УДК 681.3

В.В. Кангин, Д.Н. Ямолдинов

SCADA ДЛЯ РАСПРЕДЕЛЕННОЙ СИСТЕМЫ УПРАВЛЕНИЯ НА ОСНОВЕ ПРОМЫШЛЕННОЙ СЕТИ PLCNET

Арзамасский политехнический институт (филиал) НГТУ им. Р.Е. Алексеева

Рассматриваются вопросы проектирования SCADA для распределенной системы управления (РСУ), построенной на основе промышленной сети PlcNet. Показано, что при проектировании SCADA-систем проблема обмена информацией между контроллерным и диспетчерским уровнями РСУ является центральной. Решение этой проблемы лежит в плоскости организации клиент-серверных отношений между SCADA-системой и OPC-сервером, поставляемым разработчиком сетевой аппаратуры. В этом случае проблемы обмена сводятся к информационному обмену между SCADA-системой, играющей роль OPC-клиента, и OPC-сервером. Разработаны алгоритмы просмотрщика тегов, позволяющего визуализировать значения 32 тегов, принятых по сети из контроллеров нижнего уровня, а также процедур и функций, обеспечивающих следующие возможности просмотрщика тегов: запуск OPC-сервера, добавление в него имени сегмента и имен тегов, определение типа тегов и вывода значений тегов в окна на форме просмотрщика тегов, останов OPC-сервера.

Ключевые слова: распределенная система управления, промышленная сеть, SCADA-система, клиент-серверные отношения, OPC-клиент, OPC-сервер.

Постановка задачи

В настоящее время наблюдается качественный скачок в развитии систем управления технологическим оборудованием и технологическими процессами. Системы управления стали децентрализованными, распределенными. Таким образом, в настоящее время можно говорить о распределенных системах управления.

Промышленные сети представляют собой физическую среду, на базе которой реализуются РСУ. На нижнем уровне этой структуры находятся контроллеры – устройства, осуществляющие непосредственное управление технологическими процессами в реальном времени. Контроллер должен решать задачу управления быстро, в темпе протекания технологического процесса.

На верхнем уровне РСУ расположены рабочие станции, позволяющие выполнять следующие задачи:

- сбор данных о параметрах технологического процесса, их обработка и архивирование принятой информации;
- визуализация информации о ходе технологического процесса, сигнализация о предаварийных и аварийных ситуациях;
- формирование сводок, журналов, отчетов и т.д.;
- формирование команд оператора по изменению параметров настройки и режимов работы контроллеров;
- автоматическое управление ходом технологического процесса (только в ряде случаев и для решения задач невысокого быстродействия).

Рабочие станции верхнего уровня представляют собой автоматизированные рабочие места (АРМ) производственного персонала, обслуживающего технологический процесс: операторы, технологи, наладчики, диспетчеры и т.д. Верхний уровень РСУ называют *диспетчерским,* или *операторским,* уровнем. Диспетчерский уровень представлен персональными компьютерами (ПК), рабочими станциями, промышленными компьютерами и т.д.

Контроллеры, решая задачу непосредственного управления технологическим оборудованием, принимают сигналы с датчиков и выдают управляющие сигналы на исполнитель-

ные механизмы. Одновременно с этим они решают еще одну важную задачу: передают информацию о состоянии технологического процесса на компьютеры верхнего диспетчерского уровня. В ряде случаев контроллеры могут обмениваться данными и между собой. Промышленная сеть и есть та физическая среда, по которой происходит обмен информацией между активными компонентами (узлами) РСУ.

Каждый технологический параметр представлен в сети в виде некоторой переменной – тега, который характеризуется величиной, качеством передачи (приема) по сети, типом и т.д.

Выделение в РСУ двух уровней позволяет вести речь о программном обеспечении (ПО) для верхнего и для нижнего уровня. На нижнем уровне широко используются специализированные языки программирования по стандарту IEC 61131-3. К таким языкам, принадлежащим к группе языков Functional Block Diagrams (FBD) – функциональных блоковых диаграмм, относится язык, входящий в систему программирования контроллеров UltraLogik. Этот язык позволяет легко выполнить проектирование управляющих программ высокой степени сложности. Какие бы алгоритмы не реализовала управляющая программа контроллера, важно, чтобы она еще передавала определенную нужную информацию в сеть. В UltraLogik это делается просто. Достаточно той или иной переменной (параметр технологического процесса, состояние того или иного датчика, управляющий сигнал и т.д.) присвоить статус «сетевая», как она гарантированно будет периодически передаваться контроллером в сеть.

