



Oslo Børs News Feed Technical specification

Document version 1.1 / 11-03-2010

Effective from 11-03-2010

(this page intentionally left blank for two-sided printing)

Table Of Contents

1. Introduction	7
1.1 Scope of this document	7
1.2 Definitions, abbreviations and acronyms	7
1.3 Contact information	7
1.4 Support, Calendar and core hours	7
1.5 Notification policy	8
2. Session control	9
2.1.1 Messages.....	9
2.1.2 Message scenario	12
2.1.3 FC state diagram	14
2.1.4 FC states and message processing	15
Common message processing for all states	15
2.1.5 State INIT	15
2.1.6 State AWAIT_CONNECT_ACK.....	16
2.1.7 State AWAIT_LOGON_ACK.....	16
2.1.8 State AWAIT_COMMAND_ACK	16
2.1.9 State AWAIT_CMD_END.....	16
2.1.10 State AWAIT_LOGOFF_ACK.....	17
2.2 Initiating data transfer commands	17
2.2.1 Batch sessions.....	17
2.2.2 Real-time sessions	17
2.3 FC requests – “commands”	17
2.4 FS responses – “transactions”	18
2.4.1 Transaction layout	18
3. Presentation and encoding of data.....	19
3.1 Transaction meta-format	19
3.2 Special characters.....	19
3.3 Transaction head	19
3.4 Transaction body.....	20
3.5 Unknown field tags	20
3.6 Null field values	20
3.7 Field value types	21
4. Transactions and fields	22
4.1 Real time transactions	22
4.1.1 NewsItem (n)	22
4.2 Basic data transactions (batch).....	22
4.2.1 Equity (Be)	22
4.2.2 Bond (Bb)	22
4.2.3 Issuers (Bc).....	23
4.2.4 Fields (Bf).....	23
4.2.5 Transactions (Bt)	23
4.3 Field descriptions	24
4.4 Constant values for specific fields	25
4.4.1 marketCode (Mc).....	25
4.4.2 newsSource (Ns).....	25
4.4.3 newsType (Nt)/ newsTypeEnglish (NTe).....	25
4.4.4 newsLanguage (NLa)	26

List Of Tables

Table 1 - Document history	5
Table 2 - Definitions, abbreviations and acronyms	7
Table 3 – Message descriptions.....	11
Table 4 - States	15
Table 5 – Message handling common in all states	15
Table 6 - Message handling in AWAIT_CONNECT_ACK state.....	16
Table 7 – Message handling in AWAIT_LOGON_ACK state	16
Table 8 – Message handling in AWAIT_CMD_START_ACK state	16
Table 9 – Message handling in AWAIT_CMD_END state	17
Table 10 – Message handling in AWAIT_LOGOFF_ACK state	17
Table 11 - News Item transaction.....	22
Table 12 - Equity transaction	22
Table 13 - Bond transaction	23
Table 14 - Issuers transaction.....	23
Table 15 - Fields transactions.....	23
Table 16 - Transactions transaction.....	23
Table 16 - Field descriptions	24
Table 17 - marketCode constant values.....	25
Table 18 - newsSource constant values	25
Table 19 - newsType/newsTypeEnglish constant values	26
Table 20 - newsLanguage constant values	26

List Of Figures

Figure 1 – Typical message exchange scenario feed client – feed server	12
Figure 2 – State diagram for client	14

Document history

Ver.	Date	Description
1.0	01.02.2009	Oslo Børs News Feed Release 1.0 First official version of documentation.
1.1	11.03.2010	Added the field compID to the Equities and Bonds transaction. Removed the orderBook Id from the News Item transaction. Added the Subscription Rights transaction

Table 1 - Document history

1. Introduction

The Oslo Børs News Feed is a service from Oslo Børs where news is distributed through the feed 365/24/7. The news feed is based on the OCDF protocol (Oslo Børs Continuous Data Feed).

1.1 Scope of this document

This is a document aimed at the client-side system developer to provide the necessary information for implementing a client system for receiving information regarding news in real-time from the company disclosures system at OB.

1.2 Definitions, abbreviations and acronyms

FC	Feed Client. Client side of the communication.
FS	Feed Server. OB side of the communication.
ISIN	International Security Identification Number, an internationally accepted ISO standard for identifying securities.
ABM	Alternative Bond Market
OAX	Oslo Axess
OB	Oslo Børs.

Table 2 - Definitions, abbreviations and acronyms

1.3 Contact information

Oslo Børs ASA

Address: P.O. Box 460, Sentrum
0105 OSLO
NORWAY

Telephone: (+47) 22 341 700 (switchboard)
(+47) 22 341 990 (OSE HelpDesk, operational and 1st line support)

Fax: (+47) 22 42 68 47

E-Mail: HelpDesk@oslobors.no (operations)
products@oslobors.no (data content support)

Web:

The latest released version of this document can be downloaded from Oslo Børs' website, along with file-based test data. Other information about Oslo Børs (OB) can be found at <http://www.oslobors.no>.

OB regulations / trading rules: <http://www.oslobors.no/ob/loverogregler?languageID=1>

1.4 Support, Calendar and core hours

The Oslo Børs News Feed will be running 365/24/7, except from a short maintenance window around midnight.

