Data acquisition in OpenSCADA

Contents table

Introduction

Data acquisition of the SCADA (Supervisory Control and Data Acquisition)-system is its integral part, which get data from sources of different type. The nature of data, which operates SCADA, is characterized by signals of basic value's types (integer, real, boolean and string). The signals vary over time and has their history, life. In the theory of technological processes (TP) under the signal it is meant the value of TP sensor in the ADC code, "raw" signal or in the real value. Signals can be combined into groups, which are often called parameters. For example, the developed data sources can provide the structures of parameters with the predefined set of related signals. In addition to the direct data acquisition in the function of this mechanism is also included the transfer of actions to control devices of TP; usually it is a gate valve, pumps and control valves. Taken together, this process is known as computer-process interface (CPI).

Sources of data are characterized by their great variety, which can be divided into three groups.

- Sources of "raw" data, providing the ADC code or levels of discrete signals, and also the sources which include simple processing. Usually, it is the modules of the allocated CPI or the simplest industrial programmable logic controllers (PLCs).
- Powerful industrial PLCs, which have significant computing power and the possibility of formation of complex parameters with different structure.
- Local or related data sources. For example, the CPI as expansion cards, and also the data of the hardware and software environment in which the system operates.

The variety of data sources has created a wide range of mechanisms to access them. Local data sources are different in application programming interface (API), and network sources, in their turn, in transport and protocol interaction level. In general, this has led to the fact that the addition of support for a new data source requires the creation of interface module or driver. Taking into account the great variety of sources, it is extremely expensive and actually impossible to cover the entire spectrum of the market of these devices. The situation is somewhat simplified with the network source due to the presence of the number of standard and free interaction protocols, but many sources still use their own protocols: private, commercial or protocols, tied to private mechanisms of the limited range of commercial operating systems (OS).

In terms of OpenSCADA system the following objects to serve the data acquisition mechanism are provided:

- Attribute - object of reflection of the signal data, it includes the current value with the type of signal and the history of changes of value;
- Parameter - object of the attributes' (signals') group with the structure corresponding to the characteristics of the separate data source;
- Controller - object of the separate data device. Typically, this is a separate CPI module or the devices of industrial PLC.

To account the features of different data acquisition devices, as well as the different mechanisms of interaction in the OpenSCADA the modular subsystem "Data acquisition" is provided. The module of the subsystem is the driver for interfacing with a data source of specific type. Each module can contain a configuration of several devices of this type in the form of "Controller" objects of OpenSCADA. The general scheme of objects of "Data acquisition" subsystem is shown in Figure 1.

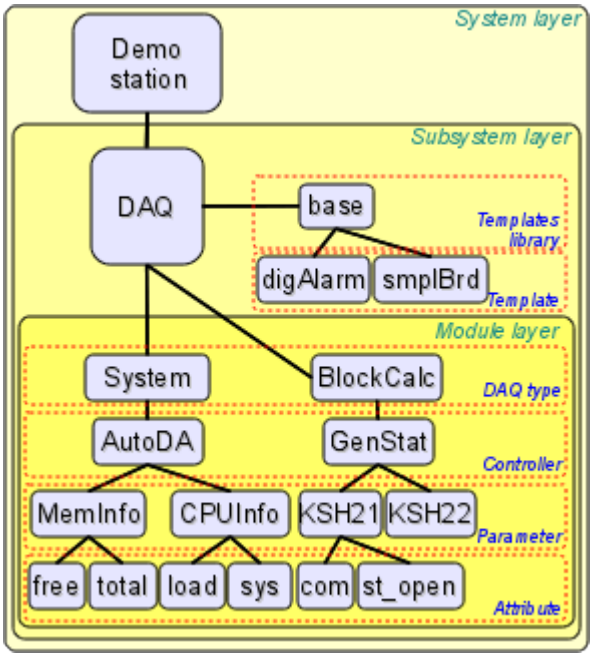

Fig. 1. The subsystem's "Data acquisition" scheme.

1. Data acquisition methods

Taking into account variety of the data sources, and also the ways of their possible interaction data acquisition methods can be divided to simple synchronous, simple asynchronous, package and passive ones.

To the examination of the mechanisms below the following objects will be involved:

- ObjectSCADA - any object of the SCADA-system, applying for the signal value, for example, archives and visualizers;
- DAQParamAttribute - attribute of the parameter of subsystem "Data acquisition" which is an intermediary for access to the value of the signal of data source;
- DAQParamAttributeArch - attribute's archive object;
- HardwarePLC - data source object, for example, modules of the allocated CPI or industrial PLC.

1.1. Simple synchronous acquisition mechanism

The mechanism is characterized by requests to the data source synchronously with the request to the attribute of parameter (Fig. 2). This mechanism is usually used when working with local sources of data, characterized by low latency, ie delay in response to the request. With this method you can get actual data directly with the request, but the time of the request of object will include the time for transportation and processing of the request by the data source.

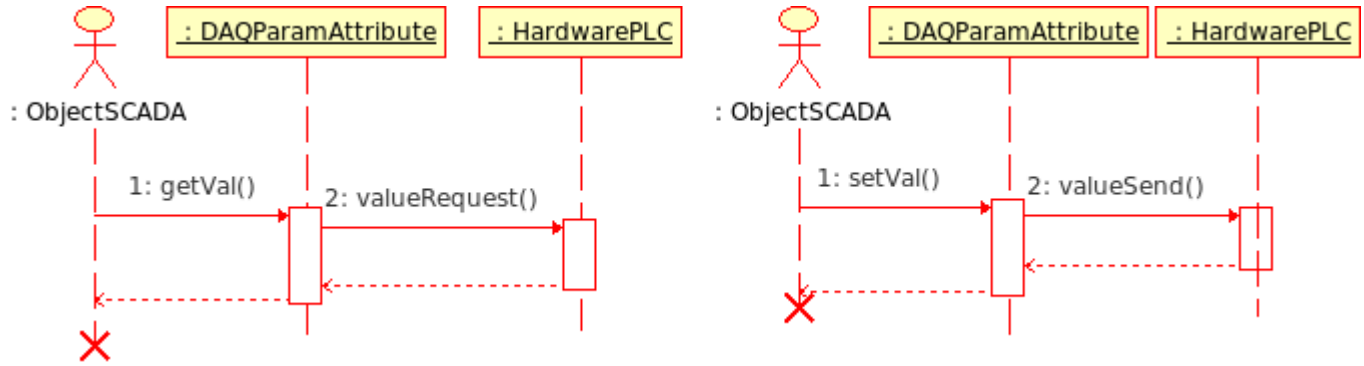

Fig. 2. Diagram of the sequence of interaction with the synchronous requests.

In accordance with the diagram above, we obtain the following sequence of requests for data acquisition and their transfer:

- object of the SCADA-system sends the value request to the object of attribute of the parameter DAQParamAttribute::getVal();
- object of the attribute of parameter, receiving the request, sends it to the data source HardwarePLC::valueRequest();
- source of data after processing the request returns the result;
- object of the attribute of parameter, receiving the result, returns its to the SCADA-system object.

In OpenSCADA this mechanism is implemented by the following modules of subsystem "Data acquisition".

- *ModBus* - module of access to data of the sources through the family of ModBus protocols. In the module the synchronous mode for recording data is implemented.
- *DiamondBoards* - module of the data access to the PC/104 card of Diamond Systems company. PC/104 boards are available on the ISA-bus, hence are local and available relatively quickly. When data acquisition is made not by interruption the access to the values of the ADC is synchronous. Recording mode of the DAC values always works synchronously.
- *DAQGate* - module of the reflection of the controller's objects of the remote OpenSCADA-stations on the local one. In the module the synchronous mode for recording data is implemented.
- *BlockCalc* - calculator in the language of block diagrams. The source of data for it is the custom block diagram. Attributes of parameters of the module synchronously address the inputs/outputs of the blocks of block scheme.

