationDate	String	8	Date	8	
ClientOrderID	String	16	Alpha	16	
OrderNote	String	25	Alpha	25	
ClearingMemberID	String	4	Alpha	4	
PositionEffect	String	1	Alpha	1	
SettlType	String	1	Alpha	1	
OrderSource	Long		Alpha	1	
OrderNumber	String	8	Numeric	8	
OrderDate	String	8	Date	8	
OrderStatus	String	2	Alpha	2	
CurrentCreditValue	String	14	Numeric	12.2	
ListID	String	6	Alpha	6	
DirectElectronicAccess	String	1	Alpha	1	
ClientID	String	10	Numeric	10	
ClientIDQualifier	String	1	Alpha	1	
InvestmentDecisionID	String	10	Numeric	10	
InvestmentDecisionIDQualifier	String	1	Alpha	1	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
ExecutionWithinFirmID	String	10	Numeric	10	
ExecutionWithinFirmIDQualifier	String	1	Alpha	1	
NonExecutingBrokerID	String	10	Numeric	10	
TradingCapacity	String	1	Alpha	1	
LiquidityProvision	String	1	Alpha	1	
Timestamp	String	20	DateTime	20	Order entry/reentry confirm date and time
SpecialInstructions	String	120	Alpha	120	Only used for XNET. This field does not exist in messages originating from the ETS interface.

This object is used to confirm that the new order or a hit & take order has been accepted by the target exchange. It is also sent to the member firm when the order comes from the member's OASIS-ORAMA or from ATHEX via EMRW.

The message returns to the member firm the unique order number that was given to the order in "OrderNumber" field and the trading date in the "OrderDate" field. The "OrderSource" is decided according to the origin of the New Order Message. The "MemberSequenceNumber" contains the original number of the New Order message that was received from the member firm. If the order was entered by OASIS-ORAMA, this field is zero.

The object has the following methods:

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields.

Messages that do not contain a "SpecialInstructions" field will result in objects with null value in this field.

Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.1.5. ConfirmOrderEdit (TC)

This object corresponds to the message Confirm of Order Edit (TC). In the table that follows, all the object properties are described as well as the corresponding fields of the Confirm of Order Edit message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	"TC" (Read Only)
Reserved			Alpha	4	" " Reserved for ETS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	' ' = ETS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	(Read Only)
VenueID	String	4	Alpha	4	
BoardID	Long		Alpha	1	
OrigClientOrderID	String	16	Alpha	16	
ClientOrderID	String	16	Alpha	16	
CSDAccountID	String	12	Alpha	12	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
OrderNumber	String	8	Numeric	8	
OrderDate	String	8	Date	8	
ExpirationDate	String	8	Date	8	
LeavesQuantity	String	12	Numeric	12	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
AveragePrice	String	9	Numeric	5.4	
OrderStatus	String	2	Alpha	2	
CancelSource	Long		Alpha	1	
CancelReasonCode	Long		Alpha	1	
EditType	Long		Alpha	1	The EditType of the original MC (Order Edit) message.
CurrentCreditValue	String	14	Numeric	12.2	
EditedDisclosedVolume	String	12	Numeric	12	
OrderNote	String	25	Alpha	25	
ListID	String	6	Alpha	6	
Timestamp	String	20	DateTime	20	Order edit date and time

For every Order Edit Message accepted by the target exchange, a Confirm of Order Edit Message is sent to the originating member firm. It is also sent to the member firm when the Order Edit Message comes from OASIS ORAMA or from EMRW. The edited (canceled/activated/deactivated) order is identified by the Order number and Order date. This message is also sent when an ATO/MKT order is canceled at the open because it could not be matched or when an order requiring immediate matching, e.g. FOK, IOC cannot be matched upon receipt.

The "CancelSource" field is determined according to the origin of the Order Edit Message. In case of order edit by OASIS Server (*for example an ATO order that canceled at the open*), the field is blank.

The "TraderID" field contains the Trader ID from the Order Edit message. In case of order edit by OASIS Server, the field contains the Trader ID of the original order.

The "Member sequence number" contains the original number of Order Edit message that was received from a member firm. If the request was entered by OASIS ORAMA this field is zero.

The object has the following methods :

```
Function  FormatMessage
Returns String
```

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.1.6. ConfirmOrderChange (TD)

This object corresponds to the message Confirm of Order Change (TD). In the table that follows, all the object properties are described as well as the corresponding fields of the Confirm of Order Change message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TD” (Read Only)
Reserved			Alpha	4	“ “ Reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	(Read Only)
VenueID	String	4	Alpha	4	
BoardID	Long		Alpha	1	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
OrderNumber	String	8	Numeric	8	
OrderDate	String	8	Date	8	
ChangedPrice	String	9	Numeric	5.4	
ChangedVolume	String	12	Numeric	12	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
ChangedDisclosedVolume	String	12	Numeric	12	
ChangedAutoDisclosedVolume	String	12	Numeric	12	“000000000000” (Default) For Future Use
LeavesQuantity	String	12	Numeric	12	
AveragePrice	String	9	Numeric	5.4	
ChangedCSDAccountID	String	12	Alpha	12	
ChangedGOIFlag	String	1	Alpha	1	Not used
ChangedShortSellFlag	String	1	Alpha	1	
ChangedOriginalPriceType	Long		Alpha	1	
ChangedLife	Long		Alpha	1	
ChangedExpirationDate	String	8	Date	8	
OrigClientOrderID	String	16	Alpha	16	
ClientOrderID	String	16	Alpha	16	
ChangedOrderNote	String	25	Alpha	25	
ChangedClearingMemberID	String	4	Alpha	4	Clearing sub-account ID
ChangedPositionEffect	String	1	Alpha	1	
ChangedSettlType	String	1	Alpha	1	
ChangedSource	Long		Alpha	1	
ChangedOrderStatus	String	2	Alpha	2	
CurrentCreditValue	String	14	Numeric	12.2	
ListID	String	6	Numeric	6	
SpecialConditions	Long		Alpha	1	Used for FIX translation.
ChangedDirectElectronicAccess	String	1	Alpha	1	
ChangedClientID	String	10	Numeric	10	
ChangedClientIDQualifier	String	1	Alpha	1	
ChangedInvestmentDecisionID	String	10	Numeric	10	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
ChangedInvestmentDecisionIDQualifier	String	1	Alpha	1	
ChangedExecutionWithinFirmID	String	10	Numeric pha	10	
ChangedExecutionWithinFirmIDQualifier	String	1	Alpha	1	
ChangedNonExecutingBrokerID	String	10	Numeric	10	
Timestamp	String	20	DateTime	20	Order changedate and time
ChangedSpecialInstructions	String	120	Alpha	120	Only used for XNET. This field does not exist in messages originating from the ETS interface.

The object is used when:

- Member changes through ATHEX Gateway an order (OrderChange, MD), and the change is confirmed.

- A Market order was partially matched and changed its status to limit. In this case OrderNumber will be the same with the Open order one. ChangedPrice will take the value given to the order from the order book.

The "Member sequence number" contains the original number of Order Edit message that was received from a member firm. If the request was entered by ORAMA (ETW), this field is zero.

The object has the following methods :

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields.

Messages that do not contain a "SpecialInstructions" field will result in objects with null value in this field.

Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.1.7. Trade (TF)

This object corresponds to the message Confirm of Trade (TF). In the table that follows, all the object properties are described as well as the corresponding fields of the Confirm of Trade message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TF” (Read Only)
Reserved			Alpha	4	“ “ Reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	(Read Only)
VenueID	String	4	Alpha	4	
BoardID	Long		Alpha	1	
ClientOrderID	String	16	Alpha	16	Can be used by the member to identify the original quote/combo order. (If ClientOrderID is used as an identifier, member must ensure its uniqueness.) IMPORTANT: This can also be achieved through field OrdRefId.
OrigClientOrderID	String	16	Alpha	16	Used in trade reporting to refer to previous ClientOrderID in a sequence of messages.
CSDAccountID	String	12	Alpha	12	
OrderRelFlag	String	1	Alpha	1	
GOIFlag	String	1	Alpha	1	Not used
ShortSellFlag	String	1	Alpha	1	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
ClearingMemberID	String	4	Alpha	4	
PositionEffect	String	1	Alpha	1	
SettlType	String	1	Alpha	1	
OrderSource	Long		Alpha	1	A single character alphanumeric type indicating the source of the Order. Possible values : ‘C’ CTCI –ODL ‘M’ ORAMA-ETW ‘R’ EMRW (ATHEX Supervision Application).
OrderNumber	String	8	Numeric	8	* see below
OrderDate	String	8	Date	8	* see below
ExpirationDate	String	8	Date	8	* see below
OrderStatus	String	2	Alpha	2	* see below
LeavesQuantity	String	12	Numeric	12	Order Remaining Volume * see below
AveragePrice	String	9	Numeric	5.4	Format is 5.4 * see below
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
Side	Long		Alpha	1	
Volume	String	12	Numeric	12	
Price	String	9	Numeric	5.4	
ContraMemberID	String	4	Alpha	4	Filled only for trade reports
ContraTraderID	String	5	Alpha	5	Filled only for trade reports
TradeNumber	String	6	Numeric	6	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
CurrentCreditValue	String	14	Numeric	12.2	
ListID	String	6	Alpha	6	
TradeSource	Long		Alpha	1	
PhaseID	Long		Alpha	1	
SecurityStatus	Long		Alpha	1	
TradeType	String	2	Alpha	2	
TradeStatus	String	2	Alpha	2	
OrdRefId	String	8	Alpha	8	
LastLiquidityIndicator	String	1	Alpha	1	
NotionalAmmount	String	14	Numeric		Trade Value
DirectElectronicAccess	String	1	Alpha	1	
ClientID	String	10	Numeric	10	
ClientIDQualifier	String	1	Alpha	1	
InvestmentDecisionID	String	10	Numeric	10	
InvestmentDecisionIDQualifier	String	1	Alpha	1	
ExecutionWithinFirmID	String	10	Numeric	10	
ExecutionWithinFirmIDQualifier	String	1	Alpha	1	
NonExecutingBrokerID	String	10	Numeric	10	
TradingCapacity	String	1	Alpha	1	
LiquidityProvision	String	1	Alpha	1	
WaiverIndicator	String	4	Alpha	4	
BestBidPrice	String	9	Numeric	5.4	
BestBidQuantity	String	12	Numeric	12	
BestOfferPrice	String	9	Numeric	5.4	
BestOfferQuantity	String	12	Numeric	12	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
Timestamp	String	20	DateTime	20	Trade date and time

* Not filled for trade reports, i.e. OrderRelFlag == “R”

This message is sent to each side when orders are automatically matched to create a trade. This message is also sent to each side or trade, when OASIS (or a different exchange) receives Trade Reports from either CTCI or ETW or EMRW or XNet.

The object has the following methods :

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.1.8. Reject (TR)

This object corresponds to the message Reject (TR). In the table that follows, all the object properties are described as well as the corresponding fields of the Reject message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TR” (Read Only)
Reserved			Alpha	4	“ “ Reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	(Read Only)

VenueID	String	4	Alpha	4	
RejectReasonCode	String	3	Numeric	3	
ClientOrderID	String	16	Alpha	16	
OrderStatus	String	2	Alpha	2	Left blank (“ “) in case of unsuccessful order entry, or edit/change to an incorrect order number. Otherwise, this indicates the status of the order following the unsuccessful edit/change.
OriginalMessageText	String	400	Alpha	400	
Timestamp	String	20	DateTime	20	Date and time of rejection

A Reject Message is returned to the originating member firm when an invalid order entry/edit/change message is received by the target exchange. It is also returned to the originating member firm when an invalid quote entry/change/cancel message is received by the target exchange (see paragraphs 4.2.3.2.1 and 4.2.3.2.2). The Reject reason code explains why the exchange could not carry out the request.

Attention: In cases where the original message exceeds 400 bytes, only the first 400 bytes will be included in the OriginalMessageText field.

The field MemberSequenceNumber has the same value with the value of MemberSequenceNumber from the object that was rejected.

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.2. **Quotation Negotiation Handling objects**

Quotation negotiation handling objects correspond to messages send/received to/from OASIS or XOrder server through ATHEX Gateway from/to Applications using ATHEX Gateway. These objects are:

- QuoteEntryChange
- QuoteCancel
- QuoteStatusReport
- QuoteRequest
- QuoteRequestConfirmation
- QuoteRequestExecution
- QuoteRequestInfo
- QuoteResponsibilitySuspendResume
- QuoteAlarm

It must be noted that the member shall use only the QuoteStatus Report messages sent from the target exchange in order to verify that these quote entries, alterations and/or deletions have been accepted by the target exchange.

These objects are not supported for the Xorder Server connection.

4.2.3.2.1. QuoteEntryChange (MA)

This object corresponds to the message QuoteEntryChange (MA). In the table that follows, all the QuoteEntryChange object properties are described as well as the corresponding fields of the QuoteEntryChange message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“MA”
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	‘C’ = CTCI
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
VenueID	String	4	Alpha	4	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
Side	String	1	Alpha	1	
CSDAccountID	String	12	Alpha	12	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
BuyPrice	String	9	Numeric	5.4	
BuyVolume	String	12	Numeric	12	
SellPrice	String	9	Numeric	5.4	
SellVolume	String	12	Numeric	12	
QuoteID	Long	8	Numeric	8	Blank (for new quotes) or valid QuoteId (for changing existing quotes)
QuoteMsgID	String	16	Alpha	16	
QuoteNote	String	25	Alpha	25	
ClearingMemberID	String	4	Alpha	4	
DirectElectronicAccess	String	1	Alpha	1	
ClientID	String	10	Numeric	10	
ClientIDQualifier	String	1	Alpha	1	
InvestmentDecisionID	String	10	Numeric	10	
InvestmentDecisionIDQualifier	String	1	Alpha	1	
ExecutionWithinFirmID	String	10	Numeric	10	
ExecutionWithinFirmIDQualifier	String	1	Alpha	1	
NonExecutingBrokerID	String	10	Numeric	10	

This object is used by a member firm to enter a new quote or edit an existing one. The target exchange accepts the quote if the data is valid. (i.e., the data is readable and the values are allowable) and a QuoteStatusReport (TA) is sent to the member firm.

Otherwise, a Reject Message (TR) is sent to the member firm with a code specifying the reason why the quote was rejected (RejectReasonCode).

A unique quote identification is then assigned to the quote by OASIS Server in the specified date.

The odject has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

The above two methods are unique methods of all the objects corresponding to messages.

4.2.3.2.2. QuoteCancel (ME)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“ME”
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	‘C’ = CTCI
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
VenueID	String	4	Alpha	4	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
QuoteID	Long	8	Numeric	8	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
QuoteMsgID	String	16	Alpha	16	
QuoteNote	String	25	Alpha	25	

The object has the following methods :

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.2.3. QuoteStatusReport (TA)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TA”
Reserved		4	Alpha	4	Reserved for OASIS.
MessageSource	Long	1	Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
VenueID	String	4	Alpha	4	
Side	Long		Alpha	1	
CSDAccountID	String	12	Alpha	12	
SecurityID	String	15	Alpha	15	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
BuyRemainingVolume	String	12	Numeric	12	
BuyPrice	String	9	Numeric	5.4	
SellRemainingVolume	String	12	Numeric	12	
SellPrice	String	9	Numeric	5.4	
QuoteID	Long	8	Numeric	8	
QuoteMsgID	String	16	Alpha	16	
QuoteNote	String	25	Alpha	25	
QuoteStatus	Long	1	Alpha	1	
ClearingMemberID	String	4	Alpha	4	
QuoteSource	String	1	String	1	
CurrentCreditValue	String	14	Numeric	12.2	
DirectElectronicAccess	String	1	Alpha	1	
ClientID	String	10	Numeric	10	
ClientIDQualifier	String	1	Alpha	1	
InvestmentDecisionID	String	10	Numeric	10	
InvestmentDecisionIDQualifier	String	1	Alpha	1	
ExecutionWithinFirmID	String	10	Numeric	10	
ExecutionWithinFirmIDQualifier	String	1	Alpha	1	
NonExecutingBrokerID	String	10	Numeric	10	
Timestamp	String	20	DateTime	20	Quote confirmation date and time

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.2.4. QuoteRequest (MJ)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“MJ”
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	‘C’ = CTCI
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
VenueID	String	4	Alpha	4	
Side	Long		Alpha	1	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
Volume	Long	12	Numeric	12	For future use
QuoteReqID	String	16	Alpha	16	

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.2.5. QuoteRequestConfirmation (TJ)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TJ”
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
QuoteRequestRejectReason	String	3	Numeric	3	
VenueID	String	4	Alpha	4	
Side	Long		Alpha	1	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
Volume	Long	12	Numeric	12	FFU
QuoteReqID	String	16	Alpha	16	
ActionSource	String	1	String	1	
Timestamp	String	20	DateTime	20	

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.2.6. QuoteRequestExecution (TP)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TP”
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
QuoteRequestStatus	Long		Alpha	1	
QuoteRequestRejectReason	String	3	Numeric	3	
VenueID	String	4	Alpha	4	
Side	Long		Alpha	1	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
Volume	Long	12	Numeric	12	FFU
QuoteReqID	String	16	Alpha	16	
Timestamp	String	20	DateTime	20	

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.2.7. QuoteRequestInfo (TK)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	"TK"
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	' ' = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	"000000"
VenueID	String	4	Alpha	4	
Side	Long		Alpha	1	
SecuritySymbol	String	15	Alpha	15	
SecurityCode		12	Alpha	12	
Currency	String	3	Alpha	3	Only used for XNET.
Volume	Long	12	Numeric	12	FFU
QuoteReqID	String	16	Alpha	16	Client specified identifier of the quote request
Timestamp	String	20	DateTime	20	

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.2.8. QuoteResponsibility (TN)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TN”
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
VenueID	String	4	Alpha	4	
MarketID	String	1	Alpha	1	
SecuritySymbol	String	15	Alpha	15	
SecurityCode	String	12	Alpha	12	
Currency	String	3	Alpha	3	Only used for XNET.
QuoteResponsibilityStatus	Long	1	Alpha	1	
QuotationLevel	Long	1	Alpha	1	
ReasonOfSuspension	Long	1	Alpha	1	
Timestamp	String	20	DateTime	20	

The object has the following methods :

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage   [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.2.9. QuoteAlarm (TM)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TM”
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
VenueID	String	4	Alpha	4	
SecuritySymbol	String	15	Alpha	15	
SecurityCode	String	12	Alpha	12	
Currency	String	3	Alpha	3	Only used for XNET.
Alarm Type	Long	1	Alpha	1	
Alarm Reason	Long	1	Alpha	1	
Timestamp	String	20	DateTime	20	

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage   [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.3. Hit And Take Order Handling Objects

4.2.3.3.1. HitAndTakeOrderEntry (MF)

This object corresponds to the message Hit & Take Order (MF). In the table that follows, all the HitAndTakeOrderEntry object properties are described as well as the corresponding fields of the Hit & Take Order message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“MF” (Read Only, Filled by ODL)
Reserved			Alpha	4	“ “ (Filled by ODL), reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘C’ = CTCI (Default, Filled by ODL)
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000” (Read Only, Filled by ODL G/W)
VenueID	String	4	Alpha	4	
BoardID	Long		Alpha	1	Possible values: 'S': Special-Terms board 'F': Forced-sales board

Property/Field Name	Property Type	Len	Field Type	Len	Remark
CSDAccountID	String	12	Alpha	12	
GOIFlag	String	1	Alpha	1	Not used
ShortSellFlag	String	1	Alpha	1	
HitOrderNumber	String	8	Numeric	8	
HitOrderDate	String	8	Date	8	
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
Volume	String	12	Numeric	12	
ClientOrderID	String	16	Alpha	16	
OrderNote	String	25	Alpha	25	
BuyClearingMemberID	String	4	Alpha	4	
DirectElectronicAccess	String	1	Alpha	1	
ClientID	String	10	Numeric	10	
ClientIDQualifier	String	1	Alpha	1	
InvestmentDecisionID	String	10	Numeric	10	
InvestmentDecisionIDQualifier	String	1	Alpha	1	
ExecutionWithinFirmID	String	10	Numeric	10	
ExecutionWithinFirmIDQualifier	String	1	Alpha	1	
NonExecutingBrokerID	String	10	Numeric	10	

This object is used by a member firm to **enter a hit & take order**. The target exchange accepts the **hit & take** order if the data is valid (i.e., the data is readable and the values are allowable). A Confirm of a New Order Message (TB) (paragraph 4.2.3.1.4**Error! Reference source not found.**) and a Confirm of Trade Message (TF) (paragraph 4.2.3.1.7) are sent to the member firm. Otherwise, a Reject Message (TR) (paragraph 4.2.3.1.8**Error! Reference source not found.**) is sent to the member firm with a reason code explaining why the order was rejected.

The odject has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.4. Trade Reports (PreAgreed Trades) Handling Object

The Trade report handling object corresponds to the message send to OASIS through ATHEX Gateway from Applications using ATHEX Gateway in order to enter/approve/disapprove/cancel a trade report.

It must be noted that the member shall use only the TradeInfo (TF) message sent from the target exchange in order to verify that these pre-agreed price trades entries, approvals, dissaprovals and/or cancelations have been accepted by the target exchange.

4.2.3.4.1. TradeReportEntry (MI)

This object corresponds to the message TradeReportEntry (MI). In the table that follows, all the radeReportEntry object properties are described as well as the corresponding fields of the TradeReportEntry message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“MI” (Read Only, Filled by ODL)
Reserved	String	4	Alpha	4	“ “ (Filled by ODL), reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘C’ = CTCI (Default, Filled by ODL)
MemberID	String	4	Alpha	4	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000” (Read Only, Filled by ODL G/W)
VenueID	String	4	Alpha	4	
BoardID	Long		Alpha	1	For trade reports only value “B” (report only board) is allowed
SecurityID	String	15	Alpha	15	
SecurityIDSource	String	1	Alpha	1	
Currency	String	3	Alpha	3	Only used for XNET.
Price	String	9	Numeric	5.4	
Volume	String	12	Numeric	12	
ClientTradeID	String	16	Alpha	16	For internal use of the member firm. Returned in TF message’s ClientOrderId field.
SellMemberID	String	4	Alpha	4	
SellTraderID	String	5	Alpha	5	
SellDirectElectronicAccess	String	1	Alpha	1	
SellClientID	String	10	Numeric	10	
SellClientIDQualifier	String	1	Alpha	1	
SellInvestmentDecisionID	String	10	Numeric	10	
SellInvestmentDecisionIDQualifier	String	1	Alpha	1	
SellExecutionWithinFirmID	String	10	Numeric	10	
SellExecutionWithinFirmIDQualifier	String	1	Alpha	1	
SellNonExecutingBrokerID	String	10	Numeric	10	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
SellCSDAccountID	String	12	Alpha	12	
ShortSellFlag	String	1	Alpha	1	
SellClearingMemberID	String	4	Alpha	4	
SellPositionEffect	String	1	Alpha	1	
SellTradingCapacity	String	1	Alpha	1	
BuyMemberID	String	4	Alpha	4	
BuyTraderID	String	5	Alpha	5	
BuyDirectElectronicAccess	String	1	Alpha	1	
BuyClientID	String	10	Numeric	10	
BuyClientIDQualifier	String	1	Alpha	1	
BuyInvestmentDecisionID	String	10	Numeric	10	
BuyInvestmentDecisionIDQualifier	String	1	Alpha	1	
BuyExecutionWithinFirmID	String	10	Numeric	10	
BuyExecutionWithinFirmIDQualifier	String	1	Alpha	1	
BuyNonExecutingBrokerID	String	10	Numeric	10	
BuyCSDAccountID	String	12	Alpha	12	
BuyToCoverFlag	String	1	Alpha	1	
BuyClearingMemberID	String	4	Alpha	4	
BuyPositionEffect	String	1	Alpha	1	
BuyTradingCapacity	String	1	Alpha	1	
SDivFlag	String	1	Alpha	1	
DuplFlag	String	1	Alpha	1	
TradeReportMethod	String	2	Alpha	2	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
TradeEditType	Long		Alpha	1	
TradeNumber	String	6	Numeric	6	
Timestamp	String	20	DateTime	20	

This message is used by a member firm to enter/approve/disapprove/cancel a **Trade report (pre-agreed price trade)**.

Message details and business logic for the use of this message described in chapter 6.2.4 Trade Report Entry

A unique trade number is generated by OASIS.

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.5. Market Data Objects

Market objects correspond to messages sent from an exchange through ATHEX Gateway to Applications using ATHEX Gateway. These objects are:

- CreditLimitInfo
- HitTakeOrderInfo
- SecurityStatus
- SystemStatus
- MarketStatus

- SecurityPrices
- ExchangeNotes

It must be noted that most of the above status messages are supported only for the OASIS connection and not for the XOrder server connection. Check bellow to see which of them are supported for the XOrder server connection.

4.2.3.5.1. CreditLimitInfo (TL)

This object corresponds to the message Credit Limit Information (TL). In the table that follows all the object properties are described as well as the corresponding fields of the Credit Limit Information message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TL” (Read Only)
Reserved			Alpha	4	“ “ Reserved for OASIS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
MemberSequenceNumber	String	6	Numeric	6	(Read Only)
ClearingSpace	String	4	Alpha	4	
ClearingSubAccountId	String	4	Alpha	4	
CreditLimit	String	14	Numeric	12.2	

This message is sent from OASIS Server specifically to each Member at the beginning of each trading day and for each credit limit method. It is also sent whenever the Member’s credit limit for a method is changed during trading day.

This message is also sent by the XOrder Server connection.

The object has the following methods :

```
Function  FormatMessage
returns string
```

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage as string

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.5.2. HitAndTakeOrderInfo (CH)

This object corresponds to the message Hit & Take Order Information (CH). In the table that follows all the object properties are described as well as the corresponding fields of the Hit & Take Order Information message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“CH” (Read Only)
MessageSource	Long		Alpha	1	‘ ’ = OASIS
VenueID	String	4	Alpha	4	
OrderNumber	String	8	Numeric	8	
OrderDate	String	8	Date	8	
SecuritySymbol	String	15	Alpha	15	
SecurityCode	String	12	Alpha	12	
Currency	String	3	Alpha	3	Only used for XNET.
BoardID	Long		Alpha	1	
Side	Long		Alpha	1	
Price	String	9	Numeric	5.4	
Volume	String	12	Numeric	12	
DisclosedVolume	String	12	Numeric	12	
SpecialConditions	Long		Alpha	1	Possible values are: “A” All Or None (Apply only Special board)

					“M” Minimum Fill (Apply only Special board) “O” Multiple of (Apply only Special board)
ConditionVolume	String	12	Numeric	12	
OrderStatus	String	2	Alpha	2	

This message is sent from OASIS Server to all Members whenever a special order for which a Hit & Take order can be entered is entered and/or altered (changed, activated, deactivated, canceled, partially or totally matched).

More specifically, whenever an order is entered and/or altered in the Forced Sales board or in the Special Terms board, a Hit & Take Order Information message is sent to all Members.

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.5.3. SecurityPrices (CD)

This object corresponds to the message Security Prices (CD). In the table that follows all the object properties are described as well as the corresponding fields of the Security Prices message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“CD” (Read Only)
VenueID	String	4	Alpha	4	
SecuritySymbol	String	15	Alpha	15	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
SecurityCode	String	12	Alpha	12	
StartOfDayPrice	String	9	Numeric	5.4	
FloorPrice	String	9	Numeric	5.4	
CeilingPrice	String	9	Numeric	5.4	
AccruedInterest	String	14	Numaric	8.6	

This is sent from OASIS Server at the beginning of each trading day and whenever the floor or ceiling percentage level of a security is changed.

This message is also sent by the XOrder Server connection.

The object has the following methods :

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage   [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.5.4.   SecurityStatus (CA)

This object corresponds to the message Security Status (CA). In the table that follows all the object properties are described as well as the corresponding fields of the Security Status message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“CA” (Read Only)
VenueID	String	4	Alpha	4	
SecuritySymbol	String	15	Alpha	15	

SecurityCode	String	12	Alpha	12	
SecurityStatus	Long		Alpha	1	
PhaseID	Long		Alpha	1	
SecurityPrice	String	9	Numeric	5.4	
HaltReasonCode	String	2	Numeric	2	
HaltStartTime	String	12	Time	12	
MarketID	Long		Alpha	1	
Timestamp	String	20	DateTime	20	

In cases of negative price values the sign (-) is incorporated in this field, reducing the numeric part to 4.4 format. The negative sign should occupy the first character of the integer part, but should be omitted in positive values. E.g a price of "-1.23" translates to "-00012300"

This message is sent at the beginning of each trading day and whenever the market phase changes for a specific security or whenever the security status changes.

The object has the following methods :

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.5.5. MarketStatus (CC)

This object corresponds to the message Market/Board Status (CC). In the table that follows all the object properties are described as well as the corresponding fields of the Market/Board Status message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“CC” (Read Only)
VenueID	String	4	Alpha	4	
MarketID	Long		Alpha	1	
BoardID	Long		Alpha	1	
Status	Long		Alpha	1	

This message is sent from OASIS Server whenever the market status or board status is changed. Whenever the change applies to the whole market (as in cases of market halt, run-off and end) the Main Board Id ‘M’ is used in the board id field. For non-main boards, the message is sent when the board opens and closes.

The object has the following methods :

Function FormatMessage
Returns String

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage As String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.5.6. SystemStatus (CB)

This object corresponds to the message System Status (CB). In the table that follows all the object properties are described as well as the corresponding fields of the System Status message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“CB” (Read Only)
VenueID	String	4	Alpha	4	
Status	Long		Alpha	1	

This message is sent from OASIS Server whenever trading in the entire system is halted or resumed.

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.5.7. ExchangeNotes (TO)

This object corresponds to the message Exchange Notes (TO). In the table that follows all the object properties are described as well as the corresponding fields of the Exchange Notes message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TO”
Reserved			Alpha	4	Reserved for OASIS.
MessageSource	Long		Alpha	1	‘ ’ = OASIS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000”
NoteType	String	1	Alpha	1	
MessageNote	String	50	Alpha	50	
Timestamp	String	20	DateTime	20	

This message is sent from OASIS Server each time a note shall be forwarded to a member. An exchange note message field NoteType set to “1” (Throttling Warning) informs the member that their CTCI rejections have reached half of the disabling threshold. In this case, the member should take care to adjust/amend any possible issues causing excessive message rejections from the central platform.

The object has the following methods :

Function FormatMessage Returns String
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage As String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

4.2.3.6. Clearing Management Objects

Clearing management objects correspond to messages send **only** through the ATHEX Gateway. These objects are:

- DSSEntry
- DSSConfirm
- DSSTrade
- DSSBroadcast

4.2.3.6.1. DSSEntry (MS)

This object corresponds to the message DSS Entry (MS). In the table that follows, all the DSS Entry object properties are described as well as the corresponding fields of the DSS Entry message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“MS” (DSS Message)
Reserved			Alpha	4	“ “ (Filled by ODL), reserved for ETS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘C’ = CTCI (Default, Filled by ODL)
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MemberSequenceNumber	String	6	Numeric	6	“000000” (Read Only, Filled by ODL G/W)
VenueID	String	4	Alpha	4	
Type	Long		Alpha	1	‘A’ (Default) For Future Use.
ClientOrderID	String	16	Alpha	16	
Length	String	5	Numeric	5	
DSSMessage	String		Alpha		
CRC	String	4	Alpha	4	

This object is used by a member firm to send a request to the DSS Server through the DSS interface only. The target server accepts the request if the data is valid. (i.e., the data is readable and the values are allowable) and replies with a DSS Confirm (TS) message. The DSSMessage field should contain a valid DSS request according to DSS specifications.

The odject has the following methods :

Function FormatMessage returns string
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage as String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

The above two methods are unique methods of all the objects corresponding to messages.

4.2.3.6.2. DSSConfirm (TS)

This object corresponds to the message DSS Confirm message (TS). In the table that follows, all the object properties are described as well as the corresponding fields of the DSS Confirm message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TS” (DSS Message)
Reserved			Alpha	4	“ “ (Filled by ODL), reserved for ETS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘ ’ = DSS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000” (Read Only, Filled by ODL G/W)
VenueID	String	4	Alpha	4	
Type	Long		Alpha	1	‘A’ (Default) For Future Use.
NewOrderNumber	String	12	Numeric	12	
ClientOrderID	String	16	Alpha	16	
Length	String	5	Numeric	5	
DSSMessage	String		Alpha		
CRC	String	4	Alpha	4	

This object is used by a member firm to receive request confirmation from the DSS Server through the DSS interface only.

The message returns to the member firm the unique message number that was assigned to the request by the DSS. This message number is used by DSS to identify the request and binds a request with corresponding responses. If the request failed to be accepted by DSS the newOrderNumber is 0.

The DSSMessage field contains information regarding the acceptance or rejection of the request.

The object has the following methods :

```
Function  FormatMessage
returns string
```

Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage as String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

The above two methods are unique methods of all the objects corresponding to messages.

4.2.3.6.3. DSSTrade (TU)

This object corresponds to the message DSS Trade Message (TU). In the table that follows, all the object properties are described as well as the corresponding fields of the DSS Reply message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“TU” (DSS Message)
Reserved			Alpha	4	“ “ (Filled by ODL), reserved for ETS. This field exists only in message and there is no respective property
MessageSource	Long		Alpha	1	‘ ’ = DSS
MemberID	String	4	Alpha	4	
TraderID	String	5	Alpha	5	
MemberSequenceNumber	String	6	Numeric	6	“000000” (Read Only, Filled by ODL G/W)
VenueID	String	4	Alpha	4	
Type	Long		Alpha	1	‘A’ (Default) For Future Use.
OrderNumber	String	12	Numeric	12	
ClientOrderID	String	16	Alpha	16	
Length	String	5	Numeric	5	

Property/Field Name	Property Type	Len	Field Type	Len	Remark
DSSMessage	String		Alpha		
CRC	String	4	Alpha	4	

This object is used by a member firm to receive unsolicited information or request execution results from the DSS Server through the DSS interface only.

The DSSMessage contains information, as a response to a request submitted by a MS message or as a result of an information dissemination procedure.The object has the following methods :

Function FormatMessage
returns string

Constructs a message (sequence of characters) from the fields of the object.

Sub ParseMessage [in] bsMessage as String

Takes as input a message string and it splits it to the object fields. Typical error code for this function :

E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

The above two methods are unique methods of all the objects corresponding to messages.

4.2.3.6.4. DSSBroadcast (CS)

This object corresponds to the message DSS Broadcast Message (CS). In the table that follows, all the object properties are described as well as the corresponding fields of the DSS Broadcast message.

Property/Field Name	Property Type	Len	Field Type	Len	Remark
MessageType	String	2	Alpha	2	“CS” (DSS Message)
Reserved			Alpha	4	“ “ (Filled by ODL), reserved for ETS. This field exists only in message and there is no respective property
VenueID	String	4	Alpha	4	
Type	Long		Alpha	1	‘A’ (Default)

Property/Field Name	Property Type	Len	Field Type	Len	Remark
					For Future Use.
Length	String	5	Numeric	5	
DSSMessage	String		Alpha		
CRC	String	4	Alpha	4	

This object is used by a member firm to receive broadcast information from the DSS Server through the DSS interface only.

The object has the following methods :

Function FormatMessage returns string
Constructs a message (sequence of characters) from the fields of the object.
Sub ParseMessage [in] bsMessage as String
Takes as input a message string and it splits it to the object fields. Typical error code for this function : E_ INVALIDARG (0x80070057) Invalid argument passed (NULL) in the function or invalid length of the message.

The above two methods are unique methods of all the objects corresponding to messages.

4.3. Control Messages

Except from the data messages between Comm Server and the ATHEX Gateway (MB, MC, MD, TB, CA etc.) there is another kind of messages called “Control” messages which contain a different kind of information. An equivalent interaction mechanism has been implemented between ATHEX Gateway and the XOrder server.

With the use of the Administrator object an application can obtain these messages and do appropriate actions through the NewCCMMessage function. The exploitation of the control messages is not necessary from the member’s application but it does help to create a robust and more accurate system. All the control messages are available only during the trading day and are not stored in the local members file.

The NewCCMMessage function always returns a string but the fields inside the message, specially the CTRL_DATA, many times are of numeric form.

4.3.1. *Control Message Format*

These are fixed-length ‘packets’. Their format is described in Figure 4-1:

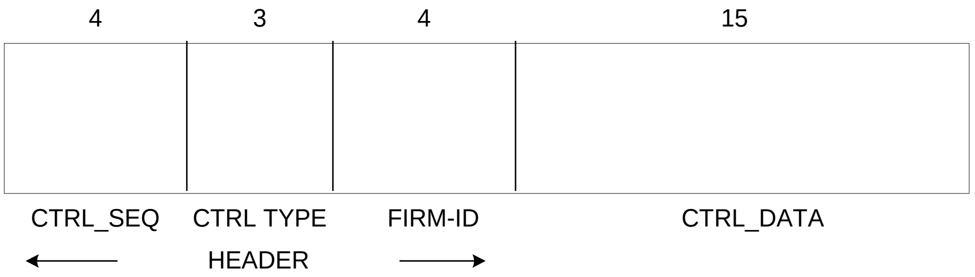


Figure 4-1 Control Message Format

4.3.2. *Control Message Header*

The fields of the header of the messages are:

Name	Length	Type
CTRL_SEQ	4	Numeric
CTRL_TYPE	3	Numeric
Firm –ID	4	Numeric

4.3.3. *Control Messages received from ATHEX Gateway*

The messages received from the ATHEX Gateway are the following:

Code	Remarks	Format

500	PT_MT_LINKDOWN
	Comm Server has a severe problem sending messages to OASIS Server.
		CCM.SEQ_NO = XXXX
		CCM.CTRL_TYPE = 500
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = ‘\ 0’

501	PT_MT_LINKRESUMED
	Problem resolved.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 501
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = ‘\ 0’

Code	Remarks	Format

502	PT_MT_LINKRSC
	Temporary resource issue caused while the Comm Server tries to transfer the member message to OASIS Server
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 502
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = ‘\ 0’

503	PT_MT_LINKRSCOK
	Problem resolved.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 503
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = ‘\ 0’

508	PT_MT_LASTRECVD
	The last message received from the member application.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 508
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = Contains in arithmetic form the last message that received (NUM [8]).

