Описание OpenSCADA

Роман Савоченко (rom_as@oscada.org)

7 июн, 2012

Оглавление

Введение.

Данный документ является описанием "open source" проекта системы именуемой "OpenSCADA". OpenSCADA представляет собой открытую SCADA систему, построенную по принципам модульности, многоплатформенности и масштабируемости.

В качестве политики разработки данной системы выбраны "open source" принципы. Выбор данной политики определяется необходимостью создания открытой, надёжной и общедоступной SCADA системы. Данная политика позволяет привлечь к разработке, тестированию, развитию, распространению и использованию продукта значительное количество разработчиков, энтузиастов и других заинтересованных лиц с минимизацией и распределением финансовых затрат.

Система OpenSCADA предназначена для сбора, архивирования, визуализации информации, выдачи управляющих воздействий, а также других родственных операций, характерных для полнофункциональной SCADA системы. Благодаря высокому уровню абстракции и модульности система может использоваться во многих смежных областях.

Система OpenSCADA может применяться:

- на промышленных объектах в качестве полнофункциональной SCADA системы;
- во встраиваемых (embeded) системах в качестве среды исполнения в том числе внутри PLC (программируемых логических контроллерах);
- для построения различных моделей (технологических, химических, физических, электрических процессов);
- на персональных компьютерах, серверах и кластерах для сбора, обработки, представления и архивации информации о системе и её окружении.

В качестве базовой (хостовой) операционной системы для разработки и использования выбрана ОС Linux, которая является оптимальным решением в вопросах:

- надёжности (большая доля серверов и кластеров работает на GNU/Linux);
- гибкости/масштабируемости (ввиду своей открытости и модульности позволяет строить решения под любые требования);
- доступности (благодаря лицензии GPL является полностью свободной системой, а при высокой квалификации пользователя и бесплатной);
- популярности, развитости, поддержке, распространённости (система активно развивается множеством энтузиастов, фирм и государственными учреждениями во всем мире, получает всё большую поддержку на пользовательском и корпоративном рынке, активно внедряется в государственные структуры различных стран).

Поскольку проект разрабатывается и реализуется по принципам многоплатформенности, то не составляет проблемы портировать его на другие ОС, что в дальнейшем и планируется.

Сердцем системы является модульное ядро. И в зависимости от того, какие модули подключены, система может выступать как в роли различных серверов, так и в роли разнообразных клиентов, а также совмещать эти функции в одной программе. Это позволяет реализовывать клиент-серверную архитектуру SCADA системы на базе одних и тех же компонентов/модулей, экономя при этом машинную память, дисковое пространство, а также ценное время программистов.

Серверные конфигурации системы предназначены для сбора, обработки, выдачи воздействий, архивирования, протоколирования информации от различных источников, а также предоставления этой информации клиентам (UI, GUI, TUI ...). Модульная архитектура позволяет расширять функциональность сервера без его перегрузки.

Клиентские конфигурации могут строиться на основе различных графических библиотек (GUI/TUI ToolKits), как используя ядро программы и его модули (путём добавления к нему UI-user interface модуля), так и в качестве самостоятельного приложения, подключая ядро OpenSCADA как библиотеку.

Возможность гибкой конфигурации системы позволяет строить решения под конкретные требования надёжности, функциональности и размеры системы.

1. Функции системы.

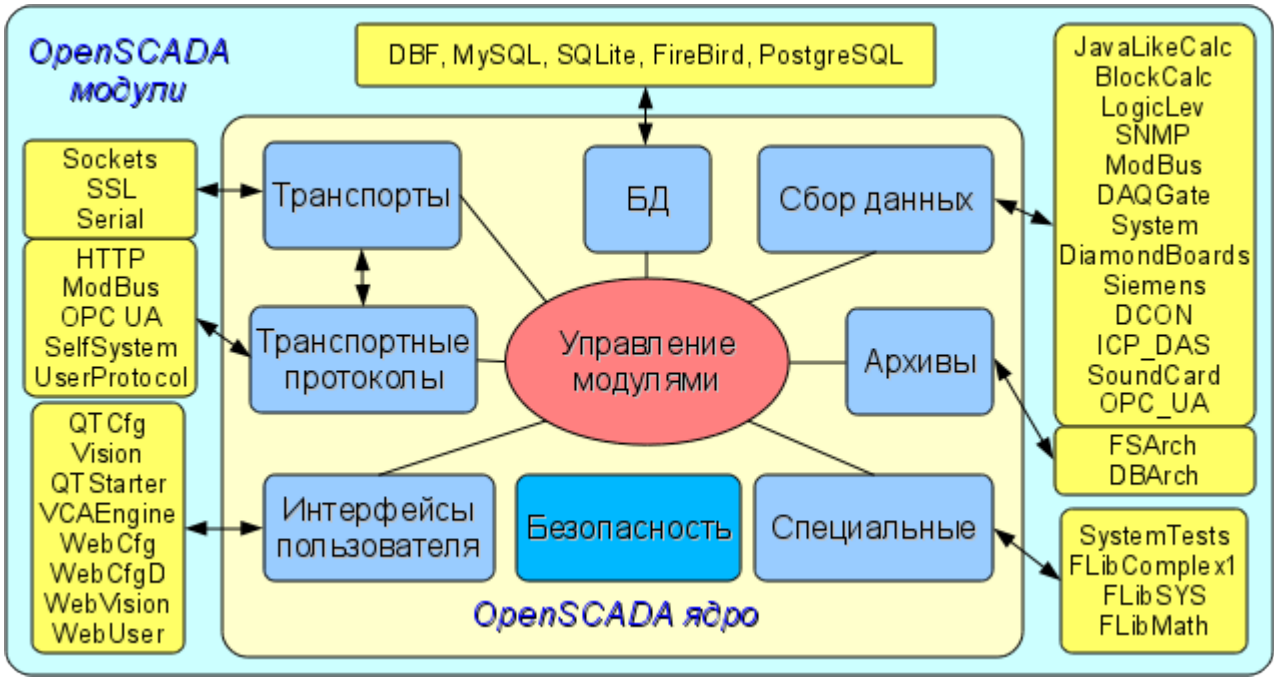

Рис. 1. Блочная схема системы OpenSCADA

1.1. Модульность.

Для придания гибкости и высокой степени масштабируемости система OpenSCADA построена по модульному принципу. Тесная интеграция модулей с ядром улучшает стабильность системы в целом, благодаря повторному использованию отлаженного кода. Однако сам процесс разработки собственного кода модулей OpenSCADA накладывает большую ответственность, возможные ошибки вводят элемент нестабильности в систему. Возможность создания распределённых конфигураций сглаживает эту опасность.

Модули системы OpenSCADA хранятся в динамических библиотеках. Каждая динамическая библиотека может содержать множество модулей различного типа. Наполнение динамических библиотек модулями определяется функциональной связностью самих модулей. Динамические библиотеки допускают горячую замену, что позволяет в процессе функционирования производить обновление отдельных частей системы. Метод хранения кода модулей в динамических библиотеках является основным для системы OpenSCADA, поскольку поддерживается практически всеми современными операционными системами(ОС). Однако это не исключает возможности разработки других методов хранения кода модулей.

На основе модулей реализованы следующие функциональные части системы OpenSCADA:

- базы данных;
- коммуникационные интерфейсы, транспорты;
- протоколы коммуникационных интерфейсов;
- источники данных и сбор данных;
- архивы (сообщений и значений);
- интерфейсы пользователя (GUI, TUI, WebGUI, speach, signal ... .);
- дополнительные модули, специальные.

Управление модулями осуществляется подсистемой «Управление модулями». Функциями подсистемы является: подключение, отключение, обновление модулей, а также другие операции, связанные с модулями и библиотеками модулей.

1.2. Подсистемы.

Архитектурно система OpenSCADA делится на подсистемы. Подсистемы могут быть двух типов: обычные и модульные. Модульные подсистемы обладают свойством расширения посредством модулей. Каждая модульная подсистема может содержать множество модульных объектов. Например, модульная подсистема «Базы данных» содержит модульные объекты типов баз данных. Модульный объект является корнем внутри модуля.

Всего система OpenSCADA содержит 9 подсистем из них 7 подсистем являются модульными. 9 подсистем системы OpenSCADA являются базовыми и присутствуют в любой конфигурации. К списку 9 подсистем могут добавляться новые подсистемы посредством модулей. Подсистемы системы OpenSCADA:

- Безопасность.
- Управление модулями.
- Базы данных (модульная).
- Транспорты (модульная).
- Транспортные протоколы (модульная).
- Сбор данных (модульная).
- Архивы (модульная).
- Интерфейсы пользователя (модульная).
- Специальные (модульная).

1.3. PLC и другие источники динамических данных. Подсистема "Сбор данных".

Для обеспечения поддержки источников динамических данных, будь то PLC-контроллеры, платы УСО, виртуальные источники и т.д., предназначена подсистема «Сбор данных». В функции этой подсистемы входит предоставление полученных данных в структурированном виде и обеспечение управления этими данными, например, модификация данных.

Подсистема «Сбор данных» является модульной и, как следствие, содержит модульные объекты типов источников динамических данных. Например, на октябрь 2007г, система OpenSCADA поддерживает следующие типы источников данных:

- Платы сбора данных от "Diamond systems".
- Сбор данных операционной системы (ОС).
- Блочный вычислитель.
- Вычислитель на Java-подобном языке.
- Транспортёр данных подсистемы "Сбор данных" от одной OpenSCADA станции к другой.
- Доступ к логическим контроллерам посредством протокола "ModBUS".
- Сбор данных сетевых устройств посредством протокола SNMP.
- Источник данных логического уровня системы OpenSCADA.
- Доступ к высокоинтеллектуальным логическим контроллерам посредством протокола MPI и коммуникационного процессора CIF50PB, фирмы Hilscher GMBH.

Каждый тип источника выполнен в виде отдельного модуля, который может быть подключен/отключен. Каждый тип источника может содержать отдельные источники (контроллеры).

Отдельно взятый контроллер может содержать параметры определённых модулем типов. Например, параметры аналогового типа; основной информацией, которую они предоставляет, является значение целого или вещественного типа. Структурно параметр представляет собой список атрибутов, которые и содержат данные. Атрибуты могут быть четырёх базовых типов: символьная строка(текст), целое, вещественное и логический тип.

Структуры контроллеров, параметров и их типов содержатся в подсистеме "Сбор данных", а объекты модулей выполняют их заполнение в соответствии с собственной спецификой.

Источник динамических данных может быть удалённым, т.е. быть подключен на удалённой системе OpenSCADA. Для связи с такими источниками данных используется транспортный тип контроллеров (Transporter). Функцией данного типа источника данных является отражение источников данных удалённой OpenSCADA станции на локальную станцию.

1.4. Базы данных. Подсистема "Базы данных"

Для хранения данных системы повсеместно используются базы данных (БД). В целях систематизации доступа и управления базами данных в системе OpenSCADA предусмотрена подсистема "Базы данных". Для обеспечения поддержки различных БД/СУБД подсистема выполнена модульной.

В роли модульных объектов, содержащихся в подсистеме, выступает тип БД/СУБД, т.е. модуль подсистемы «Базы данных» практически содержит реализацию доступа к определённому типу БД. Например, модули: DBF, MySQL, SQLite.

Объект типа БД/СУБД в свою очередь содержит список объектов отдельных БД данного типа, а объект БД содержит список объектов таблиц, которые и содержат данные в табличной форме.

Практически все данные системы OpenSCADA хранятся в той или иной БД. Инструментарий системы позволяет легко переносить данные из одного типа БД в другой, и, как следствие, оптимально подбирать тип БД под конкретную область применения системы OpenSCADA. Перенос информации с одной БД в другую может быть выполнен двумя способами. Первый — это изменение адреса рабочей БД и сохранение всей системы на неё, второй — это прямое копирование информации между БД. Кроме копирования поддерживается и функция прямого редактирования содержимого таблиц БД.

Для организации централизованного доступа распределённой системы к единой БД предусматриваются два способа. Первый это использование сетевых СУБД, например, MySQL. Второй способ это использование транспортного типа БД на локальных системах для доступа к одной центральной БД (Планируется.). Функцией транспортной БД является пересылка запросов к БД на удалённую OpenSCADA систему.

Данные могут храниться также в конфигурационном файле системы. Реализован механизм полного отражения структуры БД на структуру конфигурационного файла. Т.е. стандартную конфигурацию можно размещать в конфигурационном файле. Суть такого механизма в том, что данные системы по умолчанию, например, при старте без БД можно описывать в конфигурационном файле. В дальнейшем эти данные могут переопределяться в БД. Кроме этого для случаев невозможности запуска какой либо БД вообще можно все данные хранить в конфигурационном файле.

Для доступа к базам данных используется механизм регистрации БД. Зарегистрированные в системе БД доступны всем подсистемам системы OpenSCADA и могут использоваться в их работе. Благодаря этому механизму можно обеспечить распределёность хранения данных. Например, различные библиотеки могут храниться и распространяться независимо, а подключение библиотеки будет заключаться в простой регистрации нужной БД.

В дальнейшем планируется реализация дублирования БД путём связывания зарегистрированных БД. Этот механизм позволит значительно повысить надёжность системы OpenSCADA в целом путём резервирования механизма хранения данных. (Планируется.)

1.5. Архивы. Подсистема "Архивы".

Любая SCADA система предоставляет возможность архивирования собранных данных, т.е. формирование истории изменения (динамики) процессов. Архивы условно можно разделить на два типа: архивы сообщений и архивы значений.

Особенностью архивов сообщений является то, что архивируются так называемые события. Характерным признаком события является время возникновения этого события. Архивы сообщений обычно используются для архивирования сообщений в системе, т.е. ведение логов и

протоколов. В зависимости от источника сообщения могут классифицироваться по различным критериям. Например, это могут быть протоколы аварийных ситуаций, протоколы действий операторов, протоколы сбоев связи и др.

Особенностью архивов значений является их периодичность, определяемая промежутком времени между двумя смежными значениями. Архивы значений применяются для архивирования истории непрерывных процессов. Поскольку процесс непрерывный, то и архивировать его можно только путём введения понятия квантования опроса значений, поскольку иначе мы получаем архивы бесконечных размеров, ввиду непрерывности самой природы процесса. Кроме этого практически мы можем получать значения с периодом ограниченным самими источниками данных. Например, довольно качественные источники данных в промышленности редко позволяют получать данные с частотой более 1 кГц. И это без учёта самих датчиков имеющих ещё менее качественные характеристики.

Для решения задач архивирования потоков данных в системе OpenSCADA предусмотрена подсистема "Архивы". Подсистема "Архивы" позволяет вести как архивы сообщений, так и архивы значений. Подсистема "Архивы" является модульной. Модульным объектом, содержащимся в подсистеме "Архивы", выступает тип архиватора. Тип архиватора определяет способ хранения данных, т.е. хранилище (файловая система, СУБД, сеть и т.д.). Каждый модуль подсистемы "Архивы" может реализовывать как архивирование сообщений, так и архивирование значений. Подсистема "Архивы" может содержать множество архивов, обслуживаемых различными модулями подсистемы.

Сообщение в системе OpenSCADA характеризуется датой, уровнем важности, категорией и текстом сообщения. Дата сообщения указывает на время создания сообщения. Уровень важности указывает на степень важности сообщения. Категория определяет адрес или условный идентификатор источника сообщения. Обычно, категория содержит полный путь к источнику сообщения в системе. Текст сообщения, собственно, и несёт смысловую нагрузку сообщения.

В процессе архивирования сообщения пропускаются через фильтр. Фильтр работает по уровню важности и категории сообщения. Уровень сообщения в фильтре указывает, что нужно пропускать сообщения с указанным или более высоким уровнем важности. Для фильтрования по категории применяются шаблоны или регулярные выражения, которые определяют какие сообщения пропускать. Каждый архиватор содержит собственные настройки фильтра. Следовательно можно легко создавать различные специализированные архиваторы для архива сообщений. Например, архиваторы сообщений можно специализировать на:

- логи, для хранения отладочной информации и другой рабочей информации сервера;
- различные протоколы (протокол действий клиентов, протокол нарушений и исключений, протокол событий ...).

В виду похожей природы сообщения и нарушения подсистема "Архивы" содержит буфер текущих нарушений, который содержит активные на данный момент нарушения с использованием категории сообщения в роли ключа-идентификатора нарушения. Доступ к списку-буферу текущих нарушений осуществляется путём указания отрицательного значения уровня сообщения. Так, формирование сообщения с отрицательным уровнем -2 вызывает помещение в буфер активных нарушений этого сообщения с уровнем 2, а так-же дублирование его непосредственно в архиве сообщения. При последующем формировании сообщения, в такой же категории, но положительным уровнем, скажем 1, будет осуществлено удаление указанного нарушения из буфера нарушений, а само сообщение так-же попадёт в архив сообщений. Такой механизм позволяет одновременно вести учёт активных нарушений и протоколировать их прохождение в архиве сообщений. При запросе к архиву сообщений, указание положительного уровня осуществляет запрос к архиву сообщений, а отрицательного к буферу-списку текущих нарушений.

Архив значений в системе OpenSCADA выступает как независимый компонент, который включает буфер обрабатываемый архиваторами. Основным параметром архива значения является источник данных. В роли источника данных могут выступать атрибуты параметров системы OpenSCADA, а также другие внешние источники данных (пассивный режим). Другими источниками данных могут быть сетевые архиваторы удалённых OpenSCADA систем, среда программирования системы OpenSCADA и др.

Ключевым компонентом архивирования значений непрерывных процессов является буфер значений. Буфер значений предназначен для промежуточного хранения массива значений, полученных с определённой периодичностью (квантом времени). Буфер значений используется как для непосредственного хранения больших массивов значений в архивах значений как перед непосредственным «сбросом» на физические носители, так и для манипуляций с кадрами значений, т.е. в функциях покадрового запроса значений и их помещения в буфера архивов.

Для организации выделенных архиваторов в распределённых системах можно использовать транспортный тип архиватора (Планируется.). Функцией транспортного типа архиватора является отражение удалённого центрального архиватора на локальной системе. Как следствие архиваторы транспортного типа выполняют передачу данных между локальной системой и архиватором удалённой системы, скрывая от подсистем локальной системы реальную природу архиватора.

1.6. Коммуникации. Подсистемы "Транспорты" и "Транспортные протоколы".

Поскольку система OpenSCADA закладывается как высоко-масштабируемая система, то поддержка коммуникаций должна быть достаточно гибкой. Для удовлетворения высокой степени гибкости коммуникации в системе OpenSCADA реализованы в подсистемах "Транспорты" и "Транспортные протоколы", которые являются модульными.

Подсистема «Транспорты» предназначена для обмена неструктурированными данными между системой OpenSCADA и внешними системами. В роли внешних систем могут выступать и удалённые OpenSCADA системы. Под неструктурированными данными понимается массив символов определённой длины. Модульным объектом, содержащимся в подсистеме «Транспорты», выступает тип транспорта. Тип транспорта определяет механизм передачи неструктурированных данных. Например, это могут быть:

- сокеты (TCP/UDP/UNIX);
- каналы;
- разделяемая память.

Подсистема "Транспорты" включает поддержку входящих и исходящих транспортов. Входящий транспорт предназначен для обслуживания внешних запросов и отправки ответов. Исходящий транспорт, наоборот, предназначен для отправки сообщений и ожидания ответа. Следовательно, входящий транспорт содержит конфигурацию данной станции как сервера, а исходящий транспорт содержит конфигурацию удалённого сервера. Модуль подсистемы "Транспорты" реализует поддержку как входящего, так и исходящего транспортов.

Подсистема "Транспортные протоколы" предназначена для структуризации данных, полученных от подсистемы "Транспорты". По сути, подсистема "Транспортные протоколы" является продолжением подсистемы "Транспорты" и выполняет функции проверки структуры и целостности полученных данных. Так, для указания протокола, в связке с которым должен работать транспорт, предусмотрено специальное конфигурационное поле. Модульным объектом, содержащимся в подсистеме "Протоколы", является сам протокол. Например, транспортными протоколами могут быть:

- HTTP (Hyper Text Transfer Protocol);
- SelfSystem (OpenSCADA системный протокол).

Полную цепочку связи можно записать следующим образом:

- сообщение передаётся в транспорт;
- транспорт передаёт сообщение связанному с ним протоколу путём создания нового объекта протокола;
- протокол проверяет целостность данных;
- если пришли все данные, то сообщить транспорту о прекращении ожидания данных и передать ему ответ иначе сообщить, что нужно ожидать ещё;
- транспорт, получив подтверждение, отсылает ответ и удаляет объект протокола;
- если подтверждения нет, то транспорт продолжает ожидание данных, и в случае их поступления передаёт их сохранённому объекту протокола.

Поддерживаются протоколы и для исходящих транспортов. Исходящий протокол берёт на себя функцию общения с транспортом и реализацию особенностей протокола. Внутренняя сторона доступа к протоколу реализуется потоковым образом с собственной структурой для каждого протокольного модуля. Такой механизм позволяет выполнять прозрачный доступ к внешней системе, посредством транспорта, просто указывая имя протокола, с помощью которого обслуживать передачу.

Благодаря стандартному API-доступа к транспортам системы OpenSCADA можно легко менять способ обмена данными, не затрагивая самих обменивающихся систем. Например, в случае локального обмена можно использовать более быстрый транспорт на основе разделяемой памяти, а в случае обмена через интернет и локальную сеть использовать TCP или UDP сокеты.

1.7. Интерфейсы пользователя. Подсистема "Интерфейсы пользователя".

SCADA-системы, как класс, предполагают наличие интерфейсов пользователя. В OpenSCADA для предоставления пользовательских интерфейсов предусмотрена подсистема "Пользовательские интерфейсы". Под пользовательским интерфейсом системы OpenSCADA понимается не только среда визуализации, с которой должен работать конечный пользователь, но и всё, что имеет отношение к пользователю, например:

- среды визуализации;
- конфигураторы;
- сигнализаторы.

Подсистема "Пользовательские интерфейсы" является модульной. Модульным объектом подсистемы выступает собственно конкретный интерфейс пользователя. Модульность подсистемы позволяет создавать различные интерфейсы пользователей на различных GUI/TUI библиотеках и использовать наиболее оптимальное из решений в конкретно взятом случае, например, для сред исполнения программируемых логических контроллеров можно использовать конфигураторы и визуализаторы на основе Web-технологий (WebCfg, WebUI), а в случае стационарных рабочих станций использовать те же конфигураторы и визуализаторы, но на основе библиотек типа QT, GTK.

1.8. Безопасность системы. Подсистема "Безопасность".

Система OpenSCADA является разветвлённой системой, которая состоит из десятка подсистем и может включать множество модулей. Следовательно, предоставление всем неограниченного доступа к этим ресурсам является по крайней мере небезопасным. Поэтому для разграничения доступа в системе OpenSCADA предусмотрена подсистема "Безопасности". Основными функциями подсистемы "Безопасности" является:

- хранение учётных записей пользователей и групп пользователей;
- аутентификация пользователей;
- проверка прав доступа пользователя к тому или иному ресурсу.

1.9. Управление библиотеками модулей и модулями. Подсистема "Управление модулями".

Система OpenSCADA построена по модульному принципу, что подразумевает наличие множества модулей, которыми необходимо управлять. Для выполнения функции управления модулями системы OpenSCADA предусмотрена подсистема "Управление модулями". Все модули на настоящий момент поставляются в систему посредством разделяемых библиотек(контейнеров). Каждый контейнер может содержать множество модулей различного типа.

Подсистема "Управление модулями" реализует контроль за состоянием контейнеров и позволяет выполнять горячее добавление, удаление и обновление контейнеров и содержащихся в них модулей.

1.10. Непредусмотренные возможности. Подсистема "Специальные".

Разумеется, предусмотреть всех возможных функций невозможно, поэтому в системе OpenSCADA предусмотрена подсистема "Специальные". Подсистема "Специальные" является модульной и предназначена для добавления в систему OpenSCADA непредусмотренных функций путём модульного расширения. Например, с помощью подсистемы «Специальные» могут быть реализованы:

- тесты системы OpenSCADA и её модулей;
- библиотеки функций пользовательского программирования.

1.11. Пользовательские функции. Объектная модель и среда программирования системы.