- *JavaLikeCalc* - calculator on the Java-like high level language. The source of data it supports is the user program on the Java-like language. Attributes of the parameter of module synchronously address the inputs/outputs of the user computing function.
- *LogicLev* - module of the logic-level parameters of data acquisition, see more about it in section 2. The source of data for this module are the other parameters of subsystem "Data acquisition" and the execution context of the parameters' template. Attributes of the parameters of module synchronously address the attributes of other parameters in the reflective mode of parameters of subsystem "Data acquisition", or the inputs/outputs of the execution context of the template when work under the template.

1.2. Simple asynchronous acquisition mechanism

The mechanism is characterized by requests to the data source, regardless of the request to the attribute of parameter (Fig. 3). Usually, requests to the source of the data are made periodically in the own inquiry task of the single controller and with the blocks of few signals. This request to the parameter's attribute returns the value obtained from the last connection session with the data source. This mechanism is usually used when working with remote (network) data sources, characterized by high latency, ie delay in the response to the request.

With this method it is possible to optimize the time resource spent on one signal, and thereby increase the maximum number of requested signals during the time interval of the inquiry.

As an example, lets examine an industrial PLC Siemens S7-315 during requesting him on the bus Profibus (1,5 Mbit/s). The average processing time of the MPI-request of this controller is 30 ms. If you use a synchronous mechanism for each signal, ie one request for each signal, then in one second we can get something about 33 signals. And if you apply an asynchronous mechanism, ie in the MPI-package to receive up to 220 bytes or 110 signals of integer type of 16-bit, then we can for one second get up to the 3630 signals. As you can see, the effectiveness of asynchronous mechanism in this case is 110 times, namely, the maximum capacity of MPI-package.

The disadvantage of asynchronous mechanism is that the request of the value of attribute of the parameter returns not actual at the time of request value, but value of the last session of the inquiry of the controller. However, taking into account that the source of data can be updated at intervals of ADC hardware limitations, and the sensors themselves may have certain restrictions on the reaction rate, the using of an asynchronous acquisition mechanism could have a serious grounds.

Application of asynchronous mechanism for recording the values to the PLC is a fairly rare fact, because recording of values usually involves impact of the operator on the TP. Operator on the fact rarely makes adjustments to the process, therefore, the recording can be performed synchronously. However, there are situations, such as managing of the TP by the regulator on SCADA-system, acting as a runtime of PLC.

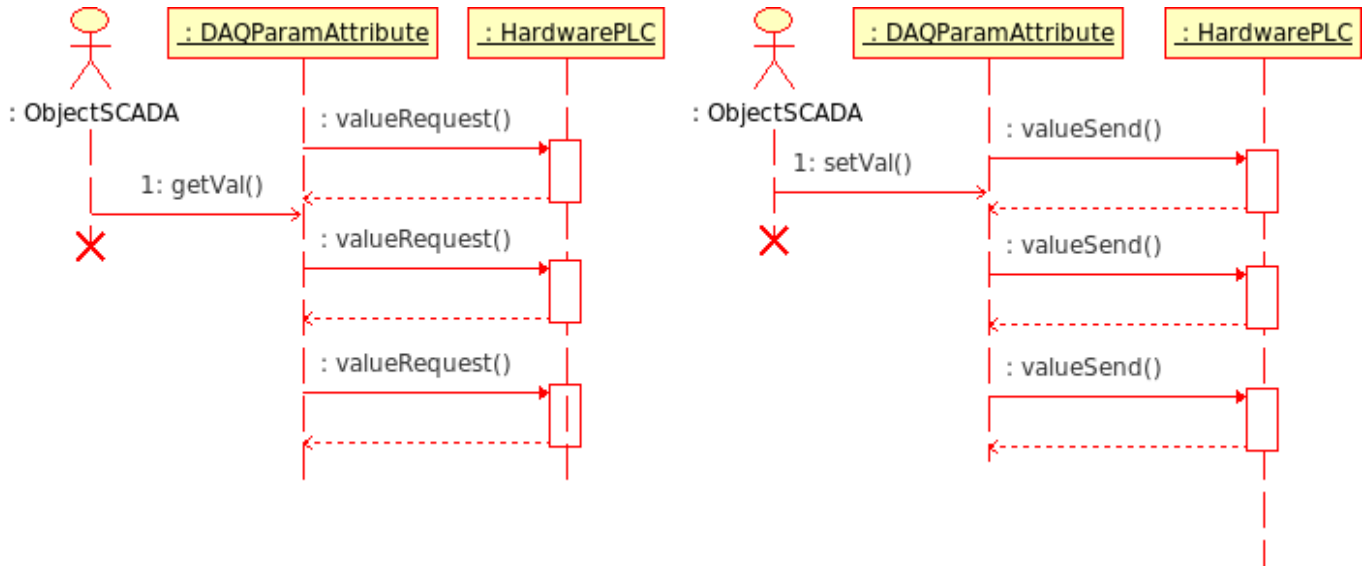

Fig. 3. Diagram of interaction sequence with asynchronous requests.

In accordance with the diagram above, we obtain the following picture:

- object of the attribute of parameter (or the parent object of the controller) performs the periodic requests HardwarePLC::valueRequest() to get the value of a signal or group of signals;
- received signal values stored in the objects of parameter's attributes locally;
- an object of SCADA-system sends the value request to the object of parameter's attribute DAQParamAttribute::getVal() and gets locally saved value of the previous session of the inquiry of data source.

In OpenSCADA this mechanism is implemented by the following modules of subsystem "Data acquisition".

- *Siemens* - module of access to the data of Siemens controllers of S7 series. In this module an asynchronous mode is implemented as for reading data and for recording (optional) to the PLC.
- *ModBus* - module of access to data sources through the family of ModBus protocols. In the module an asynchronous mode of reading data is implemented.
- *SNMP* - module of access to the data of the network devices through the Simple Network Management Protocol. In the module an asynchronous mode of reading data is implemented.
- *System* - module of access to the data of the execution area of OpenSCADA. In the module an asynchronous mode of reading data is implemented.
- *DAQGate* - module of the reflection of controller's objects of the remote OpenSCADA-stations on the local one. In the module an asynchronous mode of reading data is implemented.

1.3. Package acquisition mechanism

Package data acquisition mechanism is characterized by the acquisition of data for each signal by the packet that includes the history of its changes. Ie per one session of data inquiry we obtain multiple values of history of the signal. Package mechanism works in conjunction with synchronous and asynchronous mechanisms.

In the case of working with the synchronous mechanism the actual transfer of the archive of data source for operational work in the system is done (Fig. 2). As the simple synchronous mechanism, it is desirable to apply only to low-latency data sources or to the sources whose work is a session type, for example, in the commercial account to read the values of the counters.

When working in conjunction with an asynchronous mechanism the history of the received signals is usually placed directly in the archives (Fig. 4), and the current value of the parameter's attribute is set to last value of the package. This combination is effective during the acquisition of the fast data or during the synchronization of the archives after the loss of connection to the remote data source.

