THOMSON REUTERS

A NOTE ON TAG 789 - NEXTEXPECTEDMSGSEQNUM

Version 4.4 of the FIX protocol introduced tag 789/NextExpectedSeqNum in the FIX Logon message as a discretionary tag. The functionality associated with this tag is described in the FIX Transport document v1.1, but this description has proved somewhat difficult to interpret.

The purpose of this document is to make a clear unambiguous statement of the functionality related to tag 789/NextExpectedSeqNum which we believe is in alignment with the FIX Protocol, Ltd. definition. It is hoped this will be of value to stakeholders involved in FIX systems where Tag 789 is used.

When is a FIX Session Established?

A FIX session is *Fully Established* only after the following three stages are complete:

1. Establishing a connection: In many instances this involves creating a TCP socket.
2. Authentication: The exchange and validation of FIX Logon messages.
3. Synchronisation: The retransmission of messages as necessary by one or both parties.

When establishing a FIX session, a client should not send any application message before:

- They have received all messages from the server in sequence number order, up to and including the FIX Logon message that started the logon sequence. This may include retransmitted or *gap filled* messages.
- The have retransmitted or *gap filled* all messages implicitly re-requested by the server using Tag 789.

Retransmitted Messages and Gap Filling

FIX messages can be considered as belonging to one of two categories:

- FIX Session Level messages
- Application Level messages

The FIX Session Level messages are:

- Heartbeat
- Test Request
- Resend Request
- Reject
- Sequence Reset
- Logout
- Logon

An application level message is any message that is not a FIX session level message.

Document Version: 1
Document Reference: <Custom property: FBID>
Date of issue: 9 December 2010

THOMSON REUTERS

Use of Tag 789/NextExpectedMsgSeqNum in a FIX Logon makes the use of the Resend Request message unnecessary when a FIX session is established.

After a FIX session has been established, the recipient of a message stream (either a client or server) can explicitly request retransmission of lost messages using a Resend Request message.

However a client requests the retransmission of messages, the response is essentially the same.

Retransmission of Application Level Messages

Application level messages are retransmitted containing tag 43/PossDupFlag = Y. This indicates to the receiver that the message has been retransmitted.

Retransmission of FIX Session Level Messages

FIX Reject messages are retransmitted in the same way as application level messages, including Tag 43/PossDupFlag = Y in the message to indicate retransmission.

All other kinds of FIX session message are *Gap Filled.*

Gap Filling is done in two ways:

- A Sequence Reset message may be sent to take the place of a single FIX session level message when it is Gap Filled.
  When the Sequence Reset message is used in this way tag 36/NewSeqNum is not sent.
  Quoting FIX Transport v1.1, tag 34/MsgSeqNum should “conform to standard message sequencing rules”. The value of tag 34 will match that of the replaced message.
- A Sequence Reset message may be sent to take the place of more than one FIX session level message when Gap Filling.
  When the Sequence Reset message is used in this way tag 36/NewSeqNum is sent.
  The presence of tag 36 indicates that the recipient should allow a gap in sequence numbers starting from the value in tag 34/MsgSeqNum and ending with the value in tag 36/NewSeqNum.
  The sequence number of the next message the recipient receives should match the value sent in tag 36.

Whenever a FIX SequenceReset message is used to gap fill in either of the ways described above tag 123/GapFillFlag = Y.

Application level messages may be gap filled using a sequence reset message along with, and in the same way as, FIX session level messages.

Sequence Reset

The FIX Sequence Rest message can be used to reset the sequence number on a FIX message stream to some arbitrary higher number.

When a FIX engine needs to reset the sequence number on a message stream to a higher number because, for example there are no persisted messages to retransmit, a FIX Sequence Reset message should be sent containing tag 36/NewSeqNum and tag 123/GapFillFlag = Y, as if gap filling several messages.

A FIX engine may send a FIX SequenceReset message containing tag 123 = N but, quoting the FIX Transport document v1.1, this should only be done in order “*to recover from a disaster situation which cannot be recovered via the use of Sequence Reset”.*

The recipient of a FIX Sequence Reset message containing tag 123 = N should ignore the value of tag 34/MsgSeqNum in the message and expect the next message received to have a sequence number equal to the value sent in tag 36/NewSeqNum.

A FIX Sequence Reset message should never contain tag 123 = N when sent in response to either a FIX Logon containing tag 789 or a FIX ResendRequest.

In the normal course of events it is not expected that the sequence reset message with 123=N would be used in this way by a client or a server application.

Conventions

Client and server behaviour in a set of relevant scenarios will be described below using sequence diagrams.

In each scenario the next expected in and outbound message sequence numbers both at the client and server will be described using the convention described below.

In each scenario these values will be described before the scenario runs and when it is complete. In most cases the numbers chosen at the start of each scenario are arbitrary, the exception being the *Sunny Day start of week: Scenario 1*.

