SCADA system for monitoring and control of small industrial producing unit

Hristo Nenov[1]

Abstract – **The topic of this paper is design and development of SCADA system for remote autonomous production units (Remote Automation Unit - RAU) from a "small farm" type. For the purpose of developing the material base of the farm is reduced to a limited number of controlled functional units which could vary depending on the particular implementation on a real production site.**

Keywords – **SCADA, MODBUS, Java, JPA, PLC.**

I. INTRODUCTION

In modern economies it is almost impossible to imagine industries where computers and machines have not have taken control of a significant part of the production processes. The main reason behind this is the need to increase the competitiveness of production in the sector. The formula for achieving this goal is to increase the production quantity and reduce the production costs. The implementation of different tools for mechanization of the production creates an opportunity for that. Agriculture is no exception and modern technologies are becoming more widely used in this industry. The combination of the latest information technologies, global information network Internet, more available control devices and the use of renewable energy sources make the appearance of modern agriculture - autonomous, more efficient and predictable.

II. ARCHITECTURE OF THE SYSTEM

A. General scheme

Remote Automation Unit (RAU) from "small farm" type consist the following modules:

1. Greenhouses with the following adjacent manageable systems and sensors:
   - Drip irrigation system with controllable valve;
   -Sprinkling system with controllable valve;
   -Ventilation;
   -Measuring devices for air temperature;
   -Measurement devices for humidity.
2. Wells - local inexhaustible sources with the following adjacent manageable systems and sensors:
   - Pumping systems;
   - Sensors for water level.
3. Water storage reservoir - capacitive buffer source, which is fed with water from local inexhaustible sources and supplies water for the entire farm, the following adjacent manageable systems and sensors:
   -Pumping systems for water;
   - Water level sensor.

For the purpose of real testing, simplified configuration is selected and it is deployed as follows:

- Greenhouse with an individual drip irrigation system, sprinkling system, ventilation system, temperature sensor and an air sensor for humidity;
- Water storage supplying the greenhouse, with a pumping system for water supply and one sensor for water level;
- Two wells, each with a pumping system connected to the water supply and water retaining with one sensor for water level

B. Physical layer

Physical layer - covers the management and transfer of data to and from various sensors and controllable devices (sensors, valves, relays). This layer is designed to the specific realities of production sites and goals.

B. Industrial Interface layer

The layer is implemented on one or more programmable logic controllers (PLC) and includes logic for gathering and initial processing of data to and from physical sensors and controllable devices and providing them to the layers of higher-level system. PLC interface has its own logic, which is connected to the other layers of the system by appropriate industrial protocols.

C. Application layer

The layer implements the application management logic. It consist two aspects:

- Management of autonomous remote objects (small farms);
- Business logic implemented on the Central System Proxy Server.

D. Presentation layer

The layer implements Human Machine Interface (HMI) for developing SCADA systems. This is layer of the highest level and provides a graphical interface for system operators

[1]Hristo Nenov is with the Faculty of Computer Science and Automation at Technical University - Varna, 1 "Studentska"str., Varna 9000, Bulgaria, E-mail: h.nenov@tu-varna.bg.

through which they can monitor the working condition, to read different measurements, to oversee occurred events, alarms and archival data

E. Data layer

The layer implements mechanisms for temporary and permanent storage of system data. It covers two aspects:

- Temporary storage of data within the RAU (individual small farms)
- Permanent storage of data source RAU on a main server.

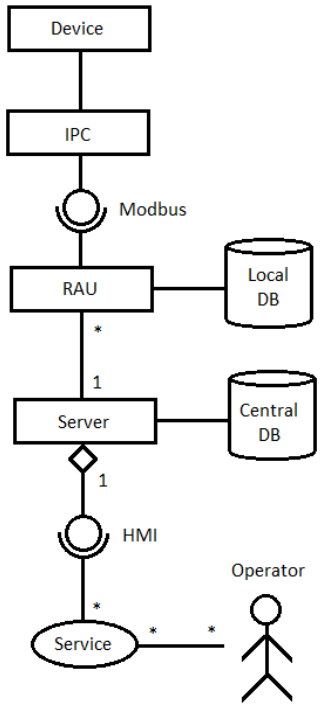


Fig.1. General architecture of the system.

