Сбор данных в OpenSCADA.

Оглавление

Введение

Сбор данных SCADA(Supervisory Control and Data Acquisition)-системы является её неотъемлемой частью, которая занимается получением данных из источников различного происхождения. Природа данных с которыми работает SCADA характеризуется сигналами базовых типов значений (целое, вещественное, логическое и строка). Сигналы изменяются во времени и обладает историей, жизнью. В теории управления технологическими процессами (ТП) под сигналом понимается значение датчика установки ТП в коде АЦП, 'сырой' сигнал, или в реальном значение. Сигналы могут объединяться в группы по смысловой нагрузке, часто называемые параметрами. Например, развитые источники данных могут предоставлять структуры параметров с предопределённым набором связанных сигналов. Кроме непосредственного сбора данных в функции этого механизма также входит и передача воздействий на исполнительные устройства управления ТП, обычно это задвижки, насосы и регулирующие клапана. В совокупности этот процесс получил название Устройство Сопряжения с Объектом (УСО).

Источники данных характеризуются большим разнообразием, которое можно условно разделить на три группы:

- Источники 'сырых' данных, предоставляющие код АЦП или уровни дискретных сигналов, а так-же включающие простейшую обработку. Обычно это модули рассосредоточенного УСО или простейшие промышленные программируемые логические контроллеры (ПЛК).
- Мощные промышленные ПЛК, обладающие значительной вычислительной мощностью и возможностью формирования сложных параметров с различной структурой.
- Локальные или сопутствующие источники данных. Например, УСО в виде плат расширения, а также данные аппаратного и программного окружения в котором функционирует система.

Разнообразие источников данных породило большой спектр механизмов доступа к ним. Локальные источники данных различаются интерфейсами программирования приложения (API), а сетевые источники, в свою очередь, транспортным и протокольным уровнями взаимодействия. В целом это привело к тому, что добавления поддержки нового источника данных требует создание модуля сопряжения или драйвера, а учитывая большое разнообразие это крайне накладно и фактически нереально охватить весь спектр рынка таких устройств. Ситуация несколько упрощается с сетевыми источниками, благодаря наличия ряда стандартных и свободных протоколов взаимодействия, однако многие источники всё-же используют собственные протоколы, закрытые коммерческие или протоколы завязанные на закрытые механизмы ограниченного круга коммерческих операционных систем (ОС).

В терминах системы OpenSCADA предоставляются следующие объекты для обслуживания механизма сбора данных:

- Атрибут — Объект отражения данных сигнала, включает текущее значение с типом сигнала и историю изменения значений.
- Параметр — Объект группы атрибутов (сигналов) со структурой соответствующей особенностям отдельно взятого источника данных.
- Контроллер – Объект отдельного устройства данных. Как правило это отдельный модуль УСО или устройство промышленного ПЛК.

Для учёта особенностей различных устройств сбора данных, а также различных механизмов взаимодействия в OpenSCADA предусмотрена подсистема 'Сбор данных', которая является модульной. В качестве модуля подсистемы выступает драйвер для сопряжения с источником данный отдельного типа. Каждый модуль может содержать конфигурацию нескольких устройств этого типа в виде объектов 'Контроллер' системы OpenSCADA. Общая схема объектов подсистемы 'Сбор данных' изображена на рис.1.

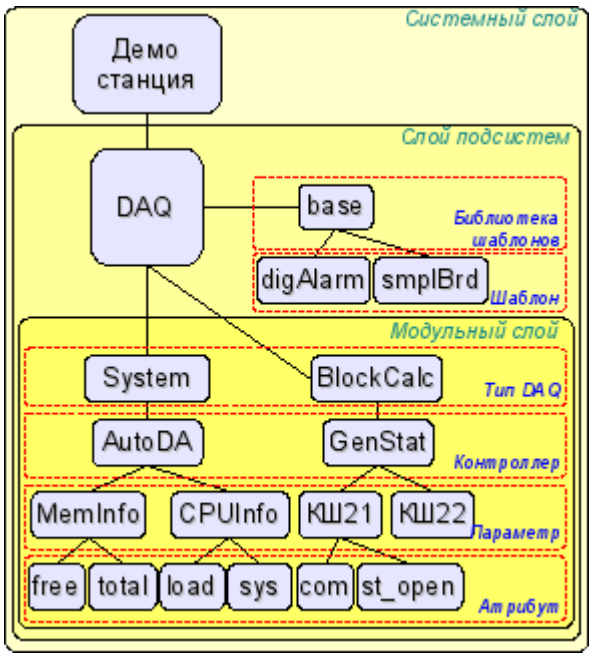

Рис. 1. Схема подсистемы 'Сбор данных'.

1. Методы сбора данных

Учитывая различие свойств источников данных, а также возможные варианты взаимодействия, методы сбора данных можно разделить на: простой синхронный, простой асинхронный, пакетный и пассивный.

В рассмотрении механизмов, ниже, будут участвовать следующие объекты:

- ObjectSCADA — любой объект SCADA-системы обращающийся за значением сигнала, например архивы и визуализаторы.
- DAQParamAttribute — объект атрибута параметра подсистемы 'Сбор данных', выступающий посредником в доступе к значению сигнала источника данных.
- DAQParamAttributeArch — объект архива атрибута параметра подсистемы 'Сбор данных'.
- HardwarePLC — объект источника данных, например модули рассосредоточенного УСО или промышленные ПЛК.