NxtOut_C	= Next Expected Outbound Sequence Number at Client
NxtIn_C	= Next Expected Inbound Sequence Number at Client
NxtOut_S	= Next Expected Outbound Sequence Number at Server
NxtIn_S	= Next Expected Inbound Sequence Number at the Server

Scenario 1 : Start of Day – Sunny Day Logon

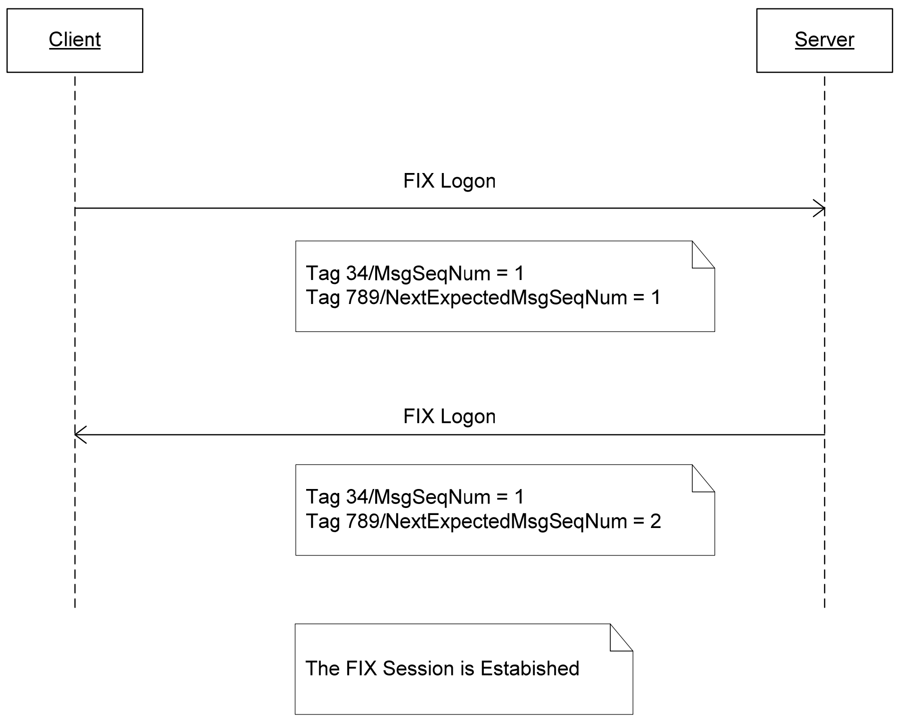
In this scenario the client and the server have both reset sequence numbers. This will typically happen at the start of each trading period. The client sends a valid FIX Logon.

Outcome: The FIX session is established.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 1
NxtIn_C	= 1
NxtOut_S	= 1
NxtIn_S	= 1

THOMSON REUTERS


Client
Server
FIX Logon
Tag 34/MsgSeqNum = 1
Tag 789/NextExpectedMsgSeqNum = 1
FIX Logon
Tag 34/MsgSeqNum = 1
Tag 789/NextExpectedMsgSeqNum = 2
The FIX Session is Estabished

When the scenario is complete the sequence number settings are:

NxtOut_C	= 2
NxtIn_C	= 2
NxtOut_S	= 2
NxtIn_S	= 2

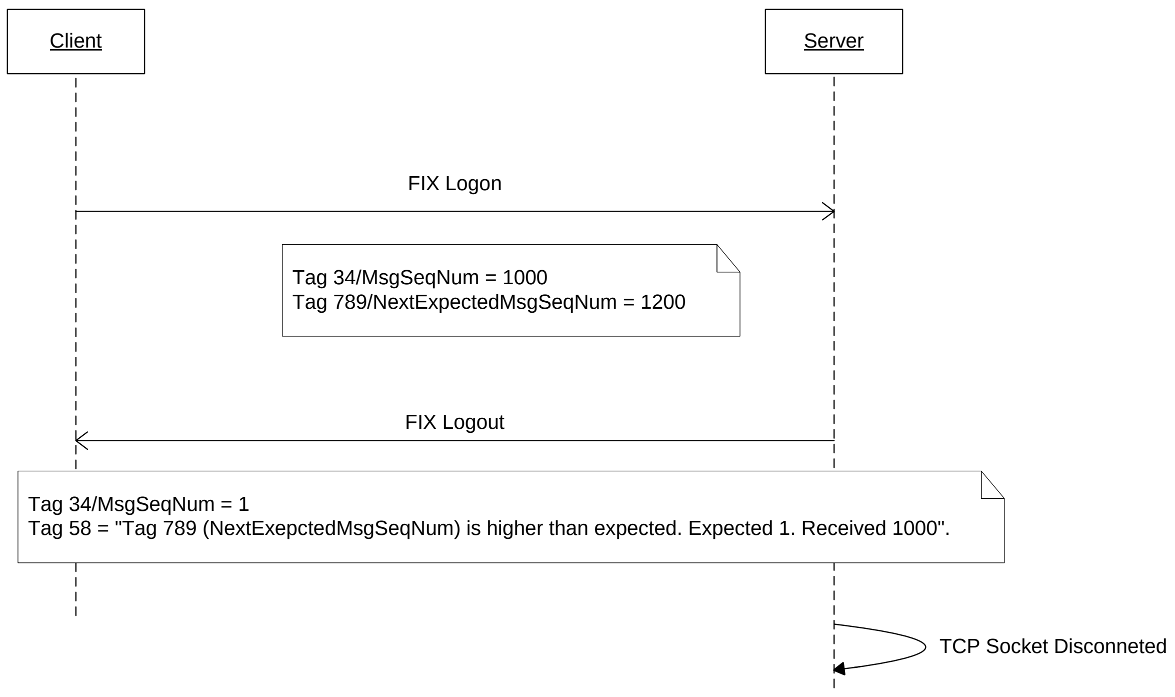
Scenario 2 : Client sends a FIX Logon with Tag 789 Too High

In this scenario the client sends a FIX Logon containing Tag 789 set to a higher value than NxtOut_S. This could conceivably happen at the start of a trading Day if the client does not reset their sequence numbers.

