UDC 004.5

The implementation of technology of multi-user client-server applications for systems of decision making support

Sergei Chernyi

D.Sc. in engineering
Kerch State Marine Technological University

Abstract
RFC6455support, JSON in JavaScript format have been analyzed and studied. The model of the formation of an expert group for decision-making support has been proposed. The process of interacting of software modules is described from the perspective of the processes of coordination of the system.
Key words: PROCESS, CLIENT-SERVER, MULTI-USER, MATHEMATICAL STRUCTURE, PING, WEBSOCKETS, ARCHITECTURE, SOFTWARE, JSON, TCP

Nowadays, multi-user client-server applications are sufficiently adaptive and dynamic field of software. Multi-user applications of different platforms are the most complex examples of such applications because they combine the complex elements of the mathematical structure, elements of apparatus of logic with real time that allows to represent the region of platform of visual solutions for far remote users.

There is a problem related to the fact that all processes take place in a network that is a complex and unstable system, and the slightest increase in ping (a signal on the network) leads to a sharp deterioration in the quality of visualization of the flow of graphical representation of the application that is not so noticeable even in IPTV and video conferencing. Therefore, the technologies used to optimize network delays, are of the greatest interest in the development of such applications.

The aim of the article is to study the problems and peculiarities of a multi-user client-server interaction, issues of synchronization of clients to ensure the most similar status display of the platform with minimal network logs to ensure the efficiency of application of the system for decision making support of the expert group. It is also necessary to carry out such analysis as: client-server architecture; WebSockets technologies; existing software systems that perform similar tasks; development of architecture of client-server application; development of client-server application using the WebSockets technology.

The object of research is multi-user client-server application that allows you to manage the team of experts of the server, perform them on the server and return the result of performing back to all connected clients.

Under the concept WebSocket++ we understand the header library for programming language C++ with open source code that supports the standard RFC6455 "Protocol WebSocket". This library allows you to integrate the client and server functionality of WebSocket into software modules C++ worked out by user and at the same time, the library uses interchangeable transport network modules, stream-based input-output C++ and Boost Asio.

Key features of the library:

- the full support RFC6455;
- the interface based on system of events;
- the security support WebScoket, IPv6, and proxies;
- the flexible management of dependencies (C++11 or Boost);

- the replaceable network transport modules (input-output streams or Boost Asio);
- the portability and cross-platform.

The JSON technology (data interchange format is a textual format based on JavaScript programming language and is often used with this very language; JSON is easy for people to read) is of great importance for decision making support systems (DSS)

Being originated from JavaScript, the format is not dependent on the development environment and is almost universal in use with any programming language. For many languages there is a complete code for the creation and processing of data in JSON format.

PicoJSON Library is a simple JSON encoder and decoder for the C++programming language.

Peculiarities of the library:

- only header files are connected for use;
- the absence of external dependencies (only STL);
- the decoding of the data in STL entity.

WebSocket is the Protocol of full duplex communication over a TCP connection, intended for the exchange of messages between the browser and the web server in real time.

To establish a WebSocket connection the client and the server use a Protocol similar to HTTP. The client creates a special HTTP request to which the server responds in a certain way.

The WebSocket Protocol (RFC 6455) is intended for solution of any problems, and removal of limitations of the exchange of data between browser and server. It allows you to send any data on any domain, safe and almost without unnecessary network traffic.

WebSoscket Protocol is built on top of TCP, because TCP is the Protocol with the principles of reliable connection and the connection is established between two computers, and then data are sent between them just as if the information is written to a file on one computer can be read from the same file on the other computer. The connection is reliable and consistent: that is, all the information to be sent is guaranteed to reach the recipient in the same order in which it was sent. A TCP connection can be considered as a continuous stream of data because the Protocol itself takes care of disaggregated of data into packets and sending them over the network.

To extend functionality one can use a collection of C++ libraries - Boost, which are freely distributed under the Boost Software License along with the source code. The Boost library is significantly oriented on research and extensibility (meta-programming and generic programming with the use of templates).