Любая современная SCADA система должна содержать механизмы, предоставляющие возможность программировать на пользовательском уровне, т.е. содержать среду программирования. Система OpenSCADA содержит такую среду. С помощью среды программирования системы OpenSCADA можно реализовывать:

- Алгоритмы управления технологическими процессами.
- Крупные динамические модели реального времени технологических, химических, физических и других процессов.
- Адаптивные механизмы управления по моделям.
- Пользовательские процедуры управления внутренними функциями системы, её подсистемами и модулями.
- Гибкое формирования структур параметров на уровне пользователя, с целью создания параметров нестандартной структуры и заполнения её по алгоритму пользователя.
- Вспомогательные вычисления.

Среда программирования системы OpenSCADA представляет собой комплекс средств, организующих вычислительное окружение пользователя. В состав комплекса средств входят:

- объектная модель системы OpenSCADA;
- модули библиотек функций;
- вычислительные контроллеры подсистемы «Сбор данных» и другие вычислители.

Модули библиотек функций предоставляют множество функций определённой направленности, расширяющих объектную модель системы. Библиотеки могут реализоваться как набором функций фиксированного типа, так и функциями, допускающими свободную модификацию и дополнение.

Библиотеки функций фиксированного типа могут предоставляться стандартными модулями системы, органично дополняя объектную модель. Функции таких библиотек будут представлять собой интерфейс доступа к средствам модуля на уровне пользователя. Например, «Среда визуального представления данных» может предоставлять функции для выдачи различных сообщений. Используя эти функции, пользователь может реализовывать интерактивные алгоритмы взаимодействия с системой.

Библиотеки функций свободного типа предоставляют среду написания пользовательских функций на одном из языков программирования. В рамках модуля библиотек функций могут предоставляться механизмы создания библиотек функций. Так, можно создавать библиотеки аппаратов технологических процессов, а в последствии использовать их путём связывания. Различные модули библиотек функций могут предоставлять реализации различных языков программирования.

На основе функций, предоставляемых объектной моделью, строятся вычислительные контроллеры. Вычислительные контроллеры выполняют связывание функций с параметрами системы и механизмом вычисления.

2. SCADA системы и их структура.

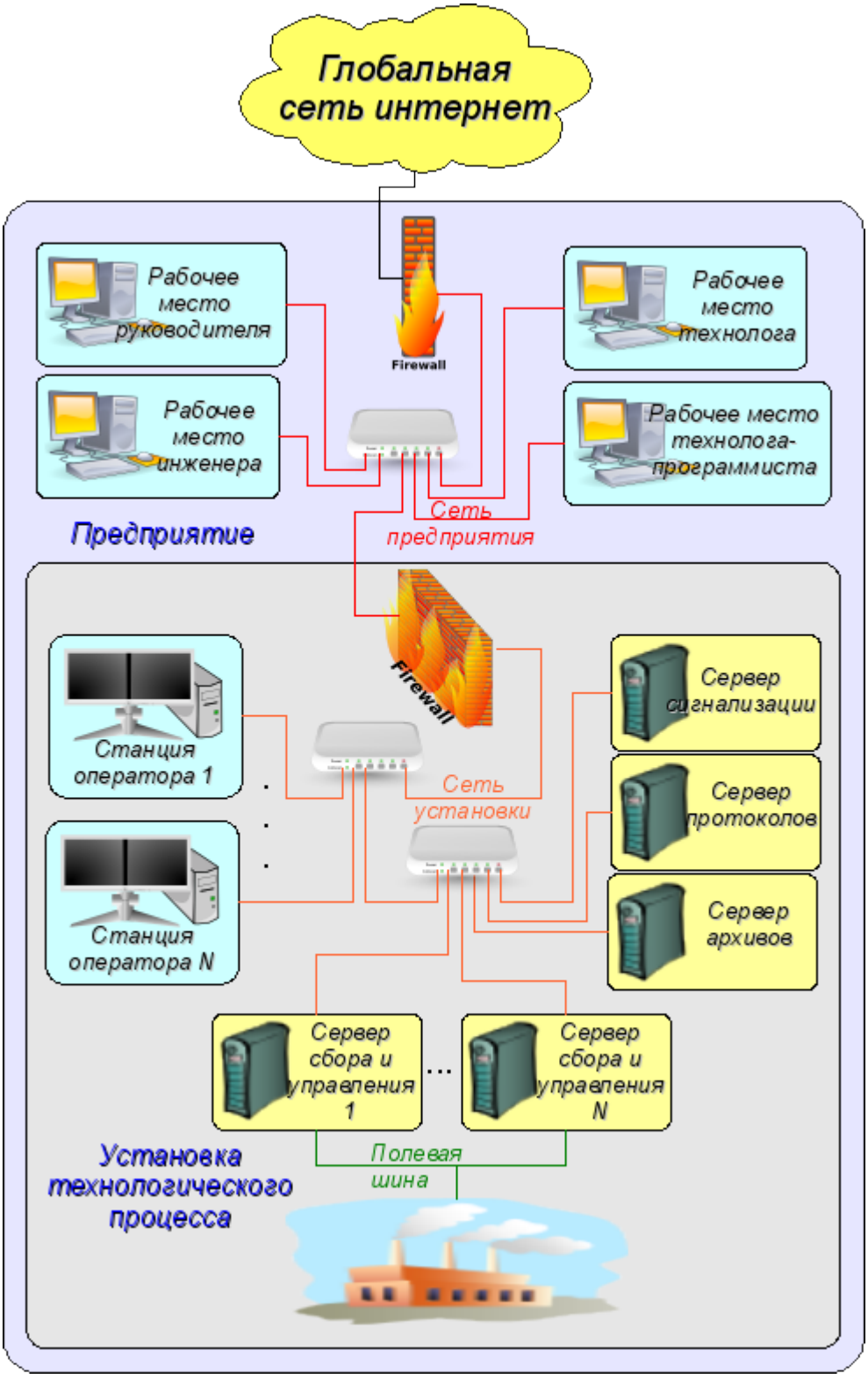

Рис. 2. SCADA-система.

SCADA (Supervisory Control And Data Acquisition), в общем виде, имеют распределённую архитектуру вроде изображённой на рис. 2. Элементы SCADA систем, в смысле программного обеспечения, выполняют следующие функции: **Сервер сбора:** представляет собой задачу или группу задач занимающихся сбором данных из источников данных, или же сами выступают в роли источником данных. В задачи сервера входит:

- получение и/или формирование данных;
- обработка данных;
- обслуживание запросов на доступ к данным;

- обслуживание запросов на модификацию данных.

Сервер архивирования: представляет собой задачу или группу задач занимающихся архивированием данных. В задачи сервера входит:
- архивирование данных SCADA-системы;
- обслуживание запросов на доступ к архивным данным;
- импорт/экспорт архивов.

Сервер протоколирования: представляет собой задачу или группу задач занимающиеся архивированием сообщений. В задачи сервера входит:
- архивирование сообщений узлов SCADA-системы;
- обслуживание запросов на доступ к архивным сообщениям;
- импорт/экспорт архивов.

Сервер сигнализации: представляет собой задачу или группу задач выполняющие функции сервера протоколирования в отношении узкой категории сообщений сигнализации.

Рабочее место оператора: представляет собой постоянно функционирующее GUI(Grafical User Interface) приложение выполненное в одномониторном, многомониторном или панельном режиме и выполняющее функции:
- предоставление пользовательского интерфейса для контроля за состоянием технологического процесса;
- предоставление возможности формирования управляющих воздействий;
- предоставление возможности изучения и анализа истории технологического процесса;
- предоставление инструментария для генерации отчётной документации.

Рабочее место инженера: представляет собой GUI приложение используемое для конфигурации SCADA системы. В задачи приложения входит:
- предоставление инструментария для манипуляции системными функциями системы;
- предоставление инструментария рабочего места оператора;
- предоставление инструментария для манипуляции архитектурой SCADA системы в целом (распределение функций между станциями, создание удаление станций ...).

Рабочее место руководителя: представляет собой GUI приложение, как правило, выполненное в одномониторном режиме и выполняющее функции:
- предоставление пользовательского интерфейса для контроля за состоянием технологического процесса;
- предоставление инструментария для изучения и анализа истории технологического процесса как непосредственно с активного сервера, так и на основе отдельных архивов;
- предоставление инструментария для генерации отчётной документации.

Рабочее место технолога: полностью включает в себя функции рабочего места оператора плюс модель технологического процесса (без непосредственной связи с технологическим процессом).

Рабочее место технолога-программиста: полностью включает в себя функции рабочего места технолога плюс инструментарий для создания моделей технологических процессов.

3. Варианты конфигурирования и использования.

3.1. Простое серверное подключение.

В простейшем случае систему OpenSCADA можно сконфигурировать в серверном режиме (рис. 3.1) для сбора и архивирования данных. Данная конфигурация позволяет выполнять следующие функции:

- опрос контроллеров;
- архивирование значений параметров;
- обслуживание клиентских запросов на получение различных данных сервера;
- предоставление конфигурационного WEB-интерфейса;
- удалённая конфигурация из системы OpenSCADA посредством QT-интерфейса или другого локального интерфейса.
- вторичное регулирование (регулирование в вычислительных контроллерах);
- моделирующие, корректирующие и дополняющие вычисления в вычислительных контроллерах.

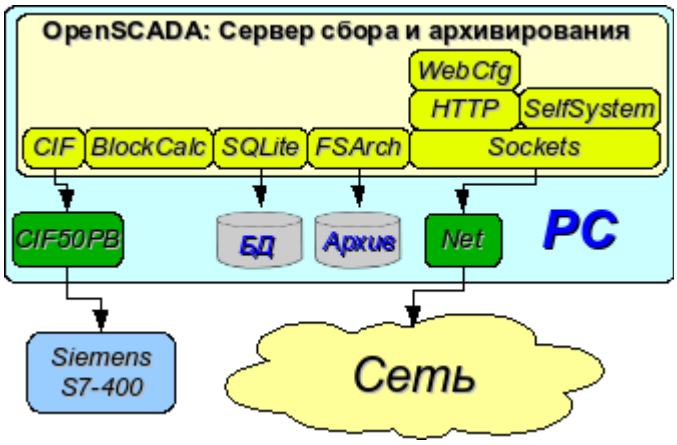

Рис. 3.1. Простое серверное подключение.

3.2. Дублированное серверное подключение.

Для повышения надёжности и производительности система OpenSCADA допускает множественное резервирование (рис. 3.2), при котором контроллеры одного экземпляра отражаются в другом. При использовании подобной конфигурации возможно распределение нагрузки опроса/вычисления на различных станциях. Данная конфигурация позволяет выполнять функции:

- опрос контроллеров;
- архивирование значений параметров;
- обслуживание клиентских запросов на получение различных данных сервера;
- резервирование параметров;
- резервирование архивов;
- распределение нагрузки опроса по серверам;
- предоставление конфигурационного WEB интерфейса;
- вторичное регулирование (регулирование в вычислительных контроллерах);
- моделирующие, корректирующие и дополняющие вычисления в вычислительных контроллерах с возможностью распределения нагрузки по серверам.

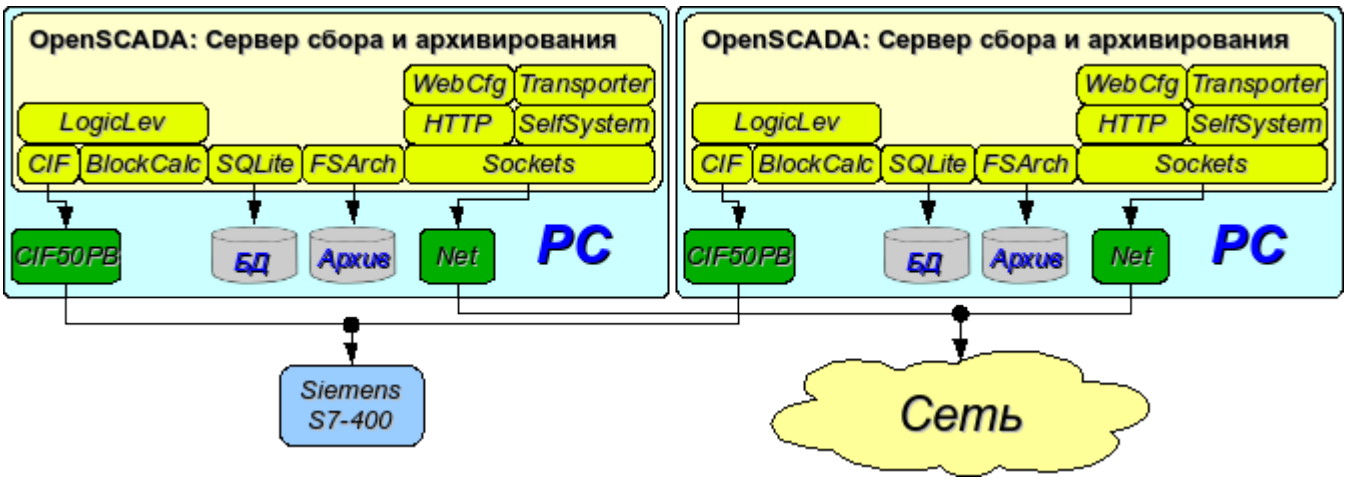

Рис. 3.2. Дублированное серверное подключение.

3.3. Дублированное серверное подключение на одном сервере.

Частным случаем дублированного соединения является дублированное соединение в рамках одного сервера (рис. 3.3), т. е запуск нескольких станций на одной машине с перекрещиванием параметров. Целью данной конфигурации является повышение надёжности и отказоустойчивости системы путём резервирования ПО.

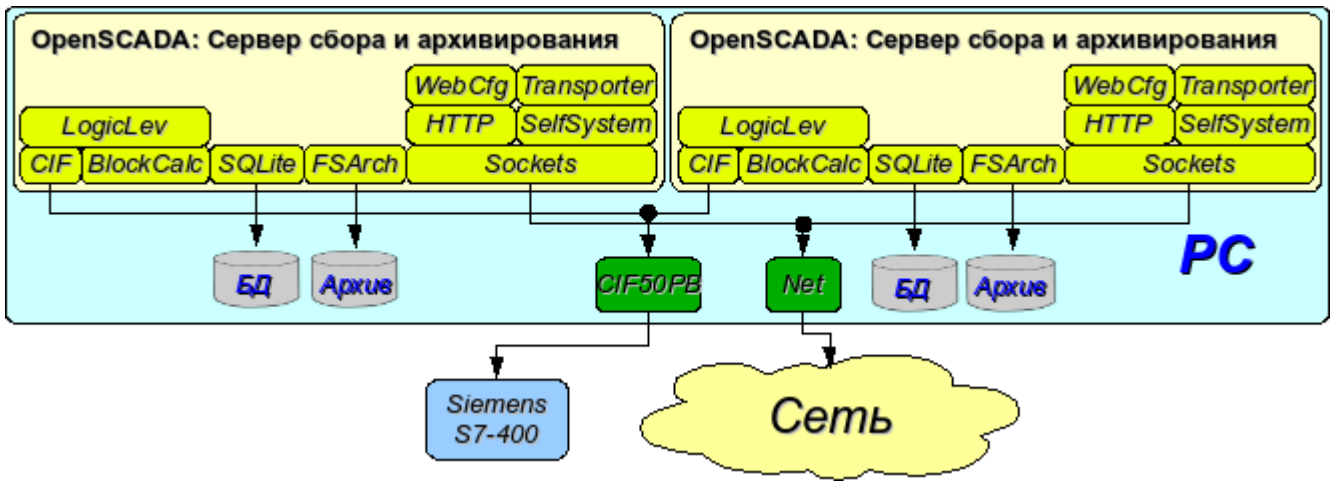

Рис. 3.3. Дублированное серверное подключение на одном сервере.

3.4. Клиентский доступ посредством Web-интерфейса. Место руководителя.

Для визуализации данных, содержащихся на сервере, хорошим решением является использование пользовательского WEB-интерфейса (рис. 3.4). Данное решение позволяет использовать стандартный WEB-браузер у клиента и следовательно является наиболее гибким, поскольку не привязано к одной платформе, т.е. является многоплатформенным. Однако это решение имеет существенные недостатки – это невысокая производительность и надёжность. В связи с этим рекомендуется использовать данный метод для визуализации некритичных данных или данных, имеющих резервный высоконадёжный способ визуализации. Например, хорошим решением будет использование этого метода у начальства промышленных установок, где всегда существует операторская с надёжным способом визуализации. Данная конфигурация позволяет выполнять следующие функции:

- опрос сервера на предмет получения данных визуализации и конфигурации;
- визуализация данных в доступном для понимания виде;
- формирование протоколов, отчётов;
- манипуляция параметрами, допускающими изменение.

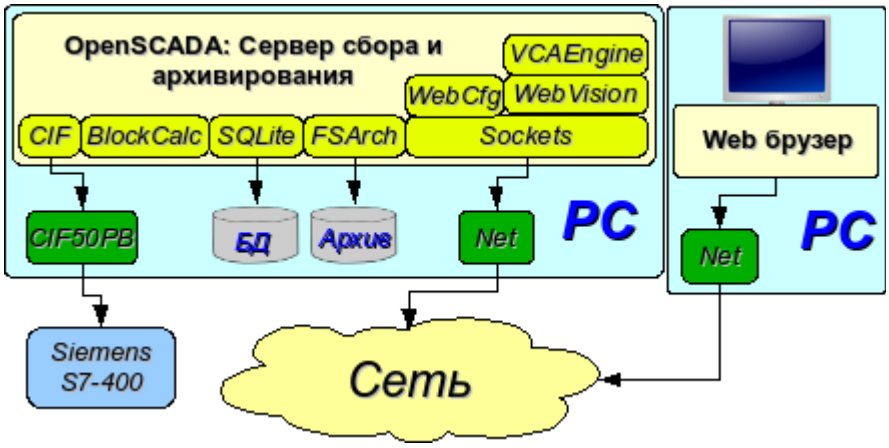

Рис. 3.4. Клиентский доступ посредством Web-интерфейса. Место руководителя.

3.5. Автоматизированное рабочее место (место руководителя/оператора).

Для визуализации критических данных, а также в случае если требуется высокое качество и производительность, можно использовать визуализацию на основе системы OpenSCADA сконфигурированной с GUI модулем (рис. 3.5). Данная конфигурация позволяет выполнять следующие функции:

- опрос сервера на предмет обновления текущих значений;
- визуализация опрошенных данных в доступном для понимания виде;
- формирование протоколов и отчётов;
- манипуляция параметрами, допускающими изменения.

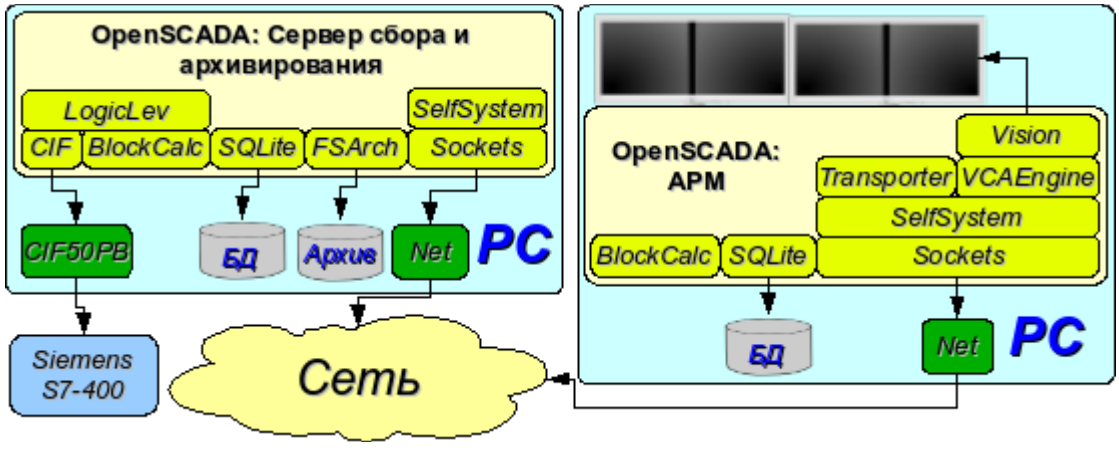

Рис. 3.5. Автоматизированное рабочее место (место руководителя/оператора)

3.6. АРМ с сервером сбора и архивирования на одной машине (место оператора, модель ...).

Полнофункциональная клиент-серверная конфигурация на одной машине (рис. 3.6) может использоваться для повышения надёжности системы в целом путём запуска клиента и сервера в разных процессах. Данная конфигурация позволяет без последствий для сервера останавливать клиент и выполнять с ним различные профилактические работы. Рекомендуется для использования на станциях оператора путём установки двух машин совмещающих в себе станции оператора и резервированный сервер. Данная конфигурация позволяет выполнять следующие функции:

- опрос контроллеров;
- обслуживание клиентских запросов;
- визуализация;
- выдача управляющих воздействий;
- генерация протоколов и отчётов;
- вторичное регулирование;
- моделирующие, корректирующие и дополнительные вычисления в вычислительных контроллерах;
- сбор и визуализация информации о персональном компьютере, сервере ... .

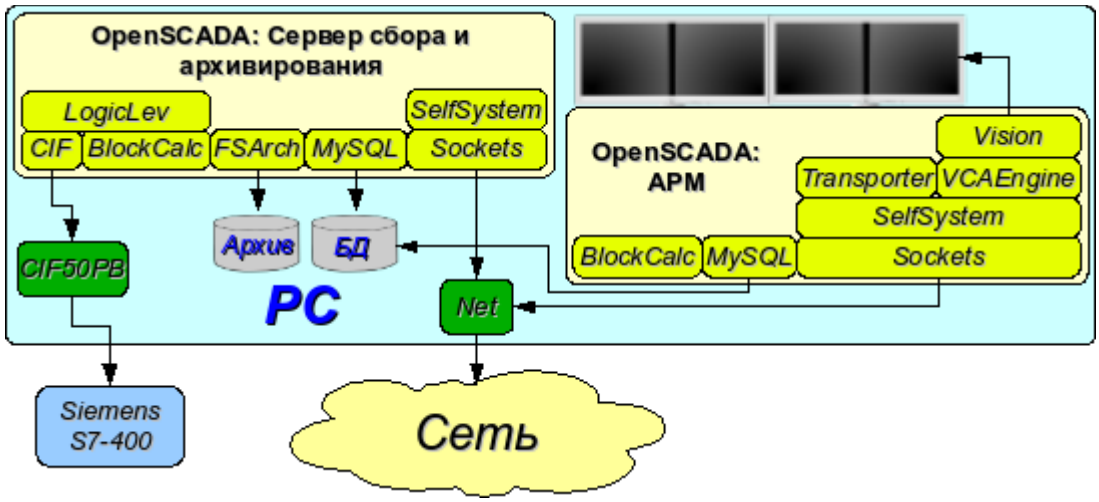

Рис. 3.6. АРМ с сервером сбора и архивирования на одной машине (место оператора, модель ...)

3.7. Простейшее смешанное подключение (модель, демонстрация, конфигуратор ...).

Смешанное подключение совмещает функции сервера и клиента (рис. 3.7). Может использоваться для тестовых, демонстрационных функций, а также для предоставления моделей технологических процессов как единое целое. В этом режиме могут выполняться следующие функции:

- опрос контроллеров;
- обслуживание клиентских запросов;
- визуализация;
- выдача управляющих воздействий;
- генерация протоколов и отчётов;
- вторичное регулирование;
- моделирующие, корректирующие и дополняющие вычисления в вычислительных контроллерах;
- сбор и визуализация текущей информации о персональном компьютере, сервере, модели ... ;
- конфигурация баз данных, подключений и др.

