V. Yu. Kotsubynskiy, Cand. Sc. (Eng.), Assist. Prof.; R. S. Holovaha

IMPROVEMENT OF THE MESSAGES TRANSMISSION PROCESS, USING FIX/FAX PROTOCOL IN FINANCIAL CLIENT-SERVER SYSTEMS

Problem dealing with of delays appearance during transmission of large volumes of information in the sphere of electronic trade in financial markets is considered, modern methods of financial data organization were studied, detailed analysis of the approaches to solution of described problem is realized.

Key words: *data transmission, FIX-protocol, FAX- protocol.*

Introduction. The sphere of electronic trade in financial markets attracts great attention since rapid development of the information technologies and Internet . Due to modern facilities of automation of this process there is no necessity for trader to be directly at the exchange, as he is able to buy any assets with the help of his trade terminal, making orders through the Internet.

Wide spreading of electronic trade in financial markets gave the significance impulse to the development of technologies in this sphere.

Survey of the previous research. In 1992 there had been elaborated the Financial Information eXchange (FIX) protocol for the structural organization of the financial data, being transmitted. FIX- protocol is the international standard for financial information transmission between participants of exchange tradings in real time mode. FIX protocol is supported by the majority of the largest world's banks and exchanges as well as by electronic systems for internet-trading [1].

Usage of the FIX-protocol has such advantages:

- ideal structure of transmitted data, makes FIX-messages simple for reading both by programme code and by man. FIX messages consist of sets of fields, separated from each other by special code;
- FIX-protocol allows to check the integrity of data, being transmitted, since it determines the length of message body and controls sum of all bytes;
- support of lost data returning. FIX – protocol determines the field that corresponds to the ordinal number of the message. If the sequence of data is violated, then each of parts will be able to make inquiry for the lost messages;
- safety of data exchange. FIX-protocol supports authorization of both parts and data coding using different methods [2].

Eventually, volumes of trading by financial assets increased considerably, causing the increase of data volume, being transmitted, appearance of significant delay during message transfer and failure of client-server systems to operate in real time mode.

The problem of delay appearance during messages transition, entailed by enlargement of data volumes, transmitted in financial nets was discussed in 2004 in New York at the conference of company FPL (FIX Protocol, Ltd. that possesses the rights and supports FIX-protocol specification). The format of classic fields of FIX-protocol is considered to be rather quite volumetric and its processing is complicated.

Delay of data delivery resulted in failure to hold tenders in real time mode and traders were not able to trade. Thus, FIX-protocol had to be optimized.

In 2006 the first version of FAST-protocol (FAST 1.0) was issued. FAST (FIX Adapted for STreaming) protocol – is a technology, directed at optimization of data presentation in the net. It is used for providing of high productive capacity and low delay during message transmission between financial systems by means of messages compression with the help of different means [3 – 4].

FAST-protocol uses different technologies for data compression:

- data coding of messages, namely usage of samples. Samples determine data that transmitted and their sequence, that allows to exclude transmission of tags, identifying the fields.
- Coding of the level of FIX-message fields. FAST-protocol divides field into different types,

allows to avoid generally, or ensure partial transition of appropriate data;

– volume limitation according to type of data that being transmitted.

Usage of Stop Bit. Coding of symbol SOH divides FIX-message fields, in values of the most coding data [5].

FAST-protocol allows to ensure considerable data compression during their transmission, but another problem appears – additional processes during message transportation, namely coding and decoding. While inefficient realization of functionality of these processes, time on their fulfillment will be significant, therefore usage of FAST may be useless.

