BUDAPESTI ÉRTÉKTŐZSDE
BUDAPEST STOCK EXCHANGE

K2

Connection Server

Frequently Asked Questions

Author: A. Pollák, IT Development Division
Version: **1.2**
Date: 17 January 2011
Path:
Filename: K2_FAQ_1_2E

cee stock exchange group

Contents

Document History

Version	Author	Date	Summary of Changes
1.0	A. Pollák	11 January 2011	First draft
1.2	A. Vass	17 January 2011	Revised version

Introduction

The purpose of this document is to provide useful practical information to Exchange members using the K2 connection server on questions regarding the everyday operation and the settings of the K2 software. (Thorough knowledge of the subjects included in the full K2 documentation is a prerequisite for understanding this practical guide.)

The K2 is apt for connecting both to the MMTS I cash market trading system and to the MMTS II derivatives trading system of the BSE.

Correspondently, the K2 software components designated for use with the cash or the derivatives markets are marked with 1 and 2 respectively. This document will discuss commands for the cash market components designated by number 1 in the command name, command parameter or in the file name of the parameter file, however the components marked with 2 for the derivatives market work in the same way.

Installation of the K2 connection server software

Production environment

- ***How to set up the K2 software? (First installation on a trader member's computer)***
  Please, follow the instructions of the **K2_inst_guide_1_7E.pdf** document made available at the BSE web site under the following link:
  http://www.bse.hu/topmenu/members/members_technikai_informaciok/members_technical_doc_en/members_tradesoftw_doc.html
  K2 documentation can be downloaded compressed in **k2doc.zip**.
  Regarding the first installation and first use please see also the section comparing the K2 and the TW software under the miscellaneous questions.

- ***After the first installation configuration files are missing from the k2/cfg directory. How can the files be provided?***
  The contents of the said directory (5 files) are sent by the BSE in mail when the trader member or developer firm has already signed the agreement for the use of the K2 connection server.

- ***How to perform version upgrades?***
  Please, follow the instructions of the concise „*K2 install guide"* document made available for download separately in the **K2_new_version_install_instr_2007_06.pdf** file under the same link already mentioned above.

K2 server connecting to the test (simulation) environment of the BSE

- ***Can the test and production K2 server connecting to the simulation and production trading system of the BSE share the same computer?***

  Our experiences show that under normal load conditions the test and the production K2 server can be operated on the same hardware. The production and the test K2 servers must be installed in the **k2** and the **testk2** Linux/Unix user accounts respectively.
  Further differences in the configuration parameter settings of the two accounts:

- The **test1** (and **test2**) service ports must be specified in the „etc/service" file in the **testk2** account (See page 17 of the **K2_inst_guide_1_7E.pdf** document)
- In the **k2/cfg/pgw1.cfg** configuration file the value of the following parameters must be set differently:
  - *ipckey* (the key referring to the shared memory tables)
  - *ifssservice* (**ifss** and **ifss2** in the **k2** production account; **test1** and **test2** in the **testk2** test account)
  - *logpath* (must be set differently only if the full or absolute path is used, E.g. /home/k2/k2/log)
  - *rlogpath* (the same is true as mentioned above for logpath)
- In the a **k2/cfg/pgwtsmr1.ini** configuration file of the test account the **broadcastAddress** parameter must hold the IP address of the central simulation server of the BSE:

  broadcastAddress=192.168.15.1

- ***What are the advantages of installing the test K2 server on a separate computer?***

  If the test K2 server is installed on a separate computer, then the parameters can be the same as those of the production K2. The only exception is the IP address mentioned above (the **broadcastAddress** parameter in the **k2/cfg/pgwtsmr1.ini** configuration file). With this solution the test K2 server can also serve as a kind of cold backup of the production K2.

Directory structure

- ***What are the functions and the contents of the directory structure unpacked in the installation process?***
  - k2/**bin** contains programs and command files
  - k2/**cfg** contains configuration files (ifss.uaf, pgw1.cfg, pgw2.cfg, pgwtsmr1.ini, pgwtsmr2.ini).
  - k2/**log** contains log files (pgw1.log, ifss1.log, pgw2.log, ifss2.log); and the daily transaction files required for the contingent intraday restart are also stored here (restart log) (YYYYMmmDD_X.rlg , where X can be 1 or 2)
  - k2/**pkg** contains files (and template programs) stored in different subdirectories that can be used for the API programming

Shared memory tables in the K2 operation

- ***What is the role of the shared memory tables and the processes using them?***
  The **pgw1** process (program component) is responsible for the connection with the central trading system of BSE and for maintaining the trading data queried from the central system. The **ifss1** process satisfies the information requests from the K2 client programs by sending them the demanded trading data that was written into the shared memory tables by the pgw1 process. At the same time the **ifss1** process receives transactions from the client programs and stores them in the corresponding shared memory tables (orderentry, mmorder) which will be forwarded into the trading system by **pgw1**.
  The detailed description of the shared memory tables can be found in chapter 34 (and 35) of the **K2_refifsc_2_9E.pdf** programmers' reference document.

