eSpeed API C/C++ Application Programming Interface

ADDITIONAL FEATURES

System Version: 1.4.13

Revision Date: April 24, 2006

1 Overview

This document describes the details of additional facilities provided to applications by the eSpeed API for the following requirements:

- A mechanism for receiving notifications to a trading application that the customer trade feed application is connected to eSpeed.
- A mechanism for processing market data in a customer application architecture where there is a separate application instance for processing market data and then one or more other application instances for submitting orders to eSpeed.

Full eSpeed API documentation is available in the *C/C++ Application Program Interface: eSpeedAPI Reference Guide*.

2 Customer Support

Customer Support is available between 7:00am and midnight U.S. Eastern time, Monday through Friday, excluding holidays. When calling, please assist us by being ready with your product and account information.

The eSpeed Customer Support group has a series of phone numbers and e–mail addresses to meet various user needs. They are as follows:

- If you are a Customer with questions about possible trading scenarios or specific trading features (e.g. price improvement) please contact your account representative.

- If you are a Customer experiencing technical difficulty, have questions on how to use the system, please call or write:

  eSpeed Call Center (US) — (+1) (212) 610-2300 or support@espeed.com
  eSpeed Help Desk (Europe) — (0)20-7894–8600 or support@espeed.co.uk

- If you are a Customer or Salesperson requesting information on how to make changes to your electronic account, or require a new access account, please call or write:

  eSpeed Customer Access (US) — (212) 610-2300 or customeraccess@espeed.com
  eSpeed Customer Access (Europe) — (0)20-7894-8886 or customeraccess@espeed.co.uk

- If you are a Customer with questions regarding API development issues, downloading the latest version of the SDK or JNI, or testing a trade feed or market–making application, please call (between the hours of 9:00am and 6:00pm EST) or write:

  eSpeed Customer Integration — (+1) (212) 610-3560 or customerintegration@espeed.com

- If you are a Customer with questions regarding specific trades, verification of a trade, or are experiencing delivery problems with a trade, please call:

  eSpeed Trade Support (US) — (212) 610-2300
  eSpeed Trade Support (Europe) — (0)20-7894-8600

3 Background

3.1 STP Status

As users trade on eSpeed customers leverage the eSpeed API to facilitate a seamless flow of their trade executions through straight-through-processing (STP) to their risk, position and back office systems. If the straight-through-processing link is severed for any reason this status information is not available to the trading applications and/or end users. A new facility is provided that will deliver notifications to customer applications of the connection status of an associated straight-through-processing account.

3.2 Decoding market data for multiple users.

To disseminate details of the eSpeed API provides participant list fields in the market data stream for up to ten price tiers, such as the bid and offer lists. To access the information delivered in these fields client applications are required to call a `CFETIDecodeDataField` interface that is provided by the eSpeed API for this purpose. This function populates and delivers to the client application a data structure describing the break-down of entries in the participant list. If in such a list the entry corresponds to the user's own order then the API indicates this in the returned data structure. If the entry corresponds to another user in the same legal entity as the user then this also is indicated in the returned data. Similarly, to disseminate details of market availability the eSpeed API provides a field in the market data stream that the client application must decode using the `CFETIDecodeDataField` interface. A data structure describing the availability to the session requesting the decoded data is populated and returned to the client application. Full details of the mechanisms for decoding participant lists and market availability are given in the *C/C++ Application Program Interface: eSpeedAPI Reference Guide.*

In an application architecture where market data is subscribed to by a single user and traders use other eSpeed login accounts, the application must call the decode interface for each trading session in order to determine the correct ownership of each entry in a participant list or to determine market availability for each user. If the market data subscription is done in a separate application instance from the trading users, then even this is not possible and there is no means to access the information.

A new facility is now provided that provides customer applications with a mechanism to use the decode interface to obtain the details for all trading sessions in a single call.

4 User Connection Status

Notification of the connection status of eSpeed user accounts is provided to other eSpeed user accounts through configuration managed by eSpeed. Once the account has been suitably configured then, when its own trading system sessions have been successfully been established, the client application will be notified automatically regarding the connection status of users that it is configured to observe. Customers should contact their eSpeed customer integration representative to arrange for the necessary configuration to be completed.

4.1 Observed user list

If the customer account is configured to receive connection status notifications then, after a successful login, the client application will receive a notification to the application system callback providing the details of the eSpeed login accounts associated with the authenticated login.