509	PT_MT_RETRANSMIT
	Issued when Comm Server wants a retransmission of messages from the ATHEX Gateway.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 509
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.DATA = Contains first (in arithmetic form) the “FROM” sequence number and then the “TO” sequence number.

510	PT_MT_LOGINOK
	ATHEX GATEWAY Login/Password success.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 510
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = ‘\ 0’

511	PT_MT_LOGINREJECT
	ATHEX GATEWAY Login/Password failed.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 511
		CCM.FIRM_ID = YYYY – ID of Firm

Code	Remarks	Format
	Reason is kept in the CTRL_DATA.REASON field.	CCM.CTRL_DATA = Reason of failure

530	PT_MT_INVALIDCTRL
	Invalid control message sent from the ATHEX GATEWAY. Sequence number of the invalid message is in the CTRL_DATA field.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 530
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA Sequence Number (in arithmetic form) of invalid message (NUM [4])

540	PT_MT_INVALIDDATA
	Invalid application message sent from ATHEX GATEWAY. Sequence number of the invalid message in field CTRL_DATA.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 540
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA Sequence Number (in arithmetic form) of invalid message (NUM [8])

541	PT_MT_RXMTRANGERR
	ATHEX GATEWAY sent a retransmission request with invalid range.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 541
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA contains the Sequence number of the invalid message (NUM [8])

550	PT_MT_UNDEFINED
	Undefined error message.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 550
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA .= '\0'

570	PT_MT_INVALID_SEQ_NUM	CCM. SEQ_NO = XXXX CCM.CTRL_TYPE = 570

Code	Remarks	Format
	Invalid Sequence number.	CCM.FIRM_ID = YYYY – ID of Firm CCM.CTRL_DATA Contains first the sequence (in arithmetic form) of the message received and then the sequence which was supposed to receive from the ATHEX Gateway.

590	PT_MT_LOGGEDOUT
	Comm Server has loged out the ATHEX Gateway.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 590
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = '\ 0'

4.3.4. *Control Messages send from ATHEX Gateway*

The messages send from the ATHEX Gateway are the following:

Code	Remarks	Format

401	MT_PT_LINKRSC
	Temporary resource issue in sending messages from the ATHEX Gateway to the members trading application.
		CCM.SEQ_NO = XXXX
		CCM.CTRL_TYPE = 401
		CCM.FIRM_ID = YYYY – ID of Firm.
		CCM.CTRL_DATA = '\ 0'

402	MT_PT_LINKRSCOK
	Problem resolved
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 402
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = '\ 0'

405	MT_PT_RETRANSMIT
	Issued when ATHEX GATEWAY wants a retransmission from the Comm server.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 405
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.DATA Contains first the sequence of the start of the range to be retransmitted

Code	Remarks	Format
		(NUM [8]) and then the sequence of the end of the retransmission (NUM [8]).

406	MT_PT_LASTRECVD
	The last message received by the ATHEX Gateway.
		CCM.SEQ_NO = XXXX
		CCM.CTRL_TYPE = 406
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA Contains in numeric format the last message received (NUM [8])

407	MT_PT_LOGIN
	Attempt to login by the ATHEX GATEWAY to the Comm Server.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 407
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA Includes the password. The user cannot see the password for security purposes.

430	MT_PT_INVALIDCTRL
	Invalid control message. The Sequence number is shown in the CTRL_DATA field.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 430
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM. CTRL_DATA contains in numeric format the Sequence number of the incorrect message (NUM [4])

440	MT_PT_INVALIDDATA
	Invalid information message. The Sequence number is shown in the CTRL_DATA field.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 440
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA contains in numeric format the Sequence number of the incorrect message (NUM [8])

441	MT_PT_RXMTRANGERR
	The Comm Server sent a message for retransmission of messages that were out of range.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 441
		CCM.FIRM_ID = YYYY – ID of Firm

Code	Remarks	Format
		CCM.CTRL_DATA contains in numeric format the Sequence number of the incorrect message (NUM [4])

450	MT_PT_UNDEFINED
	Unidentified error message.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 450
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA .= ‘\0’

470	MT_PT_INVALID_SEQ_NUM
	Invalid Sequence number.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 470
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA Contains first (in numeric format) the message number received by the Comm Server and then the message number that it should have received by it.

490	MT_PT_LOGOUT
	ODL requests a Logout.
		CCM. SEQ_NO = XXXX
		CCM.CTRL_TYPE = 490
		CCM.FIRM_ID = YYYY – ID of Firm
		CCM.CTRL_DATA = ‘\ 0’

4.4. Data flow

Messages sent by the Member Application to OASIS or the Xorder Server, using the relevant objects, will cause the confirmation or rejection messages to be sent as listed in the following Table:

s/n	Messages sent	Messages received

1	New Order (MB)	• *Confirm of New Order (TB)* or, • *Reject (TR)* *(if the incoming order meets the Matching Criteria and a Trade is produced, then a Confirm of Trade Message TF will be received)*
2	Order Edit (MC)	• *Confirm of Order Edit (TC)* or, • *Reject (TR)*
3	Order Change (MD)	• *Confirm of Order Change (TD)* or, • *Reject (TR)* *(if the changed order meets the Matching Criteria and a Trade is produced, then a Confirm of Trade Message will be received)*
4	Trade Report Entry (MI)	• *Confirm of Trade (TF)* or, • *Reject (TR)* *(in case of valid entry of a 2-firm pre-agreed price trade the TF message will be sent to the member of each side)*
5	Hit & Take Order (MF)	• *Confirm of New Order (TB) and a Confirm of Trade (TF)* or, • *Reject (TR)*
7	Quote Entry / Change (MA)	• *Quote Status Report (TA) or* • *Reject (TR)*
8	Quote Cancel (ME)	• *Quote Status Report (TA) or* • *Reject (TR)*
9	Quote Request (MJ)	• *Quote Request Info (TK) or* • *Reject (TR)*

5. FIX client application

In this chapter there is a detailed description of the FIX messages available to client applications, as well as supplements concerning the rules of ATHEX trading system. ATHEX Gateway and more specifically the ASE Service supports FIX protocol version 4.4.

It must be noted that the FIX protocol interface of the ATHEX Gateway supports only a part of the functionality supported by the ODL application message interface. More specifically, the following functionalities are supported:

1. Single General Order Handling
2. Quotation Negotiation Handling
3. Trade Reports Handling
4. Market Messages
5. DSS & Clearing

In terms of messages, the following are supported:

OASIS Trading

- Order Entry
- Order Edit
- Order Change
- Order Entry Confirmation
- Order Edit Confirmation
- Order Change Confirmation
- New Trade Confirmation
- Rejection
- TradeReportEntry
- Quote Entry Change
- Quote Cancel
- Quote Status Report
- Quote Request
- Quote Request Response

- Quote Request Info
- Quote Responsibility Suspend Resume
- Quote Alarm
- Security Status
- Security Price
- Security Info
- Combination Info
- Credit Limit Info
- Market Status
- Exchange Notes

DSS & Clearing

FIX access to DSS & clearing functionality adheres to its ODL equivalent, by employing custom messages each carrying one particular payload field (DSS Message) formatted in FIXML 5.0 SP2.

- DSS Entry
- DSS Confirm
- DSS Trade
- DSS Broadcast

5.1. Single General Order Handling

5.1.1. *New Order – Single*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	D = NewOrderSingle
Message Body			
11	ClOrdID	Y	Client specified identifier of the order. Max size is 16-char.

<table>
<tr><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Class or source of the PartyID (448) value.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>D</td><td>D (Proprietary/Custom Code)
Used for the PartyRole values 1, 4, and 36</td></tr>
<tr><td>P</td><td>Short code identifier
Used for the PartyRole values 3, 12, 26 and 122</td></tr>
</table></td></tr>
<tr><td>➡</td><td>2376</td><td>PartyRoleQualifier</td><td>N</td><td>Qualifier of the specified PartyRole (452).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>22</td><td>Algorithm
Used for the PartyRole values 12 and 122</td></tr>
<tr><td>23</td><td>Firm or legal entity
Used for the PartyRole value 3</td></tr>
<tr><td>24</td><td>Natural person
Used for the PartyRole values 3, 12 and 122</td></tr>
</table></td></tr>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Executing Firm</td></tr>
<tr><td>3</td><td>Client ID
(MIFID II: Client identification code)</td></tr>
<tr><td>4</td><td>Clearing Firm</td></tr>
<tr><td>12</td><td>Executing trader
(MIFID II: Execution within firm)</td></tr>
<tr><td>26</td><td>Correspondent broker (MIFID II: Non-executing broker)</td></tr>
<tr><td>36</td><td>Entering trader (Trader ID)</td></tr>
<tr><td>122</td><td>Investment Decision Maker
(MIFID II: Investment decision within firm)</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">1</td><td>Account</td><td>Y</td><td>Investor ID</td></tr>
<tr><td colspan="2">15</td><td>Currency</td><td>N</td><td>Used only for XNET</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Security identification</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID and StopSymbol(5521) if given.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>8</td><td>Exchange Symbol</td></tr>
<tr><td>A</td><td>Bloomberg Symbol</td></tr>
</table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
<tr><td colspan="2">40</td><td>OrdType</td><td>Y</td><td>Type of the order.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Market</td></tr>
<tr><td>7</td><td>Limit or Better</td></tr>
<tr><td>3</td><td>Stop</td></tr>
<tr><td>4</td><td>Stop Limit</td></tr>
<tr><td>A</td><td>On Close</td></tr>
</table>
For Stop Orders (OrdType=3 or OrdType=4) ATHEX Gateway will use the tags StopSymbol(5521) with value the SecuritySymbol of the order. Also ATHEX Gateway will use the value of tag StopSymbolType(5527) ‘S’ which means stop on Symbol.</td></tr>
</table>

<table>
<tr><td>59</td><td>TimeInForce</td><td>N</td><td><table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Day</td></tr><tr><td>1</td><td>Good Till Cancel (GTC)</td></tr><tr><td>2</td><td>At the Opening</td></tr><tr><td>3</td><td>Immediate Or Cancel (IOC)</td></tr><tr><td>4</td><td>Fill Or Kill (FOK)</td></tr><tr><td>6</td><td>Good Till Date (GTD)</td></tr></table></td></tr>
<tr><td>432</td><td>ExpireDate</td><td>N</td><td>Date the order expires. Required if TimeInForce (59) is GTD (6)</td></tr>
<tr><td>58</td><td>Text</td><td>N</td><td></td></tr>
<tr><td>44</td><td>Price</td><td>N</td><td>Required for limit OrdTypes.</td></tr>
<tr><td>99</td><td>StopPx</td><td>N</td><td>Required if OrdType=3 or OrdType=4.</td></tr>
<tr><td>38</td><td>OrderQty</td><td>Y</td><td>This tag is used to specify the quantity of the order.</td></tr>
<tr><td>63</td><td>SettlType</td><td>N</td><td>Indicates order settlement period.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Normal</td></tr><tr><td>1</td><td>Immediate</td></tr></table>Absence of this field is interpreted as normal settlement period</td></tr>
<tr><td>528</td><td>OrderCapacity</td><td>N</td><td>Indicates order capacity.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>A</td><td>Agency (AOTC)</td></tr><tr><td>P</td><td>Principal (DEAL)</td></tr><tr><td>R</td><td>Riskless principal (MTCH)</td></tr></table>Absence of this field is interpreted as agency</td></tr>
</table>

<table>
<tr><td>77</td><td>PositionEffect</td><td>N</td><td>For use in derivatives omnibus accounting.
Indicates whether the resulting position after a trade should be an opening position or closing position.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>O</td><td>Open</td></tr><tr><td>C</td><td>Close</td></tr></table>Absence of this field is interpreted as opening position.</td></tr>
<tr><td>210</td><td>MaxShow</td><td>N</td><td>Denotes the disclosed volume of the order. If not sent, it is assumed equal to OrderQty(38). If equal to 0, then it denotes an inactive order</td></tr>
<tr><td>54</td><td>Side</td><td>Y</td><td>Side of the order.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Buy</td></tr><tr><td>2</td><td>Sell</td></tr><tr><td>5</td><td>Sell short</td></tr><tr><td>R</td><td>Buy to cover</td></tr></table></td></tr>
<tr><td>60</td><td>TransactTime</td><td>Y</td><td>Time this order request was initiated.</td></tr>
<tr><td colspan="4"><b>Standard Trailer</b></td></tr>
</table>

The following table contains the extra and custom tags that can be sent in this message.

Tag	Field Name	Req	Description
5506	BoardID	N	Identifies board. Absence of this field is interpreted as the default value ‘M’ (the main board).
5521	StopSymbol	N	Required if OrdType=3 or OrdType=4. For stop orders, this is the instrument identification of stop symbol. If missing, tag 48 will be used. For class or source of this field, check tag 22.
5527	StopSymbolType	N	Possible values are ‘I’ for stop on index order and ‘S’ for stop on symbol orders. Absence of this tag means ‘S’ i.e. stop on symbol.

<table>
<tr><td>5561</td><td colspan="2">MBListID</td><td>N</td><td>Not required. If given, it will be used for clearing purposes.</td></tr>
<tr><td>1724</td><td colspan="2">OrderOrigination</td><td>N</td><td>Indicates order origination.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Order is not submitted using Direct Electronic Access (DEA)</td></tr><tr><td>5</td><td>Order is submitted using Direct Electronic Access (DEA)</td></tr></table>Absence of this field is interpreted as “order is not submitted using Direct Electronic Access (DEA)”.</td></tr>
<tr><td>2593</td><td colspan="2">NoOrderAttributes</td><td>N</td><td>Number of order attributes.</td></tr>
<tr><td>➡</td><td>2594</td><td>OrderAttributeType</td><td>N</td><td>Indicates order attribute type. Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>2</td><td>Liquidity provision activity order</td></tr></table>Required if NoOrderAttributes(2593) is greater than 0.</td></tr>
<tr><td>➡</td><td>2595</td><td>OrderAttributeValue</td><td>N</td><td>Indicates order attribute value. Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>Y</td><td>True</td></tr><tr><td>N</td><td>False (Default value)</td></tr></table>Required if NoOrderAttributes(2593) is greater than 0.</td></tr>
<tr><td>5999</td><td colspan="2">SpecialInstructions</td><td>N</td><td>Only used for XNET.</td></tr>
</table>

5.1.2. *Order Cancel/Replace Request (a.k.a. Order Modification Request)*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	G = OrderCancelReplaceRequest
Message Body			
37	OrderID	Y	Unique identifier of the order as assigned by the exchange.

<table>
<tr><td colspan="2">41</td><td>OrigClOrdID</td><td>Y</td><td>Client specified original identifier of the order. Max size is 16-char.</td></tr>
<tr><td colspan="2">11</td><td>ClOrdID</td><td>Y</td><td>Client specified identifier of the order. Max size is 16-char.</td></tr>
<tr><td colspan="2">18</td><td>ExecInst</td><td>N</td><td>Not required. Used only for order suspension/release from suspension.

Valid values are shown below.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>S</td><td>Suspend</td></tr>
<tr><td>q</td><td>Release from suspension (user defined value)</td></tr>
</table></td></tr>
<tr><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Class or source of the PartyID (448) value.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>D</td><td>D (Proprietary/Custom Code)
Used for the PartyRole values 1, 4, and 36</td></tr>
<tr><td>P</td><td>Short code identifier
Used for the PartyRole value 3, 12, 26 and 122</td></tr>
</table></td></tr>
<tr><td>➡</td><td>2376</td><td>PartyRoleQualifier</td><td>N</td><td>Qualifier of the specified PartyRole (452).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>22</td><td>Algorithm
Used for the PartyRole values 12 and 122</td></tr>
<tr><td>23</td><td>Firm or legal entity
Used for the PartyRole value 3</td></tr>
<tr><td>24</td><td>Natural person
Used for the PartyRole values 3, 12 and 122</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Executing Firm</td></tr><tr><td>3</td><td>Client ID
(MIFID II: Client identification code)</td></tr><tr><td>4</td><td>Clearing Firm</td></tr><tr><td>12</td><td>Executing trader
(MIFID II: Execution within firm)</td></tr><tr><td>26</td><td>Correspondent broker (MIFID II: Non-executing broker)</td></tr><tr><td>36</td><td>Entering trader (Trader ID)</td></tr><tr><td>122</td><td>Investment Decision Maker
(MIFID II: Investment decision within firm)</td></tr></table></td></tr>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">1</td><td>Account</td><td>N</td><td>Investor ID</td></tr>
<tr><td colspan="2">15</td><td>Currency</td><td>N</td><td>Used only for XNET</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Security identification</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID and StopSymbol(5521) if given.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>8</td><td>Exchange Symbol</td></tr><tr><td>A</td><td>Bloomberg Symbol</td></tr></table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
</table>

<table>
<tr><td>40</td><td>OrdType</td><td>N</td><td>Type of the order.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Market</td></tr><tr><td>7</td><td>Limit or Better</td></tr><tr><td>A</td><td>On Close</td></tr></table></td></tr>
<tr><td>59</td><td>TimeInForce</td><td>N</td><td><table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Day</td></tr><tr><td>1</td><td>Good Till Cancel (GTC)</td></tr><tr><td>2</td><td>At the opening</td></tr><tr><td>6</td><td>Good Till Date (GTD)</td></tr></table></td></tr>
<tr><td>432</td><td>ExpireDate</td><td>N</td><td>Date the order expires. Required if TimeInForce (59) is GTD (6)</td></tr>
<tr><td>58</td><td>Text</td><td>N</td><td></td></tr>
<tr><td>44</td><td>Price</td><td>N</td><td>Required for limit OrdTypes.</td></tr>
<tr><td>38</td><td>OrderQty</td><td>Y</td><td>This tag is used to specify the quantity of the order.</td></tr>
<tr><td>77</td><td>PositionEffect</td><td>N</td><td>For use in derivatives omnibus accounting.
Indicates whether the resulting position after a trade should be an opening position or closing position.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>O</td><td>Open</td></tr><tr><td>C</td><td>Close</td></tr></table></td></tr>
<tr><td>63</td><td>SettlType</td><td>N</td><td>Indicates order settlement period.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Normal</td></tr><tr><td>1</td><td>Immediate</td></tr></table></td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td>Absence of this field is interpreted as normal settlement period</td></tr>
<tr><td>210</td><td>MaxShow</td><td>N</td><td>Denotes the disclosed volume of the order. If not send, it is assumed equal to OrderQty(38).</td></tr>
<tr><td>54</td><td>Side</td><td>Y</td><td>Side of the order.
<table>
<tr><td>Value</td><td>Meaning</td></tr>
<tr><td>1</td><td>Buy</td></tr>
<tr><td>2</td><td>Sell</td></tr>
<tr><td>5</td><td>Sell short</td></tr>
<tr><td>R</td><td>Buy to cover</td></tr>
</table></td></tr>
<tr><td>60</td><td>TransactTime</td><td>Y</td><td>Time this order request was initiated.</td></tr>
<tr><td colspan="4"><b>Standard Trailer</b></td></tr>
</table>

The following table contains the extra and custom tags that can be sent in this message.

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td>5506</td><td>BoardID</td><td>N</td><td>Identifies board. Absence of this field is interpreted as the default value ‘M’ (the main board).</td></tr>
<tr><td>5561</td><td>MBListID</td><td>N</td><td>Not required. If given, it will be used for clearing purposes.</td></tr>
<tr><td>1724</td><td>OrderOrigination</td><td>N</td><td>Indicates order origination.Valid values are:
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Order is not submitted using Direct Electronic Access (DEA)</td></tr>
<tr><td>5</td><td>Order is submitted using Direct Electronic Access (DEA)</td></tr>
</table>
Absence of this field is interpreted as “order is not submitted using Direct Electronic Access (DEA)”.</td></tr>
<tr><td>5999</td><td>SpecialInstructions</td><td>N</td><td>Only used for XNET.</td></tr>
</table>

5.1.3. *Order Cancel Request*

<table>
<tr><th colspan="2">Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td colspan="5"><b>Standard Header</b></td></tr>
<tr><td colspan="2">35</td><td>MsgType</td><td>Y</td><td>F = OrderCancelRequest</td></tr>
<tr><td colspan="5"><b>Message Body</b></td></tr>
<tr><td colspan="2">37</td><td>OrderID</td><td>Y</td><td>Unique identifier of the order as assigned by the exchange.</td></tr>
<tr><td colspan="2">41</td><td>OrigClOrdID</td><td>Y</td><td>Client specified original identifier of the order. Max size is 16-char.</td></tr>
<tr><td colspan="2">11</td><td>ClOrdID</td><td>Y</td><td>Client specified identifier of the order. Max size is 16-char.</td></tr>
<tr><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Default value = D (Proprietary/Custom Code)</td></tr>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Executing Firm</td></tr>
<tr><td>36</td><td>Entering trader (Trader ID)</td></tr>
</table></td></tr>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">15</td><td>Currency</td><td>N</td><td>Used only for XNET</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Security identification</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>8</td><td>Exchange Symbol</td></tr>
<tr><td>A</td><td>Bloomberg Symbol</td></tr>
</table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
<tr><td colspan="2">58</td><td>Text</td><td>N</td><td></td></tr>
</table>

<table>
<tr><td>54</td><td>Side</td><td>Y</td><td>Side of the order.<table><tr><td>Value</td><td>Meaning</td></tr><tr><td>1</td><td>Buy</td></tr><tr><td>2</td><td>Sell</td></tr><tr><td>5</td><td>Sell short</td></tr><tr><td>R</td><td>Buy to cover</td></tr></table></td></tr>
<tr><td>60</td><td>TransactTime</td><td>Y</td><td>Time this order cancel request was initiated.</td></tr>
<tr><td colspan="4"><b>Standard Trailer</b></td></tr>
</table>

The following table contains the extra and custom tags that can be sent in this message.

Tag	**Field Name**	**Req**	**Description**
5561	MBListID	N	Not required. If given, it will be used for clearing purposes.

5.1.4. *Execution Report*

Tag	**Field Name**	**Req**	**Description**
Standard Header			
35	MsgType	Y	8 = ExecutionReport
Message Body			
37	OrderID	Y	Unique identifier of the order as assigned by the exchange.
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.
41	OrigClOrdID	N	ClOrdID of the previous accepted order (NOT the initial order of the day) when formatting or replacing an order. Max size is 16-char.
11	ClOrdID	N	Client specified identifier of the order. Max size is 16-char. In Quote/Combo related reports, this should be used to identify the original quote/combo order. (for this reason, the member should take care to utilize this field on input of quote/combo orders, and ensure its uniqueness)

<table>
<tr><td>17</td><td>ExecID</td><td>Y</td><td>Server specified unique identifier of the message.
If ExecType (150) is Trade, Trade Cancel or Trade Correct, this field is 8 characters long and contains the following:
• The first 6 characters contain the TradeNumber.
• The 7th character is equal to ExecType(150) field
• The 8th character is ‘S’ in case of sell side fill and ‘B’ in case of buy side fill</td></tr>
<tr><td>19</td><td>ExecRefId</td><td>N</td><td>Reference to the execution being cancelled/changed.
Required if ExecType (150) is Trade Cancel or Trade Correct.
In case of trade report related execution reports, ExecRefID refers to the last TradeReportID of the trade report.</td></tr>
<tr><td>150</td><td>ExecType</td><td>Y</td><td>Describes the purpose of the execution report.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>New</td></tr>
<tr><td>4</td><td>Canceled</td></tr>
<tr><td>5</td><td>Replace</td></tr>
<tr><td>8</td><td>Rejected</td></tr>
<tr><td>9</td><td>Suspended</td></tr>
<tr><td>C</td><td>Expired</td></tr>
<tr><td>D</td><td>Restated (unsolicited order cancellations/changes)</td></tr>
<tr><td>F</td><td>Trade</td></tr>
<tr><td>G</td><td>Trade Correct</td></tr>
<tr><td>H</td><td>Trade Cancel</td></tr>
</table></td></tr>
<tr><td>453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
</table>

<table>
<tr><th></th><th></th><th></th><th></th><th></th></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Class or source of the PartyID (448) value.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>D</td><td>D (Proprietary/Custom Code)
Used for the PartyRole values 1, 4, and 36</td></tr>
<tr><td>P</td><td>Short code identifier
Used for the PartyRole value 3, 12, 26 and 122</td></tr>
</table></td></tr>
<tr><td>➡</td><td>2376</td><td>PartyRoleQualifier</td><td>N</td><td>Qualifier of the specified PartyRole (452).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>22</td><td>Algorithm
Used for the PartyRole values 12 and 122</td></tr>
<tr><td>23</td><td>Firm or legal entity
Used for the PartyRole value 3</td></tr>
<tr><td>24</td><td>Natural person
Used for the PartyRole values 3, 12 and 122</td></tr>
</table></td></tr>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Executing Firm</td></tr>
<tr><td>3</td><td>Client ID
(MIFID II: Client identification code)</td></tr>
<tr><td>4</td><td>Clearing Firm</td></tr>
<tr><td>12</td><td>Executing trader
(MIFID II: Execution within firm)</td></tr>
<tr><td>26</td><td>Correspondent broker (MIFID II: Non-executing broker)</td></tr>
<tr><td>36</td><td>Entering trader (Trader ID)</td></tr>
<tr><td>122</td><td>Investment Decision Maker</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td></td><td><table><tr><td></td><td>(MIFID II: Investment decision within firm)</td></tr></table></td></tr>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">1</td><td>Account</td><td>Y</td><td>Investor ID</td></tr>
<tr><td colspan="2">15</td><td>Currency</td><td>N</td><td>Used only for XNET interface.</td></tr>
<tr><td colspan="2">32</td><td>LastQty</td><td>N</td><td>Quantity bought/sold on this (last) fill. Required if ExecType = Trade or Trade Correct.</td></tr>
<tr><td colspan="2">31</td><td>LastPx</td><td>N</td><td>Price of this (last) fill. Required if ExecType = Trade or Trade Correct.</td></tr>
<tr><td colspan="2">528</td><td>OrderCapacity</td><td>N</td><td>Indicates order capacity.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>A</td><td>Agency (AOTC)</td></tr><tr><td>P</td><td>Principal (DEAL)</td></tr><tr><td>R</td><td>Riskless principal (MTCH)</td></tr></table>Absence of this field is interpreted as agency</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Security identification</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID and StopSymbol(5521) if given.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>8</td><td>Exchange Symbol</td></tr><tr><td>A</td><td>Bloomberg Symbol</td></tr></table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
</table>

<table>
<tr><td>40</td><td>OrdType</td><td>Y</td><td>Type of the order.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Market</td></tr>
<tr><td>3</td><td>Stop</td></tr>
<tr><td>4</td><td>Stop Limit</td></tr>
<tr><td>7</td><td>Limit or Better</td></tr>
<tr><td>A</td><td>On Close</td></tr>
</table>
For Stop Orders (OrdType=3 or OrdType=4) ATHEX Gateway will add the tags StopSymbol(5521) with value the Stop SecuritySymbol or Stop Index of the order. Also ATHEX Gateway will set the value of tag StopSymbolType(5527) ‘S’ (Stop On Symbol) or ‘I’ (Stop On Index).
Not provided in OrderCancelRequest execution reports.</td></tr>
<tr><td>59</td><td>TimeInForce</td><td>N</td><td>
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Day</td></tr>
<tr><td>1</td><td>Good Till Cancel (GTC)</td></tr>
<tr><td>2</td><td>At the Opening</td></tr>
<tr><td>3</td><td>Immediate Or Cancel (IOC)</td></tr>
<tr><td>4</td><td>Fill Or Kill (FOK)</td></tr>
<tr><td>6</td><td>Good Till Date (GTD)</td></tr>
</table>
</td></tr>
<tr><td>432</td><td>ExpireDate</td><td>N</td><td>Date the order expires. Required if TimeInForce (59) is GTD (6)</td></tr>
</table>

<table>
<tr><td>39</td><td>OrdStatus</td><td>Y</td><td>In case that the execution report is an order entry/change/cancel confirmation, it indicates the status of the order.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>New</td></tr>
<tr><td>9</td><td>Suspended</td></tr>
<tr><td>1</td><td>Partially filled</td></tr>
<tr><td>2</td><td>Filled</td></tr>
<tr><td>4</td><td>Cancelled</td></tr>
<tr><td>C</td><td>Expired</td></tr>
<tr><td>N</td><td>Not released (user defined value)</td></tr>
<tr><td>I</td><td>Inactive (user defined value)</td></tr>
</table></td></tr>
<tr><td>378</td><td>ExecRestatement Reason</td><td>N</td><td>Required if ExecType = D (Restated).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>GT renewal/restatement</td></tr>
<tr><td>2</td><td>Verbal change</td></tr>
<tr><td>4</td><td>Broker Option</td></tr>
</table></td></tr>
<tr><td>58</td><td>Text</td><td>N</td><td></td></tr>
<tr><td>44</td><td>Price</td><td>Y</td><td>Not provided in OrderCancelRequest execution reports.</td></tr>
<tr><td>99</td><td>StopPx</td><td>N</td><td>Required if OrdType(40)=3 or OrdType(40)=4.</td></tr>
<tr><td>38</td><td>OrderQty</td><td>Y</td><td>This tag is used to specify the quantity of the order.</td></tr>
</table>

<table>
<tr><td>77</td><td>PositionEffect</td><td>N</td><td>For use in derivatives omnibus accounting.
Indicates whether the resulting position after a trade should be an opening position or closing position.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>O</td><td>Open</td></tr><tr><td>C</td><td>Close</td></tr></table></td></tr>
<tr><td>63</td><td>SettlType</td><td>N</td><td>Indicates order settlement period.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Normal</td></tr><tr><td>1</td><td>Immediate</td></tr></table>Absence of this field is interpreted as normal settlement period</td></tr>
<tr><td>210</td><td>MaxShow</td><td>N</td><td>Denotes the disclosed volume of the order. If not send, it is assumed equal to OrderQty(38). If equal to 0, then it denotes an inactive order</td></tr>
<tr><td>336</td><td>TradingSessionID</td><td>N</td><td>Security phase identification. Sent only in case of trade. Possible values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>O</td><td>Opening Trading Phase</td></tr><tr><td>T</td><td>Continuous Trading Phase</td></tr><tr><td>A</td><td>At the Closing Price trading Phase</td></tr><tr><td>C</td><td>Closing Price Continuous Trading Phase</td></tr></table></td></tr>
<tr><td>151</td><td>LeavesQty</td><td>Y</td><td></td></tr>
<tr><td>14</td><td>CumQty</td><td>Y</td><td></td></tr>
<tr><td>6</td><td>AvgPx</td><td>Y</td><td></td></tr>
</table>

<table>
<tr><td>54</td><td>Side</td><td>Y</td><td>Side of the order.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Buy</td></tr>
<tr><td>2</td><td>Sell</td></tr>
<tr><td>5</td><td>Sell short</td></tr>
<tr><td>R</td><td>Buy to cover</td></tr>
</table></td></tr>
<tr><td>851</td><td>LastLiquidityInd</td><td>N</td><td>Applicable only on OrdStatus of Partial or Filled and if BoardID(5506) = ‘M’ (Main board).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Added Liquidity</td></tr>
<tr><td>2</td><td>Removed Liquidity</td></tr>
<tr><td>4</td><td>Auction (user defined value)</td></tr>
</table></td></tr>
<tr><td>60</td><td>TransactTime</td><td>Y</td><td>Time this order request was initiated.</td></tr>
<tr><td>381</td><td>GrossTradeAmt</td><td>N</td><td>Notional amount</td></tr>
<tr><td colspan="4"><b>Standard Trailer</b></td></tr>
</table>

The following table contains the extra and custom tags that can be sent in this message.

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td>5501</td><td>OrigSource</td><td>N</td><td>A 1 character alphanumeric type indicating the source of the Order.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>C</td><td>CTCI –API</td></tr>
<tr><td>M</td><td>ORAMA-ETW</td></tr>
<tr><td>R</td><td>EMRW (ATHEX supervision application)</td></tr>
</table></td></tr>
<tr><td>5506</td><td>BoardID</td><td>N</td><td>Identifies board. Absence of this field is interpreted as the default value ‘M’ (the main board).</td></tr>
</table>

<table>
<tr><td>5509</td><td>OrderRelFlag</td><td>N</td><td>Possible values:
1: Normal
2: Quote
3: Combo</td></tr>
<tr><td>5512</td><td>GOIFlag</td><td>N</td><td>Group of investor flag (Not used)</td></tr>
<tr><td>5521</td><td>StopSymbol</td><td>N</td><td>Required if OrdType=3 or OrdType=4.
For stop orders, this is the instrument identification of stop symbol. If missing, tag 48 will be used.
For class or source of this field, check tag 22.</td></tr>
<tr><td>5522</td><td>SecurityStatus</td><td>N</td><td>Security status. Possible values are:
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>A</td><td>Active</td></tr>
<tr><td>N</td><td>Not active</td></tr>
<tr><td>S</td><td>Suspended</td></tr>
<tr><td>H</td><td>Halted</td></tr>
<tr><td>R</td><td>Resumed (Resumed Pre-opening of a Halt)</td></tr>
</table></td></tr>
<tr><td>5527</td><td>StopSymbolType</td><td>N</td><td>Possible values are ‘I’ for stop on index order and ‘S’ for stop on symbol orders. Absence of this tag means ‘S’ i.e. stop on symbol.</td></tr>
<tr><td>5529</td><td>ODLTradeType</td><td>N</td><td>Type of trade.
For possible values see section 5.6 (Custom Fix Tags)</td></tr>
<tr><td>5532</td><td>RejectReasonCode</td><td>N</td><td>Indicates to a member firm the reason that a requested action could not take place.</td></tr>
<tr><td>5545</td><td>CurrentCreditValue</td><td>N</td><td></td></tr>
<tr><td>5561</td><td>MBListID</td><td>N</td><td>Not required. If given, it will be used for clearing purposes.</td></tr>
</table>

<table>
<tr><td colspan="2">5508</td><td>CancelReasonCode</td><td>N</td><td>A 1 character alpha field used to indicate the reason for order cancellation.
Possible values are:
“C” canceled by CTCI request
“M” canceled by ETW request
“R” canceled by EMRW request
“I” cancellation of IOC order
“F” cancellation of FOK order
“P” cancellation of MKT order
“O” cancellation of ATO order
“L” cancellation of Life order
“S” cancellation of Covered Sale Order by CSD</td></tr>
<tr><td colspan="2">1724</td><td>OrderOrigination</td><td>N</td><td>Indicates order origination.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Order is not submitted using Direct Electronic Access (DEA)</td></tr><tr><td>5</td><td>Order is submitted using Direct Electronic Access (DEA)</td></tr></table>Absence of this field is interpreted as “order is not submitted using Direct Electronic Access (DEA)”.</td></tr>
<tr><td colspan="2">2593</td><td>NoOrderAttributes</td><td>N</td><td>Number of order attributes.</td></tr>
<tr><td>➡</td><td>2594</td><td>OrderAttributeType</td><td>N</td><td>Indicates order attribute type. Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>2</td><td>Liquidity provision activity order</td></tr></table>Required if NoOrderAttributes(2593) is greater than 0.</td></tr>
<tr><td>➡</td><td>2595</td><td>OrderAttributeValue</td><td>N</td><td>Indicates order attribute value. Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr></table></td></tr>
</table>

				Y True N False (Default value) Required if NoOrderAttributes(2593) is greater than 0.
645		MktBidPx	N	Current best bid price
646		MktOfferPx	N	Current best offer price
134		BidSize	N	Current best bid size
135		OfferSize	N	Current best offer size
2668		NoTrdRegPublications	N	Number of trade publication reasons.
➡	2669	TrdRegPublicationType	N	DefaultValue – 0 (Pre-trade transparency waiver) Required if NoTrdRegPublications(2668) > 0.
➡	2670	TrdRegPublicationReason	N	Possible values are: **Value** **Meaning** 0 No preceding order in book as transaction price set within average spread of a liquid instrument (NLIQ) 1 No preceding order in book as transaction price depends on system-set reference price for an illiquid Instrument (OLIQ) 2 No preceding order in book as transaction price is subject to conditions other than current market price (PRIC) 3 No public price preceding order as public reference price was used for matching orders (RFPT) 7 Deferral due to "Illiquid instrument" (ILQD)

			5	No public priceDeferral due to "Size specific" (SIZE)
5999	SpecialInstructions	N	Only used for XNET.	

5.1.5. *Order Cancel Reject*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	9 = OrderCancelReject
Message Body			
37	OrderID	Y	Unique identifier of the order as assigned by the exchange.
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.
41	OrigClOrdID	Y	ClOrdID of the previous accepted order (NOT the initial order of the day) when formatting or replacing an order. Max size is 16-char.
11	ClOrdID		Client specified identifier of the order. Max size is 16-char.
39	OrdStatus	Y	
1	Account	N	Investor ID
434	CxlRejResponseTo	Y	‘1’ = Order Cancel Request ‘2’ = Order Cancel/Replace Request
102	CxlRejReason	N	
58	Text	N	
60	TransactTime	Y	Time this order request was initiated.
Standard Trailer			

The following table contains the extra and custom tags that can be sent in this message.

