

## ASTS Bridge+ (pre-trade validation module)

**Warning:** when the pre-trade module is used, all transactions that are sent to the Moscow Exchange trading and clearing system are routed through an additional software which is not part of ASTS Bridge software. This may increase latency. Installation of additional software on the ASTS Bridge server is done at one's own risk and no support for this software is provided by Moscow Exchange.

This document describes one particular feature of ASTS Bridge that allows users to connect an external library (DLL) that performs pre-trade validation. Every client transaction is routed to this library and it can either approve or reject it.

An external DLL must implement three functions:

```
const
    ORDER_VALIDATION_ERROR_LEN = 256;
```

*Delphi*

```
function UserConnected(UserId, ErrorMessage: PAnsiChar): BOOL; stdcall;
```

*C++*

```
BOOL WINAPI UserConnected(char *UserId, char* ErrorMessage);
```

*This function is called when a user connects. Input parameter – user ID. Function must return either TRUE to allow user to connect or FALSE to reject this connection. It is possible to specify the rejection reason in the ErrorMessage parameter (text string, maximum length 255 characters). In case of rejection user will receive Error -30 (MTE\_LOGON).*

*Delphi*

```
procedure UserDisconnected(UserId: PAnsiChar); stdcall;
```

*C++*

```
void WINAPI UserDisconnected(char * UserId);
```

*This function is called when a user disconnects from the bridge. Input parameter – user ID. Function does not return any reply.*

*Delphi*

```
function ValidateOrderEx(UserId, TransName, Params, JsonParams,
    ErrorMessage: PAnsiChar): BOOL; stdcall;
```

*C++*

```
BOOL WINAPI ValidateOrderEx(char * UserId, char *TransName,
    char* Params, char* JsonParams, char *ErrorMessage);
```

*This function is called for every transaction sent by a client. Input parameters: user ID, transaction ID (TrnasName) and transaction parameters:*

- in Trading System buffer format (Params);

- in JSON format (JsonParams). JSON is formed by ASTS Bridge based on TransName transaction metadata. JsonParams can be NULL, If invalid transaction name specified or transaction parameters can't be parsed.

Function must return either TRUE to allow this transaction or FALSE to reject it. It is possible to specify the rejection reason in the ErrorMessage parameter (text string, maximum length 255 characters). In case of rejection user will receive Error -18 (MTE\_TRANSREJECTED).

**Remarks.** For compatibility the following obsolete prototype is also supported:

Delphi

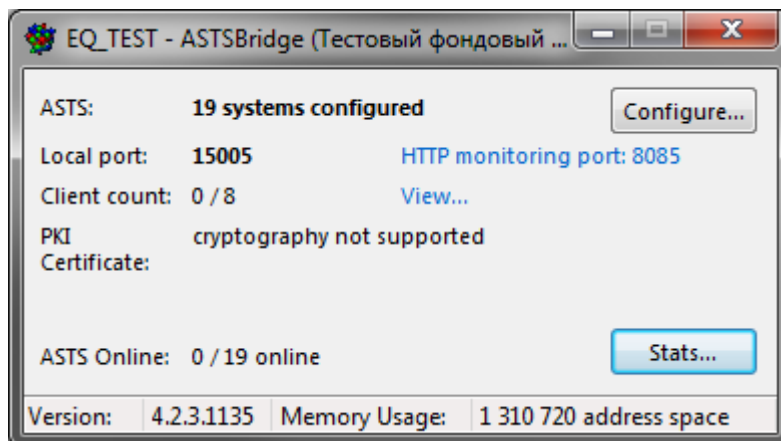
```
function ValidateOrder(UserId, TransName, Params,
  ErrorMessage: PAnsiChar): BOOL; stdcall;
```