1.1. Простой синхронный механизм сбора

Механизм характеризуется запросами к источнику данных синхронно с запросом к атрибуту параметра (рис.2). Данный механизм обычно применяется при работе с локальными источниками данных, характеризующимися низкой латентностью т.е. задержкой в ответе на запрос. С помощью этого метода можно получить актуальные данные непосредственно с запросом, однако время запроса объекта будет включать время транспортировки и обработки запроса источником данных.

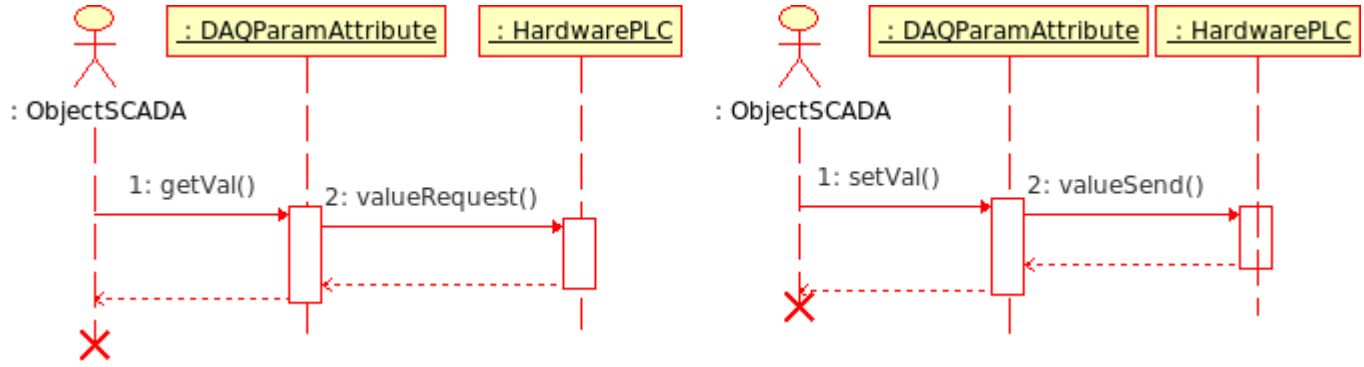

Рис. 2. Диаграмма последовательности взаимодействия при синхронных запросах.

В соответствии с диаграммой выше мы получаем следующую последовательность запросов получения данный и их передачи:

- Объект SCADA-системы шлёт запрос значения к объекту атрибута параметра DAQParamAttribute::getVal().
- Объект атрибута параметра получив запрос шлёт его источнику данных HardwarePLC::valueRequest().
- Источник данных, обработав запрос, возвращает результат.
- Объект атрибута параметра получив результат возвращает его объекту SCADA-системы.

В OpenSCADA такой механизм реализуют следующие модули подсистемы 'Сбор данных':

- *ModBus* — модуль доступа к данным источников посредством семейства протоколов ModBus. В модуле реализован синхронный режим для записи данных.
- *DiamondBoards* — модуль доступа к данным PC/104 плат фирмы Diamond Systems. Платы PC/104 размещаются на ISA-шине, следовательно являются локальными и доступны сравнительно быстро. В режиме сбора данных не по прерыванию доступ к значениям АЦП осуществляется синхронно. Режим записи значения ЦАП всегда работает синхронно.
- *Transporter* — модуль отражения объектов контроллеров удалённых OpenSCADA-станций на локальную. В модуле реализован синхронный режим для записи данных.
- *BlockCalc* — вычислитель на языке блочных схем. В качестве источника данных в нём выступает пользовательская блочная схема. Атрибуты параметров модуля синхронно обращаются к входам/выходам блоков блочной схемы.
- *JavaLikeCalc* — вычислитель на Java-подобном языке высокого уровня. В качестве источника данных в нём выступает пользовательская программа на Java-подобном языке. Атрибуты параметров модуля синхронно обращаются к входам/выходам пользовательской вычислительной функции.
- *LogicLev* — модуль логического уровня параметров сбора данных, детальнее о нём в разделе 2. В качестве источника данных этого модуля выступают другие параметры подсистемы 'Сбор данных' и контекст исполнения шаблона параметров. Атрибуты параметров модуля синхронно обращаются к атрибутам других параметров, в режиме отражения параметров подсистемы 'Сбор данных', или к входам/выходам контекста исполнения шаблона, в режиме работы по шаблону.

1.2. Простой асинхронный механизм сбора

Механизм характеризуется запросами к источнику данных независимо от запроса к атрибуту параметра (рис.3). Обычно запросы к источнику данных осуществляются периодически в собственной задаче опроса отдельно взятого контроллера и блоками по несколько сигналов. При этом, запросом к атрибуту параметра возвращается значение полученное последним сеансом связи с источником данных. Данный механизм обычно применяется при работе с удалёнными (сетевыми) источниками данных, характеризующимися высокой латентностью т.е. задержкой в ответе на запрос.

С помощью этого метода можно обеспечить оптимизацию временного ресурса затраченного на один сигнал и тем самым увеличить максимальное количество опрашиваемых сигналов за интервал времени опроса.

В качестве примера рассмотрим промышленный ПЛК Siemens S7–315 при опросе его по шине Profibus (1.5Мбит). Среднее время обработки MPI-запроса этим контроллером составляет 30мс. Если использовать синхронный механизм для каждого сигнала, т.е. один запрос на каждый сигнал, то в течении одной секунды мы сможем получить около 33 сигналов. А если применить асинхронный механизм, т.е. в одном MPI-пакете получать до 220байт или 110 сигналов целочисленного типа на 16-разрядов, то мы сможем за одну секунду получить до 3630 сигналов. Как можно видеть эффективность асинхронного механизма, в данном случае, составляет 110 раз, а именно значение максимальной ёмкости MPI-пакета.