Tag	Field Name	Req	Description
5561	MBListID	N	Not required. If given, it will be used for clearing purposes.

5.2. Quotation Negotiation

5.2.1. *Quote*

<table>
<tr><th colspan="2">Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td colspan="5"><b>Standard Header</b></td></tr>
<tr><td colspan="2">35</td><td>MsgType</td><td>Y</td><td>S = Quote</td></tr>
<tr><td colspan="5"><b>Message Body</b></td></tr>
<tr><td colspan="2">117</td><td>QuoteID</td><td>Y</td><td>Identifier of the quote entry.</td></tr>
<tr><td colspan="2">15</td><td>Currency</td><td>N</td><td>Not used.</td></tr>
<tr><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Class or source of the PartyID (448) value.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>D</td><td>D (Proprietary/Custom Code)
Used for the PartyRole values 1, 4, and 36</td></tr>
<tr><td>P</td><td>Short code identifier
Used for the PartyRole value 3, 12, 26 and 122</td></tr>
</table></td></tr>
<tr><td>➡</td><td>2376</td><td>PartyRoleQualifier</td><td>N</td><td>Qualifier of the specified PartyRole (452).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>22</td><td>Algorithm
Used for the PartyRole values 12 and 122</td></tr>
<tr><td>23</td><td>Firm or legal entity
Used for the PartyRole value 3</td></tr>
<tr><td>24</td><td>Natural person
Used for the PartyRole values 3, 12 and 122</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Executing Firm</td></tr><tr><td>3</td><td>Client ID
(MIFID II: Client identification code)</td></tr><tr><td>4</td><td>Clearing Firm</td></tr><tr><td>12</td><td>Executing trader
(MIFID II: Execution within firm)</td></tr><tr><td>26</td><td>Correspondent broker (MIFID II: Non-executing broker)</td></tr><tr><td>36</td><td>Entering trader (Trader ID)</td></tr><tr><td>122</td><td>Investment Decision Maker
(MIFID II: Investment decision within firm)</td></tr></table></td></tr>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">1</td><td>Account</td><td>Y</td><td>Investor ID</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Security identification</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>8</td><td>Exchange Symbol</td></tr><tr><td>A</td><td>Bloomberg Symbol</td></tr></table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
<tr><td colspan="2">58</td><td>Text</td><td>N</td><td></td></tr>
<tr><td colspan="2">54</td><td>Side</td><td>N</td><td>Side of quote..<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Buy</td></tr><tr><td>2</td><td>Sell</td></tr></table>Absence of this field is interpreted as both sides.</td></tr>
</table>

132	BidPx	N	Bid price. Required if BidSize (134) is specified.
134	BidSize	N	Bid quantity. Required if BidPx (132) is specified.
133	OfferPX	N	Offer price. Required if OfferSize (135) is specified.
135	OfferSize	N	Offer quantity. Required if OfferPx (133) is specified.
Standard Trailer			

The following table contains the extra and custom tags that can be sent in this message.

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td>1166</td><td>QuoteMsgID</td><td>Y</td><td>Client specified identifier of the quote. User-defined tag from FIX 5.0 SP2.</td></tr>
<tr><td>1724</td><td>OrderOrigination</td><td>N</td><td>Indicates order origination.Valid values are:
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Order is not submitted using Direct Electronic Access (DEA)</td></tr>
<tr><td>5</td><td>Order is submitted using Direct Electronic Access (DEA)</td></tr>
</table>
Absence of this field is interpreted as “order is not submitted using Direct Electronic Access (DEA)”.</td></tr>
</table>

5.2.2. *QuoteCancel*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	Z = QuoteCancel
Message Body			
117	QuoteID	Y	Identifier of the quote entry.
15	Currency	N	Not used.

<table>
<tr><td colspan="2">298</td><td>QuoteCancelType</td><td>Y</td><td>Type of quote cancel request<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Cancel for Instrument</td></tr></table></td></tr>
<tr><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers. Should be “2”.</td></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Default value = D (Proprietary/Custom Code)</td></tr>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Executing Firm</td></tr><tr><td>36</td><td>Entering trader (Trader ID)</td></tr></table></td></tr>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">1</td><td>Account</td><td>Y</td><td>Investor ID</td></tr>
<tr><td colspan="2">295</td><td>NoQuoteEntries</td><td>Y</td><td>Specifies the number of quote entries specified. The value is this field should always be “1”.</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Security identification</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>8</td><td>Exchange Symbol</td></tr><tr><td>A</td><td>Bloomberg Symbol</td></tr></table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
<tr><td colspan="5"><b>Standard Trailer</b></td></tr>
</table>

The following table contains the extra and custom tags that can be sent in this message.

Tag	Field Name	Req	Description
1166	QuoteMsgID	Y	Client specified identifier of the quote. User-defined tag from FIX 5.0 SP2.
58	Text	N	

5.2.3. *QuoteStatusReport*

<table>
<tr><th colspan="2">Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td colspan="5"><b>Standard Header</b></td></tr>
<tr><td colspan="2">35</td><td>MsgType</td><td>Y</td><td>AI = QuoteStatusReport</td></tr>
<tr><td colspan="5"><b>Message Body</b></td></tr>
<tr><td colspan="2">117</td><td>QuoteID</td><td>Y</td><td>Identifier of the quote entry.</td></tr>
<tr><td colspan="2">297</td><td>QuoteStatus</td><td>Y</td><td>Identifies the status of the quote.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Accepted</td></tr>
<tr><td>5</td><td>Rejected</td></tr>
<tr><td>6</td><td>Removed from Market</td></tr>
<tr><td>17</td><td>Cancelled (user defined value)</td></tr>
</table></td></tr>
<tr><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Class or source of the PartyID (448) value.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>D</td><td>D (Proprietary/Custom Code)
Used for the PartyRole values 1, 4, and 36</td></tr>
<tr><td>P</td><td>Short code identifier
Used for the PartyRole value 3, 12, 26 and 122</td></tr>
</table></td></tr>
<tr><td>➡</td><td>2376</td><td>PartyRoleQualifier</td><td>N</td><td>Qualifier of the specified PartyRole (452).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>22</td><td>Algorithm
Used for the PartyRole values 12 and 122</td></tr>
<tr><td>23</td><td>Firm or legal entity
Used for the PartyRole value 3</td></tr>
<tr><td>24</td><td>Natural person</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td></td><td><table><tr><td></td><td>Used for the PartyRole values 3, 12 and 122</td></tr></table></td></tr>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Executing Firm</td></tr>
<tr><td>3</td><td>Client ID
(MIFID II: Client identification code)</td></tr>
<tr><td>4</td><td>Clearing Firm</td></tr>
<tr><td>12</td><td>Executing trader
(MIFID II: Execution within firm)</td></tr>
<tr><td>26</td><td>Correspondent broker (MIFID II: Non-executing broker)</td></tr>
<tr><td>36</td><td>Entering trader (Trader ID)</td></tr>
<tr><td>122</td><td>Investment Decision Maker
(MIFID II: Investment decision within firm)</td></tr>
</table></td></tr>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">1</td><td>Account</td><td>N</td><td>Investor ID</td></tr>
<tr><td colspan="2">58</td><td>Text</td><td>N</td><td></td></tr>
<tr><td colspan="2">54</td><td>Side</td><td>N</td><td>Side of quote.<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Buy</td></tr>
<tr><td>2</td><td>Sell</td></tr>
</table>Absence of this field is interpreted as both sides.</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Security identification</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID.<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>8</td><td>Exchange Symbol</td></tr>
<tr><td>A</td><td>Bloomberg Symbol</td></tr>
</table></td></tr>
</table>

➡	207	SecurityExchange	Y	Values are given by ATHEX. Max length is 4 characters
132		BidPx	N	Bid price. Required if BidSize (134) is specified.
134		BidSize	N	Bid quantity. Required if BidPx (132) is specified.
133		OfferPX	N	Offer price. Required if OfferSize (135) is specified.
135		OfferSize	N	Offer quantity. Required if OfferPx (133) is specified.
60		TransactTime	Y	Time this order request was initiated.
Standard Trailer				

The following table contains the extra and custom tags that can be sent in this message.

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td>1166</td><td>QuoteMsgID</td><td>Y</td><td>Client specified identifier of the quote. User-defined tag from FIX 5.0 SP2.</td></tr>
<tr><td>5532</td><td>RejectReasonCode</td><td>N</td><td>Indicates to a member firm the reason that a requested action could not take place.
Required if QuoteStatus(297) = Rejected (5).</td></tr>
<tr><td>198</td><td>SecondaryOrderID</td><td>N</td><td>ODL message id. Used in recovery mechanism.</td></tr>
<tr><td>1724</td><td>OrderOrigination</td><td>N</td><td>Indicates order origination.Valid values are:
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Order is not submitted using Direct Electronic Access (DEA)</td></tr>
<tr><td>5</td><td>Order is submitted using Direct Electronic Access (DEA)</td></tr>
</table>
Absence of this field is interpreted as “order is not submitted using Direct Electronic Access (DEA)”.</td></tr>
<tr><td>5545</td><td>CurrentCreditValue</td><td>N</td><td></td></tr>
</table>

5.2.4. *QuoteRequest (ODL MJ and ODL TJ/TP/TK in case of quote request acceptance)*

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td colspan="4"><b>Standard Header</b></td></tr>
<tr><td>35</td><td>MsgType</td><td>Y</td><td>R = QuoteRequest</td></tr>
<tr><td colspan="4"><b>Message Body</b></td></tr>
<tr><td>131</td><td>QuoteReqID</td><td>Y</td><td>Client specified identifier for quote request.</td></tr>
<tr><td><b>Component</b></td><td><b>QuoteReqGrp</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers. Should be 2.</td></tr>
<tr><td>➡ 447</td><td>PartyIDSource</td><td>Y</td><td>Default value = D (Proprietary/Custom Code)</td></tr>
<tr><td>➡ 452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Executing Firm</td></tr><tr><td>36</td><td>Entering trader (Trader ID)</td></tr></table></td></tr>
<tr><td>➡ 448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td>146</td><td>NoRelatedSym</td><td>Y</td><td>Number of instruments for which quotes are to be requested. The value in this field should always be “1”.</td></tr>
<tr><td><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡ 48</td><td>SecurityID</td><td>Y</td><td></td></tr>
<tr><td>➡ 22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>8</td><td>Exchange Symbol</td></tr><tr><td>A</td><td>Bloomberg Symbol</td></tr></table></td></tr>
<tr><td>➡ 454</td><td>NoSecurityAltID</td><td>N</td><td>Number of SecurityAltID entruies. Default value: 1</td></tr>
<tr><td>➡ ➡ 455</td><td>SecurityAltID</td><td>N</td><td></td></tr>
</table>

➡	➡	456	SecurityAltIDSource	N	Identifies class or source of the SecurityAltID **Value** **Meaning** 8 Exchange Symbol A Bloomberg Symbol
➡	54		Side	N	Side of requested quote. **Value** **Meaning** 1 Buy 2 Sell Absence of this field is interpreted as both sides.
➡	207		SecurityExchange	Y	Values are given by ATHEX. Max length is 4 characters
➡	38		OrderQty	N	Quantity ordered
60			TransactTime	N	Time this quote request was accepted.
Standard Trailer					

The following table contains the extra and custom tags that can be sent in this message.

Tag	Field Name	Req	Description
5574	ODLMsgType	Y	ODLMsgType. Possible values are: “TJ” “TP” “TK”
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.

* The QuoteRequest message issued by a tading firm is sent to the OASIS and then to all eligible market makers. Also, the same message is sent back to the sending firm as a confirmation.

5.2.5. *QuoteRequestReject (ODL TJ/TP Messages in case of rejection)*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	AG = QuoteRequestReject

<table>
<tr><th colspan="5">Message Body</th></tr>
<tr><td colspan="2">131</td><td>QuoteReqID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">658</td><td>QuoteRequestRejectReason</td><td>Y</td><td>Reason Quote Request was rejected.
Default value = 99 (Other)</td></tr>
<tr><td colspan="2">146</td><td>NoRelatedSym</td><td>Y</td><td>Specifies the number of repeating symbols specified. The value is this field will always be “1”.</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td></td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>8</td><td>Exchange Symbol</td></tr><tr><td>A</td><td>Bloomberg Symbol</td></tr></table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX.</td></tr>
<tr><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Default value = D (Proprietary/Custom Code)</td></tr>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Executing Firm</td></tr><tr><td>36</td><td>Entering trader (Trader ID)</td></tr></table></td></tr>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">60</td><td>TransactTime</td><td>Y</td><td>Time this quote request was rejected.</td></tr>
<tr><th colspan="5">Standard Trailer</th></tr>
</table>

The following table contains the extra and custom tags that can be sent in this message.

Tag	Field Name	Req	Description
5532	RejectReasonCode	N	Indicates to a member firm the reason that a requested action could not take place.
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.

5574	ODLMsgType	Y	ODLMsgType. Possible values are: “TJ” “TP”

5.2.6. *News (MktMaker ALARM) (ODL TM Message)*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	B = News
Message Body			
42	OrigTime	N	Time of message origination.
148	Headline	Y	Reason for notification. Valid values are: **Value** **Meaning** W Quote Warning A Quote Alarm
146	NoRelatedSym	Y	Specifies the number of repeating symbols specified. The value is this field will always be “1”.
Component	**Instrument**	**Y**	
➡ 48	SecurityID	Y	Exchange Symbol
➡ 22	SecurityIDSource	Y	Identifies class or source of the SecurityID Default value: 8 (Exchange Symbol)
➡ 454	NoSecurityAltID	N	Number of SecurityAltID entruies. Default value: 1
➡ ➡ 455	SecurityAltID	N	Bloomberg Symbol
➡ ➡ 456	SecurityAltIDSource	N	Identifies class or source of the SecurityAltID Default value: A (Bloomberg Symbol)

➡ 207	SecurityExchange	Y	Values are given by ATHEX. Max length is 4 characters
33	NoLinesOfText	Y	Number of lines of text (always 1)
58	Text	Y	Message note. In this case, it shall be the meaning of the AlarmReason
Standard Trailer			

The following table contains the extra and custom tags that can be sent in this message.

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td>5562</td><td>AlarmReason</td><td>N</td><td>Used only in case of Headline(148) = “A” (Quote Alarm)
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Prices are missing</td></tr>
<tr><td>2</td><td>BID price is missing</td></tr>
<tr><td>3</td><td>ASK price is missing</td></tr>
<tr><td>4</td><td>The Price Spread is too big</td></tr>
<tr><td>5</td><td>Quantities are too small</td></tr>
<tr><td>6</td><td>BID Quantity is too small</td></tr>
<tr><td>7</td><td>ASK Quantity is too small</td></tr>
</table>
In case this field is given, the meaning text field is also displayed in Text(58) field.</td></tr>
<tr><td>198</td><td>SecondaryOrderID</td><td>N</td><td>ODL message id. Used in recovery mechanism.</td></tr>
</table>

5.2.7. *News (ODL TN Message)*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	B = News
Message Body			
42	OrigTime	N	Time of message origination.

<table>
<tr><td colspan="3">148</td><td>Headline</td><td>Y</td><td>Reason for notification. Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>S</td><td>Suspend quotation responsibility</td></tr><tr><td>R</td><td>Resume quotation responsibility</td></tr></table></td></tr>
<tr><td colspan="3">146</td><td>NoRelatedSym</td><td>Y</td><td>Specifies the number of repeating symbols specified. The value is this field will always be “1”.</td></tr>
<tr><td colspan="3"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td colspan="2">48</td><td>SecurityID</td><td>Y</td><td>Exchange Symbol</td></tr>
<tr><td>➡</td><td colspan="2">22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID
Default value: 8 (Exchange Symbol)</td></tr>
<tr><td>➡</td><td colspan="2">454</td><td>NoSecurityAltID</td><td>N</td><td>Number of SecurityAltID entruies. Default value: 1</td></tr>
<tr><td>➡</td><td>➡</td><td>455</td><td>SecurityAltID</td><td>N</td><td>Bloomberg Symbol</td></tr>
<tr><td>➡</td><td>➡</td><td>456</td><td>SecurityAltIDSource</td><td>N</td><td>Identifies class or source of the SecurityAltID.
Default value: A (Bloomberg Symbol)</td></tr>
<tr><td>➡</td><td colspan="2">207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
<tr><td colspan="3">33</td><td>NoLinesOfText</td><td>Y</td><td>Number of lines of text (always 1)</td></tr>
<tr><td colspan="3">58</td><td>Text</td><td>Y</td><td>Message note. In this case, it shall be the meaning of the AlarmReason</td></tr>
<tr><td colspan="6"><b>Standard Trailer</b></td></tr>
</table>

The following table contains the extra and custom tags that can be sent in this message.

Tag	Field Name	Req	Description

<table>
<tr><td>5563</td><td>QuotationResponsibility Level</td><td>N</td><td>Identifies the quotation responsibility level
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>M</td><td>Market</td></tr>
<tr><td>I</td><td>Instrument</td></tr>
<tr><td>F</td><td>Firm</td></tr>
<tr><td>U</td><td>Underlying</td></tr>
</table></td></tr>
<tr><td>5564</td><td>ReasonOfSuspension</td><td></td><td>Identifies the reason for quote responsibility suspension. Sent only if Headline(148)=’S’ (Suspend quotation responsibility)
1 = Manual suspension
2 = Suspension because of auction
3 = Suspension because of limit up
4 = Suspension because of limit down
5 = Minimum daily order volume has been covered</td></tr>
<tr><td>198</td><td>SecondaryOrderID</td><td>N</td><td>ODL message id. Used in recovery mechanism.</td></tr>
</table>

5.3. Trade Reports

5.3.1. *Trade Capture Report (TCR) (ODL MI/TF conditionally)*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	AE = Trade Capture Report
Message Body			
571	TradeReportID	Y	Firm’s identifier. Entered by member in incoming TCR message and echoed by trading system in confirmation messages.
880	TrdMatchID	N	Identifier assigned to a trade by a matching system.
572	TradeReportRefID	N	Used in edit transactions to refer to previous TradeReportID.

<table>
<tr><td colspan="2">487</td><td>TradeReportTrans Type</td><td>Y</td><td>Type of transaction being reported.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>New</td></tr><tr><td>1</td><td>Cancel</td></tr><tr><td>2</td><td>Replace</td></tr></table></td></tr>
<tr><td colspan="2">856</td><td>TradeReportType</td><td>Y</td><td>Type of trade report.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Submit</td></tr><tr><td>1</td><td>Alleged</td></tr><tr><td>2</td><td>Accept</td></tr><tr><td>3</td><td>Decline</td></tr><tr><td>5</td><td>No/Was (i.e. Expired)</td></tr><tr><td>6</td><td>Trade Report Cancel</td></tr></table></td></tr>
<tr><td colspan="2">32</td><td>LastQty</td><td>N</td><td>Traded quantity. Required for all messages published from the system and trade report submissions.
The system will not validate the LastQty submitted for a trade report cancellation with the original values.
It is not possible to submit trades with zero or negative value for LastQty(32).</td></tr>
<tr><td colspan="2">31</td><td>LastPx</td><td>N</td><td>Traded price specified in the instrument’s trading currency. Required for all messages published from the system and trade report submissions.
It is not possible to submit trades with negative value for LastPx(31)</td></tr>
<tr><td colspan="2">15</td><td>Currency</td><td>N</td><td>XNET only.</td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Exchange Symbol</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>8</td><td>Exchange Symbol</td></tr><tr><td>A</td><td>Bloomberg Symbol</td></tr></table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
</table>

<table>
<tr><td colspan="2">573</td><td>MatchStatus</td><td>N</td><td>Will be populated in all messages from the system.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Matched</td></tr><tr><td>1</td><td>Unmatched</td></tr></table></td></tr>
<tr><td colspan="2">552</td><td>NoSides</td><td>Y</td><td>The number of sides in the Trade Capture Report.<table><tr><th>Value</th><th>Case</th></tr><tr><td>1</td><td>System’s or modifier’s pree agreed or regular trade cancellation</td></tr><tr><td>2</td><td>Pre agreed Initiator</td></tr></table></td></tr>
<tr><td>➡</td><td>54</td><td>Side</td><td>Y</td><td>Side of the order.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Buy</td></tr><tr><td>2</td><td>Sell</td></tr><tr><td>5</td><td>Sell short</td></tr><tr><td>R</td><td>Buy to cover</td></tr></table></td></tr>
<tr><td>➡</td><td>1</td><td>Account</td><td>Y</td><td>Investor ID</td></tr>
<tr><td>➡</td><td>1724</td><td>OrderOrigination</td><td>N</td><td>Indicates order origination.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Order is not submitted using Direct Electronic Access (DEA)</td></tr><tr><td>5</td><td>Order is submitted using Direct Electronic Access (DEA)</td></tr></table>Absence of this field is interpreted as “order is not submitted using Direct Electronic Access (DEA)”.</td></tr>
</table>

<table>
<tr><td>➡</td><td colspan="2">528</td><td>OrderCapacity</td><td>N</td><td>Indicates order capacity.Valid values are:
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>A</td><td>Agency (AOTC)</td></tr>
<tr><td>P</td><td>Principal (DEAL)</td></tr>
<tr><td>R</td><td>Riskless principal (MTCH)</td></tr>
</table>
Absence of this field is interpreted as agency.</td></tr>
<tr><td>➡</td><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.
For buy-side party and for two members’ pre agreed trade this number will be 5.
For all other cases will be 7. (Sell side pre agreed, buy side intrafirm pre agreed, regular trade cancellation</td></tr>
<tr><td>➡</td><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td>Identifier of the party.</td></tr>
<tr><td>➡</td><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Class or source of the PartyID (448) value.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>D</td><td>D (Proprietary/Custom Code)
Used for the PartyRole values 1, 4, 12, 17, 18, 36, 37, 39 and 124</td></tr>
<tr><td>P</td><td>Short code identifier
Used for the PartyRole value 3, 26, 122, 123, 125 and 126</td></tr>
</table></td></tr>
<tr><td>➡</td><td>➡</td><td>2376</td><td>PartyRoleQualifier</td><td>N</td><td>Qualifier of the specified PartyRole (452).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>22</td><td>Algorithm
Used for the PartyRole values 12, 37, 122 and 125</td></tr>
<tr><td>23</td><td>Firm or legal entity
Used for the PartyRole values 3 and 126</td></tr>
<tr><td>24</td><td>Natural person
Used for the PartyRole values 3, 12, 37, 122, 125 and 126</td></tr>
</table></td></tr>
</table>

<table>
<tr>
<td>➡</td>
<td>➡</td>
<td>452</td>
<td>PartyRole</td>
<td>Y</td>
<td>Role of the specified PartyID (448).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Executing Firm</td></tr>
<tr><td>3</td><td>Client ID
(MIFID II: Client identification code)</td></tr>
<tr><td>4</td><td>Clearing Firm</td></tr>
<tr><td>12</td><td>Executing trader
(MIFID II: Execution within firm)</td></tr>
<tr><td>26</td><td>Correspondent broker (MIFID II: Non-executing broker)</td></tr>
<tr><td>36</td><td>Entering trader (Trader ID)</td></tr>
<tr><td>122</td><td>Investment Decision Maker
(MIFID II: Investment decision within firm)</td></tr>
<tr><td>17</td><td>Contra Executing firm</td></tr>
<tr><td>126</td><td>Contra Client ID</td></tr>
<tr><td>18</td><td>Contra Clearing Firm</td></tr>
<tr><td>37</td><td>Contra Executing trader
(MIFID II: Execution within firm)</td></tr>
<tr><td>39</td><td>Contra Investor ID</td></tr>
<tr><td>123</td><td>Contra Correspondent broker
(MIFID II: Contra non-executing broker)</td></tr>
<tr><td>124</td><td>Contra Entering trader (Contra Trader ID)</td></tr>
<tr><td>125</td><td>Contra Investment Decision Maker
(MIFID II: Contra investment decision within firm)</td></tr>
</table>
</td>
</tr>
</table>

➡	77	PositionEffect	N	For use in derivatives omnibus accounting. Indicates whether the resulting position after a trade should be an opening position or closing position.Valid values are: Value Meaning O Open C Close
	198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.
	570	PreviouslyReported	Y	Indicates if the trade capture report was previously reported to the counterparty.Valid values are: Value Meaning N Not reported to the counterparty Y Previously reported to the counterparty (“DUPL”)
	60	TransactTime	N	Time the trade, cancellation or correction occurred. Required for messages sent by the server
Standard Trailer				

The following table contains the extra and custom tags that can be sent in this message.

Tag	Field Name	Req	Description
5506	BoardID	N	Identifies board. Absence of this field is interpreted as the default value ‘M’ (the main board).
5529	ODLTradeType	Y	Trade Report Type. For possible values see section 5.6 (Custom Fix Tags)
5545	CurrentCreditValue	N	
1839	TradePriceCondition	N	Indicates if the trade capture report is a special dividend transaction. Valid values are: Value Meaning

				13 – Special dividend transaction (SDIV) Absence of this field is interpreted as no special dividend transaction
	2668	NoTrdRegPublications	N	Number of trade publication reasons.
➡	2669	TrdRegPublicationType	N	DefaultValue – 1 (Post-trade transparency waiver) Required if NoTrdRegPublications(2668) > 0.
➡	2670	TrdRegPublicationReason	N	Possible values are: **Value** – **Meaning** 6 – Deferral due to "Large in scale" (LRGS)

5.3.2. *Trade Capture Report Ack (TCRA) (ODL TF/TR conditionally)*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	AR = Trade Capture Report Ack
Message Body			
571	TradeReportID	Y	Unique Identifier of the TCR
880	TrdMatchID	Y	Identifier assigned to a trade by a matching system.
572	TradeReportRefID	N	Firm’s identifier. Entered by member in incoming TCR message and echoed by trading system in confirmation messages.
856	TradeReportType	Y	Value submitted with the trade report.

<table>
<tr><td colspan="2">150</td><td>ExecType</td><td>Y</td><td>Describes the purpose of the trade capture report.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>New</td></tr><tr><td>4</td><td>Canceled</td></tr><tr><td>5</td><td>Replace</td></tr><tr><td>C</td><td>Expired</td></tr></table></td></tr>
<tr><td colspan="2">573</td><td>MatchStatus</td><td>N</td><td>Will be populated in all messages from the system.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Matched</td></tr><tr><td>1</td><td>Unmatched</td></tr></table></td></tr>
<tr><td colspan="2"><b>Component</b></td><td><b>Instrument</b></td><td><b>Y</b></td><td></td></tr>
<tr><td>➡</td><td>48</td><td>SecurityID</td><td>Y</td><td>Exchange Symbol</td></tr>
<tr><td>➡</td><td>22</td><td>SecurityIDSource</td><td>Y</td><td>Identifies class or source of the SecurityID.<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>8</td><td>Exchange Symbol</td></tr><tr><td>A</td><td>Bloomberg Symbol</td></tr></table></td></tr>
<tr><td>➡</td><td>207</td><td>SecurityExchange</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
<tr><td colspan="2">58</td><td>Text</td><td>N</td><td>Text specifying the reason for the rejection.</td></tr>
<tr><td colspan="2">32</td><td>LastQty</td><td>N</td><td>Value submitted with the trade report.</td></tr>
<tr><td colspan="2">31</td><td>LastPx</td><td>N</td><td>Value submitted with the trade report.</td></tr>
</table>

<table>
<tr><td>77</td><td>PositionEffect</td><td>N</td><td>For use in derivatives omnibus accounting.
Indicates whether the resulting position after a trade should be an opening position or closing position.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>O</td><td>Open</td></tr><tr><td>C</td><td>Close</td></tr></table></td></tr>
<tr><td>198</td><td>SecondaryOrderID</td><td>N</td><td>ODL message id. Used in recovery mechanism.</td></tr>
<tr><td>528</td><td>OrderCapacity</td><td>N</td><td>Indicates order capacity.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>A</td><td>Agency (AOTC)</td></tr><tr><td>P</td><td>Principal (DEAL)</td></tr><tr><td>R</td><td>Riskless principal (MTCH)</td></tr></table>Absence of this field is interpreted as agency</td></tr>
<tr><td>60</td><td>TransactTime</td><td>N</td><td>This will be the time stamped by the server.</td></tr>
<tr><td colspan="4"><b>Standard Trailer</b></td></tr>
</table>

The following table contains the extra and custom tags that can be sent in this message.

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td>5532</td><td>RejectReasonCode</td><td>N</td><td>Code further specifying the reason for rejection. Used if TrdRptStatus (939) is Rejected (1).</td></tr>
<tr><td>5506</td><td>BoardID</td><td>N</td><td>Identifies board. Absence of this field is interpreted as the default value ‘M’ (the main board).</td></tr>
<tr><td>5529</td><td>ODLTradeType</td><td>Y</td><td>Trade Report Type. For possible values see section 5.6 (Custom Fix Tags)</td></tr>
<tr><td>5545</td><td>CurrentCreditValue</td><td>N</td><td></td></tr>
<tr><td>1724</td><td>OrderOrigination</td><td>N</td><td>Indicates order origination.Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Order is not submitted using Direct Electronic Access (DEA)</td></tr></table></td></tr>
</table>

			5 Order is submitted using Direct Electronic Access (DEA) Absence of this field is interpreted as “order is not submitted using Direct Electronic Access (DEA)”.

5.4. Market Data Messages

5.4.1. *SecurityStatus (ODL CA Message)*

Tag	Field Name	Req	Description
	Standard Header	Y	MsgType = f
Component	**Instrument**	**Y**	
➡ 48	SecurityID	Y	Exchange Symbol
➡ 22	SecurityIDSource	Y	Identifies class or source of the SecurityID. **Value** **Meaning** 8 Exchange Symbol A Bloomberg Symbol
➡ 207	SecurityExchange	Y	Values are given by ATHEX. Max length is 4 characters
Component	**SecAltIDGroup**	**N**	
➡ 454	NoSecurityAltID	N	Number of SecurityAltID entruies. Default value: 1
➡ ➡ 455	SecurityAltID	N	Bloomberg Symbol
➡ ➡ 456	SecurityAltIDSource	N	Identifies class or source of the SecurityAltID Default value: A (Bloomberg Symbol)
60	TransactTime	N	This will be the time stamped by the server.
	Standard Trailer	Y	

The following table contains the custom tags that can be received in this message.

Tag	Field Name	Req	Description
5574	ODLMsgType	Y	ODLMsgType=CA See Section 5.6
5522	SecurityStatus	Y	See Section 5.6
5511	PhaseID	Y	See Section 5.6
5530	SecurityPrice	Y	See Section 5.60
5531	ODLHaltReason	Y	See Section 5.6
5502	MarketID	Y	See Section 5.6
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.

5.4.2. *Security Prices (ODL CD Message)*

Tag	Field Name	Req	Description
	Standard Header	**Y**	MsgType = f
Component	**Instrument**	**Y**	
48	SecurityID	Y	
22	SecurityIDSource	Y	Identifies class or source of the SecurityID Default value: 8 (Exchange Symbol)
207	SecurityExchange	Y	
Component	**SecAltIDGroup**	**N**	
➡ 454	NoSecurityAltID	N	Number of SecurityAltID entruies. Default value: 1
➡ ➡ 455	SecurityAltID	N	Bloomberg Symbol
➡ ➡ 456	SecurityAltIDSource	N	Identifies class or source of the SecurityAltID Default value: A (Bloomberg Symbol)
333	LowPx	Y	Indicates the minimum possible price of the instrument.

332	HighPx	Y	Indicates the maximum possible price of the instrument
	Standard Trailer	**Y**	

The following table contains the custom tags that can be received in this message.

Tag	Field Name	Req	Description
5574	ODLMsgType	Y	ODLMsgType=CD See Section 5.6
5530	SecurityPrice	Y	See Section 5.6
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.
159	AccruedInterestAmt	N	Accrued Interest

5.4.3. *Trading Session Status (ODL CC Message)*

Tag	Field Name	Req	Description
	Standard Header	**Y**	MsgType = h
336	TradingSessionID	Y	A 1 character alphanumeric field indicating the market/board status. Possible values : For the main board: • 'P' : Pre-Call • 'J' : Calculated projected opening price • 'T' : Continuous/Auction event • 'C' : Closing price trading • 'R' : Run-off • 'E' : End of trading • 'H' : Halt • 'S' : Stop (Used only in Auction Market) • 'N' : No Orders accepted until the next Status change (Used only for XNET interface) For the other boards:

			- 'O' : Open - 'E' : End
340	TradSesStatus	Y	Valid values are: 1 = Halted 2 = Open 3 = Closed 4 = Pre-Open 5 = Pre-Close
	Standard Trailer	**Y**	

The following table contains the custom tags that can be received in this message.

Tag	Field Name	Req	Description
5574	ODLMsgType	Y	ODLMsgType=CC
5502	MarketID	Y	See Section 0
5506	BoardID	Y	See Section 0
207	SecurityExchange	Y	
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.

5.4.4. *Credit Limit Info (ODL TL Message)*

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td></td><td><b><i>Standard Header</i></b></td><td><b><i>Y</i></b></td><td>B = News</td></tr>
<tr><td>148</td><td>Headline</td><td>Y</td><td>Reason for notification. Valid values are:
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>C</td><td>Credit Limit Information</td></tr>
</table></td></tr>
<tr><td>33</td><td>NoLinesOfText</td><td>Y</td><td>Number of lines of text (always 1)</td></tr>
<tr><td>58</td><td>Text</td><td>Y</td><td>Default value = "Credit Limit Information"</td></tr>
<tr><td></td><td><b><i>Standard Trailer</i></b></td><td><b><i>Y</i></b></td><td></td></tr>
</table>

The following table contains the extra and custom tags that can be received in this message.

Tag	Field Name	Req	Description

<table>
<tr><td colspan="2">453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
<tr><td>➡</td><td>447</td><td>PartyIDSource</td><td>Y</td><td>Default value = D (Proprietary/Custom Code)</td></tr>
<tr><td>➡</td><td>452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).<table><tr><td>Value</td><td>Meaning</td></tr><tr><td>1</td><td>Executing Firm</td></tr><tr><td>4</td><td>Clearing Firm</td></tr><tr><td>12</td><td>Trader ID</td></tr></table></td></tr>
<tr><td>➡</td><td>448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td colspan="2">5574</td><td>ODLMsgType</td><td>Y</td><td>ODLMsgType=TL</td></tr>
<tr><td colspan="2">5550</td><td>CreditLimit</td><td>Y</td><td>See Section 0</td></tr>
<tr><td colspan="2">5558</td><td>ClearingSpace</td><td>Y</td><td>See Section 0</td></tr>
<tr><td colspan="2">198</td><td>SecondaryOrderID</td><td>N</td><td>ODL message id. Used in recovery mechanism.</td></tr>
</table>

5.4.5. *News (ODL TO Message)*

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td></td><td><b><i>Standard Header</i></b></td><td><b><i>Y</i></b></td><td>B = News</td></tr>
<tr><td>148</td><td>Headline</td><td>Y</td><td>Reason for notification. Valid values are:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>M</td><td>Message Note</td></tr></table></td></tr>
<tr><td>33</td><td>NoLinesOfText</td><td>Y</td><td>Number of lines of text (always 1)</td></tr>
<tr><td>58</td><td>Text</td><td>Y</td><td>Message note</td></tr>
<tr><td></td><td><b><i>Standard Trailer</i></b></td><td><b><i>Y</i></b></td><td></td></tr>
</table>

The following table contains the extra and custom tags that can be received in this message.

Tag	Field Name	Req	Description

198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.
5577	NoteType	1.	A single character alphanumeric field to identify the nature of the news message: Possible values are: “0”: Free text message from the exchange “1”: Throttling warning. That informs the member firm that its CTCI rejections have reached half of the limit. Action should be taken to eliminate the cause of excessive rejections from the member.

5.5. DSS & Clearing

5.5.1. *DSS Entry*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	DE = DSS Entry
Message Body			
11	ClOrdID	Y	Client specified identifier of the order. Max size is 16-char.
453	NoPartyIDs	Y	Number of party identifiers.
➡ 447	PartyIDSource	Y	Default value = D (Proprietary/Custom Code)
➡ 452	PartyRole	Y	Role of the specified PartyID (448). **Value** – **Meaning** 1 – Executing Firm 12 – Trader ID
➡ 448	PartyID	Y	
100	ExDestination	Y	Values are given by ATHEX. Max length is 4 characters
5575	DSSMessageType	N	For future use.

