

Functional characteristics and demands of OpenSCADA system

The page contains the information allowing to receive the general representation about functions which the OpenSCADA system can carry out by the current moment. Functions are grouped on spheres of application of OpenSCADA system. For reception of a picture as a whole of functions planned or make now are included also. The page also have demands of OpenSCADA system for it building and execution.

Table of contents

Functional characteristics and demands of OpenSCADA system	1
1. The employment area of system OpenSCADA	2
1.1. SCADA system's server	3
1.2. Station of the operator of technological process, the board of the dispatcher, the panel of monitoring, etc.	5
1.3. The environment of execution of controllers (PLC)	6
2. Requirements for OpenSCADA	8
2.1. Execution	8
2.2. Building	9

1. The employment area of system OpenSCADA

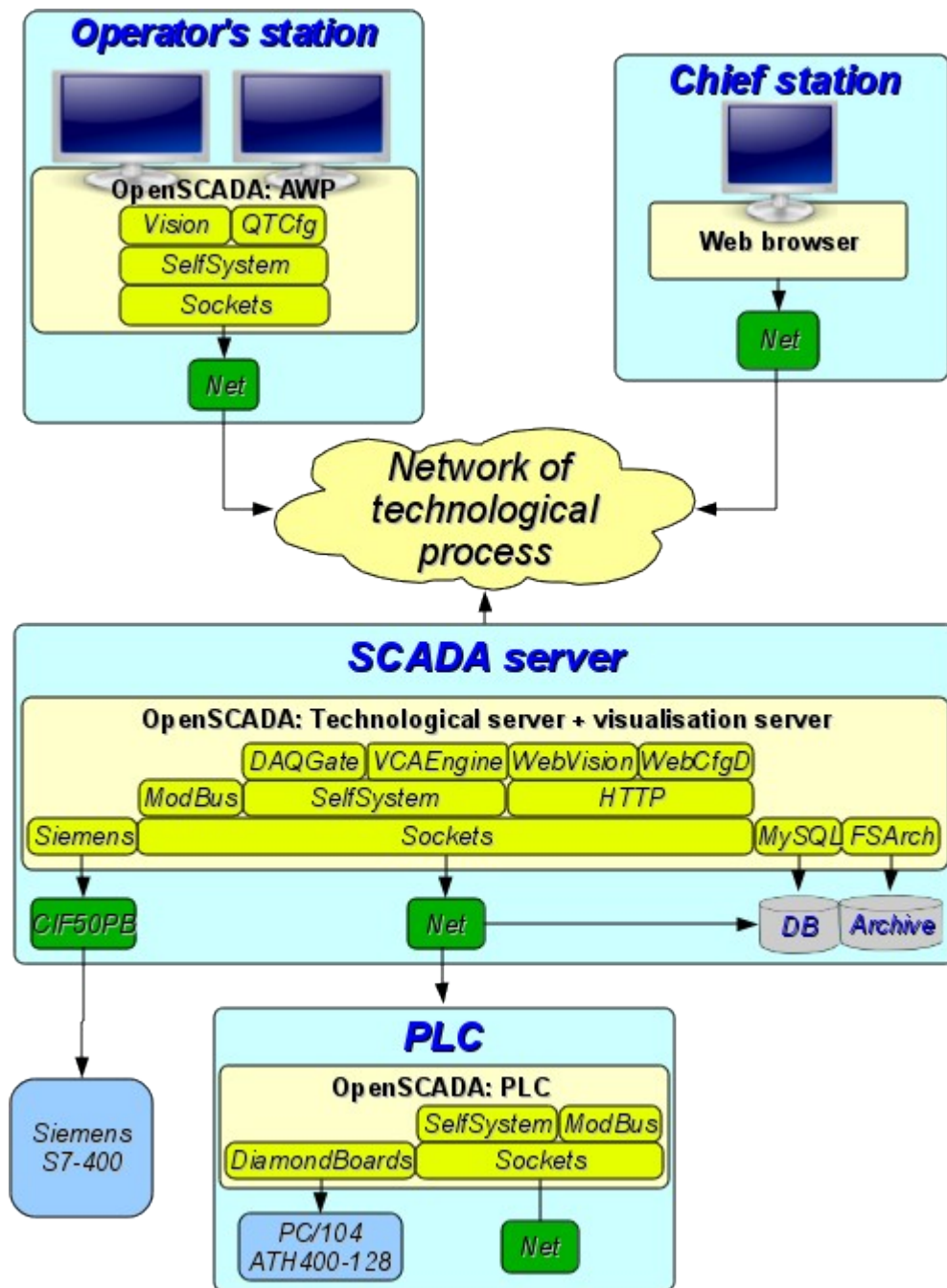


Fig. 1. OpenSCADA system's roles

1.1. SCADA system's server:

- The visual control and management by means of the interfaces:
 - Remote visualization server grounded on visualization and control area (VCA) engine [VCAEngine](#). The module UI.Vision local starting and connecting to the visualization server.
 - Remote WEB interface. By means of a Web-browser, the visualization module [WebVision](#) and the module of a kernel of visual control area [VCAEngine](#).
 - Simple remote Web-interfaces of user. By mean Web-browser and UI-module [WebUser](#).
- Data acquisition (DAQ) from sources:
 - Information about a platform (hardware-software) on which the server works. By means of the DAQ-module [System](#).
 - Data acquisition from sources which support protocol SNMP (Simple Network Management Protocol). By means of the DAQ-module [SNMP](#).
 - Data acquisition from controllers of firm Siemens of S7 series. By means of the DAQ-module [Siemens](#).
 - Data acquisition of industrial controllers under the protocol ModBus. By means of the DAQ-module [ModBus](#).
 - Data acquisition of industrial controllers under the protocol DCON. By means of the DAQ-module [DCON](#).
 - Formation of derivative structures of parameters on the basis of templates of parameters and data from other sources. By means of the DAQ-module [LogicLev](#).
 - Data acquisition from other servers and PLC, based on OpenSCADA, possibly for duplication. By means of the DAQ-module [DAQGate](#).
 - Data acquisition from sound controller's input channels. By means of the DAQ-module [SoundCard](#).
 - Data acquisition from hardware of firm [ICP DAS](#). By means of the DAQ-module [ICP DAS](#).
 - Data acquisition from sources which support protocol OPC-UA. By means of the DAQ-module [OPC UA \(RU\)](#).