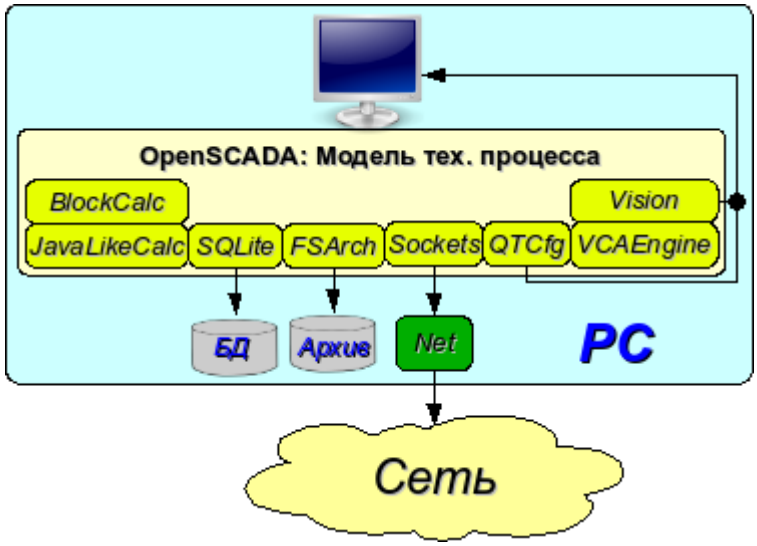

Рис. 3.7. Простейшее смешанное подключение (модель, демонстрация, конфигуратор ...)

3.8. Устойчивая, распределённая конфигурация.

Данная конфигурация является одним из вариантов устойчивого/надёжного соединения (рис. 3.8). Устойчивость достигается путём распределения функций по:

- серверам опроса;
- центральному серверу архивирования и обслуживания клиентских запросов;
- клиентам: АРМы и WEB-клиенты.

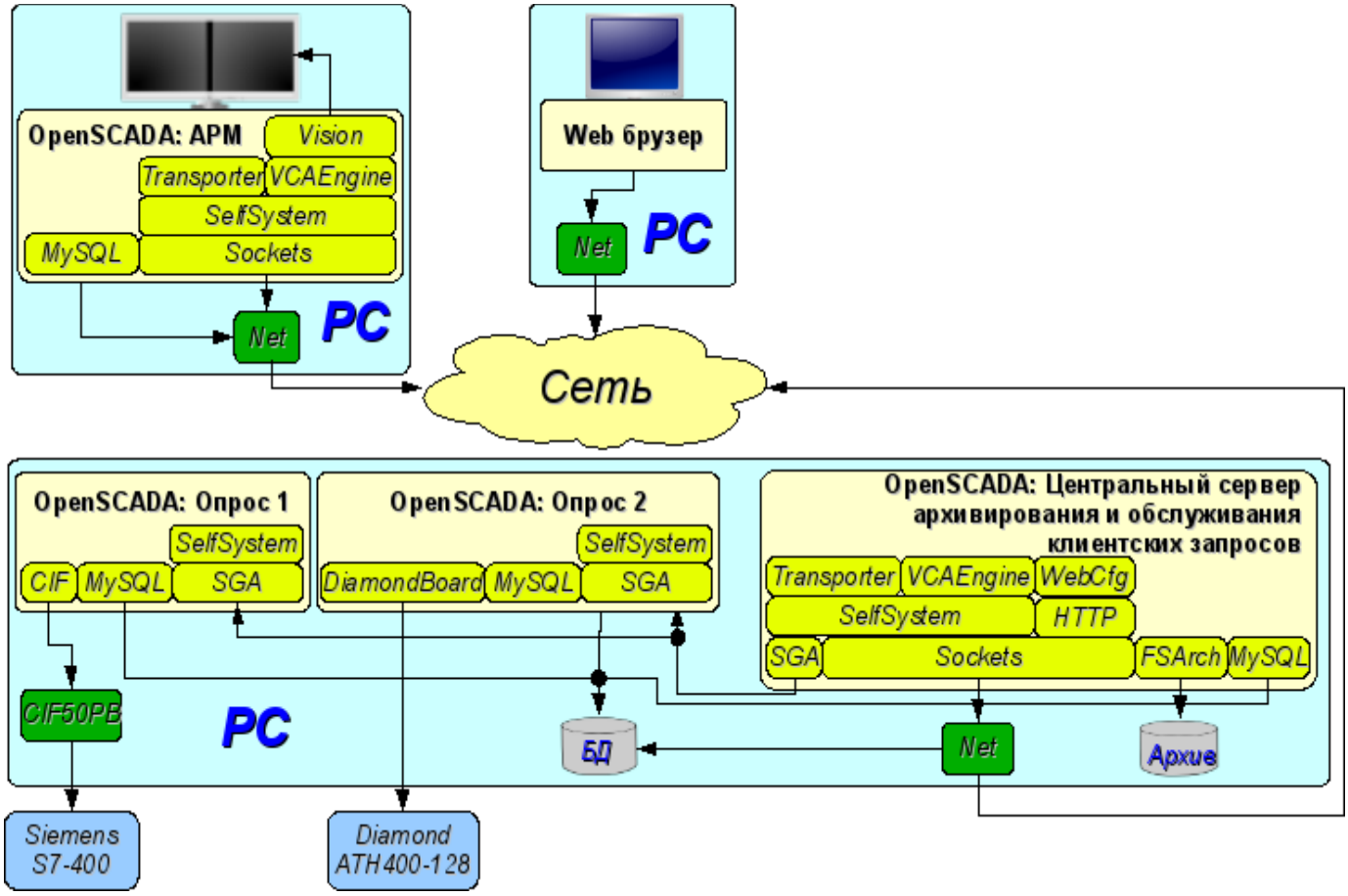

Рис. 3.8. Устойчивая, распределённая конфигурация

Сервер опроса конфигурируется на основе системы OpenSCADA и представляет собой задачу (группу задач), занимающихся опросом контроллера (группы контроллеров одного типа). Полученные значения доступны центральному серверу через любой транспорт, поддержка которого добавляется путём подключения соответствующего модуля транспорта. Для снижения частоты опроса и величины сетевого трафика, сервер опроса может оснащаться небольшим архивом значений. Конфигурация сервера опроса хранится в одной из доступных БД.

Центральный сервер архивирования и обслуживания клиентских запросов выполняет функцию централизованного сбора и обработки параметров серверов опроса и их значений. Доступ к серверам опроса выполняется посредством одного из доступных в OpenSCADA транспортов+протоколов (на примере это SGA). Для предоставления единого интерфейса доступа к параметрам и контроллерам используется модуль Transporter, который отражает данные серверов опроса на структуру локальных параметров.

Для выполнения внутренних вычислений и дополнительного анализа параметров используются вычислительные контроллеры.

Для разностороннего и глубокого архивирования используются различные модули архивов.

Для доступа клиентов к серверу используются доступные для OpenSCADA сетевые транспорты, на примере — это Sockets, и транспортные протоколы, на примере — это протокол OpenSCADA "SelfSystem".

Конфигурация центрального сервера хранится в одной из доступных БД (на примере это сетевая СУБД MySQL).

Для предоставления пользовательского WEB-интерфейса используется модуль WebCfg посредством транспортного протокола "HTTP".

Различные клиенты в их числе АРМ и WEB-клиенты выполняются на отдельных машинах в необходимом количестве. АРМ реализуется на основе системы OpenSCADA. В его функции входит опрос значений параметров из центрального сервера и их визуализация на GUI интерфейсе(ах). Для получения параметров в АРМ также используется модуль отражения удалённых параметров Transporter. Для предоставления доступа к архивам может использоваться модуль архива сетевого типа. Конфигурация АРМ может храниться в одной из доступных БД (в примере это сетевая СУБД MySQL, расположенная на машине центрального сервера архивирования).

4. Конфигурация и настройка системы.

Как можно видеть в разделе выше, OpenSCADA предоставляет возможность конфигурации для исполнения в различных ролях. Поддержка этой возможности обеспечивается развитыми механизмами конфигурации и хранения конфигурационных данных. Данный раздел содержит описание этих механизмов, призванное дать представление о гибкости и разнообразии, позволив тем самым использовать OpenSCADA на 100%.

При описании механизмов конфигурации и способов её хранения в этом разделе будет делаться упор на описание общесистемных механизмов. Особенности конфигурации и использования модулей подсистем OpenSCADA предоставляются в собственной документации модулей.

В OpenSCADA используется формализованный подход к описанию конфигурационных интерфейсов, основанный на языке XML. Фактически особенности конфигурации компонента системы предоставляется самим компонентом, пронизывая тем самым всю систему, как нервная система организма. В терминах OpenSCADA это называется интерфейсом контроля OpenSCADA (Control interface). На основе интерфейса контроля формируются графические интерфейсы конфигурации пользователя посредством модулей OpenSCADA. Такой подход имеет следующие важные преимущества:

- Масштабируемость. Можно подключать только нужные модули конфигурирования или вообще использовать только удалённые механизмы.
- Исключение необходимости обновления конфигураторов с добавлением нового модуля/функции, а также исключение "распухания" конфигуратора, обеспечивающего поддержку всей истории уже ненужных и устаревших модулей/функций.
- Простота создания графических интерфейсов конфигурации на различной основе за счёт чёткой формализованости.
- Предоставляется возможность динамической конфигурации, т.е. конфигурацию можно выполнять непосредственно при работе системы как локально, так и удалённо, непосредственно контролируя результат.
- Простая и целевая расширяемость конфигурационного интерфейса, путём добавления полей конфигурации на языке описания интерфейса управления только в компонентах, этого требующих.

В OpenSCADA уже предоставляется три модуля конфигурации на разной основе визуализации. Отметим их и их возможности конфигурации:

- Модуль конфигурации на библиотеке графического интерфейса QT(http://www.trolltech.com/qt) — UI.QTCfg. Предоставляет развитый интерфейс конфигурации, позволяющий управлять как локальной станцией, так и удалёнными станциями в локальной и глобальной сетях, включая безопасное соединение.
- Модуль конфигурации на основе динамических WEB-технологий (DHTML) — UI.WebCfgD. Предоставляет развитый интерфейс конфигурации, позволяющий управлять как локальной станцией сервера, так и удалёнными станциями в локальной и глобальной сетях, включая работу по безопасному соединению. Клиентское подключение осуществляется посредством обычного Web-браузера.
- Модуль конфигурации на основе статических WEB-технологий (XHTML) — UI.WebCfg. Предоставляет достаточный интерфейс конфигурации, позволяющий управлять локальной станцией сервера посредством обычного Web-браузера.

Значения конфигурации, изменённые в конфигураторах, а также большинство данных сохраняются в базах данных (БД). Учитывая модульность подсистемы "БД", ими могут быть различные БД. Причём предоставляется возможность хранения разных частей OpenSCADA как в разных БД одного типа, так и в БД разных типов.

Кроме БД данные о конфигурации могут содержаться в конфигурационном файле OpenSCADA, а также передаваться посредством параметров командной строки при вызове OpenSCADA. Сохранение конфигурации в конфигурационном файле осуществляется на равне с БД. Типовым именем конфигурационного файла OpenSCADA является /etc/oscada.xml. Формат конфигурационного файла и параметры командной строки рассмотрим в отдельном разделе.

Многие настройки и конфигурация объектов OpenSCADA, которые исполняются или уже включены, не применяются сразу-же по внесению изменений, поскольку конфигурация читается/применяется обычно только при включении или запуске. Следовательно для применения изменений, в таких случаях, достаточно включить/выключить включенный объект или перезапустить исполняющийся — остановить/запустить.

Дальнейшее рассмотрение конфигурации OpenSCADA будет производиться на основе интерфейса конфигуратора UI.QTCfg, однако принципы работы будут полностью соответствовать и остальным конфигураторам благодаря общности в используемом интерфейсе контроля OpenSCADA.

Рассмотрение начнём с конфигурации системных параметров OpenSCADA, которая размещается в трёх вкладках корневой страницы станции:

- Вкладка "Станция" содержит основные информационные и конфигурационные поля станции, рис.4а. Перечислим предоставляемые поля и прокомментируем их:
  - *ID* — содержит информацию об идентификаторе станции. Указывается параметром командной строки --Station. При загрузке ищется соответствующий идентификатору станции раздел в конфигурационном файле и, если не обнаруживается, то используется первый доступный.
  - *Имя станции* — указывает локализованное имя станции.
  - *Программа* — содержит информацию об имени программы. Обычно это OpenSCADA или имя основанного на OpenSCADA решения.
  - *Версия* — содержит информацию о текущей версии программы.
  - *Имя хоста* — содержит информацию о имени машины, на которой запущена станция.
  - *Системный пользователь* — содержит информацию о пользователе, от имени которого выполняется программа в системе (ОС).
  - *Операционная система* — содержит информацию о имени и версии ОС, ядре ОС, на которой исполняется программа.
  - *Частота (МГц)* — содержит информацию о частоте центрального процессора, которым исполняется программа. Значение частоты проверяется раз в 10 секунд и позволяет отслеживать её изменение, например, механизмами управления питанием.
  - *Разрешение часов реального времени (мс)* — содержит информацию о возможности или разрешении часов реального времени ОС. Позволяет сориентироваться с минимальным интервалом времени периодических задач, например, для задач сбора данных.
  - *Внутренняя кодировка* — содержит информацию о кодировке, в которой хранятся текстовые сообщения внутри программы.
  - *Конфигурационный файл* — содержит информацию о конфигурационном файле, используемом программой. Устанавливается параметром командной строки --Config.
  - *Рабочая директория* — указывает на рабочую директорию станции. Используется в относительной адресации объектов на файловой системе, например, файлов БД. Допускает изменение пользователем для сохранения данных системы в другую БД. При этом значение этого поля не сохраняется в БД, а может быть изменено только в секции "WorkDB" конфигурационного файла.
  - *Директория иконок* — указывает на директорию, содержащую иконки программы. Если в дереве навигации конфигуратора отсутствуют иконки, то Вы неправильно указали значение этого поля.
  - *Директория модулей* — указывает на директорию модулей для OpenSCADA. Если значение этого поля некорректно, то при запуске Вы не увидите никакого графического интерфейса, а только информацию в консоли о корректном запуске ядра OpenSCADA.
  - *Рабочая БД* — указывает на рабочую базу данных (БД), а именно на БД, используемую для хранения основных данных программы. Изменение этого поля отмечает все объекты программы как модифицированные, что позволяет сохранить или загрузить данные станции из указанной основной БД.

- *Сохранять систему при выходе* — указывает на необходимость сохранения изменённых данных при завершении работы.
- *Период сохранения системы* — указывает на периодичность в секундах, с которой сохранять изменённые данные станции.
- *Язык* — указывает на язык сообщений программы. Изменение этого поля допустимо, однако приводит к изменению языка сообщений только для интерфейса и динамических сообщений!
- *Базовый язык текстовых переменных* — используется для включения режима поддержки многоязыковых текстовых переменных путём указания непустого базового языка. Значение базового языка выбирается из списка двухсимвольных кодов языков, обычно только текущий и базовый языки в списке. Далее для текстовых переменных на небазовом языке в таблицах БД будут создаваться отдельные колонки. Под текстовыми переменными подразумеваются все текстовые поля конфигуратора, которые могут быть переведены на другой язык. Числа и другие символьные значения к их числу не относятся и не переводятся.
- *Сообщения:* — раздел группы параметров, управляющих работой с сообщениями станции:
  - *Наименьший уровень:* — указывает на уровень сообщений, начиная с которого необходимо их обрабатывать. Сообщения ниже этого уровня будут игнорироваться. Необходимо, например, для исключения из обработки отладочных сообщений уровня 0.
  - *В системный логер(syslog)* — указывает на необходимость направления сообщений в системный логер, механизм ОС для работы с сообщениями системы и ПО. При включении этого параметра появляется возможность управлять и контролировать сообщения OpenSCADA механизмами ОС.
  - *На стандартный выход(stdout)* — указывает на использование в качестве вывода сообщения стандартного механизмы вывода в консоль. Выключение этого свойства исключит весь вывод в консоль, если не указан следующий параметр.
  - *На стандартный выход ошибок(stderr)* — указывает на использование в качестве вывода сообщения стандартного механизмы вывода ошибок, обычно тоже направляется в консоль.
  - *В архив* — указывает на необходимость вывода сообщений в архив сообщений OpenSCADA. Этот параметр обычно включен, а его выключение приводит к фактическому отключению архивирования сообщений на станции.
- Вкладка "Подсистемы" содержит список подсистем (рис.4b) и позволяет выполнять прямые переходы к ним с помощью контекстного меню.
- Вкладка "Задачи" содержит таблицу со списком задач открытых различными компонентами OpenSCADA (рис.4c). Из таблицы можно получит различную информацию о задачах, а также назначить процессора для задач, на многопроцессорных системах.
- Вкладка "Помощь" содержит краткую помощь для данной страницы, рис.4d. В данном случае это доступные параметры командной строки и поля конфигурационного файла для данной страницы.

Для модификации полей этой страницы могут потребоваться права привилегированного пользователя. Получить такие права можно, включив вашего пользователя в группу суперпользователя "root", или, войдя на станцию от имени суперпользователя "root".

Нужно отметить ещё один важный момент: поля идентификаторов всех объектов OpenSCADA недопустимы для прямого редактирования, поскольку являются ключом для хранения данных объектов в БД. Однако поменять идентификатор объекта можно с помощью команды переноса и последующей вставки объекта (Cut->Paste) в конфигураторе.

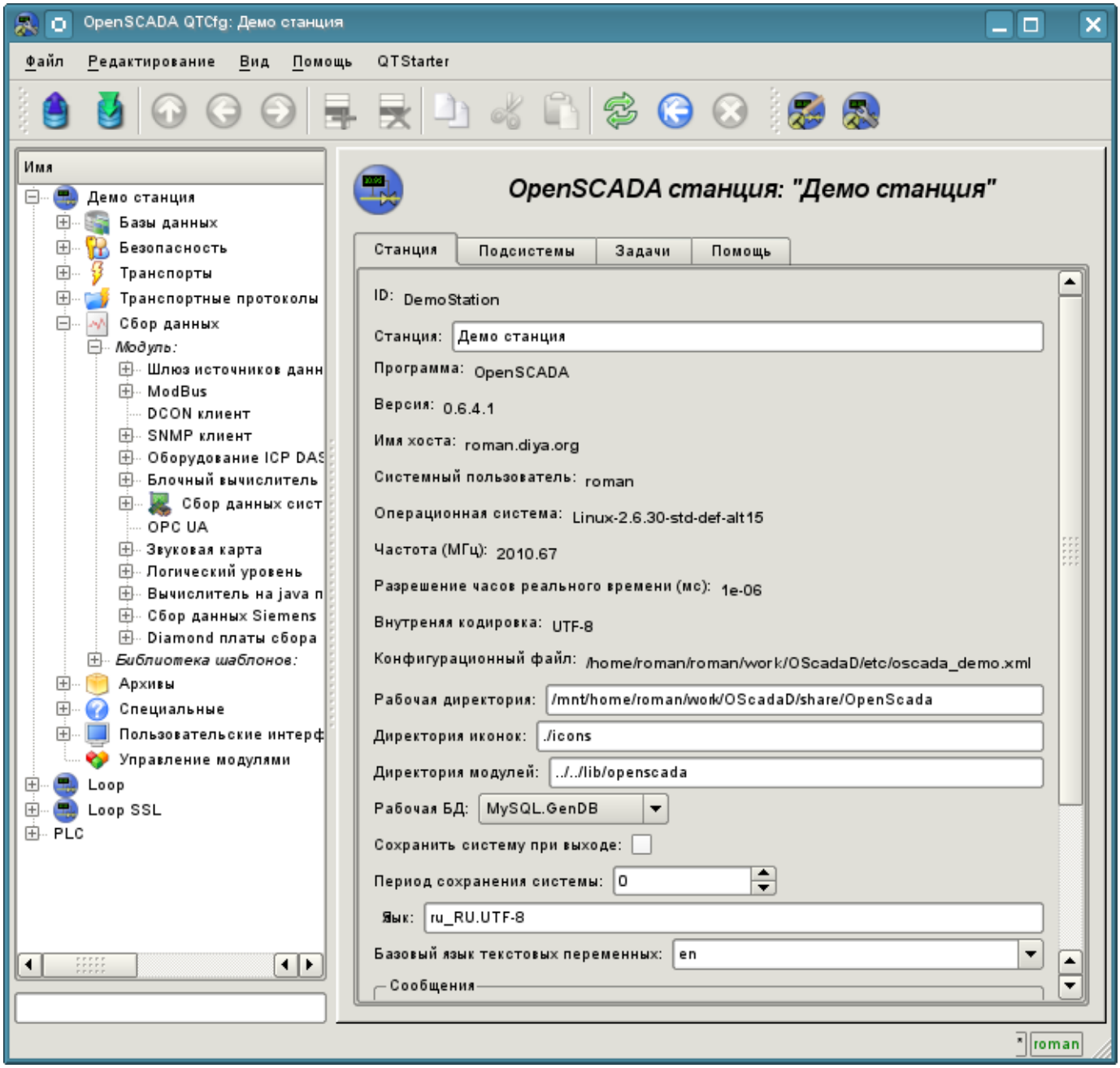

Рис. 4а. Вкладка "Станция" корневой страницы конфигурации системы.

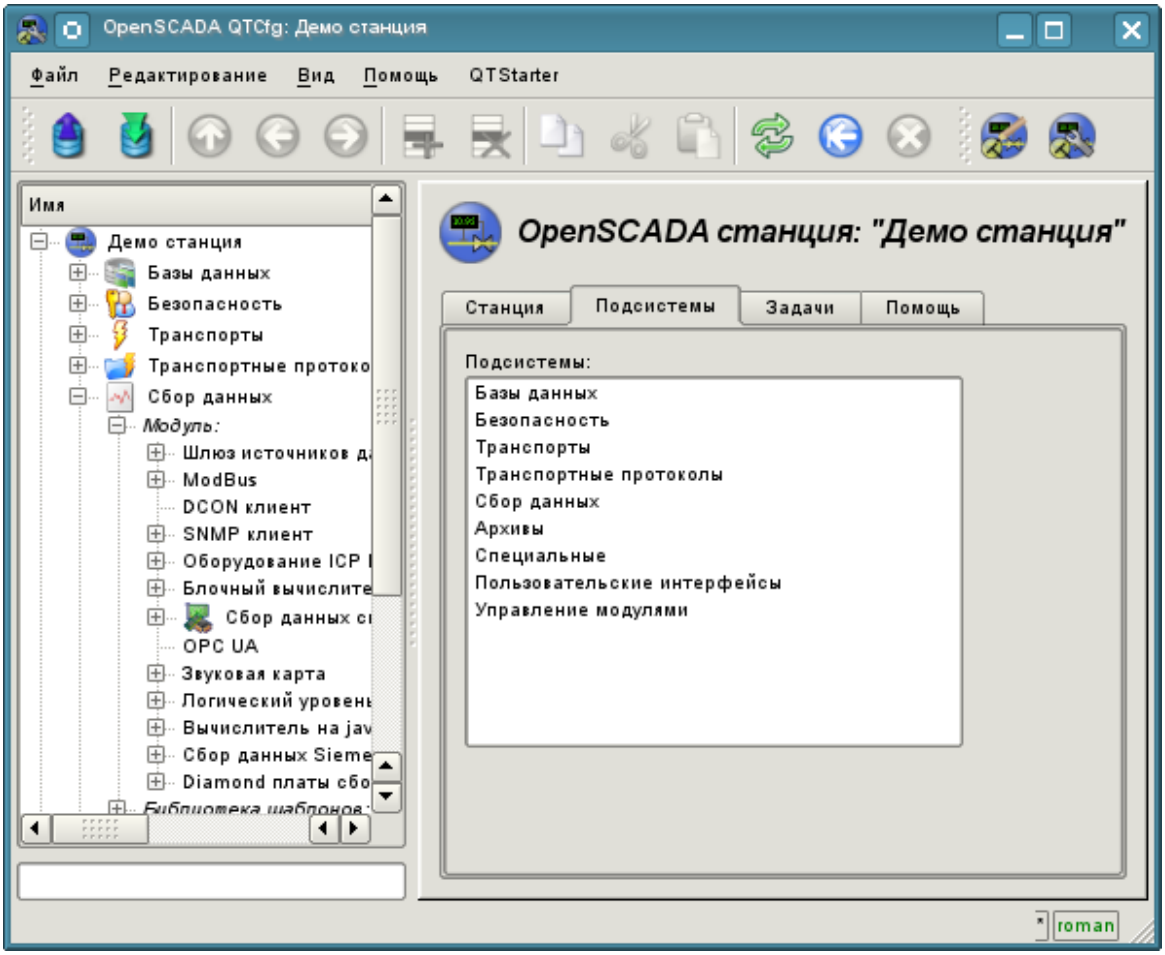

Рис. 4b. Вкладка "Подсистемы" корневой страницы конфигурации системы.