C++

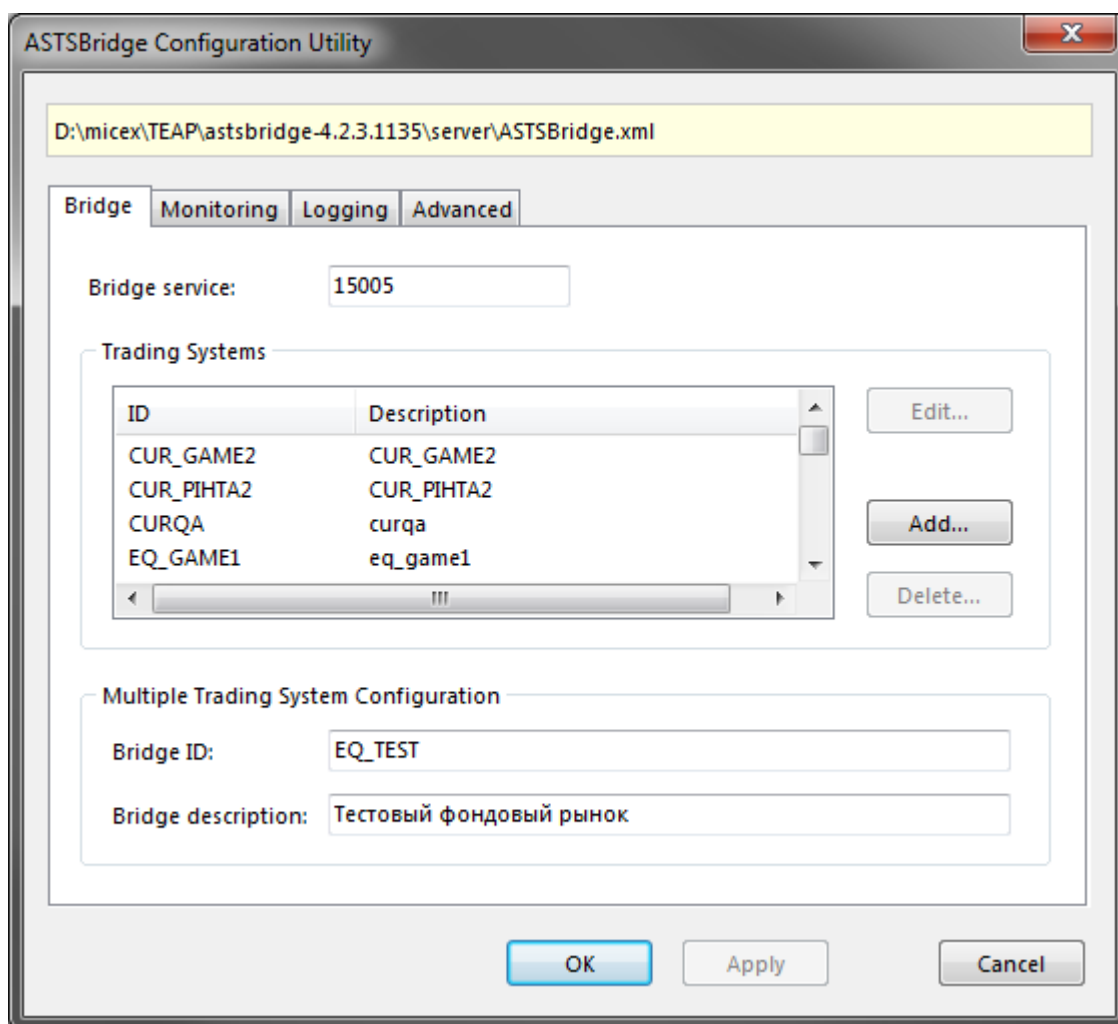
```
BOOL WINAPI ValidateOrder(char * UserId, char *TransName,
  char* Params, char *ErrorMessage);
```

Packed with this document is an example with source code (OrderValidation). This demo DLL asks the question "Allow transaction %s from user %s with arguments "%s"?" for every transaction being passed through. You can connect this DLL as further described and try to send transactions using the TEClient.exe application (included in ASTS Bridge package) to see how it works.

It is possible to link different DLLs to different clients. To set up a link to DLL use the bridge configuration tool (available starting from ASTS Bridge version 4.1): start ASTS Bridge and click Configure...