 - Data acquisition from different sources, which have utilities for access to it data or it accessibly through simple special network protocols. Made by getting procedure writing on language of user programming by DAQ-module [JavaLikeCalc](#), and also transport-protocol-module [User Protocol](#).
- Providing data to upper-level systems:
 - By means of interfaces:
 - Serial interface (RS232, RS485, Modem, ...), by helps of transport module [Serial](#).
 - IP-networks sockets and network levels protocols TCP, UDP and Unix, by helps of transport module [Sockets](#).
 - Security sockets layer (SSL), by helps of transport module [SSL](#).
 - By means of protocols:
 - Self OpenSCADA protocol, by helps of transport's protocol module [SelfSystem](#).
 - ModBUS family protocol (TCP, RTU and ASCII), by helps of transport's protocol module [ModBUS](#).
 - "OPC UA" protocol, by helps of transport's protocol module [OPC UA \(RU\)](#).
 - Simple special protocols, developed by users by helps of transport's protocol module [User Protocol](#).
- Implementation of user calculations in languages:
 - Language of block schemes. By means of the DAQ-module [BlockCalc](#).
 - With the help of Java-like language of a high level. By means of the DAQ-module [JavaLikeCalc](#).
- Archiving messages, conducting reports on various categories and levels by means of mechanisms:
 - Files in a XML-format or the flat text with packing the out-of-date archives. By means of the archiving module [FSArch](#).
 - In tables of archival DB. By means of the archiving module [DBArch](#).

- In plans. On other server, it is possible to the allocated archiving server, based on OpenSCADA.
- Archiving values of the collected data by means of mechanisms:
 - Files with double packing: consecutive and standard archiver gzip. By means of the archiving module [FSArch](#).
 - In tables of archival DB. By means of the archiving module [DBArch](#).
- Configuration and management of a server through:
 - The WEB-interface. By means of a Web-browser and the UI-module [WebCfgD](#) and [WebCfg](#).
 - From the remote configuration station. By means of the UI-module at configuration station [QTCfg](#) and the interface of management OpenSCADA reflected in the protocol [SelfSystem](#).
- Data storage of a server in a DB of types:
 - MySQL. By means of the DB-module [MySQL](#).
 - SQLite. By means of the DB-module [SQLite](#).
 - PostgreSQL. By means of the DB-module [PostgreSQL](#).
 - DBF. By means of the DB-module [DBF](#).
 - FireBird. By means of the DB-module [FireBird](#).
 - In plans. DB accessible on other server based on OpenSCADA.
 - In plans. LDAP.