ПО верхнего уровня реализуется в виде SCADA-системы (Supervisory Control and Data Acquisition – система сбора данных и оперативного диспетчерского управления). Задачи, которые выполняют SCADA-системы, были обозначены ранее, но центральной является задача по приему тегов из сети. Для этих целей используются специальные приложения (программы), называемые OPC-серверами. OPC- сервер считывает значения тегов из сети и передает их приложению – клиенту. SCADA-системы являются клиентами OPC-серверов [1].

Высокая стоимость существующих SCADA-систем, отсутствие качественной технической документации, избыточность, за которою приходится платить пользователю, делают процесс внедрения SCADA-систем в производство весьма непростым. Однако проектирование SCADA-систем можно выполнить с помощью таких программных средств, как Delphi, Visual C++, VBA. Они имеют в своем составе все необходимые средства для работы с OPC – серверами, графикой, мультипликацией, базами данных и т.д.

Модель РСУ

Многоуровневая модель современной РСУ приведена на рис. 1. Ей можно поставить в соответствие модель сетевого обслуживания диспетчерского и контроллерного уровней РСУ (рис. 2).

Контроллерный уровень обслуживается одной из промышленных сетей на основе протоколов CAN, PLCNET, PROFIBUS и других. Верхний, диспетчерский, уровень, как правило, обслуживается локальной сетью на основе протоколов Ethernet, TokenRing, ARCNet и других.

Один или несколько ПК в такой модели должны выполнять роль шлюза, т.е. обеспечивать передачу информации из сети одного типа в сеть другого типа.

На рис. 3 приведена более подробная реализация РСУ. Нижний, контроллерный, уровень объединяет N контроллеров с помощью промышленной сети PlcNet. Верхний диспетчерский уровень объединяет ряд ПК с помощью сети Ethernet. Среди этих ПК один должен выполнять роль мастера (Master) для сети PlcNet, т.е. он должен иметь возможность принимать теги из сети PlcNet от контроллеров.

Для сети типа PlcNet в качестве контроллеров можно использовать PC-base контроллеры ADAM-5510. Сеть PlcNet организована на базе интерфейса RS-485 и реализуется в виде сегментов. Каждый из сегментов связан с одним из COM-портов ПК-мастера. Для преобразования сигналов интерфейса RS-232 в сигналы интерфейса RS-485 используется преобразователь интерфейсов ADAM-4520.