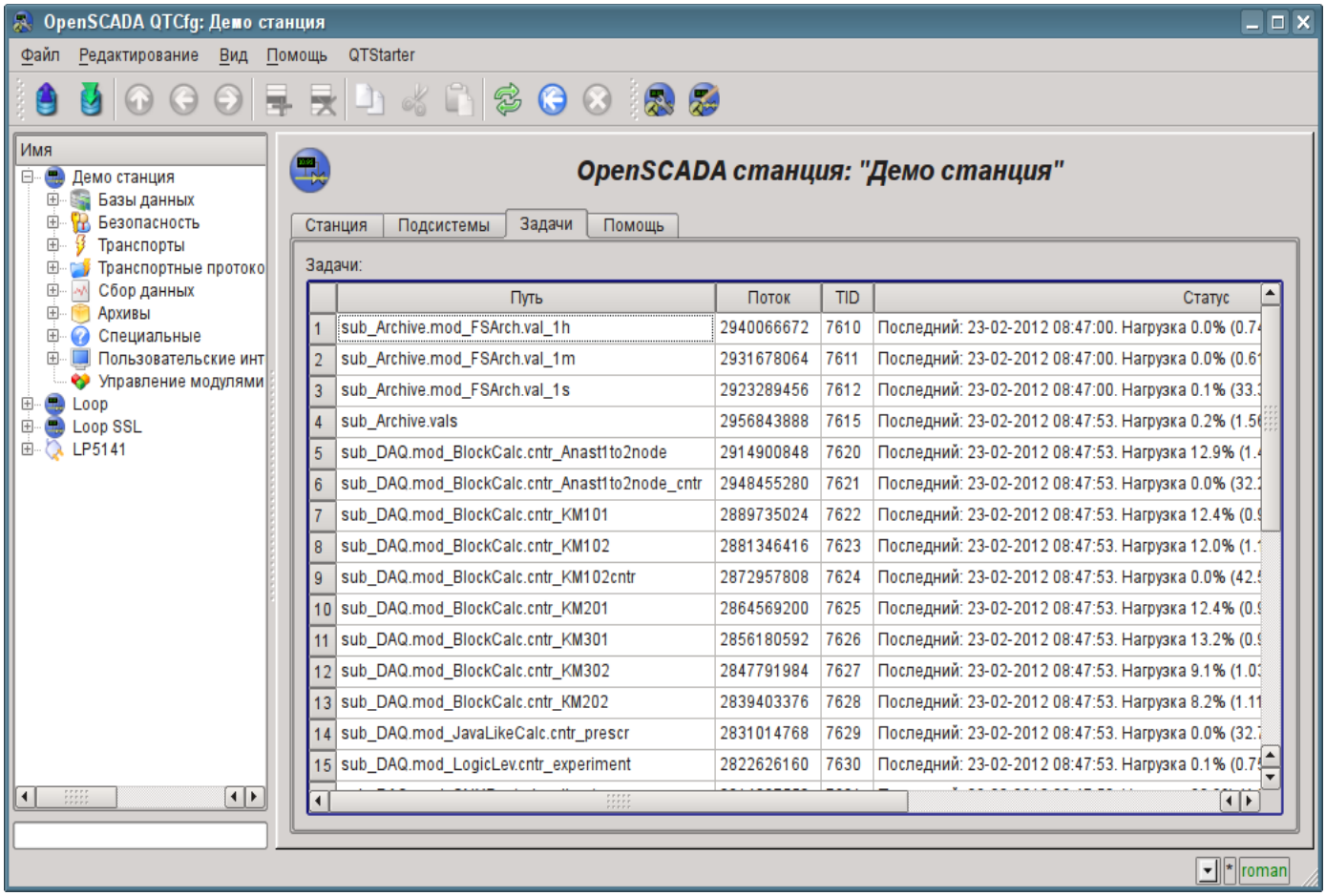

Рис. 4c. Вкладка "Задачи" корневой страницы конфигурации системы.

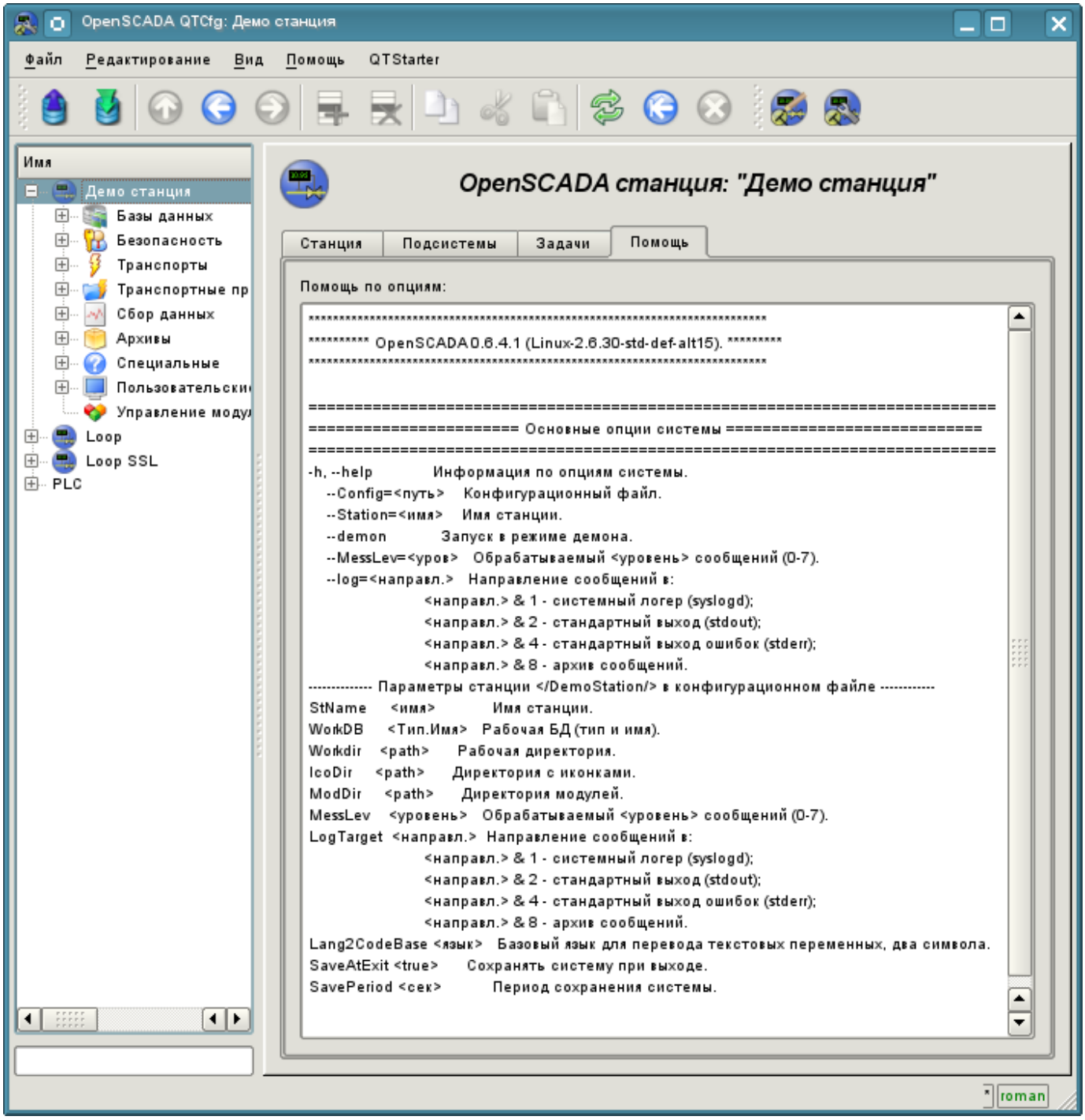

Рис. 4d. Вкладка "Помощь" корневой страницы конфигурации системы.

При рассмотрении страниц конфигурации модульных подсистем будут описаны общие для всех модулей свойства. Однако нужно отметить, что каждый модуль может предоставлять как дополнительные вкладки, так и отдельные поля для конфигурации собственных особенностей функционирования для страниц, объекты которых наследуются модулями. Об особенностях и дополнениях модулей можно ознакомиться в отдельной документации на них.

4.1. Подсистема "БД"

Подсистема является модульной и содержит иерархию объектов изображённую на рис.4.1a. Для конфигурации подсистемы предусмотрена корневая страница подсистемы "БД", содержащая вкладки "Модули" и "Помощь". Вкладка "Модули" (рис.4.1b) содержит список модулей подсистемы "БД", доступных на станции. Вкладка "Помощь" содержит краткую помощь для данной страницы.

Для модификации полей страниц этой подсистемы могут потребоваться права привилегированного пользователя или включение вашего пользователя в группу "БД".

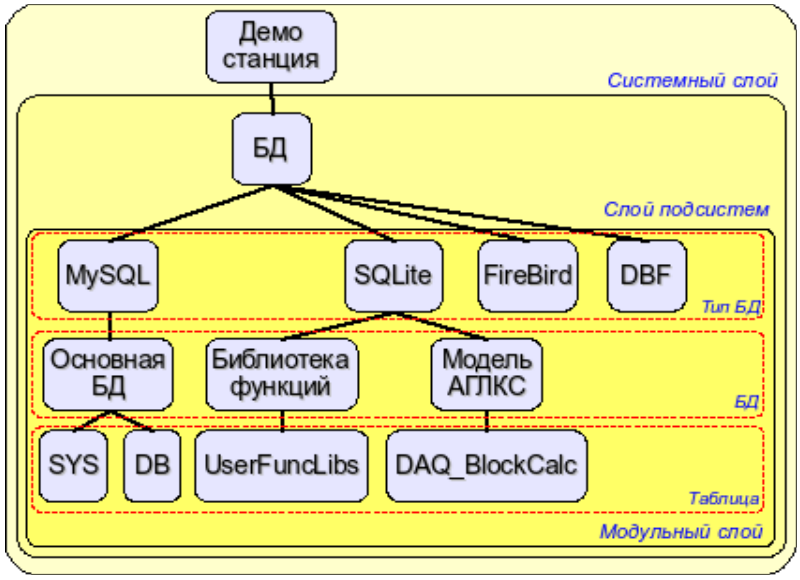

Рис. 4.1a. Иерархическая структура подсистемы "БД".

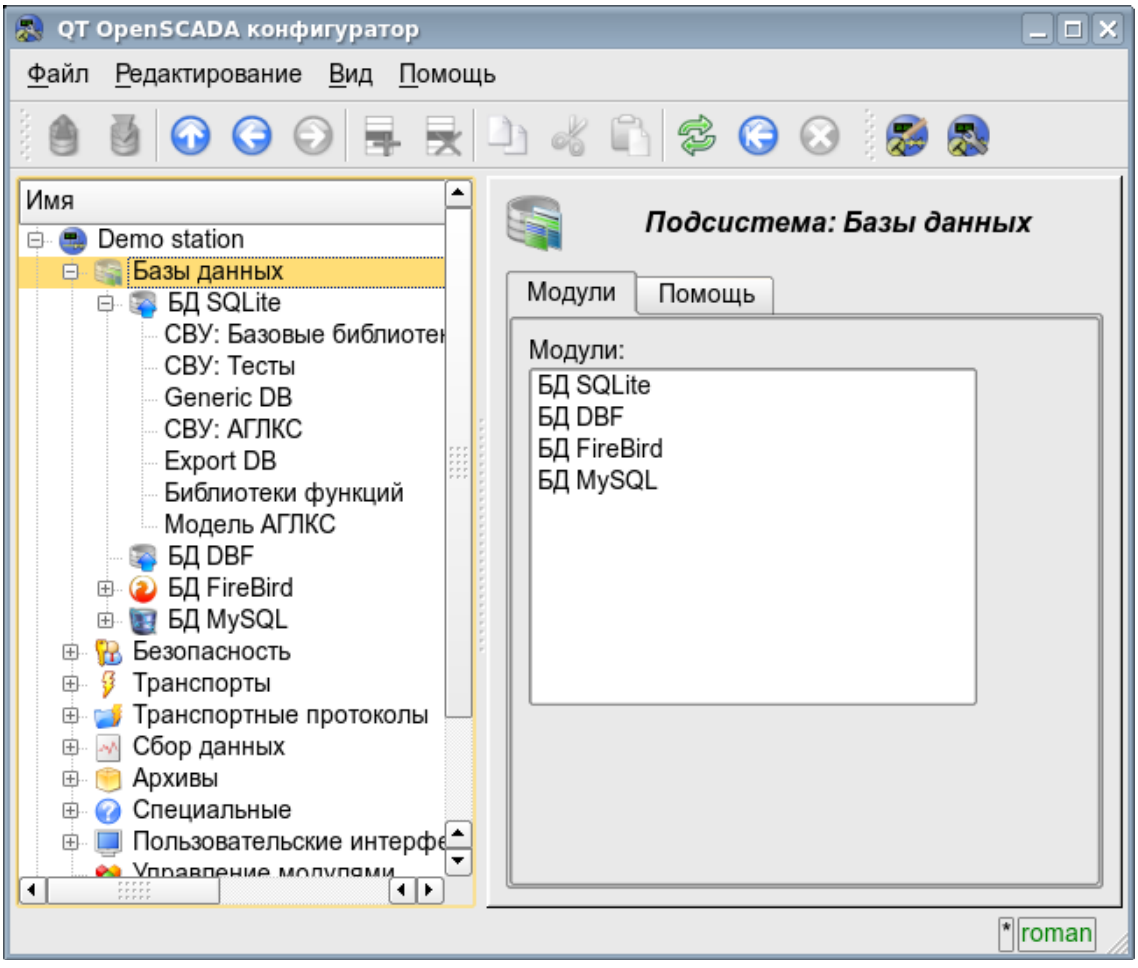

Рис. 4.1b. Вкладка "Модули" корневой страницы подсистемы "БД".

Каждый модуль подсистемы "БД" предоставляет конфигурационную страницу с вкладкам "БД" и "Помощь". Вкладка "БД" (рис.4.1c) содержит список БД, зарегистрированных в модуле и флажок признака полного удаления БД при выполнении команды удаления. В контекстном меню списка БД предоставляется пользователю возможность добавления, удаления и перехода к нужной БД. Во вкладке "Помощь" содержится информация о модуле подсистемы "БД" (рис.4.1d):

- *Модуль* — идентификатор модуля.
- *Имя* — имя модуля.
- *Тип* — тип модуля, идентификатор подсистемы, к которой модуль принадлежит.
- *Источник* — разделяемая библиотека — источник данного модуля.
- *Версия* — версия модуля.
- *Автор* — автор модуля.
- *Описание* — краткое описание модуля.
- *Лицензия* — лицензионное соглашение распространения модуля.

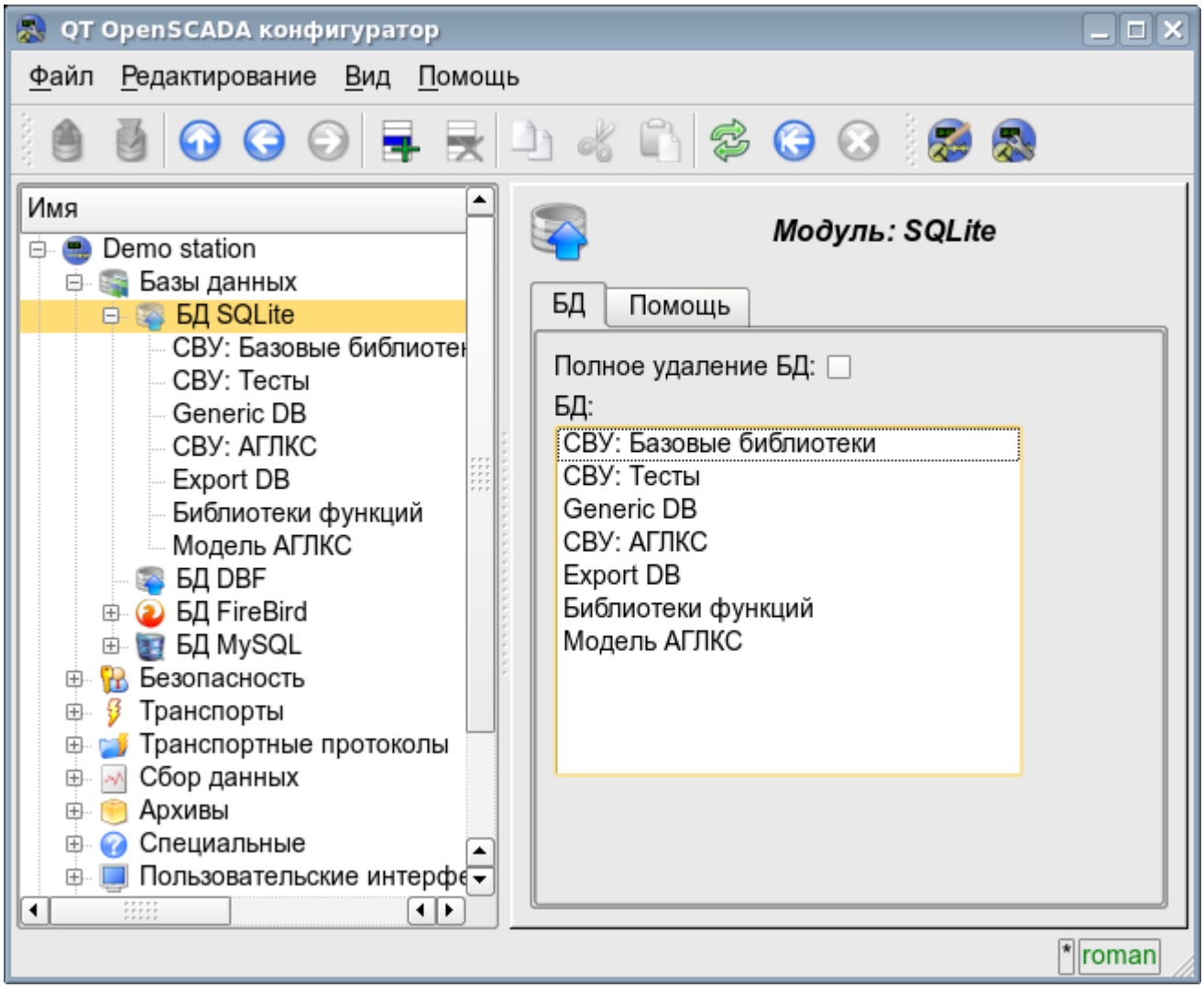

Рис. 4.1c. Вкладка "БД" модуля подсистемы "БД".

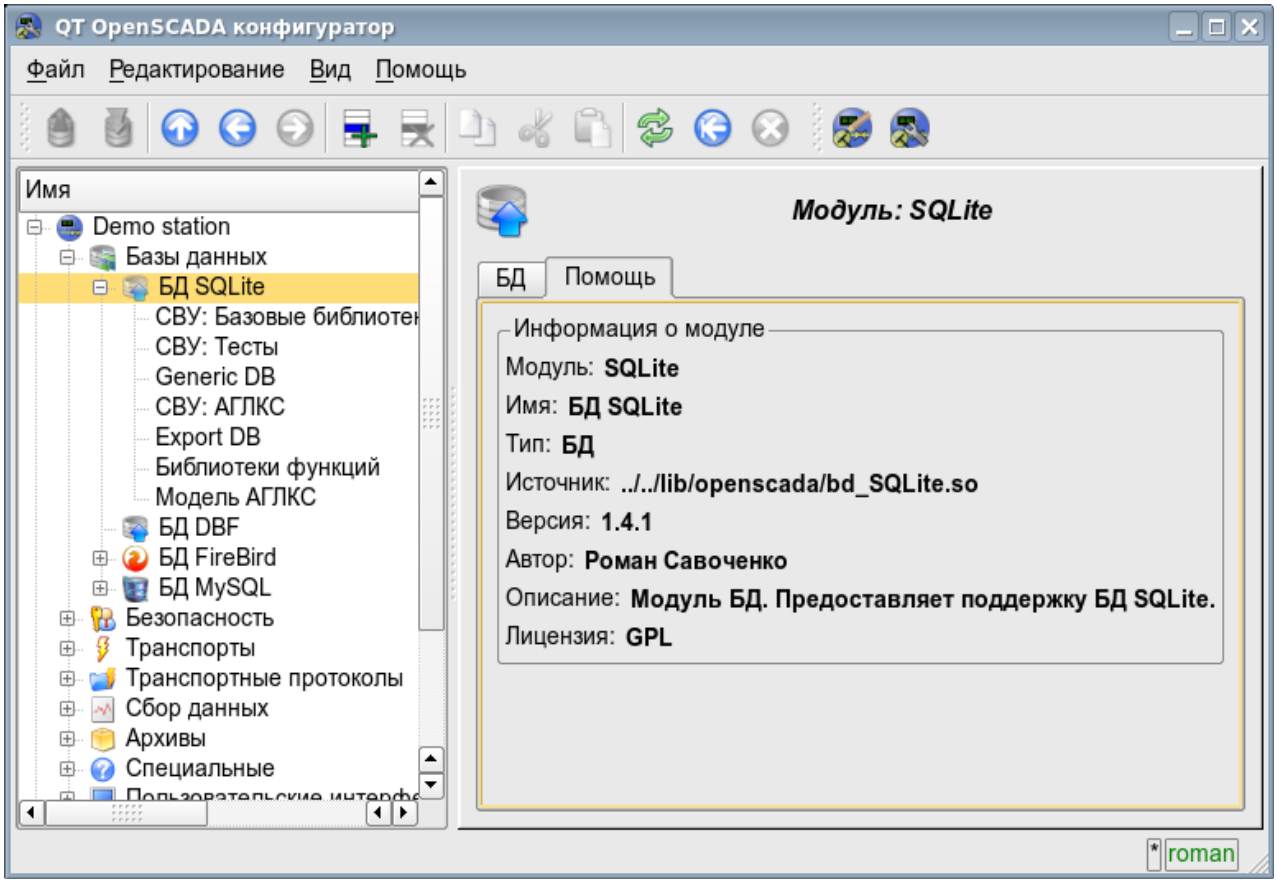

Рис. 4.1d. Вкладка "Помощь" модуля подсистемы "БД".

Каждая БД содержит собственную страницу конфигурации с вкладками "База данных", "Таблицы" и "SQL", в случае поддержки SQL-запросов. Кроме основных операций можно выполнять копирование содержимого БД стандартной функцией копирования объектов в конфигураторе. Операция копирования содержимого БД подразумевает копирование исходной БД в БД назначения, при этом содержимое БД назначения не очищается перед операцией копирования. Копирование содержимого БД производится при условии включения обоих БД, иначе будет выполняться простое копирование объекта БД.

Вкладка "База данных" (рис.4.1e) содержит основные настройки БД в составе:

- Раздел "Состояние" — содержит свойства, характеризующие состояние БД:
  - *Включен* — состояние БД "Включен".
  - *Доступные БД* — перечень таблиц, которые содержит БД. Контекстным меню данного свойства предоставляется возможность физического удаления таблиц из БД.
  - *Загрузить систему из БД* — команда для выполнения загрузки из данной БД. Может использоваться при переносе данных в БД между станциями. Например, можно сохранить участок одной станции в экспортную БД, физически перенести БД на другую станцию, подключить её в этой подсистеме и вызвать данную команду.
- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — содержит информацию об идентификаторе БД.
  - *Имя* — указывает имя БД.
  - *Описание* — краткое описание БД и её назначения.
  - *Адрес* — адрес БД в специфичном для типа БД (модуля) в формате. Описание формата записи адреса БД, как правило, доступно во всплывающей подсказке этого поля.
  - *Кодовая страница* — указывает на кодовую страницу, в которой хранятся и предоставляются текстовые значения БД. Значение кодовой страницы БД в связке с внутренней кодировкой станции используется для прозрачного перекодирования текстовых сообщений при обмене между станцией и БД.
  - *Включать* — указывает на состояние "Включен", в которое переводить БД при загрузке.

Вкладка "Таблицы" (рис.4.1f) содержит список открытых таблиц. При нормальной работе программы эта вкладка пуста, поскольку после завершения работы с таблицами программа их закрывает. Наличие открытых таблиц говорит о том, что программа сейчас с таблицами работает или таблицы открыты пользователем для изучения их содержимого. В контекстном меню перечня открытых таблиц можно открыть таблицу для изучения (команда "Добавить"), закрыть открытую страницу (команда "Удалить") и перейти к изучению содержимого таблицы.