Недостатком асинхронного механизма является то, что запрос значения атрибута параметра возвращает не актуальное, на момент запроса, значение, а значение последнего сеанса опроса контроллера. Впрочем, если учесть, что источник данных может обновляться с периодичностью аппаратных ограничений АЦП, да и сами датчики могут иметь определённые ограничения в скорости реакции, то применение асинхронного механизма сбора может иметь серьёзные основания.

Применение асинхронного механизма для записи значений в ПЛК является достаточно редким явлением поскольку запись значений обычно подразумевает воздействие оператора на ТП, что по факту достаточно редкое явление, которое можно выполнять синхронно. Однако существуют ситуации, например управление ТП регуляторами на SCADA-системе, выполняющей функции среды исполнения ПЛК.

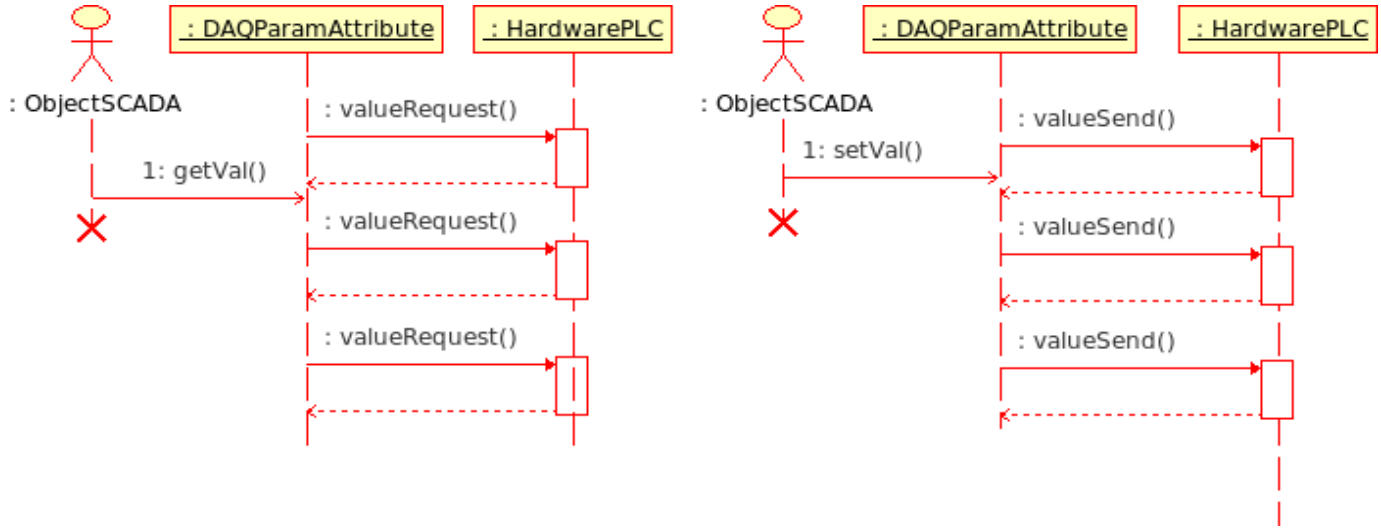

Рис. 3. Диаграмма последовательности взаимодействия при асинхронных запросах.

В соответствии с диаграммой выше мы получаем следующую картину:

- Объект атрибута параметра, или вышестоящий объект контроллера, выполняет периодические запросы HardwarePLC::valueRequest() для получения значения сигнала или группы сигналов.
- Полученные значения сигналов размещаются в объектах атрибутов параметров, локально.
- Объект SCADA-системы шлёт запрос значения к объекту атрибута параметра DAQParamAttribute::getVal() и получает сохранённое локально значение предыдущего сеанса опроса источника данных.

В OpenSCADA такой механизм реализуют следующие модули подсистемы 'Сбор данных':

- *Siemens* — модуль доступа к данным контроллеров фирмы Siemens серии S7. В данном модуле асинхронный режим реализован как для чтения данных так и для их записи (опционально) на ПЛК.
- *ModBus* — модуль доступа к данным источников посредством семейства протоколов Mod Bus. В модуле реализован асинхронный режим чтения данных.
- *SNMP* — модуль доступа к данным устройств сети посредством Simple Network Management Protocol. В модуле реализован асинхронный режим чтения данных.
- *System* — модуль доступа к данным окружения исполнения OpenSCADA. В модуле реализован асинхронный режим чтения данных.
- *Transporter* — модуль отражения объектов контроллеров удалённых OpenSCADA-станций на локальную. В модуле реализован асинхронный режим чтения данных.

1.4. Пакетный механизм сбора

Пакетный механизм сбора данных характерен сбором данных каждого сигнала пакетом, включающим историю его изменения. Т.е. за один сеанс опроса данных получается несколько значений истории сигнала. Пакетный механизм работает совместно с синхронным и с асинхронными механизмами.

В случае работы с синхронным механизмом выполняется фактический проброс архива источника данных для оперативной работы в системе (рис.2). Как и простой синхронный механизм его желательно применять только на низколатентных источниках данных или с источниками работа которых является сеансовой, например в сфере коммерческого учёта для чтения значений счётчиков.