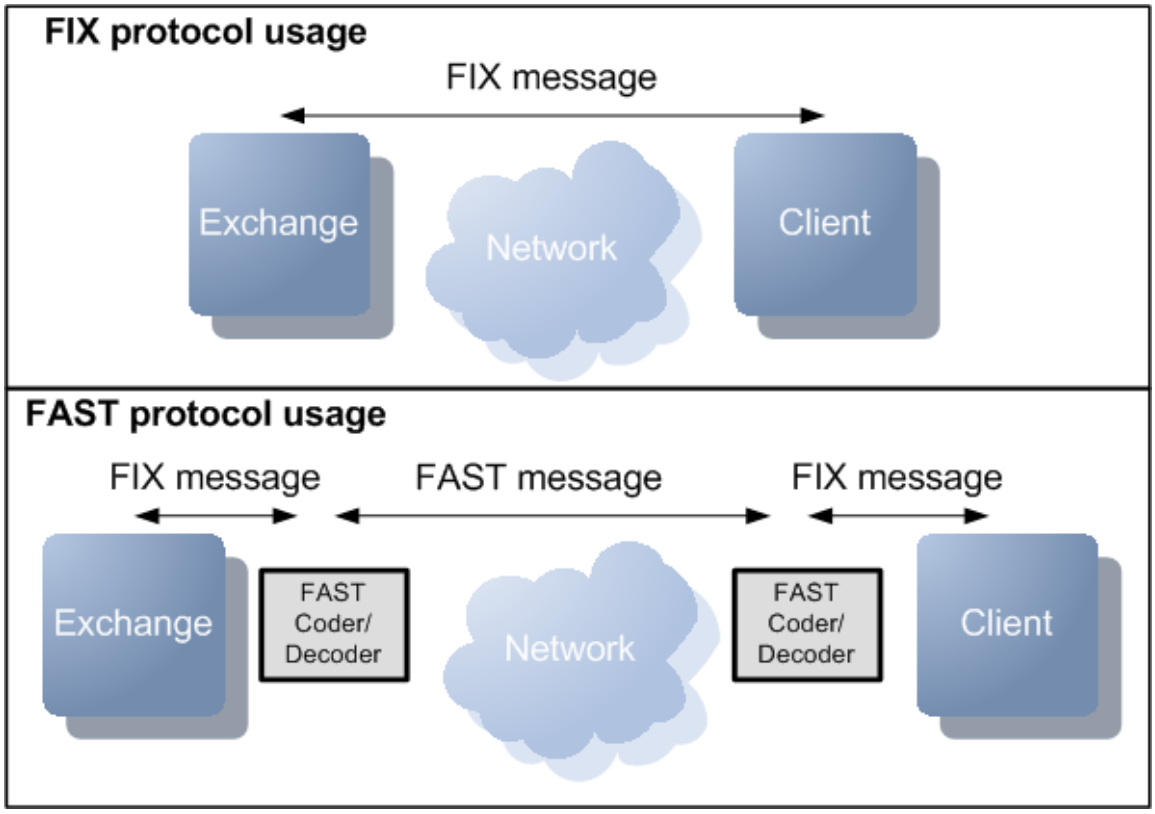

Fig. 1. Comparison of FIX- and FAST- protocol usage during message transition

Modern electronic system for internet-trading ensure creation of commercial operations with graphic interface, therefore all financial data are submitted to trader in pure form. On program level it means that after decoding process the system processes FIX-messages and transmits corresponding values for displaying it for user.

Thus, complete system, that ensures data transmission from exchange server to graphical displaying for client, requests appearance of several processes including; data reading from exchange information channel, data decoding, messages processing, transmission of corresponding values to the client.

Such electronic trade in financial markets is carried out in real time mode, delay during data transmission should be low, thus importance of time minimization for fulfillment of above-mentioned processes is obvious.

The objective of the paper is improvement of the system throughput during financial data transmission from exchange server to obtaining of corresponding values by the client.

Materials and results of research. Nowadays there exist known methods of time optimization , necessary to fulfillment of above mentioned processes. Let us carry out detailed analysis and realize by programme each of the methods.

1. Paralleling of streams. That is, programme code realization with the possibility of different

tasks execution in different streams. This approach ensures asynchronous functioning of several processes simultaneously, that, in its turn, increases rate of code fulfillment several times, depending on the number of created streams.