- ***What are the shared memory tables used in the K2 server?***
  The K2 server uses the following shared memory tables:
  market, instr, sector, board, secboard, firm, user, order, trade, negdeal, orderentry, mmorder, auditevent, priceparam, systime.
  - The **systime** table is a specific table consisting of a single record (row) only. In the client API the querying of this table is different from that of the other shared memory tables (it is performed by using a specific function). The table can be used also for monitoring the **pgw1** process from the client side API.
  - The **secboard** table always has the same number of rows as the **priceparam** table. Actually the priceparam is a supplementary table appended to the secboard table.
  - The **orderentry** and **mmorder** tables hold the transactions sent in by the client programs, which means that their records are written by the **ifss1** process. Based on the responses received from the central trading system the **pgw1** process overwrites and refreshes certain fields of this table such as the status field.

Operating the K2 server

- ***Can we run the K2 server round the clock (continuously for 24 hours)?***
  The K2 server program has to be started and stopped each and every trading day. The K2 server can be started only after the central trading system of the BSE is already up and running and fulfils the login request from the trader users. (The start will be successful in most cases after 8 a.m.) The trader users can remain logged in for at least 20 minutes following the end of day close of trading (after the end of the last trading session), which means until 5:30 p.m. at present. At that time the operators of the BSE perform a forced logoff for the users resulting in the automatic termination of the pgw1 process. However the ifss1 process will remain operational and will be able to receive further requests of the K2 client programs for providing the day's trading data from the shared memory tables. Transactions sent in at this phase will not be executed, and the K2 server program can be stopped at any convenient point of time.

- ***Is it possible to automate the start and stop of the K2 server?***
  Under the Linux/Unix operating systems command execution can be scheduled using **crontab**. In this manner it is possible to start and stop the K2 server program without manual intervention. The required login password which is entered from the consol keyboard at manual start can be specified with the –p option of the k2start command.

- ***How can I change the login password of the K2 server?***
  At the time of the first login to the trading system of the BSE it is necessary to change the initial password provided by exchange, if one wants access to the full functionality. The password can be changed using the **k2changepwd** command before issuing k2start!

- ***How can I monitor processes and the memory tables?***
  The **mem1** command lists information about the processes of the K2 server and about the memory tables:

```
==> k2test2 1
KEY: 0x70677731 SHMID: 2785282 SEMID: 3244032
---------------------------------------
                Num     Max       Recsize
market            4       4            48
```

```
sector            31     31           44
instr             44     44           48
board             25     25           88
secboard         552    552         1392
firm             623    623          300
user             355    356          324
order            375  30000          248
trade             45 120000          212
negdeal            0   1000          172
auditevent_t      48    500           60
auditevent_g       9    500           56
auditevent         3  60000          332
orderentry       208  50000          232
mmorder            0  10000          300
badreplymsg        4  25000          256
---------------------------------------
Last orderno:          588061
Last tradeno:          178879
Last last_n tradeno:   178879
Last negdealno:        588061
Stats Version:     0x4d2a9240
Log Time:                  0
Transactions:             64
Rejections:                0
---------------------------------------
     pid     ppid comm  vshm   shm  starttime  utime  stime status
00027396 00000001 pgw1   n/a   n/a    4057832      0      0  Sleep
00027399 00000001 ifss   n/a   n/a    4057837      0      0  Sleep
00027401 00027396 pgw1   n/a   n/a    4057843      0      0  Sleep
00027714 00027687 mem1   n/a   n/a    4078363      0      0    Run
00027715 00027714 mem1   n/a   n/a    4078363      0      0    Run
=======================================
```

The first part of the list displays the fill up status of the most important shared memory tables. The first column in the list (Num) indicates the number of filled up (completed, not empty) records while the second column (Max) displays the total number of records allocated for the given table. If the first number reaches the value of the second number (Max), then the process executing the increment will halt with an error code. (This is not true for the first 7 static tables which do not change their size during the trading day. The **pgw1** will stop if the *order, auditevent,* and *badreplymsg* tables become full, however the total fill up of the *orderentry,* and *mmorder* tables will cause the halt of the **ifss1** process.