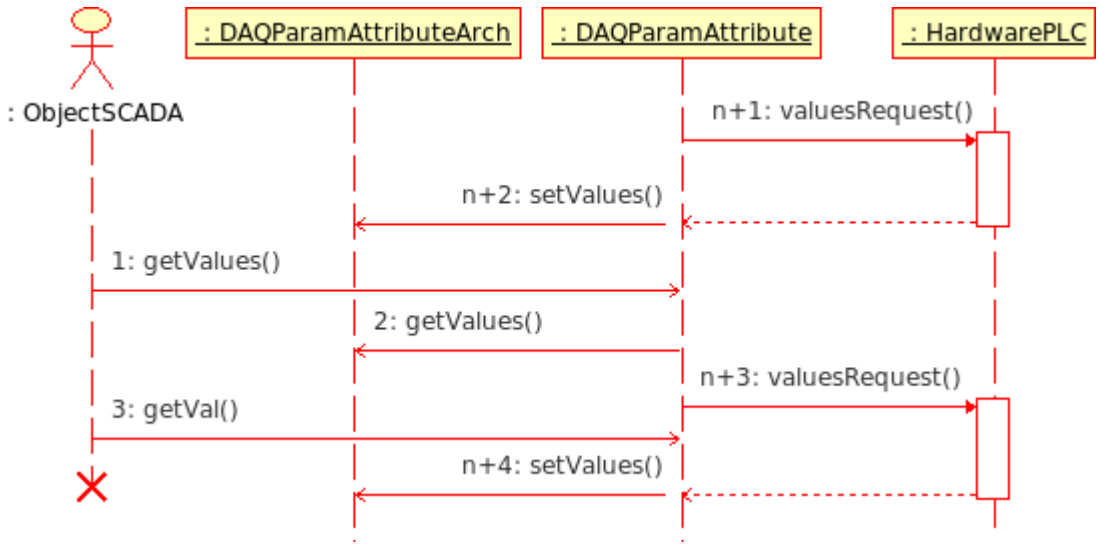

Fig. 4. Diagram of interaction sequence with the asynchronous requests of the package mechanism.

In accordance with the diagram above, we obtain the following behavior of the package mechanism for asynchronous requests:

- object of the attribute of parameter (or the parent object of the controller) performs the periodic

requests HardwarePLC::valueRequest() to get the value's packages of a signal or group of signals;
- received value's packages of signal are placed in the archive by the request DAQParamAttributeArch::setValues(), and the last value of the packages is located in the objects of parameters' attributes;
- object of SCADA-system sends the request of the archive's fragment to the object of parameter's attribute DAQParamAttribute::getValues(), and he relays the request to the archive DAQParamAttributeArch::getValues(). As the result the fragment of the archive, available after the previous session of the inquiry of data source, is returned;
- object of the SCADA-system sends the request of the last value of the object of parameter's attribute DAQParamAttribute::getVal() and gets the locally saved value of the previous session of the inquiry of data source.

In OpenSCADA this mechanism is implemented by the following modules of subsystem "Data acquisition".
- *DiamondBoards* - module for data access of PC/104 cards of Diamond Systems company. PC/104 cards are available on the ISA-bus, hence, are local and available relatively quickly. When data acquisition is done through interruption the expectation of the packets of fast (up to 200 kHz) in one second (up to 200,000 values in the package) is made and the subsequent placing of packets data in the archives of the DAQ parameters' attributes.
- *DAQGate* - module of reflection of controller's objects of remote OpenSCADA-stations on the local one. The synchronous and asynchronous packet mode of reflection of the archives of remote OpenSCADA-stations is provided.

1.4. Passive acquisition mechanism

The feature of the passive data acquisition mechanism is the initiative of the providing data in the SCADA-system from the data source. This mechanism is quite rare, but can occur in certain conditions or restrictions of the possibility of using the direct data acquisition mechanisms, Fig. 5. An example of such a situation can be the geographically allocated systems of data acquisition through mobile networks GPRS/EDGE. In such networks, empowering the individual client nodes with the real IP-address or the formation of a corporate wireless network can be rather expensive, and therefore more accessible is an initiative of the data transfer session from client dynamic IP-addresses to the one real IP-address of the SCADA-system server. Nevertheless it is possible to work through the network DBMS of the dealer.

Impacts of the modification are transmitted to the source of data at the time of data transfer session by the source.

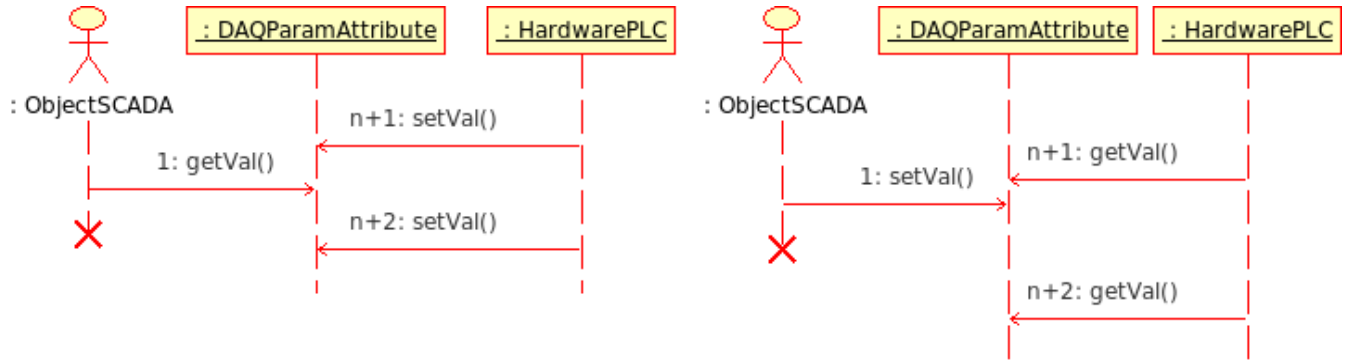

Fig. 5. Diagram of interaction sequence with the passive working mode.

In accordance with the diagram above, we obtain the following behavior of the passive mechanism:
- data source object carries out periodic connection sessions with the object of the parameter's attribute DAQParamAttributeArch::setVal() to transfer its own data and receive influence commands;
- object of the SCADA-system sends the request to the last value of the object of parameter's attribute DAQParamAttribute::getVal() and gets the locally stored value of the previous connection session of the data source.

In OpenSCADA this mechanism has not been yet used, but in principle there is the possibility of its realization in the system.

2. Virtual data sources

In addition to physical data acquisition the function of the virtual data acquisition is also important. Virtual data are the data obtained inside the system both independently and on the basis of physical data. Practically the formation mechanisms of virtual data are implemented in conjunction with the mechanism of user computing. Among the industrial controllers and SCADA-systems the different programming languages are used. In the case of controllers such languages can be for example low-level languages (assemblers), but in recent years the high-level languages (C, Pascal and others) are increasingly used, as well as the formal languages of IEC 61131-3 (sequential function chart SFC, function block diagrams FBD, LD relay circuits and text ST, IL). In the case of SCADA-systems computings are often provided with the help of high-level programming languages and formal languages.

In the OpenSCADA system the programming interfaces and virtual data sources on the basis of different languages in separate modules of a subsystem "Data acquisition" can be implemented. At the time of version 0.6.3.2 the available modules of virtual calculators are:

- Calculator on Java-like language: JavaLikeCalc;
- Block calculator: BlockCalc.

At the OpenSCADA kernel the mechanism for user-defined functions or API of user programming is integrated. User functions can be provided by any object of the system, including modules in accordance with their functionality, thus providing the user with the set of functions for the control of one or another object. User API functions can be either static, ie implementing the fixed functionality of an individual object, and the dynamic ones, ie formed by the user for the desired task in the language of the user high-level programming.

Module JavaLikeCalc provides the system with the mechanism to create dynamic user-defined functions and libraries for Java-like language. Description of functions for Java-like language is to tie up the parameters of the function by the algorithm. In addition, the module has the functions of the direct calculations by creating a computer controllers with the associated computational function. Module provides the mechanism to precompile the context-dependent functions that are used to embed the user algorithms directly in the context of the various components of OpenSCADA. For example, the mechanism of the parameters' templates of subsystem "Data acquisition" and the visual control engine (VCA).

Module BlockCalc provides the OpenSCADA system with the mechanism for creating user calculations. Mechanism of calculations based on the formal language of block diagrams (functional blocks). Languages of block programming based on the concept of block diagrams (functional blocks). And depending on the nature of the block, block scheme can be: logic circuits, relay logic circuits, a model of technological process and others. The essence of the block scheme is that it contains the list of blocks and links between them. From a formal point of view the block - is an element (function), which has inputs, outputs and an algorithm for computing. Based on the concept of programming area block - is a frame of values associated with the object of function. Inputs and outputs of blocks are to be connected to get the whole block scheme.