In order to increase the rate of tasks realization, it is expedient to realize the programming code so that simultaneously not more streams be operational than the number of processor kirnals are available in the system. In other case, tasks will be executed according to queue, that corresponds to synchronous execution, consequently it makes no sense to use parallel streams.

As for system, being elaborated, it is expedient to use four streams, that will realize such functionality as:

- Reading of coded data (FAST-message) in information channel;
- message decoding in FIX format;
- FIX-message processing;
- Transmission of corresponding data to one or more clients.

It is obvious, that usage for above considered system of paralleling approach of streams increases programme rate, consequently decreases delay during data transmission from exchange server 4 times (for 4 nuclear processor).

2. Factory of objects. For programmes the main goals of which are data transmission, creation of new objects is critical moment, as it favors memory allocation, that , in its turn, is labour-consuming process, which slows down the whole system. Therefore, it is reasonable to create all necessary objects during system initialization in order to avoid unnecessary delays, that is not always possible in practice.

It is necessary to create new object of the FIX-message during each new data arrival fro exchange server for considered system. This object will save coded data of FAST-message for its further processing. Object factory allows to get rid of periodicity of new object creation. Realization of this functionality is realized in such a way:

- certain number of FIX objects are being created during system initialization, that will be used as required;
- after the usage of all objects, created during initialization, new objects will be created during further inquires to the factory.

It is obvious that this approach in any case has advantages, i.e. influences the reduction of time during data transmission to client. But the given value is in direct proportion to number of messages, transmitted from exchange server and the size, correspondingly, of the objects factory itself, i.e. the number of objects, to be created during initialization.

Let us carry out testing of factory extracting 1 million.of new FIX objects:

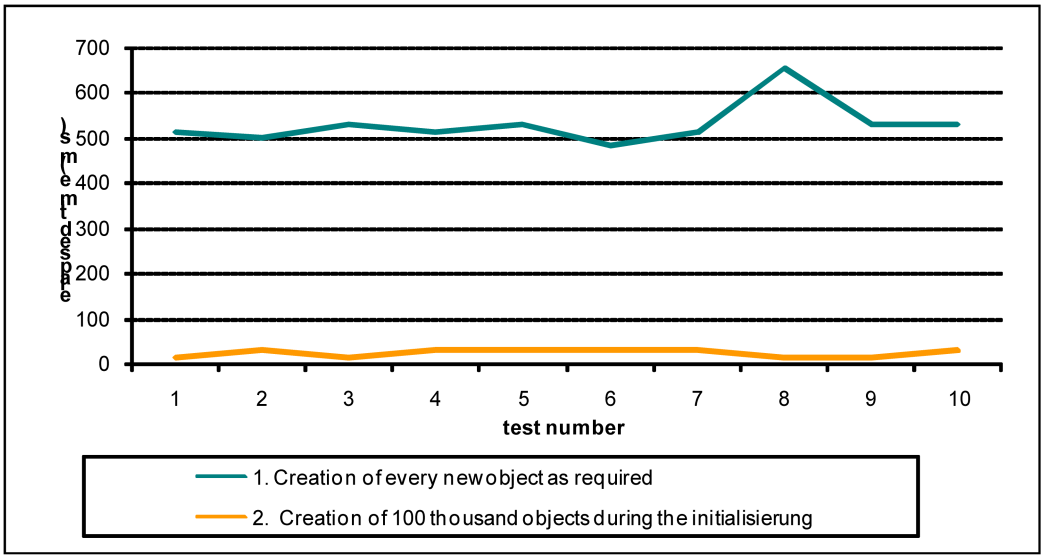

Fig. 2. Analysis of objects factory application

Results of test are presented in Fig. 2, show that time spent on creation of new object every time as required, is greater (approximately 20 times) than the time that approaches zero during the usage of object factory.

We are able to determine any size of factory, i. e. number of objects to be created during system initialization, that willinfluence only main memory.

3. Direct work with bytes arrays and applying bit-wise operations. Since all digital information in computer systems is presented in the form of bytes, it is obvious, that work directly with them requires the least time losses in comparison with bytes reducing to these or other types of data, that are more suitable for work.