Outcome: The FIX session is not established, terminated by the server sending a FIX Logout and closing the TCP socket. The FIX session is not disabled at the server.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 1000
NxtIn_C	= 1200
NxtOut_S	= 1
NxtIn_S	= 1


Client
Server
FIX Logon
Tag 34/MsgSeqNum = 1000
Tag 789/NextExpectedMsgSeqNum = 1200
FIX Logout
Tag 34/MsgSeqNum = 1
Tag 58 = "Tag 789 (NextExepctedMsgSeqNum) is higher than expected. Expected 1. Received 1000".
TCP Socket Disconneted

When the scenario is complete the sequence number settings are:

NxtOut_C	= 1001
NxtIn_C	= 1200
NxtOut_S	= 2
NxtIn_S	= 1

Scenario 3 : Server sends a FIX Logon with Tag 789 Too High

This is not possible if the server is correctly implemented. For the server to send a FIX Logon with tag 789 too high, the sequence number in the clients FIX Logon would, by definition, have to have been too low. When the sequence number in the clients FIX Logon is too low, the server responds with a FIX Logout, as in scenario 2.

Although this scenario is unlikely, a client application should handle the scenario where it receives a FIX Logon from the server containing tag 789 set to a value higher than expected. The client application should respond by sending a FIX Logout containing Tag 58/Text = “Tag 789 (NextExpectedSeqNum) is higher than expected. Expected X. Received Y”, where X = NxtOut_C and Y = The value in Tag 789 in the incoming FIX Logon message.

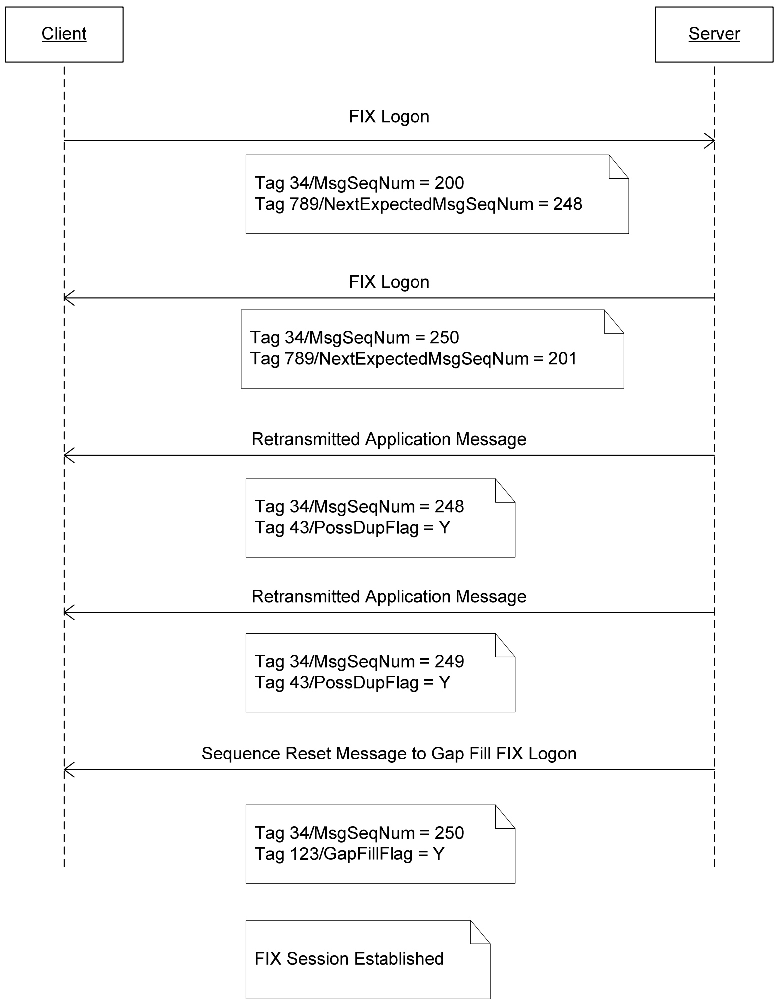
Scenario 4 : Client sends a FIX Logon with Tag 789 lower than expected

In this scenario the client sends a FIX Logon containing tag 789 set to a value that is less than NxtIn_S. This scenario might arise after an unexpected connection loss where messages were lost on the message stream from the server to the client.