Architecture Pear-To-Pear (P2P, peer-to-peer, or decentralized peer-to-peer) is widely applicable too, architecture is a computer network based on the equality of the participants. In this network (often) there are almost no dedicated servers, and each node is both a client and performs the functions of the server. Unlike client-server architecture, this organization allows us to keep working capacity of the network with any number and any combination of the available nodes. Today, the third generation of P2P networks is characterized by the absence of a dedicated centralized servers and fundamentally new search algorithm that is based on the key concept of a distributed hash table which is supported by the network members. Distributed hash tables help to solve the problem of scaling, by choosing different sites to index some values of the hash function (that is used to index files), allowing you to search for any file on the network quickly and efficiently.
The advantages of a decentralized system can consist in much lower cost in high load projects in comparison with the centralized systems, in increased reliability of preserving of the system health with a large number of participants. While writing a client-server application, you need to determine the data transfer Protocol. For software module, made in the framework of this paper we have defined 6 kinds of events: global synchronization, registration of a new expert for DSS, broadcast renewal of experts' dialogue, the account of activity of the expert, update of the position of an expert and the output of the expert from examination.

It will be appropriate to show the coordination of the group of experts for the DSS through the global event of synchronization (event for one expert (peer assessment) asks the server about complete current state of examination). Along with global synchronization there is also an event update of the examination process, which is distributed to all experts from the server and reports about all changes that took place since the last synchronization. The event of registration of a new expert happens when the expert opens the page in a browser for the first time. Before closing the page the client sends a message about his logout to the server. If it is impossible to send a message of logout, the server itself will determine that a player is not responding, and remove him from the game. Both events are sent to all other players in the nearest update of the game world. The last two events are the movement of the player

and update of his status. The first event means that the client sends his message to the server, that message contains the information about the buttons pressed by the user, the server in response to this message will send the update status of the player with information about his new coordinates.

All communication between the client and server is encoded into JSON format that allows you to turn data into a structure of a programming language (arrays and objects to JavaScript, std::vector, std::map for C++) quickly and it is easy to work with them. When messages are being encorded into JSON format we obtain significant savings in network traffic, for example, compared with XML [1].

After the announcement of the communication Protocol between the client and the server, you first need to program the WebSocket connection. The client code is quite simple, the standard object of JavaScript with the same name of WebSocket is used. This object has only one public method - the send function, that sends messages from the client to the server over an already established connection. To create the connection, you need to create an instance of the WebSocket passing the server address to constructor. The connection object has three events that you can put handlers, - user-defined functions, on. The first two types of events are invoked once for all time of the operation of the object, the first onOpen event appears after correct installation of the message. The second – onClose event occurs during the connection break, the code of the closed connection is passed to the handler function. The third event is the appearance of new messages from the server, it is called onMessage. Every time when the server sends something this function is the first to be performed, it decodes the message from JSON into JavaScript object and distributes event further on the client.

To work with WebSocket on the server one can use a websocketpp library, which is an add-in on another library - Boost ASIO. The syntax of the library work is simple: it is enough to call some few methods to create a listening server. After creating an instance of the websocketpp object, which in its turn gets class of the server from Boost ASIO, you need to initialize it and specify the port that you want to listen to. Then, as well as in the client WebSocket you can specify three event handlers: open, close, and message. All three functions take a reference to an object-identifier of a user. The class - server of the Boost ASIO itself is actually a container for other connections. For the work of application you need to install handlers for two functions: message and close. The first one will perform operations similar with a customer - decode messages in JSON format and send them further on the program in accordance with the instructions. To decode JSON on the server we can use the picojson library, because it is made as an extension of the standard types of STL.

After describing of the communication Protocol of client and server and writing of the connection code, the system that processes the actions of pressing the buttons by the user was implemented. In the Appendix attached to this work, there is no need to press a few keys simultaneously, so, with every further press, the information about all previous presses is deleted. Every time when an input event occurs, the client packages information about a pressed key and it is sent to the server.

On the server there is a structure of "expertise", which keeps a state of examination. This is the most important and largest structure, which consists of other structures and base types.
Key fields of the structure of "expertise" is information about the current process status, time of start of the examination, the last action time, the nature of the process, an instance of the connection and the list of experts at the present moment.

The structure of "expert" is a very important structure.

Information about the activity of the expert is in the structure of the expert to optimize the algorithm of action validation by reducing of the array iterate.

For all objects on the server, functions of data serialization in JSON format were created, as well as functions that combine the game objects for different types of game events in a special way.
The server contains the same expert action-event as the client. A set of instructions is performed when you receive an event and if the instructions are provided with the return value, the server will forward the result of their work back to the client.
If the expert leaves the server, he first sends out the message that he leaves the examination so that the server could inform other experts and break the connection with them correctly. If the examiner faces some problem and the message does not come, the server will create a message stating that the expert has left the examination, will remove it from the processing and notify other experts. This is done by means of the WebSocket built-in ping-pong system that all the time checks the connection between the client and the server and in case of

problems on the communication channel closes the connection with an error code.