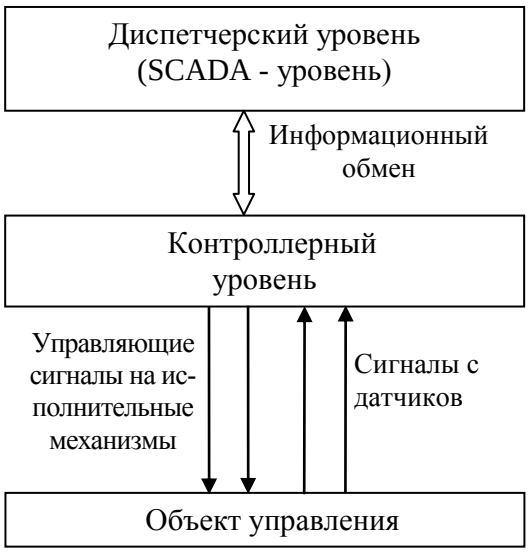

Рис. 1. Многоуровневая модель современной РСУ

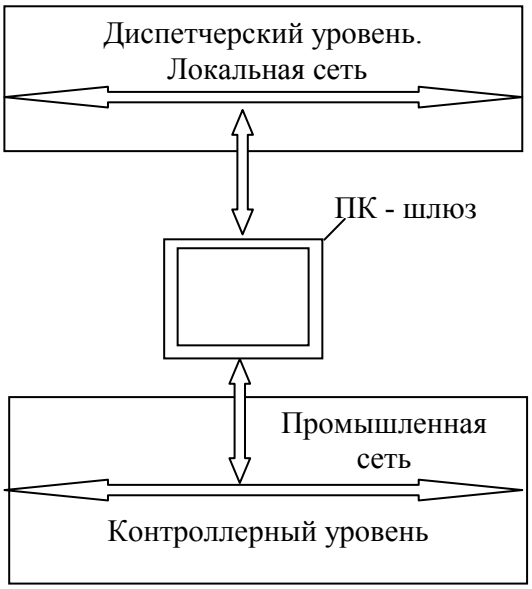

Рис. 2. Модель сетевого обслуживания диспетчерского и контроллерного уровней РСУ

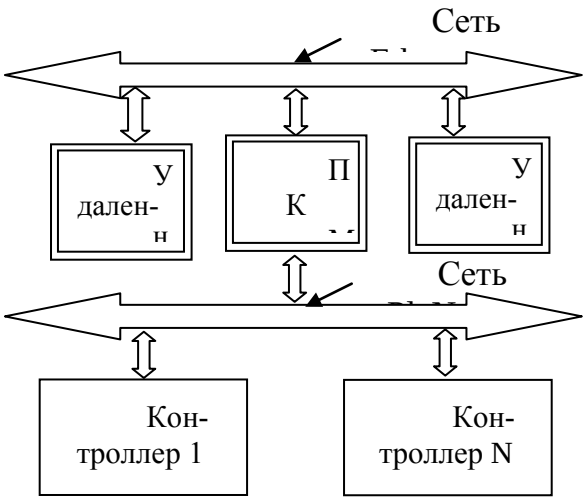

Рис. 3. РСУ на основе промышленной сети PlcNet

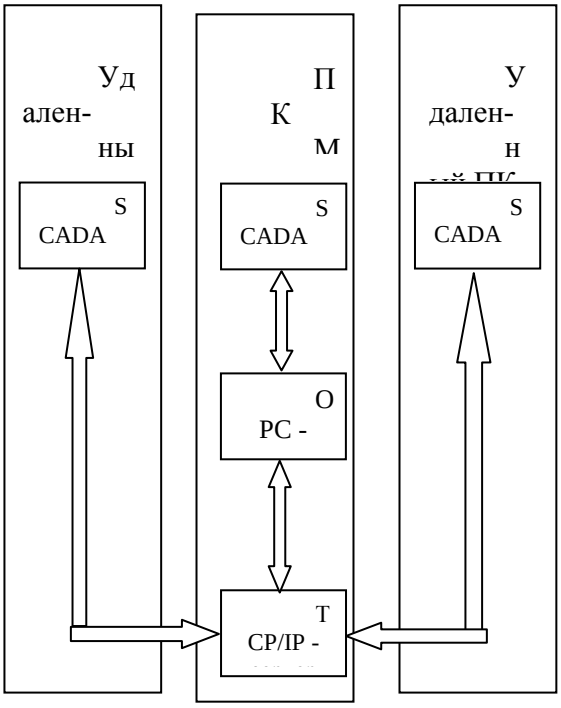

Рис. 4. Взаимодействие программного обеспечения компьютеров диспетчерского уровня

Максимальное количество контроллеров, подключенных к одному сегменту сети, равно 32. ПК-мастер принимает информацию с контроллеров по сети PlcNet и затем может ее передавать по сети Ethernet удаленным ПК, образующим диспетчерский уровень.

На рис. 4 показано взаимодействие программного обеспечения компьютеров диспетчерского уровня в части передачи информации, поступившей от контроллера в ПК-мастер по сети PlcNet.

Прием этой информации из сети выполняется с помощью специальной программы OPC-сервера.

SCADA-система, расположенная на ПК-мастере, взаимодействует с OPC-сервером напрямую. SCADA-системы, расположенные на других удаленных ПК сети Ethernet, получают информацию из OPC-сервера через другую программу (TCP/IP-сервер), которая должна располагаться на ПК-мастере. Для сети PlcNet широко используется OPC-сервер фирмы Fastwel – Fastwel PLCNet OPC Server. Этот сервер позволяет не только читать информацию из контроллеров, но и передавать ее в обратном направлении – из SCADA-системы в контроллеры РСУ.

Организация SCADA-системы

Рассмотрим более подробно процесс приема информации из контроллеров РСУ в SCADA-систему ПК-мастера. Для этого SCADA-систему реализуем в виде просмотрщика тегов – переменных сети PlcNet [2]. Внешний вид рабочего окна просмотрщика тегов показан на рис. 5.

Алгоритм работы просмотрщика тегов приведен на рис. 6. На рис. 7 показан алгоритм функции *GetItemType*, которая возвращает тип тега по его значению ItemValue.

