iLink 3 Binary Order Entry - Session Layer

iLink 3 uses the FIX Performance (FIXP) protocol to establish and manage bi-directional sessions. Per the FIXP protocol, a FIX session is defined as a bi-directional stream of ordered messages between two parties within a continuous sequence number series. With iLink 3, the Market Segment Gateways (MSGW) will support weekly client FIX sessions and also support mid-week initialization.

FIXP Session Attributes

In iLink 3, for each customer, the MSGW starts the session at the beginning of the week. A FIXP session will be maintained by default on a weekly basis; however, the customer can Negotiate, Establish, and Terminate multiple times. Each FIXP session will also be represented with a Universally Unique ID (UUID), which should be a current timestamp.

The term, *FIX connection*, has a new connotation in iLink 3. A successful logon consists of a two-step process that involves first Negotiation and then Establishment.

The exchange will support a FIXP session created at the beginning of the week all through the end of that week. The customer can then Establish and Terminate FIX connections multiple times intra-day or at the end of each day, with the same UUID or Negotiate the new FIXP session with a new UUID as well.

Contents

Universally Unique Identifier (UUID)

Each FIXP session established at the start of the week, intra-day, or at the beginning of each trading day is represented with a unique UUID value set by the customer as a 64-bit value. CME Group recommends using the system timestamp which represents the number of microseconds since epoch (Jan 1, 1970) as the timestamp. Business messages received from the exchange are recoverable for that sequence number + UUID combination.

UUID Rules

The following rules apply to the UUID:

- UUID must be sent in the Negotiate message and that ID is used for the lifetime of the connection.
- Customer must reset sequence numbers back to 1 for each direction (from customer to CME and from CME to customer) every time they negotiate with new UUID.
- UUID in the Negotiate message is greater than the UUID used in the last successful Negotiate/Establish message.
    - This will prevent usage of duplicate UUID's intraweek, which can affect subsequent retransmission of those messages.
    - For this reason it is strongly recommended to use current timestamp value (number of microseconds since epoch) as UUID.
- RequestTimestamp must be current timestamp in nanoseconds.
- Customers will be able to use the same UUID across all segments at the same time.
- One UUID can be used by more than one customer.
- A UUID previously used on segment ‘X’ can be used to negotiate on segment ‘Y’ if never used on ‘Y’ by the given customer and higher than the most recent UUID used by that customer on ‘Y’.

Re-Setting UUID

In the event of customer application failure, it may be necessary to reset the UUID intra-session on the same trading day to force synchronization of sequence numbers to and from the exchange.

Important:

- Do not Terminate the FIXP session and Re-Negotiate with a new UUID as a normal response to a Not Applied message
- Re-Negotiate with a new UUID should be used only to recover from a disaster situation that cannot be recovered via the use of Sequence message
- Re-Negotiating with a new UUID will mean recovering messages sent by the exchange in the previous FIXP session with the previous UUID
- Re-Negotiating with a new UUID resets the inbound and outbound sequence numbers back to "1" so sending a higher than expected sequence number to the exchange in the Establish message will result in successful Establishment followed by a Not Applied message and the new NextSeqNo in the Establish message will be adopted as the new expected sequence number

Weekly Reset

CME will internally reset its inbound and outbound sequence numbers to “1” at the start of each week. The customer should also reset their sequence numbers in both directions prior to negotiating and establishing a FIX session on Sunday.

When FIXP sessions are established at the start of each week, the customer must initiate a session by sending the Establish message (after Negotiation) with NextSeqNo as “1”.

CME will respond with an Establishment Acknowledgment confirmation with NextSeqNo as “1”.

- If the customer does not reset the outbound sequence number to the exchange to “1” at the start of each week, they will receive a Terminate message informing them of this error in the Code field.
- If a new UUID is assigned to subsequent FIXP sessions after the first, the sequence stream is reset to '1' in both directions (from customer to exchange and exchange to customer).

Session Security

The customer will authenticate an iLink session by digitally signing FIXP Negotiate and Establish messages using a hashed message authentication code (HMAC). In general, the customer interaction with CME will be as follows:

1. Customer logs into the CME Self Service Portal using 2-factor authentication over a secure channel (HTTPS).
2. CME generates a pair of cryptographically random keys—an Access Key and a Secret Key—for the customer. The keys are provided to the customer and also securely stored in a secure vault with CME.
3. Customer connects to iLink and sends a digitally signed Negotiate or Establish message. The digital signature is generated using a hashed message authentication code (HMAC) based on the contents of the message and the Secret Key and is included in the Login message or Negotiate and Establish messages sent to CME.
4. CME processes the Negotiate or Establish messages, calculates the digital signature using the same HMAC based process, and compares the calculated signature with the signature provided on the customer's Negotiate or Establish messages. If the signatures match, the Negotiate or Establish messages are authentic and validates that the sender has the Secret Key.

The details of of calculating the digital signature are described below.

Steps to Sign the Negotiate and Establish Messages

1. Create the canonical Negotiate or Establish message
2. Create Signature using Secret Key provided by CME and Canonical Message
3. Populate HMAC Signature plus Access Key ID fields of the Negotiate and Establish messages

Only the field value—not the field name—must be used for the calculation of HMAC signature.

Example: where SessionID=<ABC>, use only 'ABC'.

Step 1 - Create the Canonical Message

These field values must be assembled in this order with the values concatenated into a single string delimited by the new line character (i.e. '\n').

Negotiate Message

The required FIXP fields in Negotiate message are:

- RequestTimestamp
- UUID
- SessionID
- FirmID

An Example Canonical message for Negotiate:

"RequestTimestamp" + "\n" + "UUID" + "\n" + "SessionID+ "\n" + FirmID

Sample Code to Generate Canonical Negotiate Message

```
public String prepareCanonicalMessageForNegotiate(){

long requestTimestamp = 1563720650008L;

long uUID = 1563720660068L;

String session = "ABC";

String firmID = "007";


StringBuilder canonicalMsg = new StringBuilder();

canonicalMsg.append(requestTimestamp).append(New_Line);

canonicalMsg.append(uUID).append(New_Line);

canonicalMsg.append(session).append(New_Line);

canonicalMsg.append(firmID);

return canonicalMsg.toString();

}
```

Establish Message

The required FIXP fields in the Establish message are:

- RequestTimestamp
- UUID
- SessionID
- FirmID
- TradingSystemName
- TradingSystemVersion
- TradingSystemVendor
- NextSeqNo
- KeepAliveInterval

An Example Canonical message for Establish:

"RequestTimestamp" + "\n" + "UUID" + "\n" + "SessionID+ "\n" + FirmID+ "\n" + "TradingSystemName" + "\n" + "TradingSystemVersion" + "\n" + "TradingSystemvendor" + "\n" + "NextSeqNo" + "\n" + "KeepAliveInterval"

Sample Code to Generate Canonical Establish Message

```
public String prepareCanonicalMessage For Establish(){
        long requestTimestamp = 1563720650008L;
        long uUID = 1563720660068L
        String session = "ABC";
        String firmID = "007";
        String tradingSystemName = " ";
        String tradingVersionId = " ";
        String tradingSystemVendorId = " ";
        long nextSeqNo = 1;
        long keepAliveInterval = 30;

        StringBuilder canonicalMsg = new StringBuilder();
        canonicalMsg.append(requestTimestamp).append(New_Line);
        canonicalMsg.append(uUID).append(New_Line);
        canonicalMsg.append(session).append(New_Line);
        canonicalMsg.append(firmID).append(New_Line);
        canonicalMsg.append(tradingSystemName).append(New_Line);
        canonicalMsg.append(tradingSystemVersionId).append(New_Line);
        canonicalMsg.append(tradingSystemVendorId).append(New_Line);
        canonicalMsg.append(nextSeqNo).append(New_Line);
        canonicalMsg.append(keepAliveInterval);
        return canonicalMsg.toString();
}
```

Step 2 - Create Signature Using the Secret Key and Canonical FIX Message

The signature is a hash (digest) of the canonical message created in Step 1 using the Secret Key provided by CME.

The Secret Key downloaded from Request Center is **Base64 URL Encoded**. Customers must decode the secret key first.

Example of Creating Signature Using HMAC SHA256 in Java 8+

The HMAC signature is populated in a fixed length string field (i.e. String32Req) since SHA-256 signature is always 32 bytes.

> While real logic provides two options to encode this field—as a string and as a byte array—the byte array option should be used.