При работе совместно с асинхронным механизмом история полученых сигналов обычно прямо помещается в архивы (рис.4), а текущее значение атрибута параметра устанавливается в последнее значение пакета. Данная комбинация эффективна при сборе быстрых данных или при синхронизации архивов после потери связи с удалённым источником данных.

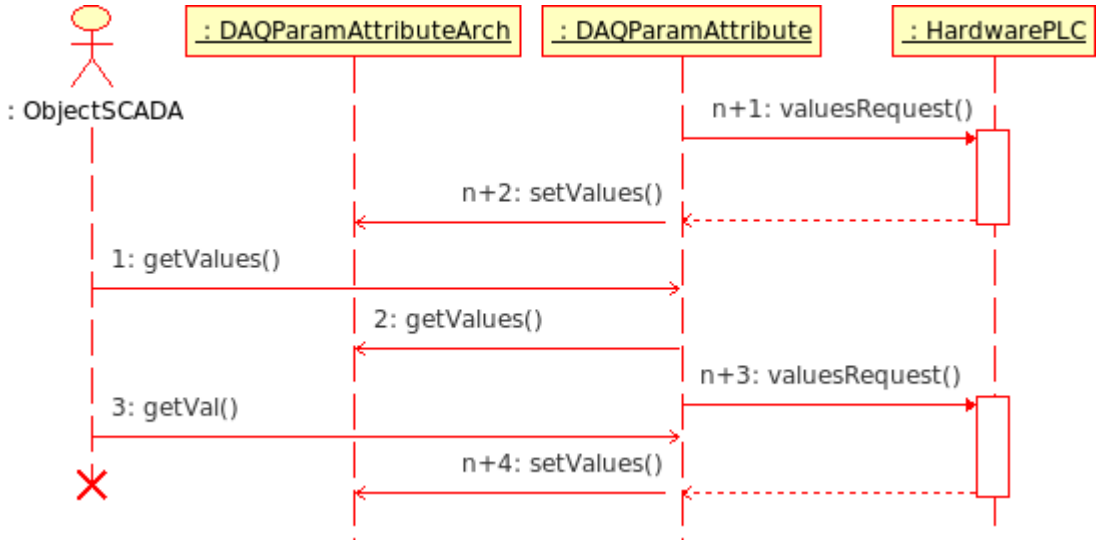

Рис. 4. Диаграмма последовательности взаимодействия при асинхронных запросах пакетного механизма.

В соответствии с диаграммой выше мы получаем следующее поведение пакетного механизма при асинхронных запросах:

- Объект атрибута параметра, или вышестоящий объект контроллера, выполняет периодические запросы HardwarePLC::valuesRequest() для получения пакетов значений сигнала или группы сигналов.
- Полученные пакеты значений сигналов помещаются в архив запросом DAQParamAttributeArch::setValues(), а последнее значение пакетов размещается в объектах атрибутов параметров.
- Объект SCADA-системы шлёт запрос фрагмента архива к объекту атрибута параметра DAQParamAttribute::getValues(), а тот перенаправляет запрос к архиву DAQParamAttributeArch::getValues(). В результате возвращается фрагмент архива доступный после предыдущего сеанса опроса источника данных.
- Объект SCADA-системы шлёт запрос последнего значения к объекту атрибута параметра DAQParamAttribute::getVal() и получает сохранённое локально значение предыдущего сеанса опроса источника данных.

В OpenSCADA такой механизм реализуют следующие модули подсистемы 'Сбор данных':

- *DiamondBoards* — модуль доступа к данным PC/104 плат фирмы Diamond Systems. Платы PC/104 размещаются на ISA-шине, следовательно являются локальными и доступны сравнительно быстро. В режиме сбора данных по прерыванию осуществляется ожидание

пакетов быстрых значений (до 200кГц) за одну секунду (до 200000 значений в пакете) и последующее размещение данных пакетов в архивах атрибутов параметров DAQ.

- *Transporter* — модуль отражения объектов контроллеров удалённых OpenSCADA-станций на локальную. На данном этапе модуль не реализует пакетного режима, однако планируется реализация синхронного и асинхронного пакетных режимов отражения архивов удалённых OpenSCADA-станций.

1.3. Пассивный механизм сбора

Пассивный механизм сбора данных характерен инициативой предоставления данных в SCADA-систему со стороны источника данных. Этот механизм является достаточно редким явлением, однако может иметь место в случае определённых условий или ограничений в возможности использования прямых механизмов сбора данных, рис.5. Примером такой ситуации могут служить географически рассосредоточенные системы сбора данных посредством мобильных сетей GPRS/EDGE. В таких сетях наделение клиентских узлов отдельными/реальными IP-адресами или формирование корпоративной мобильной сети может оказаться дорогим удовольствием, поэтому доступнее оказывается инициатива сеанса передачи данных с клиентских динамических IP-адресов на один реальный IP-адрес сервера SCADA-системы. Хотя возможна работа и через сетевую СУБД-посредника.

Воздействия на модификацию передаются источнику данных в момент сеанса передачи данных источником.