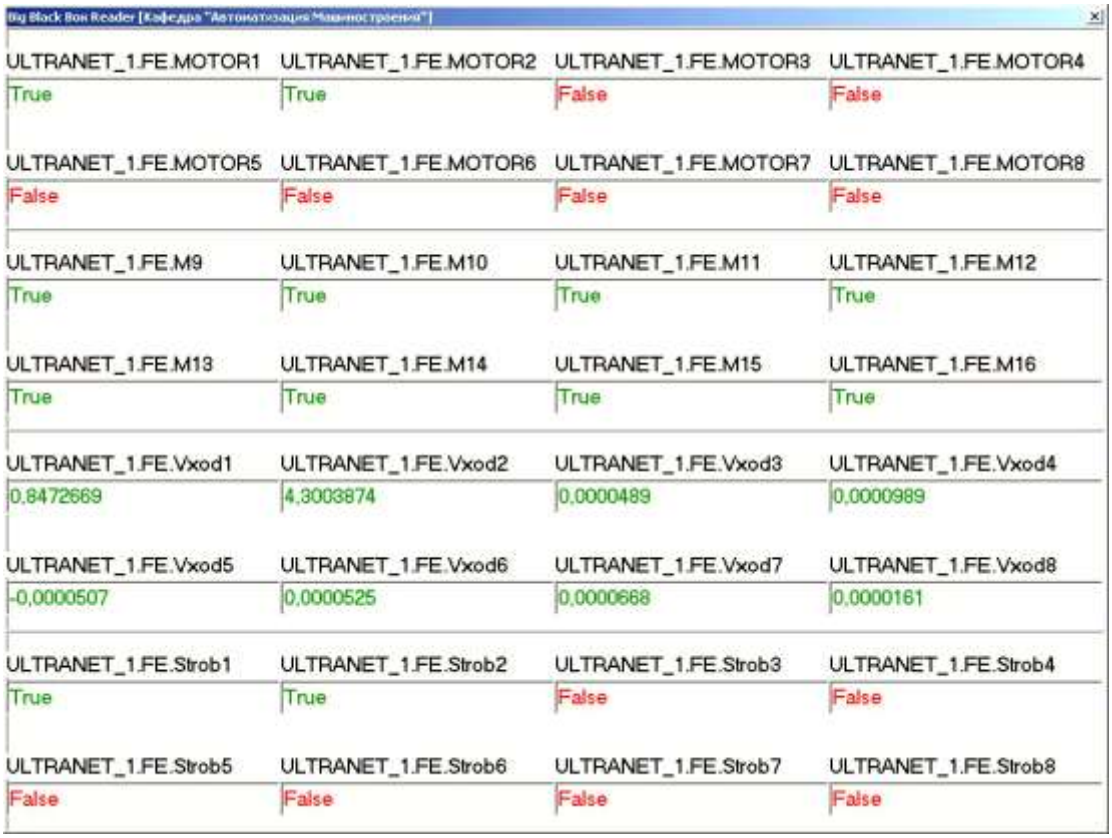

Рис. 5. Окно просмотрщика тегов

Важным обстоятельством при работе всего просмотрщика тегов является то, что вся информация, которую выдает OPC-сервер, представлена в строковом формате *string* [3]. Функция *GetItemType* используется в процедуре *TMainForm.Timer1Timer*.

Сначала (блок *2*) типу тега присваивается значение Timer. Функция формирует результат в виде строки. Затем (блок *3*) анализируется значение *ItemValue* на равенство True или *False*. Эти два значения может принимать только переменная или константа типа *Boolean*. При выполнении одного из этих условий, типу тега присваивается значение *Boolean*.

Если тег не имеет тип *Boolean*, то начинается посимвольная проверка ItemValue (блоки *5*, *6*, *7*, *8*). Если в результате этой проверки оказывается, что *i*-й символ принимает значение «.» или «,», то типу тега сразу присваивается значение *Float* – действительное число (блок *9*). Точка или запятая являются обязательными разделителями в действительных числах.