Encoding options for SHA-256 (256 bits long):

- Base64URL encoding FIX ASCII: 6 bits per char = CHAR(44)including padding character
- Hex: 4 bits per char = CHAR(64)
  - HEX example 617633EFEF2F98DCA32DD3BD8407122B9B375CF4BC076930F24C1F7A0F930094
- Binary: 8 bits per byte = BINARY(32)
  - Binary example byte array:[97, 118, 51, -17, -17, 47, -104, -36, -93, 45, -45, -67, -124, 7, 18, 43, -101, 55, 92, -12, -68, 7, 105, 48, -14, 76, 31, 122, 15, -109, 0, -108]

> The signature should be encoded as a byte array.

Signature calculation in Java:

```
public byte[] calculateHMAC(final String canonicalRequest, final String userKey) {
    byte[] hash = null;

    try
    {
      // Init HMAC instance
      Mac sha256HMAC;
      sha256HMAC = Mac.getInstance("HmacSHA256");

      // Initialize HMAC instance with the key
      // Decode the key first, since it is base64url encoded

      byte[] decodedUserKey = Base64.getUrlDecoder().decode(userKey);
      SecretKeySpec secretKey = new SecretKeySpec(decodedUserKey, "HmacSHA256");
      sha256HMAC.init(secretKey);


      // Calculate HMAC
      hash = sha256HMAC.doFinal(canonicalRequest.getBytes("UTF-8"));

    } catch (NoSuchAlgorithmException | InvalidKeyException | IllegalStateException |
UnsupportedEncodingException e)
    {
      e.printStackTrace();
    }

    return hash;
  }


}
```

Step 3 - Populate HMAC Signature Plus Access Key ID fields of the Negotiate and Establish Messages

- tag 39004-AccessKeyID - contains the AccessKeyID
- tag 39005-HMACSignature - contains the HMAC signature

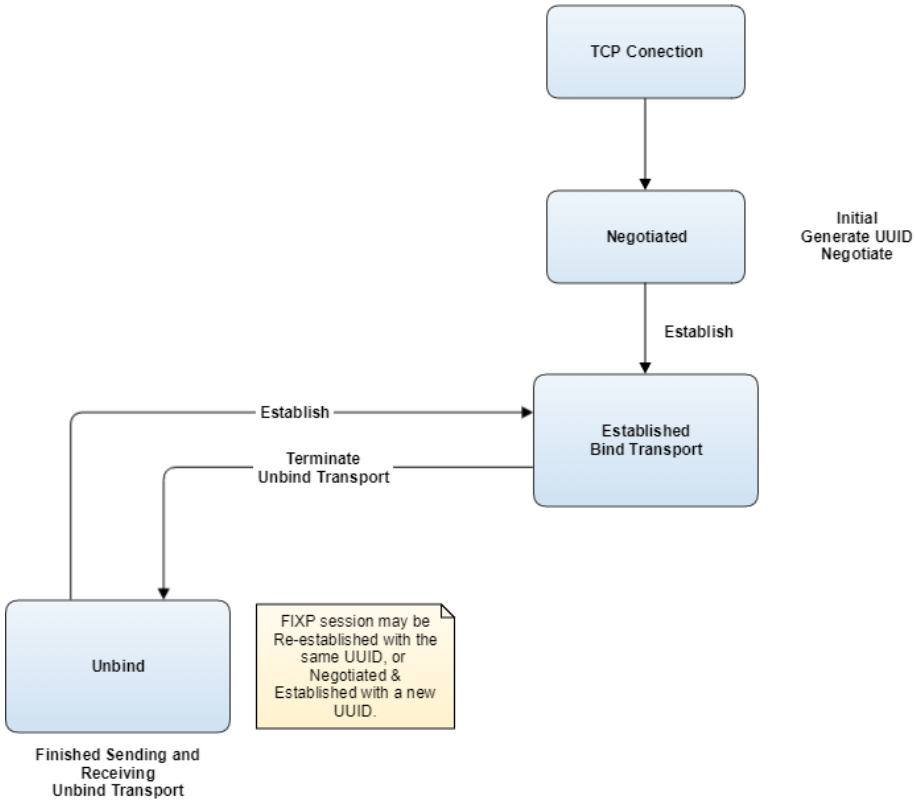
FIXP Session Messages

FIXP session messages are summarized as follows.

Stage	Message Name	From | To	Purpose
Initialization	Negotiate	Client System to CME Globex	Initiates Connection
	Negotiation Response	CME Globex to Client System	Accepts Connection
	Negotiation Reject	CME Globex to Client System	Rejects Connection
Binding	Establish	Client System to CME Globex	Binds Connection
	Establishment Acknowledgment	CME Globex to Client System	Accepts Binding
	Establishment Reject	CME Globex to Client System	Rejects Binding
Transferring	Sequence	Client System to CME Globex CME Globex to Client System	Initiates a Sequenced Flow, Keep Alive
	Retransmit Request	Client System to CME Globex	Requests Replay
	Retransmission	CME Globex to Client System	Accepts Replay
	Retransmit Reject	CME Globex to Client System	Rejects Replay
	Not Applied	CME Globex to Client System	Negative Acknowledgment of Missed Messages
Unbinding	Terminate	Client System to CME Globex CME Globex to Client System	Kill Connection Ungracefully or Gracefully

Overview Diagram

This diagram provides an overview of the session Establishment, Binding, and Unbinding process.



Initialization and Binding

The customer uses the Negotiate and Establish messages to authenticate themselves with the Market Segment Gateway. In iLink 3, there are three different types of Initialization and Binding operations

Beginning of Week Initialization and Binding

Beginning of Week initialization and Binding will involve the very first messages (Negotiate and Establish) that the customer sends for the week. The customer must reset their inbound and outbound sequence numbers to “1” prior to initialization at the start of the week as well as pick a globally unique UUID for each FIXP session.

If the client system to CME Globex outbound sequence number is not reset to “1” at the beginning of the week prior to Start of Week Initialization, and the client sends a Establish message (Negotiate does not have NextSeqNo), the Market Segment Gateway responds with a Terminate message. The Terminate message will have a Code indicating a failure to reset sequence numbers at the start of the week. The customer must then reset sequence numbers and reattempt the Initialization.

GTC and GTD Orders - Beginning of Week

Good Till Cancel (GTC) or Good Till Date (GTD) orders sent by a customer from the previous week continue to be eligible for execution after Termination (done for day).

- If a fill is generated for the GTC or GTD order in the current week before Start of Week Initialization, then the exchange processes the Execution Report and increments its previous sequence number for the previous UUID (which is the default UUID of 0) accordingly.
- When the customer sends Negotiate and Establish messages to Initialize at the Start of the Week for the current week, they will receive an Establishment Acknowledgment message with
    - PreviousSeqNo = “1”
    - PreviousUUID="0". This way the client knows that a message was published using the default CME assigned UUID of 0 since iLink 3 requires the exchange also to reset its sequence numbers to “1” and UUID to "0" prior to Start of Week Initialization.
- As a result, the client application sends a Retransmit Request for the PreviousUUID and PreviousSeqNo and receives the undelivered Execution Report on the GTC order
- The sequence gap is now filled and normal processing continues.

Mid-Week Initialization and Binding

Mid-Week Initialization and Binding will involve Negotiating and Establishing a new FIXP session once the previous one has ended gracefully following Termination. The customer must populate NextSeqNo with the correct value from the previous sequence stream associated with the previous UUID and reattempt the Initialization, or reset sequence number to 1 with the new UUID.

If the customer's inbound sequence number is reset to “1” when Initializing a new FIXP session with the same UUID as the previous one and the customer sends an Establish message (Negotiate does not have NextSeqNo), then the Market Segment Gateway responds with a Terminate message. The Terminate message will have a code indicating an error condition regarding usage of lower than expected sequence numbers. The customer must populate NextSeqNo with the correct value from the previous sequence stream associated with the previous UUID, or populate NextSeqNo with the correct value from the previous sequence stream associated with the previous UUID and reattempt the Initialization.

GTC and GTD Orders - Mid-Week

Good Till Cancel (GTC) or Good Till Date (GTD) orders sent by a customer from the previous trading day continue to be eligible for execution after Termination.

- If a fill is generated for the GTC or GTD order in the current trading day before Initialization then the exchange processes the Execution Report and increments its previous sequence number for the previous UUID which is the last known UUID from the previous session.
- When the customer sends Negotiate and Establish messages to Initialize then they will receive an Establishment Acknowledgment message with:
  - PreviousSeqNo as “last known value+1”. This is higher than the client’s last known expected inbound sequence number which is "last known value" for the PreviousUUID.
  - PreviousUUID="last known value".
- As a result, the client application sends a Retransmit Request for the PreviousUUID and PreviousSeqNo and receives the undelivered Execution Reports on the GTC order.
- The sequence gap is now filled and normal processing continues.