III. REALIZATION OF THE SYSTEM

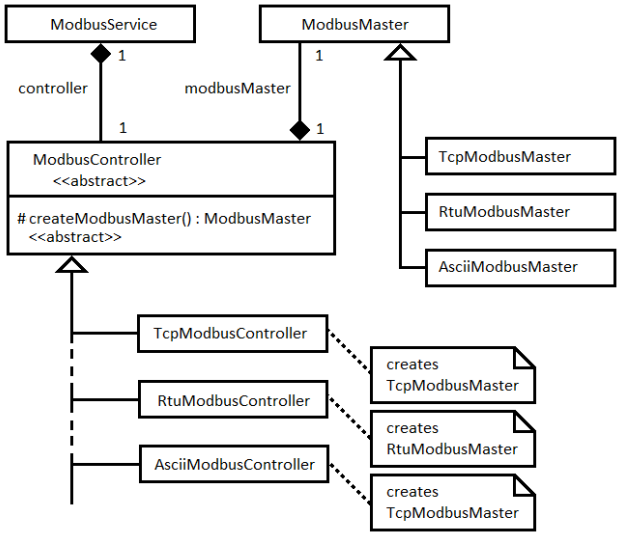
Each Remote Automation Unit (RAU) is designed to operate in unreliable network connectivity to the central server, but it should provide reliable and trouble-free autonomous management of the production process based on the current configuration. In implementation ScheduledExecutor Service (Java 1.7+) is used to start services at regular configurable intervals:

- Activities of gathering information from the sensory part of the system, primary processing and storage of information in a local database
- Logic of autonomous management of physical devices (pumps, valves, relays, etc.) based on a current configuration to ensure the needs of production
- Generate and store events and alarms;
- Send the collected data, events and alarms to the central server.

Data collection (Snapshots) and their replication to a central server are performed by the SnapshotProducerService and SnapshotConsumerService, working in parallel on RAU and implementing the principle Producer / Consumer.

The exchange service for MODBUS contains abstract controller class Modbus Controller, which implements operations via MODBUS composite instance of ModbusMaster - base library class in modbus4j. The heirs of ModbusController implement different MODBUS modes - TCP, RTU, ASCII, as appropriate instantiation type - successor to ModbusMaster.

Fig.2. Service and Controller for data exchange over MODBUS.



Updating the status of RAU (Remote Automation Unit).

The internal state of RAU is managed by the class DeviceManager, which maintains a collection of devices – "Device", which represent the inputs and outputs of the PLC. Synchronization of values with those of the registers of the PLC, is carried out by ModbusService, which uses class ModbusController, encapsulating work with MODBUS. ModbusService and DeviceManager sharing data for the state through class Snapshot as the exchange is initiated only from ModbusService.

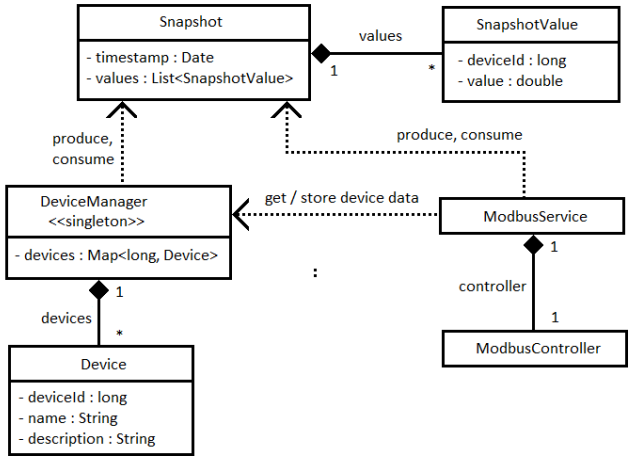


Fig.3. Updating of state of RAU.

Model of automatic control of RAU.

Automatic control is realized by class AutomationController, which applies logical chains of rules. Each type of rule implements specific logic. This mechanism realizes design pattern "*Chain of responsibility"* and relies on polymorphism technique.