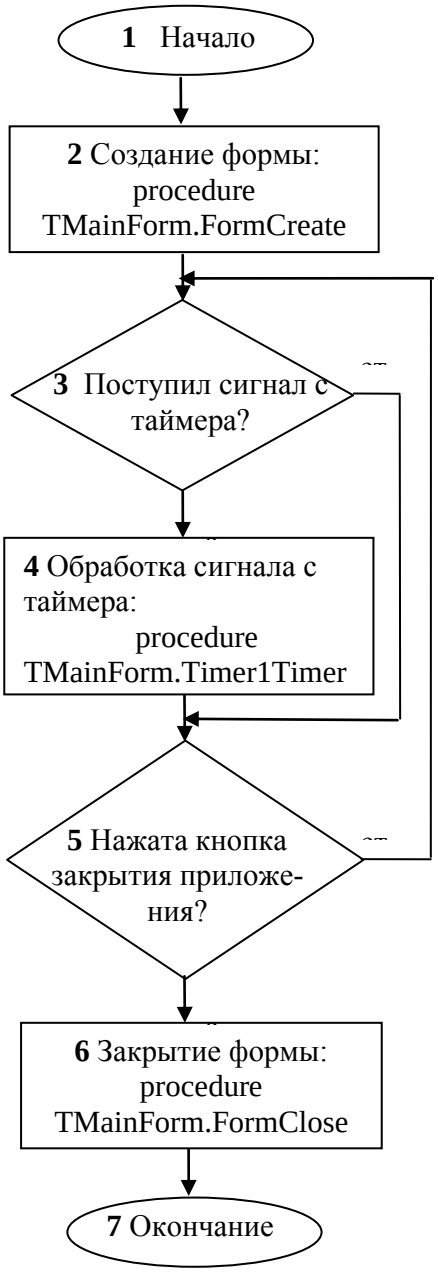

Рис. 6. Алгоритм работы просмотрщика тегов

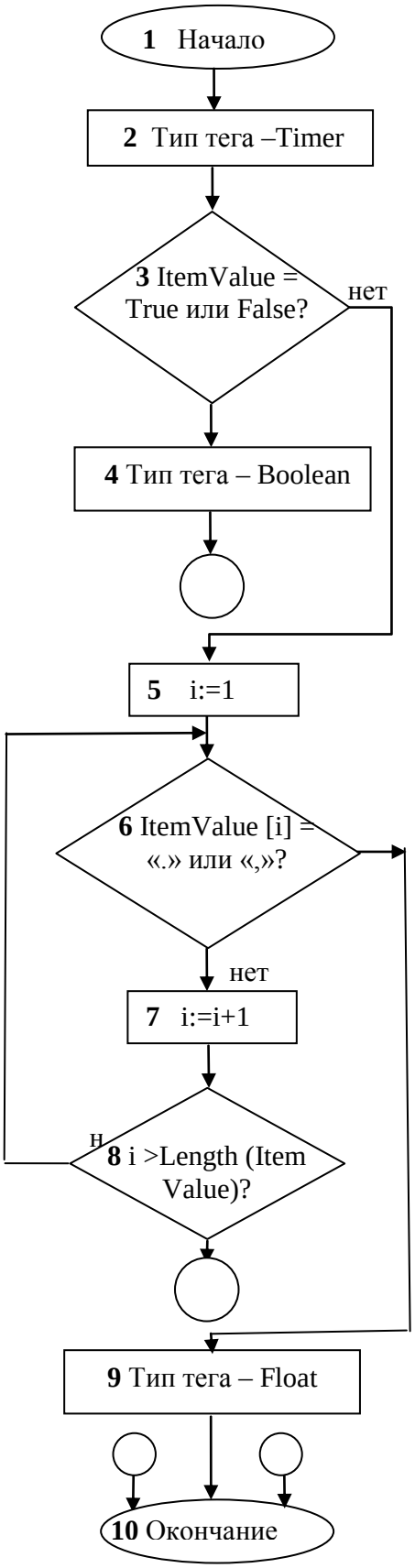

Рис. 7. Алгоритм функции GetItemType

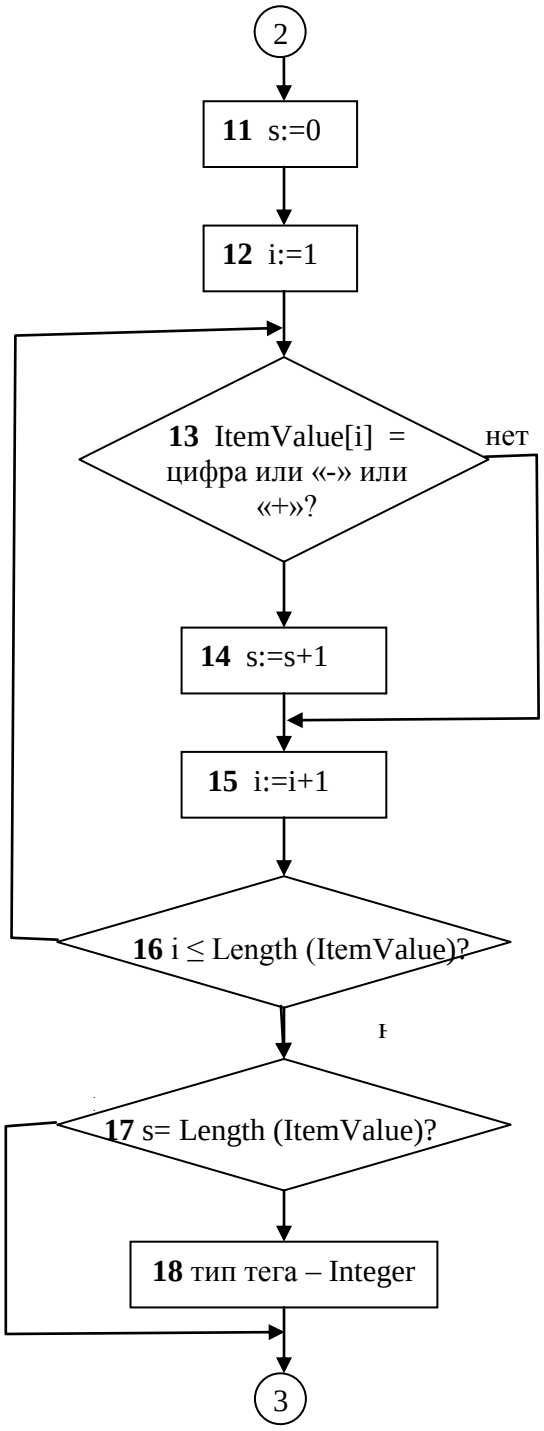

Рис. 7. (Окончание) Алгоритм функции GetItemType

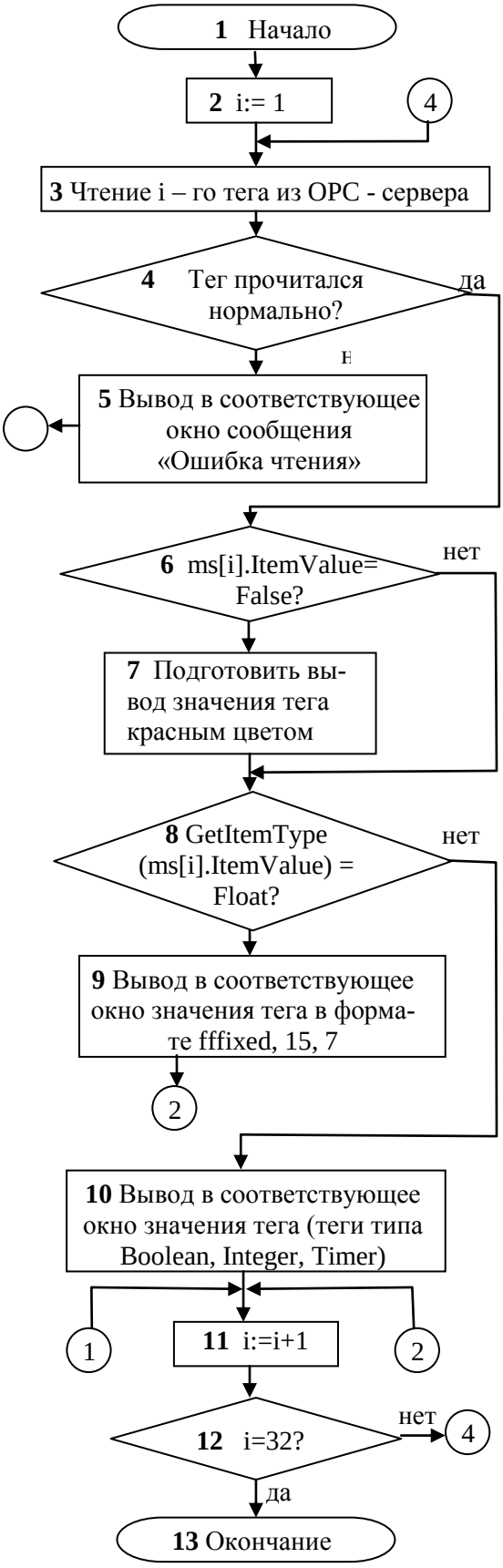

Рис. 8. Алгоритм процедуры TMainForm.Timer1Timer

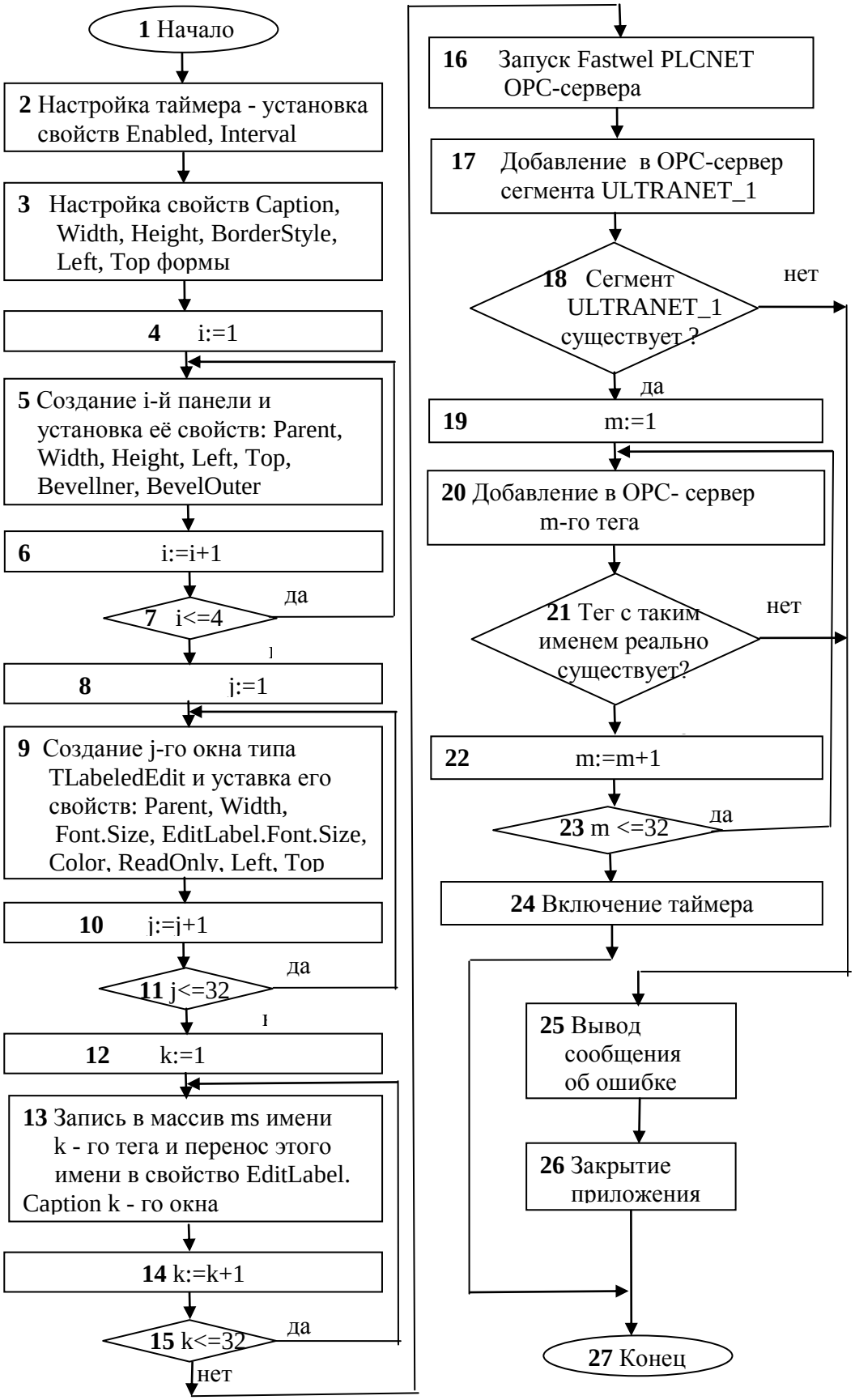

Рис. 9. Алгоритм процедуры TMainForm.FormCreate

Если точка или запятая в значении *ItemValue* от сутствуют, то начинается его проверка на тип Integer – целое число (блоки *11, 12, 13, 14, 15, 16, 17*). Признаком целого числа является наличие в нем лишь цифр от 0 до 9 и знаков «+» или «–». Естественно, что все эти знаки представлены символами в строковом выражении.