As we have a task to increase the throughput of transmission system, i.e. decrease time needed for message processing, it is expedient, if possible, to ensure programme code operation directly with the bytes. Let us apply such an approach while decoding input data, i.e.FASR-messages,arriving from exchange server in the form of bytes arrays:

1)
```
offset=0;
while (offset < buffer.length-1) {
   value = (value << 7) | (buffer[offset++]);
}
value = (value << 7) | (buffer[offset++] & Byte.MAX_VALUE);
```

2)
```
Integer intDecodedVal = Integer.parseInt(strBuffer);
StringBuilder intDecBinSb = new StringBuilder(Integer.toBinaryString(intDecodedVal));
for(int i = intDecBinSb.length()-8; i >=0; i-=8) {
   intDecBinSb.deleteCharAt(i);
}
int value = Integer.parseInt(intDecBinSb.toString(), 2);
```

Direct work with bytes arrays in the example 1 allows to apply bit-wise operations, which, taken together, provide considerable advantages while usage (approximately 10 times as compared with the example 2).

Since it is quite obvious that program code of the example 2 includes redundant operations, such as byte conversion into binary code, that is redundant as compared with the example 1 an creation of new objects, as it was described in point 2.

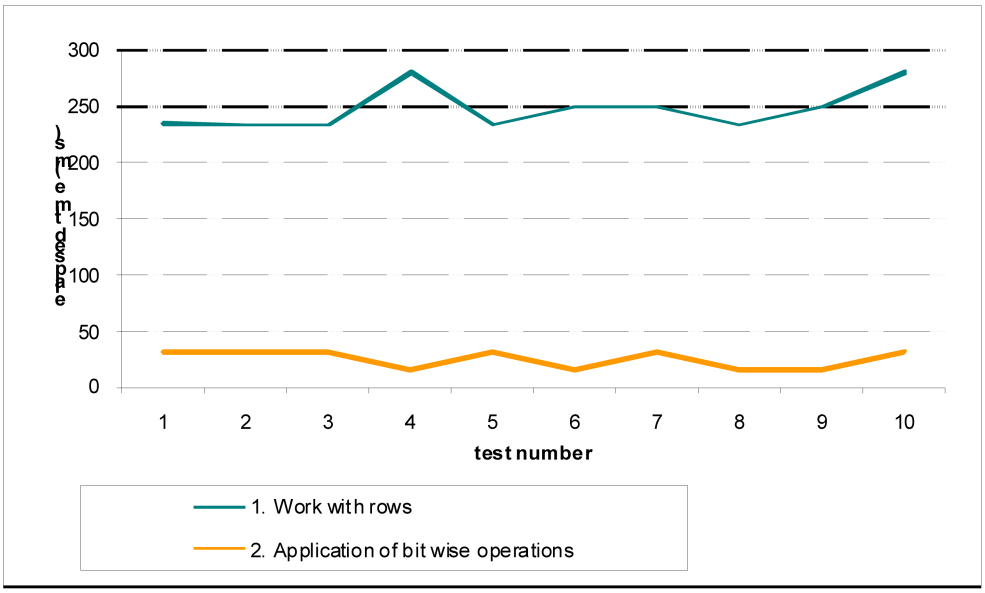

Fig. 3. Analysis of bit wise operations application

4. Universality dispossession of program code. It is obvious that code universality substantially reduces its volume, but at the same time, decreases its operation rate. For the majority of programmes, the given fact is not of primary importance, that is why, it is not paid much attention to. But for programming systems intended for message transmission, where data stream is high and main evaluation criterion is transmission rate, code universality can considerably influence the time of information processing.