Outcome: The FIX Session is established after retransmission of messages from the server.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 200
NxtIn_C	= 248
NxtOut_S	= 250
NxtIn_S	= 200



When the scenario is complete the sequence number settings are:

NxtOut_C	= 201
NxtIn_C	= 251
NxtOut_S	= 251
NxtIn_S	= 201

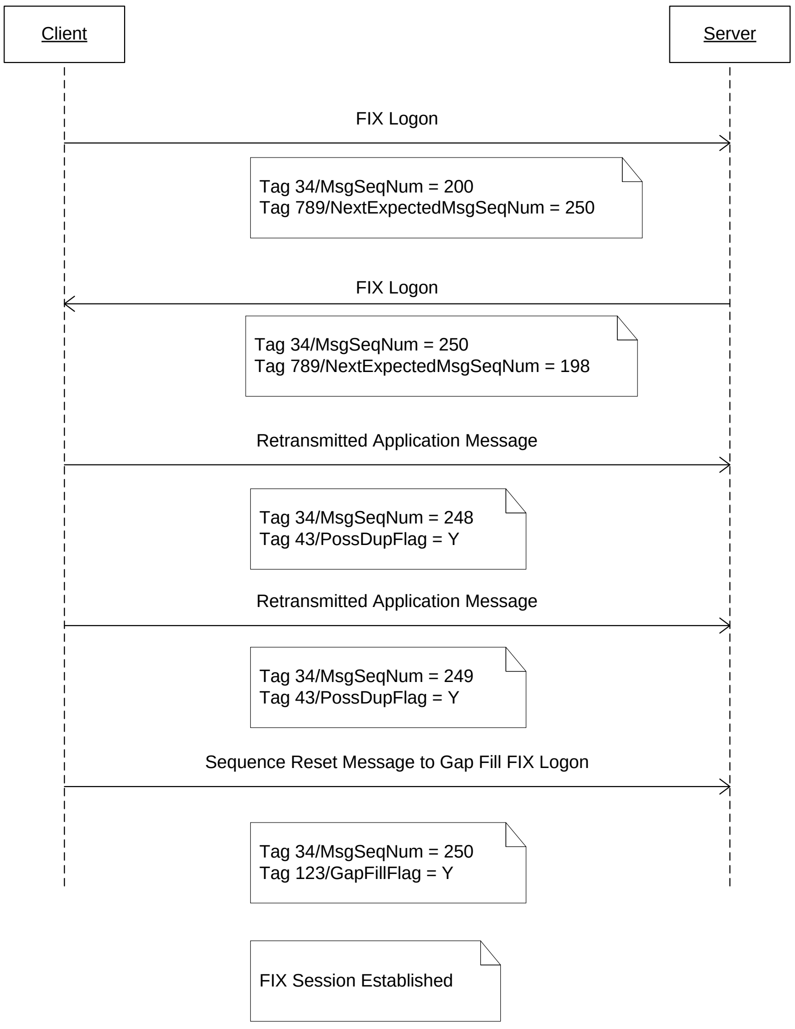
Scenario 5 : Server sends a FIX Logon with Tag 789 lower than expected

In this scenario the client sends a valid FIX Logon and the server responds with a FIX Logon containing tag 789 set to a value that is less than NxtIn_C. This scenario might arise after an unexpected connection loss where messages were lost on the message stream from the client to the server.

Outcome: The FIX Session is established after retransmission of messages from the Client.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 200
NxtIn_C	= 250
NxtOut_S	= 250
NxtIn_S	= 198



When the scenario is complete the sequence number settings are:

NxtOut_C	= 201

THOMSON REUTERS

NxtIn_C	= 251
NxtOut_S	= 251
NxtIn_S	= 201

Scenario 6 : Both client and server send a FIX Logon with Tag 789 lower than expected

In this scenario the FIX Logon from the client contains Tag 789 < NxtOut_S and the FIX Logon from the server contains Tag 789 < NxtOut_C. Messages have been lost on both message streams.