Command	Command Status	Command Data Type
CFETI_QUERY_SUBSCRIBED_USERS_ACCEPTED	CFETI_SUCCESS	CFETI_TRADER_USER_LIST_DESC*

The command data delivered is a pointer to an eSpeed API data structure containing the list of eSpeed user login accounts that the eSpeed API is configured to observe and the number of logins in that array. This data structure is defined as follows:

```
typedef struct CFETI_TRADER_USER_LIST_DESC CFETI_TRADER_USER_LIST_DESC;
struct CFETI_TRADER_USER_LIST_DESC {
          unsigned int numEntries;
          const char** traderList;
};
```

where

`numEntries`	The number of users that will be observed
`traderList`	Array of eSpeed user ids that will be observed.

4.2 User Connection Status

Once the observing user has established its own trading system sessions the eSpeed API will automatically issues requests to the trading system on behalf of the application to subscribe to the connection status of the eSpeed user ids in the observed user list. This is repeated for each trading system session that is established by the observing user.

Subsequent to receipt of a connection-accepted notification delivered to the trading system session callback the client application will receive user updates for each user to indicate the current status of each of the users identified. If the connection status of any of the observed users then changes at any time during the lifetime of the session of the observing user, additional notifications will be delivered using the same mechanism.

Command	Command Status	Command Data Type
CFETI_SUBSCRIBE_USER_UPDATE	CFETI_SUCCESS	CFETI_API_SUBSCRIBED_USER_INFO_DESC*

The command data delivered is a pointer to an eSpeed API data structure containing the details of the user connection status for the user in question. This data structure is defined as follows:

```
typedef struct CFETI_API_SUBSCRIBED_USER_INFO_DESC
                    CFETI_API_SUBSCRIBED_USER_INFO_DESC;
struct CFETI_API_SUBSCRIBED_USER_INFO_DESC {
          CFETI_TRADE_SESS_ID sessionId;
          CFETI_TRADING_SYSTEM tradingSystem;
          const char* szSubscribedUsername;
          unsigned int connectionStatus;
};
```

`sessionId`	eSpeed API trading system session id allocated to the user that is being observed.
`tradingSystem`	The eSpeed trading system identifier for which the notification is being delivered.
`szSubscribedUsername`	The eSpeed username of the user.

`connectionStatus`	The connection status of the user. This is an enumerated value defined in cfeti_consts.h. The possible values are:
	
	`CFETI_USER_STATUS_LOGGED_IN`
	`CFETI_USER_STATUS_LOGGED_OUT`
	
	(For users of libESPD these are defined in espd/consts.h).

At the time of writing not all eSpeed trading systems will respond to the user connection status request that the API issues on behalf of the application. The customer application will therefore not receive connection status for these trading systems. For more information on the availability of the user connection status mechanism please contact your eSpeed customer integration representative.

5 Market Data

When a user is subscribed to additional trading sessions the handling of some market data fields is modified by the eSpeed API when the observing user has subscribed to instruments from that trading system.

5.1 Participant Lists

Participant list fields decoded using `CFETIDecodeDataField` are decoded into a data structure of type `CFETI_PARTICIPANT_LIST_DESC.` This data structure contains an array of pointers to elements of type `CFETI_PARTICIPANT_DESC`. Within each participant element the unsigned integer field `tsId` is the trading system session id of the participant if that session is known to the eSpeed API, some other non-zero value if the session is in the same legal entity as the decoding user. Otherwise the value zero is delivered in this field.

If the decoding trading session is observing other users for the same eSpeed trading system, and those users are have current trading system sessions, then the observing user will have received user connection status notifications to indicate that the observed users were logged in. Included in the data structure delivered with that notification is a trading system session id. When decoding market data participant lists, if the observed user then has one or more entries in such a list, then the decoded result shall indicate the identity of this session in the trading system session id field of the decoded participant entry.