Support will only be available at the Børs' core hours. Core hours are specified in the document "Core Hours and Timeline" found on http://www.oslobors.no/ob/md_ocdf.

1.5 Notification policy

- Major changes – three months
- Minor changes – one month
- Bugfixes – as soon as possible if critical and coordinated with major/minor releases where appropriate.

2. Session control

Oslo Børs Continuous News Feed is distributed to clients over a TCP/IP connection, through Internet or fixed lines.

Session control is in this context the process of a client system (FC) establishing a connection to the server system (FS) and both parts maintaining this connection for delivering data to the FC application. This relates to the “session layer” of the OSI reference model.

2.1.1 Messages

All session control messages exchanged between FS and FC start with S_.

Internally generated «messages» within FS or FC are in this documentation indicated and given names starting with I_, but these are NOT exchanged between FS and FC.

The following table lists the session control messages passed between FC and FS:

Message	Parameters ¹	FS ² ⇒	FC ³ ⇐	Description
S_ACK	[; infoText]	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Positive acknowledge to a request, sent both ways.
S_ABORT	; status [; infoText]	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Abort message that unconditionally will cause the session to be stopped in any state, even while in command processing mode. This message can be sent both ways. Defined status codes are: -1 Undefined error -2 Operator initiated shutdown -3 Internal error -4 Time-out -5 Error count exceeded

¹ Optional parameters are listed in brackets, others are mandatory. N/A implies no parameters.

² Messages sent from FS to FC

³ Messages sent from FC to FS

Message	Parameters ¹	FS ² ⇒	FC ³ ⇐	Description
S_CMD_END_REQ	; status [; infoText]	<input checked="" type="checkbox"/>		Command end request message sent from FS to FC. Acknowledge is required from FC. Status is an integer code with processing status for the command: 0 OK -1 No data -2 Illegal (request)command -3 Illegal (restart)parameter -4 Illegal parameters -5 Not a tradeday -6 Configuration file can not be read -7 Other error Can also include an informational text message.
S_CMD_START_REQ	; command [; args]		<input checked="" type="checkbox"/>	Command start request message, sent from FC to FS. If the command can be started, a S_ACK is sent as reply to the message, otherwise a S_NAK is returned.
S_CONNECT_REQ	[; infoText]		<input checked="" type="checkbox"/>	Connection request from FC to FS.
S_ENQ_REQ	; seqNo [; infoText]	<input checked="" type="checkbox"/>		Enquiry request message or «aliveness poll», sent from FS to FC with regular time intervals. FC shall immediately respond with a S_ENQ_ACK message, returning the sequence number received in S_ENQ_REQ from FS.
S_ENQ_ACK	; seqNo [; infoText]		<input checked="" type="checkbox"/>	Enquiry acknowledge message sent from FC as response to a S_ENQ_REQ. The sequence number (integer between 1 and 99) in the S_ENQ_REQ message should be returned in the S_ENQ_ACK message.
S_LOGOFF_REQ	[; infoText]		<input checked="" type="checkbox"/>	Logoff message, sent from FC to FS. Shall in any case be responded to with a S_ACK from the FS.
S_LOGON_REQ	; userId ; password		<input checked="" type="checkbox"/>	Logon request message, sent from FC to FS. User ID and password is sent as plain text parameters, to be checked against FS system's password validation mechanisms. Response should be S_ACK if logon is accepted, S_NAK otherwise.

Message	Parameters ¹	FS ² ⇒	FC ³ ⇐	Description
S_NAK	; status [; infoText]	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Negative acknowledge to a request, sent both ways. Status codes that can be returned are: -1 Undefined error -2 Invalid logon (userID/password) -3 Invalid FC host address -4 Internal FS error, cannot execute command. -5 Unexpected message in current state
I_ERROR	N/A			Internal (FS) error «message»: Too many errors flagged during communication with FC. FS shall respond with S_ABORT. The error limit should be configurable, with a suggested default value of 3.
I_TIMEOUT	N/A			Internal (FS) time-out «message»: Too long time has passed without response from FC. FS shall respond with S_ABORT. The time-out value should be configurable in the range 1s to 300s, with a suggested default value of 30s. Different timeout values may be configured for different states.

Table 3 – Message descriptions

[; **infoText**] fields are optional text fields, and may contain useful text information for determining error cause in both FS and FC. This could e.g. be process identification, state information, and expected behaviour when error detected and so on.

Examples and actual text from FS presented here may be altered without further notice.

2.1.2 Message scenario

The following figure describes a normal session between FS and FC.