Вкладка "SQL" (рис.4.1g) доступна только для баз данных, поддерживающих SQL-запросы, и содержит поле ввода запроса, кнопку отправки запроса и таблицу с результатом запроса. Для управления контекстом транзакции запроса предусмотрено отдельное конфигурационное поле.

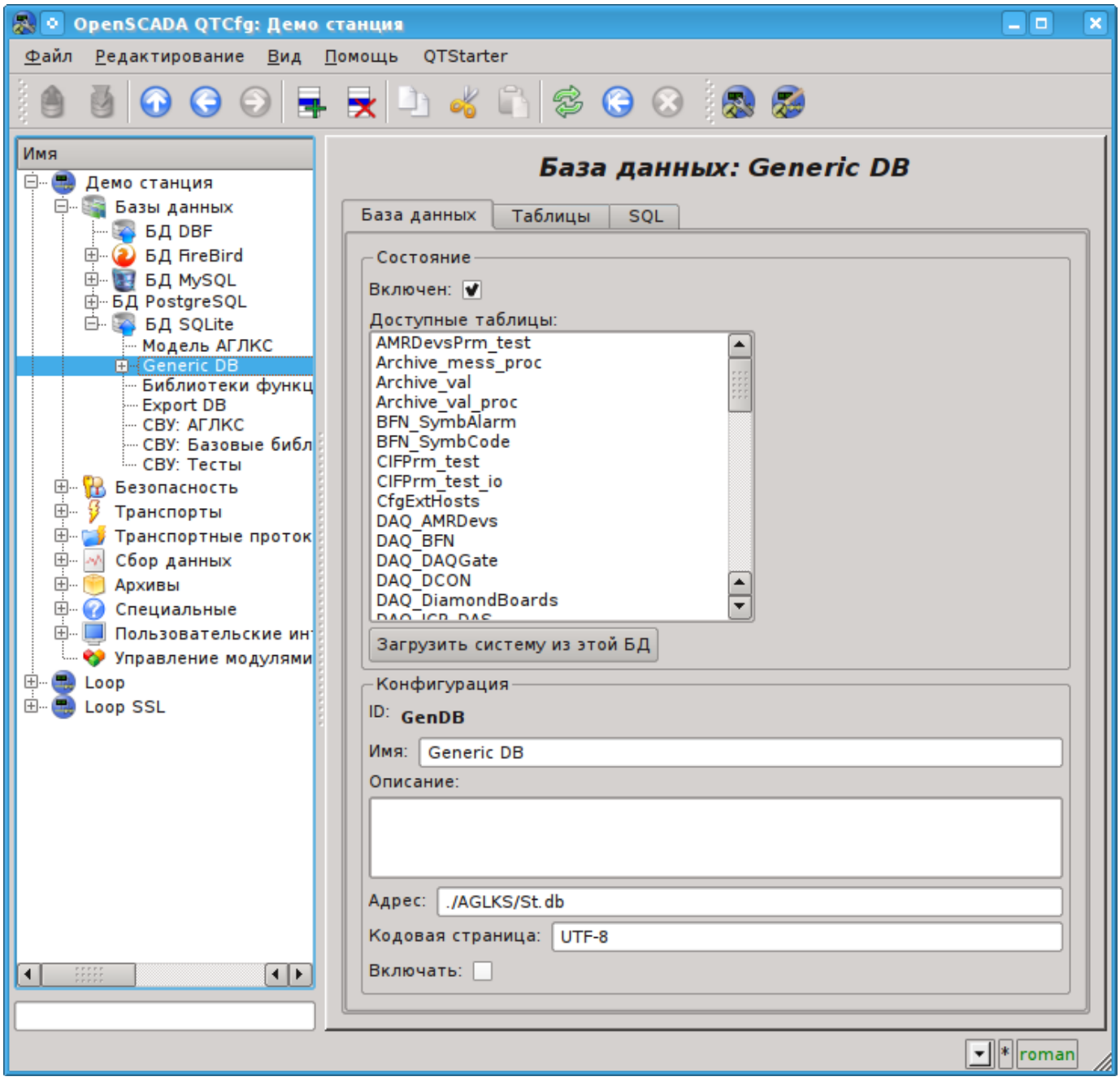

Рис. 4.1e. Вкладка "База данных" БД модуля подсистемы "БД".

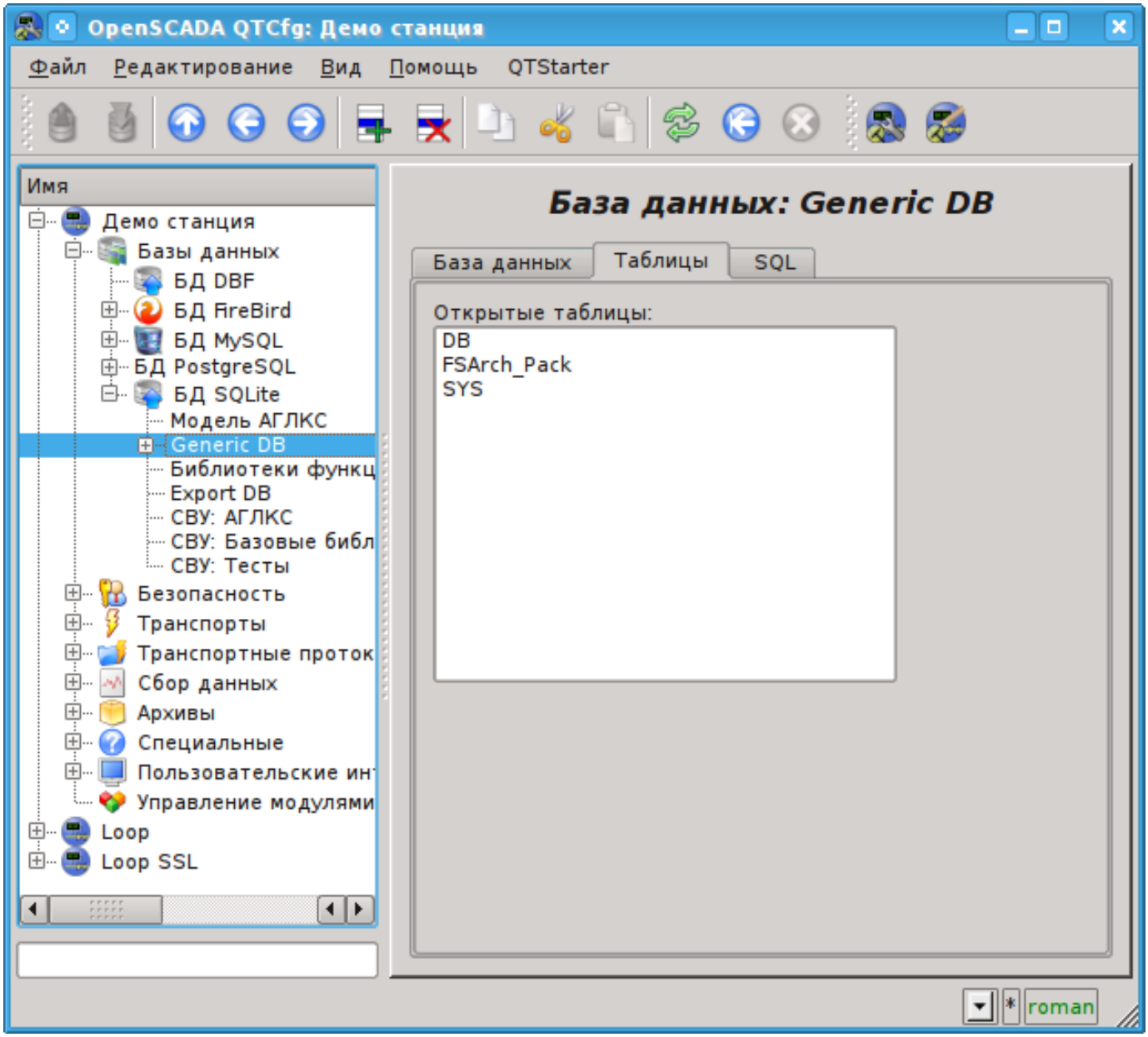

Рис. 4.1f. Вкладка "Таблицы" БД модуля подсистемы "БД".

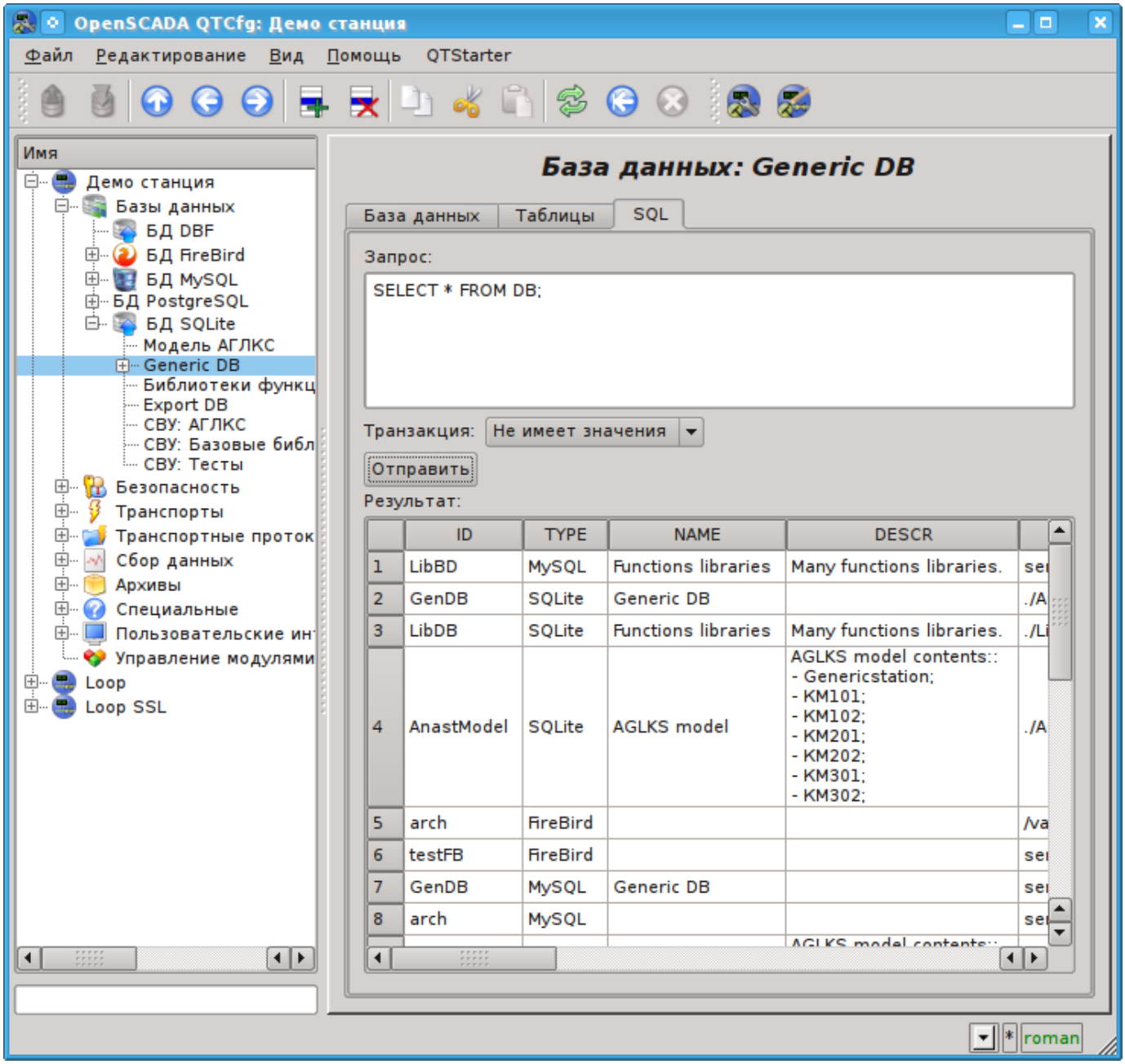

Рис. 4.1g. Вкладка "SQL" БД модуля подсистемы "БД".

Страница изучения содержимого таблицы содержит только одну вкладку "Таблица". Вкладка "Таблица" (рис.4.1h) содержит поле имени таблицы и таблицу с содержимым. Таблицей содержимого предоставляются функции:

- редактирование содержимого ячеек таблицы;
- добавление записи (строки);
- удаление записи (строки).

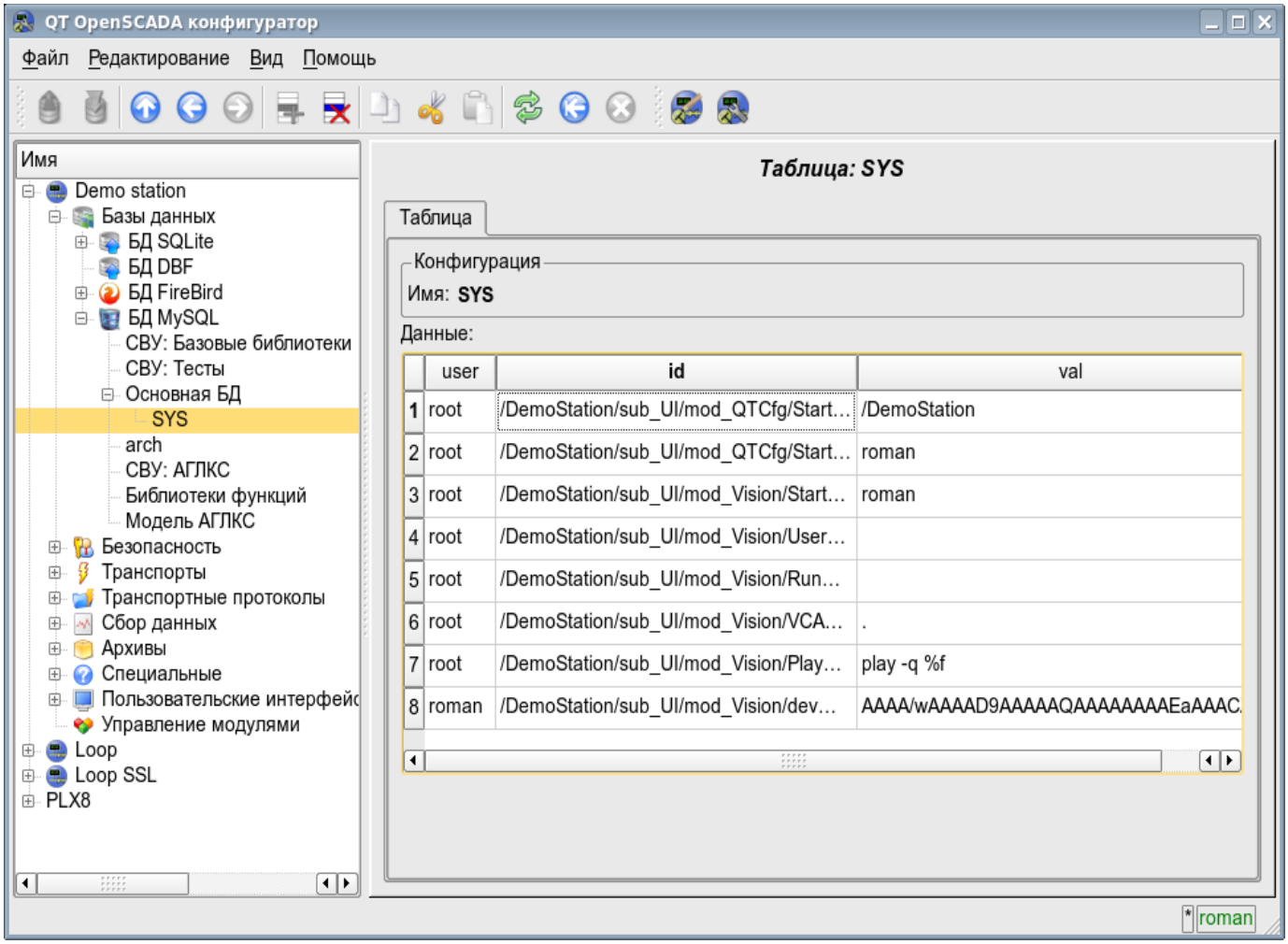

Рис. 4.1h. Вкладка "Таблица" таблицы БД модуля подсистемы "БД".

4.2. Подсистема "Безопасность"

Подсистема не является модульной. Для конфигурации подсистемы предусмотрена корневая страница подсистемы "Безопасность", содержащая вкладки "Пользователи и группы пользователей" и "Помощь". Вкладка "Пользователи и группы пользователей" (рис.4.2а) содержит списки пользователей и групп пользователей. Пользователь в группе "Security" и с правами привилегированного пользователя может добавить, удалить пользователя или группу пользователей. Все остальные пользователи могут перейти к странице пользователя или группы пользователя. Вкладка "Помощь" содержит краткую помощь для данной страницы.

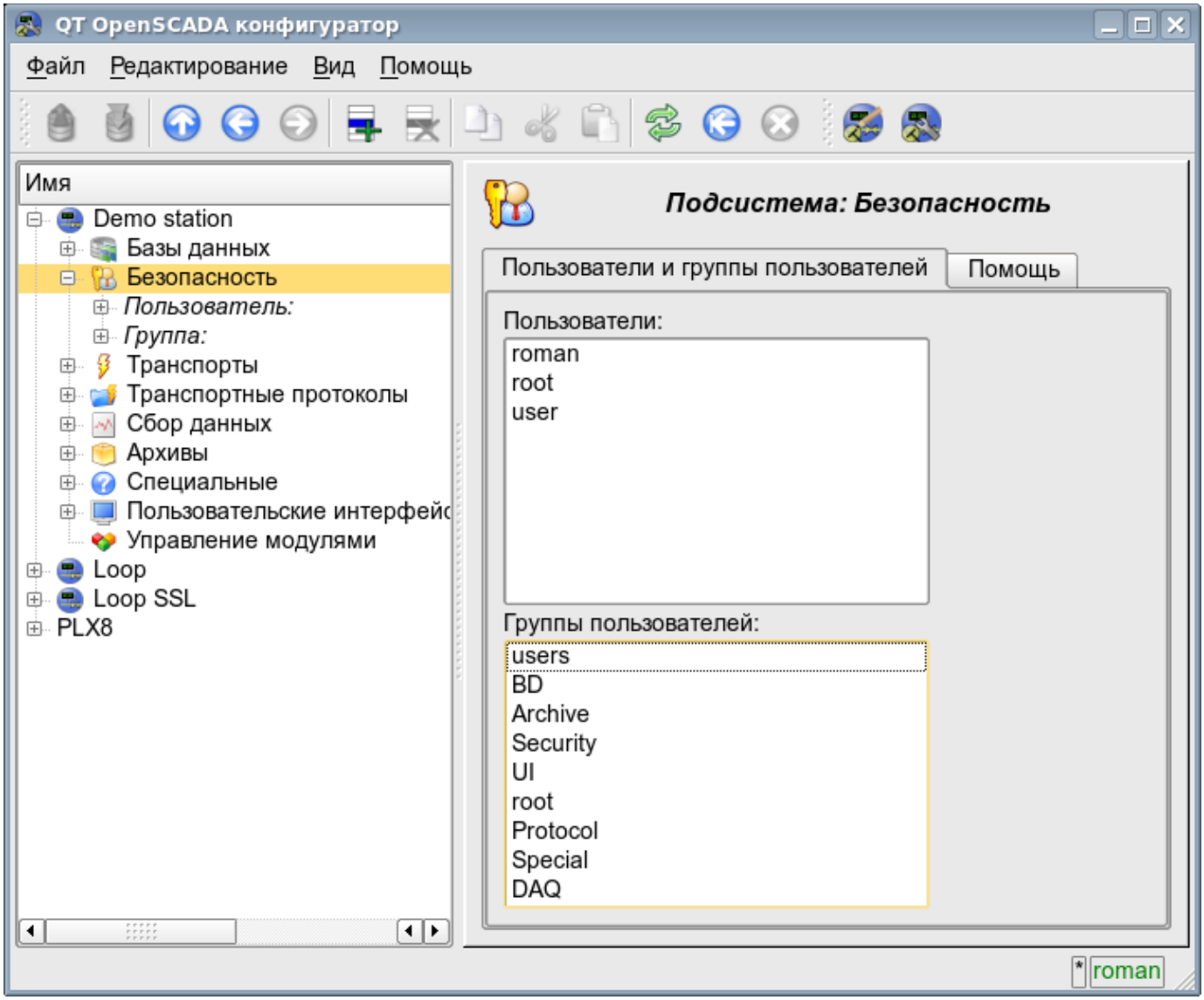

Рис. 4.2а. Вкладка "Пользователи и группы пользователей" корневой страницы подсистемы "Безопасность".

Для конфигурации пользователя предоставляется страница, содержащая только вкладку "Пользователь" (рис.4.2b). Вкладка содержит конфигурационные данные профиля пользователя, которые может изменять сам пользователь, пользователь в группе "Security" или привилегированный пользователь:

- *Имя* — информация об имени (идентификаторе) пользователя.
- *Полное имя* — указывает на полное имя пользователя.
- *Изображение пользователя* — указывает изображение пользователя. Изображение может быть загружено или выгружено.
- *БД пользователя* — адрес БД для хранения данных пользователя.
- *Пароль* — поле для изменения пароля пользователя. Всегда отображает "******".
- *Группы пользователей* — таблица с перечнем групп пользователей станции и признаком принадлежности пользователя к группам.

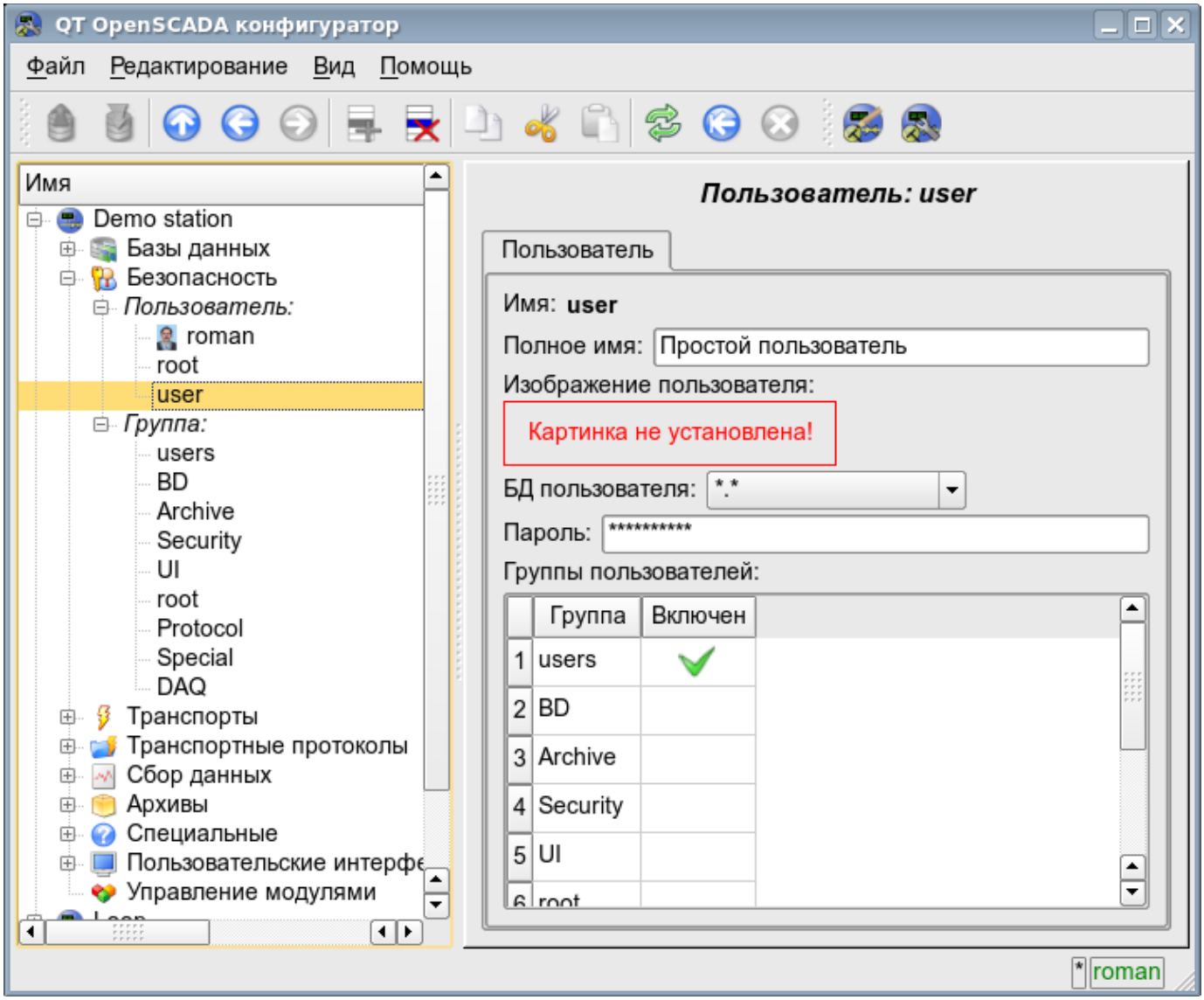

Рис. 4.2b. Вкладка "Пользователь" страницы пользователя подсистемы "Безопасность".