212	XmlDataLen	Y	Specifies length of XML data in field 213
213	XmlData	Y	Contains an XML (FIXML 5.0 SP2) data message.
Standard Trailer			

5.5.2. *DSS Confirm*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	DC = DSS Confirm
Message Body			
11	ClOrdID	Y	Client specified identifier of the order. Max size is 16-char.
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.
453	NoPartyIDs	Y	Number of party identifiers.
➡ 447	PartyIDSource	Y	Default value = D (Proprietary/Custom Code)
➡ 452	PartyRole	Y	Role of the specified PartyID (448). **Value** **Meaning** 1 Executing Firm 12 Trader ID
➡ 448	PartyID	Y	
37	OrderID	Y	Unique identifier of the order.
100	ExDestination	Y	Values are given by ATHEX. Max length is 4 characters
5575	DSSMessageType	N	For future use.
212	XmlDataLen	Y	Specifies length of XML data in field 213
213	XmlData	Y	Contains an XML (FIXML 5.0 SP2) data message.
Standard Trailer			

5.5.3. *DSS Trade*

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Description</th></tr>
<tr><td colspan="4"><b>Standard Header</b></td></tr>
<tr><td>35</td><td>MsgType</td><td>Y</td><td>DT = DSS Trade</td></tr>
<tr><td colspan="4"><b>Message Body</b></td></tr>
<tr><td>11</td><td>ClOrdID</td><td>Y</td><td>Client specified identifier of the order. Max size is 16-char.</td></tr>
<tr><td>198</td><td>SecondaryOrderID</td><td>N</td><td>ODL message id. Used in recovery mechanism.</td></tr>
<tr><td>453</td><td>NoPartyIDs</td><td>Y</td><td>Number of party identifiers.</td></tr>
<tr><td>➡ 447</td><td>PartyIDSource</td><td>Y</td><td>Default value = D (Proprietary/Custom Code)</td></tr>
<tr><td>➡ 452</td><td>PartyRole</td><td>Y</td><td>Role of the specified PartyID (448).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Executing Firm</td></tr>
<tr><td>12</td><td>Trader ID</td></tr>
</table></td></tr>
<tr><td>➡ 448</td><td>PartyID</td><td>Y</td><td></td></tr>
<tr><td>37</td><td>OrderID</td><td>Y</td><td>Unique identifier of the order.</td></tr>
<tr><td>100</td><td>ExDestination</td><td>Y</td><td>Values are given by ATHEX. Max length is 4 characters</td></tr>
<tr><td>5575</td><td>DSSMessageType</td><td>N</td><td>For future use.</td></tr>
<tr><td>212</td><td>XmlDataLen</td><td>Y</td><td>Specifies length of XML data in field 213</td></tr>
<tr><td>213</td><td>XmlData</td><td>Y</td><td>Contains an XML (FIXML 5.0 SP2) data message.</td></tr>
<tr><td colspan="4"><b>Standard Trailer</b></td></tr>
</table>

5.5.4. *DSS Broadcast*

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	DB = DSS Broadcast

Message Body			
100	ExDestination	Y	Values are given by ATHEX. Max length is 4 characters
198	SecondaryOrderID	N	ODL message id. Used in recovery mechanism.
5575	DSSMessageType	N	For future use.
212	XmlDataLen	Y	Specifies length of XML data in field 213
213	XmlData	Y	Contains an XML (FIXML 5.0 SP2) data message.
Standard Trailer			

5.6. Custom Fix Tags

Depending on the originating ODL message and on configuration, custom tags (> 5000) may present in execution reports. The following table presents these tags with a short description for the values they can hold.

Tag	Field Name	Req
1194	ExerciseStyle	User-defined field taken from FIX 5.0 SP2. It is used only for derivatives. Possible values: “0” European “1” American
5023	OrderType	A 1 character alphanumeric field indicating the type of the order. Possible value : “N” New “I” Inactive
5501	OrigSource	A 1 character alphanumeric type indicating the source of the Order. Possible values : ‘C’ : CTCI –ODL ‘M’ : ORAMA-ETW ‘R’ : EMRW (ATHEX supervision application).

5502	MarketID	A 1 character alphanumeric field indicating the trading market. For possible values, please refer to ATHEX Rulebook: http://www.ase.gr/content/en/about/regulations/files/ATHEX%20final.pdf This market should have such a status so as to accept orders (namely its status should not be: "R", "E", "H", "S").
5506	BoardID	A 1 character alpha field that identifies the trading board. Possible Values: "M" Main board "S" Special conditions board "B" Report Only board "F" Forced sales board
5508	CancelReasonCode	A 1 character alpha field used to indicate the reason for order cancellation. Possible values are: "C" canceled by CTCI request "M" canceled by ETW request "R" canceled by EMRW request "I" cancellation of IOC order "F" cancellation of FOK order "P" cancellation of MKT order "O" cancellation of ATO order "L" cancellation of Life order "S" cancellation of Covered Sale Order by CSD
5509	OrderRelFlag	"1" Normal "2" Quote "3" Combo "4" Trade Report

5511	PhaseID	A 1 character alpha code that identifies the trading phase. Possible values are: “ ” Start of day (Before the Pre-opening) “P” Pre-opening Trading Phase “O” Opening Trading Phase “T” Continuous Trading Phase “A” At the Closing Price trading Phase “C” Closing Price Continuous Trading Phase “E” End Of Trading Phase “S” Stop phase (Use in auction market)
5512	GOIFlag	Group Of Investor flag (Not used)
5521	StopSymbol	A 15 characters alphanumeric field that indicates either the index or security ID related to stop condition.
5522	SecurityStatus	A 1 character flag that indicates whether the security is active or not. Possible values are: “A” Active “N” Not active “S” suspended “H” Halted “R” resumed (Resumed Pre-opening of a Halt)
5527	StopSymbolType	Possible values are ‘I’ for stop on index order and ‘S’ for stop on symbol orders. Absence of this tag means ‘S’ i.e. stop on symbol.
5529	ODLTradeType	A 2 character field to identify the type of trade. Possible values: Execution Reports: “MB” Main Board Trade

		“MO” Main Board Opening Trade “MC” Main Board Closing Trade “MM” Main Board, Combination to Combination Trade “ST” Special Board Trade “OL” Odd-lot Board Trade “FS” Forced sale Board Trade Trade Capture Reports: “RA” Trade report Method 6-1 “RD” Trade report Spot-1 “RU” Trade report Spot-2 “RR” Trade report Settlement – Incomplete Buy “RS” Trade report Settlement – Incomplete Sell “RB” Trade report Method 6-1 Same Day Settlement “RM” Trade report Method 6-1 Special Fees “RN” Trade report Method 6-1 Special Fees & Same Day Settlement “RL” Trade Report Lending – Borrowing “RC” Trade report method 6-1, No CCP “RG” Trade report method 6-1, No CCP & Same Day Settlement “RT” Trade Report Lending – Borrowing, Market Making “RF” Trade Report Lending – Borrowing, Failed Trade “RE” Trade Report Method 7-1 Derivatives market “RQ” Trade Report - Other
5530	SecurityPrice	A numeric field that contain the security price whenever “Security Status” message is sent (see also Price). Possible values:

		At the beginning of the Pre-Opening phase - Start of day price At the beginning of the Opening phase - Zeroes At the beginning of the Continuous phase - Opening price At the beginning of the Closing phase - Zeroes At the beginning of the Closing price phase – Closing price At the End of Trading – Closing price At a resumption – Resumed Opening price In all other cases – Last sale price
5531	ODLHaltReason	A 2 characters alphanumeric field indicating the cause of the halt or suspension. For possible values, please refer to HaltReasonCode in Appendix B. ODL-Client-API: ETS_Broker properties
5532	RejectReasonCode	A 3 characters numeric field used to indicate to a member firm the reason that a requested action could not take place. For possible values, please refer to RejectReasonCode in Appendix B. ODL-Client-API: ETS_Broker properties
5550	CreditLimit	A 14 characters numeric (12.2) field indicating the credit limit assigned to the clearing firm
5545	CurrentCreditValue	A 14 characters numeric (12.2) field indicating the current credit value of the clearing firm
5558	ClearingSpace	A 4 characters alphanumeric field indicating the clearing space.
5561	MBListID	A 6 characters alphanumeric field indicating the List ID.
5562	AlarmReason	Used only in case of Headline(148) = “A” (Quote Alarm). This attribute may take one of the following values: “1” Prices are missing “2” BID price is missing

		"3" ASK price is missing "4" The Price Spread is too big "5" Quantities are too small "6" BID Quantity is too small "7" ASK Quantity is too small
5563	QuotationResponsibility Level	Identifies the quotation responsibility level. Possible values: "M" Market "I" Instrument. The particular responsibilty also refers to the all instruments wich are based on (derived from) instrument in matter. "F" Firm
5564	ReasonOfSuspension	Identifies the reason for quote responsibility suspension. Sent only if Headline(148)='S' (Suspend quotation responsibility) "1" Manual suspension "2" Suspension because of auction "3" Suspension because of limit up "4" Suspension because of limit down "5" Minimum daily order volume has been covered
5572	DaysAvrgTrdVal	A 14 characters numeric (12.2) field indicating the average daily turnover for a specific security
5574	ODLMsgType	A 2 characters alphanumeric field to identify the type of ODL status message Possible values are: "CA" security status message "CD" security prices message "TL" credit limit info message

5575	DSSMessageType	A single character alphanumeric field to identify the type of DSS message. 'A' by default. For future use.
5577	NoteType	A single character alphanumeric field to identify the nature of the news message: Possible values are: "0": Free text message from the exchange "1": Throttling warning. This informs the member firm that its CTCI rejections have reached half of the limit resulting in disabling of its ODL link. Action should be taken to eliminate the cause of excessive rejections from the member.
5578	SecurityISIN	A 12-character alphanumeric field, indicating security's ISIN code.
5999	SpecialInstructions	A 120-character alphanumeric field, used only for XNET interface

5.7. OrdType and TimeInForce Combinations

In the table bellow, all combinations of the FIX fields OrdType(40) and TimeInForce(59) are described.

FIX		**Description**
OrdType	**TimeInForce**	
1	0	*MKT order, no conditions, valid only for current day*
1	1	*MKT order, no conditions, time validity is good till cancel*
1	2	*ATO order, no conditions, valid only for current day*
1	3	*MKT order, time validity = Immediate Or Cancel*
1	4	*MKT order, time validity = Fill Or Kill*
1	6	*MKT order, no conditions, time validity is good till date*
7	0	*LIMIT order, no conditions, valid only for current day*
7	1	*LIMIT order, no conditions, time validity is good till cancel*

7	2	*INVALID COMBINATION*
7	3	*LIMIT order, time validity = Immediate Or Cancel*
7	4	*LIMIT order, time validity = Fill Or Kill*
7	6	*LIMIT order, time validity = good till date*
3	0	*MKT order, stop on security or index, valid only for current day*
3	1	*MKT order, stop on security or index, time validity = good till cancel*
3	2	*INVALID COMBINATION*
3	3	*INVALID COMBINATION*
3	4	*INVALID COMBINATION*
3	6	*MKT order, stop on security or index, time validity = good till date*
4	0	*LIMIT order, stop on security or index, valid only for current day*
4	1	*LIMIT order, stop on security or index, time validity = good till cancel*
4	2	*INVALID COMBINATION*
4	3	*INVALID COMBINATION*
4	4	*INVALID COMBINATION*
4	6	*LIMIT order, stop on security or index, time validity = good till date*
A	0	*ATC order, no conditions, valid only for current day*
A	1	*INVALID COMBINATION*
A	2	*INVALID COMBINATION*
A	3	*INVALID COMBINATION*
A	4	*INVALID COMBINATION*
A	*6*	*INVALID COMBINATION*

5.8. Unsolicited order changes

The following actions induce unsolicited (ExecType = ‘D’) ExecutionReport messages. Tag 378 (ExecRestatementReason) is also provided in such Execution reports as indicated by the table below:

Execution Report			Event Description of Order Activation / Inactivation
ExecType	OrdStatus	*ExecRestatementReason*	
D (Restated)	I (Inactive)	*1 (GT renewal/restatement)*	*A life order bebomes inactive by the trading system when at the start of trading session it does not pass the validation rules (price limits, credit limit etc)*
		4 (Broker Option)	*An order becomes inactive by the trading sytem when it is fully executed (MaxShow(210) = 0) but the OrderQty(38) is still greater than zero*
D (Restated)	‘4’ (Cancelled)	*4 (Broker Option)*	*FOK/IOC/Market order cancellation by the trading system as of unfulfilled condition*
D (Restated)	‘C’ (Expired)	*1 (GT renewal/restatement)*	*Expired life order (either due to order expiry date or series expiration)*
D (Restated)	0 (New)	*1 (GT renewal/restatement)*	*A valid life order is restated at the beginning of subsequent days.*
			Stop/ATC order released
		4 (Broker Option)	*Market order partially filled, restated as ‘limit or better’ at last match price*
D (Restated)	I (Inactive)	*2 (Verbal Change)*	*An order is deactivated on demand by ATHEX member staff.*
D (Restated)	4 (Cancelled)	*2 (Verbal Change)*	*An order is cancelled on demand by ATHEX member staff.*

6. Instructions for implementing business logic

6.1. General Concepts of the OASIS system

6.1.1. *Tick size mechanism*

Trading in OASIS is restricted to ‘valid’ prices. The validity of a price depends, amongst others, on the ‘price tick’. This is defined as the minimum allowed price deviation and the difference between two valid prices is bound to be a multiple of the designated tick size.

This restriction applies to:

- every order sent to the system.

- the entry of the opening price following a Corporate Action, the price of which should always be at a valid tick.

If for some reason a value which does not correspond to a valid price tick is produced (e.g. while calculating the securities floor and ceiling), then the value is rounded to a valid price tick. For the calculation of floors it is rounded upwards to the nearest price tick, while for the calculation of ceilings it is rounded downwards to the previous price tick. This rounding rule rounds towards the interior of the range denoted by the floor/ceiling pair.

The opening prices of securities traded in the system are already placed at a valid price tick. If the provided opening price of a security is not placed at a valid price tick, the system itself rounds it to the nearest price tick. If the price is equidistant from the nearest price ticks, then it is rounded downwards.

The table with the price ticks currently in the System is presented in Paragraph 0 (Appendix on OASIS Server parameters).

6.1.2. *Lot size*

For each security traded in OASIS a Lot Size has been set.

The securities are traded at a volume multiple of the minimum lot size on the Main Boards and at a lower quantity of the minimum lot size on the Odd Lot Boards.

The entry of orders with:

- a lower quantity than
- integer multiple of
- non integer multiple of

the Lot Size (Mixed Lot) is allowed on the Pre-Agreed Trades boards. The entry of orders through the ATHEX Gateway is not allowed on those Boards though.

6.1.3. *Order Priority Change*

When an order is changed through an Order Change (MD) message (ODL client application) or Order Cancel/Replace Request (FIX client application), the time **priority** of the order shall change:

- If the order price changes;
- If the order Disclosed Volume increases.

In both cases the order time stamp is updated.

6.1.4. *Member Credit Limit*

The system checks the total value of orders executed or not for each member.

Each Member has a daily amount as the maximum limit of transaction's value. This amount is communicated to the Member at the beginning of each trading day and each time this limit is readjusted by ATHEX (see message Credit Limit Information TL, paragraph 4.2.3.5.1).

Each time the Member's user enters a new order or changes / edits an existing order, or when a trade is made, the current credit value is communicated to the Member through the following messages:

- Confirm of New Order (TB, see paragraph 4.2.3.1.4).
- Confirm of Order Change (TD, see paragraph 0).
- Confirm of Order Edit (TC, see paragraph 4.2.3.1.5).
- Confirm of Trade (TF, see paragraph 4.2.3.1.7).

The system does not allow the entry of an order by the member which causes the member's credit limit to be exceeded. In the event of sending a new order message or a message to change an order resulting the current credit value in exceeding this limit, the OASIS Server will reject the order and will return a TR message with an "081" error code.

6.2. Business logic and verifications before sending messages

Chapter 4 presented a detailed description of the objects and relevant messages sent by the members' applications to the OASIS or the target exchanges in general. **It is required** that the applications of the Members verify the messages they send, both in terms of **formatting** and **content.** The values of the object's properties should be within the range of the **possible field values,** as these were defined in Chapter 4.

In the following section, additional instructions are given concerning the completion of the properties of Application Request type objects addressed to OASIS.

6.2.1. *New Order*

6.2.1.1. Initial Verifications

The sending of the New Order message to the OASIS Server through the OrderEntry object should be done only if the system has not Halted (namely no CB message with Status="H" has been sent).

Next, the limitations / verifications that need to be performed for each property of the OrderEntry object are given.

6.2.1.2. MessageSource

MessageSource = "C". All messages sent through the CTCI – ATHEX Gateway should have the value "C" in the MessageSource field.

6.2.1.3. MemberID

The Member code set forth by the ATHEX is filled in. In order for a Member to be able to send orders it has to be active.

6.2.1.4. TraderID

The code of a Member's **active CTCI user.** Such a user is set by the ATHEX.

6.2.1.5. VenueID

This should be a valid venue id as specified in the International Standard ISO 10383.

6.2.1.6. BoardID

The code of the board in which the order will be entered. For possible values, please refer to ATHEX Rulebook:

http://www.ase.gr/content/en/about/regulations/files/ATHEX%20final.pdf

Orders for the Pre-Agreed Price board are entered through the ATHEX Gateway by using an ODL Client application only.

This board should have such a status so as to accept orders. Specifically its status should not be: "R", "E", "H","S" if it is a main board and it should have a Status = Open (Status = "O") if it is not a main board.

Appendix G, OASIS Server parameterspresents the orders allowed in the form of a table depending on a board type.

6.2.1.7. OrderType

This property may be "N" or "I" for automatic insertion of a new active order or an automatic insertion of a new inactive order respectively. For orders directed to Xorder Server two more options exist: 'M' for manual insertion of a new active order, and 'X' for manual insertion of a new inactive order.

6.2.1.8. Side

This property may be "B" or "S" for a buy order or a sell order respectively.

6.2.1.9. CSDAccountID

It shall be completed with the code of the client who gave the order. All characters must be upper case.

6.2.1.10. GOIFlag

This field is not used anymore. The Group Of Inverstor functionality is removed.

6.2.1.11. **ShortSellFlag**

This attribute indicates whether the order is a short sell order or a buy to cover order or none of the above. It may take the following values:

- “N”: Normal
- “Y”: Short Sell / Buy To Cover

6.2.1.12. **SecurityID**

It is completed with the identification of the security for which the order is being entered. The class or source of the SecurityID is identified by the SecurityIDSource field (see bellow). The security needs to be active (Security Status = “A” or “R”) and it should be in a trading phase which allows the entering of an order (PhaseID = “P”, “O”, “T”, “C”). At any time the Member’s application should know the status of a new security by using the information it receives through the Security Status (CA) messages.

6.2.1.13. **SecurityIDSource**

It is completed with the class or source of the SecurityID. It may take the following values:

- “8”: Exchange Symbol
- “A”: Bloomberg Symbol.

6.2.1.14. **Currency**

This attribute indicated the currency for this trade report. It is used in messages directed to Xorder Server.

6.2.1.15. **Price**

This property is completed with the order price. This price should be a valid price tick (see Paragraph 5.1.1) and it should be equal to or more than the price floor and less than or equal to the price ceiling.

In cases of negative price values (as used in combination orders) the sign (-) is incorporated in this field, reducing the numeric part to 4.4 format. The negative sign should occupy the first character of the integer part, but should be omitted in positive values. E.g a price of “-1.23” translates to “-00012300”.

In the event that the order is of MKT or ATO type, the value entered in this field is “000000000”.

6.2.1.16. **Volume**

If the BoardID property mentioned earlier is “M” and thus we enter an order in the main board, the Volume property must be a multiple of the security Lot Size (see Paragraph 6.1.2).

Otherwise if the BoardID property mentioned is “O”, the Volume property needs to be less than the security Lot Size and greater than 0.

A Mixed Lot Volume (a volume greater than the Lot Size but not an integral multiple) is not allowed.

6.2.1.17. **DisclosedVolume**

In the event that we set a condition in the order, namely the SpecialConditions property is different than “N”, or the order entered is MKT / ATO, or the order concerns odd-lot, the DisclosedVolume should be equal to the Volume.

Otherwise, it should be a multiple of the Lot Size and less than or equal to the Volume field.

It should be noted that an order with a DisclosedVolume less than the Volume, as soon as it is executed for the number of items indicated in the DisclosedVolume it changes to non-active.

6.2.1.18. **AutoDisclosedVolume**

AutoDisclosedVolume = “000000000000”. Reserved for future use.

6.2.1.19. **ConditionVolume**

Condition Volume = “000000000000”. Reserved for future use.

6.2.1.20. **OrderLifeTime**

- This attribute may take the following values:“D” for orders that if they remain not executed or partially not executed until the end of the current day’s trading session they will be cancelled by the system.

- “C” for orders that remain in the orders book (for the upcoming days trading sessions as well) until fully executed or cancelled by the Member.

- “E” for orders that will remain active in the orders book until they are executed or up until the date set in the ExpirationDate property.

6.2.1.21. **SpecialConditions**

This attribute may take the following values:

- “N”: No condition is entered.

- “I”: for Immediate or Cancel orders. They are entered only during the continuous trading phase of a security (a CA message with Phase Id = “T” has been received).

- “F”: for Fill or Kill orders. They are entered only during the continuous trading phase of a security (a CA message with Phase Id = “T” has been received).

- “S”: for Stop orders on a Security.

- “D”: for Stop orders on an Index.

6.2.1.22. OriginalPriceType

This attribute may take the following values:

- “L”: for orders with a price limit.

- “O”: for orders at opening price. Orders at opening price are entered only when the security is in the Pre-opening phase (a CA message with Phase Id = “P” has been received).

- “M”: for Market (MKT) orders.

- “C”: for orders at closing price.

6.2.1.23. StopSymbol

If the attribute SpecialConditions has a value of “S” or “D”, then the StopSymbol attribute is filled using the identification of the Security or Index included in the condition. The Security must be active (Security Status other than “N”).

In any other case StopSymbol = “   “ (12 blanks)

6.2.1.24. StopPrice

If the Special Conditions attribute has a value of “S” or “D”, then the StopPrice attribute gets the value which if reached or exceeded by the Stop Symbol the order will be activated.

If the order is a sell order (Side = “S”) then the StopPrice should be less than the Security or Index Last Price used as StopSymbol.

If the order is a buy order (Side = “B”) then the StopPrice should be greater than the Security or Index Last Price used as StopSymbol.

Given that the ATHEX Gateway does not provide security price information, the Member’s application should always use the price information coming from another source in order to verify the correctness of the StopPrice attribute.

If the SpecialConditions attribute does not have a value of “S” or “D”, then the Stop Value must be set to “000000000”

6.2.1.25. ExpirationDate

If the OrderLifetime attribute has the value “E”, the ExpirationDate attribute is completed with the date (YYYYMMDD) on which the order will expire. This date must be later than the current date.

In the event that this date falls on a weekend or a non working day, the Server will replace it with the exact previous working day.

In any other case ExpirationDate = “   “ (8 blanks)

6.2.1.26. **ClientOrderID**

This attribute should be completed using 16 characters and it is intended for internal use by the member.

6.2.1.27. **OrderNote**

This attribute should be completed using 25 characters and it is intended for internal use by the member.

6.2.1.28. **ListID**

This attribute should be completed using 6 characters and it is intended for clearing procedure instructions.

6.2.1.29. **ClearingMemberID**

This attribute should be completed using 4 characters indicating the clearing sub-account ID.

6.2.1.30. **PositionEffect**

This attribute indicates whether the resulting position after a trade should be an opening position or closing position. This field must contain a valid value in all cases; however it will be ignored for non-derivative products.

- “O”: open
- “C”: close (netting)

6.2.1.31. **SettlType**

This attribute indicates the order settlement period. It is especially used in the repos (lending) market to indicate the necessity of immediate settlement of the corresponding order in case of fail trade

- “1”: Immediate Settlement
- “0”: Normal settlement period

6.2.2. *Order Edit (cancellation/activation/deactivation)*

6.2.2.1. **Initial Verifications**

Before the Order Edit message is sent using the OrderEdit object (ODL client) or Order Cancel Request message (FIX client), it should be verified whether the Order to be edited (cancelled/activated/deactivated) has not one of the following statuses:

- It has already been cancelled (OrderStatus = “X”).

- The order has been completely matched and no unexecuted part of it exists (OrderStatus = “M”).

- It has been rejected (OrderStatus = “D”).

- It concerns a Pre-Agreed Price trade (OrderStatus is “RA”, “RD”, “RU”, “RR”, “RS”, “RB”, “RM”, “RN”, “RL”, “RT”, “RF” or “RE”).

- It is an inactive order and the user tries to deactivate it (Entered EditType = ‘S’).

In order for the Member’s application to know the order status, it should use the confirmation messages (TB, TC, TD, TF) described in Paragraph 4.2.3.2.

Order Editing is not permitted while subject security is in Phase ‘E’ (end-of-trading) as well as phase ‘S’* (Stop). *only applicable on main type market securities.

Furthermore, for each attribute of the OrderEdit object the following should hold true:

6.2.2.2. **MessageSource**

MessageSource = “C”. All messages sent through the CTCI – ATHEX Gateway should have the value “C” in the MessageSource field.

6.2.2.3. **MemberID**

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to cancel/activate/deactivate.

In order for a Member to be able to cancel/activate/deactivate one of its orders, the Member should be active or partially suspended.

6.2.2.4. **TraderID**

The code of a Member’s **active CTCI user.** Such user is set by the ATHEX.

6.2.2.5. **VenueID**

This should be a valid venue id as specified in the International Standard ISO 10383.

6.2.2.6. **BoardID**

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to cancel/activate/deactivate.

6.2.2.7. **SecurityID**

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to cancel/activate/deactivate.

6.2.2.1. **SecurityIDSource**

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to cancel/activate/deactivate.

6.2.2.1. **Currency**

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to cancel/activate/deactivate.

6.2.2.2. **OrderNumber**

The OrderNumber of the OrderEdit object should be the same as the NewOrderNumber of the TB message (Confirm of New Order) that was sent as a confirmation by the OASIS Server upon the entry of the order.

6.2.2.3. **OrderDate**

The OrderDate of the OrderEdit object should be the same as the OrderDate of the TB message (Confirm of New Order) that was sent as a confirmation by the OASIS Server upon the entry of the order.

6.2.2.4. **OrigCientOrderID**

This attribute should be completed using 16 characters and should be blank or the same as the ClientOrderID of the order, received in the previous confirmation message (TB, TD, TC) for this order.

6.2.2.5. **CientOrderID**

This attribute should be completed using 16 characters and it is intended for internal use by the member.

6.2.2.6. **EditType**

The value of this attribute shall indicate the action to be performed on the specific order. This attribute may take one of the following values:

- ‘C’ Cancel order.
- ‘S’ Suspend (deactivate) order.

- ‘U’ Unsuspend (activate) order.

6.2.2.7. **UnsuspendDisclosedVolume**

The value of this attribute shall indicate the volume that is announced to the market, when an inactive order is activated.

6.2.2.8. **OrderNote**

This attribute should be completed using 25 characters and it is intended for internal use by the member.

6.2.2.9. **ListID**

This attribute should be completed using 6 characters and it is intended for clearing procedure instructions.

6.2.3. ***Order Change***

6.2.3.1. **Initial Verifications**

Before the Order Change message is sent using the OrderChange object (ODL client) or Order Cancel/Replace Request message (FIX client), it should be verified whether the Order to be changed is still Open in the Market which means that:

- The order has not been completely matched (there exists an unexecuted part) (OrderStatus <> “M”).

- It has not been previously cancelled (OrderStatus <> “X”).

In order for the Member’s application to know the order status, it should use the confirmation messages (TB, TC, TD, TF) as described in Paragraph 4.2.3.2.

Also, the system should not be Halted (no CB message with Status = “H” has been received).

Furthermore, for each attribute of the OrderChange object the following should hold true:

6.2.3.2. **MessageSource**

MessageSource =”C”. All messages sent through the CTCI – ATHEX Gateway should have the value “C” in the MessageSource field.

6.2.3.3. **MemberID**

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to change.

In order for a Member to be able to change its order it has to be active.

6.2.3.4. TraderID

The code of a Member’s **active CTCI user.** Such user is set by the ATHEX.

6.2.3.5. VenueID

This should be a valid venue id as specified in the International Standard ISO 10383.

6.2.3.6. BoardID

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to change.

This Board should have such a status so as to accept orders. Namely its status should not be: “R”, “E”, “H”, “S”) if it is a main board and it should have a Status = Open (Status = “O”) if it is not a main board.

6.2.3.7. SecurityID

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to change.

In addition, the specified security needs to be active (Security Status = “A” or “R”) and it should be in a trading phase which allows the entering of an order (PhaseID = “P”, “O”, “T”, “C”). At any time the Member’s application may know the status of a new security by using the information it receives through the Security Status (CA) messages.

6.2.3.8. SecurityIDSource

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to change.

6.2.3.9. Currency

The value of this attribute should be the same as that of the equivalent attribute of the order that we want to change.

6.2.3.10. OrderNumber

The OrderNumber of the OrderChange object should be the same as the NewOrderNumber of the TB message (Confirm of New Order) that was sent as a confirmation by the OASISServer upon the entry of the order.

6.2.3.11. OrderDate

The OrderDate of the OrderChange object should be the same as the OrderDate of the TB message (Confirm of New Order) that was sent as a confirmation by the OASISServer upon the entry of the order.

6.2.3.12. ChangedPrice

This attribute is completed with the new price of the order. This price should be at a valid price tick (see Paragraph 6.1.1) and it should be greater or equal to the floor price and less or equal than the ceiling price.

In cases of negative price values (as used in combination orders) the sign (-) is incorporated in this field, reducing the numeric part to 4.4 format. The negative sign should occupy the first character of the integer part, but should be omitted in positive values. E.g a price of “-1.23” translates to “-00012300”.

In the event that the order is of the MKT,ATO or ATC type, the value entered in this field is “000000000”.

6.2.3.13. ChangedVolume

In the event that the initial order has been partially executed, the ChangedVolume attribute should be greater than the Matched Volume (the number of the items for which a trade has been made):

ChangedVolume > Matched Volume >= 0

The Member’s application should calculate the Matched Volume through the Trade Confirmation messages (TF).

In addition, as in the case of a new order entry, should the BoardID attribute have an “M” value and thus we enter an order in the main board, the Changedolume attribute should be a multiple of the Lot Size of the security (see Paragraph 5.1.2). Otherwise if the BoardID attribute is “O”, the ChangedVolume attribute needs to be less than the security Lot Size and greater than 0. A Mixed Lot Volume (namely a volume greater than the Lot Size but not an integral multiple) is not allowed.

6.2.3.14. ChangedDisclosedVolume

The ChangedDisclosedVolume should be a multiple of the Lot Size, greater than 0 and less or equal than the order’s items for which no trade has been made:

0 < ChangedDisclosedVolume <= Remaining Volume

where Remaining Volume = ChangedVolume – Matched Volume

In the case of an order where one of the following applies:

- SpecialConditions not equal to “N”.
- The order is MKT / ATO.
- The order concerns odd-lot.

The ChangedDisclosedVolume should be equal to the Remaining Volume:

ChangedDisclosedVolume = Remaining Volume where Remaining Volume = ChangedVolume – Matched Volume

6.2.3.15. **ChangedAutoDisclosedVolume**

ChangedAutoDisclosed Volume = “000000000000” (12 zeroes). Reserved for future use.

6.2.3.16. **ChangedCSDAccountID**

It is filled with the code of the client who gave the order. It might also differ from that of the initial order. All characters must be upper case.

6.2.3.17. **ChangedGOIFlag**

This field is not used anymore. The Group Of Inverstor functionality is removed.

6.2.3.18. **ChangedShortSellFlag**

This attribute indicates whether the order is a short sell order or a buy to cover order or none of the above. It may take the following values:

- “N”: Normal
- “Y”: Short Sell / Buy To Cover

6.2.3.19. **ChangedOriginalPriceType**

This attribute may take the following values:

- “L”: for orders with a price limit.
- “O”: for orders at opening price. Orders at opening price are entered only when the security is in the Pre-opening phase (Security Phase Id = “P”).
- “M”: for Market (MKT) orders.
- “C”: for orders at closing price.

The value of this attribute may differ from the equivalent value of the attribute of the initial order.

6.2.3.20. **ChangedLife**

ChangedLife may take three values:

- “D” for orders that if they remain non executed or partially non executed until the end of the current day’s trading session they will be cancelled by the system.
- “C” for orders that remain in the orders book (for the upcoming days’ trading sessions as well) until fully executed or cancelled by the Member.

- “E” for orders that will remain active in the orders book until they are executed or up until the date set in the ChangedExpirationDate attribute.

The value of this attribute may differ from the equivalent value of the attribute of the initial order.

6.2.3.21. **ChangedExpirationDate**

If the ChangedLife attribute has the value “E”, then the ChangedExpirationDate attribute is completed with the date (YYYYMMDD) on which the order will expire. This date must be later than the current date.

In the event that this date falls on a weekend or a non working day, the Server will replace it with the exact previous working day.

In any other case OrderLifeTime = “   “ (8 blanks)

The value of this attribute may differ from the equivalent value of the attribute of the initial order.

6.2.3.22. **OrigClientOrderID**

This attribute should be completed using 16 characters and should be blank or the same as the ClientOrderID of the order, received in the previous confirmation message (TB, TD, TC) for this order.

6.2.3.23. **ClientOrderID**

This ClientOrderID attribute should be completed using 16 characters and it is intended for internal use by the Member.

The value of this attribute may differ from the equivalent value of the attribute of the initial order.

6.2.3.24. **ChangedOrderNote**

This ChangedOrderNote attribute should be completed using 25 characters and it is intended for internal use by the Member.

The value of this attribute may differ from the equivalent value of the attribute of the initial order.

6.2.3.25. **ChangedClearingMemberID**

At the present phase it is completed using 4 characters and shall be identical to the ClearingMemberID field of the corresponding New Order (MB) message.

6.2.3.26. **ListID**

This attribute should be completed using 6 characters and it is intended for clearing procedure instructions.

6.2.3.27. **ChangedPositionEffect**

This attribute indicates whether the resulting position after a trade should be an opening position or closing position. This field must contain a valid value in all cases; however it will be ignored for non-derivative products.

- "O": open
- "C": close (netting)

6.2.3.28. **ChangedSettlType**

This attribute indicates the order settlement period. It is especially used in the repos (lending) market to indicate the necessity of immediate settlement of the corresponding order in case of fail trade

- "1": Immediate Settlement
- "0": Normal settlement period

6.2.3.29. **Remarks**

Changes are allowed only in the attribute values of Paragraphs 6.2.3.10 to 6.2.3.21.

The following order elements are not available for change by the Order Change message:

1. Side
2. ConditionVolume
3. SpecialConditions
4. STOPSymbol
5. StopPrice

Should there be a need for changes in the above fields, then the order should be cancelled (Order Cancel) and a New Order should be entered.

6.2.4. ***Trade Report Entry***

6.2.4.1. **Initial Verifications**

The sending of the Trade Report Entry Message to the OASIS Server through the TradeReportEntry object should be done only if the system has not Halted (namely no CB message with Status="H" has been sent).

This message is used by a member firm to **enter a 1-firm Trade Report or a 2-firm Trade Report**. In case of 2-firm Trade Report, only the seller is allowed to send this message and identify the buyer side with "BuyMemberID" and "BuyTraderID" fields.

This message is also used in order to:

- Approve or disapprove the trade report in case member is the buyer side.
- Request for trade report cancellation if member is the seller side.

If the trade report has been entered/approved/disapproved/cancelled, a Trade Message (TF) is returned by OASIS to each member firm participating in this trade report. Otherwise, a Reject Message (TR) is sent to the member firm with a reason code explaining why the request was not accepted.

The reception of a Trade Message (TF) with TradeStatus field equal to “L ” (alleged) from a member’s application denotes the entry of a trade report from another (or the same) member.

Once the trade report is registered with OASIS:

- 1-firm Trade Report is completed.
- 2-firm Trade Report is pending approval from buyer side member. The buyer can complete the trade, through a Trade Report Entry (MI) action discussed below.

A unique trade number is generated by OASIS.

Next, the limitations / verifications that need to be performed for each property of the TradeReportEntry object are given.

In trade reporting, field ClientOrderID can be both

- The (TF) echo of the client-side originated identifier (MI) ClientTradeID (i.e. in responses to the client).
- The trading system identifier of a Trade Report in messages originating from the trading system (i.e. Trade Report status relays).

Field OrigClientOrderID (TF) is at all times used to refer to the previous ClientOrderID in a sequence of trade reporting messages.

6.2.4.2. **MessageSource**

MessageSource = “C”. All messages sent through the CTCI – ATHEX Gateway should have the value “C” in the MessageSource field.

6.2.4.3. **MemberID**

The Member code set forth by the ATHEX is filled in. In order for a Member to be able to send trade reports, it has to be active.

6.2.4.4. **TraderID**

The code of a Member’s **active CTCI user.** Such user is set by the ATHEX.

6.2.4.5. VenueID

This should be a valid venue id as specified in the International Standard ISO 10383.

6.2.4.6. BoardID

The code of the board in which the trade report will be entered. For trade reports only the value “B” (report only board) is allowed.

This Board should have a Status = Open (Status = “O”).

Appendix G, OASIS Server parameterspresents the orders allowed in the form of a table depending on a board type.

6.2.4.7. SecurityID

It is completed with the identification of the security for which the trade report is being entered. The class or source of the SecurityID is identified by the SecurityIDSource field (see bellow). The security needs to be active (Security Status = “A” or “R”) and it should be in a trading phase which allows the entering of a trade report (PhaseID = “P”, “O”, “T”, “C”). At any time the Member’s application should know the status of a new security by using the information it receives through the Security Status (CA) messages.

6.2.4.1. SecurityIDSource

It is completed with the class or source of the SecurityID. It may take the following values:

“8”: Exchange Symbol

“A”: Bloomberg Symbol

6.2.4.1. Currency

This attribute indicated the currency for this trade report. For future use.