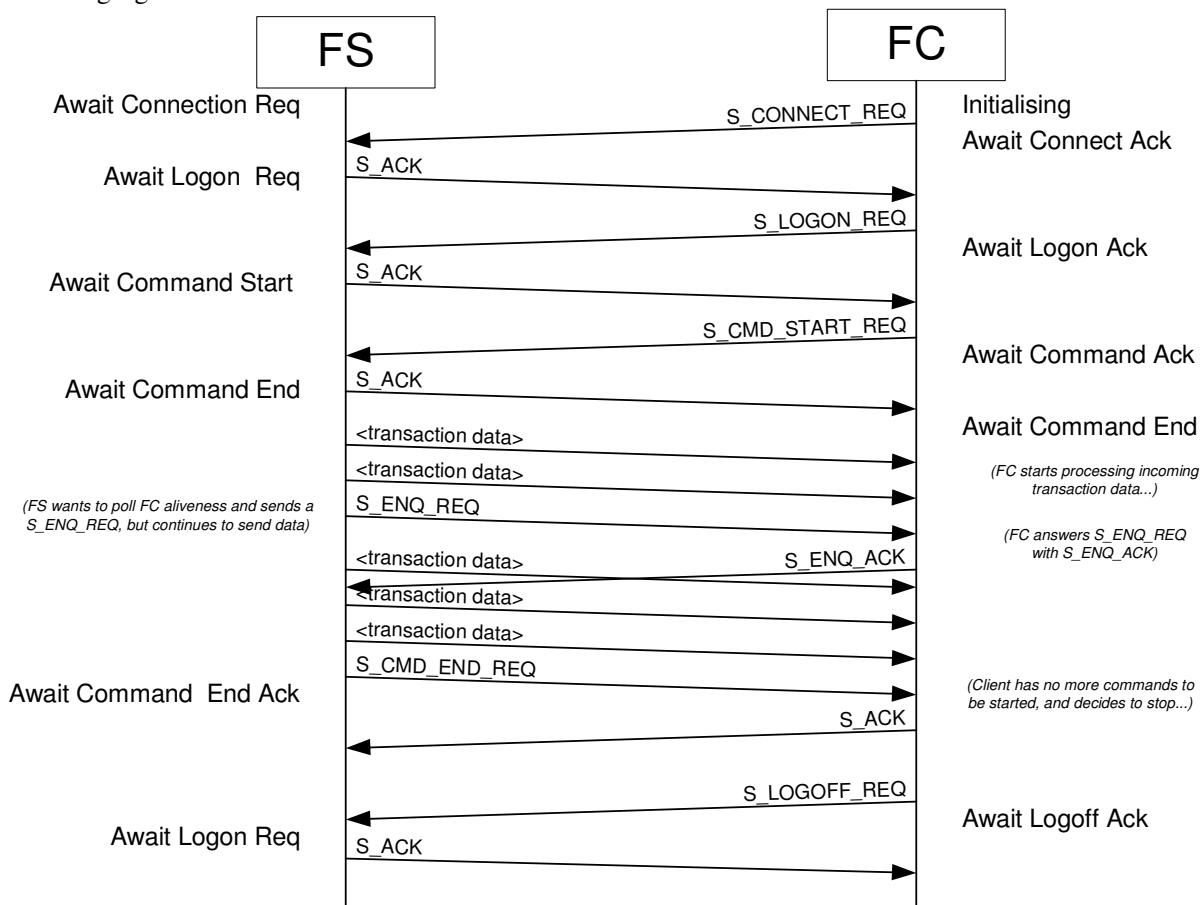


Figure 1 – Typical message exchange scenario feed client – feed server

A typical FC processing sequence will be like this:

1. Initial processing
2. Connect
3. Logon
4. Get transaction definitions using command TRANSACTIONS_DEF
5. Get field definitions using command FIELDS_DEF
6. Get instrument definitions using command FIXED_DATA
7. Get real-time data using command REALTIME from GSN 0
8. If REALTIME session interrupted, continue from last GSN received
9. Finish processing when S_CMD_END_REQ returns with OK status code

Below is a more detailed example of how the message exchange might occur:

Example:

From FC to FS (Out from FC)	
From FS to FC (In to FC)	
<i>(FC initiates a session by opening a socket connection to FS, which starts a FS process)</i>	
S_CONNECT_REQ	<i>FC initiates connection</i>
S_ACK;pid=29978 state=AWAIT_CONNECTI...	<i>FS acknowledges connection</i>
S_LOGON_REQ;FEEDUSER;secret	<i>FC logs on</i>
S_ACK; AWAIT_COMMAND_START	<i>FS acknowledges logon</i>
S_CMD_START_REQ;TRANSACTIONS_DEF	<i>FC asks for transaction definitions</i>
S_ACK;AWAIT_COMMAND_END	
Bt;;nNewsItem;Tagn;DsOfficial news;CMdREALTIME	<i>FS sends transaction</i>
Bt;;nBondIndex;TagBBI;DsBond index data;CMdFIXED_DATA	
...	
S_CMD_END_REQ;0;pid=29978 state=AWAIT_...	<i>FS indicates end of data</i>
S_ACK	<i>FC acknowledges end of data</i>
S_CMD_START_REQ;FIELDS_DEF	<i>FC asks for field definitions</i>
S_ACK	
Bf;;nearningsPrShare;TagEPs;FtFloat;Vf19980708;Vt20020502;DsEarnings per share. Given in the same currency as currencyQuotation	
...	
S_CMD_END_REQ;0;pid=29978 state=AWAIT_...	<i>FS indicates end of data</i>
S_ACK	<i>FC acknowledges end of data</i>
S_CMD_START_REQ;FIXED_DATA	<i>FC asks for instrument definitions</i>
S_ACK	
Bd;;iNO0007057396;Id19980619;Sc3;SnBEB fwd des 1998;sBEB8X;Tb;Cs100;Dt6;Ed19981217;IuNO0003102113	
...	
S_CMD_END_REQ;0;pid=29978 state=AWAIT_...	<i>FS indicates end of data</i>
S_ACK	<i>FC acknowledges end of data</i>
S_CMD_START_REQ;REALTIME;1	<i>FC asks for realtime data from seqNo 0</i>
S_ACK	
Sc;1;t083544;iNO120004522;1a125.75;1Va5000;1Ao2	
...	
S_ENQ_REQ;65	<i>FS checks that client is alive</i>
S_ENQ_ACK;65	<i>FC answers that it is alive and well</i>
Me;54322;t142231;Mc1;Ms1	
...	
S_CMD_END_REQ;0;pid=29978 state=AWAIT_...	<i>FS indicates end of data</i>
S_ACK	<i>FC acknowledges end of data</i>
S_LOGOFF_REQ	<i>FC is finished, no more commands</i>
	<i>FS closes down connection</i>
<i>(Finished real-time processing, FC closes socket, FS process terminates)</i>	