With the purpose of filling user programming API with user functions the following specialized modules of static user programming API functions are created:

- Library of function for the compatibility with SCADA Complex1: FLibComplex1;
- Library of standard mathematical functions: FLibMath;
- Library of System API functions: FLibSYS.

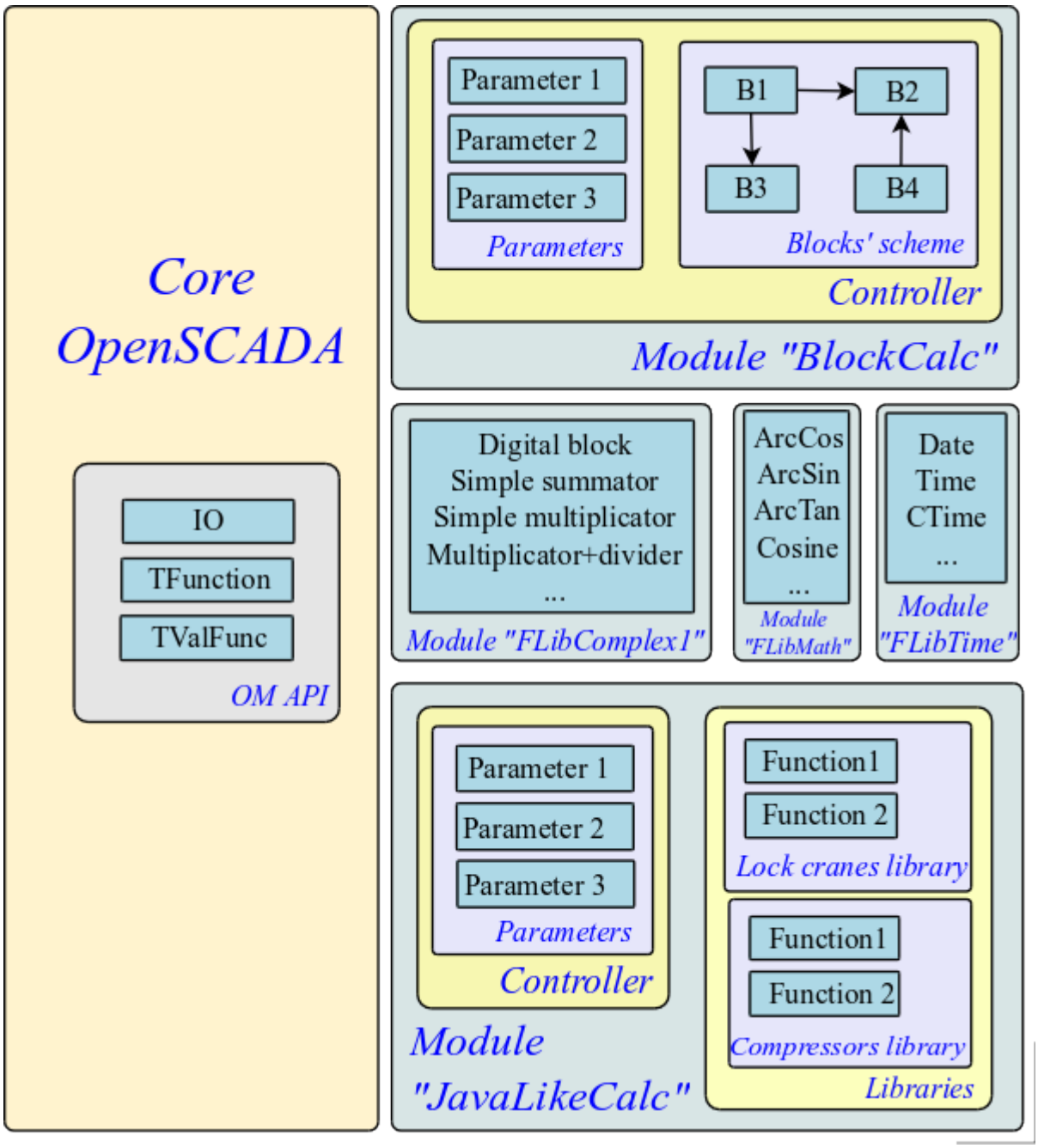

Fig. 6. The overall structure of the components of the programming area

3. Logic level of data processing

Above we talked that type of data source can vary from a "raw" to the complex. The "raw" means the source that provides only the basic signal (integer, real, boolean, string, ...) separately. Under the complex it is meant the source that groups the signals and in the parameter of subsystem "Data acquisition" it provides the attributes of an additional purpose, covering practically all diagnostic tasks, ie the parameter is the complete object, which do not need any additions.

Taking into account this variation, the situation may occur, when the information in the object of data source controller's parameter, is insufficient to describe the real TP object in general and the derived object of a higher level of abstraction is needed. The solution of this situation is the formation of complementary parameters, which is not obvious and confusing. The better solution is to use layer, so-called "Logic level", serving for the flexible formation of parameters, containers of signals with the necessary structure, and which has post-processing.

Functionally "Logic level" is intended to provide the OpenSCADA system with mechanism of free formation of parameters' objects, containers of signals of the necessary structure.

Operating appointment of the "Logic level" is:

- expansion of the scope of the OpenSCADA system by increasing the flexibility of description of parameter's objects of subsystem "Data acquisition";
- reduction of labor costs for the creation of complex automated systems.

The conception of "Logic level" based on the parameters' templates for which in the subsystem "Data acquisition" it is provided the container of the templates libraries (Fig. 1). Each library contains templates of parameters that can be used by the modules of "Data acquisition" subsystem for the implementation of parameters based on templates. The modules of OpenSCADA, which use the templates in their work, are:

- LogicLev - module of the implementation of the classical conception of "Logic level".
- Siemens - data acquisition module for Siemens controllers Series S7. Taking into account the high flexibility and functionality of this controllers, which allows you to create complex data types of different structure, all the parameters of this module work on templates.

General mechanism of the "Logic level" on the example of the LogicLev module is shown in Fig. 7.

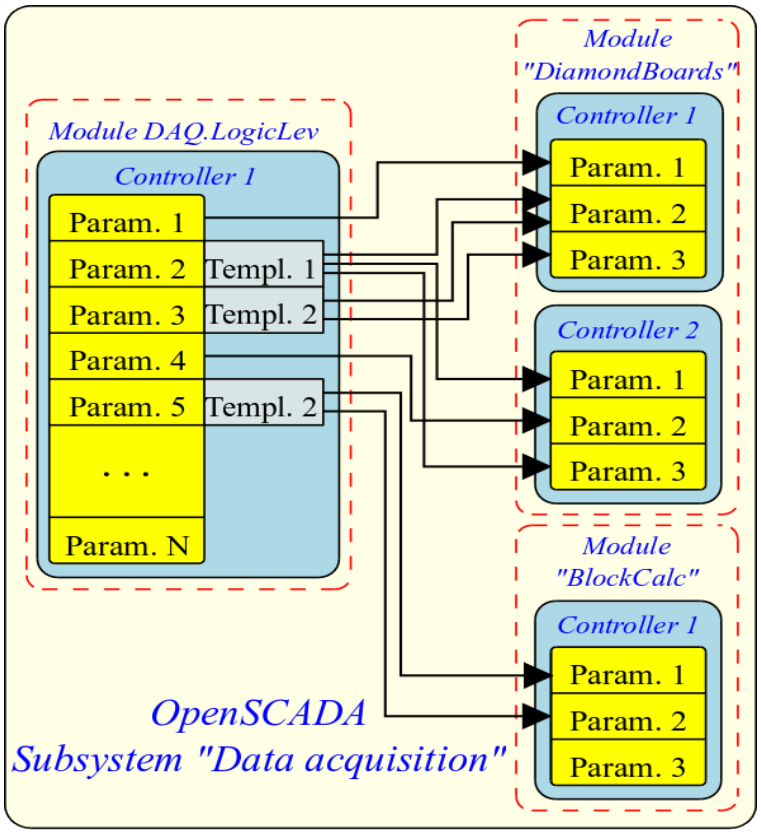

Fig. 7. The mechanism of the "Logic level" on the example of LogicLev module.