6.2.4.2. Price

This property is completed with the order preagreed price. This price should be a valid price tick (see Paragraph 6.1.1).

6.2.4.3. Volume

This property is completed with the preagreed price trade volume.

6.2.4.4. ClientTradeID

This property is completed using 16 characters and it is intended for internal use by the member.

6.2.4.5. SellMemberID

The seller Member code set forth by the ATHEX is filled in. In order for a Member to be able to send trade reports it has to be active.

6.2.4.6. SellTraderID

The code of the seller Member’s **active CTCI user.** Such user is set by the ATHEX.

6.2.4.7. SellClientOrderId

This attribute should be completed using 16 characters and it is intended for internal use by the member.

6.2.4.8. SellCSDAccountID

It shall be completed with the code of the client from the seller side who entered the trade report. All characters must be upper case.

6.2.4.9. ShortSellFlag

This attribute indicates whether the sell side of the trade report is a short sell order or or not. It may take the following values:

- “N”: Normal
- “Y”: Short Sell

6.2.4.10. SellMemberOrderNumber

This attribute should be completed using 11 characters and it is intended for internal use by the member.

6.2.4.11. SellClearingMemberID

This attribute should be completed using 4 characters indicating the clearing sub-account ID of the seller.

6.2.4.12. BuyMemberID

The buyer Member code set forth by the ATHEX is filled in. In order for a Member to be able to send trade reports it has to be active.

6.2.4.13. BuyTraderID

The code of the buyer Member’s **active CTCI user.** Such user is set by the ATHEX.

6.2.4.14. **BuyClientOrderId**

This attribute should be completed only in 1-firm Pre-Agreed Price Trade using 16 characters and it is intended for internal use by the member. In 2-firm Pre-Agreed Price Trade should be blank.

6.2.4.15. **BuyCSDAccountID**

It should be completed only in 1-firm Pre-Agreed Price Trade with the code of the client from the buyer side who entered the trade report. In 2-firm Pre-Agreed Price Trade should be blank. All characters must be upper case.

6.2.4.16. **BuyToCoverFlag**

This attribute indicates whether the buy side of the trade report is a buy to cover order or or not. It may take the following values:

- “N”: Normal
- “Y”: Short Sell

6.2.4.17. **BuyClearingMemberID**

This attribute should be completed using 4 characters indicating the clearing sub-account ID of the buyer.

6.2.4.18. **TradeReportMethod**

This attribute may take the following values:

- “RA”: Trade Report – Method 6-1“RD”: Trade Report - Spot-1
- “RU”: Trade Report - Spot-2
- “RR”: Trade Report - Settlement – Incomplete buy
- “RS”: Trade Report – Settlement – Incomplete sell
- “RB”: Trade Report – Same-day Settlement (SDS)
- “RM”: Trade Report – Special Fees (SF)
- “RN”: Trade Report – SDS & SF
- “RC” Trade report method 6-1, No CCP
- “RG” Trade report method 6-1, No CCP & Same Day Settlement
- “RL”: Trade Report – Lending-Borrowing

- “RT”: Trade Report – Lending-Borrowing, Market Making (MM)
- “RF”: Trade Report – Lending-Borrowing, Failed Trade
- “RE”: Trade Report – Method 7-1
- “RQ” : Trade Report - Other

6.2.4.19. **TradeEditType**

A 1 character alpha field identifying the type of trade report status change apllied from TradeReportEntry.

Possible values:

“N”	New
“A”	Approve Trade Report
“D”	Disapprove Trade Report
“C”	Cancel of Trade Report

6.2.4.1. **TradeNumber**

The TradeNumber of the TradeReportEntry object should be the same as the TradeNumber of the TF message (Trade) that was sent as a confirmation by the OASIS Server upon the entry of a new trade report using the initial TradeReportEtry message by the seller.

This field identifies the trade number of the trade report to be approved/dissapproved/cancelled. If TradeEditType == ‘N’ then it must be padded with spaces.

6.2.5. ***Hit & Take Order***

6.2.5.1. **Initial Verifications**

The sending of the Hit & Take Order message to the OASIS Server through the HitAndTakeOrderEntry object should be done only if the order to be hit is still Open in the Market (OrderStatus = “O”).

In order for the Member’s application to be informed of the available orders that can be hit, it should use the Hit & Take Order Information messages (CH).

Also, the system should not be Halted (no CB message with Status = “H” has been received).

Next, the limitations / verifications that need to be performed for each property of the HitAndTakeOrderEntry object are given.

6.2.5.2. MessageSource

MessageSource = “C”. All messages sent through the CTCI – ATHEX Gateway should have the value “C” in the MessageSource field.

6.2.5.3. MemberID

The Member code set forth by the ATHEX is filled in. In order for a Member to be able to send orders it has to be active.

6.2.5.4. TraderID

The code of a Member’s **active CTCI user.** Such user is set by the ATHEX.

6.2.5.5. VenueID

This should be a valid venue id as specified in the International Standard ISO 10383.

6.2.5.6. BoardID

The code of the board in which, the order will be entered. For hit & take orders, the following values are allowed:

- “S”: Special conditions board
- “F”: Forced sales board

This Board should have a Status = Open (Status = “O”).

Appendix C presents the orders allowed in the form of a table depending on a board type.

6.2.5.7. CSDAccountID

It should be completed with the code of the client who gave the order. All characters must be upper case.

6.2.5.8. GOIFlag

This field is not used anymore. The Group Of Inverstor functionality is removed.

6.2.5.9. ShortSellFlag

This attribute indicates whether the order is a short sell order or a buy to cover order or none of the above. It may take the following values:

- “N”: Normal

- “Y”: Short Sell / Buy To Cover

6.2.5.10. **HitOrderNumber**

It is the Order Number of the order to hit.

6.2.5.1. **HitOrderDate**

It is the Order Entry date of the order to hit.

6.2.5.2. **SecurityID**

It is completed with the identification of the security for which the hit & take order is being entered. The class or source of the SecurityID is identified by the SecurityIDSource field (see bellow). The security needs to be active (Security Status = “A” or “R”) and it should be in a trading phase which allows the entering of an order (PhaseID = “P”, “O”, “T”, “C”). At any time the Member’s application should know the status of a new security by using the information it receives through the Security Status (CA) messages.

6.2.5.3. **SecurityIDSource**

It is completed with the class or source of the SecurityID. It may take the following values:

- “8”: Exchange Symbol
- “A”: Bloomberg Symbol.

6.2.5.4. **Currency**

This attribute indicated the currency for order. It is used in messages directed to Xorder Server.

6.2.5.5. **Volume**

It is completed with the volume to be hit. The Volume, DisclosedVolume, SpecialCondition and ConditionVolume of the order to be hit, as they are contained in the corresponding Hit & Take Information message (CH message), must be respected.

6.2.5.6. **ClientOrderID**

This attribute should be completed using 16 characters and it is intended for internal use by the member.

6.2.5.7. **OrderNote**

This attribute should be completed using 25 characters and it is intended for internal use by the member.

6.2.5.8. **BuyClearingMemberID**

This attribute should be completed using 4 characters indicating the clearing sub-account ID of the buyer.

6.2.6. ***Quote Entry/Change***

6.2.6.1. **Initial Verifications**

The sending of the QuoteEntryChange message to the OASIS Server through the QuoteEntryChange object should be done only if the system has not Halted (namely no CB message with Status=”H” has been sent).

This message is used in order to:

- enter a new quote on a specific security.
- change an existing quote on a specific security.

Once a quote on a specific security is entered, it is assigned a quote identification (QuoteID field) which is communicated to the member (market maker) through the QuoteStatusReport (TA message) returned as confirmation to the initial QuoteEntryChamge message.

If the member wishes to change or cancel (see paragraph 6.2.7) an existing quote, then he should reference to this quote using this identification.

Next, the limitations / verifications that need to be performed for each property of the QuoteEntryChange object are given.

6.2.6.2. **MessageSource**

MessageSource = “C”. All messages sent through the CTCI – ATHEX Gateway should have the value “C” in the MessageSource field.

6.2.6.3. **MemberID**

The Member code set forth by the ATHEX is filled in. In order for a Member to be able to send a quote on a specific security, it has to be active and be assigned price quotation reposibilities for this security.

6.2.6.4. **TraderID**

The code of a Member’s **active CTCI user.** Such a user is set by the ATHEX.

6.2.6.5. **VenueID**

This should be a valid venue id as specified in the International Standard ISO 10383.

6.2.6.6. **Side**

This property indicates which sides of the quote are filled and may take one of the following values:

- “B”: buy side only
- “S”: sell side only
- “ “: both sides

6.2.6.7. CSDAccountID

It shall be completed with the code of the client. All characters must be upper case.

6.2.6.8. SecurityID

It is completed with the identification of the security for which the quote is being entered. The class or source of the SecurityID is identified by the SecurityIDSource field (see bellow). The security needs to be active (Security Status = “A” or “R”) and it should be in a trading phase which allows the entering of an order (PhaseID = “P”, “O”, “T”, “C”). At any time the Member’s application should know the status of a new security by using the information it receives through the Security Status (CA) messages.

6.2.6.9. SecurityIDSource

It is completed with the class or source of the SecurityID. It may take the following values:

- “8”: Exchange Symbol
- “A”: Bloomberg Symbol.

6.2.6.10. Currency

This attribute indicated the currency for this quote. It is used in messages directed to Xorder Server.

6.2.6.11. BuyPrice

This property is completed with the price of the buy side of the quote. This price should be a valid price tick (see Paragraph 5.1.1) and it should be equal to or more than the price floor and less than or equal to the price ceiling.

6.2.6.12. BuyVolume

This property is completed with the disclosed volume of the buy side of the quote. If the BoardID property mentioned earlier is “M” and thus we enter an order in the main board, the Volume property must be a multiple of the security Lot Size (see Paragraph 6.1.2).

Otherwise if the BoardID property mentioned is “O”, the Volume property needs to be less than the security Lot Size and greater than 0.

A Mixed Lot Volume (a volume greater than the Lot Size but not an integral multiple) is not allowed.

6.2.6.1. SellPrice

This property is completed with the price of the sell side of the quote. This price should be a valid price tick (see Paragraph 5.1.1) and it should be equal to or more than the price floor and less than or equal to the price ceiling.

6.2.6.2. SellVolume

This property is completed with the disclosed volume of the sell side of the quote. If the BoardID property mentioned earlier is “M” and thus we enter an order in the main board, the Volume property must be a multiple of the security Lot Size (see Paragraph 6.1.2).

Otherwise if the BoardID property mentioned is “O”, the Volume property needs to be less than the security Lot Size and greater than 0.

A Mixed Lot Volume (a volume greater than the Lot Size but not an integral multiple) is not allowed.

6.2.6.1. QuoteID

This attribute should be a valid quote identification (in case of quote change) or shall be left blank in case of new quote.

A valid quote identification shall be returned in a previous QuoteStatusReport message when a quote for the specific security has been initially entered.

6.2.6.2. QuoteMsgID

This attribute should be completed using 16 characters and it is intended for internal use by the member.

6.2.6.3. QuoteNote

This attribute should be completed using 25 characters and it is intended for internal use by the member.

6.2.6.4. ClearingMemberID

This attribute should be completed using 4 characters indicating the clearing sub-account ID.

6.2.7. *Quote Cancel*

6.2.7.1. Initial Verifications

The sending of the QuoteCancel message to the OASIS Server through the QuoteCancel object should be done only if the system has not Halted (namely no CB message with Status=”H” has been sent).

This message is used in cancel an existing quote.

Once a quote on a specific security is entered, it is assigned a quote identification (QuoteID field) which is communicated to the member (market maker) throuht the QuoteStatusReport (TA message) returned as confirmation to the initial QuoteEntryChamge message.

If the member wishes to cancel an existing quote, then he should reference to this quote using this identification.

Next, the limitations / verifications that need to be performed for each property of the QuoteCancel object are given.

6.2.7.2. **MessageSource**

MessageSource = "C". All messages sent through the CTCI – ATHEX Gateway should have the value "C" in the MessageSource field.

6.2.7.3. **MemberID**

The Member code set forth by the ATHEX is filled in. In order for a Member to be able to send a quote on a specific security, it has to be active and be assigned price quotation reposibilities for this security.

6.2.7.4. **TraderID**

The code of a Member's **active CTCI user.** Such a user is set by the ATHEX.

6.2.7.5. **VenueID**

This should be a valid venue id as specified in the International Standard ISO 10383.

6.2.7.6. **SecurityID**

The value of this attribute should be the same as that of the equivalent attribute of the quote that we want to cancel.

6.2.7.7. **SecurityIDSource**

The value of this attribute should be the same as that of the equivalent attribute of the quote that we want to cancel

6.2.7.8. **Currency**

The value of this attribute should be the same as that of the equivalent attribute of the quote that we want to cancel.

6.2.7.9. **QuoteID**

This attribute should be identification of the quote that we want to cancel

6.2.7.10. **QuoteMsgID**

This attribute should be completed using 16 characters and it is intended for internal use by the member.

6.2.7.11. QuoteNote

This attribute should be completed using 25 characters and it is intended for internal use by the member.

6.3. Make use of the information sent by OASIS Server

The Member's application should use the information it receives from the OASIS Server in order to know the status of an order it has sent, cancelled or changed, the trades made, the system, markets and securities status, and finally the incorrect messages it has sent and the reasons for which these were rejected by the OASIS Server.

6.3.1. *Confirmation messages (TB, TC, TD, TF)*

Paragraph 4.2.3 described the confirmation messages sent by the OASIS Server to the Member's application.

The Member's application must:

- Manage the confirmation or rejection for each message it has sent.

- Use the confirmation messages so that it knows the status of the orders/quotes sent to the OASIS Server. In this way, through the TB, TC, TD, TF, TA messages, it is able to know the "evolution" of an order/quote (i.e. entered, partially executed, changed, cancelled), as well as its status at any given time (i.e. order elements, matched volume, remaining volume etc.). In this way the sending of incorrect messages is avoided (i.e. attempt to cancel a fully executed order). Through the TB, TC, TD & TF messages, it is also informed of the remaining credit limit value and so to avoid orders that will cause the exceeding of this limit.

- Use the confirmation messages so that it knows the status of the trade reports sent to the OASIS Server. In this way, through the TF messages, it is able to know if the requested trade report is entered/approved/disapproved/requested for cancel/expired.

6.3.2. *Associating trades to orders (TF)*

In general, any trade (TF) refers to the participating order through field OrderNumber (coupled with OrderDate). Due to their inherently composite nature, this does NOT apply to trades originating from quotes or standard combination orders.

To appropriately associate incoming trade messages to orders using exchange identifiers, the client application should adhere to the following logic. On reception of a (TF) inspect field OrdRelFlag and act accordingly:

(TF) OrdRelFlag	(TF) Index field	Map to

Normal – ‘N’	OrderNumber	(TB) OrderNumber
Quote – ‘Q’	OrderRefID	(TA) QuoteID *refers to the quotation responsibility
Combination – ‘C’	OrderRefID	(TB) OrderNumber *refers to the combination order number
Trade Report – ‘T’	TradeNumber	(TF) TradeNumber *refers to the trade number of the trade report

Note that in all cases OrderDate should be used in conjunction to “Index Field” for appropriate access across difference trading dates.

Alternatively, client applications may make use of field **ClientOrderID** that is optionally supplied during order/quote/trade report entry and returned with subsequent confirmations. Note that in trades (TF), field ClientOrderID refers to the origination quote responsibility and combination order in respective cases. Finally, since field ClientOrderID is entirely at the discretion of the client application, caution should be exercised in supplying unique values as appropriate.

6.3.3. *Status messages (CA, CB, CC, CD, TL, CH, TO)*

Paragraph 4.2.3.5 described the status messages sent by the OASIS Server to the Member’s application.

The Member’s application must:

- Use the Security Prices messages (CD) so that it is able to know the change in the allowed price fluctuation percentage, if this has been made during the current session.

- Use the CA, CB and CC status messages (CB messages not issued with respect to Xnet) so that it is able to know the system status, as well as the markets and securities status. In paragraphs 6.2.1, 6.2.2 and 6.2.3 we mentioned that, before sending a message (MB, MC, MD, MI, ML, MF), the system, markets, boards and securities status should be verified. The paragraphs that follow will examine in detail the time when these messages are sent.On XNET interface a CC (Market Status) message will be sent for each currency supported on each Venue Id.

- Use the Credit Limit Information messages (TL) so that it is able to know its daily trading session credit limit.

- Use the Hit & Take Order Information messages (CH) so that it is able to know the available orders that can be hit with a Hit & Take Order message (MF).

- Use the Exchange Notes messages (TO) to receive useful information from ATHEX.

6.3.3.1. Sending status and information messages

6.3.3.2. System status change, credit limit information and security information

Figure 6-1 presents the transitions of the system among the various possible statuses, as well as the messages sent by the OASIS Server through the ATHEX Gateway.

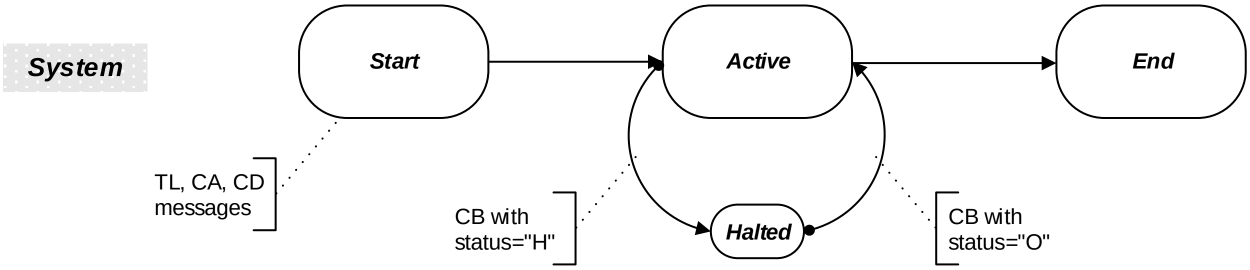


Figure 6-1 System Status Change

A **prerequisite** for the sending of a message to the OASIS Server is **that the system has an "Active" status.** Thus, if the Member's application has received a CB message with Status "H", the sending of messages should be interrupted until the reception of a CB message with Status "O". CB messages are not issued with respect to Xnet.

6.3.3.3. Market/Board status change (CC)

In order to send a new order, to edit an order or to change an order through the MB, MF, MC and MD messages or to perform a trade report (preagreed price trade) through the MI message, **the board in which the order will be entered must have a status which allows it to accept orders.** Specifically its status should not be: "R", "E", "H", "S" if it is a main board and it should be "O" (open) if it is not a main board. Otherwise the message will be rejected.

Figure 6-2 presents the transition of the various types of board among the various statuses during a typical trading session, as well as the CC messages sent to the Members applications. These figures also show the CA messages sent and which concern the securities negotiated in the various boards. In the paragraph that follows, we will examine in more detail the CA messages. CA messages are not issued with respect to Xnet.

The Market timing (namely the time when the statuses of the various boards change) is given in Appendix G, OASIS Server parameters, Paragraph 13.3.

Market

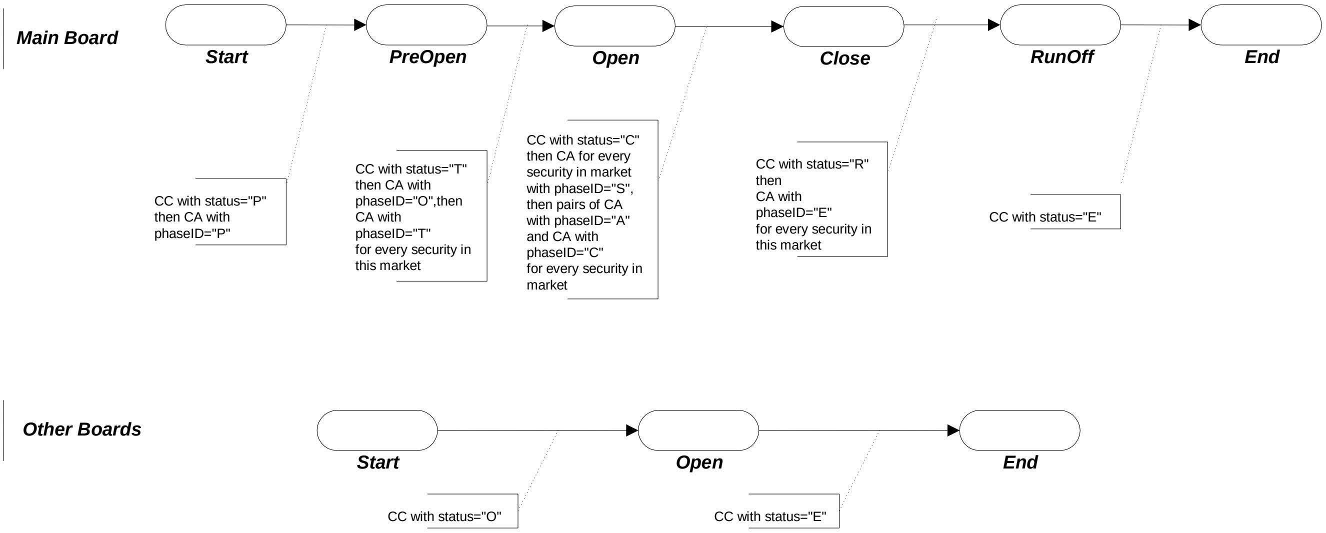

Main Board
Start
PreOpen
Open
Close
RunOff
End
CC with status="P" then CA with phaseID="P"
CC with status="T" then CA with phaseID="O",then CA with phaseID="T" for every security in this market
CC with status="C" then CA for every security in market with phaseID="S", then pairs of CA with phaseID="A" and CA with phaseID="C" for every security in market
CC with status="R" then CA with phaseID="E" for every security in this market
CC with status="E"
Other Boards
Start
Open
End
CC with status="O"
CC with status="E"

Figure 6-2, Boards Status Change

It should be noted that

Figure 6-2 describes the Boards status change during a typical trading session. Should for any reason the operation of a Market be temporarily halted, a CC message will be sent with Status = “H”. In this case no MB, MC, MD, MF, MI or ML messages should be sent for that specific market and the application must wait until the reception of a message with a new Status.

6.3.3.4. Security status change (CA)

Each security monitors the status of the Main Board of the Market in which it belongs and it is for that reason that each time that the status of a market’s main board changes, this market’s securities’ Phase IDs change as well.

Figure 6-2 shows the CA messages sent during the board transitions

A security may be totally suspended or halted, should the ATHEX surveillance deem that such action is necessary. The difference between total suspension and halting lies in the fact that the duration of halting can not exceed one day, in contrast with the duration of the total suspension. CA messages with SecurityStatus = “S” or “H” respectively are sent in order for the Members applications to be informed.

In both statuses the system **does not allow the cancellation and inactivation** of orders for the given security. The cancellation of trades by the surveillance is permitted. Alternatively, the given security might be assigned a non active status during which no action whatsoever may be taken with regard to the said security. In this case, a CA message with SecurityStatus= “N” is sent.

Upon withdrawal of a suspension or non active status, the security reverts to its previous phase before the suspension / deactivation. The withdrawal is either made automatically (resume) in the event of a temporary suspension, or manually (active) in the other two cases. It is possible for a security to transit from a suspension status to another suspension status or even to a non active status. Furthermore, a security may revert from the non active status to a suspension status.

Thus, no assumption may be made with regard to the possible sequence / correlation of these statuses of a security and **the general tactics that always maintain the last status of each security should be followed** and upon that information the decision of whether an attempted action would or would not be accepted should be based.

CA messages are not issued with respect to Xnet.

6.3.4. *Reject message (TR)*

The purpose of the Reject message is to inform the Member application that the message received by the OASIS Server has not been accepted.

The programmer should always respect and **use the error code** which may possibly be returned by any call to the ATHEX Gateway. It is always bad practice to ignore these codes because they either lead to the termination of the application (if not handled correctly) or they lead to the unnecessary execution of larger parts of code, until the error is detected somewhere else. A possible side effect of the non timely diagnosis of such errors is also the increased number of rejections, because a program that ignores rejections will probably continue to send invalid messages. This will have a negative effect on the performance of the ATHEX Gateway and the OASIS Server.

Paragraph 3.10 gives technical details regarding the handling of errors.

NOTE

For more information on the business logic, please refer to ATHEX Rulebook:

http://www.ase.gr/content/en/about/regulations/files/ATHEX%20final.pdf

7. Technical information and guidelines

7.1. General infrastructure of ATHEX Gateway

A schematic presentation of the ATHEX Gateway with a primary-backup setup, is given below:

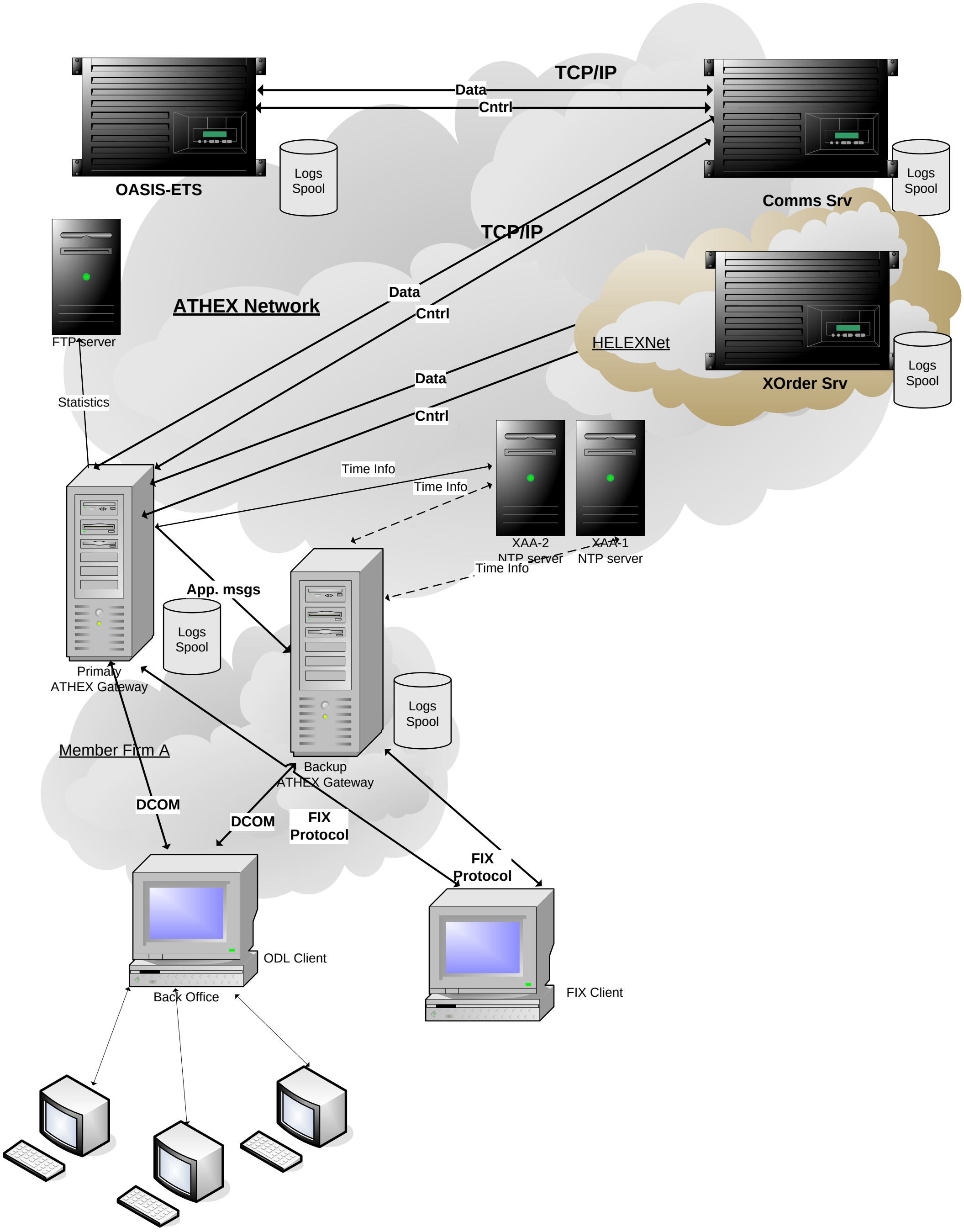


Figure 7-1. Schematic diagram of the ATHEX Gateway

The above figure shows a Member (Member A) using the ATHEX Gateway:

- **Member A** uses a primary – backup configuration and thus takes advantage of the fault tolerance capability of the ATHEX Gateway. The backup ATHEX Gateway is always aware of the status of the execution in the primary ATHEX Gateway, so in case of an operation failure of the primary ATHEX Gateway, this second server can continue the operation of the service **without any significant time delay**; just by switching its operating mode to primary.

- **A Member may also** use a primary – standalone configuration not depicted above. In this case no extra backup server is used. In the event of an operation failure of the ATHEX Gateway, the member should solve the problem (repair or substitute the ATHEX Gateway) and then reconnect to the Comm Server and/or the Xorder server. All messages would then be recovered from OASIS and/or the Xorder server and the ATHEX Gateway could continue its operation.

- Finally, a multiple gateway configuration (not depicted above) can also be applied. Any number of gateways may be utilized concurrently and each will be a recipient of all incoming messages. The facilitation of this model lies on the discretion of the member. For example, a member may:
    - Facilitate all gateways concurrently to implement
        - Load balancing.
        - Dedicated channels for particular sets of distinguished transactions
    - Only connect to a second gateway in case the first is non-functional for any reason.

7.1.1. *Time synchronization of the ATHEX Gateway with OASIS.*

The ATHEX Gateway (primary and backup) must synchronize its clock with the OASIS. The Network Time Protocol (NTP) –which is an internet standard- is designed to synchronize the clocks of computers over a network. An NTP client must be installed and configured appropriately in every ATHEX Gateway to achieve the synchronization with the ATHEX NTP servers (as shown in Figure 7-1) over the ATHEXnet.

7.1.2. *Statistical information generation and transfer*

Each trading day, statistical information is generated in the ATHEX Gateway. The statistics are related to the number of transaction/queries executed by the given ATHEX Member using the ATHEX Gateway as well as the time required to complete them. The statistical information file is produced at the end of the trading day to the ATHEX FTP server (using the FTP protocol) and can be used for further analysis. This analysis can lead to the continuous improvement of the ATHEX Gateway. This is shown in Figure 7-1 above.

7.2. Best practices for the development of ODL client

During application development with the use of the ODL Client API, there are certain points that must be taken into account for the development of efficient applications. In an ever evolving demanding environment, as is this of an ATHEX Member, the optimal performance of the software is of particular importance, so that maximum functionality is achieved. Some general programming instructions follow:

1. Before attempting to design the application to be used, it is appropriate that a detailed study of the accompanying material and examples has been completed, so that a clear view of the behaviour of the ATHEX Gateway exists.

2. The programmer is provided with a set of events which indicate certain statuses occurring asynchronously during the use of the ATHEX Gateway. It is always good programming practice to limit to the minimum the time period during which the application remains in the code that handles these events because:

    a. The simultaneous serving of a larger number of events is facilitated.

    b. Situations in which one event occurs during the serving of the other are avoided. This case of overlapped events, if not properly handled, may lead to various serious errors in the application, as well as in a possible loss / corruption of data.

    The application that will use the ATHEX Gateway should be designed so that it will be prepared to receive and to serve an incoming event at any time.

3. Along with the primary information contained in the objects determined by the ATHEX Gateway, it is good practice that the programmer also keeps additional information which will make his search of specific information produced by the primary information faster. Such a possible example is the set of executed parts in an order.

4. The programmer should always respect and use the error code which may be possibly returned by any call to the ATHEX Gateway.

5. Due to the relatively large number of information transmitted through the ATHEX Gateway to the Member firm, it is appropriate to use smart and mainly fast classification and search manners for information concerning the basic entities for trading, such as orders and trades.

6. The checks should be performed exactly before the sending of messages because the environment conditions evolve in real time.

7.3. ODL Client Application Development Environment

The development of the member’s applications (ODL Clients) will be performed in a certain computer and network environment. The main features of this environment will be described below. The existence and operation of the servers for testing environment is presumed.

7.3.1. *Testing Environment*

The testing environment from the member’s side is at least a set of the following elements:

- The computer system GATEWAY 1 (ODL-Server Test), which has two network connections, one to link the computer to the member's intranet and one to link the computer to the internal ATHEX network.

- The test terminal OASIS/ETW-ORAMA, which will be connected to the central test system OASIS via the ATHEX internal network.

- A computer with Microsoft Windows Operating System, where the ODL-Client-API is installed and will be the contact point between ODL and the member's application. This computer will be linked only to the member's intranet and will be communicating with the GATEWAY 1 system, described above.

Attention: The usage and operation of such an environment should be constant, in order to constantly provide results that can be compared to previous test runs. This will help the enhancement and evolultion of the members' applications. The setup of the environment described above is performed, based on special technical specifications and instructions, which are being supplied by ATHEX, based on relevant ATHEX decisions.

7.4. FIX Client Application Development

ASE Service (the G/W component handling FIX connections) is designed to perform no buffering of incoming (client to exchange) messages. It is however dependent upon G/W capacity (and overall system availability) to forward messages to the exchange.

In non-common instances of reduced system performance due to stress or technical issues, it may be the case that the ATHEX G/W will not be able to transmit messages to the exchange. Under such circumstances messages sent by the client are subject to buffering *on TCP level* (TCP buffer). It is therefore possible that a client application may receive TCP level rejections (e.g. error code returns to a Send() call of the client, see WSAEWOULDBLOCK error 10035).

The FIX client should be capable of appropriately handling TCP errors and retransmit orders or adjust transmission rates accordingly.

7.5. Guidelines summary

In short, applications should take into account and appropriately handle:

1. The **architecture of the ATHEX Gateway** and its **usage instructions**.

2. The **configuration** of the market (at all relevant levels: system, market, securities, members, users etc.)

3. The **status** of the system, markets, boards and securities at any time.

4. The **possible values** that the parameters could take at any time.

5. The system's **confirmation messages**.

6. Messages should be sent to the system in accordance with the system schedule (at the relevant levels of: system, markets, boards etc.). The schedule is given at a timing level by the system configuration. However, since due to the system load there are some small time delays (usually some tenths of a second), and also because it is possible to dynamically change the system timing from the ATHEX, it is more correct that the Member's application be informed about the system, markets and securities status by the status messages received to that end. Thus, an efficient application should be designed **with an event driven and not time driven logic**.

7. The **rejects** sent by the OASIS and/or Xorder server.

8. The **syntactic correctness and the possible values** of the fields of the messages sent to the system.

9. API error **return codes** including TCP level return codes (FIX only).

Practically, **all of the above aim at** minimizing to the greatest possible extent the rejections sent by the central system and the load that these might constitute for the common network and computer infrastructure of the ATHEX.

For the efficient use of the ATHEX Gateway for connection of ATHEX member's applications to the OASIS, it must be noted that:

1. OrderNumber and OrderDate fields are the only reference keys for a specific order. The OrderDate value can take the date of the previous day in case of Life Orders.

2. Member's applications can be:

   a) Management Applications using an Administrator object for ATHEX Gateway management, connection to Comm Server and reception of the whole set of information (even from previous days) concerning the member.

   b) Order Applications using a Broker object for order insertion, change, cancel and monitoring.

   c) FIX client Applications that do not make use of the ODL-Client-API.

3. In a multiple concurrently active Gateway scheme, an application that is connected to more than one gateways simultaneosuly should expect this callback to be executed for the same message as many times as the number of gateways the application is connected to.

For the efficient use of the ATHEX Gateway for connection of member's applications to the Xorder server and Xnet, it must be noted that:

1. Not all operations are defined throughout different exchanges and different policies may apply with respect to response and unsolicited messages.

8. Appendix B. ODL-Client-API: ETS_Broker properties

In this appendix there is a detailed description of the object fields that exist on the ODL-Client-API application. The objects were described on Chapter 4.2.3 Application Requests objects.

8.1. OASIS and Xnet Interface

This chapter describes the fields of the objects when the corresponding messages are directed to or received from the OASIS and the Xnet interfaces only.

Field	Description
AccruedInterest	A 14-character numeric field (8.6) that indicates the accrued interest of a bond.
ActionSource	A 1-character alpha source-type field (see field 'Source') incidating the source of action for a message.
AlarmReason	A 1-character alpha field incidating the reason of a quote alarm. Possible values: '1' Prices are missing '2' BID price is missing '3' ASK price is missing '4' The Price Spread is too big '5' Quantities are too small '6' BID Quantity is too small '7' ASK Quantity is too small
AlarmType	A 1-character alpha field incidating the type of a quote alarm. Possible values: 'W' Warning 'A' Alarm
AnnualPayments	A 3-character numeric field used to indicate the bond's annual payments.
AutoDisclosedVolume	A 12-character volume-type numeric field used to indicate the portion of the total volume to be disclosed automatically in an undisclosed volume order. Reserved for future use.