1.2. Station of the operator of technological process, the board of the dispatcher, the panel of monitoring, etc.:

- The visual control and management by means of the interfaces:
 - The local (fast) interface based on QT library. By means of the visualization module [Vision](#) and the module of a kernel of the visual control area [VCAEngine](#) include ability of visualization from remote engine of VCA, visualization server.
 - Remote WEB interface. By means of a Web-browser, the visualization module [WebVision](#) and the module of a kernel of visual control area [VCAEngine](#).
 - Simple remote Web-interfaces of user. By mean Web-browser and UI-module [WebUser](#).
- Data acquisition (DAQ) from sources:
 - Data acquisition from other servers and PLC, based on OpenSCADA, for data transportation and for duplication. By means of the DAQ-module [DAQGate](#).
 - Data acquisition from sources which support protocol SNMP (Simple Network Management Protocol). By means of the DAQ-module [SNMP](#).
 - Data acquisition from sources which support protocol OPC-UA. By means of the DAQ-module [OPC UA \(RU\)](#).
- Implementation of the user calculations in languages:
 - Language of block schemes. By means of the DAQ-module [BlockCalc](#).
 - With the help of Java-like language of a high level. By means of the DAQ-module [JavaLikeCalc](#).
- Archiving messages, conducting reports on various categories and levels by means of mechanisms:
 - Files in a XML-format or the flat text with packing the out-of-date archives. By means of the archiving module [FSArch](#).
 - In tables of archival DB. By means of the archiving module [DBArch](#).
 - In plans. On other server, it is possible to the allocated archiving server, based on OpenSCADA.
- Configuration and management of station through:
 - The WEB-interface. By means of a Web-browser and the UI-module [WebCfgD](#) or [WebCfg](#).
 - The QT-interface. By means of the UI-module [QTCfg](#).
 - From the remote configuration station. By means of the UI-module at configuration station [QTCfg](#) and the interface of management OpenSCADA reflected in the protocol [SelfSystem](#).
- Data storage of station in a DB of types:
 - MySQL. By means of the DB-module [MySQL](#).
 - SQLite. By means of the DB-module [SQLite](#).
 - PostgreSQL. By means of the DB-module [PostgreSQL](#).
 - DBF. By means of the DB-module [DBF](#).
 - FireBird. By means of the DB-module [FireBird](#).
 - In plans. DB accessible on other server based on OpenSCADA.
 - In plans. LDAP.

1.3. The environment of execution of controllers (PLC):

- Data acquisition (DAQ) from sources:
 - Cards of data acquisition of firm [Diamond Systems](#). By means of the DAQ-module [DiamondBoards](#).
 - Information on a platform (hardware-software) on which the server works. By means of the DAQ-module [System](#).
 - Data acquisition from sources which support protocol SNMP (Simple Network Management Protocol). By means of the DAQ-module [SNMP](#).
 - Data acquisition of industrial controllers under the protocol ModBus. By means of the DAQ-module [ModBus](#).
 - Data acquisition of industrial controllers under the protocol DCON. By means of the DAQ-module [DCON](#).
 - Formation of derivative structures of parameters on the basis of templates of parameters and data from other sources. By means of the DAQ-module [LogicLev](#).
 - Data acquisition from other servers and PLC, based on OpenSCADA, possibly for duplication. By means of the DAQ-module [DAQGate](#).
 - Data acquisition from sound controller's input channels. By means of the DAQ-module [SoundCard](#).
 - Data acquisition from hardware of firm [ICP DAS](#). By means of the DAQ-module [ICP_DAS](#).
 - Data acquisition from sources which support protocol OPC-UA. By means of the DAQ-module [OPC UA \(RU\)](#).
 - Data acquisition from different sources, which have utilities for access to it data or it accessibly through simple special network protocols. Made by getting procedure writing on language of user programming by DAQ-module [JavaLikeCalc](#), and also transport-protocol-module [User Protocol](#).
- Providing data to upper-level systems:
 - By means of interfaces:
 - Serial interface (RS232, RS485, Modem, ...), by helps of transport module [Serial](#).
 - IP-networks sockets and network levels protocols TCP, UDP and Unix, by helps of transport module [Sockets](#).
 - Security sockets layer (SSL), by helps of transport module [SSL](#).
 - By means of protocols:
 - Self OpenSCADA protocol, by helps of transport's protocol module [SelfSystem](#).
 - ModBUS family protocol (TCP, RTU and ASCII), by helps of transport's protocol module [ModBUS](#).
 - "OPC UA" protocol, by helps of transport's protocol module [OPC UA \(RU\)](#).
 - Simple special protocols, developed by users by helps of transport's protocol module [User Protocol](#).
- Management, regulation and performance of other user calculations in languages:
 - Language of block schemes. By means of the DAQ-module [BlockCalc](#).
 - With the help of Java-like language of a high level. By means of the DAQ-module [JavaLikeCalc](#).
- Archiving messages, conducting reports on various categories and levels by means of mechanisms:
 - Files in a XML-format or the flat text with packing the out-of-date archives. By means of the archiving module [FSArch](#).
 - In tables of archival DB. By means of the archiving module [DBArch](#).
 - In plans. On other server, it is possible to the allocated archiving server, based on OpenSCADA.
- Archiving of values of the collected data by means of mechanisms:
 - Buffers in memory of the setting depth. By means of the built in archiving mechanism of the values of kernel OpenSCADA.
 - Files with double packing: consecutive and standard archiver gzip. By means of the archiving module [FSArch](#).