2.1.3 FC state diagram

The session control as seen from the FC side described as a state machine:

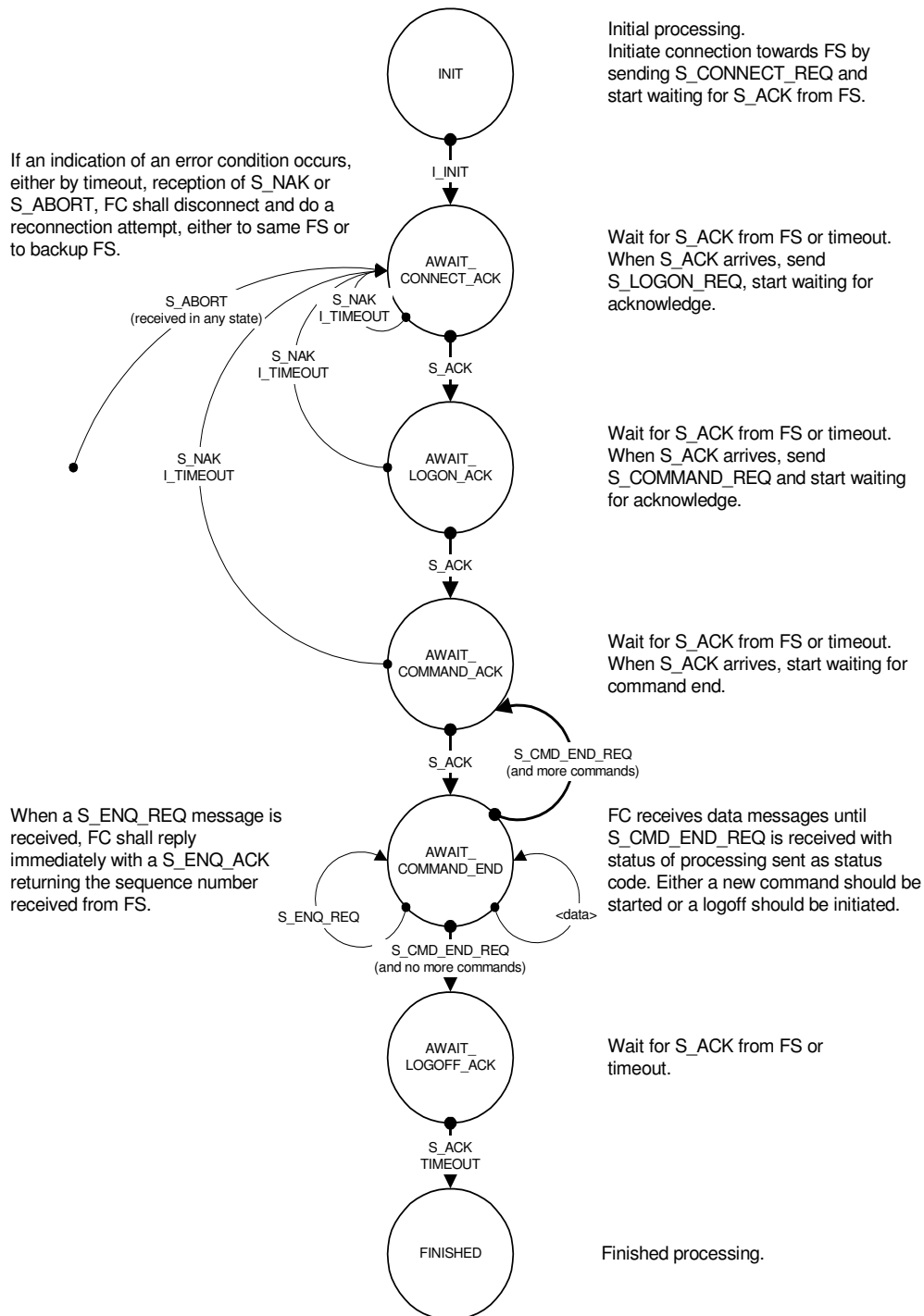


Figure 2 – State diagram for client

FC states are:

FC State	Description
INIT	Initial processing state. Initialise the FC system, attempt to open socket connection to FS and send S_CONNECT_REQ to FS.
AWAIT_CONNECT_ACK	Await connection acknowledge – S_ACK. Send S_LOGON_REQ with username and password when received.
AWAIT_LOGON_ACK	Await logon acknowledge – S_ACK. Send first command (of several scheduled) when received.
AWAIT_COMMAND_ACK	Await command acknowledge – S_ACK. Start waiting for transaction data when received.
AWAIT_COMMAND_END	Await command end request – S_CMD_END_REQ. Receive and process transactions until S_CMD_END_REQ is received. If S_ENQ_REQ received, these should be responded to with S_ENQ_ACK immediately.
AWAIT_LOGOFF_ACK	Await logoff acknowledge – S_ACK. When received, finish processing.
FINISHED	Finished processing state. Perform any postprocessing on data received.

Table 4 - States

2.1.4 FC states and message processing

Common message processing for all states

The following messages / events should be handled equally for all states. If special handling is required, this is described specially.

Message	FC action
S_ABORT	FS has for some reason (indicated by the status code) decided to abort the connection. FC shall respond by closing the socket connection to FS, close the socket connection, attempt opening again and change state to AWAIT_CONNECTION_ACK, waiting for a response from FS.
I_TIMEOUT	No valid response has been received from FS within a predefined amount of time. The elapsed time before a timeout is generated may differ from state to state, but should be configurable. Perform same action as when receiving S_ABORT.
<ANY UNDEFINED>	FC shall discard the message, increment an internal error counter. If the internal error counter exceeds a configured number of errors, the same action as for S_ABORT should be performed.

Table 5 – Message handling common in all states

2.1.5 State INIT

The FC is initialising in this state. After reading configuration files, initialising output files and databases and so on, a socket connection is established to FS, and a S_CONNECT_REQ message is sent, before state AWAIT_CONNECT_ACK is entered.

2.1.6 State AWAIT_CONNECT_ACK

The FC has opened a connection to FS, has sent a connection request message to FS and is in this state waiting for an connection acknowledge from FS or a timeout.

Message	FC action
S_ACK	FS has accepted the connection attempt from FC. FC shall send a S_LOGON_REQ to FS and enter AWAIT_LOGON_ACK state.

Table 6 - Message handling in AWAIT_CONNECT_ACK state

2.1.7 State AWAIT_LOGON_ACK

FC has established a connection, and has sent a logon request to FS. FC is waiting for FS to acknowledge the logon request or a timeout.

Message	FC action
S_ACK	FC has successfully logged on and should send a S_CMD_START_REQ message to FS. This could either be the typical start-up sequence of first downloading protocol definitions, thereafter instrument definitions, before starting real-time update, or it could be a restart of real-time update.

Table 7 – Message handling in AWAIT_LOGON_ACK state

2.1.8 State AWAIT_COMMAND_ACK

FC has issued a command to FS, and is waiting for FS to return an acknowledge on the command or a timeout.

Message	FC action
S_ACK	FS has accepted and initiated command sent by FC, and will process the command until it is terminated either by the FS (with S_CMD_END_REQ), or FC (by aborting the command by sending S_ABORT).
S_NAK	FS could not start the command from FC. This is an internal FS error, indicated by the status code returned with the S_NAK message.

Table 8 – Message handling in AWAIT_CMD_START_ACK state

2.1.9 State AWAIT_CMD_END

FC is receiving data from the FS, and proceeds until FS signals there is no more data to be transmitted or for the FS or FC to abort the data transfer due to an error.

Message	FS action
S_CMD_END	Command processing has ended in FS. The status of the processing is sent as parameter (see . FC shall respond by sending S_ACK to FS. If FC has further commands to be processed, a S_CMD_START_REQ should be send, and current state should be changed to AWAIT_CMD_START_ACK. If FC processing is finished, a FC should send a S_LOGOFF_REQ and change state to AWAIT_LOGOFF_ACK.

Message	FS action
S_ENQ_REQ	FS has sent a probe to FC to see that it is alive and well. FC shall respond immediately by sending a S_ENQ_ACK, and returning the same sequence number as received in the S_ENQ_REQ.
<data>	Any application layer data (transactions) is messages not beginning with «S_», and should be processed by the FC application. Valid or known transactions are defined by the transaction definitions, available as a command. Others should be discarded (after reporting to the FC operator).
<ANY UNDEFINED>	This applies to any messages not handled above, i.e. unknown messages starting with «S_». FC shall discard the message, increment an internal error counter. If internal error counter exceeds a configured number of errors, the session should be aborted with S_ABORT, and a retry attempt could be made.

Table 9 – Message handling in AWAIT_CMD_END state

2.1.10 State AWAIT_LOGOFF_ACK

FC has logged off, and is waiting for an acknowledge from FS or a timeout.