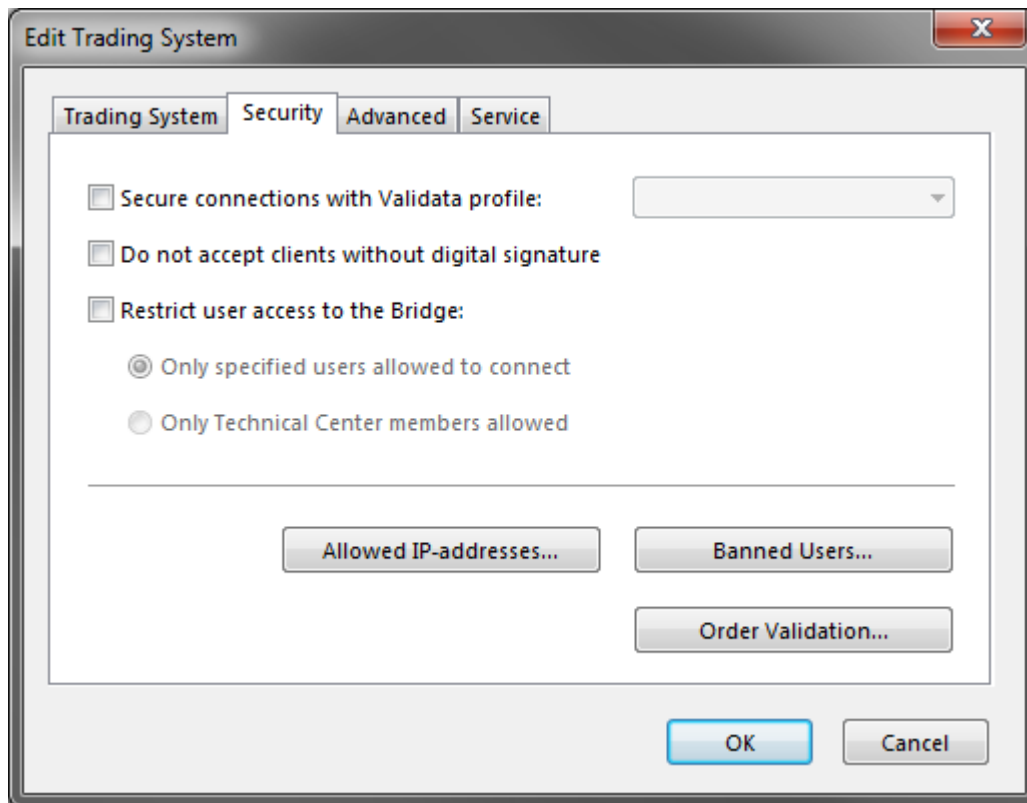


Configuration Utility will start:

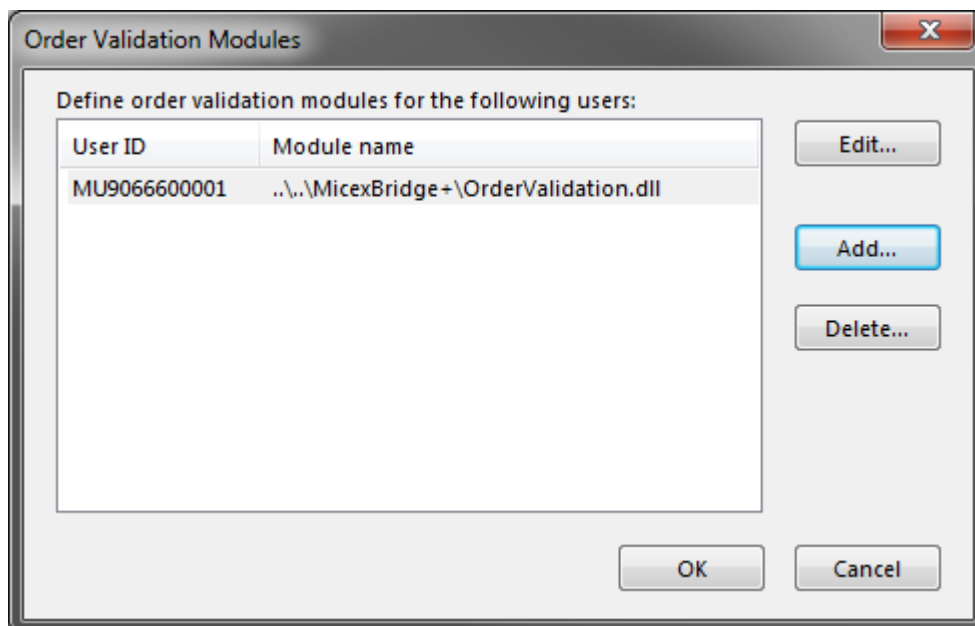


Bridge configuration file should be stored in XML format. Click Convert to XML to perform this conversion if that was not done before.

Select the trading system and click Edit to open connection settings:



Go to Security tab and click Order Validation.



Enter User ID and select a DLL that should be used to validate this user's transactions.