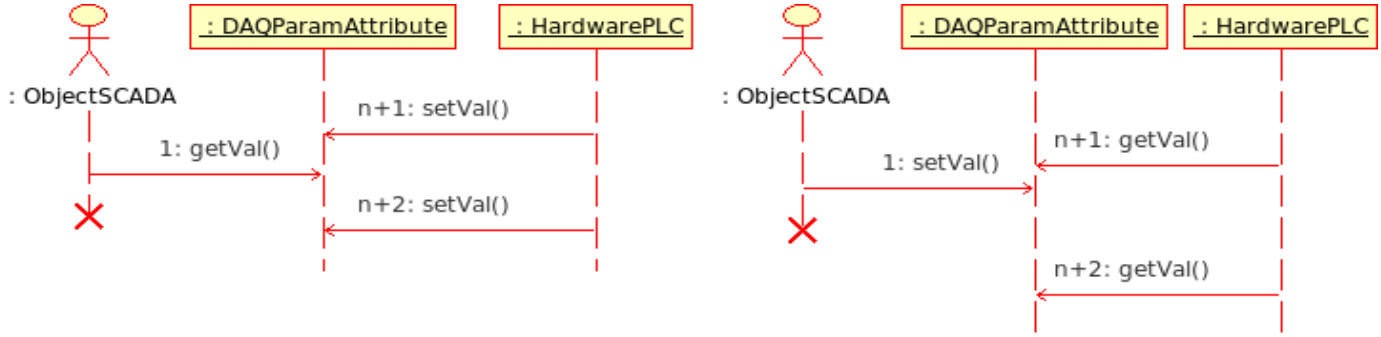

Рис. 5. Диаграмма последовательности взаимодействия при пассивном механизме работы.

В соответствии с диаграммой выше мы получаем следующее поведение пассивного механизма:

- Объект источника данных осуществляет периодические сеансы связи с объектом атрибута параметра DAQParamAttributeArch::setVal() для передачи своих данных и получения команд воздействия.
- Объект SCADA-системы шлёт запрос последнего значения к объекту атрибута параметра DAQParamAttribute::getVal() и получает сохранённое локально значение предыдущего сеанса связи источника данных.

В OpenSCADA такой механизм ещё не использован, однако принципиальная возможность его реализации в системе есть.

2. Логический уровень обработки данных

Выше мы говорили о том, что тип источника данных может колебаться от «сырого» до комплексного. Под «сырым» подразумевается источник, который предоставляет только элементарные сигналы (целое, вещественное, логическое, строка ...), причём отдельно. Под комплексным, подразумевается источник, который группирует сигналы и в параметре подсистемы 'Сбор данных', предоставляет атрибуты дополнительного назначения, покрывающие практически все диагностические задачи, т.е. параметр является законченным объектом, не требующим дополнения.

Учитывая такой разброс может возникнуть ситуация когда информации в объекте параметра контроллера источника данных недостаточно для описания реального объекта ТП в целом и нужен производный объект более высокого уровня абстракции. Решением такой ситуации может быть формирование дополняющих параметров, что является ненаглядным и вносит путаницу. Более правильным решением является использование прослойки, так называемого «Логического уровня», выполняющего функции гибкого формирования параметров, контейнеров сигналов, необходимой структуры и включающего пост-обработку.

Функционально «Логический уровень» предназначен для предоставления в системе OpenSCADA механизма свободного формирования объектов параметров, контейнеров сигналов, нужной структуры.

Эксплуатационным назначением «Логического уровня» является:

- расширение сферы применения системы OpenSCADA, за счёт увеличения гибкости описания объектов параметров подсистемы 'Сбор данных';
- сокращения затрат труда на создание сложных автоматизированных систем.

Концепция «Логического уровня» основана на шаблонах параметров, для которых в подсистеме 'Сбор данных' предусмотрен контейнер библиотек шаблонов (рис.1). Каждая библиотека содержит шаблоны параметров, которые могут использоваться модулями подсистемы 'Сбор данных' для реализации параметров на основе шаблонов. Модулями системы OpenSCADA, которые используют шаблоны в своей работе, являются:

- LogicLev — модуль реализации классической концепции 'Логического уровня'.
- Siemens — модуль сбора данных контроллеров фирмы Siemens серии S7. В виду высокой гибкости и функциональности контролеров фирмы этой серии, которая позволяет формировать комплексные типы данных различной структуры, все параметры этого модуля работают по шаблонам.

Общий механизм работы 'Логического уровня' на примере модуля LogicLev изображён на рис.6.

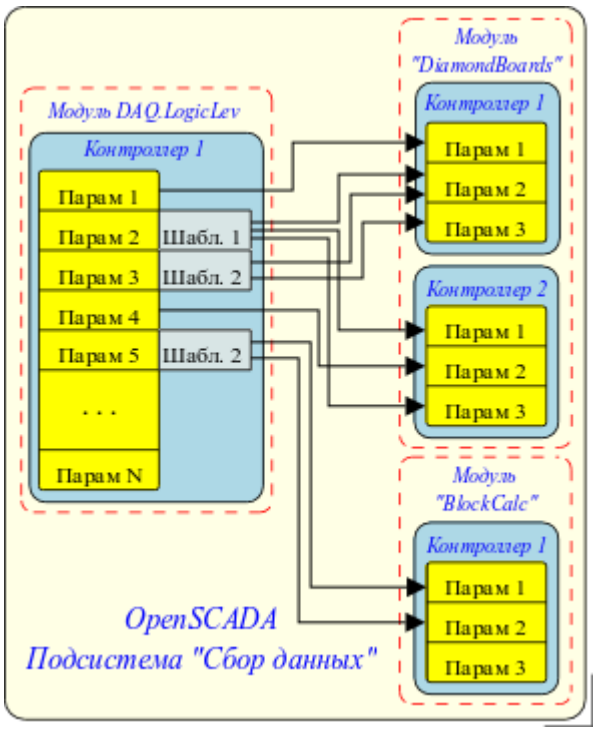

Рис. 6. Механизм работы 'Логического уровня' на примере модуля LogicLev.