- In tables of archival DB. By means of the archiving module [DBArch](#).
- Configuration and management PLC through:
 - The WEB-interface. By means of a Web-browser and the UI-module [WebCfgD](#) or [WebCfg](#).
 - From the remote configuration station. By means of the UI-module at configuration station [QTCfg](#) and the interface of management OpenSCADA reflected in the protocol [SelfSystem](#).
- Data storage PLC in a DB of types:
 - All data in a configuration file (fixed).
 - MySQL. By means of the DB-module [MySQL](#).
 - SQLite. By means of the DB-module [SQLite](#).
 - PostgreSQL. By means of the DB-module [PostgreSQL](#).
 - DBF. By means of the DB-module [DBF](#).
 - FireBird. By means of the DB-module [FireBird](#).
 - In plans. DB accessible on other server based on OpenSCADA.
 - In plans. LDAP.

2. Requirements for OpenSCADA

2.1. Execution

The demands to apparatus for OpenSCADA system execution at different roles viewed into table 1. The demands to programs for OpenSCADA system execution and its modules allow into table 2.

Table 1. The demands to apparatus for OpenSCADA system and its modules.

Role	Demands
SCADA system's server	CPU: x86_32 (more than i586), x86_64 or ARM, with frequency more 500 MHz MEM: 128 MB HDD: 10 GB include OS and place for archives
Station of the operator of technological process, the board of the dispatcher, the panel of monitoring, etc.	CPU: x86_32 (more than i586), x86_64 or ARM, with frequency more 1 GHz MEM: 512 MB HDD: 4 GB include OS without archives place
The environment of execution of controllers (PLC)	CPU: x86_32 (more than i586), x86_64 or ARM, with frequency more 133 MHz MEM: 32 MB HDD: 32 MB include OS without archives place.

Table 2. Dependences of performance of OpenSCADA system and its modules.

Component	Description
<i>Dependences of OpenSCADA system's kernel</i>	
OS Linux	The distribution kit of operating system Linux (ALTLinux, SuSELinux, Mandriva, ASPLinux, Fedora, Debian, Ubuntu ...)
"Standard libraries"	Standard set of libraries: linux-gate, libstdc++, libgcc_s, libc, libdl, librt, libcrypt, libm, libpthread. Certainly this already allow into installed distribution. Special demand is using native thread library NPTL, already used for all modern distributions of the Linux.
zlib	Compression library.
libpcre	Library for use regular expressions, compatible with Perl.
libgd (opt: --disable-LibGD)	Graphic library GD version 2, it is desirable that it will be without XPM support (dependence on library of a X-server is excluded) and support of FontConfig.
libexpat (opt: auto)	Library of XML-parser.
<i>DB.MySQL module</i>	
libMySQL	Library for access to MySQL DBMS.
<i>DB.SQLite module</i>	
libsqlite3	Library for access to built in DB SQLite version 3.
<i>DB.PostgreSQL module</i>	
libpq	Library for access to PostgreSQL DBMS version more 8.3.0.
<i>DB.FireBird module</i>	
FirebirdSS	FireBird DBMS version 2. Often is absent in distribution kits of Linux and demands individual loading from an official site (http://www.firebirdsql.org)!
<i>Transport.SSL module</i>	
libssl	Library for codifying OpenSSL.
<i>DAQ.SNMP module</i>	
libsnmp	Library for access to data of network devices under SNMP protocol.