On the figure you can see that the parameters of the logic level controller function as reflections of other parameters of "Data acquisition" subsystem (on the example of parameters 1 and 4) and the free formation of parameters based on templates 1, 2 and other parameters of "Data acquisition" subsystem (on the example of the parameters 2, 3 and 5).

Structure of the parameters with the template in their basis has the structure shown in Fig. 8.

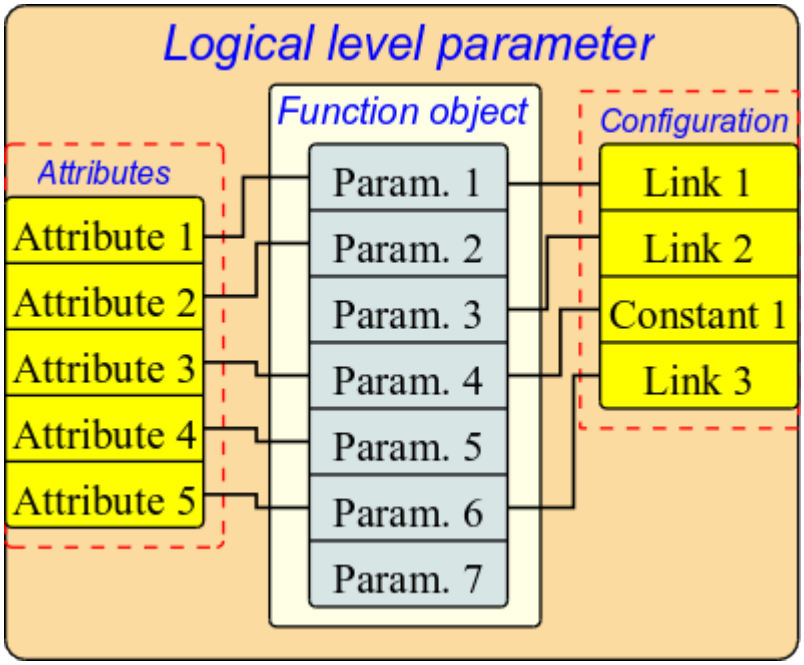

Fig. 8. Structure of the parameters, with a template in its basis.

As can be seen from the structure, the logic level parameter consists of the function object, attributes and configuration of the template. The function object is an instance of the execution of the template's function with the set of inputs/outputs and the computation program of the template on the language of user programming, usually it is the Java-like programming language of the module DAQ.JavaLikeCalc. But the template may be generally without the program, providing only the structure of transfer the inputs/outputs. Attributes in the structure represent the list of attributes of the result parameter in accordance with the template. Configuration in the structure provides the configuration of the template's properties and its external links.

The logic of the work of logic-level parameters can be written as follows:

- Parameter connects with the template from which we obtain the structure of attributes in accordance with the template's function.
- At the moment of linking the parameter with the function the linkage of an object of the parameter's function instance with the function of the template.
- Further, in accordance with the template of function, the structure of links is formed. Based on the structure of links the form of linkage the parameter is formed and the user sets the links .
- When you access the attributes of the obtained parameter the check for the presence of a direct link is done. In the case of a direct link presence the request is routed by this link, otherwise the value is taken from an object of the parameter's function instance.
- At this moment the template's function calculation works using the the object of the parameters' function. However, before the calculation the reading of the values by the links is made, and after calculation the results are recorded by these links.

Parameters' template in general provides the following:

- structure of I/O of the template's function;
- signs of the configuration and linkage of the template (constant, link);
- preliminary values of the configuration of constants and templates of links' configuration;
- signs of the attributes of the resulting parameter of the logic level types: not attribute, an attribute with full access, attribute with read-only access;
- mechanism for calculating the I/O of the templates' function using the user programming language of OpenSCADA.

Fig. 9 shows image of the configuration tab of the parameters' template of subsystems "Data acquisition" as the table with the configuration of inputs/outputs and the text of the program of user programming.

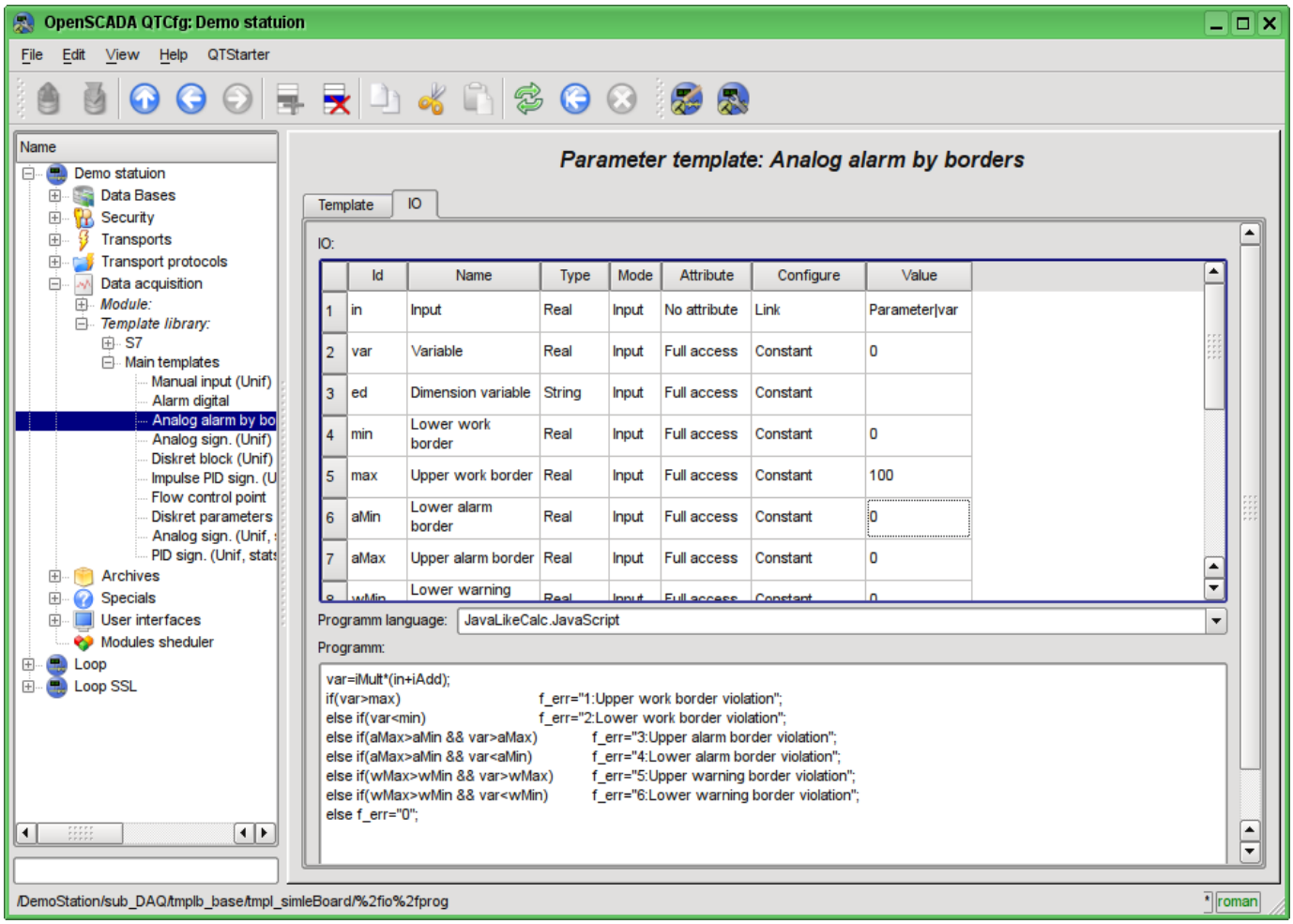

Fig. 9. The configuration tab of parameters' template of subsystem "Data acquisition".