The middle part of the list contains certain parameters of minor importance related to tracing transaction processing. The end of the list displays the active processes of the K2 server.

If one wants to list only specific processes, then the

ps –fu k2 | grep pgw1

command can be used. This command lists the pgw1 processes of the k2 user. Under normal conditions there will be two instances of this process displayed, one for maintaining connection with the central trading system and the other for monitoring the operation of the first.

The

ps –fu k2 | grep ifss1

command lists the ifss1 processes of the K2 user. Under normal operation there will be 1+n instances of this process displayed, one for handling the connection requests of the K2 client programs and n instances for the established client connections, where n denotes the number of existing client connections. There is a separate ifss1 process for each client connection.

The contents of the individual shared memory tables can be listed by the

get_table

command, which will connect to the K2 server and query the table content in the same manner like a normal client program.

The

dump1

command will make a core dump of the contents of the shared memory tables. These raw data are not so easily readable, but can be used if the ifss1 processes halts and no new client connection is possible.

- ***How can I search for error messages in the log files?***

  The *k2/log/pgw1.log* file is written by the **pgw1** process, while the *k2/log/ifss1.log* file is written by the **ifss1** processes. Each log entry (row) in the log file starts with an identifier followed by a description part. The identifier is always terminating with a hyphen followed by a single character that can be **I** for information messages, **W** for warning messages and **E** for error messages. So if we are looking for error messages we have to search for the rows containing E characters, like in the following example:

  grep [-]E ifss1.log

  The log files are continuously increasing (by appending the new entries) each day, so the contents will be more and more difficult to search and handle. Care must be taken because if the size of a log file reaches 2 GB, the handling process will not be able to write it and will halt with an error message. Therefore it is prudent to save (archive) and delete the log files from time to time (one might as well save and delete them each day after the close of trading and stopping the K2).

- ***How can I review transactions of a given date?***

  The *k2/log* directory contains also transaction log files for each day and each system (1 and 2). The contents of these transaction log files can be read only by a specific utility program. For example the following command will list the 3rd January 2011 transaction log file for the cash (MMTS I) market:

  rldump1 2011Jan03_1.rlg

- ***When it becomes necessary to restart the K2 server program during the trading and how to do it?***
  It can happen that the pgw1 process halts due to a failed network connection between the process and the central trading system. Shared memory table overflow (if a table becomes full) leads also to a process termination. In such situations the K2 server program has to be started again, but always after a previous stop command (**k2stop 1**). In case of table overflow it is necessary to increase the size of the given table in the *k2/cfg/pgw1.cfg* configuration file. After the necessary resizing the K2 server can be started with the usual start command (**k2start 1**). At the start the contents of the day's transaction log files will be loaded into the orderentry and mmorder tables, so their before-the-halt status will be restored. It is important that at the start the values of the record change sequence numbers (seqno) are generated afresh; therefore the K2 client programs can't rely on sequence numbers of the preceding session!

Configuring the K2 server

- ***What are the parameters that can be changed in the pgw1.cfg configuration file?***
  The parameters that can be modified are as follows:
  ipckey, ifssservice, logpath, rlogpath, verbose, foreground, watchpgwenabled, freesemenabled, maxorderentry, maxbadreplymsg, maxmmorder, maxauditeventtype, maxauditevent, firstorderentry, numoforderbook, orderratio, password, skiplogchk, tsmrtimeout, timesynctimeout, timesyncretry, timesynccycle, ordlistsize (this parameter can be modified only in the pgw2.cfg file)
  Most parameters don't need any modification. If a parameter is not specified in the configuration file then the default setting will prevail.