Для конфигурации группы пользователей предоставляется страница, содержащая только вкладку "Группа" (рис.4.2c). Вкладка содержит конфигурационные данные профиля группы пользователей, которые может изменять только привилегированный пользователь:

- *Имя* — информация об имени (идентификаторе) группы пользователей.
- *Полное имя* — указывает на полное имя группы пользователей.
- *БД группы пользователей* — адрес БД для хранения данных группы пользователей.
- *Пользователи* — список пользователей, включенных в данную группу. С помощью контекстного меню списка можно добавить или удалить пользователя в группе.

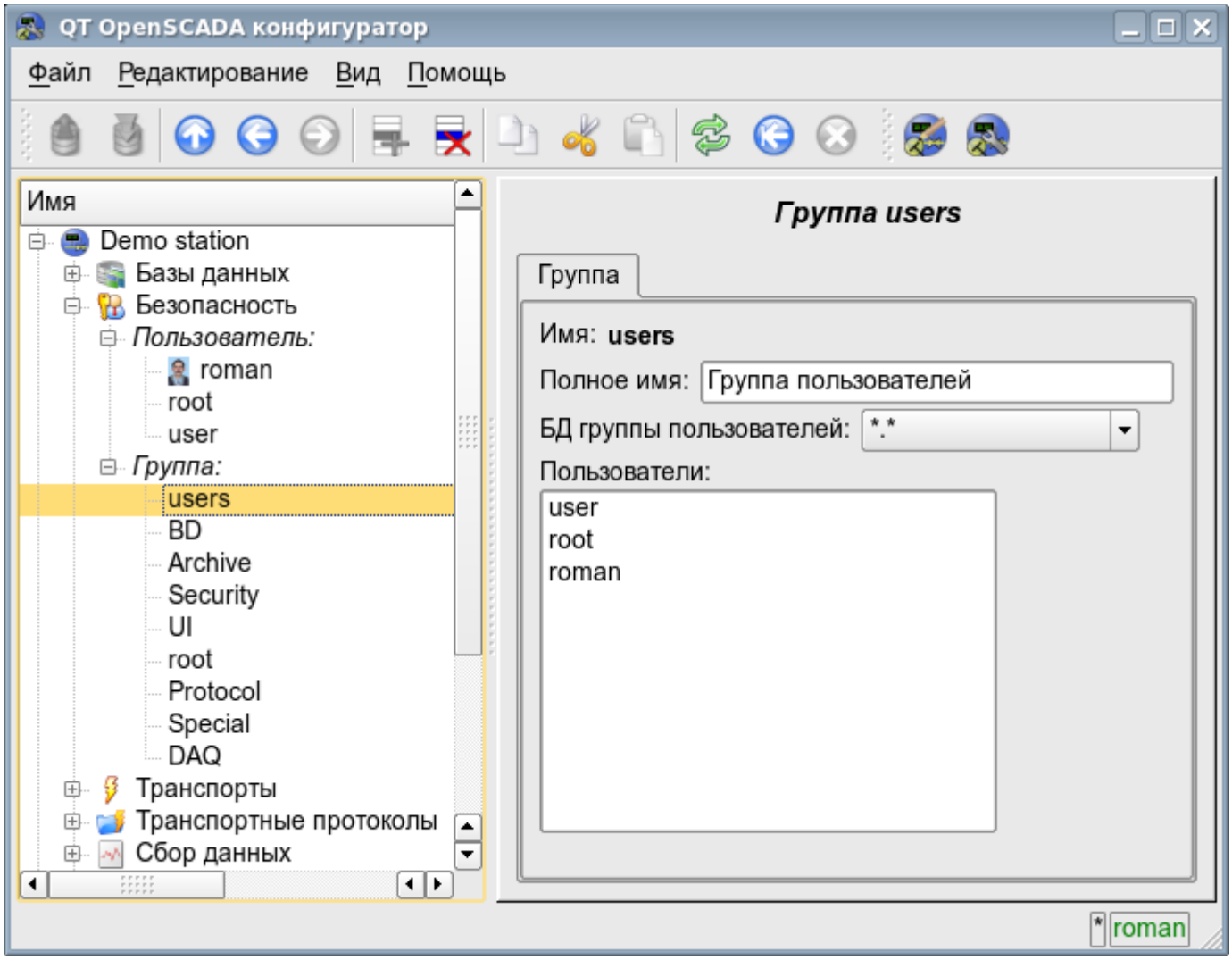

Рис. 4.2c. Вкладка "Группа" страницы группы пользователей подсистемы "Безопасность".

4.3. Подсистема "Транспорты"

Подсистема является модульной и содержит иерархию объектов, изображённую на рис.4.3a. Для конфигурации подсистемы предусмотрена корневая страница подсистемы "Транспорты", содержащая вкладки "Подсистема", "Модули" и "Помощь".

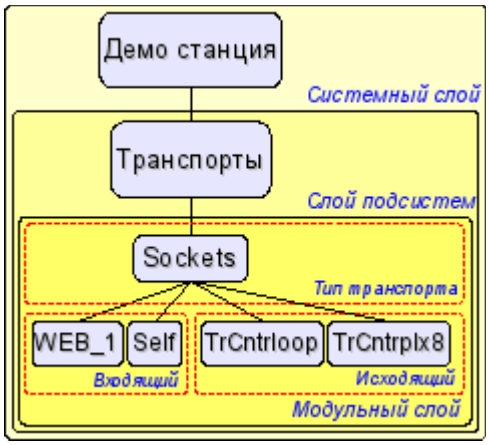

Рис. 4.3a. Иерархическая структура подсистемы "Транспорты".

Вкладка "Подсистема" (рис.4.3b) содержит таблицу конфигурации внешних для данной OpenSCADA станций. Внешние станции могут быть системными и пользовательскими, что выбирается соответствующим параметром. Системные внешние станции доступны только привилегированному пользователю и используются компонентами системного назначения, например, механизмом горизонтального резервирования и модулем DAQ.DAQGate. Пользовательские внешние станции привязаны к пользователю, который их создавал, а значит список пользовательских внешних станций индивидуален для каждого пользователя. Пользовательские внешние станции используются компонентами графического интерфейса, например, UI.QTCfg, UI.WebCfgD и UI.Vision. В таблице внешних станций возможно добавление и удаление записей про станцию, а также их модификация. Каждая запись содержит поля:

- *Id* — идентификатор внешней станции.
- *Имя* — имя внешней станции.
- *Транспорт* — выбор из списка модуля подсистемы "Транспорты" для использования его в доступе к внешней станции.
- *Адрес* — адрес внешней станции в формате, специфичном для выбранного в предыдущем поле модуля подсистемы "Транспорты".
- *Пользователь* — имя/идентификатор пользователя удалённой станции, от имени которого выполнять подключение.
- *Пароль* — пароль пользователя удалённой станции.

Вкладка "Модули" (рис.4.1b) содержит список модулей подсистемы "Транспорты" и идентична для всех модульных подсистем. Вкладка "Помощь" содержит краткую помощь для данной страницы.

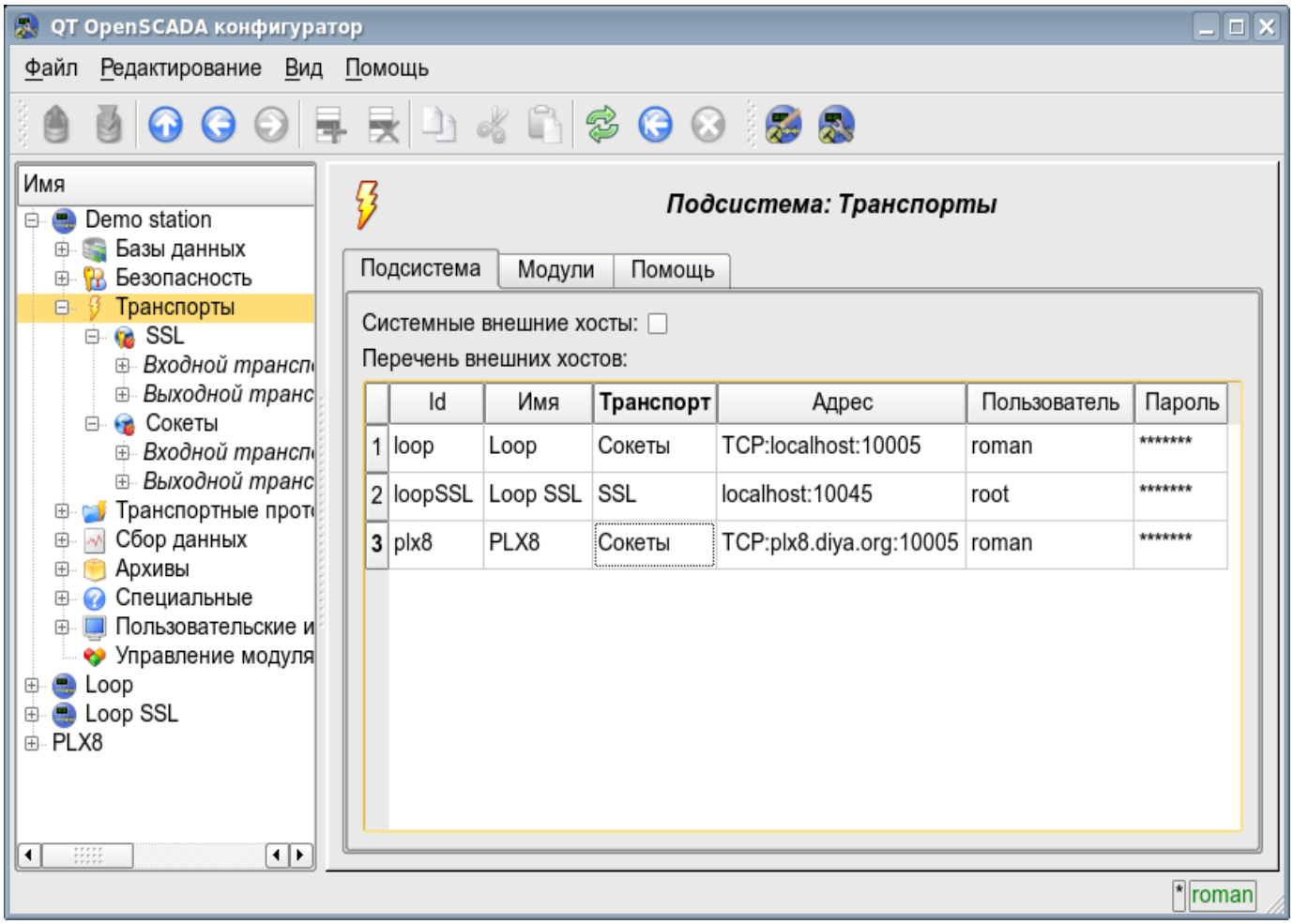

Рис. 4.3b. Вкладка "Подсистема" корневой страницы подсистемы "Транспорты".

Каждый модуль подсистемы "Транспорты" предоставляет конфигурационную страницу с вкладками "Транспорты" и "Помощь". Вкладка "Транспорты" (рис.4.3c) содержит список входящих и исходящих транспортов, зарегистрированных в модуле. В контекстном меню списков транспортов пользователю предоставляется возможность добавления, удаления и перехода к нужному транспорту. Во вкладке "Помощь" содержится информация о модуле подсистемы "Транспорты" (рис.4.1d), состав которой идентичен для всех модулей.

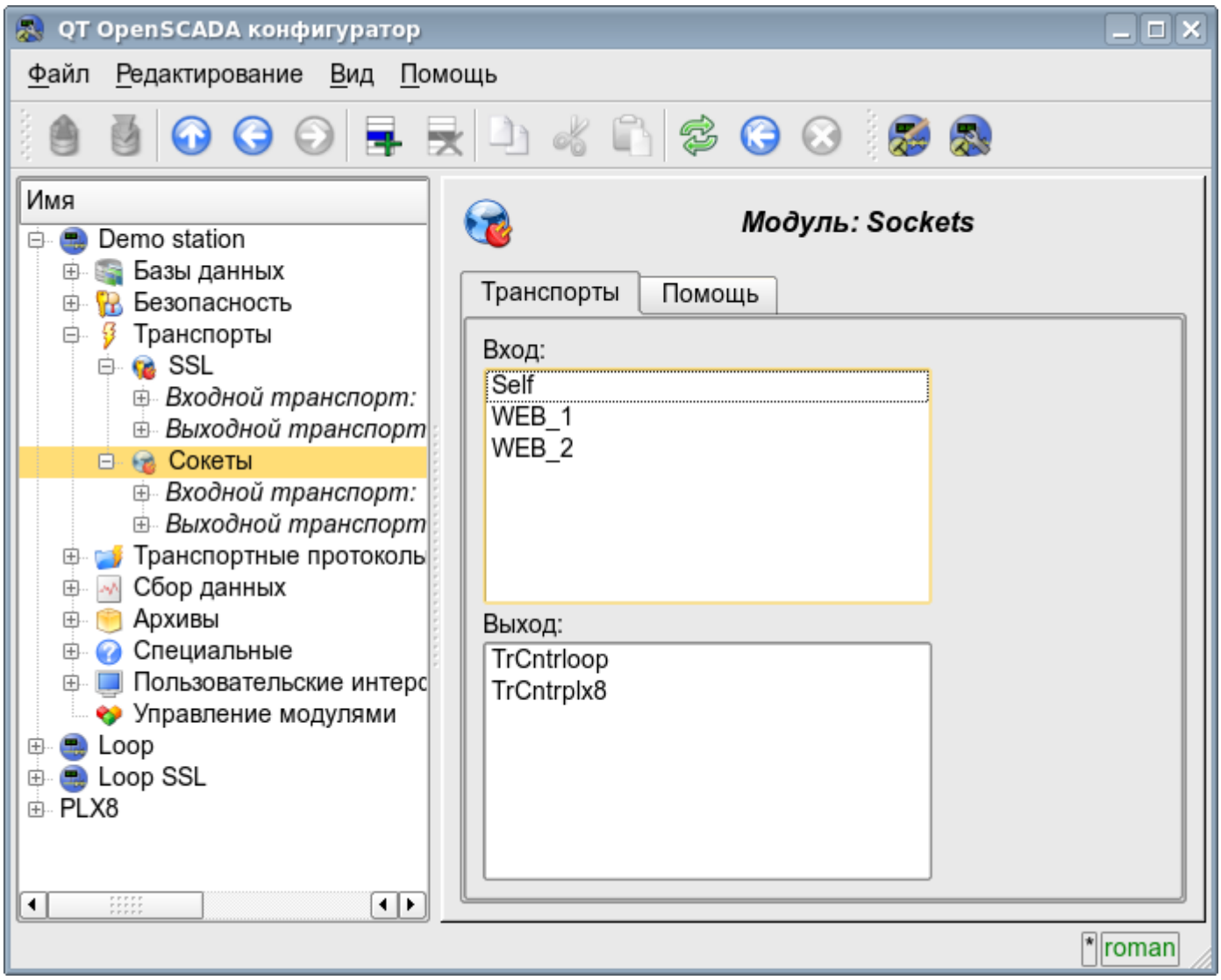

Рис. 4.3c. Вкладка "Транспорты" модуля подсистемы "Транспорты".

Каждый транспорт содержит собственную страницу конфигурации с одной вкладкой "Транспорт". Эта вкладка содержит основные настройки транспорта. Входящий транспорт (рис.4.3d) содержит:

- Раздел "Состояние" — содержит свойства, характеризующие состояние транспорта:
  - *Статус* — информация о текущем состоянии транспорта и статистика его работы.
  - *Выполняется* — состояние транспорта "Выполняется".
  - *БД транспорта* — адрес БД для хранения данных транспорта.
- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — информация об идентификаторе транспорта.
  - *Имя* — указывает имя транспорта.
  - *Описание* — краткое описание транспорта и его назначения.
  - *Адрес* — адрес транспорта в специфичном для типа транспорта (модуля) формате. Описание формата записи адреса транспорта, как правило, доступно во всплывающей подсказке этого поля.
  - *Транспортный протокол* — указывает на модуль транспортного протокола (подсистема "Транспортные протоколы"), который должен работать в связке с данным входным транспортом. Т.е. полученные неструктурированные данные этот модуль будет направлять на структуризацию и обработку указанному модулю транспортного протокола.
  - *Запускать* — указывает на состояние "Выполняется", в которое переводить транспорт при загрузке.

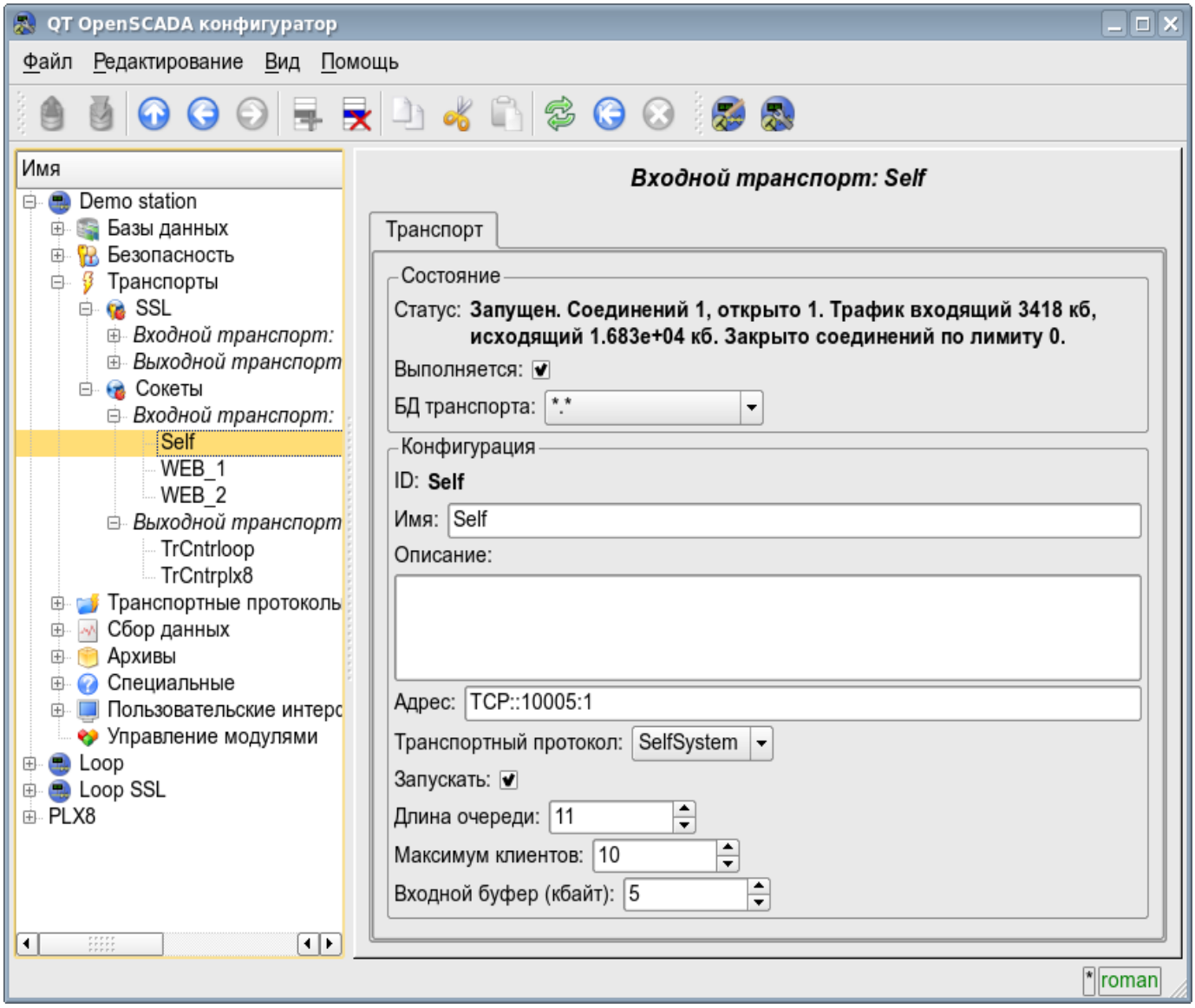

Рис. 4.3d. Вкладка "Транспорт" страницы входящего транспорта модуля подсистемы "Транспорты".

Исходящий транспорт (рис.4.3е) содержит:

- Раздел "Состояние" — содержит свойства, характеризующие состояние транспорта:
  - *Статус* — информация о текущем состоянии транспорта и статистика его работы.
  - *Выполняется* — состояние транспорта "Выполняется".
  - *БД транспорта* — адрес БД для хранения данных транспорта.
- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — информация об идентификаторе транспорта.
  - *Имя* — указывает имя транспорта.
  - *Описание* — краткое описание транспорта и его назначения.
  - *Адрес* — адрес транспорта в специфичном для типа транспорта (модуля) формате. Описание формата записи адреса транспорта, как правило, доступно во всплывающей подсказке этого поля.
  - *Запускать* — указывает на состояние "Выполняется", в которое переводить транспорт при загрузке.

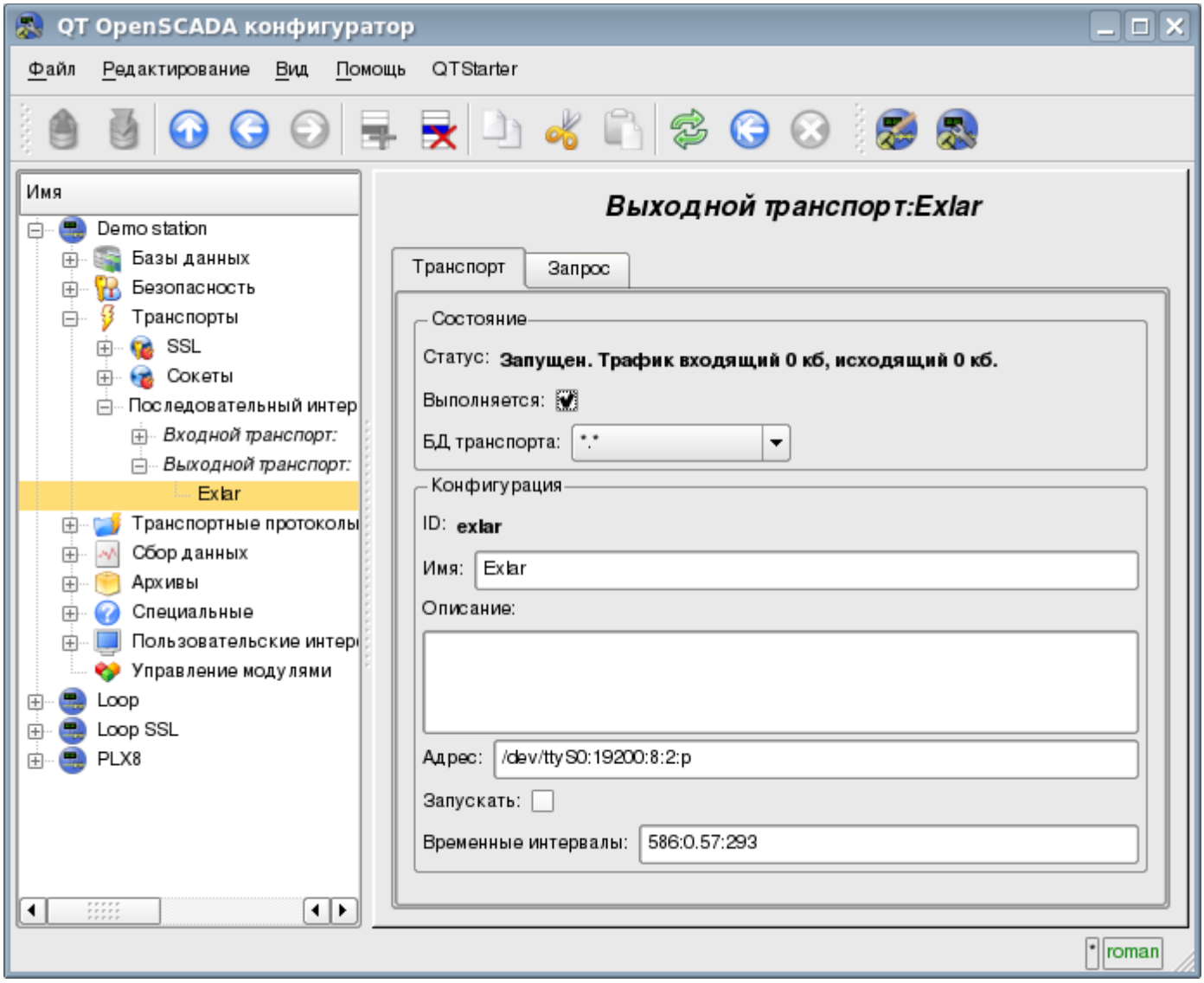

Рис. 4.3е. Вкладка "Транспорт" страницы исходящего транспорта модуля подсистемы "Транспорты".