Component	Description
<i>DAQ.System module</i>	
libsensors (opt: auto)	Hardware sensors' library versions 2 and 3.
<i>DAQ.SoundCard module</i>	
libportaudio	Multiplatform library for access to sound controller version 19 and higher.
<i>DAQ.OPC-UA module</i>	
libssl	Library for codifying OpenSSL.
<i>Modules: UI.Vision, UI.WebVision, Special.FLibSYS</i>	
libfftw3 (opt: auto)	Library for fast Fourie transfer of signals.
<i>Modules: UI.QTstarter, UI.QTCfg, UI.Vision</i>	
libQT4 (libQtCore,libQt Gui)	Library for construction of user graphic interface QT version 4.3 and higher.

* - "opt: auto" - provides for disable of using the library at build time on it absence.

2.2. Building

Dependences of system OpenSCADA for building of the OpenSCADA kernel and its modules are tabulated bellow.

Table 3. Dependences of building of OpenSCADA system and its modules.

Component	Description
<i>The general requirements for building OpenSCADA</i>	
OS Linux	The distribution kit of operating system Linux (ALTLinux, SuSELinux, Mandriva, ASPLinux, Fedora, Debian, Ubuntu ...)
g++	The compiler of language C++ from a collection of compilers GCC, including library GLibC
autotools (autoconf, automake, libtool)	Tools for formation of building environment of OpenSCADA. They are necessary only in the case of changing building environment of OpenSCADA, for example for addition of the new module or change of the fixed parameters of building.
gettext	Group of utilities for preparation and compilations of translations of the interface of programs on various languages in conformity with internationalization standard I18N.
zlib (devel)	Compression library, a package for development.
libpcre (devel)	Library for use regular expressions, compatible with Perl, a package for development.
libgd (devel, opt: --disable-LibGD)	Graphic library GD version 2, a package for development, it is desirable that it will be without XPM support (dependence on library of a X-server is excluded) and support of FontConfig. It is used for construction of trends and other images in PNG format.
libexpat (devel, opt: auto)	Library of XML-parser, package for development. The interface of management of OpenSCADA and other components are constructed on the basis of language XML.
<i>DB.MySQL module</i>	
libMySQL (devel)	Library for access to MySQL DBMS, a package for development on language C.
<i>DB.SQLite module</i>	
libsqlite3 (devel)	Library for access to built in DB SQLite version 3, a package for development.
<i>DB.PostgreSQL module</i>	

Component	Description
libpq	Library for access to PostgreSQL DBMS version more 8.3.0, a package for development.
<i>DB.FireBird module</i>	
FirebirdSS	FireBird DBMS version 2, a package for development. Often is absent in distribution kits of Linux and demands individual loading from an official site (http://www.firebirdsql.org)!
<i>Transport.SSL module</i>	
libssl (devel)	Library for codifying OpenSSL, a package for development.
<i>DAQ.JavaLikeCalc module</i>	
bison	The program of generation of parsers on the basis of grammar of language.
<i>DAQ.SNMP module</i>	
libsnmp (devel)	Library for access to data of network devices under SNMP protocol, a package for development.
<i>DAQ.System module</i>	
libsensors (devel, opt: auto)	Hardware sensors' library versions 2 and 3, a package for development.
<i>DAQ.Siemens module</i>	
glibc-kernheaders	Linux-kernel headers by library GLibC.
<i>DAQ.SoundCard module</i>	
libportaudio (devel)	Multiplatform library for access to sound controller, a package for development version 19 and higher.
<i>DAQ.OPC-UA module</i>	
libssl (devel)	Library for codifying OpenSSL, a package for development.
<i>Modules: UI.Vision, UI.WebVision, Special.FLibSYS</i>	
libfftw3 (devel, opt: auto)	Library for fast Fourie transfer of signals, package for development.
<i>Modules: UI.QTStarter, UI.QTCfg, UI.Vision</i>	
libQT4 (devel)	Library for construction of user graphic interface QT version 4.3 and higher, package for development.

* - "opt: auto" - provides for disable of using the library at build time on it absence.