  Parameters that may need to be changed:

  - The following parameters have to be set differently for the test and production K2 servers running on the same computer:
    - *ipckey, ifssservice, logpath, rlogpath*
  - Parameters for sizing the shared memory tables:
    - *maxorderentry* — size of the orderentry table; must be greater than the expected number of daily transactions sent into the K2
    - *maxbadreplymsg* — size of the badreplymsg table; this table stores the error messages sent in response to the transactions refused by the central trading system or denied by the K2 server
    - *maxmmorder* — size of the mmorder table; must be greater than the expected number of daily market-maker transactions (pairs of buy/sell market-maker orders) sent into the K2
    - *numoforderbook* — the maximum total number of orderbooks (market by order and market by price) planned to be monitored in the K2 server; it can be greater than the value of orderbookcyclic
    - *orderratio* — this parameter customizes the size of the order table; the size of the order table of the K2 server is calculated by dividing the size of the order table in the central trading system. The default value is 10, which means that under normal conditions the order table size in the K2 will be one tenth of size of the same table in the central trading system. As the trader firm receives information only about its own this one tenth size is usually enough for a firm of average market share.

- ***What are the protected licence-dependent parameters?***
  The protected licence-dependent parameters are as follows: *username, firmname, cycletime, orderentrypercycle, timesyncenabled, intrefenabled, orderbookcyclic, sleeplimit, sendsignal, cyclicobenabled, fieldchgenabled, retrieveordno, marketbyprice, marketbyorder, orderbookchange, numofclients,* ***license****.*
  The protected licence dependant parameters should not be modified!

- ***What is the difference between the licence types?***
  The trader firm using the K2 server can choose from three licence types: ENTRY, STANDARD, PLUS, which differ only in the value of certain licence-dependant parameters:

	ENTRY	STANDARD	PLUS	
orderbookcyclic	0	50	100	number of order books that can be obtained from the central trading system in a single query cycle
marketbyprice	0	1	1	enables access to order books in the aggregated market by price view
marketbyorder	0	0	1	enables access to order books in the detailed market by order view
sendsignal	0	1	1	allows sending a signal to wake up the pgw1 process if new transaction is received from a client program
fieldchgenabled	0	1	1	enables efficient field level query concentrating only on refreshed dynamic fields of the secboard table
retrieveordno	0	1	1	the order number given to a newly entered order in the central trading system will be written into the OrdNo field of the orderentry table
numofclients	2	10	10	maximum number of ifss1 processes that can run in parallel; limits the number of simultaneous client connections

- ***What privileges can be granted to a K2 client program and how can I assign them?***
  The ***k2/cfg/ifss.uaf*** user authorisation file contains the privileges assigned to the K2 client users. The file can be edited by a text editor. The records consist of four colon-separated fields:
  username:password:active/suspended:privileges(separated by commas)
  Examples:

broker1:pwd1:**s**:query	suspended user – cannot connect to the K2 server
broker2:pwd2:a:**query**	the user has a query (read only) privilege only
broker3:pwd3:a:query,**entry**	orders sent into the K2 by this user get an „A” accepted status – confirmation is required
broker4:pwd4:a:query,entry,**confirm**	the user is authorised also to confirm own or other users' orders
broker5:pwd5:a:query,entry,confirm,**bypass**	the orders sent in by this user get a „C” confirmed status immediately – no further confirmation is required
broker6:pwd6:a:query,**config**	the user can configure the list order books to be monitored in the K2

The use of the API

- ***Where can I find the header files and function libraries required for the programming?***
  The header files (and the source code API template programs for facilitating interface development) can be found in the ***k2/pkg/src*** directory. The header files:
  - *fields.h* constant, structure, enumeration and type definitions
  - *ifsapi.h* function definitions of the API
  - *ifsdefs.h* constant definitions
  - *ifsutil.h* definitions of auxiliary functions for handling fields of interface structures

  The dynamic function library is operating system platform dependent. E.g.:
  - k2/pkg/WIN32/**ifs.dll**
  - k2/pkg/Linux_RHEL4/**libifs.so**

- ***What are the functions available for API programming?***
  Prepare connections, connect and disconnect:
  - *ifsc_create, ifsc_connect, ifsc_disconnect*

  Handling sequence numbers and record indices:
  - *ifsc_set_get_seq, ifsc_set_get_idx, ifsc_set_get_field_seq*

  Query shared memory tables:
  - *ifsc_get_first_record, ifsc_get_next_record*
  - *ifsc_get_first_idx, ifsc_get_next_idx*
  - *ifsc_get_first_field_chg, ifsc_get_next_field_chg*

  Order book handling:
  - *ifsc_get_orderbook_list , ifsc_orderbook_conf*
  - *ifsc_get_first_orderbook, ifsc_get_next_orderbook*
  - *ifsc_get_marketbyprx_list , ifsc_marketbyprx_conf*
  - *ifsc_get_first_marketbyprx, ifsc_get_next_marketbyprx*

  Monitoring system time and K2 status:
  - *ifsc_get_systime*

  Entry of transactions:
  - *ifsc_orderentry, ifsc_orderentry_status_chg*
  - *ifsc_mmorder, ifsc_mmorder_status_chg*

  Error handling and trace logging for the communication:
  - *ifsc_get_last_errmsg*
  - *ifsc_set_trace*

  Besides, the auxiliary functions with *ifs_get_* and *ifs_set_* prefixes can be used for handling fields of message structures.