The input/output field of the parameter's template provides the following properties of special purpose: "Attribute", "Configure" and "Value".

The "Attribute" property is the reflecting sign of the the i/o of the template on the resulting attribute of the parameter. There the following options for this property are provided:

- *No attribute* - input/output of the template's function does not reflect on the attribute;
- *Read only* - input/output of the template's function reflects on the attribute with read-only access;
- *Full access* - input/output of the template's function reflects on the attribute with full access.

The "Configure" property is the sign indicating the using of input/output of the template's function in the resulting configuration of the template on the logic level. The following options for this property are provided:

- *Constant* - available for setting only on the level of the configuration of parameter's template as a constant;
- *Public constant* - available for setting at the parameter of logic level in the configuration section of the template as a constant;
- *Link* - available for setting at the parameter of the logical level in the configuration section of the template in the form of link.

The field "Value" describes the preset value for the constants and configuration template of the external links. Template of the configuration of external links is used to describe the mechanism of grouping and automatic allocation of external links. The structure of the template of configuration of external links is the specific for each module of subsystem "Data acquisition", which uses the template's mechanism. In the case of the logic level module the allocation is made over the external attributes of the parameters with the

template of configuration of the external link of the form: <Parameter>|<attribute>. Where <Parameter> is used to combine the parameters and place on the configuration form, and an attribute - for the associated linkage of the attributes at the appointment of the parameter.

As an example of the template's using in Figure 10 lets show an images of the parameter of the logic level module "F3". In Fig.10 the tab "Template config" is presented', it serves for the configuration, including the linkage, of the parameter's template. In Fig.11 the tab "Attributes" is shown with the list of attributes and their values, created through the template.

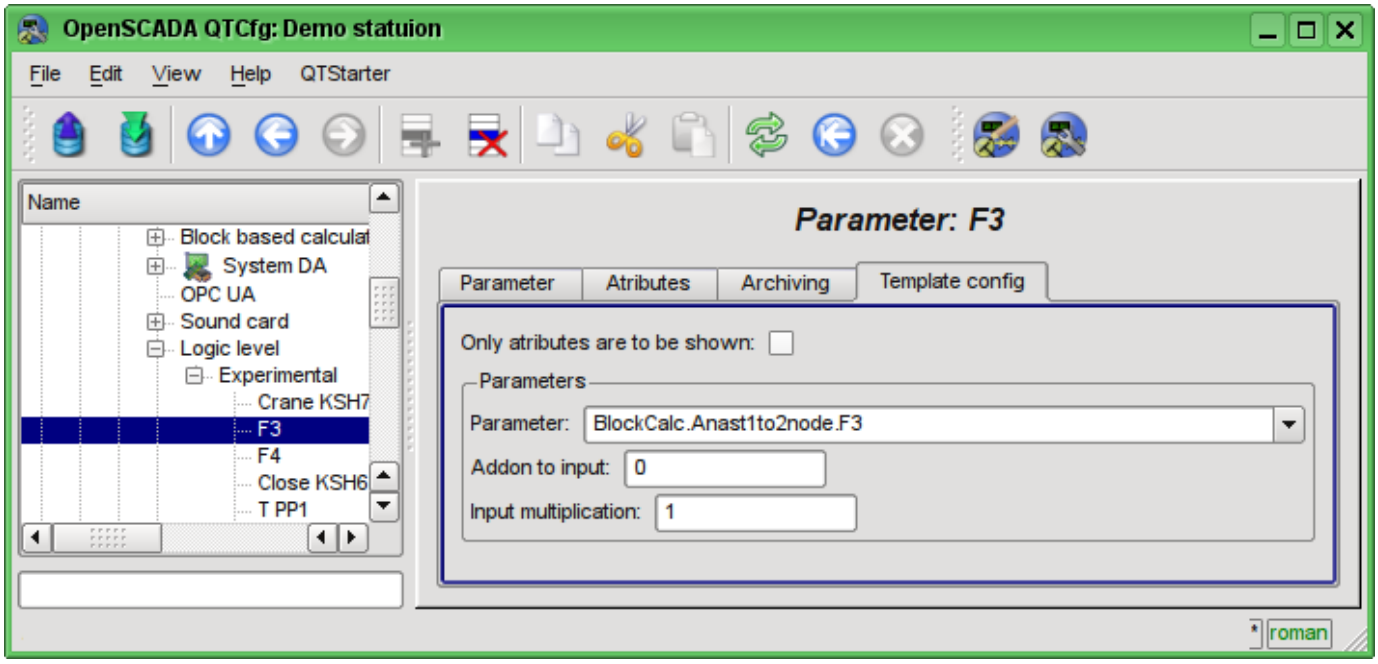

Fig. 10. The "Template config" tab of the "F3" parameter of the logic level module.

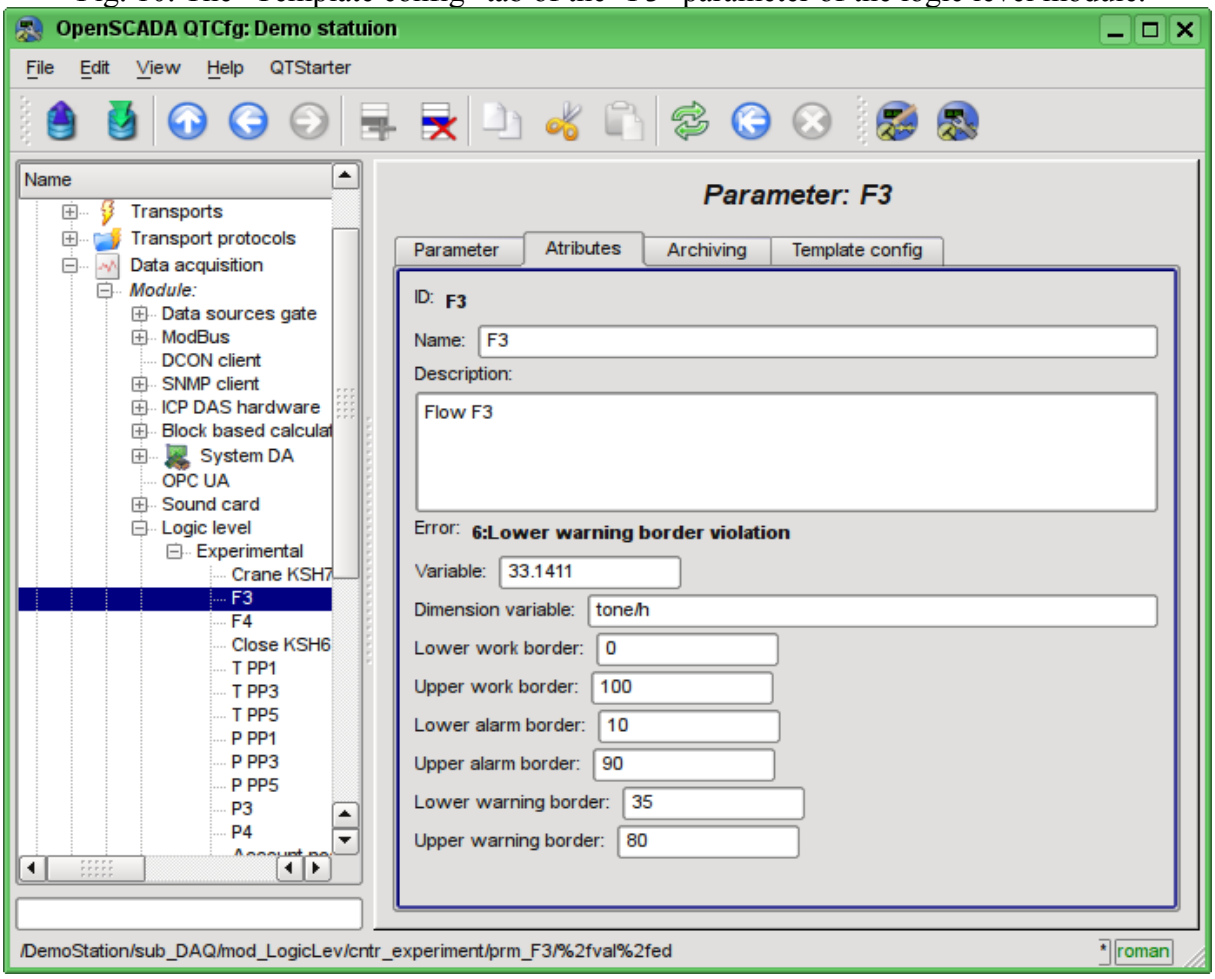

Fig. 11. The "Attributes" tab of the "F3" parameter of the logical level module.