Intra-Session Binding

Intra-session binding is applicable when the FIXP session loses connectivity with the exchange for any reason such as Termination due to a protocol violation, failure, etc. In such an event:

- The customer must Establish the FIXP session without Negotiation (Binding without Initialization).
- It is recommended that the same UUID be used, else the sequence stream will not be contiguous with the previous FIXP session.
- If a different UUID is used, the customer must reset their inbound and outbound sequence numbers to “1”.
- If the same UUID is used, the customer must use the next sequential inbound and outbound sequence numbers.

Intra-Session Binding is used for any subsequent binding after a successful Beginning of Week or Mid-Week initialization and Binding.

Intra-Session Binding uses the sequence number series that continues from the next sequence number where the customer was Terminated for the same UUID. As a result, the Intra-Session Establish message cannot have a NextSeqNo set to “1" for that same UUID.

Setting inbound and outbound sequence numbers back to "1" is possible only with the use of a new UUID which requires Negotiation and Establishment.

If any of the above requirements are not met, the client application receives a Terminate message. In addition, iLink 3 does not increment its inbound sequence number.

Intra-Session Establish provides handling for undelivered messages while the client application has not established a FIXP session.

- While the customer is not established, Execution Reports, etc. are processed by the exchange.
- As a result, the customer may receive an Establishment Acknowledgment message with PreviousSeqNo higher than expected (i.e., the Market Segment Gateway processed those Execution Reports and incremented its outbound sequence number accordingly) for the previously used UUID.
- The client application submits a Retransmit Request message for those messages persisted while the customer was not established.

Session Negotiation

A FIXP session connection is initiated by the customer using a Negotiation message. The UUID must be sent in the Negotiation message and that ID is used for the lifetime of the FIXP session. Negotiation is provided as a mechanism used for the customer to declare what ID they will be using. There is no concept of resetting a FIXP session. Instead of starting over a FIXP session, a new FIXP session is negotiated with a new UUID.

Clients must connect to their session’s assigned IP and port before session negotiation; otherwise, the Negotiation message will be rejected.

The Negotiate message also identifies the types of message flow in each direction of a FIXP session using the FlowType field. The types of message flow supported by iLink 3 are *Idempotent* from customer to exchange and *Recoverable* from exchange to customer.

- FlowType=**Recoverable** has the highest delivery guarantee and guarantees exactly-once message delivery. If gaps are detected by the customer, then missed messages may be recovered by retransmission.
- FlowType=**Idempotent** guarantees at-most-once delivery. If gaps are detected by the exchange, then the customer is notified, but no attempt is made by the exchange to recover missed messages. Any course of action in this scenario is incumbent upon the customer.

The Negotiate message should be used in these circumstances:

- Beginning of the week (start sequence numbers from 1)

The Negotiate message can be used in these circumstances as well:

- Midweek with new UUID after Termination of previous session by customer
- Midweek with new UUID after ungraceful termination (with error code) and to restart sequence numbers from 1
- Midweek with new UUID after graceful termination (without error code) and to restart sequence numbers from 1

The following required string fields cannot contain empty bytes:

- HMACSignature
- AccessKeyID
- Session
- Firm

The following Integer fields will be checked for these boundary conditions:

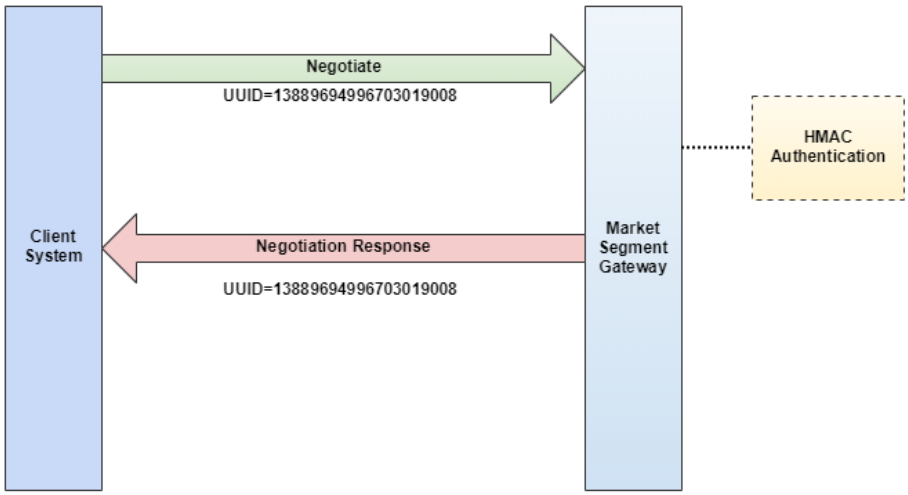
- UUID
- RequestTimestamp

The customer should send a Negotiate message to the exchange and await acknowledgement.

- When Negotiate message from customer is accepted by CME then CME will return a Negotiation Response.
- When Negotiate message from customer is rejected by CME then CME will return a Negotiation Reject.
- If no response is forthcoming from the exchange after two times the recommended KeepAliveInterval has lapsed then Negotiation can be retried or out-of-band inquiry may be made to determine application readiness.

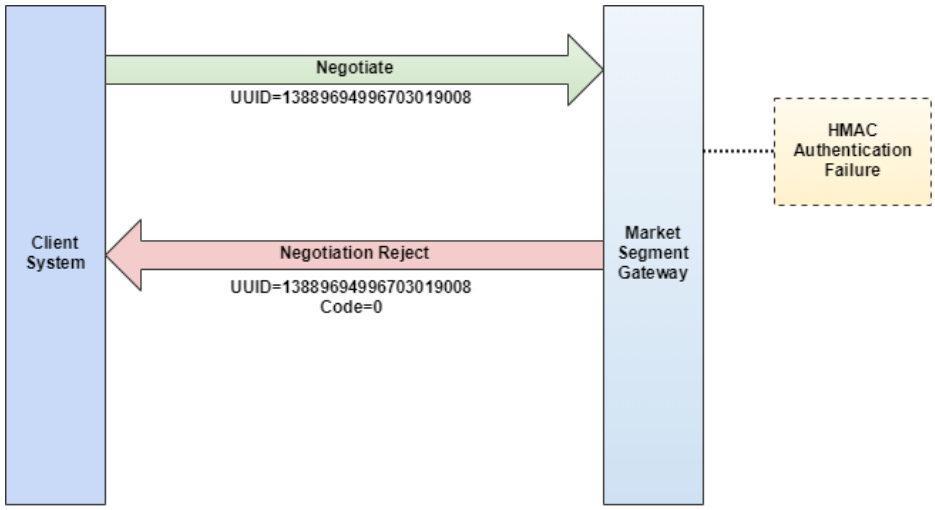
Negotiate Message Accepted

This diagram shows an example of Negotiation Accepted.



Negotiation Rejected

This diagram shows an example of negotiation Rejected.



Session Establishment

After negotiation is complete, the customer must send an Establish message to start assigning sequence numbers. Once established, the customer can proceed to submit orders and quotes. The established state is concurrent with the lifetime of a TCP connection. A customer may re-establish a previous FIXP session after reconnecting without any further negotiation (Intra-Session Initialization). Thus, Establish binds the FIXP session to the new transport instance.

Both the customer and the exchange should signal each other that a disconnection is about to occur using a Terminate message. This unbinds the transport from the FIXP session, but it does not end a logical session hence there is no need to Negotiate again intra-day for the same trading day.

- If connection is disconnected then the customer may re-establish the same session (same UUID) after reconnecting without any further negotiation which ensures that the sequence numbers will continue
    - Messages published after disconnection (cancel on disconnect) may be recovered in the next session
- If connection is terminated then the customer must negotiate and establish a new session with the a new UUID after reconnecting and this resets the sequence numbers back to 1 both ways (customer to CME and CME to customer)
    - Messages published after Termination (mass quote cancels) may be recovered in the next session through previous UUID
- If connection is terminated then the customer may also re-establish the same session (same UUID) after reconnecting without any further negotiation which ensures that the sequence numbers will continue
    - Messages published after termination (mass quote cancels) may be recovered through the same UUID

Once the Establish message is sent by the customer to CME then they must wait for an acknowledgment before initiating any other action.

- If no response is received after KeepAliveInterval, then send the Establish message again. After the second attempt, manual escalation may be required.

A FIXP session that has a recoverable flow may be re-established by sending Establish with the same session UUID and exchange of business messages may continue until all business transactions are finished.

An Establish message attempts to bind the specified logical FIXP session to the transport that the message is passed over. The response to Establish is either Establishment Acknowledgmen tor Establishment Reject. The exchange will evaluate NextSeqNo in the Establish message to determine whether it missed any messages after re-establishment. If so, it will immediately send a NotApplied message after sending EstablishmentAck.