Исходя из изображения видно, что параметры контроллера логического уровня выполняют функцию отражения других параметров подсистемы 'Сбор данных', на примере параметров 1 и 4, и произвольное формирование параметров на основе шаблонов 1,2 и других параметров подсистемы 'Сбор данных', на примере параметров 2, 3 и 5.

Структура параметров, с шаблоном в основе, имеет структуру изображённую на рис.7.

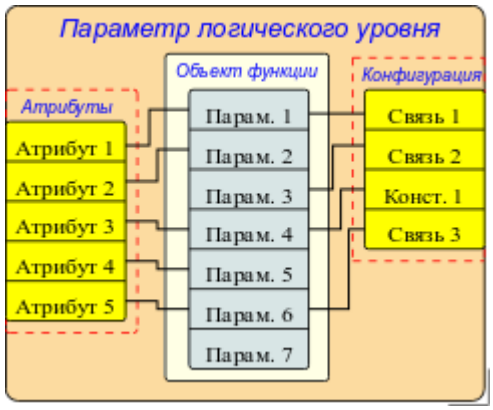

Рис. 7. Структура параметров, с шаблоном в основе.

Как можно видеть из структуры, параметр логического уровня состоит из объекта функции, атрибутов и конфигурации шаблона. Объект функции это экземпляр исполнения функции шаблона с набором входов/выходов и программой вычисления шаблона на одном из языков пользовательского программирования, обычно это Java-подобный язык пользовательского программирования модуля DAQ.JavaLikeCalc. Впрочем шаблон может быть вообще без программы, предоставляя только структуру проброса входов/выходов. Атрибуты в структуре изображают перечень атрибутов результирующего параметра, в соответствии с шаблоном. Конфигурация в структуре предоставляет конфигурацию свойств шаблона и его внешних связей.

Логику работы логического уровня параметров можно записать следующим образом:

- Параметр связывается с шаблоном из которого получается структура атрибутов, в соответствии с функцией шаблона.
- В момент связывания параметра с функцией выполняется связывание объекта экземпляра функции параметра с функцией из шаблона.

- Далее, в соответствии с шаблоном функции, формируется структура связей. Исходя из структуры связей формируется форма связывания параметра и пользователем устанавливаются связи.
- При доступе к атрибутам полученного параметра, производится проверка на наличие прямой связи. В случае наличия прямой связи запрос перенаправляется по этой связи, в противном случае значение берётся из объекта экземпляра функции параметра.
- В этот момент работает вычисление функции шаблона, по объекту функции параметров. При этом, перед вычислением производится чтение значений по связям, а после вычисления запись результатов по этим связям.

Шаблон параметров, в целом, предоставляют следующее:

- структуру входов/выходов функции шаблона;
- признаки конфигурации и связывания шаблона (константа, связь);
- предварительные значения конфигурации постоянных и шаблонов конфигурации связей;
- признаки атрибутов результирующего параметра логического уровня типов: не атрибут, атрибут с полным доступом, атрибут с доступом только на чтение;
- механизм вычисления входов/выходов функции шаблонов с использованием языка пользовательского программирования OpenSCADA.

На рис. 8 представлено изображение вкладки конфигурации шаблона параметров подсистемы 'Сбор данных' в виде таблицы с конфигурацией входов/выходов и текста программы пользовательского программирования.