4. Redundancy of the data sources

Redundancy in general and of the data sources in particular serves to increase the overall level of fault-tolerance of the solution by integrating the redundant nodes in collaboration with the main node. In case of failure of the main node the grab of the main node functions by the redundant one takes place. The redundant scheme can work in the mode of capacity allocation between the co-operating nodes.

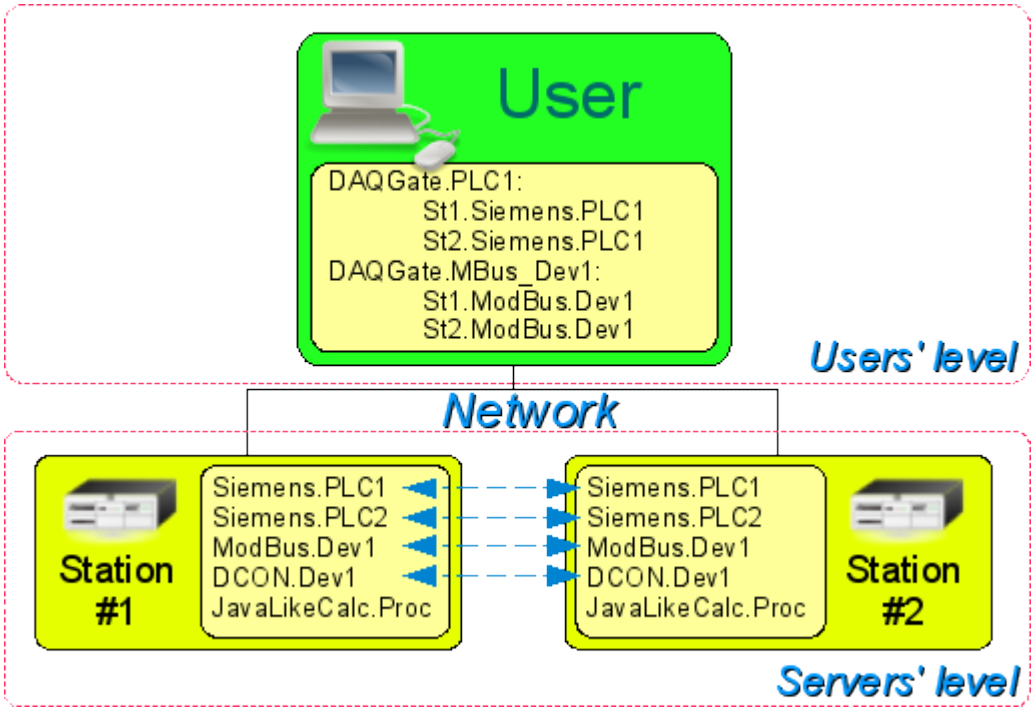

Fig. 12. Horizontal and vertical redundancy.

In the case of a subsystem "Data acquisition" of the OpenSCADA system the data redundancy (Figure 12) performs the following functions:

- Redundancy of the data acquisition mechanism. Typically, this function is realized without special arrangements by simply running of the parallel redundancy stations with the same configuration and working independently. However, in the case at the station, which works as PLC, such approach is unacceptable because of the simultaneous making of control actions and the absence of synchronization of calculators' data.
- Compensation of the data loss on the time of the node stop with the redundant node archive. There are two mechanisms of compensation. The first and the main mechanism implements the loading of the sections of the archive from the redundant station at the time of the station startup in general or of individual controllers of "DAQ" subsystem. the section of the archive is requested from the moment of the last record in the local archive and till the current time. The depth of the request is limited by the indicating of the limit time in the configuration of the redundancy. The second, complementary mechanism, performs the filling of the "holes" in the archive at the time of the actual user's request to the data. Such an approach on the one hand allows to make the predictable in time synchronization at startup and on the other hand - actually eliminates the data loss in the case of working at least one station during the entire time.
- Capacity allocation of data acquisition between the nodes. When creating complex allocated systems there can be an important question of predicting and optimizing of the overall system performance. Taking into account these problems the redundancy mechanism provides the execution of tasks of data acquisition of individual sources (OpenSCADA controllers) only at one station. The other stations' tasks would go to data synchronization mode with the executive station. In the case of loss of the connection with the executive station the task of the local data acquisition is started. It is also provided the possibility of optimal capacity allocation of the execution of data acquisition task's of the controllers' group between the stations.
- Optimization of the load on the external data sources through the data request from an external source by the only one node. In practice, we often meet highly loaded data sources or interfaces of access to the data sources, for which even the data acquisition by one station can be a problem and

would require reducing the acquisition periodicity, ie data quality. The mechanism of redundancy, except of capacity allocation between the stations as described above allows you to remove an additional load form the data source and its interfaces, thereby improving the quality of data.

- Prevention of some differences of data on different nodes associated with the mismatch of moments of time at the independent acquisition of data by individual nodes by means of receiving the data from the station with an active controller. In systems with redundant and high accountability it should be excluded or minimized the differences in the data at different stations, that means the real acquisition of data by one station and synchronization with these data of other stations.

Configuration of the redundancy starts with the addition of redundant stations in the list of OpenSCADA system stations in the tab "Subsystem" of the "Transports" subsystem (Fig.13). Then the whole configuration of the redundancy is made in the "Redundance" tab of subsystem "Data acquisition" (Fig. 14).