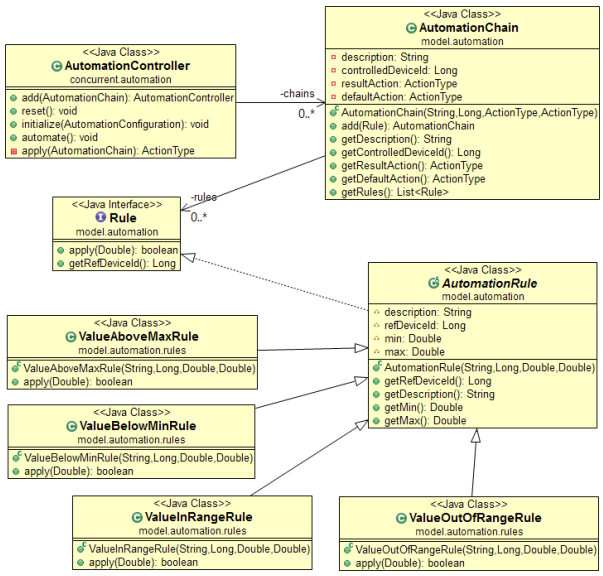


Fig.4. Model of automatic control of RAU

IV. TESTING OF THE SYSTEM

For testing purposes of the developed SCADA systems, experimental arrangement that reflects the structure of a small autonomous farm RAU and a central server are realized. The experimental setting is made up of the following components:

- Central server on which the application runs RauServer. Implemented on a separate computer that is running Apache Tomcat with tuned Jersey Servlet, REST service requests from RAU. On the same machine instance is running for MySQL database on which stored data are received from RAU;
- RAU (Remote Automation Unit), which was launched as a standalone application (service) on a separate computer. On the same machine is running instance of MySQL database on which the data is temporarily stored until it is sent to them to the central RauServer;
- MODBUS Controller.

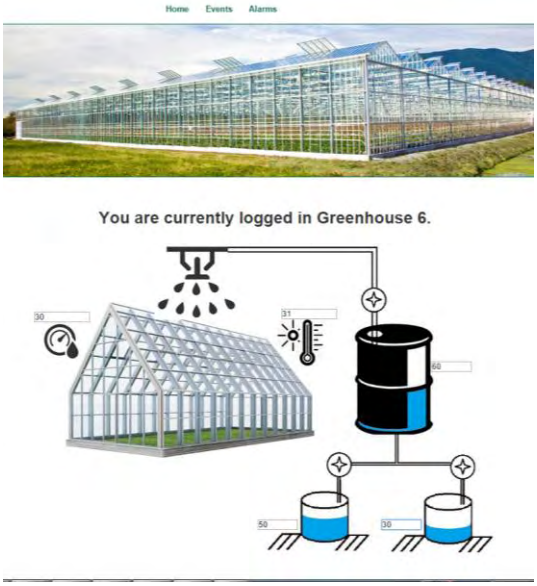


Fig.5. Monitoring of sensors and deices in real time.

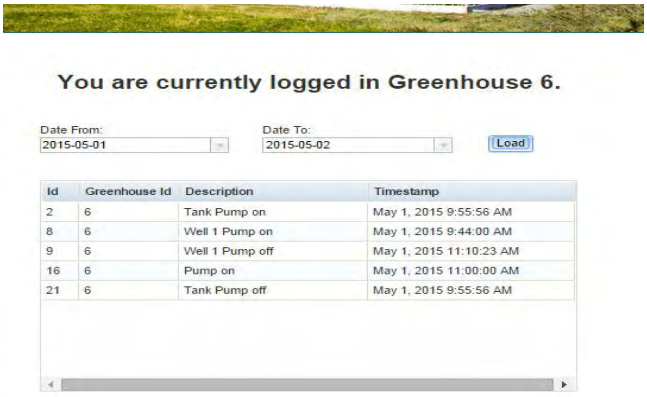


Fig.6. Monitoring of events in real time.

V. CONCLUSION

The system is fully operational. All modules are working properly, generating and exchanging correct data. The main objective of the study is achieved. There are a few aspects, which are provided for future development.

REFERENCES

[1] David Geary, Cay S. Horstmann "Core JavaServer Faces" 3rd.Edition,May.2010.
[2] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides "Gang Of Four - Design Patterns, Elements Of Reusable Object Oriented Software - Addison Wesley
[3] Kevin Mukhar, Chris Zelenak, James L. Weaver, Jim Crume JavaEE 5, From Novice to Professional
[4] Leonard Richardson, Sam Ruby –RESTful Web Services 2007.
[5] www.modbus.org