В данном фрагменте алгоритма: *i* – номер символа в строке, *s* – число искомых символов в строке. Если в результате поиска оказалось, что все символы в строке относятся к цифрам или к знакам «+» или «–», то делается вывод, что тег имеет тип *Integer* (блок *18*).

Процедура TMainForm.Timer1Timer является основной в приложении-просмотрщике тегов. Алгоритм процедуры приведен на рис. 8. Чтение тегов здесь выполняется циклически. После чтения *i*-го тега из OPC-сервера сразу выполняется проверка на правильность его чтения, что выполняется с помощью функции Succeded (HR) (блок *4*). Если тег прочитался неправильно, то в соответствующее окно формы выводится сообщение «Ошибка чтения» (блок *5*). Далее проверяется (блок *6*), равен ли тег значению *False* (тип *Boolean*). Если он имеет значение *False*, то подготавливается вывод его значения красным цветом (блок *7*). Затем (блок *8*) тип тега проверяется на *Float* –действительное число. Для проверки используется функция *GetItemType*. Если тег имеет тип *Float*, то его значение выводится (блок *9*) в соответствующее окно в формате *fffixed, 15, 7*. Если тег не относится к типу *Float*, то его значение выводится в соответствующее окно (блок *10*). Блоки *1, 11* и *13* предназначены для организации циклического вывода значений тегов в соответствующие окна формы просмотрщика тегов. Здесь параметр *i* =1..32 – номер тега. Процедура *TMainForm.FormClose* содержит всего один оператор Server:=nil, что приводит к выключению OPC-сервера.