- ***The shared memory tables can be queried based on sequence numbers or physical record indices. Which method should be used?***
  The query based on sequence numbers returns records in the order of their updates. If between two queries a record is changed more than once, only the latest update will be returned. The query based on physical record indices has no practical importance and is rarely used.

- ***How can I monitor the operation of the pgw1 process from the client side?***
  The *ifsc_get_systime* function returns a status record in which the **K2QueryTimeOffset** and the **PgwState** can be used for monitoring the status of the pgw1 process.
  The **K2QueryTimeOffset** field is incremented by 1 each second, but is reset to 0 each time the pgw1 process accomplishes a message transfer with the central trading system successfully. If the value of this field begins to grow (E.g. will become greater than 5) it means that the pgw1 process is not communicating with the central system for the moment, but there is still hope that the problem will be solved after an automatic reconnect. If the value of this field is greater than 60 (more than a minute has elapsed) than it is expedient to check the status of the server side processes of the K2 using the console.
  If the content of the **PgwState** field is 0, than the pgw1 process is up and running normally. If it is not zero, then the pgw1 was halted due to an error (that can be detected in the *k2/log/pgw1.log* file). This situation occurs regularly each day when the BSE operation performs a forced logoff for the users of the central trading system at the end of the trading day. If the situation is experienced before the end of the trading day then the K2 server program has to be restarted (started afresh after stop).

- ***How to monitor the communication between the K2 server and the K2 client programs?***
  The ***ifsc_set_trace*** function can be used to get the API to store messages exchanged with the K2 server into the ***ifstrace.log*** file. This file will be created in the directory where the client application program using the API was started.

Order entry

- ***Where can I find information required for setting the ordermethodset field?***
  This field is a 32 character long string consisting of the „0” and „1” characters. The first (leftmost) character is the 0 index character. The description of this field can be found in the **refdata1a_1_3E.pdf** document.

- ***An order entered into the K2 server was not forwarded into the trading system. What can be the problem?***
  Most probably the client program entering the order has no bypass privilege. Therefore the order was stored in the **orderentry** table with an „A” (accepted) status. A client side confirmation - setting the „C” (confirmed) status with use of the *ifsc_orderentry_status_chg* function - will be required to forward the order into the central trading system.
  It can happen that the client has the required bypass privilege but the K2 server detected an error in the transaction and the order was given a „D” (denied) status.
  The *ifsc_orderentry_status_chg* function can be used not only to confirm but also to decline the order transaction entered by setting a „D” (denied) status.

- ***How do I know that my order was admitted into the trading system?***
  The pgw1 process forwards transactions of „C” (confirmed) status into the central trading system one by one (the number of orders forwarded during a single cycle cannot be greater than the number specified in the *orderentrypercycle* configuration parameter).
  The „U” (unknown) status given to the forwarded order transaction when initiating the send message will be changed after receiving the reply to an „E” (entered) if it was admitted or to an „R” (refused) if it was rejected by the central trading system. When receiving an „E” status the **OrdNo** field will also be completed (only in the case of the STANDARD and PLUS licences) with the order identification number given to the order in the central trading system. When receiving an „R” status the error message returned will be stored in the **Msg** field.

- ***When the status of an order transaction will remain* „U" (unknown) and what can be done with it?**
  This can happen if the pgw1 process halts after forwarding the order to the central trading system, but before receiving the reply. In such cases the „U" (unknown) status will remain unchanged also after restarting (starting anew) the K2 server, but from the client side it is possible to modify it with the *ifsc_orderentry_status_chg* function. The „U" (unknown) status can be set to „C" (confirmed) again, in which case the pgw1 process will try to send the order into the central trading system again. Alternatively one can set the status to „D" (denied) indicating that the order entry is declined. Of course the „U" status can also be left unchanged which effectively doesn't differ much from setting it to „D".

- ***What date can be specified for the settlement date?***
  The settlement date doesn't need to be specified because the trading system will ignore it. This field should be entered only for the negotiated deals, but the said function is not available with the K2 server. The settlement date is filled by the central trading system in accordance to the given market rules and the field will appear in the *order* table.