The present work implements the technology of interpolation of examination workspace - latency of networking is great and not stable, it is impossible to the state of examination to equal time, so while waiting for synchronization the client deals with interpolation. The main task of interpolation is to give the player the illusion that the expert examines without delay. If workspace is synchronized without interpolation, most objects will change the location during synchronization. In the development of client-server Appendix the ways of its implementation have been identified, a software system of command management has been developed. The recommendations for use and for further development and improvement of the developed software system were given [2, 3, 4, 5].

Global views of DM about the choice of alternatives are formulated in the form of global criterion, and the solution of the multicriteria problem is reduced to construction of the composition, where

$M_1 : \Theta(U^n) \to \Theta(Q^m)$, $U^n = U_1 \times ... \times U_n$, $Q^m = Q_1 \times ... \times Q_m$,

$M_2 : \Theta(Q^m) \to \Theta(Q)$,

Q_i, Q - many of the characteristic values of local and global criteria, respectively. M_1 and M_2 are formed on the basis of the statements like: "if the characteristics, U_1..., U_n, characterizing alternative U_i are evaluated with terms t_{1i}, ..., t_{ni}, the alternative satisfies the j-th criterion with assessment of tn+j,i".

M_1 and M_2 are described by sets:

$$M_1 = \left\{ \left(t_{1i}, ..., t_{ni}, t_{n+j,i} \right) \mid n+1 \le j \le n+k, i = 1, m_1 \right\},$$

$$M_1 = \left\{ \left(t_{n+1i}, ..., t_{n+ki}, t_{n+k+1,i} \right) \mid \mathrm{i} = 1, ..., \mathrm{m}_2 \right\}.$$

The degree of satisfaction of the global criteria for alternatives $u^i \in U^n$ is calculated as follows:

$$w(u^i) = \bigcup_{t^p \in M_2} \left(\bigcup_{t^e \in M_1} u^i \circ t^e \right) \circ t^p.$$

In the learning process are specified for the assessment of global and local criteria on the basis of comparison of the selected person making the decision maker (DM) alternatives R'' from many charged $R' \supset R''$. M is changed by some $\tilde{M}$, confirming the appropriate selection:

$\tilde{w}(u^i) \prec \tilde{w}(u^j)$ for $u^i \in R'$, $u^j \in R' \setminus R''$.

Training is carried out in two stages: the formation of generalized descriptions preferences of decision-makers; modification of M with different preferences of decision makers (DM) with the order of assessments $\underline{w(u)}$. The second stage involves the following: generation of admissible sets of indicators of assessments; definition of relations preferences of pair generated alternatives; the selection of the $M = M_1 \cup M_2$ sets, which are not subject to adjustment; adjustment of the assessments according to criteria.

The reference version of model of an expert for the EEG is introduced as a tuple U^t (1) for further formation of criteria of selection: for further formation of selection criteria:

$$U^t = \left\langle \left\langle u^t_{lt}; u^t_{stg}; u^t_{pub}; u^t_{vst}; u^t_{dl}; u^t_{us}; u^t_{zv}; u^t_{pt} \right\rangle; \left\langle u^t_{sam} \right\rangle; \left\langle u^t_{usp} \right\rangle; \left\langle u^t_{imn} \right\rangle \right\rangle, \; t = \overline{1,n} \qquad (1)$$

or as an associative convolution:

$$U^t = \left\langle \begin{matrix} k_1 u^t_{lt} + k_2 u^t_{stg} + k_3 u^t_{pub} + k_4 u^t_{vst} + k_5 u^t_{dl} + k_6 u^t_{us} + k_7 u^t_{zv} + \\ + k_8 u^t_{pt} + k_9 u^t_{sam} + k_{10} u^t_{usp} + k_{11} u^t_{imn} \end{matrix} \right\rangle, \qquad (2)$$

where for t-expert; n – the number of experts in the database; u^t_{lt} – the age of the expert; u^t_{stg} – work experience in "problem area"; u^t_{pub} – number of publications on the issue; u^t_{vst} – the number of speeches related to the issue of solving the problem; u^t_{dl} – occupied position; u^t_{us} – scientific degree; u^t_{zv} – scientific title; u^t_{pt} – number of patents, certificates (related to a problem to solve); u^t_{sam} – self-assessment of competence; u^t_{usp} – the number of successfully implemented projects; u^t_{imn} – characteristics of the expert by other experts; $k_1, k_2, \ldots, k_{11}$ – weight coefficients.