Message	FS action
S_ACK	FC has logged off FS in an orderly manner, and should disconnect and finish processing.

Table 10 – Message handling in AWAIT_LOGOFF_ACK state

2.2 Initiating data transfer commands

When the feed client has logged in, a data transfer can be initiated.

2.2.1 Batch sessions

The batch requests are sent from FC to FS, and the coding of the data content returned from FS to FC in response («transactions») is based on the same framework as with real-time information.

The *batch* type of transmission is used to distribute reference data like security name, ticker and ISIN code.

2.2.2 Real-time sessions

There is only one command for ordering *real-time* information available to each client. This command has also a restart option. When the FC has logged in, a data transfer session can be initiated with the S_CMD_START_REQ message. In such a session, transactions containing “payload” data are transferred from the FS to the FC.

2.3 FC requests – “commands”

The following table is a list of the available commands that can be initiated from the FC in the S_CMD_START_REQ message:

Command	Parameter	Description
FIELDS_DEF	N/A	Send field definitions.
FIXED_DATA	N/A	Send security fixed information. See Feil! Fant ikke referanseilden. for sequence
TRANSACTIONS_DEF	N/A	Send transaction definitions.
REALTIME	[; seqNo]	Send trade system information from start of trading

Command	Parameter	Description
		day. If optional value <i>seqNo</i> is given and > 0, this implies a restart with transmission from the given sequence number.

Table 1 – FC request – “commands”

Example:

```
S_CMD_START_REQ;FIELDS_DEF
(or)
S_CMD_START_REQ;REALTIME;10234
```

2.4 FS responses – “transactions”**2.4.1 Transaction layout**

Item	Type	Range	Description
TransType	String		Transaction type identification, i.e. <i>TransTag</i> , is always the first value.
TransSeqNo	Integer	> 0	Transaction Global Sequence Number (GSN) is always the second value. GSN is only used for realtime data, i.e. empty for batch data.
<body>	N/A		One or more <i>fields</i> , see chapter 3.1 on page 19.
<TD>	N/A	[<CR> +] <LF>	Transaction delimiter character sequence.

Table 2 – Feed transaction layout

3. Presentation and encoding of data

This chapter describes the presentation or the “packaging” of the application layer information content, and relates to the “presentation layer” of the OSI reference model.

3.1 Transaction meta-format

A meta-format description of the transactions from the news feed server:

```
{Trans} = {Head} + {Body} + <TD>
{Head} = {TransType} + <SC> + {TransSeqNo}
{Body} = {Field1}[ + {Field2}...{Fieldn}]
{Fieldn} = <SC> + {Tagp} [ + {Valuep}]
```

Special characters are:

```
<TD> = {transactionDelimiterChar}[<CR>+]4<LF> = [ASCII (13)+] ASCII(10)]
<SC> = {separatorChar}';' = ASCII (59)
<AC> = {arraySeparatorChar}',' = ASCII (44)
<EC> = {escapeChar}'\' = ASCII (92)
```

[] – optional items are in brackets

{ } – item names are in braces

Ταβλε 70 - special characters

3.2 Special characters

The following special characters are used:

<i>TdtransactionDelimiterChar</i>	Transaction delimiter character used to separate transactions. ASCII (10) = <LF> (Line Feed). An additional Carriage Return ASCII (13) = <CR> (<CR><LF>) will also be interpreted as a TD.
<i>ScseparatorChar</i>	Separator character that is used to separate fields and also used in the head to separate the head elements. ASCII (59) = ';'.
<i>AcarraySeparatorChar</i>	Array value separator character used to separate array values. ASCII (44) = ','.
<i>EcescapeChar</i>	Escape character. Used in strings to escape special character so that they are interpreted as ASCII characters rather than characters with a special handling during decoding of the transaction. ASCII (92) = '\\'.

3.3 Transaction head

The transaction *Head* consist of a *TransType* and a *TransSeqNo*:

TransType	Transaction type, a string that describes the type of data in the transaction.
TransSeqNo	Transaction sequence number, a sequence number that identifies the transaction uniquely even across different sessions. It will typically be used by the client to restart communication from a defined point. The sequence number starts at 1 for the first transaction each day. The transactions have increasing numbers, but may increase by more than 1 from one transaction to another depending on the client's configuration. Only transactions received after ordering REALTIME data (see

⁴ <CR> is optional

chapter 2.3 on page 17) will have a transaction sequence number. For all other transactions, this field will be empty.

3.4 Transaction body

The transaction *Body* consist of one or more *Fields* that consist of a *Tag* and a *Value*:

Tag	Field name or identifier. The tag is encoded and delimited this way: The tag characters are always in set [A..Z, a..z,0..9]. The length of the tag is arbitrary, but the last character is always a lower-case character in the set [a..z]. If the first character of the tag is in set [a..z], this means that the tag has a length of 1 character, if in set [A..Z,0..9], it means that the tag has a length of more than 1 characters. Tags that start with a digit [0..9] indicates that the field context belong in an array (see chapter Feil! Fant ikke referansekinden. on page Feil! Bokmerke er ikke definert.).
Value	Field value. The field value will be one of the defined value types, and is implicitly determined by the tag and its specification. The value can be omitted for a field, and in that case the value implicitly is will be NULL (see chapter Feil! Fant ikke referansekinden. on page Feil! Bokmerke er ikke definert.) for the field.