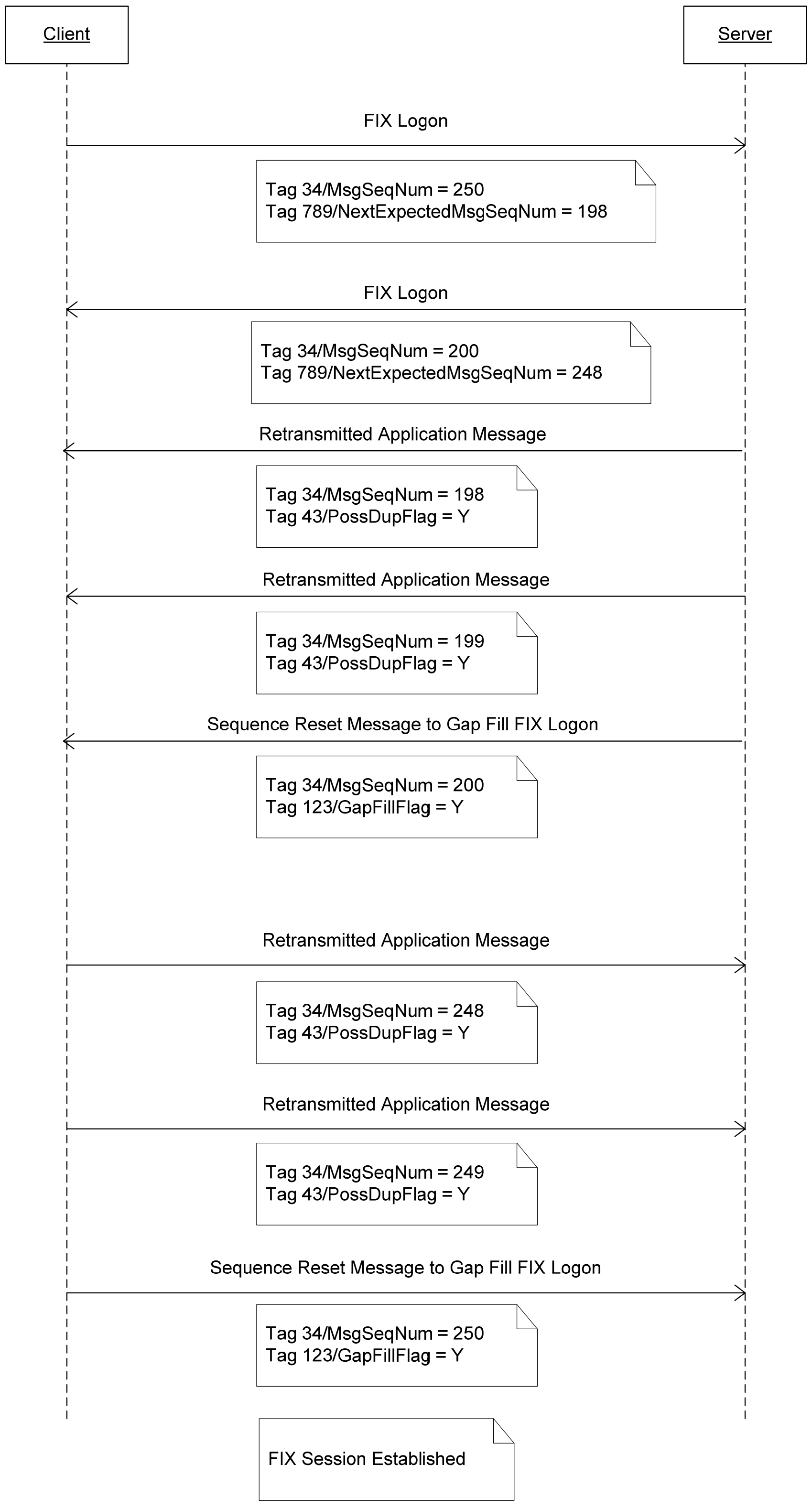
Outcome: The FIX Session is established after retransmission of messages from both the Client and the server.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 250
NxtIn_C	= 198
NxtOut_S	= 200
NxtIn_S	= 248

When the scenario is complete the sequence number settings are:

NxtOut_C	= 251
NxtIn_C	= 201
NxtOut_S	= 201
NxtIn_S	= 251



The retransmission of messages in each direction may occur simultaneously.

Scenario 7 : Client application needs to use a sequence reset to resynchronise

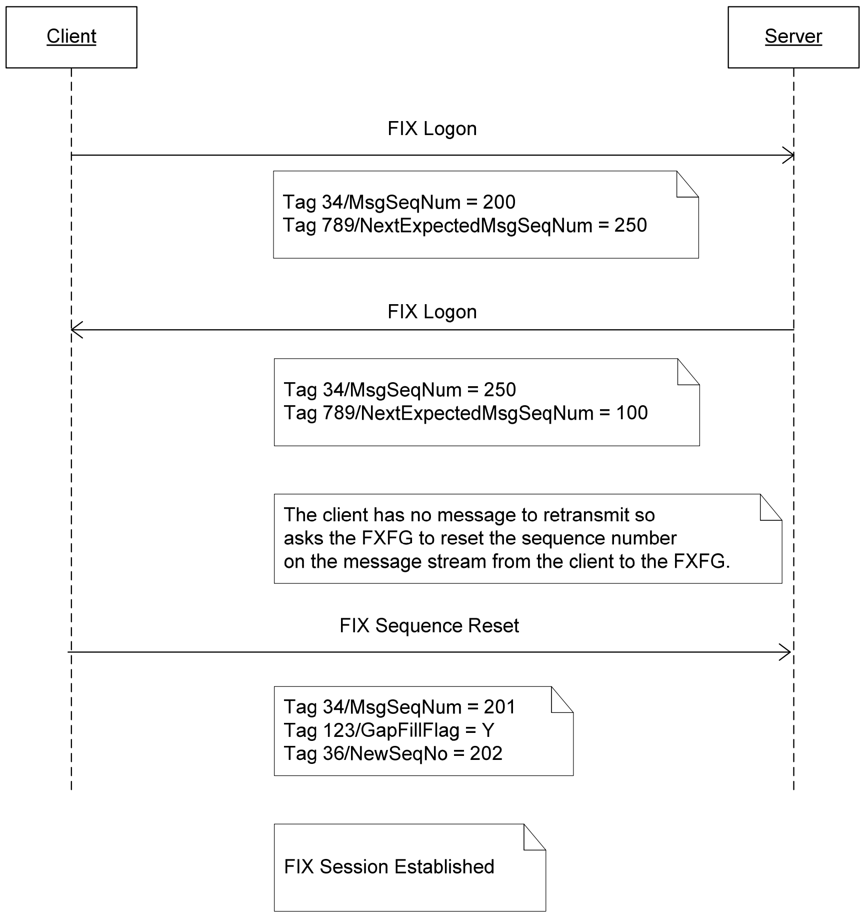
In this scenario NxtOut_C is greater than NxtIn_S and there are no persisted messages in that range for the client to retransmit. The client sends a sequence reset to reset the sequence number on its outbound message stream. This situation might arise if NxtOut_C was increased manually before the connection attempt.