Алгоритм процедуры *TMainForm. FormCreate* приведен на рис. 9. Настраиваются свойства таймера и формы (блоки *2* и *3*). Затем (блоки *4, 5, 6, 7*) выполнятся создание четырех панелей и установка их свойств. Далее (блоки 8, 9, 10, 11) выполняется создание 32 окон типа *TLabeledEdit*, размещение их по 8 в каждой из четырех панелей, установка их свойств.

Затем (блоки *12, 13, 14, 15*) в массив *ms* заносятся имена всех тегов. Эти же имена выводятся над соответствующими окнами типа *TLabeledEdit*.

В блоке *16* осуществляется запуск OPC-сервера Fastwel PLCNet OPC Server. В блоке *17* выполняется добавление в OPC-сервер сегмента ULTRANET_1.Если такой сегмент реально существует (блок *18*), то OPC-сервер имеет об этом информацию и готов с ним работать. В противном случае осуществляется вывод соответствующего сообщения об ошибке (блок *25*) и закрытие приложения (блок *26*).

Если сегмент ULTRANET_1 реально существует, то выполняется переход к блоку *19*. В блоках *19, 20, 21, 22, 23* осуществляется добавление в OPC-сервер 32 тегов из массива *ms*. Имя каждого тега уникально и должно совпадать с именем реально существующего тега, циркулирующего по сети PLCNET. Если хотя бы в одном из имен имеется несоответствие, выполняется вывод соответствующего сообщения об ошибке (блок *25*) и закрытие приложения (блок *26*).

Если все 32 тега реально существуют, то процедура заканчивает свою работу включением таймера (блок *24*).

Выводы

1. Показано, что одной из центральных проблем, решаемых при проектировании SCADA-систем, является проблема передачи информации между уровнями РСУ.

2. Определено, что проблема передачи информации из промышленной сети PlcNet на диспетчерский уровень РСУ может быть решена путем использования клиент-серверной архитектуры. В качестве OPC-клиента выступает SCADA-система, а в качестве OPC-сервера может быть использован Fastwel PLCNet OPC Server.

3. С использованием среды визуального программирования Delphi разработана SCADA-система, реализованная в виде просмотрщика тегов типов *Boolean, Float, Integer, Timer*.

4. Рассмотрены механизмы взаимодействия SCADA-системы и OPC-сервера: запуск OPC-сервера, добавление в него имени сегмента и имен тегов, определение типа тегов и вывода значения тегов в окна на форме просмотрщика тегов. Приведены алгоритмы соответствующих процедур и функций.

Библиографический список

1. **Анашкин, А.С.** Техническое и программное обеспечение распределенных систем управления / А.С. Анашкин, Э.Д. Кадыров, В.Г. Харазов. – СПб.: «П–2», 2004. – 368 с.
2. **Кангин, В.В.** Программные аспекты проектирования SCADA-систем / В.В. Кангин, М.В. Кангин, Д.Н. Ямолдинов; Нижегород. гос. техн. ун-т. – Н. Новгород 2007. – 259 с.
3. **Кангин, В.В.** Проектирование SCADA-систем / В.В. Кангин, М.В. Кангин, Д.Н. Ямолдинов; Нижегород. гос. техн. ун-т. – Н. Новгород:, 2010. – 568 с.

Дата поступления
в редакцию 11.10.2011

V.V Kangin, D.N. Yamoldinov

SCADA FOR DISTRIBUTED CONTROL SYSTEM ON BASE OF THE INDUSTRIAL NETWORK PLCNET

The Article is dedicated to questions of the designing SCADA - a systems for distributed control system (DCS), built on base of the industrial network PlcNet. It Is Shown that when designing SCADA - a systems problem exchange by information between controller level and traffic manager level DCS is central. The Decision of this problem lies in planes of the organizations client - a server relations between SCADA - a system and OPC - a server, delivered by developer of the network equipment. In this case problems of the exchange are reduced to information exchange between SCADA - a system, playing role OPC - a client, and OPC - a server. The Designed algorithms observer tags allowing observe importances 32 tags, taken on network from controller lower level, as well as procedures and function, providing following possibilities observer tags: start OPC - a server, accompaniment in it name of the segment and names tags, determination of the type tags and conclusion of importances tags in window on the form observer tags, stop OPC- server.

Key words: distributed control system, industrial network, SCADA - a system, client - a server relations, OPC - a client, OPC - a server.