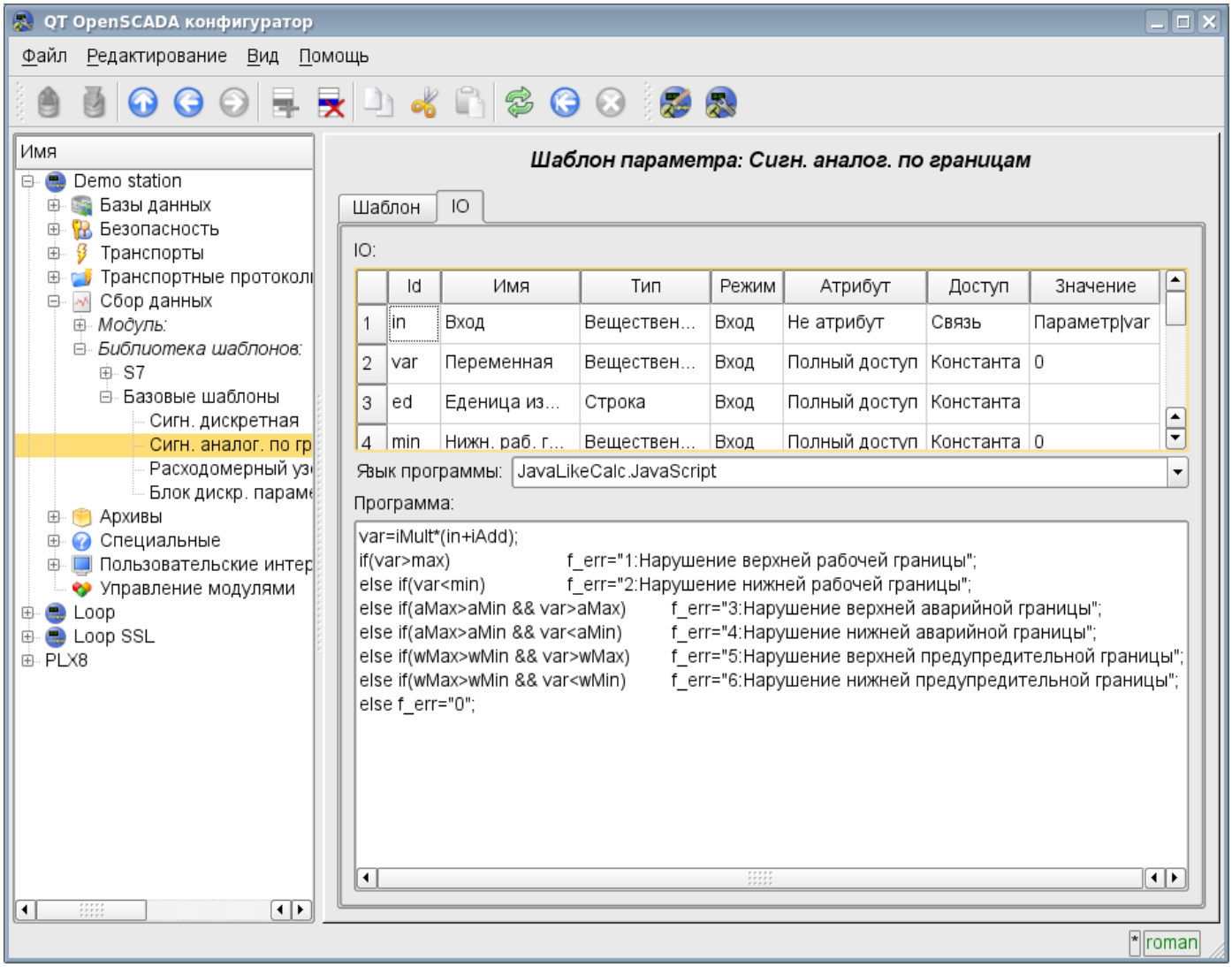

Рис. 8. Вкладка конфигурации шаблона параметров подсистемы 'Сбор данных'.

Полем входа/выхода шаблона параметра предусмотрены следующие свойства специализированного назначения: «Атрибут», «Доступ» и «Значение».

Свойство «Атрибут» выступает признаком отражения входа/выхода шаблона на результирующий атрибут параметра. Предусмотрены следующие варианты этого свойства:

- *не атрибут* — вход/выход функции шаблона не отражается на атрибут;
- *только чтение* — вход/выход функции шаблона отражается на атрибут с доступом только на чтение;
- *полный доступ* — вход/выход функции шаблона отражается на атрибут с полным доступом.

Свойство «Доступ» выступает признаком, указывающим на использование входа/выхода функции шаблона в результирующей конфигурации шаблона на логическом уровне. Предусмотрены следующие варианты этого свойства:

- *константа* — доступен для установки только на уровне конфигурации шаблона параметра, в виде постоянной;
- *публичная константа* — доступен для установки на уровне параметра логического уровня в разделе конфигурации шаблона, в виде постоянной;
- *Связь* — доступен для установки на уровне параметра логического уровня в разделе конфигурации шаблона, в виде связи.

Поле «Значение» описывает предустановленное значение для постоянных и шаблон конфигурации внешних связей. Шаблон конфигурации внешних связей используется в целях описания механизма группировки и автоматического распределения внешних связей. Структура шаблона конфигурации внешних связей специфична для каждого модуля подсистемы «Сбор данных» который использует механизм шаблонов. В случае с модулем логического уровня распределение производится над внешними атрибутами параметров, с шаблоном конфигурации внешней связи вида: <Параметр>|<атрибут>. Где «Параметр» используется для объединения параметров и помещения на форму конфигурации, а атрибут для ассоциативного связывания атрибутов при назначении параметра.

В качестве примера использования шаблона на рис.8 приведём изображения параметра модуля логического уровня 'F3'. На рис.9 представлена вкладка 'Конфигурация шаблона' для конфигурации, включая связывание, шаблона параметра. На рис.10 представлена вкладка 'Атрибуты' с перечнем атрибутов и их значений созданных посредством шаблона.