AveragePrice	A 9-character numeric price-type field indicating the daily average trading price of an order. The format of this field is 5.4
BestBidPrice	A 9-character numeric price-type field indicating the best bid price. The format of this field is 5.4
BestBidQuantity	A 12-character volume-type numeric field used to indicate the best bid volume.
BestOfferPrice	A 9-character numeric price-type field indicating the best offer price. The format of this field is 5.4
BestOfferQuantity	A 12-character volume-type numeric field used to indicate the best offer volume.
BoardID	(OASIS): A single character alpha field that identifies the trading board. Possible Values: "M" Main board "S" Special conditions board "B" Report Only board "F" Forced sales board (ORA/Xnet): Fixed 'M'
BuyClearingMemberID	A 4-character alphanumeric field indicating the clearing sub-account ID of the buyer side for a pre-agreed price trade.
BuyClientID	A 10 character numeric field indicating the short code used to identify the buy-side client of the member or participant of the trading venue. Valid values range between 0 (denoting no short code entry) and 4294967295.
BuyClientIDQualifier	A single character alphanumeric field to further qualify the BuyClientID field. Possible values : 'A' Algorithm 'L' Firm or legal entity 'N' Natural person " " Not applied when field ClientID = 0 (NONE) or 1 (INTC) or 2 (PNAL)
BuyClientOrderID	A 16-character alphanumeric field indicating the buyer members order number in a trade report. The member is responsible for the accuracy and validity of this field.

BuyExecutionWithinFirmID	A 10 character alphanumeric field indicating the short code used to identify the buy-side person or algorithm within the member or participant of the trading venue who is responsible for the execution of the transaction resulting from the order. Valid values range between 0 (denoting no short code entry) and 4294967295
BuyExecutionWithinFirmIDQualifier	A single character alphanumeric field to further qualify the BuyExecutionWithinFirmID field. Possible values : 'A' Algorithm 'L' Firm or legal entity 'N' Natural person " " Not applied when field ExecutionWithinFirmID = 0 (NONE) or 3 (NORE)
BuyGOIFlag	This field is not used anymore. The Group Of Inverstor functionality is removed.
BuyInvestmentDecisionID	A 10 character alphanumeric field indicating the short code used to identify the buy-side person or the algorithm within the member or participant of the trading venue who is responsible for the investment decision. Valid values range between 0 (denoting no short code entry) and 4294967295
BuyInvestmentDecisionIDQualifier	A single character alphanumeric field to further qualify the BuyInvestmentDecisionID field. Possible values : 'A' Algorithm 'L' Firm or legal entity 'N' Natural person " " Not applied when field InvestmentDecisionID = 0 (NONE)
BuyMemberID	A 4-character alphanumeric field indicating the buyer member in a trade report.
BuyNonExecutingBrokerID	A 10 character numeric field indicating the short code used to identify the buy-side person or participant of the trading venue who routed the order on behalf of and in the name of another member or participant of the trading venue. Valid values range between 0 (denoting no short code entry) and 4294967295.

BuyTraderID	A 5-character alphanumeric field that identifies the specific user from the buyer side in a trade report. This field identifies the individual trader authorized to use the system, as registered with OASIS.
BuyPrice	A 9-character numeric field that indicates the price of the buy side of a quote. Format is 5.4. Possible values: 0 <= n <= 99,999.9999 In cases where negative pricing is applicable (combination trading) the sign (-) is incorporated in this field, reducing the numeric part to 4.4 format. The negative sign should occupy the first character of the integer part, but should be omitted in positive values. E.g a price of "-1.23" translates to "-00012300" while possible values are: -1,000.0000 <= n <= 99,999.9999, excluding zero (reserved).
BuyRemainingVolume	A 12-character volume-type field indicating the buy side volume of a quote available in the market.
BuyToCoverFlag	A single character alpha field, indicating if the trade report is declared as buy to cover. Possible values: "N" No "Y" Yes (Buy to cover)
BuyVolume	A 12-character field indicating the disclosed volume of the buy side of a quote are available in the market.
CancelReasonCode	A single character alpha field used to indicate the reason for order cancellation. Possible values are: "C" canceled by CTCI request "M" canceled by ETW request "R" canceled by EMRW request "I" cancellation of IOC order "F" cancellation of FOK order "P" cancellation of MKT order "O" cancellation of ATO order "L" cancellation of Life order "S" cancellation of Covered Sale Order by CSD

CancelSource	A 1-character alphanumeric field to identify the source of an order cancel. Possible values : “C” CTCI “M” ORAMA – ETW (Market Works station) “R” EMRW (Control regulation workstation) “ “ OASIS
CeilingPrice	A field indicating the minimum possible price of the instrument during the day.
ChangedAutoDisclosedVolume	A 12-character numeric field used to indicate the new (after an order change) portion to be disclosed automatically in an undisclosed volume order. Reserved for future use.-character
ChangedClearingMember ID	A 4-character field alphanumeric used for specifing the new (after order change) clearing sub-account ID.
ChangedClientID	A 10 character numeric field indicating the short code used to identify the client of the member or participant of the trading venue. Valid values range between 0 (denoting no short code entry) and 4294967295.
ChangedCSDAccountID	A 12-character alpha field describing the new (after an order change) user account of the order. All characters must be upper case.
ChangedClientIDQualifier	A single character alphanumeric field to further qualify the ChangedClientID field. Possible values : ‘A’ Algorithm ‘L’ Firm or legal entity ‘N’ Natural person “ “ Not applied when field ClientID = 0 (NONE) or 1 (INTC) or 2 (PNAL)
ChangedDirectElectronicAccess	A single character alpha field indicating whether the order was submitted to the trading venue using Direct Electronic Access. Possible values : “0” False “1” True

ChangedDisclosedVolume	A 12-character field indicating the new (after an order change) volume of an order actively trading in the market. This field is mainly used in cases when only a portion of the total volume of an order is to be traded on entry. For valid value range refer to field ‘Volume’.
ChangedExecutionWithinFirmID	A 10 character alphanumeric field indicating the short code used to identify the person or algorithm within the member or participant of the trading venue who is responsible for the execution of the transaction resulting from the order. Valid values range between 0 (denoting no short code entry) and 4294967295
ChangedExecutionWithinFirmID Qualifier	A single character alphanumeric field to further qualify the ChangedExecutionWithinFirmID field. Possible values : ‘A’ Algorithm ‘L’ Firm or legal entity ‘N’ Natural person “ “ Not applied when field ExecutionWithinFirmID = 0 (NONE) or 3 (NORE)
ChangedExpirationDate	An 8-character field date-type(format YYYYMMDD) indicating the new (after an order change) date that the order will be disposed from the system. This field is valid for orders having the field ChangedLife set to «E».
ChangedGOIFlag	This field is not used anymore. The Group Of Inverstor functionality is removed.
ChangedInvestmentDecisionID	A 10 character alphanumeric field indicating the short code used to identify the person or the algorithm within the member or participant of the trading venue who is responsible for the investment decision. Valid values range between 0 (denoting no short code entry) and 4294967295
ChangedInvestmentDecisionIDQualifier	A single character alphanumeric field to further qualify the ChangedInvestmentDecisionID field. Possible values : ‘A’ Algorithm ‘L’ Firm or legal entity ‘N’ Natural person “ “ Not applied when field InvestmentDecisionID = 0 (NONE)

ChangedLife	A single character type alpha field indicating the life time of an order. Possible values : “D” Day order “C” Good until Canceled “E” Good until Expiration date
ChangedNonExecutingBrokerID	A 10 character numeric field indicating the short code used to identify the person or participant of the trading venue who routed the order on behalf of and in the name of another member or participant of the trading venue. Valid values range between 0 (denoting no short code entry) and 4294967295
ChangedOrderNote	An 25-character alphanumeric field used for order notes.
ChangedOrderStatus	A 2-character alphanumeric field indicating the status of an order. Possible values : “N “ Not Released “I “ Inactive “O ” Open “M ” Match “X ” Cancel “EP” GTC, GTD expired status “A ” Pending for approval
ChangedOriginalPriceType	An single character alphanumeric field indicating the type of the order. Possible values: “L” Limit orders “O” At The Open “M” Market Price “C” At The Close (Reserved for Future Use)
ChangedPositionEffect	A single character alpha code that identifies the position effect . For use in derivatives omnibus accounting. This field must contain a valid value in all cases; however it will be ignored in non-derivative products. Indicates whether the resulting position after a trade should be an opening position or closing position.Possible values are: “O” Open

	“C” Close (Netting)
ChangedSettlType	A single character alpha code that indicates the order settlement period. Possible values are: “1”: Immediate Settlement “0”: Normal settlement period
ChangedPrice	A 9-character numeric field indicating the price of an order or trade. Format is 5.4. Possible values: 0 <= n <= 99,999.9999 In cases where negative pricing is applicable (combination trading) the sign (-) is incorporated in this field, reducing the numeric part to 4.4 format. The negative sign should occupy the first character of the integer part, but should be omitted in positive values. E.g a price of “-1.23” translates to “-00012300” while possible values are: -1,000.0000 <= n <= 99,999.9999, excluding zero (reserved).
ChangedShortSellFlag	A 12-character alphanumeric field, indicating if the order is a short sell / buy to cover. Possible values: “N” Normal “Y” Short Sell / Buy To Cover
ChangedSource	A single character alphanumeric field indicating the source of the order or trade. Possible values : “C” CTCI –ODL “M” ORAMA-ETW “R” EMRW (ATHEX supervision application) “ “ OASIS
ChangedVolume	A 12-character numeric field indicating the new order volume. Possible values : 0 < n <= 999,999,999,999
ClearingMemberID	A 4-character alphanumeric field indicating the clearing sub-account ID.
ClearingSpace	A 4-character alphanumeric field indicating the clearing space.
ClearingSubAccountId	A 4-character alphanumeric field indicating the clearing sub-account ID.
ClientID	A 10 character numeric field indicating the short code used to identify the client of the member or participant of the trading venue. Valid values range between 0 (denoting no short code entry) and 4294967295

ClientIDQualifier	A single character alphanumeric field to further qualify the ClientID field. Possible values : 'A' Algorithm 'L' Firm or legal entity 'N' Natural person " " Not applied when field ClientID = 0 (NONE) or 1 (INTC) or 2 (PNAL)
ClientOrderID	A 16 character alphanumeric field indicating the members order number. The member is responsible for the accuracy and validity of this field. In trade reporting, the field ClientOrderID is also generated by the trading system to identify an outgoing TF message.
ClientTradeID	A 16 character alphanumeric field indicating the member's trade report number. The member is responsible for the accuracy and validity of this field.
ClosingPriceMethod	A single character numeric field indicating the closing price calculation method for given security. '1'. Last Trade '2'. Weighted Average of a fixed number of last trades '3'.Weighted Average of a percentage of last trades '4'. Weighted Average of trades concluded in a predetermined period of time before the end of the session '5'. Weighted Average of a percentage of the daily trading volume '6'. High Frequency '7'. Medium Frequency '8'. Low Frequency For more information see ATHEX Rulebook
ConditionVolume	A 12 character alphanumeric field indicating the condition volume in a MF or MOF order. Reserved for future use.
ContractSize	A 12 character volume-type (see field volume) field indicating the quantity of commodities underlying a derivative (future/option).
ContraMemberID	A 4-character alpha field indicating the counterparty member id of the trade or a series of 4 zeros (0) if the Exchange has deactivated this particular functionality.
ContraTraderID	A 5 character alpha field indicating the counterparty trader id of the trade or a series of 5 zeros (0) if the Exchange has deactivated this particular functionality.
CreditLimit	A 14-character numeric (12.2) field set from OASISindicating the Member's credit limit for a given credit limit method.

Currency	Used for XNET only. ISO 10383 compliant.
CurrentCreditValue	A 14-character numeric (12.2) field set from OASIS indicating the member's current credit value.
DayAvgTrdVal	A 14-character numeric (12.2) field indicating the average daily turnover for a specific security
DaysInYear	A 3-character numeric field indicating the days in year for bond.
DaysUntilNextPayment	A 5-character numeric field indicating the number of days until next payment for bond.
DirectElectronicAccess	A single character alpha field indicating whether the order was submitted to the trading venue using Direct Electronic Access. Possible values : "0" False "1" True
DisclosedVolume	A 12 characters field indicating the volume of an order actively trading in the market. This field is mainly used in cases when only a portion of the total volume of an order is to be traded on entry. For valid value range refer to field 'Volume'.
DuplFlag	A single character alphanumeric field that indicates whether this is a duplicative trade report. Possible values: '0' False '1' True
EditedDisclosedVolume	A 12-character field indicating the volume of an order actively trading in the market. This field is mainly used in cases when only a portion of the total volume of an order is to be traded on entry. For inactive orders, this represents the volume announced when the order becomes active. For valid value range refer to field 'Volume'.
EditType	A single character alpha field indicating the action to perform on the order. Possible values : "C" Cancel order "S" Suspend (deactivate) order "U" Unsuspend (activate) order
EditSource	A single character alpha field identifing the source of order edit.

	“C” CTCI “M” ETW (Market Works station) “R” EMRW (Control regulation workstation) “ “ OASIS
ExecutionWithinFirmID	A 10 character numeric field indicating the short code used to identify the person or algorithm within the member or participant of the trading venue who is responsible for the execution of the transaction resulting from the order. Valid values range between 0 (denoting no short code entry) and 4294967295
ExecutionWithinFirmIDQualifier	A single character alphanumeric field to further qualify the ExecutionWithinFirmID field. Possible values : ‘A’ Algorithm ‘L’ Firm or legal entity ‘N’ Natural person “ “ Not applied when field ExecutionWithinFirmID = 0 (NONE) or 3 (NORE)
ExerciseStyle	A single character alpha fied used only for options. Possible values: “0”: European “1”: American “ “: security is not an option
ExpirationDate	An 8-character Date type (format YYYYMMDD) indicating the date that an order will be disposed from the system. This field is valid for orders with OrderLifeTime field set to “E” (Good until expiration date).
FaceValue	A 9-character nummeric (7.2) field indicating the face value of bond.
FloorPrice	A field indicating the maximum price of the instrument during the day.
GOIFlag	This field is not used anymore. The Group Of Inverstor functionality is removed.
HaltReasonCode	A 2-character alphanumeric field indicating the cause of the halt or suspension. Possible values:

	01 Halt 02 Ceiling 03 Floor 04 War 05 Politics 06 Technical 07 Terrorism 08 Various 09 Religion 10 Celebration 11 Earthquake 12 Volatility Interruption 13 Series Expiration
HaltStartTime	A 12-character numeric field indicating the time that Halt/Suspend occured.
HitOrderNumber	A unique 8 digits number used to specify, along with the HitOrderDate, the order to be hit. Possible values: 1 <= n <= 99.999.999
HitOrderDate	A 8-character Date field (format YYYYMMDD) indicating the date that the order to be hit was recorded in OASIS for the first time. It is used by the trader to specify, along with the HitOrderNumber, the order to be hit.
InvestmentDecisionID	A 10 character alphanumeric field indicating the short code used to identify the person or the algorithm within the member or participant of the trading venue who is responsible for the investment decision. Valid values range between 0 (denoting no short code entry) and 4294967295
InvestmentDecisionIDQualifier	A single character alphanumeric field to further qualify the InvestmentDecisionID field. Possible values : 'A' Algorithm 'L' Firm or legal entity 'N' Natural person " " Not applied when field InvestmentDecisionID = 0 (NONE)

LastLiquidityIndicator	A single character alpha field indicating whether this trade was a result of a liquidity provider providing or liquidity taker taking the liquidity or auction or nothing of the above. Valid values are: ‘A’: Added Liquidity ‘R’: Removed Liquidity ‘N’ : Trade during auction ‘ ‘: not applicable (BoardID is not equal to ‘M’)
ListID	A 6 character alphanumeric field indicating a member-specific grouping of orders (clearing procedure instructions).
LeavesQuantity	A 12 character numeric field indicating the remainder quantity of a partially completed order.
LegRatio	A single character numeric field defining the ratio of the leg to other legs in the strategy
LegSecurityCode	A 12-character alphanumeric field of SecurityCode type, indicating the Security Code of combination product leg.
LegSecuritySymbol	A 15-character alphanumeric field of SecuritySymbol type, indicating the Security Symbol of combination product leg.
LegSide	A single character alphanumeric field that indicates if that particular leg will be bought or sold as part of the strategy (Operation on leg if buying the combination product)
LiquidityProvision	A single character alphanumeric field indicating whether an order is submitted to a a trading venue as part of a market making strategy. Possible values: ‘0’ False ‘1‘ True
MarketID	A single character alphanumeric field indicating the trading market.
MemberID	A 4 character alphanumeric field indicating the member sending/receiving the message to/from the target Venue.
MemberSequenceNumber	A 6 character numeric field indicating the unique sequence number assigned to the initial message by the ODL G/W. e.g. A new order message API call (SendMessage – MB) returns a unique value which is referenced through this field when the asynchronous order confirmation (TB) message is received.

MessageNote	A 50 character free text field containing a message from the exchange
MessageSource	A single character alpha field incidating the creator of the message. Possible values : ‘C’ from CTCI –ODL ‘ ‘ from OASIS
MessageType	A 2-character alphanumeric field indicating the type of the message.
NewOrderNumber	A unique 8 digits number set from OASISused to verify each order of the trading day. Possible values :: 1 <= n <= 99.999.999
NewOrderStatus	A 2-character alphanumeric field indicating the status of an order. Possible values : “N “ Not Released “I “ Inactive “O ” Open “M ” Match “X ” Cancel “EP” GTC, GTD expired status “A ” Pending for approval
NextPayDate	A 8-character Date field (format YYYYMMDD) indicating the next pay date for bond.
NonExecutingBrokerID	A 10 character numeric field indicating the short code used to identify the person or participant of the trading venue who routed the order on behalf of and in the name of another member or participant of the trading venue. Valid values range between 0 (denoting no short code entry) and 4294967295.
NoOfLegs	A single character numeric field defining the number of legs comprising a compbination product.
NoteType	“0”: Generic free text note from the exchange (default) “1”: Throttling Warning. Member CTCI rejections have reached half of total resulting to disabling of the member’s ODL. Action should be taken on the member’s behalf to elliminate excessive rejection causes.
NotionalAmmount	A 14-character numeric (12.2) field indicating trade value.

OptAttribute	A single character numeric field used for derivatives indicating a versioning of the contract when required due to corporate actions to the underlying.
OrderDate	A 8-character Date field (format YYYYMMDD) supplied by OASIS, indicating the date that an order was recorded in the trading system for the first time.
OrderLifeTime	A single character alpha field indicating the orders life time. Possible values : “D” Day order “C” Good until Canceled “E” Good until Expiration date (also refer to field ‘ExpirationDate’)
OrderNote	A 25-character alphanumeric field containing order notes.
OrderNumber	A unique 8 digits number set from OASISused to verify each order of the trading day. Possible values : 1 <= n <= 99.999.999
OrderRelFlag	A single character alpha field indicating the order relation. Possible values : ’N‘ Normal ‘Q’ Quote ‘C’ Combo ‘R’ Trade Report
OrderSource	A single character alphanumeric type indicating the source of the Order. Possible values : ‘C’ CTCI –ODL ‘M’ ORAMA-ETW ‘R’ EMRW (ATHEX supervision application).
OrderStatus	A 2-character alphanumeric field indicating the status of an order. Possible values : “ “ Not available (*) “N “ Not Released “I “ Inactive “O ” Open “M ” Match

	“X ” Cancel “EP” GTC, GTD expired status “A ” Pending for approval * The “ “ (not available) order status is used in cases of • Unsuccessful order entry • Unsuccessful edit/change due to incorrect order number.
OrderType	A single character alphanumeric field indicating the type of the order. Possible values : ”N” Active order, for automatic entry ”I” Inactive order, for automatic entry “M” Active order, for manual entry (only available for XNet orders) “X” Inactive order , for manual entry (only available for XNet orders)
OrdRefId	A 8 character alphanumeric field. Used in case trades are based on quote or combo ordrers. Maps to QuoteID for quote messages or Order ID (number) for combo orders. Otherwise left blank. When reffering to order number, current trading day is implied for the order referred.
OrigClientOrderID	A 16 character alphanumeric field indicating the original (current, at the time of message transmission) members order number. IF filled with blanks this field is ignored from OASIS. In trade reporting, the field ClientOrderID is also generated by the trading system to identify an outgoing TF message. Accordingly, OrigClientOrderID is used to associate to a previous ClientOrderID in a sequence of trade reporting related messages.
OriginalMessageText	A 400-character alphanumeric field that will contain the original message data when a message is returned unprocessed to the member firm.
OriginalPriceType	A single character alphanumeric field that indicates the type of order. Possible values: “L” For limit orders “O” At The Open “M” Market Price “C” At The Close

PastCashFlows	A 3-character numeric field indicating the bond’s past cash flows.
PhaseID	A single character alpha code that identifies the trading phase. Possible values are: “ ” Start of day (Before the Pre-opening) “P” Pre-opening Trading Phase “O” Opening Trading Phase “T” Continuous Trading Phase “A” At the Closing Price trading Phase “C” Closing Price Continuous Trading Phase “E” End Of Trading Phase “S” Stop phase (Use in auction market)
PositionEffect	A single character alpha code that identifies the position effect . For use in derivatives omnibus accounting. This field must contain a valid value in all cases; however it will be ignored in non-derivative products. Indicates whether the resulting position after a trade should be an opening position or closing position.Possible values are: “O” Open “C” Close (Netting)
PreviousPayDate	A 8-character Date field (format YYYYMMDD) indicating the previous pay date for bond.
Price	A 9-character numeric field that indicates the price of an order or trade. Format is 5.4. Possible values: 0 <= n <= 99,999.9999 In cases where negative pricing is applicable (combination trading) the sign (-) is incorporated in this field, reducing the numeric part to 4.4 format. The negative sign should occupy the first character of the integer part, but should be omitted in positive values. E.g a price of “-1.23” translates to “-00012300" while possible values are: -1,000.0000<= n <= 99,999.9999, excluding zero (reserved).
PriceLimits	A single character numeric field that categorizes an instrument with respect to how price limits are applied. ‘1’ - Low Velocity ‘2’ - Close price below tolerance ‘3’ - First Day ‘4’ - Resume trading

	For more information see ATHEX Rulebook
Product	A 2-character alphanumeric field indicating the type of product the security is associated with. Valid values: “3 “ : Corporate Bond “5 “ : Equity “6 “ : Government Bond “11”: Municipal “12“ : Other “13” : Financing
PutOrCall	A single character alpha field used only for options. Indicates whether an option is for a put or call. Possible values: “0”: Put “1”: Call “ “: security is not an option
TimeStamp	An 20-character alphanumeric time field in the format "YYYYMMDDhhmmssdddddd".
TradingCapacity	A single character alpha field used to indicate whether the order submission results from the member or participant of the trading venue is carrying out matched principal trading or dealing on its own account. Possible values: “0”: Deal on own account “1”: Matched principal “2”: Any other capacity
QuotationLevel	A single character alpha code that identifies the quotation level. Possible values: “M” Market “I” Instrument “U” - Underlying “F” Firm
QuoteID	A 8 characters numeric field identifying the quote throughout the current trading day. For usage with QuoteEntryChange (MA) messages, QuoteID is filled only on quote change actions to refer to a particular quote id. On new

	quote entry, QuoteID is space-filled and returned by OASIS with the quote entry confirmation.
QuoteMsgID	A 16-character alphanumeric field indicating the member-specific quote identification.
QuoteNote	A 25-character alphanumeric field containing member-specific quote notes.
QuoteRequestStatus	A single character alphanumeric field indicating the status of the quote request. Possible values: “0” Executed - To be forwarded to Market Maker(s)“1” Not Executed – Ignored by the Supervision Sub-system
QuoteResponsibilityStatus	A single character alphanumeric field indicating the status of the quotation responsibilty “S” – Suspended “R” – Resumed
QuoteRequestRejectReason	Refer to RejectReasonCode
QuoteReqID	A 16 character alphanumeric field indicating the member,s quote request number. The member is responsible for the accuracy and validity of this field.
QuoteSource	A single character alphanumeric type indicating the source of the quote. Possible values : “C” CTCI –ODL “M” ORAMA “R” EMRW
QuoteStatus	A single character alpha field indicating the status of a quote. Possible values : ‘A’ Accepted (active) ‘C' Cancelled
ReasonOfSuspension	A single character alpha field indicating the reason for quotation responsibility suspension. Possible values are: ‘1’ Manual suspension ‘2’ Suspension because of auction ‘3” Suspension because of limit up ‘4” Suspension because of limit down ‘5’ Minimum daily order volume has been covered

RedemptionValue	A 14-character numeric field (7.2) that indicates the redemption value of a bond.
RejectReasonCode	A 3-character numeric field used to indicate to a member firm the reason that a requested action could not take place. Possible values: “001” Incorrect Message Type “002” Incorrect Member ID “003” Member not active “004” Member suspended “005” Incorrect Trader ID “006” Trader not active “007” Trader suspended “008” Trader does not belong to this member “009” Trader unauthorized for this action “010” Incorrect Market ID “011” Market not active “012” Incorrect Board ID “013” Incorrect Security “014” Security not active “015” Security suspended “016” Security halted “017” Security paused “018” Security in Blackout period “019” Security does not participate in this phase “020” The trade in the security not started yet “021” The trade in the security ended “022” Can not do this action “023” Incorrect Price Type “024” Price Type not consists with the phase

	“025” Price below Floor “026” Price above Ceiling “027” Incorrect Tick size (spread) “028” Price not allowed for the Price Type “029” Order Volume too small (Lot size) “030” Order Volume too big (Reject threshold) “031” Order new volume less then the matched volume “032” Action not allowed by this type of user “033” Incorrect Side “034” Incorrect P/C flag “035” Incorrect Order Number “036” Order already matched “037” Order already canceled “038” Order suspended “039” Order not suspended “040” Order not approved “041” Order already approved “042” Order status is open “043” Order status is not open “044” Incorrect Condition “045” Incorrect Lifetime “046” Incorrect Trade Number “047” Trade already canceled “048” Trade not canceled “049” Incorrect Volume “050” Incorrect Price “051” Incorrect Published Volume “052” Disable Condition order

	“053”	Disable Life order
	“054”	Disable Price Type order
	“055”	Order already disapproved
	“056”	Incorrect Phase ID
	“057”	Incorrect Member Order Number
	“058”	Msg contains non-Ascii character[s]
	“059”	Trader did not login
	“060”	Login attempt from a used IP address
	“061”	Login attempt from an invalid IP address
	“062”	Invalid Link Number
	“063”	Invalid Index
	“064”	Invalid Index Sequence Number
	“065”	Gap between Order price and Last Sale too big
	“066”	Login attempt using your account
	“067”	Market is invalid for this user
	“68”	Odd lot orders do not trade in this phase
	“069”	Invalid trade status
	“070”	User has access to none of the markets
	“071”	Incorrect Customer Id
	“072”	Market is invalid for this member firm
	“073”	Incorrect stop order values
	“074”	Incorrect StopIndex order values
	“075”	Incorrect MF/MOF volume
	“076”	No order field was changed
	“077”	The board is currently disabled
	“078”	Invalid supervisor for trader
	“079”	The board does not trade at this time
	“080”	No supervisor exists to approve the order

	“081” Firm credit limit reject “082” Invalid order type “083” Order value outside allowed limits “084” Incorrect floor/ceiling adjustment percentage “085” Order belongs to a rpt-only trade “086” Order status is expired “087” The system is currently not active “088” Order status is neither open nor not-released “089” Incorrect clearing member “090” Incorrect trade value “091” Short transaction not allowed for security “092” Short sell below or equal to last price “093” Reserved “094” Reserved “095” Reserved “096” Incorrect Trade Report Type “097” Reserved “108” Reserved "109" Reserved "110" Reserved "111" Reserved "112" Reserved "113" Reserved "114" Reserved "115" Reserved "116" Reserved "117" Reserved “118” Technical problem

	"119" Unidentified Error "122" Reserved "124" Incorrect Venue ID "125" Incorrect Clearing Space ID "126" Incorrect Clearing Sub-Account ID “127” Incorrect value for Clearing Account ID "128" Incorrect action for Market Maker "129" Incorrect Quote Request "130" Quotation Responsibility Active "131" Incorrect value for Security ID Source "132" Incorrect value for GOI flag "133" Incorrect value for Settlement Type flag "134" Incorrect value for Position Effect flag "135" Incorrect value for Short Sell flag “136” Reject sponsored trading “137” Quotation responsibility upon request does not exist “138” Quotation responsibilities suspended “139” Valid quote already exists “140” Incorrect quote request (other reason) “141” Can not do this action. Order is Bait “142” At least one series in combo system is inactive “143” Can not do this action. Covered Sale Order “144” Invalid lifetime for Covered Sale Order “145” Invalid condition for Covered Sale Order “146” Incorrect value for ListID field “147” Incorrect value for Direct Electronic Access field “148” Incorrect value for Client Id Qualifier “149” Incorrect value for Investment Decision Id Qualifier

	“150” Incorrect value for Execution Within Firm Id Qualifier “151” Incorrect value for Trading Capacity field “152” Incorrect value for Liquidity Provision field “153” Incorrect value for Client Id “154” Incorrect value for Investment Decision Id field “155” Incorrect value for Execution Within Firm Id field “156” Incorrect value for Non Executing Broker Id field “157” Incorrect value for Special Divident flag “158” Incorrect value for Duplicative Trade Report flag
SDivFlag	A single character alphanumeric field that indicates whether this is a special dividend transaction. Possible values: ‘0’ False ‘1‘ True
SecurityCode	A 12-character alphanumeric field, indicating Bloomberg security’s identification
SecurityID	A 15-character alphanumeric field, indicating security’s identification. Value of SecurityID is dependent on field SecurityIDSource.
SecurityIDSource	A single character alphanumeric field identifying class or source of the SecurityID and StopSymbol if given. Possible values: “8” Exchange Symbol “A” Bloomberg Symbol
SecurityISIN	A 12-character alphanumeric field, indicating security’s ISIN code.
SecurityPrice	A 9-character numeric field that contain the security price whenever “Security Status” message is sent (see also Price). Possible values: At the beginning of the Pre-Opening phase - Start of day price At the beginning of the Opening phase - Zeroes At the beginning of the Continuous phase - Opening price At the beginning of the Closing phase - Zeroes At the beginning of the Closing price phase - Closing price

<table>
<tr><td></td><td>At the End of Trading - Closing price
At a resumption - Resumed Opening price
In all other cases - Last sale price</td></tr>
<tr><td>SecurityStatus</td><td>A single character flag that indicates security status.
Possible values are:
“A” Active
“N” Not active
“S” Suspended
“H” Halted
“R” Resumed (Resumed Pre-opening of a Halt)</td></tr>
<tr><td>SecuritySubType</td><td>A single character alphanumeric field indicating the type of a combination product:
“ “ – Not a combination product
“E” – Time Spread
“D” – Price Spread
“S” – Straddle
“B” - Butterfly</td></tr>
<tr><td>SecuritySymbol</td><td>A 15-character alphanumeric field, indicating security’s Exchange symbol</td></tr>
<tr><td>SecurityType</td><td>A left-aligned, space padded, 10-character alpha field, indicating security’s type:
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>CS</td><td>Common Stock</td></tr>
<tr><td>PS</td><td>Preferred Stock</td></tr>
<tr><td>MF</td><td>Mutual Fund (Exchangable Traded Fund)</td></tr>
<tr><td>EUSOV</td><td>Euro Sovereigns</td></tr>
<tr><td>TB</td><td>Treasury Bill - non US</td></tr>
<tr><td>TINT</td><td>Interest Strip From Any Bond Or Note</td></tr>
<tr><td>TIPS</td><td>Treasury Inflation Protected Securities</td></tr>
<tr><td>TCAL</td><td>Principal Strip Of A Callable Bond Or Note</td></tr>
<tr><td>TPRN</td><td>Principal Strip From A Non-Callable Bond Or Note</td></tr>
<tr><td>CORP</td><td>Corporate Bond</td></tr>
<tr><td>CPP</td><td>Corporate Private Placement</td></tr>
<tr><td>CB</td><td>Convertible Bond</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td><table>
<tr><td>DUAL</td><td>Dual Currency</td></tr>
<tr><td>EUCORP</td><td>Euro Corporate Bond</td></tr>
<tr><td>EUFRN</td><td>Euro Corporate Floating Rate Notes</td></tr>
<tr><td>XLINKD</td><td>Indexed Linked</td></tr>
<tr><td>STRUCT</td><td>Structured Notes</td></tr>
<tr><td>WAR</td><td>Warrant (The associated Product is 12 (Other))</td></tr>
<tr><td>REPO</td><td>Repo</td></tr>
<tr><td>FUT</td><td>Future</td></tr>
<tr><td>OPT</td><td>Option</td></tr>
<tr><td>OOF</td><td>Options on Futures</td></tr>
<tr><td>MLEG</td><td>Multi-leg instrument</td></tr>
<tr><td>NONE</td><td>No Security Type</td></tr>
</table></td></tr>
<tr><td>SellClearingMemberID</td><td>A 4-character alphanumeric field indicating the clearing sub- account ID of the seller side for a trade report.</td></tr>
<tr><td>SellClientID</td><td>A 10 character numeric field indicating the short code used to identify the sell-side client of the member or participant of the trading venue. Valid values range between 0 (denoting no short code entry) and 4294967295.</td></tr>
<tr><td>SellClientIDQualifier</td><td>A single character alphanumeric field to further qualify the SellClientID field.
Possible values :
‘A’ Algorithm
‘L’ Firm or legal entity
‘N’ Natural person
“ “ Not applied when field ClientID = 0 (NONE) or 1 (INTC) or 2 (PNAL)</td></tr>
<tr><td>SellClientOrderID</td><td>A 16 character alphanumeric field indicating the seller members order number in a trade report. The member is responsible for the accuracy and validity of this field.</td></tr>
<tr><td>SellExecutionWithinFirmID</td><td>A 10 character alphanumeric field indicating the short code used to identify the sell-side person or algorithm within the member or participant of the trading venue who is responsible for the execution of the transaction resulting from the order. Valid values range between 0 (denoting no short code entry) and 4294967295</td></tr>
<tr><td>SellExecutionWithinFirmIDQualifier</td><td>A single character alphanumeric field to further qualify the SellExecutionWithinFirmID field.
Possible values :
‘A’ Algorithm</td></tr>
</table>

	‘L’ Firm or legal entity ‘N’ Natural person “ “ Not applied when field ExecutionWithinFirmID = 0 (NONE) or 3 (NORE)
SellGOIFlag	This field is not used anymore. The Group Of Inverstor functionality is removed.
SellInvestmentDecisionID	A 10 character alphanumeric field indicating the short code used to identify the sell-side person or the algorithm within the member or participant of the trading venue who is responsible for the investment decision. Valid values range between 0 (denoting no short code entry) and 4294967295
SellInvestmentDecisionIDQualifier	A single character alphanumeric field to further qualify the SellInvestmentDecisionID field. Possible values : ‘A’ Algorithm ‘L’ Firm or legal entity ‘N’ Natural person “ “ Not applied when field InvestmentDecisionID = 0 (NONE)
SellMember Id	A 4-character alphanumeric field indicating the member who is the seller in a trade report.
SellNonExecutingBrokerID	A 10 character numeric field indicating the short code used to identify the sell-side person or participant of the trading venue who routed the order on behalf of and in the name of another member or participant of the trading venue. Valid values range between 0 (denoting no short code entry) and 4294967295.
SellTrader Id	A 5-character alphanumeric field that identifies the specific user from the seller side in a trade report. This field identifies the individual trader authorized to use the system, as registered with OASIS.
SellPrice	A 9-character numeric field that indicates the price of the sell side of a quote. Format is 5.4. Possible values: 0 <= n <= 99,999.9999 In cases where negative pricing is applicable (combination trading) the sign (-) is incorporated in this field, reducing the numeric part to 4.4 format. The negative sign should occupy the first character of the integer part, but should be omitted in positive values. E.g a price of “-1.23” translates to “-00012300" while possible values are:

	-1,000.0000 <= n <= 99,999.9999, excluding zero (reserved).
SellRemainingVolume	A 12-character field indicating the sell side volume of a quote available in the market.
SellVolume	A 12-character field indicating the disclosed volume of the sell side of a quote are available in the market.
SettlType	A single character alpha code that indicates the order settlement period. Possible values are: • "1": Immediate Settlement • "0": Normal settlement period
ShortSellFlag	A single character alpha field, indicating if the order is a short sell / buy to cover. Possible values: "N" Normal "Y" Short Sell / Buy To Cover
Side	A single character alpha field that specifies the buying or selling side of a transaction. Possible values: " " Member firm is both buyer and seller "B" Buy "S" Sell
Source	A 1-character alphanumeric field to identify the source of an order or trade. Possible values : "C" CTCI "M" ORAMA - ETW (Market Works station) "R" EMRW (Control regulation workstation) " " OASIS
SpecialConditions	A 1-character alpha field that indicates additional characteristics of an order. Possible values: "N" No Condition "I" Immediate Or Cancel (Apply only Main & Odd-lot board)

	“F” Fill or Kill (Apply only Main & Odd-lot board) “S” Stop on Security (Apply only Main board) “D” Stop on Index (Apply only Main board) “A” All Or None (Apply only Special board) “M” Minimum Fill (Apply only Special board) “O” Multiple of (Apply only Special board)
StartOfDayPrice	A 9-character numeric field that indicates the start of day price of a security. Format is 5.4.
Status (CB)	A single character alphanumeric field indicating the system status. Possible values : “H” Halt ”O” Resume Open
Status (CC)	A single character alphanumeric field indicating the market/board status. Possible values : For the main board: ”P” Pre-Call “J” Calculated projected opening price “T” Continuous/Auction event “C” Closing price trading “R” Run-off “E” End of trading “H” Halt “S” Stop (Used only in Auction Market) “N” No Orders accepted until the next Status change (Used only for XNET interface) For the other boards: “O” Open “E” End
StopSecurityID	Indicates security/index id used in condition of STOP orders. The value of this field is dependent on SecurityIDSource
StopPrice	A 9-character numeric field that indicates a stop price or index value in a Stop Security/Index order. When this value is reached the order is released. The format of this field will be 5.4.