Establish message should be used in these circumstances:

- Beginning of the week after Negotiate and starting sequence numbers from 1 with same UUID as Negotiate.UUID
- Midweek after disconnection and continuing sequence numbers from the same session such that UUID=previous Establish.UUID
- Midweek after termination from CME and continuing sequence numbers from the same session such that UUID=previous Establish.UUID
- Midweek after termination from customer and continuing sequence numbers from the same session such that UUID=previous Establish.UUID
- Midweek after termination from customer and negotiation (with new UUID) and restart sequence numbers from 1 such that UUID=Negotiate. UUID
- Midweek after termination from CME and negotiation (with new UUID) and restart sequence numbers from 1 such that UUID=Negotiate.UUID
- Midweek after termination from customer and negotiation (with new UUID) and restart sequence numbers from 1 such that UUID=Negotiate. UUID

If the Establish message is not sent within a timeout interval of 60 seconds after the Negotiation Response then the session will be Terminated with code=1

The following required string fields cannot contain empty bytes:

- HMACSignature
- AccessKeyID
- Session

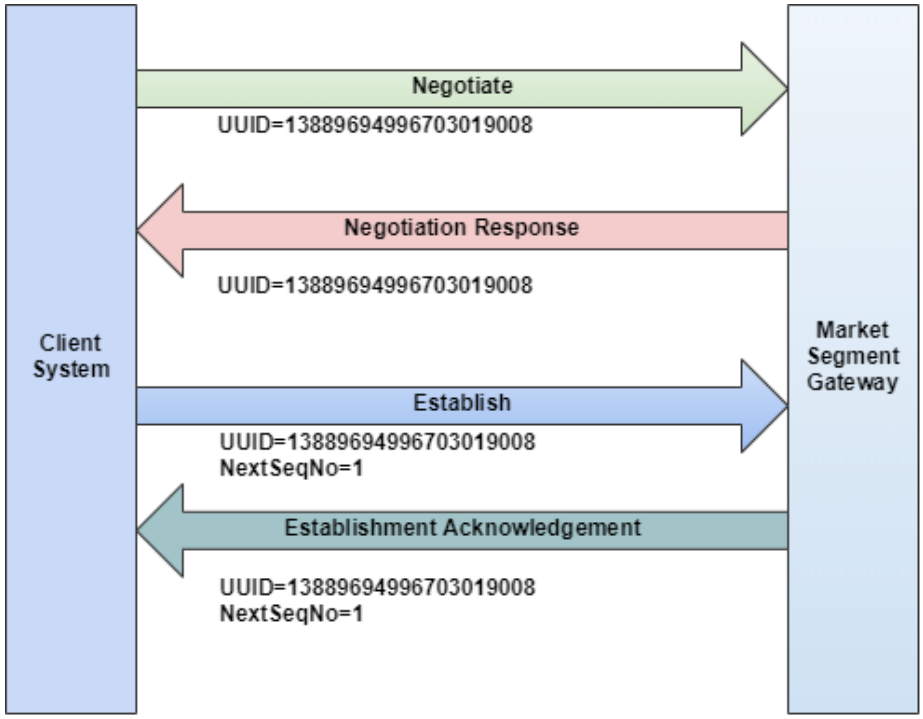
- Firm
- ApplicationSystemName
- TradingSystemVersion
- ApplicationSystemVendor

The following integer fields will be checked for these boundary conditions:

- UUID
- RequestTimestamp
- KeepAliveInterval
- NextSeqNo

Establishment Accepted

This diagram shows an example of Establishment Accepted.



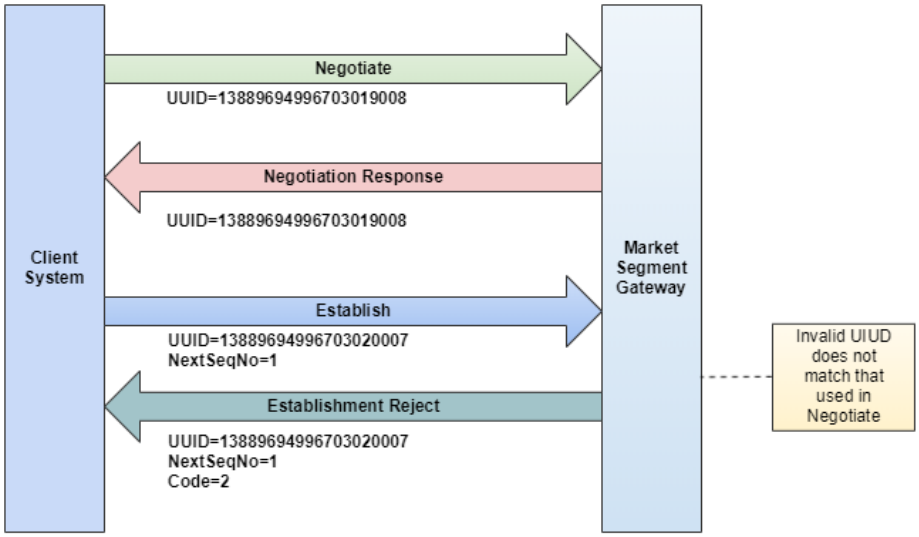
- When the Establish message from the customer is accepted by CME, then CME will return an Establishment Acknowledgment.
  - MessageSequenceNumber will be populated by CME accordingly
    - Beginning of the Week MsgSeqNo=1
    - Midweek MsgSeqNo=LastMsgSeqNo+1 if UUID=Last Establish.UUID and Establish without Negotiate
    - Midweek MsgSeqNo=1 if Establish used after Negotiate
  - CME Globex will notify the customer if any messages were published by CME while they were not connected with either default UUID assigned by CME or last known customer assigned UUID.

Since the communication flow from the exchange is recoverable, it will fill the NextSeqNo, PreviousUUID, and PreviousSeqNo fields appropriately, allowing the customer to start requesting the replay of messages that they have not received.

- The customer should evaluate NextSeqNo, PreviousSeqNo and PreviousUUID to determine whether they have missed any messages after re-establishment of a recoverable flow. If so, the customer can then immediately send a Retransmit Request.

Establishment Rejected

This diagram shows an example of Establishment Rejected.



- When the Establish message from the customer is rejected by CME, CME will return an Establishment Reject.

Transferring

Sequence

Not all FIXP messages are assigned a sequence number, but instead communicate the sequence number of the *next* business message or the *last* business message.

- All business messages will be assigned sequence numbers and FIXP messages refer to these sequence numbers
- This is unlike the official FIXP specification in which business messages have implicit sequence numbers
- Sequence message is used in both Recoverable (CME to customer) and Idempotent (customer to CME) flows
- Sequence numbering supports ordered delivery and recovery of messages

The Sequence message serves four purposes in an iLink 3 FIXP session including:

- *de facto* heartbeat message
- test request ping message
- failover trigger message (from primary to backup)
- sequence reset to let the exchange know what sequence number to expect next from the client system

This will let the exchange sync up the NextSeqNo with its own inbound sequence counter and identify a gap if any. Likewise the customer can infer what sequence number to expect next from the exchange and can sync up the value of NextSeqNo with their own inbound sequence counter and identify a gap if any.

The Sequence message can also be sent in response to a NotApplied message as a gap fill if it contains a higher than expected NextSeqNo value and not lower than expected.

The Sequence message is also sent at regular intervals similar to the heartbeat message to ensure that the connection between the customer and exchange is in a normal state. The KeepAliveInterval is specified while Binding in the Establish message. If there is other messaging activity in progress then the Sequence message need not be sent unless a KeepAliveInterval lapses, in which case it will function as a test request ping message.

- It is recommended to set the KeepAliveInterval at 5-60 seconds
- The exchange will send the Sequence message interleaved with Retransmission responses to a Retransmit Request as well

In the absence of other messaging activity, if a Sequence message is not received after 2 times the KeepAliveInterval has lapsed, then this is considered a protocol violation and will lead to the FIXP session being terminated.