Outcome: The FIX Session is established after a sequence reset from the client.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 200
NxtIn_C	= 250
NxtOut_S	= 250
NxtIn_S	= 100

If possible pls replace FXFG with “Server” in the diagram below


Client
Server
FIX Logon
Tag 34/MsgSeqNum = 200
Tag 789/NextExpectedMsgSeqNum = 250
FIX Logon
Tag 34/MsgSeqNum = 250
Tag 789/NextExpectedMsgSeqNum = 100
The client has no message to retransmit so asks the FXFG to reset the sequence number on the message stream from the client to the FXFG.
FIX Sequence Reset
Tag 34/MsgSeqNum = 201
Tag 123/GapFillFlag = Y
Tag 36/NewSeqNo = 202
FIX Session Established

When the scenario is complete the sequence number settings are:

NxtOut_C	= 202

NxtIn_C	= 251
NxtOut_S	= 251
NxtIn_S	= 202

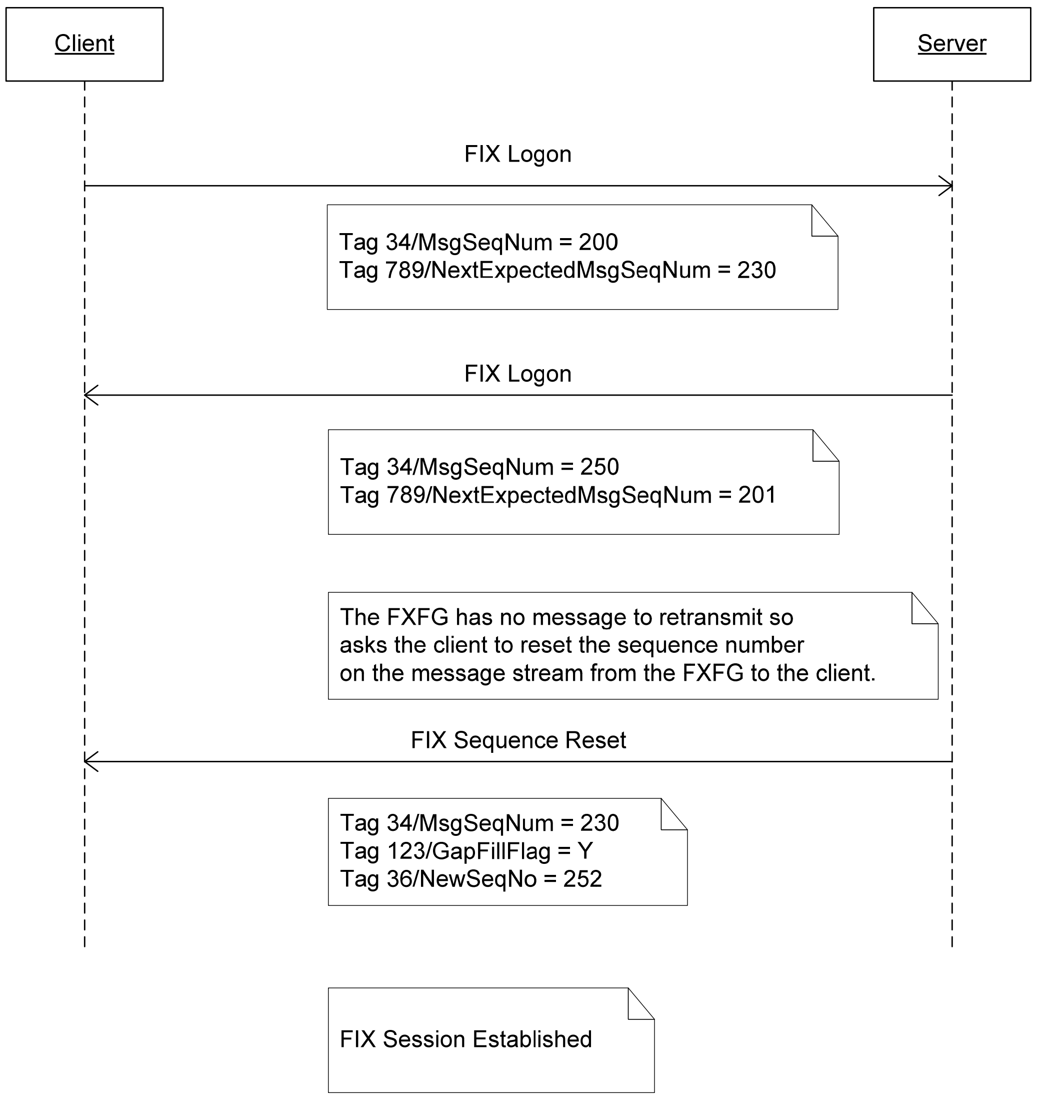
Scenario 8 : Server needs to use a sequence reset to resynchronise

In the scenario NxtOut_S is greater than NxtIn_C and there are no persisted messages in that range for the server to retransmit. The server sends a Sequence Reset message to reset the sequence number on its outbound message stream. This situation might arise if NxtOut_S was increased manually before the connection attempt.