$$u_{lt}^{t}=\left\langle u_{lt}^{t_i}\right\rangle;\ u_{lt}^{t}=\sum_i k_{1,i}u_{lt}^{t_i};\ u_{stg}^{t}=\left\langle u_{stg}^{t_i}\right\rangle;\ u_{stg}^{t}=\sum_i k_{2,i}u_{stg}^{t_i};\ u_{pub}^{t}=\left\langle u_{pub}^{t_i}\right\rangle;\ u_{pub}^{t}=\sum_i k_{3,i}u_{pub}^{t_i}$$

$$;\ u_{vst}^{t}=\left\langle u_{vst}^{t_i}\right\rangle;\ u_{vst}^{t}=\sum_i k_{4,i}u_{vst}^{t_i};\ u_{dl}^{t}=\left\langle u_{dl}^{t_i}\right\rangle;\ u_{dl}^{t}=\sum_i k_{5,i}u_{dl}^{t_i};\ u_{us}^{t}=\left\langle u_{us}^{t_i}\right\rangle;\ u_{us}^{t}=\sum_i k_{6,i}u_{us}^{t_i};$$

$$u_{zv}^{t}=\left\langle u_{zv}^{t_i}\right\rangle;\ u_{zv}^{t}=\sum_i k_{7,i}u_{zv}^{t_i};\ u_{pt}^{t}=\left\langle u_{pt}^{t_i}\right\rangle;\ u_{pt}^{t}=\sum_i k_{8,i}u_{pt}^{t_i};\ u_{sam}^{t}=\left\langle u_{sam}^{t_j}\right\rangle;\ u_{sam}^{t}=\sum_j k_{9,j}u_{sam}^{t_j};$$

$$u_{usp}^{t}=\left\langle u_{usp}^{t_q}\right\rangle;\ u_{usp}^{t}=\sum_q k_{10,q}u_{usp}^{t_q};\ u_{imn}^{t}=\left\langle u_{imn}^{t_x}\right\rangle;\ u_{imn}^{t}=\sum_x k_{11,x}u_{imn}^{t_x},$$

where $k_{1,i},\ldots,k_{8,i},k_{9,j},k_{10,q},k_{11,x}$ – weight coefficients.

The reference model view option for the selection of the expert for a group (1), a necessary condition is the limit on the weights, and integral value (2) of the reference model of the expert should be close to 1. Complex estimation (2) is a simplified version of assessment of the qualities of selected experts for the expert teams and their formation for the project team. Let us describe (2) parts:

$u_{lt}^{t}=\left\langle u_{lt}^{t}\right\rangle$ or $u_{lt}^{t'}=\left\langle k_{1,1}u_{lt}^{t}\right\rangle$,

$u_{stg}^{t}=\left\langle u_{sm};u_{sd}\right\rangle$ or $u_{stg}^{t'}=\left\langle k_{2,1}u_{sm}+k_{2,2}u_{sd}\right\rangle$,

where u_{sm} – in a related field; u_{sd} – in this problem area.

$u_{pub}^{t}=\left\langle u_{z3};u_{z5};u_{zsp};u_{zv};u_{sb}\right\rangle$ or

$u_{pub}^{t'}=\left\langle k_{3,1}u_{z3}+k_{3,2}u_{z5}+k_{3,3}u_{zsp}+k_{3,4}u_{zv}+k_{3,5}u_{sb}\right\rangle$

, where u_{z3} – for the last three years; u_{z5} – for the last five years; u_{zsp} – publications in specialized magazines; u_{zv} – total publications on the issue; u_{sb} – publications in specialized and scientometric databases.

$u_{vst}^{t}=\left\langle u_{vs};u_{vz};u_{vsp}\right\rangle$or

$u_{vst}^{t'}=\left\langle k_{4,1}u_{vs}+k_{4,2}u_{vz}+k_{4,3}u_{vsp}\right\rangle$,

where u_{vs} – reports on the problem areas in the country; u_{vz} – performances at international level; u_{vsp} – membership and participation in specialized seminars, conferences, symposia of international importance

$u_{dl}^{t}=\left\langle u_{zdl};u_{dlz}\right\rangle$ or $u_{dl}^{t'}=\left\langle k_{5,1}u_{zdl}+k_{k,2}u_{dlz}\right\rangle$,

where u_{zdl} – your position at the moment; u_{dlz} – the position which the expert has held previously.