3.5 Unknown field tags

If the client parser detects unknown field-tags, these should be discarded, allowing for introduction of new fields on the server side.

3.6 Null field values

A field does always contain a tag, but not necessarily have a value corresponding to the tag. If the tag's existence is carrying sufficient information, the value may be omitted. This should be interpreted as the field has a NULL value (which is void and undefined) as opposed to a value of 0.0 (which is valid and defined). See also chapter **Feil! Fant ikke referansekinden.** on page **Feil! Bokmerke er ikke definert.**

3.7 Field value types

Char[n]	ccc	fixed number of ASCII characters in set of [A..Z,a..z,0..9] <i>AlphaChar</i> is ASCII subset [A..Z, a..z] <i>NumChar</i> is ASCII subset [0..9] Number of characters is given in braces.
String(n)	sss	String of characters in a set defined by ISO Latin-1 character set (ISO8859-1). Embedded special characters will be escaped with <i>escapeChar</i> (i.e. embedded <i>escapeChar</i> , <i>separatorChar</i> , <i>arraySeparatorChar</i> and <i>delimiterChar</i> characters). An indication on maximum length may be given in parenthesis. It should be noted that this value can be changed on short notice.
Date	yyyymmdd	Year <i>yyyy</i> [0000-], month <i>mm</i> [01-12], date <i>dd</i> [01-31] Always encoded as 8 <i>NumChar</i>
Time	hh[mm[ss [...]]]	Time – hour <i>hh</i> [0-23], minute <i>mm</i> [0-59], second <i>ss</i> [0-59] etc. – Encoded as 2,4,6 or >6 digits <i>NumChar</i> . 12 => 12:00:00 (2 digits is hour) 1234 => 12:34:00 (4 digits is hour and minute) 123456 => 12:34:56 (6 digits is hour, minute and second) 123456789 => 12:34:56.789 (>6 digits is including sec. Fractions)
Integer	[-]n	Signed or unsigned integral number ⁵ .
Float	[-]n[.n]	Signed floating-point number ⁶ . Decimal point and decimals may be omitted.
IntVec	i ₁ ,i ₂ ,...i _n	Vector of arbitrary length containing <i>Int</i> values, separated by <i>arraySeparatorChar</i> .
FloatVec	f ₁ ,f ₂ ,...f _n	Vector of arbitrary length containing <i>Float</i> values, separated by <i>arraySeparatorChar</i> .
StringVec	s ₁ ,s ₂ ,...s _n	Vector of arbitrary length containing <i>String</i> values, separated by <i>arraySeparatorChar</i> .
None	N/A	No value associated with the field. The presence of the field itself carries sufficient information.

⁵All *integer* values can be represented as an integral number, unless otherwise specifically noted.

⁶ All *float* values may have up to 14 significant digits

4. Transactions and fields

This chapter describes all fields and which transactions they may occur in.

- Table 16 - Fields transactions on page Table 16 - Fields transactions²³ shows a mapping matrix for transactions and fields.

A field tag may be of variable length, and a lowercase *alphaChar* character marks the last character.

4.1 Real time transactions

Fields in bold are used for identification an entity. These fields are listed on the top of the table.

4.1.1 NewsItem (n)

Field Name	Field Tag
newsHeader	Nh
newsLength	NI
newsMsg	Nm
compId	CId
isinCode	i
issuerSymbol	ISYm
newsSource	Ns
newsType	Nt
newsTypeEnglish	NTe
newsURL	URLn
newsMessageId	NId
timestamp	t
newsLanguage	NLa
newsCorrectionId	NCId

Table 11 - News Item transaction

4.2 Basic data transactions (batch)

4.2.1 Equity (Be)

Field Name	Field Tag
isinCode	i
marketCode	Mc
secName	Sn
symbol	S
compId	CId

Table 12 - Equity transaction

4.2.2 Bond (Bb)

Field Name	Field Tag
isinCode	i
marketCode	Mc
secName	Sn
symbol	s
compId	CId

Table 13 - Bond transaction

4.2.3 Subscription Rights (BSr)

Field Name	Field Tag
isinCode	i
marketCode	Mc
secName	Sn
symbol	s
compId	CId

Table 14 – Subscription Rights transaction

4.2.4 Issuers (Bc)

Field Name	Field Tag
compId	CId
compName	Cn
issuerSymbol	ISYm

Table 15 - Issuers transaction

4.2.5 Fields (Bf)

Field Name	Field Tag
description	Ds
fieldType	Ft
name	n
tag	TAg
validFrom	Vf
validTo	Vt

Table 16 - Fields transactions

4.2.6 Transactions (Bt)

Field Name	Field Tag
command	CMd
description	Ds
name	n
tag	TAg

Table 17 - Transactions transaction

4.2.7 FeedEvent(Fe)

Field Name	Field Tag
description	Ds
timestamp	t

Table 18 – Feed Event transaction

4.3 Field descriptions

This table contains all available fields in the feed.