StrikePrice	A 9-character numeric price-type field (see field ‘price’) that indicates the strike price of an option instrument. Format is 5.4. Not supplied in non option instruments
SurveilanceStatus	A single character numeric field indicating whether instrument appears in a designated quote page in MRW/ORAMA clients: Possible values: 1 – not marked 2 – marked (appeas in designated page)
Symbol	A 12-character alphanumeric field that indicates the name of a security.
Time	An 8-character alphanumeric time field in the format "HHMMSSTH" that indicates the execution time of a trade. Hours are based on a 24 hour clock.
TradeEditType	A single character alpha field identifing the type of trade report status change apllied from TradeReportEntry. Possible values: “N” New “A” Approve Trade Report “D” Disapprove Trade Report “C” Request for Cancellation of Trade Report
TradeNumber	A 6-character numeric field assigned by OASIS which is used to uniquely identify every trade within a trading day. Possible Values: 1 <= n <=999999
TradeReportExecuteTimestamp	An 20-character field type Time (format YYYYMMDDhhmmssdddddd) indicating the date and time that the trade report has been entered in the system.
TradeReportMethod	A 2-character alphanumeric field to identify the type of trade report. Possible values: “RA” Trade report Method 6-1 “RD” Trade report Spot-1 “RU” Trade report Spot-2 “RR” Trade report Settlement - Incomplete Buy “RS” Trade report Settlement - Incomplete Sell “RB” Trade report Method 6-1 Same Day Settlement

	“RM” Trade report only Method 6-1 Special Fees “RN” Trade report Method 6-1 Special Fees & Same Day Settlement “RL” Trade Report Lending – Borrowing “RC” Trade report Method 6-1, No CCP “RG” Trade report Method 6-1, No CCP & Same Day Settlement “RT” Trade Report Lending – Borrowing, Market Making “RF” Trade Report Lending – Borrowing, Failed Trade “RE” Trade Report Method 7-1 - Derivatives market “RQ” Trade Report - Other
TradeReportNumber	Trade Report Number refers to the trade number of the trade report to be edited. Space-filled for trade report entry (TradeEditType == ‘N’).
TradeSource	A 1-character alphanumeric field to identify the source of an order or trade. Possible values : “C” CTCI “M” ETW (Market Works station) “R” EMRW (Control regulation workstation) “ “ OASIS
TradeStatus	A 2single character alphanumeric field that indicates the status of a Trade. “ ” Normal Completed Trade “L ” Alleged “A ” Accepted “D ” Declined “E ” Expired “X ” Cancelled trade “C ” Cancelled incomplete trade report “U ” Changed trade (XNet)
TradeType	A 2-character alphanumeric field to identify the type of trade. Possible values:

	“MB” Main Board Trade “MO” Main Board Opening Trade “MC” Main Board Closing Trade “MM” Main Board, Combination to Combination Trade “ST” Special Board Trade “OL” Odd-lot Board Trade “FS” Forced sale Board Trade “RA” Trade report Method 6-1 “RD” Trade report Spot-1 “RU” Trade report Spot-2 “RR” Trade report Settlement - Incomplete Buy “RS” Trade report Settlement - Incomplete Sell “RB” Trade report Method 6-1 Same Day Settlement “RM” Trade report Method 6-1 Special Fees “RN” Trade report Method 6-1 Special Fees & Same Day Settlement “RL” Trade Report Lending – Borrowing “RC” Trade report method 6-1, No CCP “RG” Trade report method 6-1, No CCP & Same Day Settlement “RT” Trade Report Lending – Borrowing, Market Making “RF” Trade Report Lending – Borrowing, Failed Trade “RE” Trade Report Method 7-1 Derivatives market “RQ” Trade Report - Other
TraderID	A 5-character alphanumeric field that identifies a specific user. This field identifies the individual trader authorized to use the system, as registered with OASIS.
UnderlyingProduct	A 2 character alpha field indicating the product type of a derivative’s underlying. Valid values: “2 “: Commodity “3 “ : Corporate Bond

	“4 “: Curency “5 “ : Equity “6 “ : Government Bond “7 “: Index “11”: Municipal “12“ : Other “13” : Financing
UnderlyingSecuritySymbol	A 15 character alphanumeric field, indicating underlying security’s Exchange symbol
UnderlyingSecurityCode	A 12 character alphanumeric field, indicating underlying security’s code.
UnsuspendDisclosedVolume	Activation published volume. The volume that is announced to the market, when an inactive order is activated.
VenueID	A 4-character alphanumeric field indicating host trading venue (market place). The valid venue ids are specified in the International Standard ISO 10383.
VolatilityInterrupter	A single character alphanumeric field indicating whether given security is subject to volatility interruption. ‘Y’ – Volatility interruption is enabled ‘N’ – Volatility interruption is not enabled
Volume	A 12-character numeric field that indicates the number of shares for equity securities. Possible values: 1 <= n <= 999,999,999,999
WaiverIndicator	A 4- character numeric field indicating as to whether the transaction was executed under a pre-trade waiver. This will be a bitmap populated with one or more of the following flags: 1nd bit: RFPT – Reference price 2rd bit: NLIQ – Negotiated (liquid) 3th bit: OILQ - Negotiated (illiquid) 4th bit: PRIC – Negotiated (conditions) 5th bit: SIZE – Above specified size 6th bit: ILQD – Illiquid instrument

8.2. DSS interface

This chapter describes the fields of the objects when the corresponding messages are directed to or received from the DSS interface only.

Field	Description
ClientOrderID	A 16 character alphanumeric field indicating the member’s request number. The member is responsible for the accuracy and validity of this field.
CRC	A 4 character alphanumeric field storing CRC Code for DSSMessage field
DSSMessage	The actual DSS message. Its length is given by the Length field below.
Length	A 5 digits numeric value used to specify the size of the DSSMesage field
MemberID	Not currently used by DSS inteface. For future use
MemberSequenceNumber	A 6 character numeric field indicating the unique sequence number set to the message from the member.
NewOrderNumber	A unique 12 digits number set by DSS used to verify and identify each request. Possible values :: 1 <= n <=999.999.999.999
OrderNumber	The unique 12 digits number that DSS assigned to the initial request and reported back in the NewOrderNumber field of the corresponding TS message.
TraderID	Not currently used by DSS inteface. For future use
Type	A single character alphanumeric field intended for future use. It’s default value is ‘A’.
VenueID	Not currently used by DSS inteface. For future use

9. Appendix C. Error Codes

```
//
// MessageId: ETS_W_NOTIMPLEMENTED
//
// MessageText:
//
//  Not implemented yet
//
#define ETS_W_NOTIMPLEMENTED  0x00042500

//
// MessageId: ETS_W_EXTRAINFO
//
// MessageText:
//
//  Extra info is required
//
#define ETS_W_EXTRAINFO  0x00042501

//
// MessageId: ETS_W_STATE
//
// MessageText:
//
//  Illegal State
//
#define ETS_W_STATE  0x00042502

//
// MessageId: ERGO_E_FAIL
//
// MessageText:
//
//  Function call failed
//
#define ETS_E_FAIL       0x80042503

//
// MessageId: ETS_E_EXCEPTION
//
// MessageText:
//
//  Exception while processing
//
#define ETS_E_EXCEPTION  0x80042504

//
// MessageId: ETS_E_NOTFOUND
//
// MessageText:
//
//  Data not found
//
```

```
#define ETS_E_NOTFOUND   0x80042505

//
// MessageId: ETS_E_USERUNKNOWN
//
// MessageText:
//
//  User is unknown (not logged in)
//
#define ETS_E_USERUNKNOWN   0x80042506

//
// MessageId: ETS_E_MSGUNKNOWN
//
// MessageText:
//
//  Unknown Message type
//
#define ETS_E_MSGUNKNOWN  0x80042507

//
// MessageId: ETS_E_STATE
//
// MessageText:
//
//  Illegal State
//
#define ETS_E_STATE  0x80042508

//
// MessageId: ETS_E_LOGONREJECTED
//
// MessageText:
//
//  The logon rejected by the Comm Server
//
#define ETS_E_LOGONREJECTED  0x80042509

//
// MessageId: ETS_E_NOCONNECTION
//
// MessageText:
//
//  There is a problem with the connection or no
//  resources in the machine
//
#define ETS_E_NOCONNECTION  0x8004250a

//
// MessageId: ETS_E_MAXRETRIES
//
// MessageText:
//
//  Maximum number of retries (connection)
//
```

```
#define ETS_E_MAXRETRIES  0x8004250b

//
// MessageId: ETS_E_TIMEOUT
//
// MessageText:
//
//  Maximum number of retries (connection)
//
#define ETS_E_TIMEOUT  0x8004250c

//
// MessageId: ETS_E_INVALIDARG
//
// MessageText:
//
//  Invalid argument value
//
#define ETS_E_INVALIDARG  0x8004250d

//
// MessageId: ETS_E_FATAL
//
// MessageText:
//
//  Fatal error. Unrecoverable
//
#define ETS_E_FATAL  0x8004250e

//
// MessageId: ETS_E_WAITFAILED
//
// MessageText:
//
//  A Wait... function failed. Trying to restart
//  the thread.
//
#define ETS_E_WAITFAILED  0x8004250f

//
// MessageId: ETS_E_OUTOFSEQUENCE
//
// MessageText:
//
//  An out of sequence message received
//
#define ETS_E_OUTOFSEQUENCE  0x80042510

// MessageId: ETS_E_NOTACTIVE
//
// MessageText:
//
//  User or administrator's connections is not active
//
#define ETS_E_NOTACTIVE  0x80042511
```

```
// MessageId: ETS_E_ERR_INTRF_POINTER
//
// MessageText:
//
//  A non-valid interface pointer
//
#define ETS_E_ERR_INTRF_POINTER  0x80042512

// MessageId: ETS_E_NOTINGROUP
//
// MessageText:
//
//  User is not in the authorized group
//
0x80042503 ETS_E_NOTINGROUP 0x80042513

// MessageId: ETS_E_CALLBACK_ERROR
//
// MessageText:
//
//  Problems with notification interface
//
#define ETS_E_CALLBACK_ERROR 0x80042514

// MessageId: ETS_E_PENDING_MESSAGES
//
// MessageText:
//
//  There are pending messages.
//
#define ETS_E_PENDING_MESSAGES 0x80042515

//
// MessageId: ETS_E_NOLCONNECTION
//
// MessageText:
//
//  There is a problem with the connection or no
//  resources in the machine
//
#define ETS_E_NOLCONNECTION  0x80042516

//
// MessageId: ETS_E_SECURITY
//
// MessageText:
//
//  There is a problem with the connection or no
//  resources in the machine
//
#define ETS_E_SECURITY  0x80042517

//
// MessageId: ETS_E_MAX_REJECTED_MSGS
```

```
//
// MessageText:
//
//  Maximum Number requests has been reached
//  from the ODL GW
//
#define ETS_E_MAX_REJECTED_MSGS  0x80042518
//
// MessageId: ETS_E_RANGERR
//
// MessageText:
//
//  Range Error from COMM Server
//
#define ETS_E_RANGERR  0x80042519
//
// MessageId: ETS_E_MAX_RATE_REACHED
//
// MessageText:
//
//  Maximum message transfer rate has been reached
//  from the ODL GW
//
#define ETS_E_MAX_RATE_REACHED  0x8004251B
```

10. Appendix D. Error codes received from the member application through the NewError function.

Error Code	Details
ETS_E_INVALIDARG	("CMCTService::LogGetMessages(Invalid date argument)")
<GENERAL_ERROR>	("CMCTService::LogGetMessages: Cannot Open Log File: 0x%x"),hr
ETS_E_STATE	("CMCTService::LogGetMessages: Illegal Message ID.")
ETS_E_INVALIDARG	("CMCTService::LogSendMessages(Invalid date argument")
<GENERAL_ERROR>	("CMCTService::LogSendMessages: Cannot Open Log File: 0x%x"),hr
ETS_E_STATE	("CMCTService::LogSendMessages: Illegal Message ID.")
ETS_E_INVALIDARG	("CMCTService::CheckMessageDate(Invalid date argument")
<GENERAL_ERROR>	("CMCTService::CheckMessageDate: Cannot Open Log File: 0x%x"),hr
<GENERAL_ERROR>	("CMCTService::Initialize: Registry Open Error <0x%x>"),hr
ETS_E_USERUNKNOWN	("CMCTService::SendCTCIMessage(Unknown User: %ld, <%s>)"),dwUserId,static_cast<LPCTSTR>(bsMsg)
ETS_E_MSGUNKNOWN	("CMCTService::SendCTCIMessage(Unknown Message Type: <%s>)"),static_cast<LPCTSTR>(bsMsg)
<GENERAL_ERROR>	("CMCTService::Disconnect(Error trying to send Logout CCM Message)")
<GENERAL_ERROR>	("CMCTService::get_property: Registry Query Value <0x%x>"),HRESULT_FROM_WIN32(ldRet)
<GENERAL_ERROR>	("CMCTService::LogMessage(Cannot open Log File: %s, Error: 0x%x)"), mcsLogFile, hr
<GENERAL_ERROR>	("CMCTService::ReadPreviousState: Cannot create Log File: 0x%x"),hr
<GENERAL_ERROR>	("CMCTService::ReadPreviousState: Cannot write to Log File: 0x%x"),hr

ETS_E_FAIL	("THE SYSTEM IS UNDER ATTACK - Hacker: [%s/%s]"),user.ObtainDomainName().c_str(),user.ObtainUserName().c_str()
ETS_E_FAIL	("CMCTService::AllUsersCTCINotification(Cannot notify clients about an CTCI message: %s)"),errno == EAGAIN ? ("Too many threads") : errno == EINVAL ? ("Invalid parameter") : ("Unknown error")
ETS_E_FAIL	("CMCTService::AllUsersCTCINotification(Timeout occur in an notification thread (CTCI) for machine: %s)"),m_users[dwUserId].m_wstrMachine.c_str()
ETS_E_FAIL	("CMCTService::AllUsersETSNotification(Timeout occur in an notification thread (ETS) for machine: %s)"),m_users[dwUserId].m_wstrMachine.c_str()
ETS_E_STATE	("CMCTService::ProcessCTRL_LASTRECVD(There are lost messages in CTCI. Wrong SeqNum for sending messages: ETS: %d, CTCI: %d)"),m_dwOutSeqNo-1,dwLastReceived
ETS_E_STATE	("CMCTService::ProcessCTRL_LASTRECVD(Wrong SeqNum for sending messages: ETS: %d, CTCI: %d) Trying to be synchronized with COMM -- Lost messages!!!!"),m_dwOutSeqNo-1,dwLastReceived
ETS_E_STATE	("CMCTService::ProcessCTRL_RETRANSMIT(There are lost messages in CTCI. From: %d, To: %d)"),dwRtnsmtFrom,dwRtnsmtTo
ETS_E_FAIL	("The maximum number of reject messages reached. You have to restart the service.")
ETS_E_NOTFOUND	("CMCTService::ProcessDATA_NewTrade(Order message dont found for trade message <%s>)"),ptsMsgData
ETS_E_NOTFOUND	("CMCTService::ProcessDATA_NewTrade(Trade message dont found for trade status message <%s>)"),ptsMsgData
ETS_E_MAX_REJECTED_MSGS	("CMCTService::SendCTCIMessage(Illegal state. Max Reject messages) ")
ETS_E_STATE	(" CMCTService::ProcessCTRL_RXMCTRANGERR (%s)")
ETS_E_RANGERR	("CMCSTService::ProcessCTRL_RXMCTRANGERR (Service is going to stop! Manual recovery through <InitSeqNumberValue> field) ")

11. Appendix E. ODL-Client-API: Visual basic programming example of the ETS_Broker.Administrator object

```
Dim WithEvents adm As Administrator

Private Sub adm_NewCCMMessage(ByVal bsConn As String, ByVal bsCCM
As String, ByVal fSend As Boolean)

’ TODO: Handle the new Control message
’ from OASIS system.
End Sub

Private Sub adm_NewDataMessage(ByVal bsConn As String, ByVal bsDATA
As String, ByVal swMsgId As Long, ByVal fSend As Boolean)
’ TODO: Handle the new Data message
’ from OASIS system.
End Sub

Private Sub adm_NewError(ByVal bsError As String, ByVal hError As Long)
’ TODO: Handle the error.
End Sub
Private Sub adm_NewUser(ByVal bsMachine As String, ByVal fReg As
Boolean)
’ TODO: Handle the new user
’ registration/unregistration.End Sub

‘To activate the connection to the Comm Server
 On Error GoTo err_lbl
  Dim reply As String

 reply = ""
 adm.Connect “ETS”, True, 10000, False, reply

 Exit Sub
err_lbl:
  MsgBox Err.Description + " REPLY=" + reply

‘To disconnect from Comm Server
 On Error GoTo err_lbl

 adm.Disconnect “ETS”, True, False

 Exit Sub
err_lbl:
  MsgBox Err.Description

‘ To retrieve old messages (of a previous day trading session)
 On Error GoTo err_lbl
 Dim omsgs As Variant
 Dim ocons As Variant
 Dim omids As Variant
 Dim omdts As Variant
 Dim omusr As Variant

   adm.GetOldMessages txtDate.text, 0, omids, ocons, omsgs, omdts, omusr

 Dim con As String
 Dim msg As String
 Dim mid As Long
 Dim dat As Date
 Dim uid As Integer

 For i = LBound(omids, 1) To UBound(omids, 1)
   msg = omsgs(i)
   con = ocons(i)
   mid = omids(i)
   dat = omdts(i)
   uid = omusr(i)

   ...............
 Next i

 Exit Sub
err_lbl:
   MsgBox Err.Description
```

12. Appendix F, ATHEX Gateway Failover Mechanism

12.1. Primary – Backup setup

In case a primary – backup arrangement is used in ATHEX Gateway then, initially, in normal conditions the gateways operate as it is described in the following Figure 12-1:

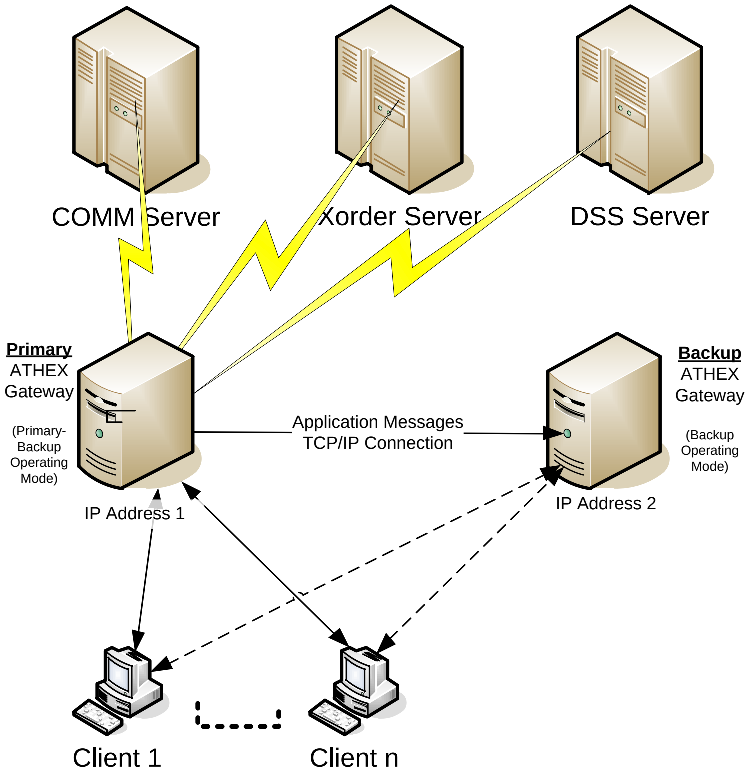


Figure 12-1: Primary – backup ATHEX Gateway operation

If a problem occurs and the Primary ATHEX Gateway is not functional anymore or the network connection between Primary ATHEX Gateway and the ATHEX Server(s) is broken then the failover mechanism can be activated. The Member staff must:

1. If the network connection between Primary ATHEX Gateway and the ATHEX Server(s) is fine but the Primary ATHEX Gateway machine is not running anymore, proceed to step 3.

2. If the network connection between Primary ATHEX Gateway and the ATHEX Server(s) is broken, terminate the ATHEX Gateway running in the Primary ATHEX Gateway.

3. On the Backup ATHEX Gateway:

    a. Start a Command prompt window as an administrator.

    b. Type c:\ATHEX\RESTART_ATHEX_GW_AS_PRIMARY.bat

c. The “Do you want to restart ASE Service (FIX Protocol)?” question will be displayed on the screen. Press ‘y’ if there is at least one FIX Client.

After all the above steps are followed, the second ATHEX Gateway will immediately operate in primary – standalone mode (without backup functionality) as it is described in Figure 12-2 bellow:

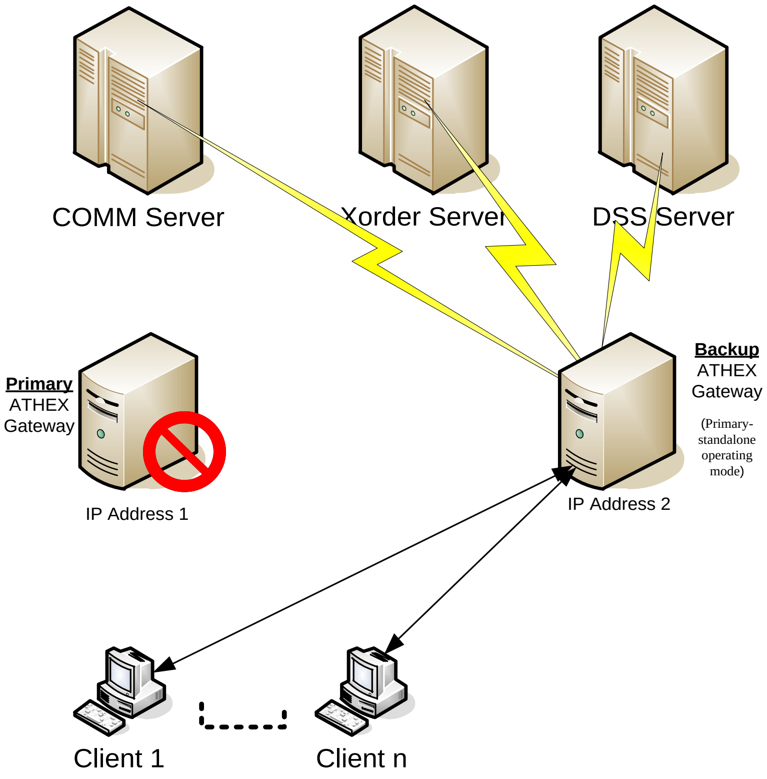


Figure 12-2: Primary – standalone ATHEX Gateway operation (after failover)

If the problem is fixed and the primary ATHEX Gateway can become operational again, then the primary – backup arrangement shall be used for **the next trading day** and **after the end of the current trading day**. In order for this configuration to become active again **from the beginning of the next trading day**, BackupServerMode value must be changed into 2 again, in Backup ATHEX Gateway.

12.2. Multiple Active gateway setup

In this section an alternative to the primary – backup setup discussed above is presented. The new feature addresses the complexity involved in switching from primary to backup. Other apparent advantages include the increased order throughput and load balancing capabilities.

The new approach ensures availability of all gateways, at any point in time, to any client application. Essentially, this is achieved by exposing all available gateways to the equal amount of information, enforcing equilibrium in that respect, much like in a primary-backup scheme. The catalytic difference between the two practices is that in a multiple gateway mode, any, or as a matter of fact, *all* of the available gateways can be actively connected to the Comms server concurrently. It is in the member's discretion to apply any policy of utilization, hence the flexibility of the overall design. The following example outlines the virtues of a multiple gateway setup.

Consider a twin gateway deployment where both ETS Services are connected at any time:

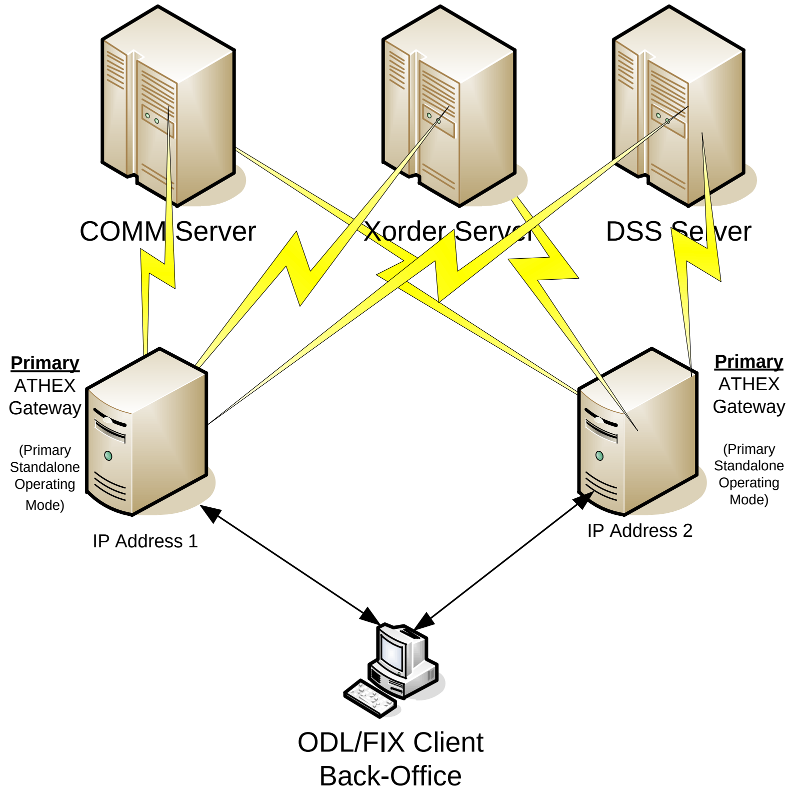


Figure 12-3: Double concurrent ATHEX Gateway operation

- The member may connect a unified "back-office" application to both gateways (thus increasing the firm's sending capabilities by effectively doubling protection valve thresholds).

- The ODL/FIX client may be designed to implement a load balancing mechanism, or "prioritize" orders by assigning custom sending criteria to a particular gateway.

- Actively use only Gateway 1 and maintain Gateway 2 as a failsafe in case of failure.

In all cases and in the event of failure of one of the Gateways, the member can expect to have lost nothing in terms of incoming messages, since the second gateway will have its own 'copies'. Thus, the client application is capable of effortlessly resuming normal operation through the second gateway. Even if the client was not registered with Gateway 2 at the time of failure of Gateway 1, all that is required is to register to the second Gateway, ask for all messages (e.g. GetLostMessages()) and establish a functional route to the exchange, while the first Gateway is being reinstated. The only requisite is that the valid clients as identified in the registry of each Gateway server should be identical (please refer to the ATHEX Gateway installation manual).

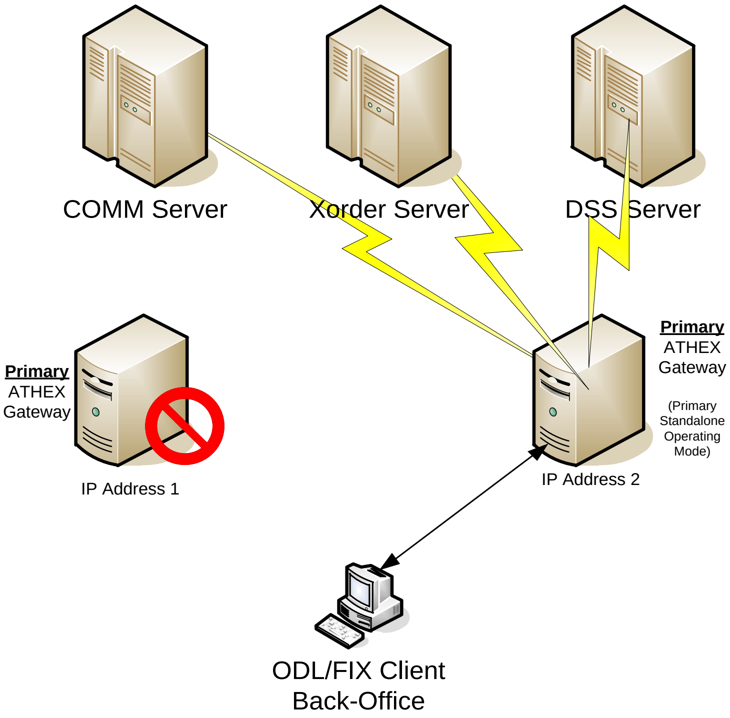


Figure 12-4: Double concurrent ATHEX Gateway operation (after failover)

It has to be noted that this configuration does not prohibit the facilitation of an extra backup gateway, as defined in a primary-backup scenario.

12.3. FIX session recovery

All messages from and to FIX clients are flushed to a log file by the ATHEX Gateway. This log file is crucial to the resynchronization of FIX clients and ATHEX Gateway.

Recovering a FIX session, involving a clean state from a G/W perspective, relates to cases of:

- G/W ASE data logfile deletion.
  - ***IMPORTANT****: This action resets G/W FIX Sequence counters*
- G/W failover (to a concurrently connected or backup G/W)

In the event of FIX session recovery for any of the above reasons, the member should adopt one of the following options prior to reestablishing a FIX session:

1. Resetting FIX sequence number counters on the client side (12.3.1)

   Or

2. Resetting FIX sequence number counters on the G/W side by either:
   - The automated procedure described below (12.3.2.1)
   - The manual procedure described below (12.3.2.1)

This session recovery mechanism ensures that no orders will re-enter the trading engine (note that there is currently no special handling of possible duplicate messages by the trading system). Furthermore, it limits the exchange of large amounts of messages between counter-parties.

12.3.1. *Resetting FIX sequence number counters on the client side*

Note that by resetting the FIX sequence counters on the client, all FIX application messages will be resent to the client upon reconnection.

In the case where the FIX client cannot connect to the GW due to wrong sequence numbers, the dumplog utility can be used on the GW to extract the current incoming and outgoing message sequence numbers of each scheduler:

1. Run a command prompt as an administrator
2. type: “CD C:\ATHEX\ODLSERVER”
3. type: “dumplog –a”. This command will generate a file for each scheduler. The filename has the format *FIX.4.4-exae-**SCHEDULER_NAME**.seqnums*. Each file consists of a single line of the form “*INCOMING_SEQ_NUM* : *OUTGOING_SEQ_NUM*“
4. Try to log on using MsgSeqNum = *INCOMING_SEQ_NUM* + 1
5. The scheduler will reply with a LogOn with MsgSeqNum = *OUTGOING_SEQ_NUM* + 1.

12.3.2. *Resetting FIX sequence number counters on the G/W side*

This option enables the client to request retransmission of application messages from the last received onwards, hence excluding messages previously received.

12.3.2.1. Automated FIX session recovery

The automated procedure involves adjustment of the client application to properly handle the recovery case by inclusion of custom FIX tags in the Logon message.

For the activation of this mechanism the FIX client must include tag **‘6000’** (LastMsgSeqNumSend) in the Logon message. This denotes a recovery logon and notifies the ATHEX Gateway on the previously terminated session’s status (in terms of sequence counters).

The optional tag ‘**6001’** (LastODLMsgID) allows the client to specify the last application message successfully received. This enables the G/W to only transmit messages not previously sent to the client.

Provision of tag 6000 in the Logon message is mandatory and will:

i. Reset the Outgoing (G/W to client) SeqNum counter to the value included in tag 6000.

ii. Reset the Incoming (client to G/W) SeqNum counter to the value included in tag 34 of the Logon message.

Provision of tag 6001 in the Logon message is optional and will:

i. Provide information as to the last application message received by the client. This is represented by the value of tag ‘198’ (SecondaryOrderID) of the last application message received. Only reports with greater values than this one will be sent to client upon reconnection.

Tag	Value	Required
6000	Should be set to <the FIX client’s expected SeqNum to be received from the G/W>	Yes
6001	Should be set to <the last received value of tag 198>, available in application messages	No

12.3.2.2. **Manual FIX session recovery**

Tha manual procedure involves intervention to registry values and restarting ASE_Service.

All proposed registry fields are available under key:

HKEY_LOCAL_MACHINE\SOFTWARE\[Wow6432Node]\ATHEX\ASE_Service\Schedulers\[FIX Scheduler]\Trader.

For the activation of this mechanism the registry must include appropriate values for fields ‘InputOffset’ and ‘OutputOffset’, prior to the restart of the ASE (FIX) Service.

The optionally-filled ‘LastSecondaryOrderID’ field allows specification of the last application message successfully received by the client application. This enables the G/W to only transmit messages not previously sent to the client.

Field	Value	Required
InputOffset	Should be set to <the FIX client's next SeqNum to be sent to the G/W decremented by one>	Yes
OutputOffset	should be set to <the FIX client's expected SeqNum to be received from the G/W decremented by one>	Yes
LastSecondaryOrderID	Should be set to <the last received value of tag 198>, available in application messages	No

12.3.2.1. Concurrent G/W failover

In attempting FIX session recovery through usage of a concurrently connected G/W the client should note the following:

- The client should always connect to its designated FIX scheduler (port). Moreover, the G/W registry configuration must adhere to naming conventions described in the ATHEX G/W Installation Guide.
- FIX session recovery to a FIX scheduler where a session had previously been established (e.g. recovery from GW2 to GW1, after having recovered the session from GW1 to GW2), requires that the ASE data log be manual deleted (including a ASE service restart) prior to the recovery attempt.
- FIX session recovery to a FIX scheduler where no previous session had been established, requires no additional action.

13. Appendix G, OASIS Server parameters

13.1. Parameters

In order for the Members applications to be informed about the configuration of the System they should store some information about the System, the Markets, the Securities, the Venues and the Indexes.

The values of Parameters for the:

- Securities
- Indexes

that are frequently changed are provided by the OASIS-ORAMA application.

The current parameters of the OASIS Server System and of the Market the system implements, are the following at the relevant levels:

13.1.1. *System*

Field	Description	OASIS
General		
Base currency	The base currency used for trading.	EURO
Credit Limit		
Calculation Method	The selection of the credit limit calculation method	Ref: Regulatory Commision, Art. 1/392/26.7.2006
Validation %	After a member reaches this percentage of the allowed daily credit limit, no more orders can be entered unless the member's daily value limit is increased.	100%

13.1.2. *Markets*

Field	Description
General	
Market ID	The market's unique ID. Not null. Cannot be edited.
Main market type	The type of the main market for this market.

Field	Description
Symbol English	The English symbol for the market.
Symbol Hellenic	The Hellenic symbol for the market.
Name English	The English name for the market.
Name Hellenic	The Hellenic name for the market.
Valid Boards	The valid boards for the market.
Valid order types	The valid order type for the market.
Rules	
Market Change Warning Period	The time that a warning message is sent before a trading state finishes.
Minimum Order Value	The minimum value (price * volume) that orders are allowed to have in the market.
Maximum Order Value	The maximum value (price * volume) that orders are allowed to have in the market.
Closing Price	
Calculate closing price method	The way of valuation of the financial instruments' closing.

13.1.3. *Securities*

The following information is kept about each security.

Field	Description	Notes
General		
ISIN	The unique ISIN code of the security.	Duplicate Key is not allowed. Valid only for non-derivative products.
English symbol	The system wide symbol of the security in English.	This field must be unique amongst all securities.
English name	The security's name in English.	

Field	Description	Notes
Hellenic symbol	The security's symbol in Greek.	
Hellenic name	The security's name in Greek.	
Price	The Last Price of the Security	
Bloomberg symbol	The Bloomberg security's identification	Duplicate Key is not allowed.
Security Status	The status of security	Active / Not active / Suspended / Halted / Resumed
Expiration date	The expiration date of a warrant or right	
Lot size	The trading unit for the specific security (lowest denomination).	
Floor/Ceiling		
Floor	Minimum percentage of allowed price.	Ref: ATHEX Rulebook, section **2.6.3 &** Art.22
Ceiling	Maximum percentage of allowed price.	Ref: ATHEX Rulebook, section **2.6.3 &** Art.22

13.1.4. *Members*

The following information is kept about each Member.

Field	Description	Notes
ID	The Member's ID	Duplicate Key is not allowed.
Credit Limit	The credit limit value	
Status	The status of the member	Active / Inactive / Suspended
English Name	The English name of the member	
Hellenic Name	The Hellenic name of the member	

13.1.5. *Traders*

The following information is kept about each Trader.

Field	Description	Notes
ID	The User's ID	Duplicate Key is not allowed.
English Name	The user's English Name	
Hellenic Name	The user's Hellenic Name	
Member	The Member ID of the Member firm the user is associated with.	The Member ID
Status	The user status	Active / Inactive / Suspended

13.1.6. *Indexes*

The following information is kept about each Index.

Field	Description	Notes
English Symbol	The English Symbol for the index	
English Name	The English Name for the index	
Hellenic Symbol	The Hellenic Symbol for the index	
Hellenic Name	The Hellenic Name for the index	
ISIN	The ISIN code for the index	Duplicate ISIN code is not allowed.
Index Value	The Index value	

13.1.7. *Price Tick*

Field	Description	Notes
Lower Bound	The lower price boundary for which the price tick is eligible.	
Upper Bound	The upper price boundary for which	
Tick	The price tick.	

It should be noted that a different price tick may be set for different securities. For up-to-date information regarding price ticks please refer to the ATHEX rulebook.