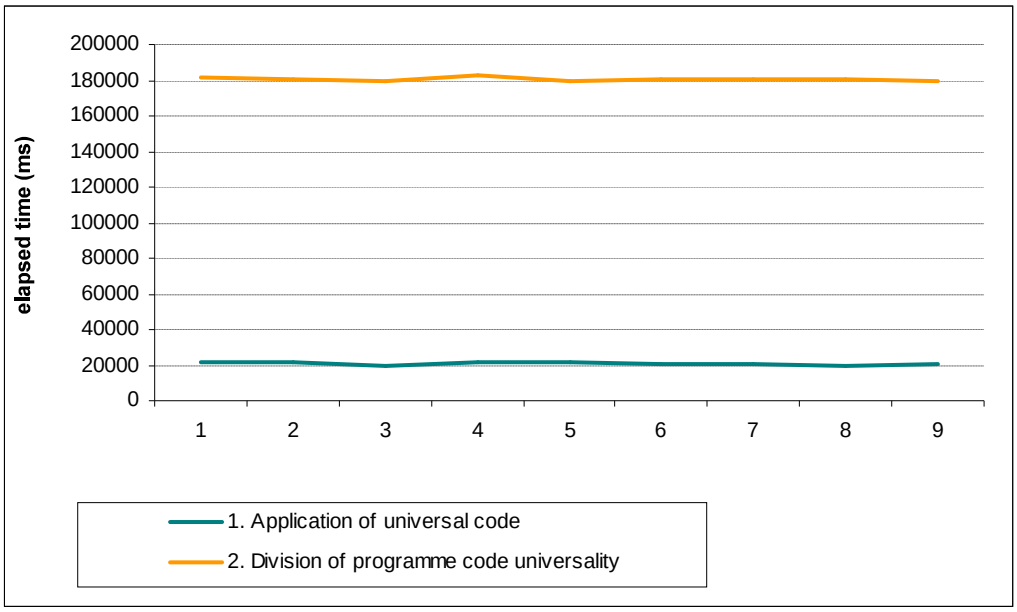

Fig. 4. Analysis of code universality application

In the developed system, in order to reach the most efficient result, it is necessary to provide the division of the total universal functionality. We will perform the given approach for the process of messages decoding, as FAST – protocol provides the distribution of all data into categories, which are defined in the following way:

1. Fields operators FAST-protocol describes 7 types of field operators, that determine the sequence of actions to be applied to various fields of FIX-message
2. Nullable fields. For such fields, decoded value “0” for numerical types of data or “” for line type of data, means transmission of “null” value, that has various application regarding other field parameters.

3. Data types. FAST-protocol describes 6 types of data, requiring corresponding rule, applied for their decoding.

Thus, having divided the given functionality, we obtain $7 \cdot 2 \cdot 6 = 84$ classes, which define rather large volume, but at the same time considerable gain of time (approximately 20 times, as shown in Fig. 4).

Data obtained as a result of the research, reflect the results of improvement of existing approaches to realization of FAST-coding/decoding process, carried out under the order of jettekfix.com company.

Conclusions. The given paper considers problems, actual while transmission of large volumes of data in the systems of e-trade on financial markets, modern approaches to the solution of these problems are optimized, the analysis of their realization is performed. As a result of research, carried out, it has been proved that the application of the suggested techniques allows to increase the throughput of data transmission system form exchange server approximately 4 times, and, in particular, FAST-messages decoding process-50times.

REFERENCES

1. FIX protocol [Електронний ресурс] // Режим доступу: http://ru.wikipedia.org/wiki/Financial_Information_eXchange.

2. Lamoureux Robert. Financial information exchange protocol / Robert Lamoureux // FIX Protocol Ltd. – 2001. – 280 p.

3. FAST protocol [Електронний ресурс] // Режим доступу: http://en.wikipedia.org/wiki/FAST_protocol.

4. Rosenborg David. FAST Protocol Technical Overview. / David Rosenborg. London: FIX Protocol Ltd. – 2006. – 10 p.

5. Rosenborg David. FAST Specification / David Rosenborg. London: FIX Protocol Ltd. – 2006. – 45 p.

Volodymyr Kotsubynskiy – Cand. Sc. (Eng.), Assistant Professor with the Department of Automation and Information – Measuring Engineering.

Roman Holovaha – Student, Department of Automation and Information – Measuring Engineering.

Vinnytsia National Technical University.