(Tag may be of variable length. A lowercase character marks the last character. This table is sorted by Field Names.)

Field Name	Field Tag	Field Type	Valid From	Valid To	Description
compId	CId	Integer	19980807		Company identification
compName	Cn	String	19980807		Company name
fieldType	Ft	String	19980807		Field type
isinCode	i	Char[12]	19980807		Unique identifier for security
issuerSymbol	ISYm	String	20081101		Symbol for issuer
newsCorrectionId	NCId	Integer	20080102		Reference to a previously distributed news message to be corrected by the current one
newsHeader	Nh	String	19980807		Header for a news message
newsId	NId	Integer	20070305		News identification
newsLanguage	NLa	String	20050606		Specifies the language of the news message. Not mandatory field.
newsLength	Nl	Integer	19980807		Number of characters in the news message body, including any escape-characters
newsMsg	Nm	String	19980807		News message
newsRef	Nr	Char[12]	19980807	20070305	Company isin for a news message
newsSource	Ns	IntegerVector	19980807		News message sources. See table "News source values" for available values
newsType	Nt	String	19990531		Type of the news (Norwegian Language)
newsTypeEnglish	Nte	String	20070305		Type of news (English language)
newsURL	URLn	String	20000501		URL (Uniform Resource Locator) pointing to OB's website, where the news message can be looked up
orderBookId	OBId	Integer	20011030		Unique identifier for a security in Equities and bonds market
secName	Sn	String	19980807		Security name
symbol	s	String	19980807		Symbol for security
validFrom	Vf	Date	19980807		First official date for the field <i>fieldname</i> on the feed
validTo	Vt	Date	19980807		Last official date for the field <i>fieldname</i> on the feed

Table 19 - Field descriptions

4.4 Constant values for specific fields

4.4.1 marketCode (Mc)

Value	Description
1	OB Equities
2	OB Bonds and Bond Derivatives
3	OMNO Derivatives Norway
4	Oslo Axess (Oslo Alternative Equities Market)
7	OB Alternative Bond Market
8	External
31	Nordic Equity Indices

Table 20 - marketCode constant values

4.4.2 newsSource (Ns)

Value	Description
1	Oslo Børs
2	ABM – Alternative Bond Market
3	Oslo Axess

Table 21 - newsSource constant values

4.4.3 newsType (Nt)/ newsTypeEnglish (NTe)

Nt values	NTe values
ANDRE BØRSMELDINGER	OTHER ANNOUNCEMENTS
AUKSJONSKALENDER KORTE STATSPA	AUCTION CALENDAR TREASURY BILL
AUKSJONSKALENDER STATSOBLIGASJ	AUCTION CALENDAR GOVERMENT BON
AVTALER	AGREEMENTS
BØRSPAUSE	MATCHING HALT
DERIVATMELDINGER	DERIVATIVE NOTICES
EKS.DATO	EX DATE
FINANSIELL KALENDER	FINANCIAL CALENDAR
FINANSIELL RAPPORTERING	FINANCIAL REPORT
FISJON / FUSJON	DEMERGER / MERGER
FLAGGING	DISCLOSURE REQUIREMENT
GENERALFORSAMLINGSINFO	GENERAL MEETING INFORMATION
IKKE-INFORMASJONSPLIKTIGE PRESSEMELDINGER	NON-REGULATORY PRESS RELEASES
INDEKSINFORMASJON	INDEX INFORMATION
INFORMASJON FRA OSLO BØRS	INFORMATION FROM OSLO EXCHANGE
INFORMASJONSDOKUMENT	INFORMATION DOCUMENT
KAPITALENDRINGER / UTBYTTEOPPLYSNINGER	SHARE CAPITAL CHANGES / DIVIDEND INFORMATION
MELDEPLIKTIG HANDEL	MANDATORY NOTIFICATION OF TRADE
NOTERING AV VERDIPAPIRER	LISTING OF SECURITIES
NYHETER1	NEWS1
NYHETER2	NEWS2
OBLIGASJONSHENDELSER	FIXED INCOME NEWS

Nt values	NTe values
OPPKJØP	ACQUISITIONS
PETROLEUM RESERVER	PETROLEUM RESERVES
PROSPEKT	PROSPECTUS
RESULTAT KORTE STATSPAPIRER	RESULT OF TREASURY BILL AUCTION
RESULTAT STATSOBLIGASJONER	RESULT OF GOVERNMENT BOND AUCTION
RESULTATUTSIKTER	EARNINGS GUIDANCE
SUSPENSJONER	TRADING HALTS
SÆRLIG OBSERVASJON	SPECIAL OBSERVATION
SØKNAD	APPLICATION
UTLEGGELSE AV STATSOBLIGASJONER	ISSUANCE OF GOVERNMENT BONDS
UTLEGGELSE KORTE STATSPAPIRER	ISSUANCE OF TREASURY BILLS
ÅRSOVERSIKT	ANNUAL INFORMATION

Table 22 - newsType/newsTypeEnglish constant values

4.4.4 newsLanguage (NLa)

Value	Description
NO	Norwegian
EN	English

Table 23 - newsLanguage constant values