13.1.8. *Halt Reason Codes*

Halt Reason ID	English Description
01	Halt
02	Ceiling
03	Floor
04	War
05	Politics
06	Technical
07	Terrorism
08	Various
09	Religion
10	Celebration
11	Earthquake
12	Volatility Interrupter
13	Series Expiration

13.1.9. *Holidays*

Field	Description	OASIS
Holiday Date	The date of the holiday	Duplicate Key is not allowed.
Description	The description of the holiday	

If an order is sent with Expiration Date on a weekend or holiday, the system sets as Expiration Date the last working date before the date initially set.

13.2. Trading model

The Trading Model in the OASIS Server (existing Markets, Trading Boards etc.) is presented in Rule 30 of ATHEX's BD as it stands.

13.3. Time schedule for system-markets

The time schedule for Systems and Markets can be found at

http://www.athex.gr/content/en/Ann.asp?AnnID=61367

13.4. Order Types

The types of orders allowed in the System Boards during the various Trading Phases appear in the following table.

			Main Boards		Special Terms Board	Forced Sales Board
Order Price	**Order Condition**	**Order Duration**	**CAM**	**CAM M**	**Hit & Take method**	**Hit & Take method**
Market	No Condition	Good for Day	X	X		
		Good Till Date	X	X		
		Good Till Cancel	X	X		
	Stop on Index	Good for Day	X	X		
		Good Till Date	X	X		
		Good Till Cancel	X	X		
	Stop on Security	Good for Day	X	X		
		Good Till Date	X	X		
		Good Till Cancel	X	X		
	Immediate or Cancel	Good for Day		X		
	Fill or Kill	Good for Day		X		
Limit	No Condition	Good for Day	X	X		X
		Good Till Date	X	X		
		Good Till Cancel	X	X		
	Stop on Index	Good for Day	X	X		
		Good Till Date	X	X		
		Good Till Cancel	X	X		
	Stop on Security	Good for Day	X	X		
		Good Till Date	X	X		
		Good Till Cancel	X	X		
	Immediate or Cancel	Good for Day		X		
	Fill or Kill	Good for Day		X		
	All or None	Good for Day			X	
	Minimum Fill	Good for Day			X	
	Multiples of	Good for Day			X	
ATO	No Condition	Good for Day	X			
ATC	No Condition	Good for Day	X	X		

Note: The trade reports are entered in the “Report only” board, they have no condition and their duration is daily.

For issues in relation to the information contained in this appendix and generally issues concerning the Stock Market operation, you may address **OASIS Operation Department (email:** Stocks Helpdesk@helex.gr)

14. Appendix H. Sequence Diagrams for Quotation Negotiation.

The following sequence diagram demonstrates Quotation Negotiation handling. This chapter intends to analyse the way Trading System processes Quotation Negotiation related messages and to make clear the correct use of these messages from the members and the market makers in order to complete the quote request and quote entry/change/cancel procedure.

14.1. Quote Entry/Change

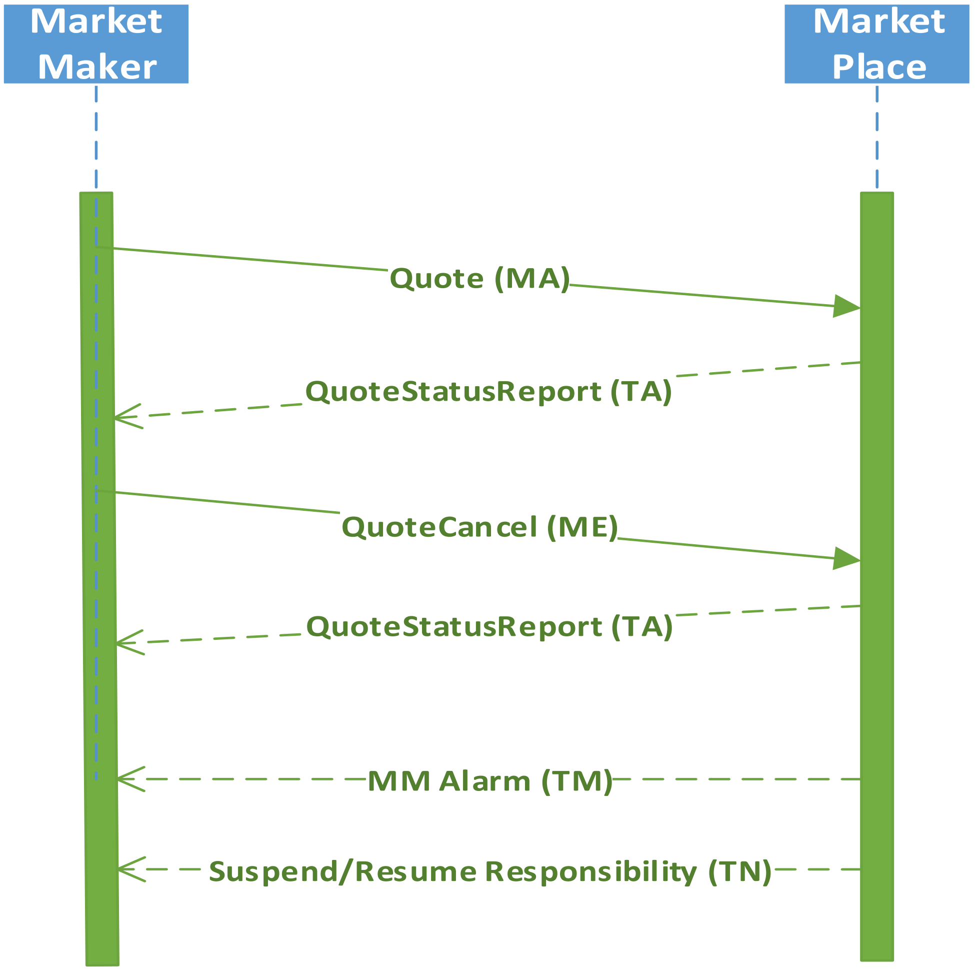


Figure 14-1 Quote Entry / Change

14.2. Quote Request

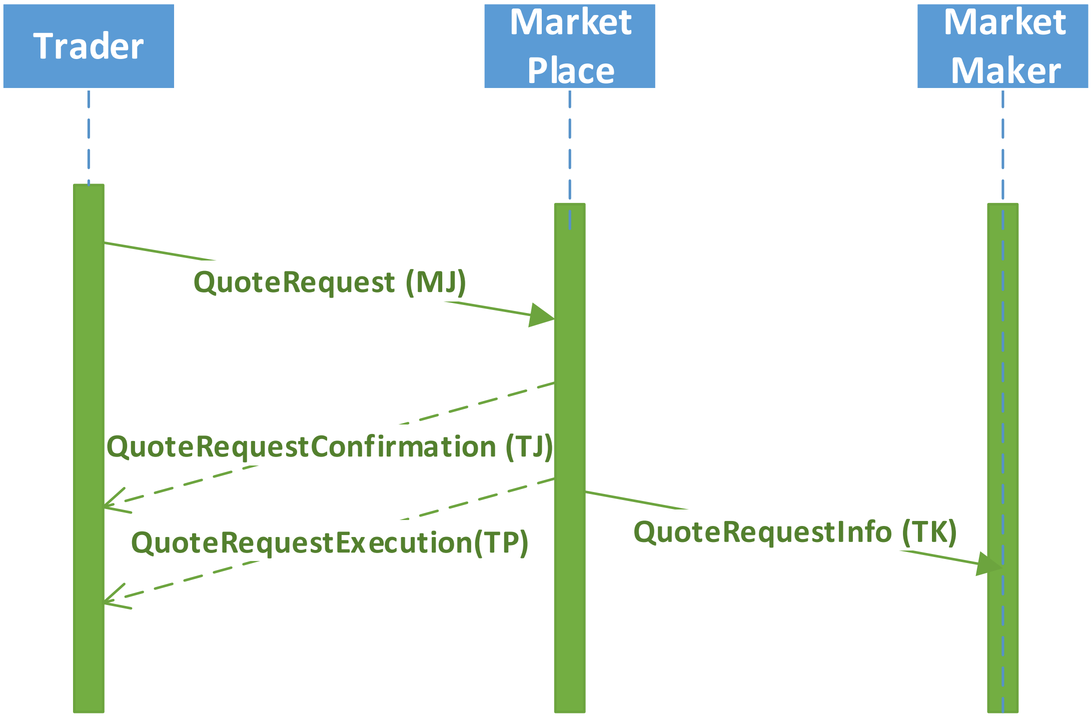


Figure 14-2 Quote Request

15. Appendix I. Sequence Diagrams for Trade Report procedure Examples.

The following sequence diagrams demonstrate Trade Report procedure examples. This chapter intends to analyse the way Trading System processes Trade Report related message and to make clear the correct use of this message from the counterparties in order to complete the Trade procedure.

15.1. Trade Report Approval

Seller inserts a TradeReportEntry and buyer approves.

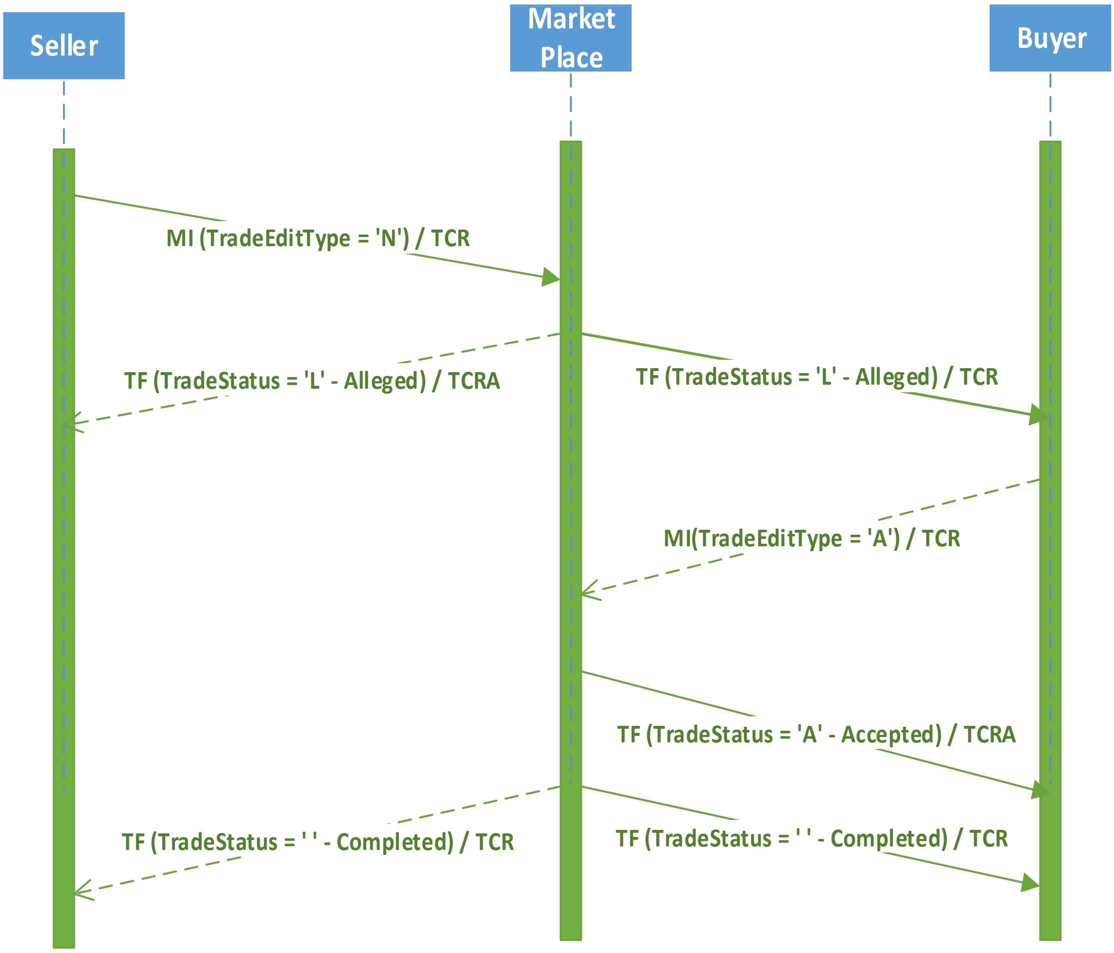

Seller
Market Place
Buyer
MI (TradeEditType = 'N') / TCR
TF (TradeStatus = 'L' - Alleged) / TCRA
TF (TradeStatus = 'L' - Alleged) / TCR
MI(TradeEditType = 'A') / TCR
TF (TradeStatus = 'A' - Accepted) / TCRA
TF (TradeStatus = ' ' - Completed) / TCR
TF (TradeStatus = ' ' - Completed) / TCR

Figure 15-1 Trade Report Approval

15.2. Trade Report Disapproval

Seller inserts a TradeReportEntry and buyer disapproves.

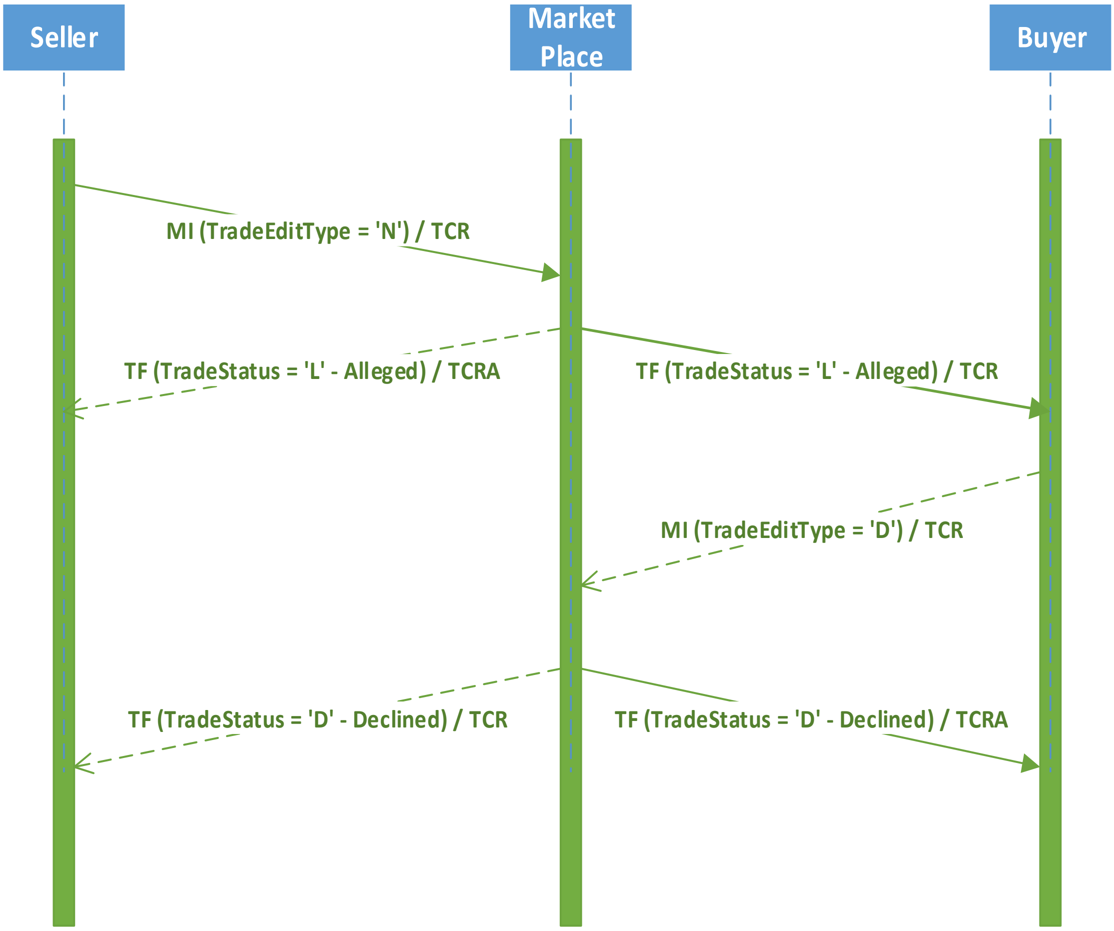

Seller
Market Place
Buyer
MI (TradeEditType = 'N') / TCR
TF (TradeStatus = 'L' - Alleged) / TCRA
TF (TradeStatus = 'L' - Alleged) / TCR
MI (TradeEditType = 'D') / TCR
TF (TradeStatus = 'D' - Declined) / TCR
TF (TradeStatus = 'D' - Declined) / TCRA

Figure 15-2 Trade Report Disapproval

15.3. Trade Report Cancellation

Seller inserts a new trade report (using a TradeReportEntry message) and cancels it (using another TradeReportEntry message) before buyer’s approval.

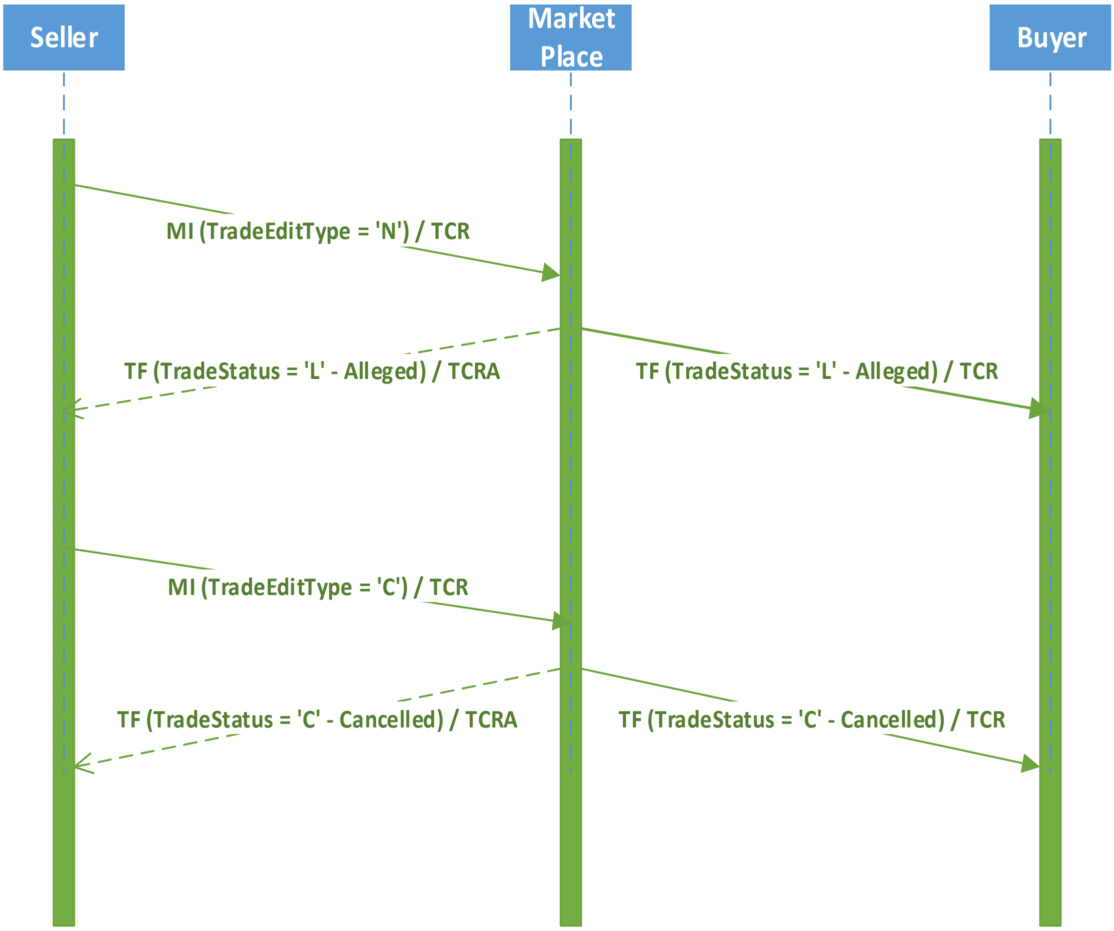

Seller
Market Place
Buyer
MI (TradeEditType = 'N') / TCR
TF (TradeStatus = 'L' - Alleged) / TCRA
TF (TradeStatus = 'L' - Alleged) / TCR
MI (TradeEditType = 'C') / TCR
TF (TradeStatus = 'C' - Cancelled) / TCRA
TF (TradeStatus = 'C' - Cancelled) / TCR

Figure 15-3 Trade Report Cancellation

15.4. Trade Report Expiration

Seller inserts a new trade report (using a TradeReportEntry message) and Approval Timeout expires in Trading System. Trade expires.

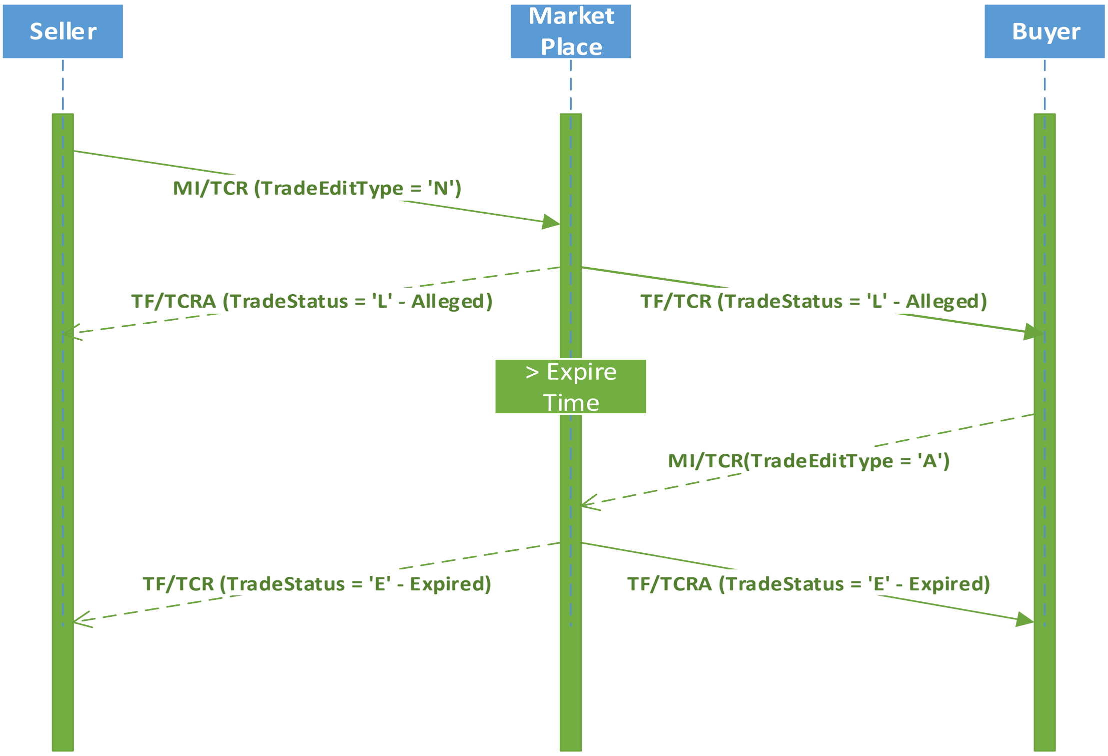

Seller
Market Place
Buyer
MI/TCR (TradeEditType = 'N')
TF/TCRA (TradeStatus = 'L' - Alleged)
TF/TCR (TradeStatus = 'L' - Alleged)
> Expire Time
MI/TCR(TradeEditType = 'A')
TF/TCR (TradeStatus = 'E' - Expired)
TF/TCRA (TradeStatus = 'E' - Expired)

Figure 15-4 Trade Report Expiration

15.5. Intra-Firm Trade Report

Seller and buyer reside in the same member firm. Firm inserts the TradeReportEntry message. Trade is completed.

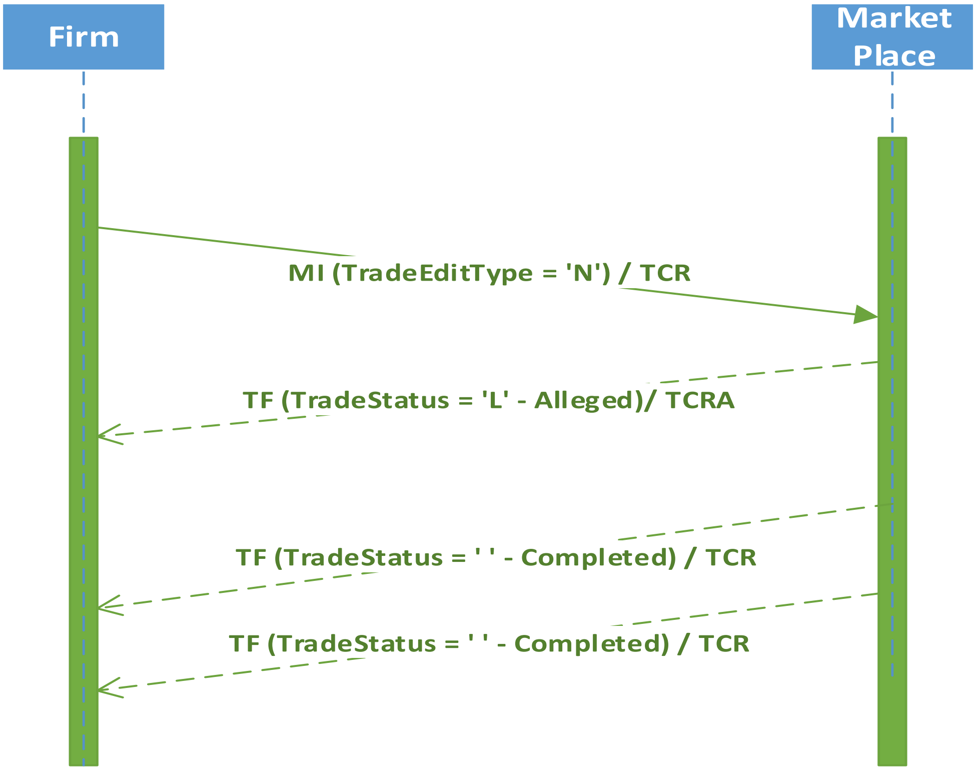

Firm
Market Place
MI (TradeEditType = 'N') / TCR
TF (TradeStatus = 'L' - Alleged)/ TCRA
TF (TradeStatus = ' ' - Completed) / TCR
TF (TradeStatus = ' ' - Completed) / TCR

Figure 15-5 Intra-Firm Trade Report

16. Appendix J. Production of MIFID II field 21 of RTS24

The following table describes the production of field 21 (New order, order modification, order cancellation, order rejections, partial or full execution) of MIFID II RTS 24 using the following confirmation messages:

- ConfirmOrderEntry (TB)
- ConfirmOrderEdit (TC)
- ConfirmOrderChange (TD)
- Trade (TF)

MIFID II – RTS 24 **Field 21**	**MessageType** **(TB/TC/TD/TF)**	OrderStatus	OrderSource	TD.SpecialConditions/TD.PriceType	Order Date	TR.OriginalMessageText
NEWO	TB				*Current _date*	
TRIG	TD	Not (Expired or Dissaproved)	‘ ‘ (AOM)	*SpecialConditions==’S’ or SpecialConditions==’D’ or PriceType==’C’*		
REME	TD	Not (Expired or Dissaproved)	‘C’ (CTCI) or ‘M’ (ORAMA/ETW)			
REMA	TD	Not (Expired or Dissaproved)	‘ ‘ (AOM)	*SpecialConditions!=’S’ and SpecialConditions!=’D’ and PriceType!=’C’*		
REMH	TD	Not (Expired or Dissaproved)	‘R’ (EMRW)			
REMO	TR					*MB, MD, MC*
EXPI	TD	Expired				
PARF	TF	Open				
FILL	TF	Matched				
CHME	TC/TD	TC.OrderStatus==Inactive or TC.OrderStatus==Open or	‘C’ (CTCI) or ‘M’ (ORAMA/ETW)			

		TD.ChangedOrderStatus=Dissapproved				
CHMO	TC/TD	TC.OrderStatus==Inactive or TC.OrderStatus==Open or TD.ChangedOrderStatus=Dissapproved	‘ ‘ (AOM) Or ‘R’ (EMRW)			
CAME	TC	Cancelled	‘C’ (CTCI) or ‘M’ (ORAMA/ETW)			
CAMO	TC	Cancelled	‘ ‘ (AOM) Or ‘R’ (EMRW)			
LOSR	TB					< *current_date*

17. Appendix K. Glossary of Terms

This glossary intends to familiarize programmers with the most common terminology.

Term	Definition
All or None, All or Nothing (AON)	Buy or sell order marked to signify that no partial matching is to be executed. The order will not be automatically cancelled, however if a complete match is not executed; to accomplish that, the order entry must be marked FOK (Fill or Kill).
At the Open (ATO) order	An order that is to be executed at the Opening Price.
At the Close (ATC) order	A market order that is to be executed at the closing price on the exchange of the stock named in the order. If it is not executed, the order is cancelled.
Average price	The average price per share of all trades resulting from a single order. This is a general definition and it depends on the case whether the term "Average" means Volume Weighted or something else.
BBO	See "Best bid and offer."
Best bid and offer	"BBO" The highest price a buyer is willing to pay, and the lowest price a seller is willing to offer, with the quantity they're willing to trade.
Better, or	An order qualifier that is generally assumed to mean "limit price or better." An order will execute at a price no worse than the specified limit price, and may execute at a better price (higher selling price for a sell order or a lower buying price for a buy order).
Block Trades (or Big Packages)	A Block trade is defined as a bilateral trade in excess of a pre-defined value for the product. This trade is done by participants that have selected one another by themselves.
Board	A logical trading area within a market wherein a specific trading procedure is utilized. In the OASIS the following are included: • Main Board • Odd Lot Board • Special Terms (Hit & Take) Board • Agreed Price Board (Trade Report (Pre-Agreed) and Cross Trade) • Forced Sales

Broadcasts	Broadcasts are items of public information generated centrally at the Exchange that are transmitted to the market participants in real-time (for example order book changes).
Broker (or Trader)	An ATHEX authorized individual who belongs to a Member and buys and sells financial instruments for the member's own account or on behalf of a Client.
Call Auction	The time when an Auction Price is calculated and orders entered during a Call Auction Phase are executed.
Call Auction Method (CAM)	In call auction method, the system will initially accept all valid order types and execute the orders that can be executed at the calculated "auction" price, after having calculated the "call" or "open" price for each security.
Ceiling	The maximum price a security can trade at during the day.
Closing Price	The price of the last transaction concluded.
Continues Automatic Matching Method (CAMM)	In continuous trading, trades are continuously and automatically being concluded by the system, by executing in real time orders given by the members.
Contra	Contra broker, order, side. The counterpart or counterparty to a trade. The buying broker enters a buy order. The contra side is the contra broker (seller) who entered the contra order (sell order) to which the buy order is matched.
Day order	An order term which specifies that an order's unexecuted quantity should be automatically cancelled by the system at the end of the trading day.
Depth	See "market depth."
Disclosed volume	Or "published volume" This is the quantity of an order that is included in a quote and publicly disseminated.
DSS	Dematerialized Securities System

Duration (Order)	An order term specifying how long unexecuted portions of an order are to stay in the order book before being cancelled. A “day” order has an order lifetime of the trading day it was entered and any unexecuted quantities will be cancelled at the end of the trading day. Other terms include Good unTil Cancelled (GTC) and Good Till Date (GTD).
Equity	Ownership interest possessed by shareholders. (Stock or Fixed Income)
ETS	The Equities Trading Server.
ETW	The Equities Trading Workstation is the member’s application needed to access the OASIS.
Fill or kill (FOK) order	An order condition specifying that the order must be cancelled if its entire volume cannot be immediately matched (the order may match against one or more contras).
Floor	The minimum price a security can trade at during the day.
Forced Sales	Type of trading facilitated by the ATHEX to meet the need of selling securities upon judicial or statutory request at irregular time periods in the ATHEX.
Good unTil Cancel (GTC order)	A good until cancel order remains in effect until it is matched or cancelled. Unmatched and un-cancelled GTC orders remain in the system for as long as their price is within the ceiling/floor limits for a security. The system will automatically cancel any GTC orders during the start-up period with limit prices outside of the ceiling floor/ limits for a security. For as long as the order is within the ceiling/floor limits it will remain in force there is no expiration of GTC orders.
Good unTil Date (GTD order)	A good until date order allows the trader to enter a specific date when the order will expire. Unmatched and un-cancelled GTD orders remain in the system for as long as their price is within the ceiling/floor limits for a security. The system will automatically cancel any GTD orders during the start-up period with limit prices outside of the ceiling floor/ limits for a security.
Good unTil Phase, (GTP order)	A GTP order is good only for the phase (state) in which it is entered.

Hit and Take trading method	By applying this method, the member has two choices; either to enter its order, or to choose to trade with an order already recorded in the system.
Immediate or cancel (IOC)	An order condition which requires immediate execution at the specified price. Any part of the order that cannot be matched immediately is automatically cancelled.
Index	See “market index.”
Lifetime, Life	See Duration (Order).
Limit down/up	Maximum price movement up and down for a security or instrument.
Limit order, (LMT)	An order with a specified maximum price or “priced order.” An order which specifies the maximum price which a buyer will pay or the minimum price a seller will accept.
Limit price	The maximum price at which an order may be executed. For example, a sell limit order can only be executed at the limit price or higher. Similarly, a buy limit order can be executed only at the limit price or lower.
Lot	The standard unit of trade (e.g. 10). It constitutes a regular unit of trade for a particular security.
Lot – Odd Lot	Securities trade made for less then the Round Lot.
Lot – Round Lot	Unit of trading on a securities exchange.
Market	The valid combination of groups of securities to be traded in a regulated manner by participating Members.
Market control	A term which refers to administrative functions such as opening the market, suspending trading, closing the market, etc.
Market depth	The display of all the available prices in the market, along with the aggregated number of shares available to these prices.

Market index	A market index is a statistical measure, often a weighted average, where individual securities are weighted by their market capitalization or percentage of the overall marketplace.
Market open	The starting time when orders can be entered.
Member	An ATHEX regulated firm company that engages in brokerage activities.
Minimum Fill, (MF order)	A Minimum Fill order required that a minimum quantity is required for execution of this order. An order with a minimum fill condition is executed only if the minimum volume (value for Fixed Income securities) can be satisfied.
OASIS	ATHEX trading platform
Odd lot	An order or trade for less than one round lot.
Opening price	The price of the first trade of the trading day (whenever happens) in a security, or the price calculated by the trading system at the opening phase (pre-opening phase).
Opening volume	The number of shares of the first trade in a security during the opening process. If no orders are matched during the opening process, this field is left blank throughout the trading day.
Order	A commitment to buy or sell a specified quantity of a security at an indicated price, subject to any attached terms or conditions. When two orders match, a trade is created.
Order driven	A term describing a market where price is directly determined by participant orders matching without intermediation, as opposed to a “price” or “quote” driven market where participant orders are matched against market maker quotes.
Order match	The process of comparing orders in the order book to create trades.
Order Types	The individual types and attributes of an order defined by the ATHEX and entered by Members into OASIS, in order to trade in a regulated manner

Partial	Or “partial fill.” Partial execution of an order which leaves an unfilled quantity on the order.
Position limit	Maximum amount of securities (position) that can be acquired by an individual holder or group of holders.
Pre – Agreed Trading (Trade Reporting)	This method supports report only trading between the members.
Pre – Open Period	The period which precedes the Call Auction. During the Pre-Call period orders are allowed to be entered into the system and quotes will be sent to members, but no matching (trading) will take place. Applicable only to Order Driven Trading
Right	Privilege granted to existing shareholders of a corporation. When a company proceeds to a share capital increase, one or more separate securities are created called “right (-s)”. The right or the rights of each share are traded separately in the market, for a period of at least one-month. The commencement and the duration of this period are externally defined by the company and are determined by the supervisory authorities of the ATHEX along with the opening price of the right.
Round lot	An order whose size is composed of one or more even round lots as defined by the rules of the ATHEX.
Rules and Regulations	The official document that describes the operational framework of the Derivatives market in Greece. http://www.ase.gr/content/en/about/regulations/files/ATHEX%20final.pdf
Share	Unit of equity ownership in a corporation. This ownership is represented by a stock certificate, which names the company and the shareholder. There are many different types of shares such as: common, preferred, bearer, registered.
Stock	Ownership of a Corporation represented by Shares That are claim on the corporation’s earnings and assets.
Surveillance	The process of monitoring trading activity and investigating activity to ensure that the rules are being followed.

Tick	Smallest possible price movement of a security.
Tick size	The legal price increment for a given price range, i.e. the price spread increment.
Trade	An agreement to deliver securities/contracts in order to receive money & Vice versa.
Trade number	A unique number assigned to each trade by the system.
Trading halt	A situation where trading is temporarily stopped in one or more securities with the expectation the security will resume trading at some time during the same trading day.
Trading Method	The method by which financial instruments are traded by Members. The OASIS Trading Methods are: Continuous Automatic Order Matching (CAMM), Call Auction (CAM), Hit & Take, Trade Reporting, Cross, Forced Sales, Quote Driven with Market Makers.
Trading Phases (or Trading Periods)	A timeframe period wherein a specific activity is allowed to be conducted on a market. For example: Start-up, pre-open phase, continuous trading, closing price trading, and market runoff.
Trading Procedures	The logical grouping of Trading Methods that has common characteristics. The OASIS Trading Procedures are: Order Driven, Agreed Price, Special Auctions, Hit & Take, and Quote Driven.
XOrder/XNet	XOrder Server allows the market participants to have indirect market access to the trading servers of the Exchanges connected to XNet