A sequence message is used in these circumstances:

- From customer - to reset sequence number in response to Not Applied message sent by CME when CME detects a sequence gap from customer
- From CME - as a heartbeat message to be sent when a KeepAliveInterval interval from CME lapses and no other message is sent to customer
- From customer - as a heartbeat message to be sent when a KeepAliveInterval interval from customer lapses and no other message is sent to CME
- From CME - when a KeepAliveInterval of the customer lapses without having received any message from them then send message with Kee pAliveIntervalLapsed=1 as a warning before initiating disconnect of socket connection

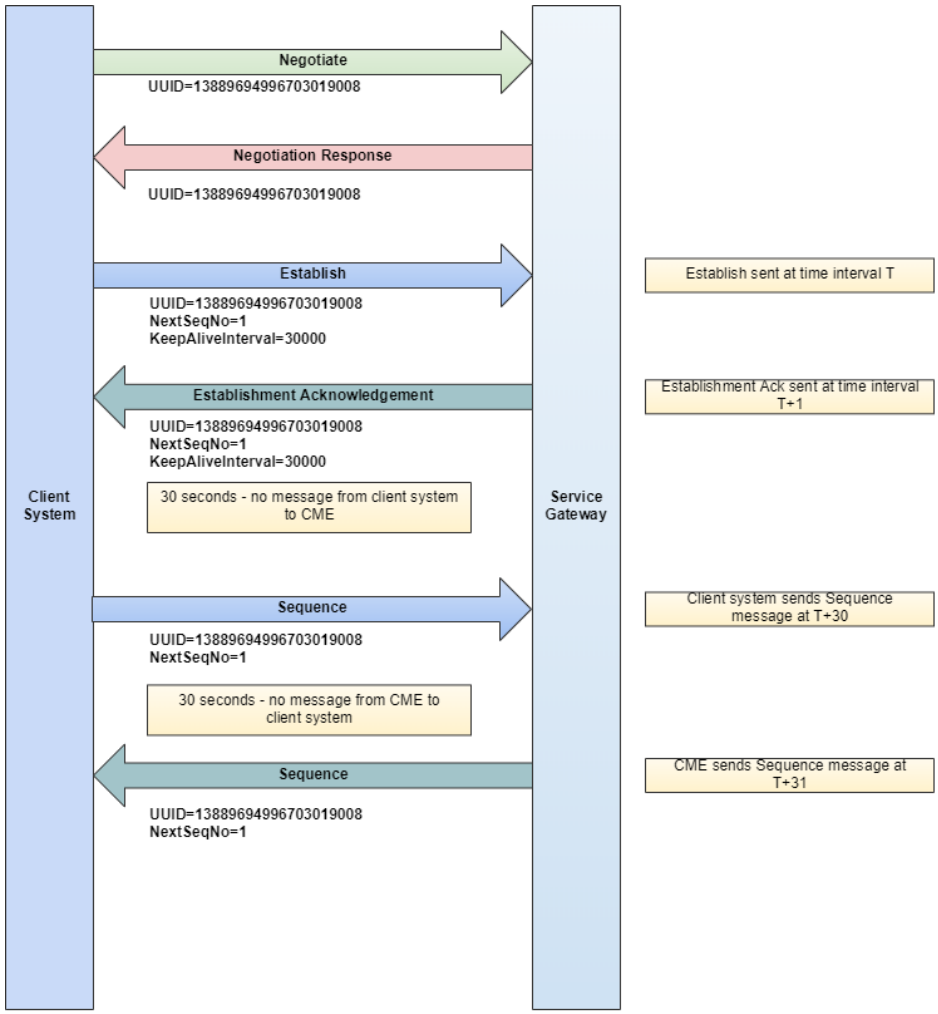
- From customer - when a KeepAliveInterval of CME lapses without having received any message from CME then send message with KeepAliveIntervalLapsed=1 as a warning before initiating disconnect of socket connection

The following integer fields will be checked for these boundary conditions:

- UUID
- NextSeqNo
- KeepAliveIntervalLapsed
- FTI

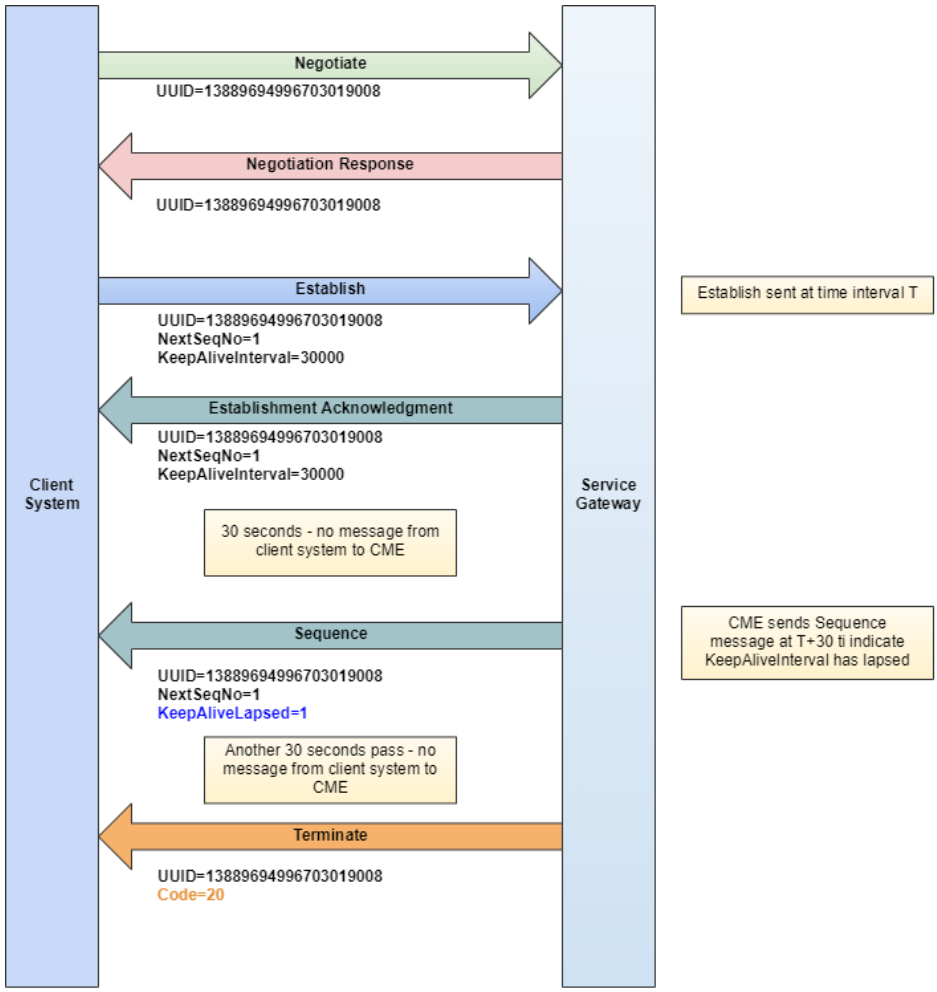
Heartbeat

This diagram shows an example of Sequence used as a heartbeat.



Warning

This diagram shows an example of Sequence used as a Warning due to KeepAliveInterval lapsing prior to Terminate.



Retransmit Request

The Retransmit Request message is sent to request a retransmission of missed messages when the customer detects a gap in the incoming sequence number series from the exchange. In iLink 3, the Retransmit Request is used to request all messages subsequent to a particular message by using the following fields:

- UUID: the UUID associated with the sequence numbers being requested
- FromSeqNo: message sequence number of first message in range to be resent
- Count: count of messages being requested (if defined as zero then this means infinity or latest sequence number)

The exchange itself will not send a Retransmit Request to recover missing messages, instead a Not Applied message is sent to indicate that the exchange has seen a gap in sequence numbers from the customer and it is the customer who can choose how best to respond to this:

- Send a Sequence message with the appropriate value in NextSeqNo where the customer wants to start sending messages from
- Retransmitting the missed messages
- Ignoring the NotApplied message and keep trying to moving ahead
    - This not advisable since the exchange will keep sending a Not Applied message for the same sequence gap
- Terminating the FIXP session and Re-Establishing with a new UUID
    - This is not advisable and should be used only in extreme failure cases to recover from lower than expected sequence number error condition since the sequence streams get reset back to '1' both ways between customer and exchange

Since the customer is receiving a recoverable flow from CME, they can use Retransmit Request to recover missed messages. When the customer sends a Retransmit Request message, the exchange resends all reject messages and/or execution responses using the same message sequence numbers as the original transmissions along with Sequence messages if any. FIXP messages are not replayed, only business messages.

- The Gateway resends messages for the current UUID or all previously used UUIDs for that week only. Messages from previous weeks are not resent.
- If the customer sends a Retransmit Request message to the Gateway—even if the Gateway has already sent a Not Applied message—then that request will be fulfilled by the exchange.

- The customer should accept messages with a higher sequence number from Market Segment Gateway after recognizing a gap. However, their application should queue messages for in-sequence processing after a requested retransmission is received and completed from the Gateway.
- Retransmission implies that subsequent messages are sequenced with a lower sequence number in response to a Retransmit Request.
- Retransmission responses from the exchange to customer may be inter-leaved with other real-time messages based on the PossRetransFlag (Tag 9765) such that replayed messages will be 9765=1 and original messages will be 9765=0.
- It is the customer's responsibility to determine whether the current batch fills the original gap. If not, then they should send another Retransmit Request for the remainder. Requests and responses should proceed iteratively until all desired messages have been retransmitted. This can be done by combining NextSeqNo+Count from all Retransmission messages.
- If the exchange is unable to retransmit requested messages, it responds with a Retransmit Reject message.
- If a customer Retransmit Request exceeds the maximum resend limit (2,500 messages) then the exchange sends a Retransmit Reject message. The customer should initiate separate Retransmit Requests for each batch of 2500 messages.
- Only one Retransmit Request is allowed in-flight at a time per FIXP session. Another Retransmit Request must not be sent until the previous request has concluded otherwise it will be rejected.