Order amendment (invalidates original + creates new order)

- ***What happens to an order if it is modified?***
  The original order will get an „A" (amended) status and will not be valid any more. In the order table a new order with a new order identifier number will be created by the central trading system.

- ***Is it possible to modify more than one order simultaneously?***
  Yes. The *ifsc_orderentry* function can be called with specifying the IFS_ACTION_TICKUPTICKDOWN transaction type. In this case prices of orders that correspond to the specified filtering condition will be modified simultaneously by the given number of price ticks.

Order withdraw

- ***Is it possible to withdraw more than one order simultaneously?***
  Yes. Care must be taken because without specifying a filtering condition all orders of your firm will be withdrawn. If the order identifier number is specified as a filtering condition then only the given order will be withdrawn, and in this case no other information is necessary for the withdrawal.

Market maker order (entry of a compound order with buy and sell side)

- ***Will a new market maker order withdraw the previous buy and sell pair?***
  Yes. According to the rules configured in the central trading system new market maker order entry will always withdraw the still open buy/sell order (or pair of orders) resulted from the previous market maker order.

- ***What can be the reason if a market maker order was refused by the central trading system?***
  Only traders with proper entitlement can send market maker orders to the central system. In the production trading system this authorization is set by the BSE or the GDMA (Government Debt Management Agency). In the simulation trading system the authorization is set by the BSE in each case and the permission will become effective at the earliest at the beginning of the next trading day. A trader user can be a market maker for more than one product and more than one trader can perform market making for the same product.

Querying orders from the central trading system

- ***How do I know what happened to my orders admitted into the central trading system?***
  The orders entered will appear in the *order* table and their status can be traced by monitoring the **Status** and **Balance** fields. Possible values of the **Status** field are described in the **K2_refdata1a_1_3E.pdf** document. The **Balance** field indicates the remaining quantity of the order eligible for matching with another order. If the Status field is set to „M" (Matched), it means the order is fully matched and the Balance field is 0.

Querying trades

- ***How can it happen that the same trade record was received twice on the client side?***
  When the K2 server downloads data from the central trading system, the trades of the firm the K2 user belongs to will be downloaded twice. First in a brief form among all trades executed in the trading system and once again as a detailed own trade in which the firm is involved either on the buy or sell side. The first record will be stored in the trade table as a new record; the second will only update the record and increase the sequence number.
  The K2 client programs perform the query based on the sequence numbers and most often they will obtain the already updated record only once. However under conditions like at the end of the day closing period when a large number of trades are generated simultaneously, a query may return data before and also after the update. In such cases both the brief and the detailed trade records will be received.

Monitoring order books

- ***How many order books can be monitored simultaneously?***
  The number of orderbooks that can be monitored simultaneously is limited by the **numoforderbook** parameter specified in the configuration file which determines how many order books can be taken on the monitoring lists. As this parameter can be freely set by the user one might as well monitor all of the order books. However there are two important issues that have to be considered:
  - Configuring too many products on the monitoring lists may heavily load the communication line between the K2 and the central trading system.
  - The PLUS licence type allows the refresh of 100 order books within one cycle (one cycle = 1 second). If I want to monitor order books of 200 products then the order books will be updated only each 200/100 = 2 seconds.
- ***What is the order book depth that can be displayed?***
  In MMTS I cash market trading system both the market by price and the market by order view of the order book displays 20 levels on both the buy and sell side. There is no such limit in the MMTS II derivatives system.

Miscellaneous

- **What are the extra or reduced functions of the K2 server if compared to the Trader Workplace (TW) front-end software?**

  The K2 server program is suitable for connecting the back office IT systems of the brokerage firm to the central trading system of the BSE through an Application Programming Interface facilitating direct trading techniques like automated order entry and decision making based on market data derived from the trading system. An example can be the automated market making. In short, the K2 can be used for building computer applications capable to respond even to the most abrupt changes of the market.

  The Trader Workplace (TW) software supports manual trading through its Graphical User Interface (GUI). However it provides functions not available from the K2 server, like managing the trading accounts or entering fixed orders (negotiated deals). Due to the lack of the trading account management function in the K2 also the trader firms using the K2 server need to install at least one instance of the TW and use it if required. Before the first use of the K2 server it is necessary to assign the required trading accounts to the K2 user. This is a prerequisite of successful order entry.