Outcome: The FIX Session is established after a sequence reset from the server.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 200
NxtIn_C	= 230
NxtOut_S	= 250
NxtIn_S	= 200


Client
Server
FIX Logon
Tag 34/MsgSeqNum = 200
Tag 789/NextExpectedMsgSeqNum = 230
FIX Logon
Tag 34/MsgSeqNum = 250
Tag 789/NextExpectedMsgSeqNum = 201
The FXFG has no message to retransmit so asks the client to reset the sequence number on the message stream from the FXFG to the client.
FIX Sequence Reset
Tag 34/MsgSeqNum = 230
Tag 123/GapFillFlag = Y
Tag 36/NewSeqNo = 252
FIX Session Established

When the scenario is complete the sequence number settings are:

NxtOut_C	= 201
NxtIn_C	= 252
NxtOut_S	= 252
NxtIn_S	= 201

Scenario 9 : Client needs to resynchronise after sending many unsuccessful FIX Logons

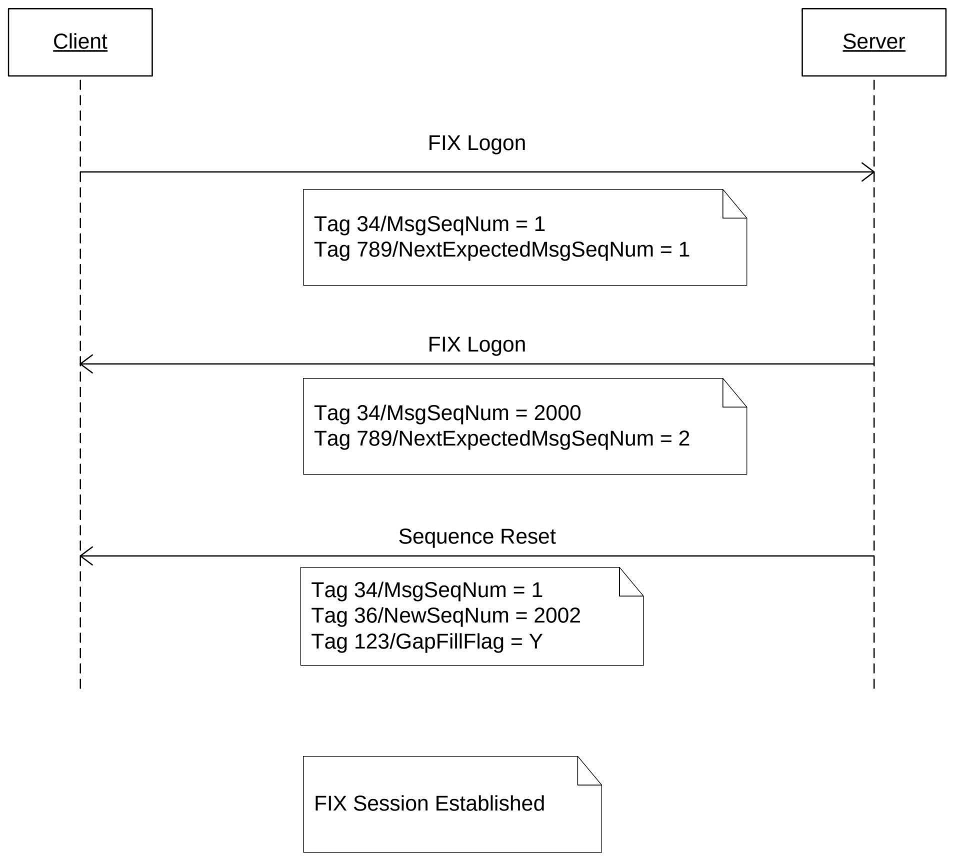
I think you have the sequence diagrams reversed… The diagram for #10 should be here and vise versa ?

In this scenario the client application has been repeatedly trying to login before the server is available. NxtOut_C is incremented on each login attempt. The sequence diagram below describes the client logon that succeeds when the service becomes available.