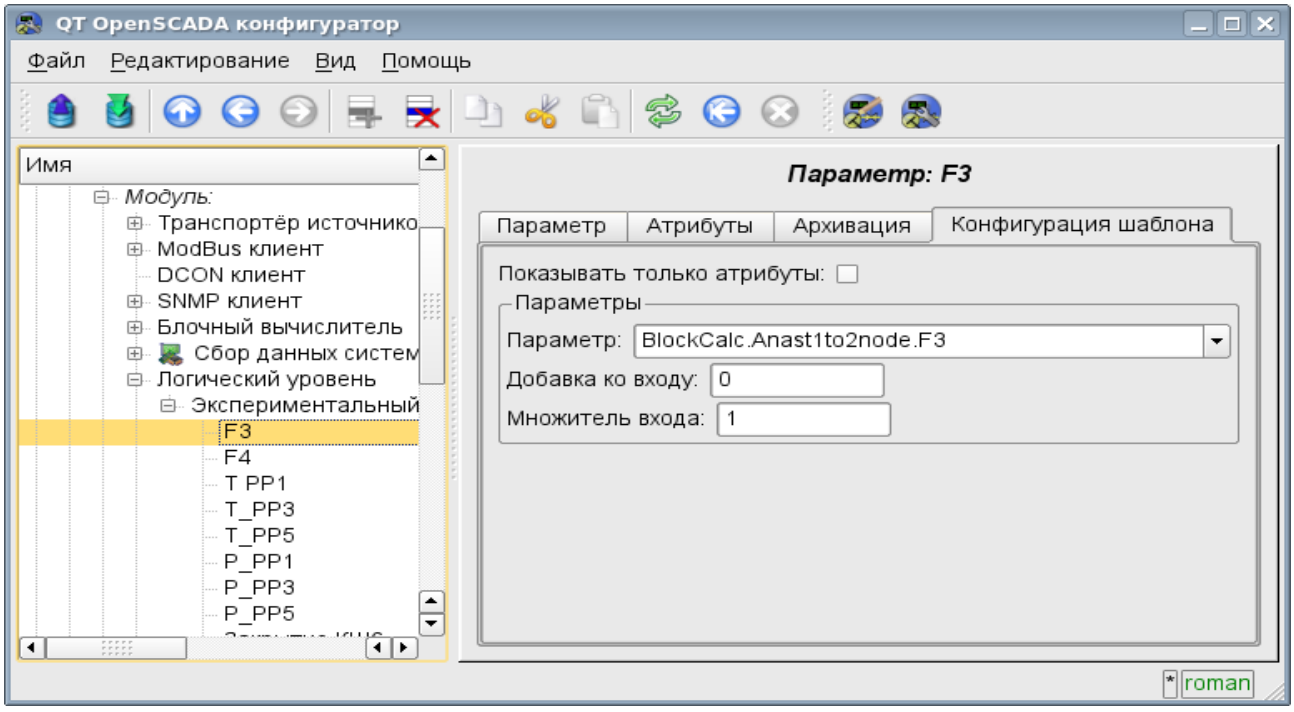

Рис. 9. Вкладка 'Конфигурация шаблона' параметра 'F3' модуля логического уровня.

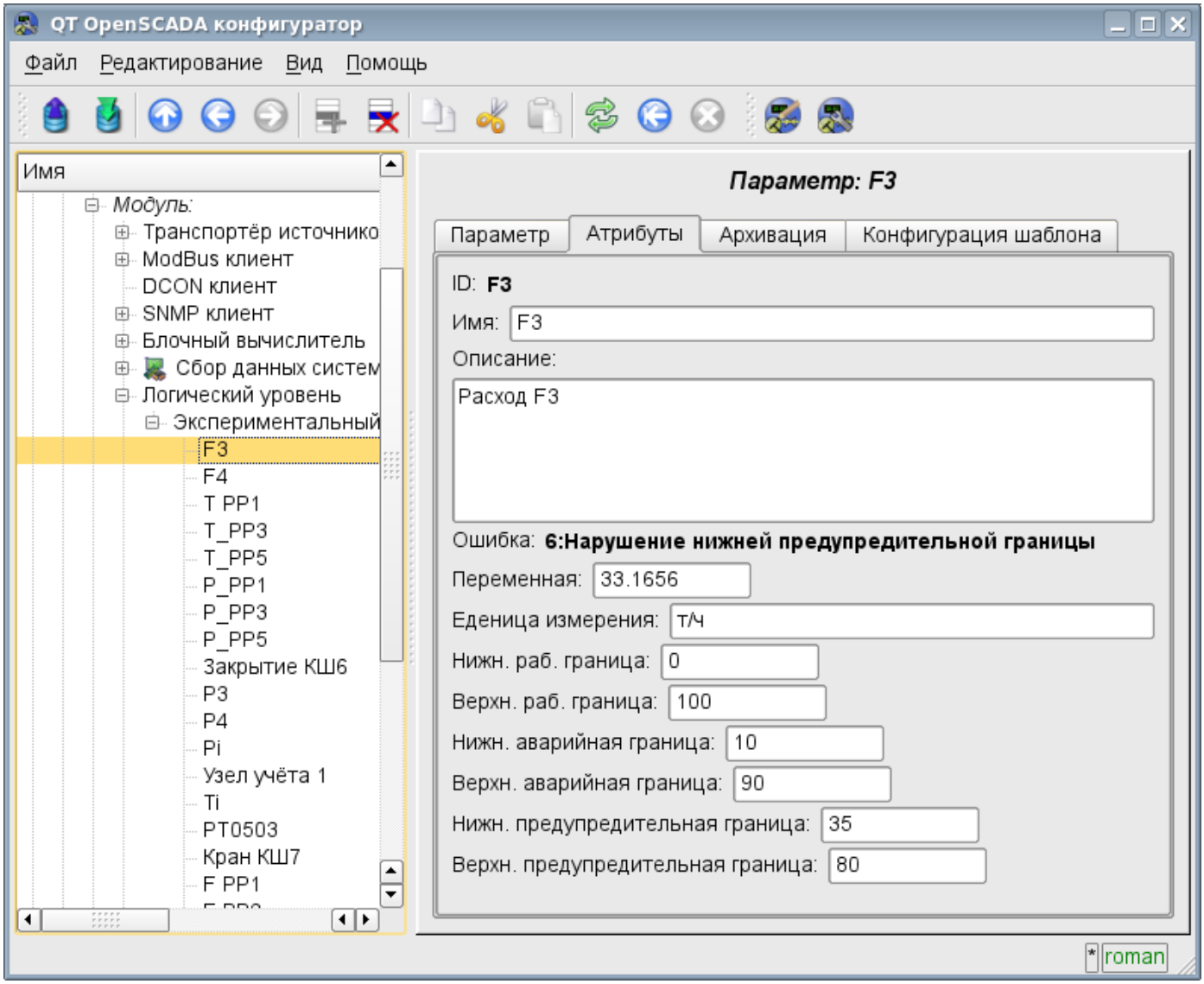

Рис. 10. Вкладка 'Атрибуты' параметра 'F3' модуля логического уровня.