$u_{us}^{t}=\left\langle u_{kn};u_{dn}\right\rangle$ or $u_{us}^{t'}=\left\langle k_{6,1}u_{kn}+k_{6,2}u_{dn}\right\rangle$,

where u_{kn} – the title of candidate of Sciences; u_{dn} – doctor of Sciences.

$u_{zv}^{t}=\left\langle u_{zd};u_{zn};u_{zpf};u_{pz}\right\rangle$ or

$u_{zv}^{t'}=\left\langle k_{7,1}u_{zd}+k_{7,2}u_{zn}+k_{7,3}u_{zpf}+k_{7,4}u_{pz}\right\rangle$,

where u_{zd} – the diploma of the senior lecturer; u_{zn} – diploma of the scientific employee; u_{zdt} – the diploma of Professor; u_{pz} – the diploma or an honorary title.

$u_{pt}^{t}=\left\langle u_{pv};u_{pn};u_{ps};u_{pd}\right\rangle$ or

$u_{pt}^{t'}=\left\langle k_{8,1}u_{pv}+k_{8,2}u_{pn}+k_{8,3}u_{ps}+k_{8,4}u_{pd}\right\rangle$,

where u_{pv} – the number of patents within the country; u_{pn} – the number of patents of international importance; u_{ps} – the number of patents registered by an expert without co-authors; u_{pd} – the number of examinations carried out in the field of patents.

$u_{sam}^{t}=\left\langle u_{sk};u_{skp}\right\rangle$ or $u_{sam}^{t'}=\left\langle k_{9,1}u_{sk}+k_{9,2}u_{skp}\right\rangle$,

where u_{sk} – self - assessment on problem area; u_{skp} – the total self-assessment of competence.

$u_{usp}^{t}=\left\langle u_{usn};u_{usj};u_{usi},u_{usv}\right\rangle$or

$u_{usp}^{t'}=\left\langle k_{10,1}u_{usn}+k_{10,2}u_{usj}+k_{10,3}u_{usi}+k_{10,4}u_{usv}\right\rangle$,

where u_{usn} – no successfully completed projects; u_{usj} – the number of successful projects 10%–25% of the total number of all projects; u_{usi} – number of successful projects %–70% of the total number of all projects; u_{usv} – number of successful projects 71%–100% of the total number of all projects.

$u_{imn}^{t}=\left\langle u_{izp};u_{ikr};u_{iuk};u_{ilz};u_{iak}\right\rangle$ or

$u_{imn}^{t'}=\left\langle k_{11,1}u_{izp}+k_{11,2}u_{ikr}+k_{11,3}u_{iuk}+k_{11,4}u_{ilz}+k_{11,5}u_{iak}\right\rangle$

where u_{izp} – the assessment of the level of expert knowledge a; u_{ikr} – ability to team work; u_{iuk} – the level of conflicts; u_{ilz} – personal acquaintance with the expert; u_{iak} – characteristic of his personal data.

The result of this work is embodied in the development of a multi-user client-server Appendix in C++ and JavaScript languages, that possesses the following features:

- the program is divided into two independent parts: a client and a server
- the client sends managing commands to the server;
- the entire program cycle takes place on the server;
- the work is carried out by the expert group, the model of which is presented in the work, and the interaction is described by the coordinating nature of the procedure;
- the client receives only necessary relevant information.

The work contains the implementation of software complex of multiplayer client-server interaction, solves the issues of synchronization of playing status among clients, provides a more accurate display of the game world with the audience network lags.

References

1. Robert L., Ira D., Michael Mehlich. Re-engineering C++ Component Models Via Automatic Program Transformation. – Available at: http://www.semanticdesigns.com/Company/Publications/WCRE05.pdf.
2. Cherny S. (2011). Optimization of code structurization process. *Eastern-European Journal of Enterprise Technologies*. 2(50), p.p. 31-34
3. Chernyi S., Zhilenkov A. (2015). Analysis of complex structures of marine systems with attraction methods of neural systems. *Metallurgical and Mining Industry*, No. 1, p.p. 37–44
4. Zhilenkov A., Chernyi S. (2015). Investigation performance of marine equipment with specialized information technology. *Procedia Engineering.* Vol. 100, p.p. 1247–1252
5. Shchokin V., Tkachuk V. (2014). Automation agglomeration production on based application neuro-fuzzy regulation of lower level. *Metallurgical and Mining Industry*. No 6,p.p. 32-39.