Outcome: The FIX Session is established after a sequence reset from the client.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 2000
NxtIn_C	= 1
NxtOut_S	= 1
NxtIn_S	= 1



When the scenario is complete the sequence number settings are:

NxtOut_C	= 2002
NxtIn_C	= 2
NxtOut_S	= 2
NxtIn_S	= 2002

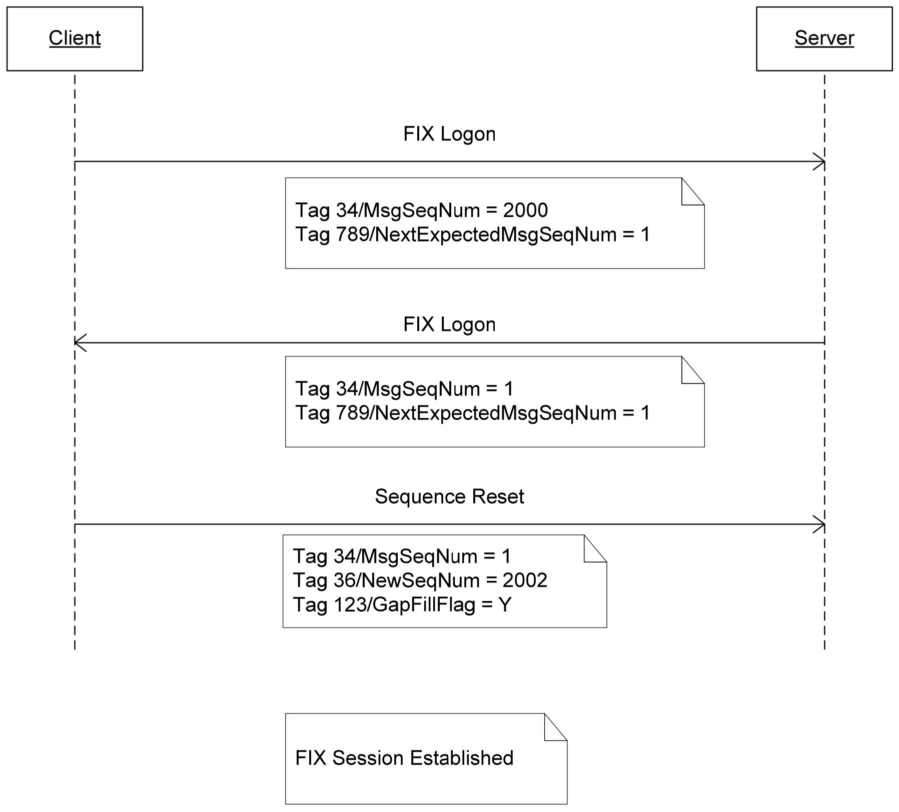
Scenario 10 – A complement to scenario 9

This is the complement of scenario 9. Not sure how this could happen but documenting the scenarios may be worthwhile.

Outcome: The FIX Session is established after a sequence reset from the server.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 1
NxtIn_C	= 1
NxtOut_S	= 2000
NxtIn_S	= 1



When the scenario is complete the sequence number settings are:

NxtOut_C	= 2
NxtIn_C	= 2002
NxtOut_S	= 2002
NxtIn_S	= 2

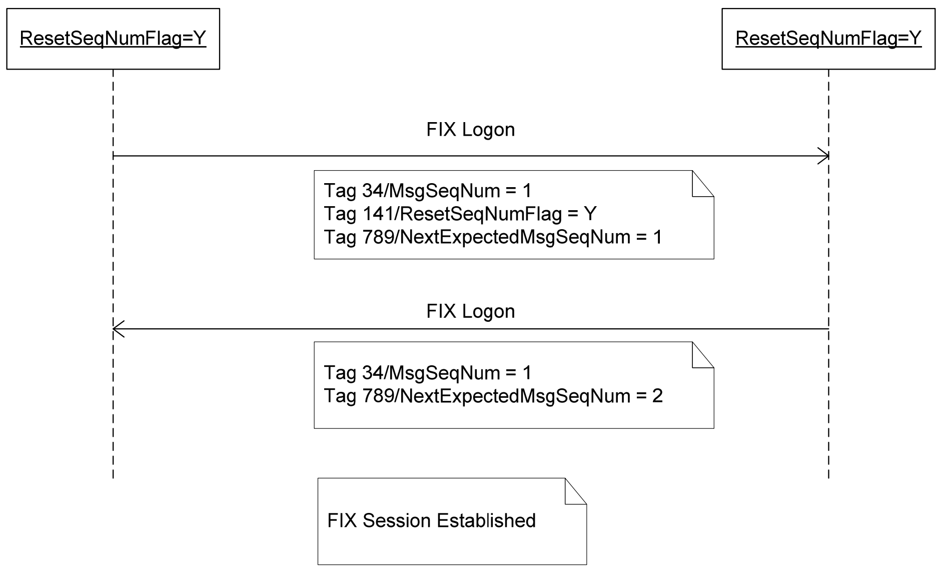
Scenario 11 – FIX Logon using tag 141/ResetSeqNumFlag and tag 789/NextExpectedSeqMsgNum

In this scenario the sequence numbers at the server on both in and out bound message streams are greater than one, the client is sending tag 141 = Y in their FIX Logon message in order to reset the sequence numbers on both message streams to 1. When Tag 141/ResetSeqnNumFlag is used tag 789/NextExpectedSeqMsgNum is effectively ignored.

Outcome: The FIX Session is established and the sequence numbers on both message streams are as they would have been if both client and server had reset sequence numbers on both message stream to 1 before the interaction started.

Before client sends FIX Logon sequence number settings are:

NxtOut_C	= 1
NxtIn_C	= 1
NxtOut_S	= 9999
NxtIn_S	= 9999



When the scenario is complete the sequence number settings are:

NxtOut_C	= 2
NxtIn_C	= 2
NxtOut_S	= 2
NxtIn_S	= 2