The following integer fields will be checked for these boundary conditions:

- UUID
- LastUUID
- RequestTimestamp
- FromSeqNo
- MsgCount

Retransmission

Retransmission message is sent in response to a valid Retransmit Request from customer.

When RetransmitRequest message from customer is accepted by CME, then CME will return a Retransmission followed by replaying business messages.

- If no messages are replayed then the Retransmission message will be followed by a Sequence message confirming the next sequence number to be sent for the current UUID.

While messages are being replayed, they may be interleaved with other real-time messages; tag 9756-PossRetransFlag on business messages provides the means to distinguish replayed messages from real-time messages:

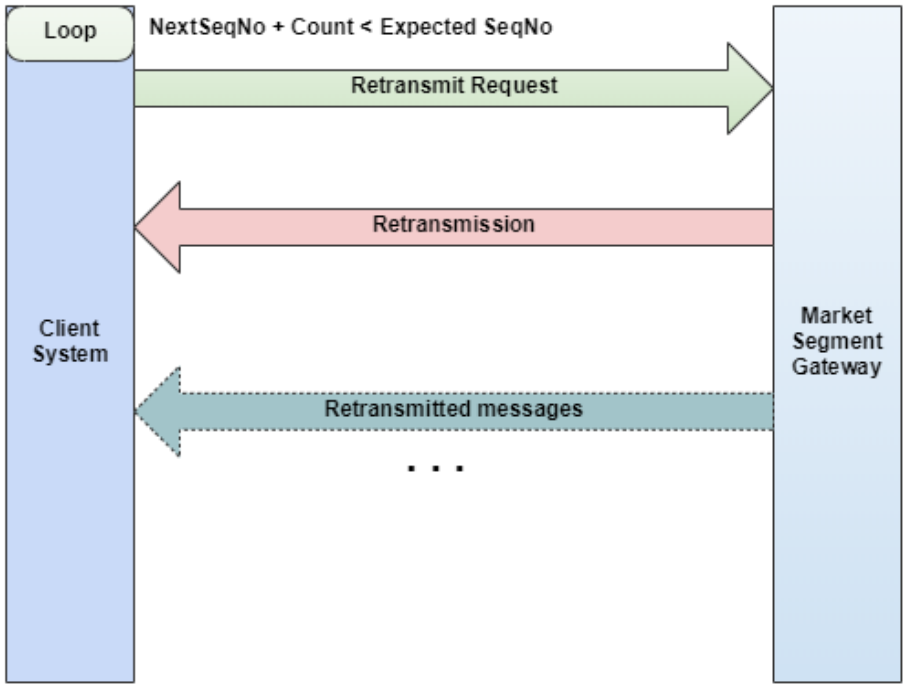
- tag 9765=1; replayed messages
- tag 9765=0; original messages

Retransmit Reject

When Retransmit Request message from customer is rejected by CME then CME will return a Retransmit Reject.

Retransmit Request

This diagram shows an example of Retransmission.



NotApplied

Since the message flow from customer to exchange is idempotent, when the exchange recognizes a sequence number gap, it will send a Not Applied message immediately and will not accept any subsequent message having a higher than expected sequence number *unless* the customer sends a Sequence message, which will instruct the exchange to ignore the gap and start with NextSeqNo as the valid sequence number.

The customer on an idempotent flow uses the Not Applied message to discover which of its requests have not been fulfilled and must decide whether to resend the transactions with new sequence numbers, send different transactions, or send a Sequence message to gap fill.

Since the message flow from exchange to customer is recoverable, this allows the Not Applied message to be re-synchronized if necessary. Thus, the customer can determine with certainty which requests have been accepted.

After receiving Not Applied from CME, the customer can choose to replay business messages with those missing sequence numbers or send a Sequence message with updated next expected sequence number.

Message Sequence Numbers

The FIXP protocol recommends maintaining separate sequence numbers for incoming and outgoing messages between the customer and the exchange. This ensures that all messages to and from the exchange are in the correct order however in iLink 3 only the business messages sent from the exchange are recoverable by the customer.

To guarantee message delivery, both the customer and exchange must maintain inbound and outbound sequence numbers. The customer's responsibility for maintaining inbound and outbound sequence numbers includes:

- Reset the inbound and outbound sequence numbers to “1” prior to Negotiating at the start of each week and subsequently at the start of a FIXP session with a new UUID
- Increment inbound sequence number by one for each message received from the exchange for a particular UUID
- Increment outbound sequence number by one for each message sent to the exchange for a particular UUID
- Issue a RetransmitRequest when a sequence gap is detected from the exchange for a particular UUID
    - Retransmit responses will be interleaved with other real-time original message transmissions if any
- If the exchange detects a sequence gap from customer, then a Not Applied message will be sent and the customer can take action as needed
    - This can include sending a Sequence message instructing the exchange to ignore the gap and proceed ahead with the NextSeqNo (recommended behavior)
    - This can include trying to replay the messages missed by the exchange (although this is not recommended behavior)
- Maintain sequence numbers between multiple FIXP sessions (a FIXP session occurs each time a customer Establishes session with CME iLink 3) on the basis of assigning a unique UUID for each connection
- If customer chooses to use fault tolerance then maintain sequence numbers for a particular UUID among the primary and backup processes during fail-over scenario
- The Sequence, Establish, Establishment Acknowledgment and Retransmission messages are sequence forming since they contain the NextSeqNo which indicates the sequence number of subsequent business messages
    - FIXP messages themselves are not assigned a sequence number; they only carry a reference as to the next expected sequence number of the next business message to be sent
    - All business messages will explicitly be assigned a sequence number

- Since there is no heart beat message, the Sequence message itself serves as the de-facto heart beat and at a minimum it must be sent after the lapse of each KeepAliveInterval if no other messages are being sent
- Since there is no test request message, the Sequence message also acts as a pinging message with KeepAliveIntervalLapsed=1 to inform the other side that no message has been received from it

CME Globex preserves all outbound sequence numbers for recovering missed business messages sent from the exchange to the customer during a failover scenario. Business messages sent from the customer to the exchange are not recoverable by the exchange.

If the customer has designed their local architecture to employ the use of a single server process that maintains multiple FIXP sessions to CME and supports multiple applications, the software should be designed to allow a session to reset and/or recover sequence numbers without affecting other actively trading connections on the server or the server process itself.

Sequence Number Higher than Expected

If the Sequence Number on a business message sent to the exchange is higher than expected then a Not Applied message will be sent to the customer to let them know that the exchange has seen a gap in sequence numbers and that those sequence numbers have not been applied by the exchange. The customer has the flexibility to take remedial action if required and resend the missing business messages or send a Sequence message indicating that the gap should be ignored. If neither of these actions are performed then a subsequent Not Applied message will continue to be sent by the exchange if still more business messages are received with higher than expected sequence numbers.

If the Sequence Number on a business message received from the exchange is higher than expected then a Retransmission Request message should be sent by the customer to recover the missing messages. This can be done for the current UUID as well as for previously used UUIDs.

Retransmit Requests will be limited to requesting 2500 messages at a time. Retransmission responses will be interleaved with other original real-time messages and these should be queued up for later processing since these should not trigger additional Retransmission Requests.

While Retransmission is in progress then the customer should accept real-time messages from the Market Segment Gateway with a higher sequence number after recognizing a gap. However, the customer should queue messages for in-sequence processing after Retransmission has concluded.

Sequence Number Lower than Expected

If the incoming Sequence message has a sequence number that is lower than expected then it indicates a serious error. In this case, a Terminate message is sent and the FIXP connection will be closed. The Code in the Terminate message will contain the code describing the reason why the Terminate message is being sent.

If the incoming Establish message has a sequence number that is lower than expected with the same UUID that was in use previously then it indicates a serious error. In this case, a Terminate message is sent and the FIXP connection will be closed. The Code in the Terminate message will contain the code describing the reason why the Terminate message is being sent.

If the incoming Establish message has a sequence number that is lower than expected with a new UUID then this will be permissible since the use of a new UUID indicates that the sequence numbers being sent to the exchange and received from the exchange should be reset back to 1. This will be an acceptable scenario for an intra-day Negotiate and Establish when the new UUID is different from the one used to Negotiate and Establish the previous FIXP session. Using a new UUID is also possible when Negotiating a new FIXP session at the start of each trading day.

Certain recovery situations will cause the exchange and customer sequence numbers to be out of sync. If a customer was terminated due to an invalid sequence number, then the recourse available to them is to Negotiate and Establish with a new UUID.