Исходящий транспорт, в дополнение, предоставляет вкладку формирования пользовательского запроса через данный транспорт (рис.4.3f). Вкладка предназначена для наладки связи, а также для отладки протоколов и содержит:

- *Время (мс)* — информация о времени, затраченном на запрос и получение ответа.
- *Режим* — указывает режим данных, из списка "Бинарный", "Текст(LF)", "Текст(CR)", "Текст(CR/LF)", в котором будет формироваться запрос и предоставляться ответ. В бинарном режиме данные записываются парами чисел в шестнадцатеричном исчислении, т.е байтами, разделёнными пробелами.
- *Ожидать таймаута* — признак ожидать таймаута при получении ответа. Многие системы при ответе на различных протоколах (HTTP) могут слать данные ответа несколькими кусками. Без этого флага будет получен и отображён только первый кусок. При установке этого флага будет осуществлено ожидание всех кусков ответа, вплоть до отсутствия данных в течении таймаута дожидания транспорта.
- *Отправить* — команда отправить запрос.
- *Запрос* — содержит запрос в выбранном режиме представления данных.
- *Ответ* — предоставляет ответ в выбранном режиме представления данных.

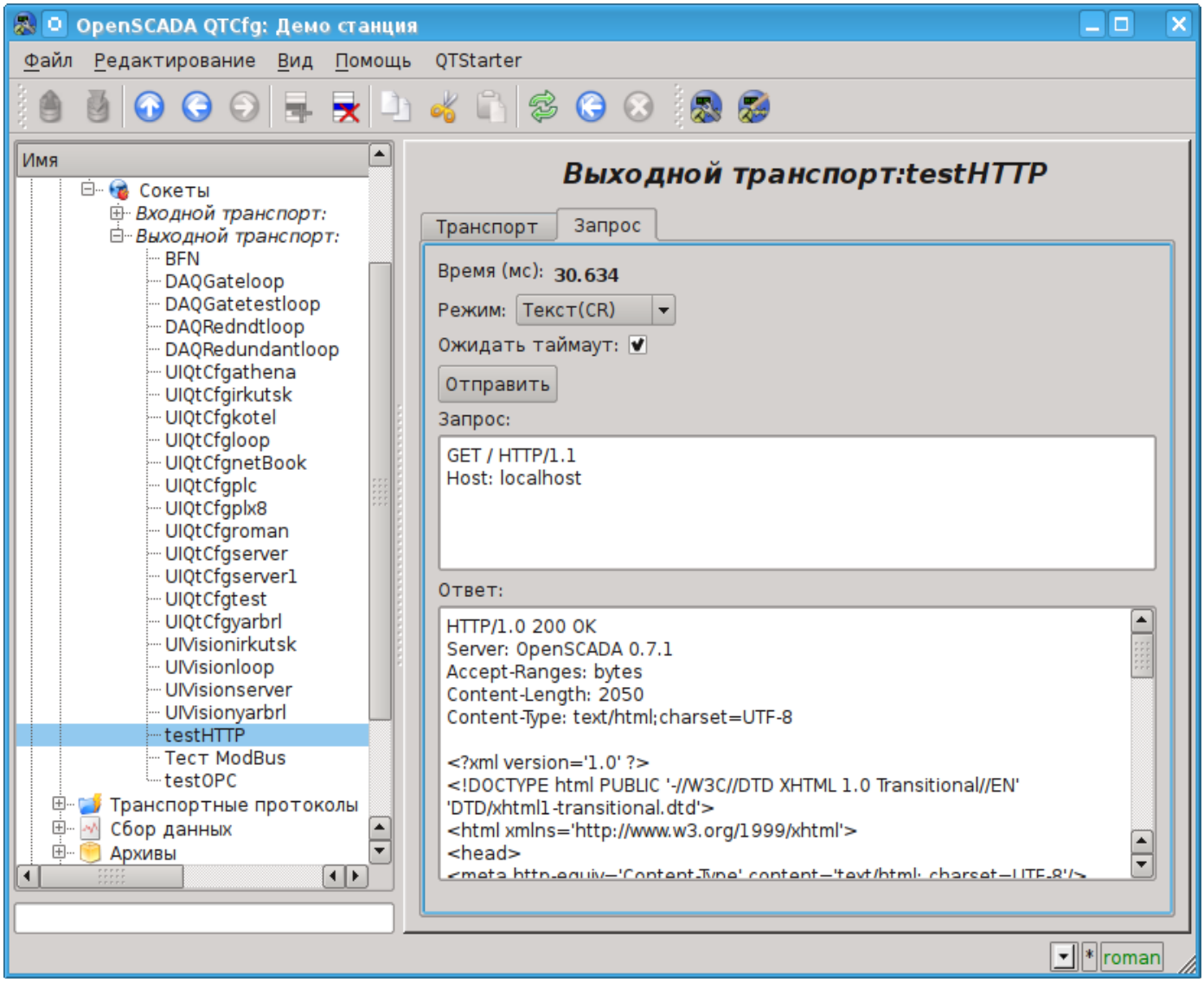

Рис. 4.3f. Вкладка "Запрос" страницы исходящего транспорта модуля подсистемы "Транспорты".

4.4. Подсистема "Транспортные протоколы"

Подсистема является модульной. Для конфигурации подсистемы предусмотрена корневая страница подсистемы "Транспортные протоколы", содержащая вкладки "Модули" и "Помощь". Вкладка "Модули" (рис.4.1b) содержит список модулей подсистемы "Транспортные протоколы" и идентична для всех модульных подсистем. Вкладка "Помощь" содержит краткую помощь для данной страницы.

Каждый модуль подсистемы "Транспортные протоколы" предоставляет конфигурационную страницу с одной вкладкой "Помощь". Во вкладке "Помощь" содержится информация о модуле подсистемы "Транспортные протоколы" (рис.4.1d), состав которой идентичен для всех модулей.

4.5. Подсистема "Сбор данных"

Подсистема является модульной и содержит иерархию объектов, изображённую на рис.4.5a. Для конфигурации подсистемы предусмотрена корневая страница подсистемы "Сбор данных", содержащая вкладки "Библиотеки шаблонов", "Модули" и "Помощь".

Для получения доступа на модификацию объектов этой подсистемы необходимы права пользователя в группе "DAQ" или права привилегированного пользователя.

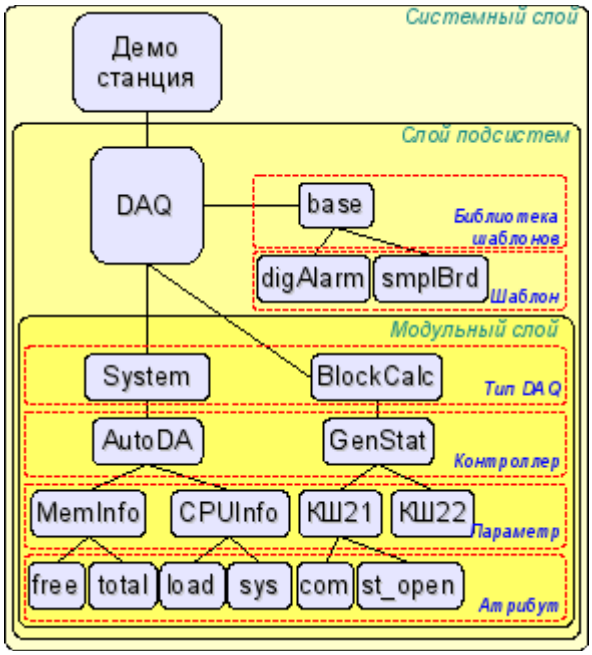

Рис. 4.5a. Иерархическая структура подсистемы "Сбор данных".

Вкладка "Резервирование" (рис.4.5b) содержит конфигурацию резервирования источников данных подсистемы "Сбор данных" станции в составе настроек:

- *Статус* — содержит информацию о работе схемы резервирования, на данный момент это время, затраченное на исполнения одного цикла задачи обслуживания резерва.
- *Уровень станции* — указывает уровень данной станции в схеме резервирования (0-255).
- *Период задачи резервирования* — указывает периодичность исполнения задачи резервирования в секундах (1-255).
- *Интервал времени восстановления соединения* — указывает через какой интервал времени осуществлять попытку восстановления соединения с потерянной резервной станцией в секундах (0-255).
- *Глубина времени восстановления данных* — указывает на максимальную глубину архивных данных для восстановления из архива удалённой станции при запуске в часах (0-12).
- *Станции* — содержит таблицу с информацией о резервных станциях. Станции можно добавлять и удалять посредством контекстного меню. Идентификатор добавленных станций нужно выбрать из списка доступных системных станций OpenSCADA. Таблица предоставляет следующую информацию о станции:
  - *ID* — идентификатор системной станции OpenSCADA, должен быть изменён после добавления путём выбора из перечня доступных;
  - *Имя* — имя системной станции OpenSCADA;
  - *Жив* — признак наличия связи с резервной станцией;
  - *Уровень* — уровень удалённой станции в схеме резервирования;
  - *Счётчик* — счётчик запросов к резервной станции или времени ожидания восстановления, в случае отсутствия связи;
  - *Запущен* — список доступных контроллеров, с признаком (+) — контроллеры локального исполнения на удалённой станции.
- *Переход к конфигурации перечня удалённых станций* — команда для перехода на страницу конфигурации удалённых OpenSCADA станций, в подсистеме "Транспорты".

- *Контроллеры* — содержит таблицу с перечнем контроллеров, доступных для резервирования, и текущее их состояние:
  - *Контроллер* — полный идентификатор контроллера;
  - *Имя* — имя контроллера;
  - *Запущен* — признак исполнения контроллера локальной станцией;
  - *Резервирование* — режим резервирования контроллера, может быть выбран из перечня: "Выключено" и "Асимметрично";
  - *Предпочтение исполнения* — конфигурация предпочтительного исполнения на указанной станции, может быть изменён; зарезервированные значения указывают на: **<Высокий уровень>** — исполнение на станции с наивысшим уровнем, **<Низкий уровень>** — исполнение на станции с самым низким уровнем, **<Оптимально>** — выбор для исполнения наименее нагруженной станции.
  - *Удалённый* — признак, указывающий на исполнение контроллера удалённой станцией и работу локальной станции в режим синхронизации данных с удалённой.

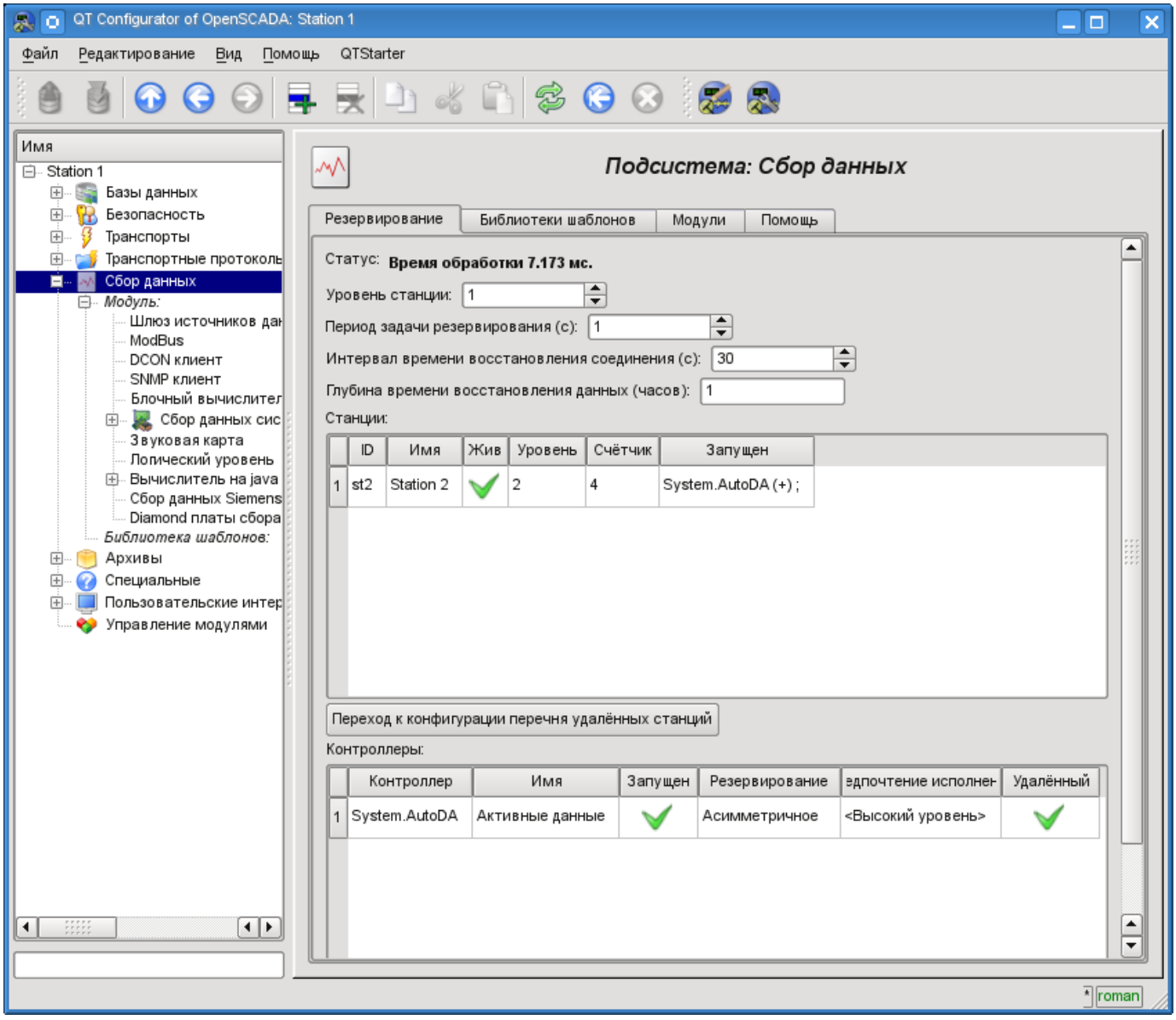

Рис. 4.5b. Вкладка "Резервирование" подсистемы "Сбор данных".

Вкладка "Библиотеки шаблонов" (рис.4.5c) содержит список библиотек шаблонов для параметров этой подсистемы. В контекстном меню списка библиотек шаблонов пользователю предоставляется возможность добавления, удаления и перехода к нужной библиотеке. Вкладка "Модули" (рис.4.1b) содержит список модулей подсистемы "Транспорты" и идентична для всех модульных подсистем. Вкладка "Помощь" содержит краткую помощь для данной страницы.

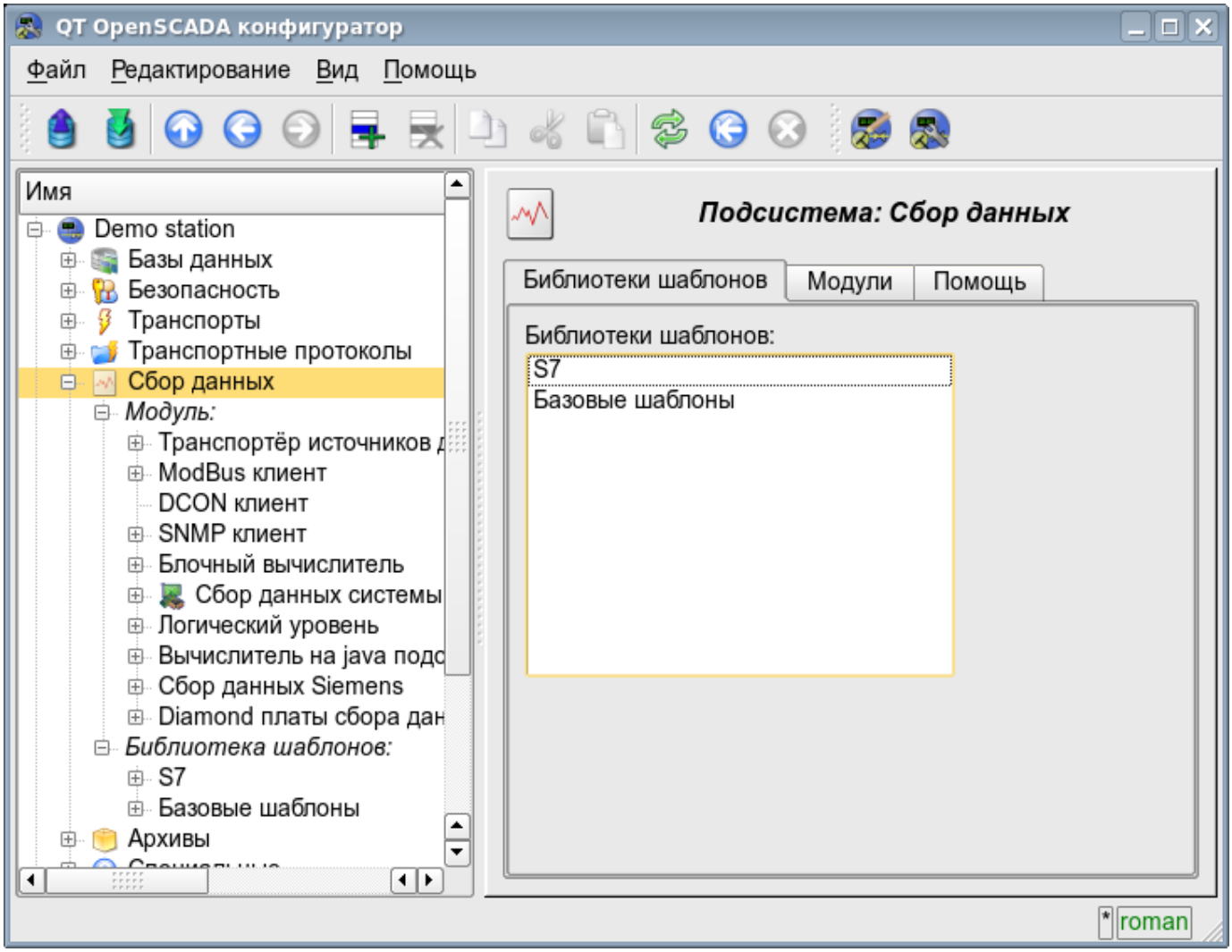

Рис. 4.5c. Вкладка "Библиотеки шаблонов" подсистемы "Сбор данных".

Каждая библиотека шаблонов подсистемы "Сбор данных" предоставляет конфигурационную страницу с вкладками "Библиотека" и "Шаблоны параметров". Вкладка "Библиотека" (рис.4.5d) содержит основные настройки библиотеки в составе:

- Раздел "Состояние" — содержит свойства, характеризующие состояние библиотеки:
  - *Доступен* — состояние библиотеки "Доступен".
  - *БД библиотеки* — адрес БД для хранения данных библиотеки и шаблонов.
- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — информация об идентификаторе библиотеки.
  - *Имя* — указывает имя библиотеки.
  - *Описание* — краткое описание библиотеки и её назначения.

Вкладка "Шаблоны параметров" (рис.4.5e) содержит список шаблонов в библиотеке. В контекстном меню списка пользователю предоставляется возможность добавления, удаления и перехода к нужному шаблону.

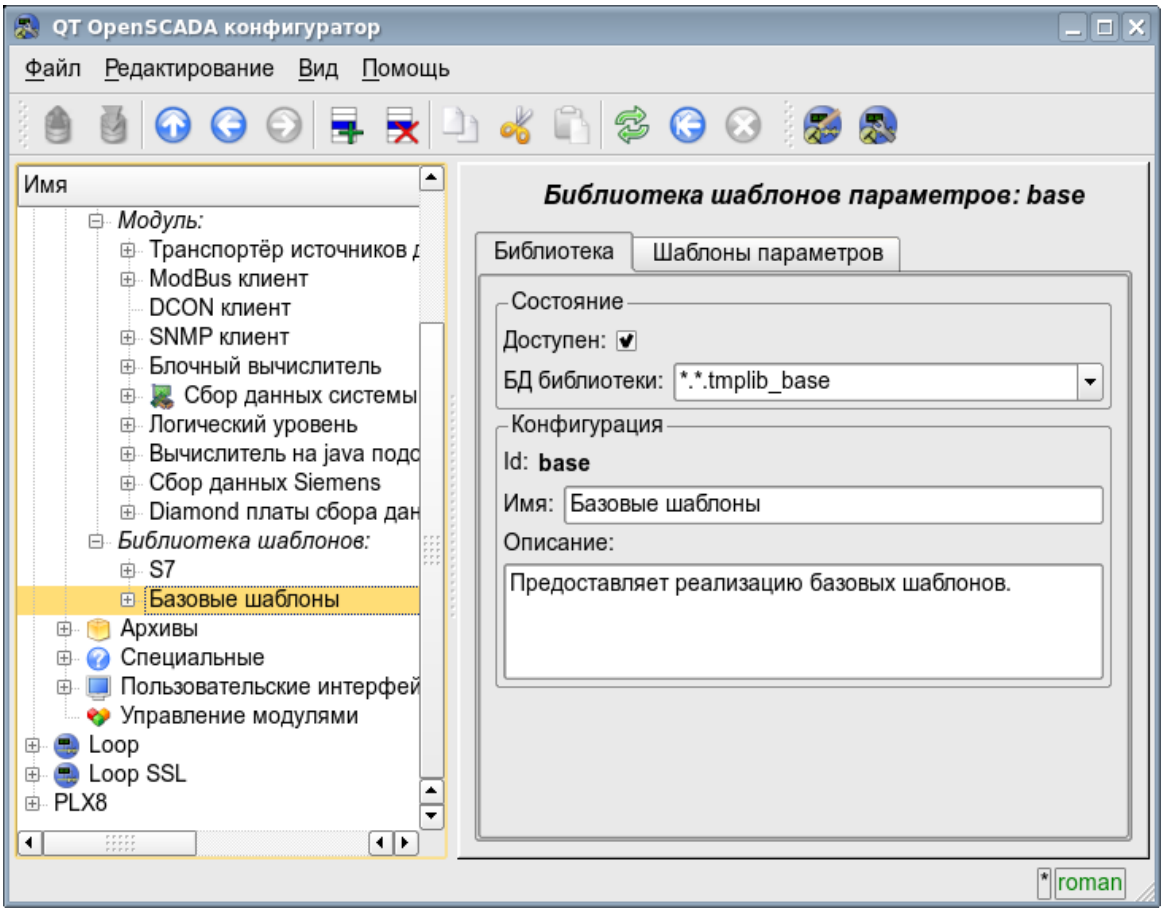

Рис. 4.5d. Основная вкладка конфигурации библиотеки шаблонов подсистемы "Сбор данных".