5.2 Market Availability

If the decoding trading session is observing other users for the same eSpeed trading system and those users are have current trading system sessions, then when there is an update to market availability the market data field `CFETF_MARKET_AVAILABILITY_MULTI_USER` is delivered to the client application. To decode the field the market data application shall invoke the `CFETIDecodeDataField` method in the usual way. The eSpeed API shall populate and deliver to the client application a data structure that provides an array of market availability data structures, one for each registered trading system session id. The market availability description for each user is as it is defined in the *C/C++ Application Program Interface: eSpeedAPI Reference Guide*.

```
/**
 * The data structure CFETI_EXTENDED_MARKET_AVAILABILITY_DESC
 * is populated when applications decode the multi-user
 * availability market data field
 * CFETF_MARKET_AVAILABILITY_MULTI_USER.
 */
typedef struct CFETI_EXTENDED_MARKET_AVAILABILITY_USER_DESC
  CFETI_EXTENDED_MARKET_AVAILABILITY_USER_DESC;
struct CFETI_EXTENDED_MARKET_AVAILABILITY_USER_DESC
{
  CFETI_TRADE_SESS_ID tsSession;
  CFETI_MARKET_AVAILABILITY_DESC availability;
};

typedef struct CFETI_EXTENDED_MARKET_AVAILABILITY_DESC
  CFETI_EXTENDED_MARKET_AVAILABILITY_DESC;
struct CFETI_EXTENDED_MARKET_AVAILABILITY_DESC {
  unsigned int numUsers;
  CFETI_EXTENDED_MARKET_AVAILABILITY_USER_DESC* pUsers;
```

```
};
```

6 Alternative Solution

An alternative mechanism is also available that can be used for eSpeed trading systems where the user connection status notification mechanism previously described is not available.

This mechanism allows the client application to register additional trading session with the eSpeed API and to deregister them. Registering a session returns an eSpeed API trading system session id to the application. When a participant list is subsequently decoded the decode interfaces will highlight all of the registered trading sessions in the participant list entries with their corresponding trading system session id. If a registered trading system session id is not found then the normal rules for decoding participant lists to identify entries owned by other users within the same legal entity are applied.

A new include file, `cfeti_extend.h` is provided and defines the additional eSpeed API interfaces that applications must use to access the methods provided to retrieve and manage external trading sessions when managing market data. (For users of libESPD this include file is espd/extend.h). Customers that require access to this include file should contact their eSpeed customer integration representative.

6.1 CFETIGetTSSessionInfo

The `CFETIGetTSSessionInfo` interface is used to retrieve an internal description of the trading system session. Client applications shall call this method whenever a new trading system session is established when they receive an accepted connection to a trading system.

```
CFETI_TS_SESSION_DESC CFETIGetTSSessionInfo(
                              const char *sessionId,
                              CFETI_TRADE_SESS_ID tsSessionId
                              );
```

sessionId	The eSpeed session id allocated to the user when the user logged in.
tsSessionId	The eSpeed trading system session id allocated to the user when the user connection to the trading system was accepted.

The internal trading system session representation is defined as follows:

```
typedef struct CFETI_TS_SESSION_DESC CFETI_TS_SESSION_DESC;
struct CFETI_TS_SESSION_DESC {
     unsigned int id1;
     unsigned int id2;
     unsigned int id3;
};
```

The details in this data structure must be passed from the customer application through which the trader connected to the eSpeed platform to the customer application that is handling market data along with any other details appropriate to the customer application (e.g. username). The mechanism for doing so must be determined by the customer.

6.2 CFETIRegisterTSSession

To register a trading system session with the eSpeed API the market data handling application must call `CFETIRegisterTSSession.` The trading system identifier and the internal representation of the trading system session from the trading application are passed in as arguments and the method returns a trading system session identifier to represent this trading system session. When subsequently the application decodes market data participant lists, if entries in the list belong to the registered session then the trading system session identifier of the session will be included in the description of the participant list entry. There is no limit to the number of trading system sessions that can be registered in this way.

```
CFETI_TRADE_SESS_ID CFETIRegisterTSSession(
                                unsigned int tsId;
                                CFETI_TS_SESSION_DESC& tsSession
                                );
```

tsId	The eSpeed trading system identifier of the trading system that the trading system session belongs to.
tsSession	The internal trading system session representation obtained by the trader session by a call to `CFETIGetTSSessionInfo`

6.3 CFETIUnregisterTSSession

To discontinue decoding market data for a trading system session the market data handling application must call `CFETIUnregisterTSSession` . After successfully removing the session the eSpeed API will then no longer identify entries in participant lists for that trading system session.

```
CFETI_RC CFETIUnregisterTSSession(
                                unsigned int tsId;
                                CFETI_TRADE_SESS_ID tsSessionId
                                );
```

tsId	The eSpeed trading system identifier of the trading system that the trading system session belongs to.
tsSessionId	The trading system session identifier allocated when the trading system session was registered.