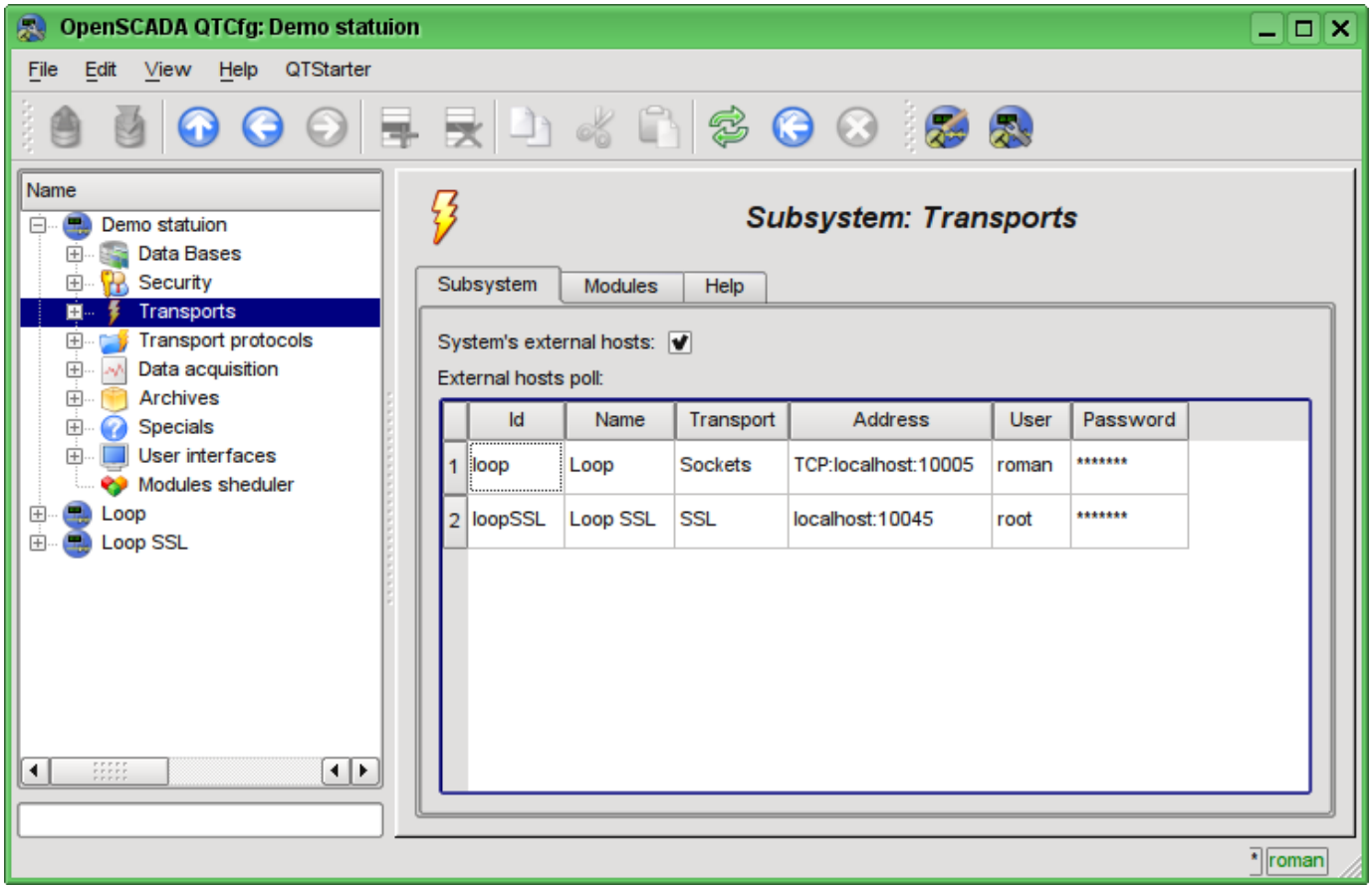

Fig. 13. The "Subsystem" tab of the "Transports" subsystem.

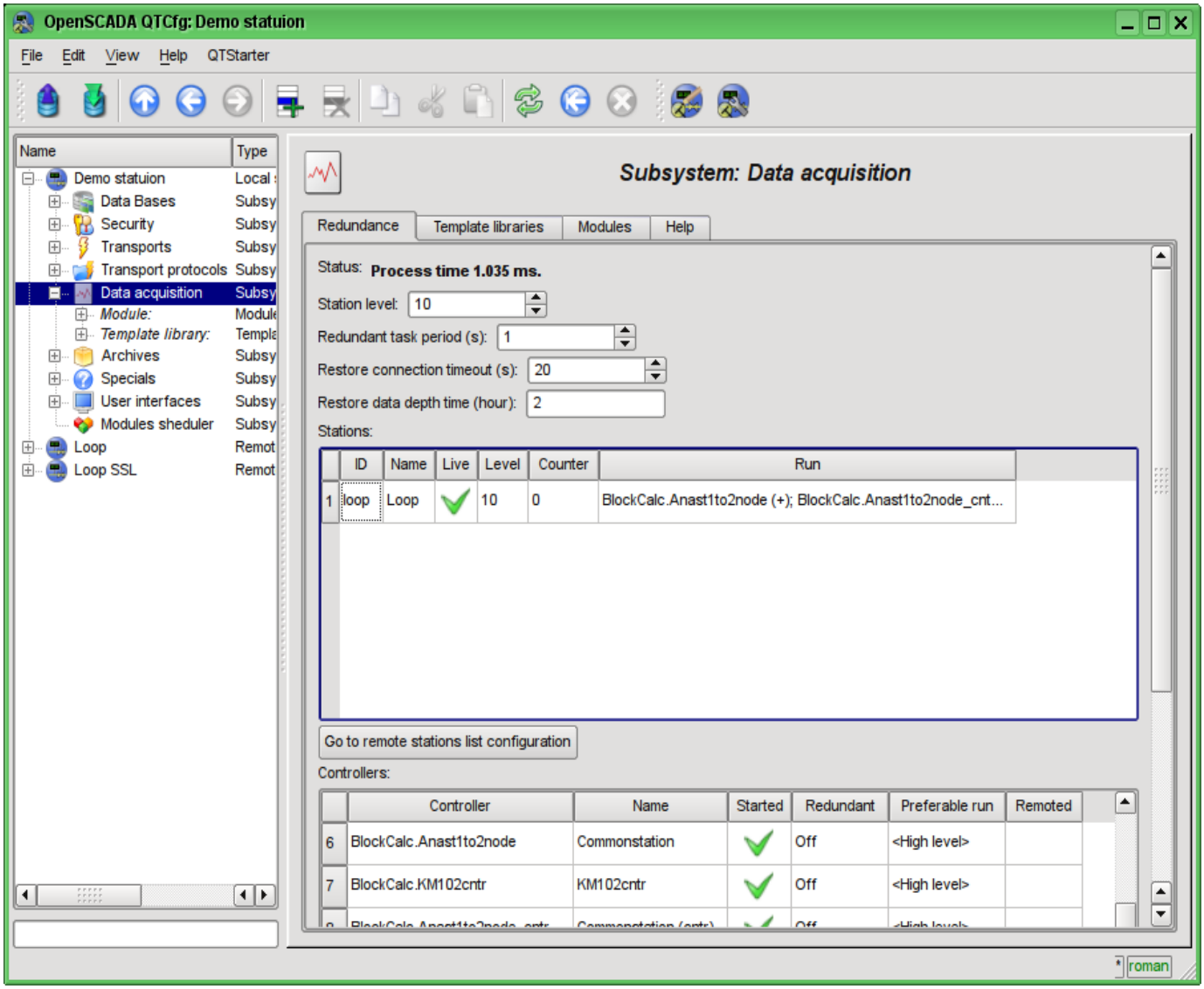

Fig. 14. The "Redundance" tab of the "Data acquisition" subsystem.

The service task of the redundancy mechanism is always running and executed at intervals which are prescribed in the appropriate configuration field. The real work on implementing the redundancy is carried out in the presence of at least one redundant station in the list of stations, and implies:

- Monitoring of the connection with external stations. In the monitoring process the requests to remote stations are made to get the information about them updated and to check connection. In the case of loss of connection with the station the repeat of connection to it is made through interval specified in the configuration field "Restore connection timeout". In the "Live" field of the station the current state of communication is displayed. In the "Counter" field the number of requests carried to the remote station, or the time remaining for the next connection attempt to the lost station is displayed. In the "Run" field there is a list of active controllers at the remote station with a sign of the local execution.
- Local planning of the controllers' execution in reserve. Planning is carried out in accordance with the station's level and preferences of controllers' execution.
- calling the data synchronization function for the local controllers working in the mode of synchronization of data from external stations. During the call, it is being prepared to request of the data from the remote station for the parameters in the controller starting from the time of the last request. On the request the only the values of modified attributes and sequence of values from an archive in case of loss of several cycles of values are returned.

To monitor the time spent in the cycle of redundancy tasks the field status is provided. When approaching the real time of execution to the cycle of the redundancy tasks it is recommended to increase the frequency of execution of this task!

For the controller of subsystem "Data acquisition" there is provided the modes of asymmetric and

symmetric redundancy. Asymmetric redundancy is working with the configuration of the controller of the remote station, as it is, and does not trying to generalize it. Symmetrical mode supposes the synchronization of configuration of the controllers of stations with the configuration of the highest level station, and suggests the changes in the configuration of all controllers of the stations when changing it on the one of the stations. Currently this mode is not implemented!