Previous UUID and Previous Sequence Number

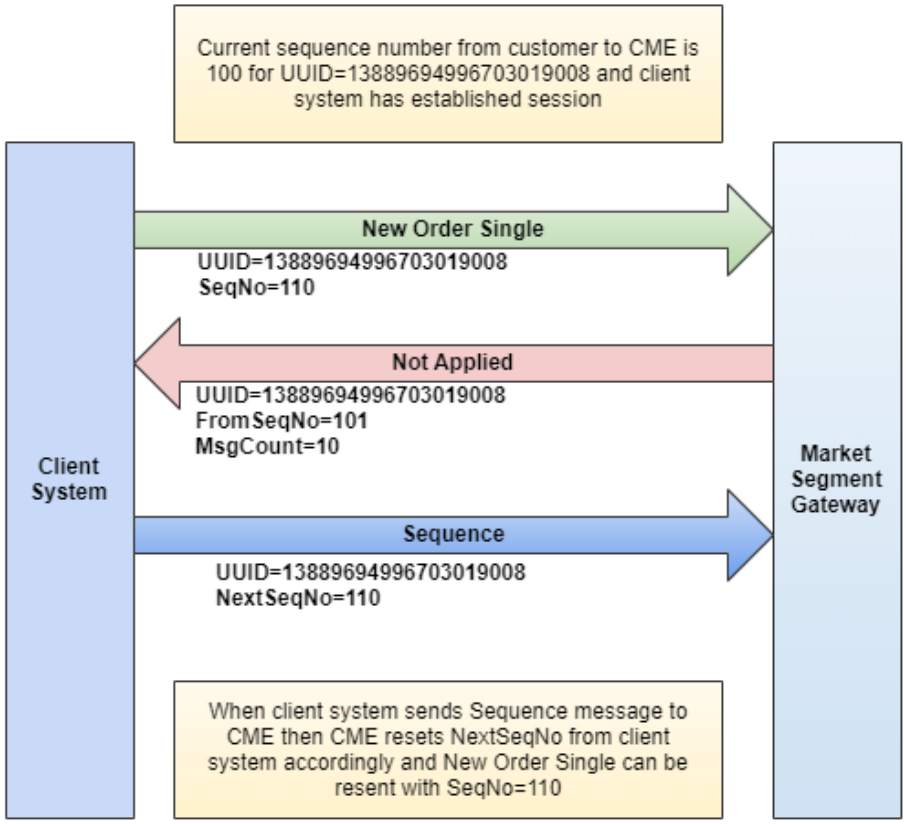
The Establishment Acknowledgment message sent to the customer from the exchange will also contain these two pieces of information – PreviousUUID and PreviousSeqNo. These can be leveraged by the customer to determine if they missed a message published by the exchange for them when they were not connected or if they simply missed a message due to other unforeseen circumstances.

If the customer detects a gap between the current UUID and current sequence number versus previous UUID and previous sequence number they can promptly issue a Retransmission Request to recover the same.

UUID=0 is reserved for CME usage as the default UUID to be assigned to the customer who have not yet connected for the first time.

Sequence Gap Fill

This diagram shows an example of Sequence Gap Fill.



Unbinding

Termination

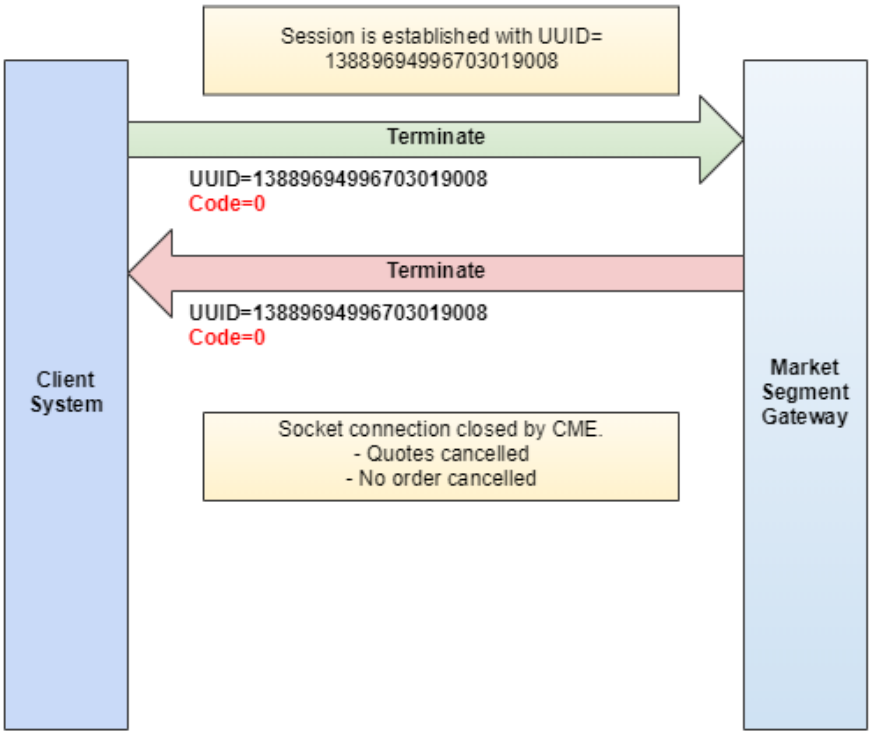
Terminate message is used to signal that the sender is going to disconnect the TCP socket connection. This unbinds the transport from the session, but it does not end a logical session. The Terminate message initiates and confirms the termination of a FIXP session.

- Terminate message can be sent by Customer to conclude the session
- Terminate message can be sent by CME upon FIXP protocol violation or other error condition such as volume controls and auto port closure
- Terminate message should be sent by receiver in response to Terminate message sent by requester (to conclude session) and then receiver can proceed to disconnect TCP socket connection
    - Requester should take down the socket only after receiving Terminate response; if no response is received after one KeepAliveInterval has passed, then proceed to disconnect assuming the socket connection has gone stale
- Terminate message sent by sender for an error condition does not require corresponding response from the receiver
    - Sender can proceed to disconnect the socket connection without waiting for receiver to first take down the socket

When the client system sends a Terminate message, the Market Segment Gateway sends a corresponding Terminate confirmation message and terminates the FIXP session. After sending a Terminate message, the customer should not send any additional messages, since this will be considered to be a protocol violation.

Terminate - Client Initiated

This diagram shows an example of client-initiated Terminate.



After sending a Terminate request, the client application should wait for a Terminate confirmation message before closing the socket connection. When a customer's FIXP session is terminated gracefully then the status of pending orders remains unaffected (for one way termination due to an error working orders may be cancelled through Cancel-on-Disconnect functionality). All active orders will continue to be eligible for execution after Termination. Any unsent Execution Report responses are sent via customer-initiated Retransmit Request after they have Established a FIXP session. If the FIXP session is Re-Negotiated intra-day with a different UUID then the unsent Execution Reports will be recoverable only by referencing the previous UUID and previous sequence number.

For the customer who chooses to use fault tolerance, the following must be noted: if the primary client application is terminated, then the backup application will also be terminated.

Graceful Termination from the exchange will take place at the end of the week when the exchange tells the customer that it is done for the day at the conclusion of the trading week and has no more messages to send.

Ungraceful Termination takes place for the following reasons:

- One of the peers receives a Terminate message due to protocol violation, a specific example of protocol violation is to send Establish with a new UUID without Negotiating
- Hard disconnect of the TCP socket connection
- Two times the KeepAliveInterval has expired and no Sequence message is received

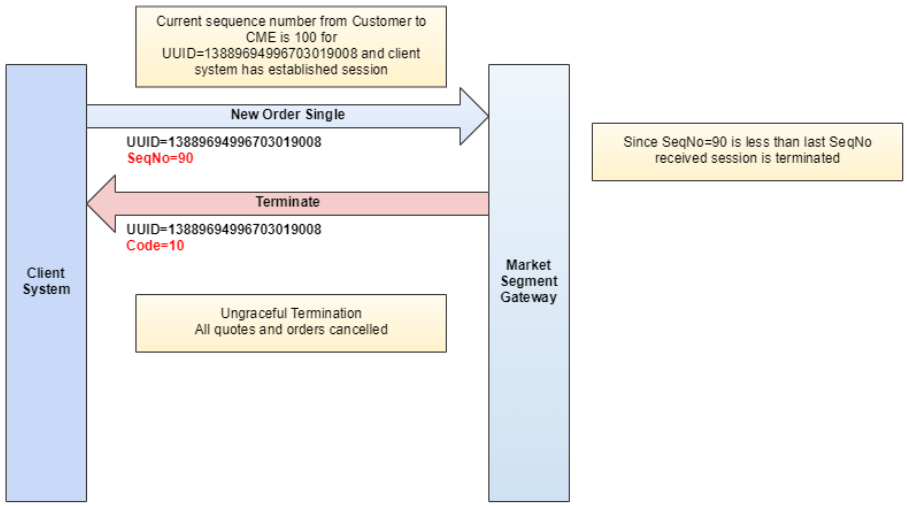
The Market Segment Gateway disregards any message that cannot be decoded or fails a data integrity check and does not increment the inbound implied sequence number. Processing the next business message will cause detection of a sequence gap and a NotApplied message will be generated. If the previously disregarded message is valid when resent, normal processing will continue; if the previously disregarded message remains in error, then NotApplied messages will loop until the next Sequence message following the bad message. The customer must detect this condition and perform the corrective processing to avoid it.

The following integer fields will be checked for these boundary conditions:

- UUID
- RequestTimestamp
- Code

Terminate - CME Initiated

This diagram shows an example of CME-initiated Terminate.



Fault Tolerance

A customer choosing to use fault tolerance must coordinate their application processes to establish separate and independent content streams to the Market Segment Gateways via TCP/IP socket connections. This group of redundant client processes (also commonly referred to as the fault tolerant group) operates together to provide fault tolerant functionality. In a typical deployment scenario, multiple redundant processes are spawned from the same executable file and each of those processes runs on a separate machine. Running redundant processes in the same machine is not recommended. If a machine fails, all the processes running on it fail simultaneously.

The client application must Negotiate and Establish with the designated primary Market Segment Gateway and then Establish with the backup Market Segment Gateway. A backup member must be ready to activate in the same data state as the former primary member being replaced. For example, inbound and outbound sequence numbers for a UUID must be maintained in a consistent state during fail-over between both processes.

Note: for active-active fault tolerance, Negotiation is not allowed on the backup connection.

The customer must use the same UUID for both primary and backup FIXP sessions (this is the only exception to the general rule of thumb that the UUID be globally unique) and then the sequence streams would be contiguous after failure.

It is important to note that the Fault Tolerance Indicator is present only in messages related to Negotiation, Establishment, and Sequence (not present in business messages).

The customer will be told beforehand of their primary and backup IP addresses and once Established, The customer will notice that the Establishment Acknowledgment and Sequence messages sent from the primary Market Segment Gateway will contain a valid value in NextSeqNo, whereas the Establishment Acknowledgment and Sequence messages sent from the backup Market Segment Gateway will show NextSeqNo as zero.

The customer must avoid populating NextSeqNo with a non-zero number in Establish and Sequence messages to the backup Market Segment Gateway, as this will be considered a protocol violation. The NextSeqNo in these messages must be set to zero. Likewise the reverse situation is also true i.e. setting NextSeqNo as zero in Establish and Sequence messages to the primary Market Segment Gateway will be considered a protocol violation.

Business messages (e.g. New Order - Single, Order Cancel/Replace Request) must be sent only through the primary content stream where sequencing is enforced per FIXP protocol. Communication over the backup content stream is solely for link maintenance; only administrative messages (Negotiate, Establish, Sequence, Terminate) are sent through the backup content stream.

In the event of a failure in an active-active fault tolerance scenario, the customer will receive a Sequence message from the backup or newly promoted primary Market Segment Gateway with the NextSeqNo containing a valid value (from where the old primary left off) along with the FaultToleranceIndicator=1 (primary) and this will serve as the trigger to let the customer know that a failure occurred and the transition to backup taking over as new primary was successful.

Customers Choosing Not to Implement Fault Tolerance

The exchange does not mandate that the customer use the fault tolerance feature, however CME strongly recommends using this functionality. A customer that does not implement fault tolerance will not be able to dynamically recover from process or network failures.

For customers that do not wish to implement fault tolerance, their application should just Negotiate and Establish with the designated primary Market Segment Gateway and not Establish with the backup Market Segment Gateway. However they are not precluded from doing so should they choose to Establish with the backup Market Segment Gateway at a later time after first Negotiating and Establishing with the primary gateway

Failover Processing

Mission-critical client applications must continue to function properly despite sudden difficulties such as process termination, hardware failure and network disconnects. Fault tolerance in a network environment is characterized by rapid recovery from such failures. One method of providing fault-tolerance is through a mechanism called fail-over; the end goal is to minimize service interruption caused by error conditions listed above.

iLink 3 detects four different categories of error conditions:

- Customer primary process failure
- Customer backup process failure
- Primary Market Segment Gateway failure
- Backup Market Segment Gateway failure

Process failure may be caused by hardware failure of the machine on which the process runs. It may also be caused by software error; faulty logic, improper memory handling, etc. Network failure includes loss of TCP/IP connectivity between the customer application and the Market Segment Gateway. This may be caused by faulty network interface card, frayed wire, power failure on network components (i.e. routers), etc.

The Market Segment Gateway initiates a controlled fail-over when it detects either process or network failure that impacts its ability to service the customer.

Customer Primary Process Failure

If the exchange does not receive a message from the primary client process within a defined interval (two times the KeepAliveInterval sent by client in the Establish message), then the exchange designates the primary client process as failed and initiates fail-over. The primary client application and backup client applications will be terminated and disconnected from the exchange and the backup client application will be expected to communicate with the designated primary Market Segment Gateway. The backup client application is notified of fault tolerance status change by the Code in the Terminate message (i.e. ErrorCodes = 26 - DisconnectFromPrimary: Backup session will be terminated as well).

- If the primary client process fails without closing the TCP connection, then it takes two times the KeepAliveInterval for the exchange to detect the primary process failure. If the customer would like to avoid the time delay in this process, then they should ensure the TCP connection is closed whenever their application fails.
- Before the fail-over, the backup client application was receiving NextSeqNo set to zero. During and after the fail-over process, the backup client application is responsible for ensuring that its UUID, inbound sequence number to the exchange and outbound sequence number from the exchange are synchronized with the primary application that just failed. This is critical since the newly elected primary member must know exactly where the failed member left off.
- If the NextSeqNo number of the Sequence message sent by the new primary client application is lower than that of the original client application for the same UUID, the exchange will terminate the client application as per FIXP protocol.
- If the NextSeqNo number of the Sequence message sent by the new primary client application is lower than that of the original client application for a different UUID, the exchange will accept that and will also inform the customer of the perceived gap by sending a NotApplied message; if the NextSeqNo=1 in this case for a new UUID then that is acceptable.

Customer Backup Process Failure

If the exchange does not receive any message from a backup client application within a defined interval (which is two times the KeepAliveInterval in the Establish message sent by the customer), then the backup client application will be disconnected and the status of the primary client application connectivity remain intact.

It is permissible for the backup client application to Negotiate and Establish with the backup Market Segment Gateway with a different UUID than that the primary client application is using, with the primary Market Segment Gateway, but the customer should be aware that the sequence stream will not be contiguous in event of a failure—therefore, it is recommended that both primary and backup client applications should use the same UUID.

Primary Market Segment Gateway Failure

If the primary Market Segment Gateway Gateway fails, then the Exhange fault tolerance mechanism initiates fail-over by electing the backup Market Segment Gateway to assume the primary role. The client application connected to this newly chosen Market Segment Gateway must act as the primary for the client application FT Group. The client application is notified to become primary by examining the value of NextSeqNo in the Sequence message, which will change from zero to a non-zero value and the FaultToleranceIndicator=1 (primary).

- Customer can re-establish a previous FIXP session (same UUID) after reconnecting to this new primary Market Segment Gateway without further negotiation, which ensures that the sequence numbers will continue
- Customer may also negotiate and establish a new session with the a new UUID after reconnecting and this resets the sequence numbers back to 1 both ways (customer to CME and CME to customer)

Backup Market Segment Gateway Failure

If a backup Market Segment Gateway fails, it will be restored to working status by market operations. For active-active fault tolerance, the backup client application must again re-establish the same session (same UUID) connection to this restored backup Market Segment Gateway to maintain a backup. The status of the primary client application connectivity remains intact.

Order Entry Service Gateway

For the pre-registered administrative information, the Party Details Definition and Party Details List functionality will be handled by the Order Entry Service Gateway (OESGW).

In such cases, a separate FIXP session is negotiated and established with the OESGW and can be left open and heart beating until a Party Details Definition Request or Party Details List Request needs to be made and the corresponding Party Details Definition Request Acknowledgment and Party Details List Report will be sent back from the OESGW confirming the request or a Business Reject will indicate that the request can not be honored.

The sequence stream from exchange to the customer of the FIXP session on the OESGW itself is also recoverable.

FIXP session on the OESGW needs to use its own UUID and sequence numbers. Also security validations are in place to ensure that only the entity (firm) of the FIXP session is able to create and recover Party Details Definitions belonging to it through the OESGW.