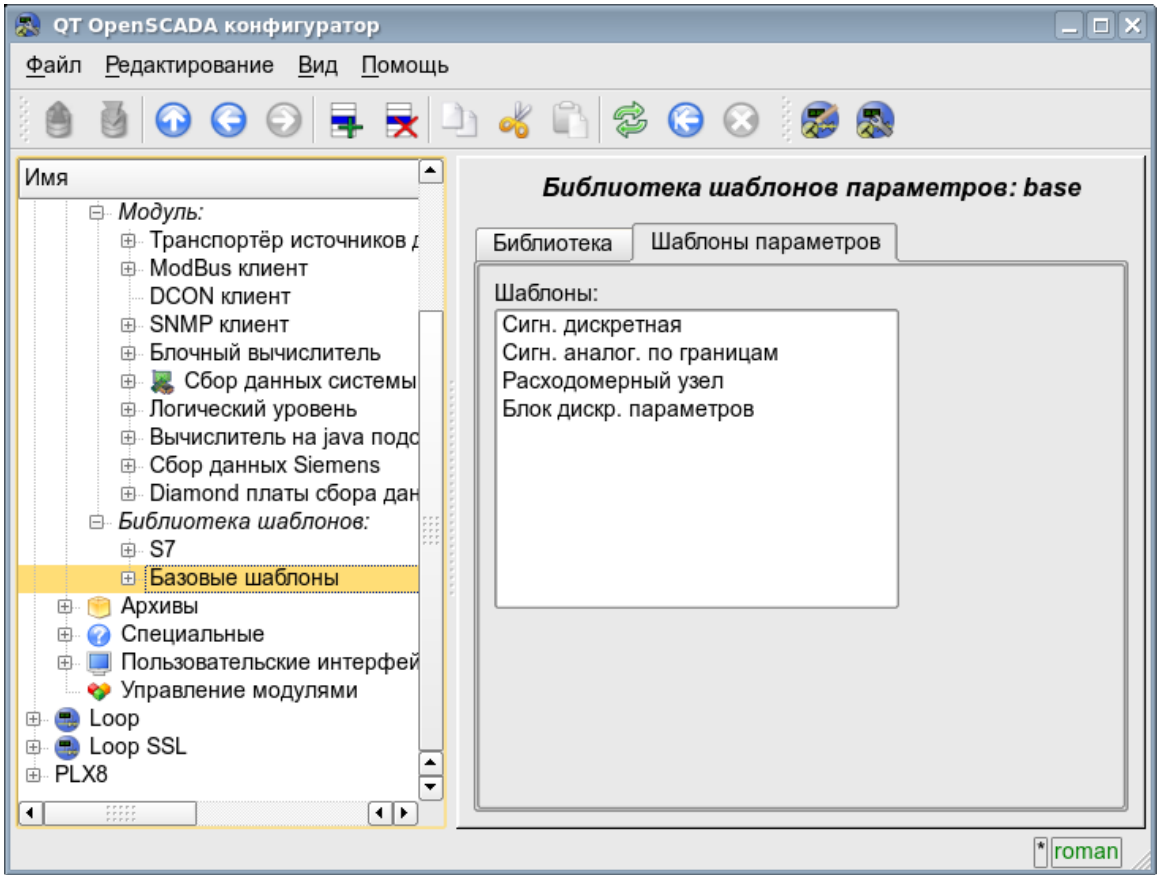

Рис. 4.5e. Вкладка списка шаблонов в библиотеке шаблонов подсистемы "Сбор данных".

Каждый шаблон библиотеки шаблонов предоставляет конфигурационную страницу с вкладками "Шаблон" и "IO". Вкладка "Шаблон" (рис.4.5f) содержит основные настройки шаблона в составе:

- Раздел "Состояние" — содержит свойства, характеризующие состояние шаблона:
  - *Доступен* — состояние шаблона "Доступен".
  - *Использовано* — количество использования шаблона. Позволяет определить факт использования и, как следствие, возможность редактирования шаблона.
- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — информация об идентификаторе шаблона.
  - *Имя* — указывает имя шаблона.
  - *Описание* — краткое описание шаблона и его назначения.

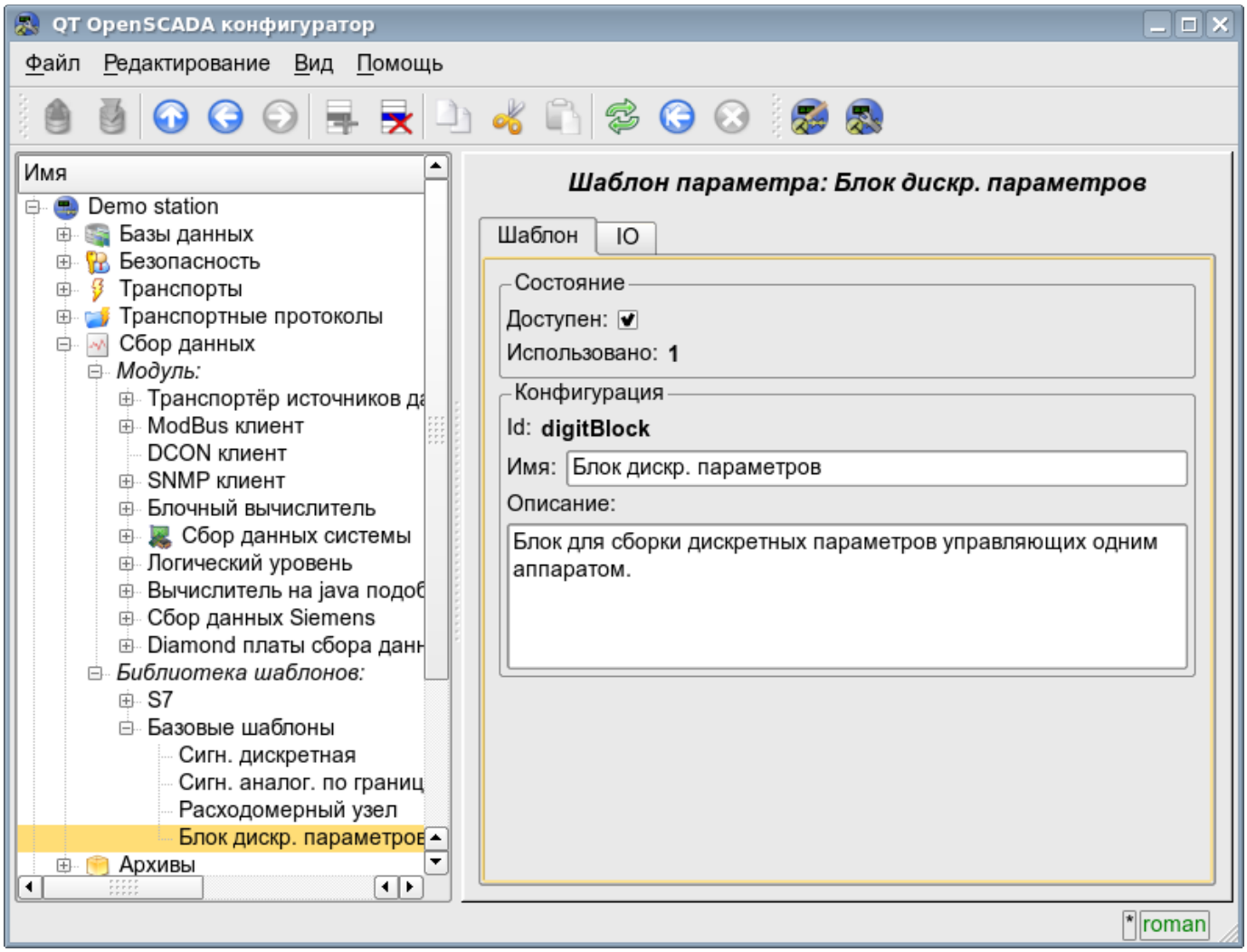

Рис. 4.5f. Главная вкладка конфигурации шаблона параметров подсистемы "Сбор данных".

Вкладка "IO" (рис.4.5g) содержит конфигурацию атрибутов (IO) шаблонов и программу шаблона на одном из языков пользовательского программирования OpenSCADA, например, DAQ.JavaLikeCalc.JavaScript. В таблицу атрибутов шаблона пользователь может, посредством контекстного меню, добавить, вставить, удалить, передвинуть вверх или вниз запись атрибута, а также отредактировать поля атрибута:

- *Id* — идентификатор атрибута.
- *Имя* — имя атрибута.
- *Тип* — выбор типа значения атрибута из списка: "Вещественный", "Целый", "Логический", "Строка", "Объект".
- *Режим* — выбор режима из списка: "Вход", "Выход". Обычно используется для указания направления передачи данных посредством связи. Т.е. для значения "Вход" данные по связи будут только получаться, а для "Выход" также будут передаваться, в случае модификации.
- *Атрибут* — режим атрибута параметра, реализованного на основе шаблона из списка: "Не атрибут", "Только для чтения", "Полный доступ". Для атрибутов шаблона, у которых

это поле установлено, будет создаваться соответствующий атрибут у параметра контроллера этой подсистемы.

- *Конфигурация* — режим конфигурации атрибута во вкладке конфигурации шаблона у параметра контроллера этой подсистемы из списка: "Константа", "Публичная константа", "Связь". В режимах "Публичная константа" и "Связь" во вкладке конфигурации шаблона будут добавлены эти атрибуты для установки константы или указания внешней связи параметра.
- *Значение* — значение атрибута по умолчанию или шаблон ссылки для доступа по ссылке. Формат шаблона ссылки зависит от компонента, который его использует. Обычно для модуля DAQ.LogicLev шаблон ссылки записывается в виде: **{Параметр}|{атрибут}**. Поле **{Параметр}** — указывает имя параметра как контейнера атрибутов. Атрибуты с одинаковым значением **{Параметр}** будут группироваться и позволят назначаться только указанием контейнера атрибутов, а отдельные атрибуты будут связаны с атрибутами контейнера в соответствии с полем **{атрибут}**.

С синтаксисом языка программы шаблона можно ознакомиться в документации модуля, предоставляющего интерпретатор выбранного языка. Например, типичным языком пользовательского программирования OpenSCADA является DAQ.JavaLikeCalc.JavaScript

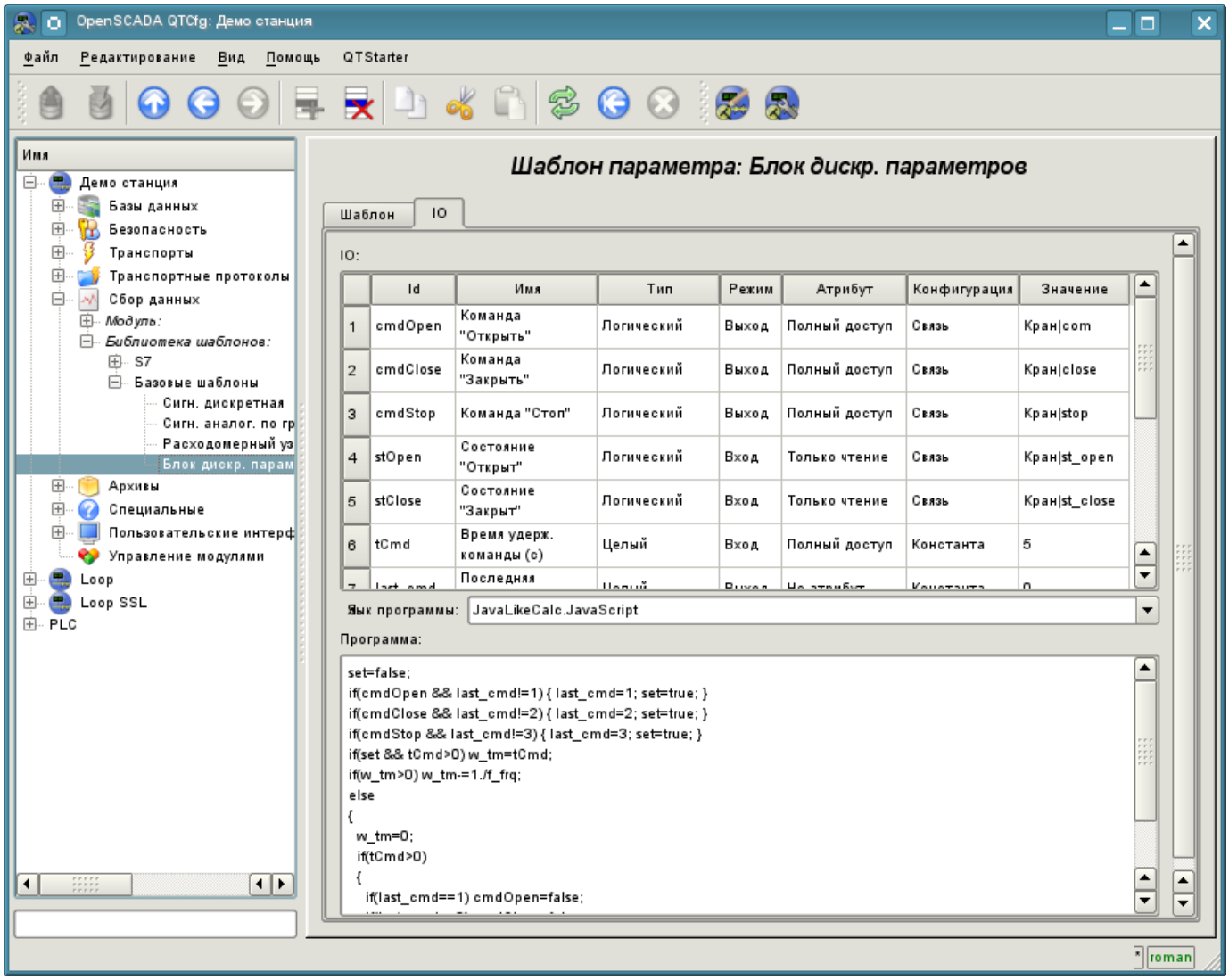

Рис. 4.5g. Вкладка конфигурации атрибутов и программы шаблона подсистемы "Сбор данных".

Каждый модуль подсистемы "Сбор данных" предоставляет конфигурационную страницу с вкладками "Контроллеры" и "Помощь". Вкладка "Контроллеры" (рис.4.5h) содержит список контроллеров, зарегистрированных в модуле. В контекстном меню списка пользователю предоставляется возможность добавления, удаления и перехода к нужному контроллеру. Во вкладке "Помощь" содержится информация о модуле подсистемы "Сбор данных" (рис.4.1d), состав которой идентичен для всех модулей.

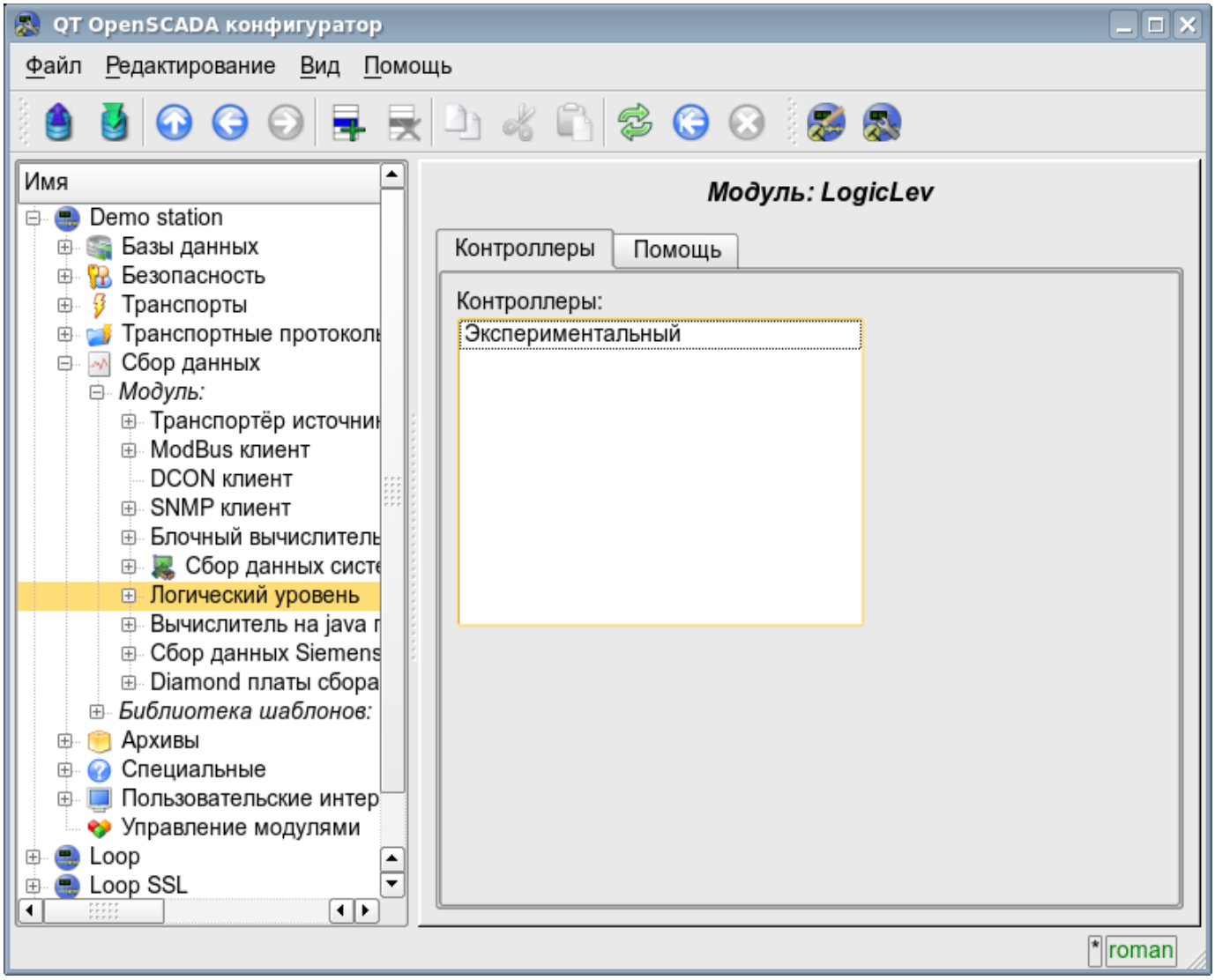

Рис. 4.5h. Вкладка "Контроллеры" модуля подсистемы "Сбор данных".

Каждый контроллер содержит собственную страницу конфигурации с вкладками "Контроллер" и "Параметры".

Вкладка "Контроллер" (рис.4.5i) содержит основные настройки. Состав этих настроек может несколько отличаться от одного модуля этой подсистемы к другому, о чём можно узнать в собственной документации модулей. В качестве примера рассмотрим настройки контроллера у модуля контроллера логического DAQ.LogicLev:

- Раздел "Состояние" — содержит свойства, характеризующие состояние контроллера:
  - *Статус* — указывает статус контроллера. В нашем случае контроллер исполняется и время вычисления составляет 656 микросекунд.
  - *Включен* — состояние контроллера "Включен". Включенный контроллер предоставляет возможность создания параметров и их конфигурации.
  - *Запущен* — состояние контроллера "Запущен". Исполняющийся контроллер выполняет физический сбор данных и/или включает механизмы доступа к этим данным.
  - *БД контроллера* — адрес БД для хранения данных контроллера и его параметров.
- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:

- *ID* — информация об идентификаторе контроллера.
- *Имя* — указывает имя контроллера.
- *Описание* — краткое описание контроллера и его назначения.
- *Включать* — указывает на состояние "Включать," в которое переводить контроллер при загрузке.
- *Запускать* — указывает на состояние "Запущен", в которое переводить контроллер при загрузке.
- *Таблицы параметров* — имена таблиц, в которых сохранять параметры, разных типов (имеются в виду объекты параметров сбора данных).
- *Планирование вычислений* — определяет периодический или по расписанию характер вычисления. В нашем примере это секундное вычисления шаблона.
- *Уровень приоритета задачи получения данных* — устанавливает приоритет задачи сбора данных этого контроллера. Используется при планировании задач операционной системой. В случае исполнения станции от имени суперпользователя "root" это поле включает планирование задачи контроллера в режиме реального времени и с указанным приоритетом.

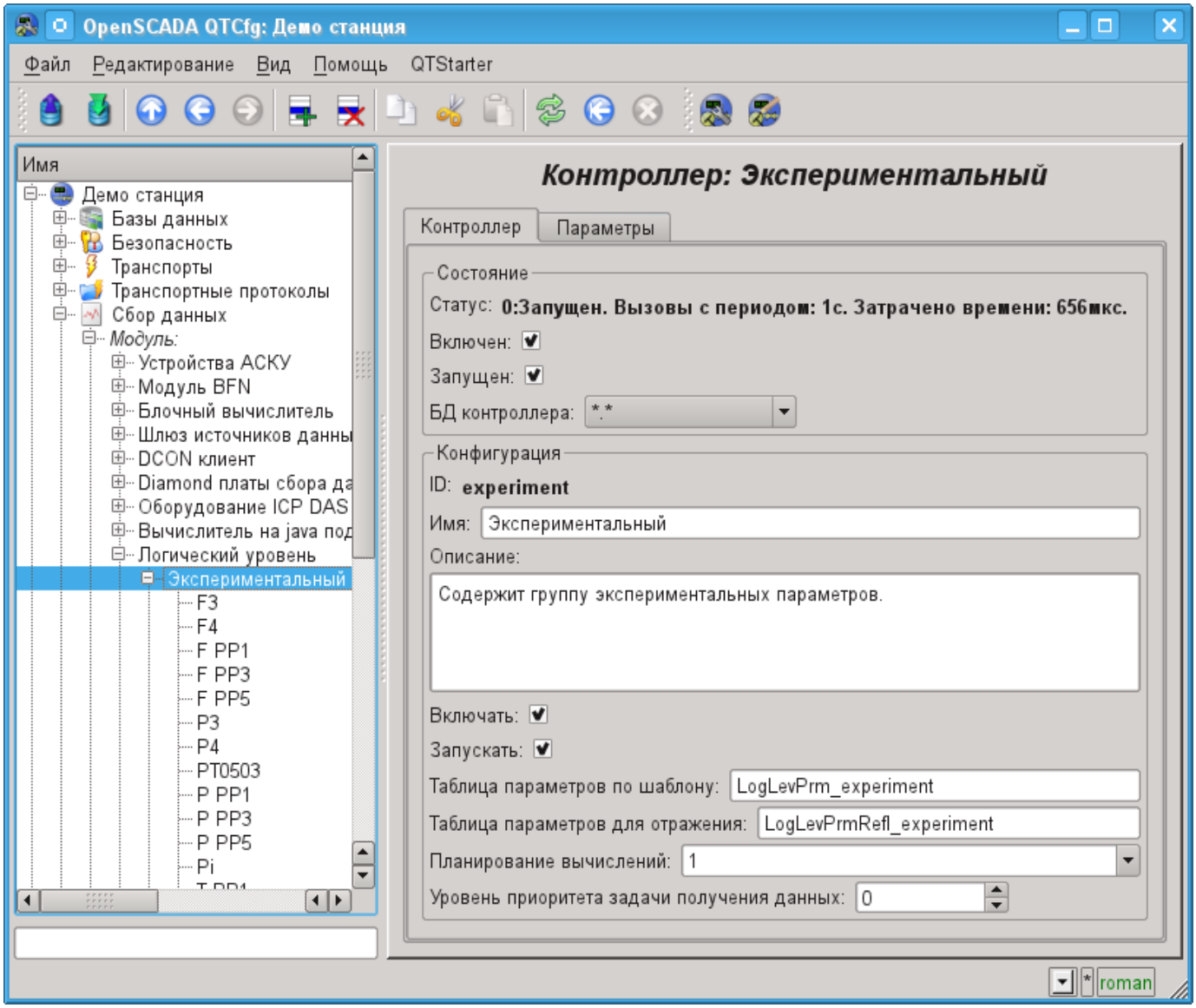

Рис. 4.5i. Главная вкладка конфигурации контроллера подсистемы "Сбор данных".

Вкладка "Параметры" (рис.4.5j) содержит список параметров в контроллере, выбор типа параметров, создаваемых по умолчанию, а также информацию об общем количестве и количестве включенных параметров. В контекстном меню, списка параметров, пользователю предоставляется возможность добавления, удаления и перехода к нужному параметру.

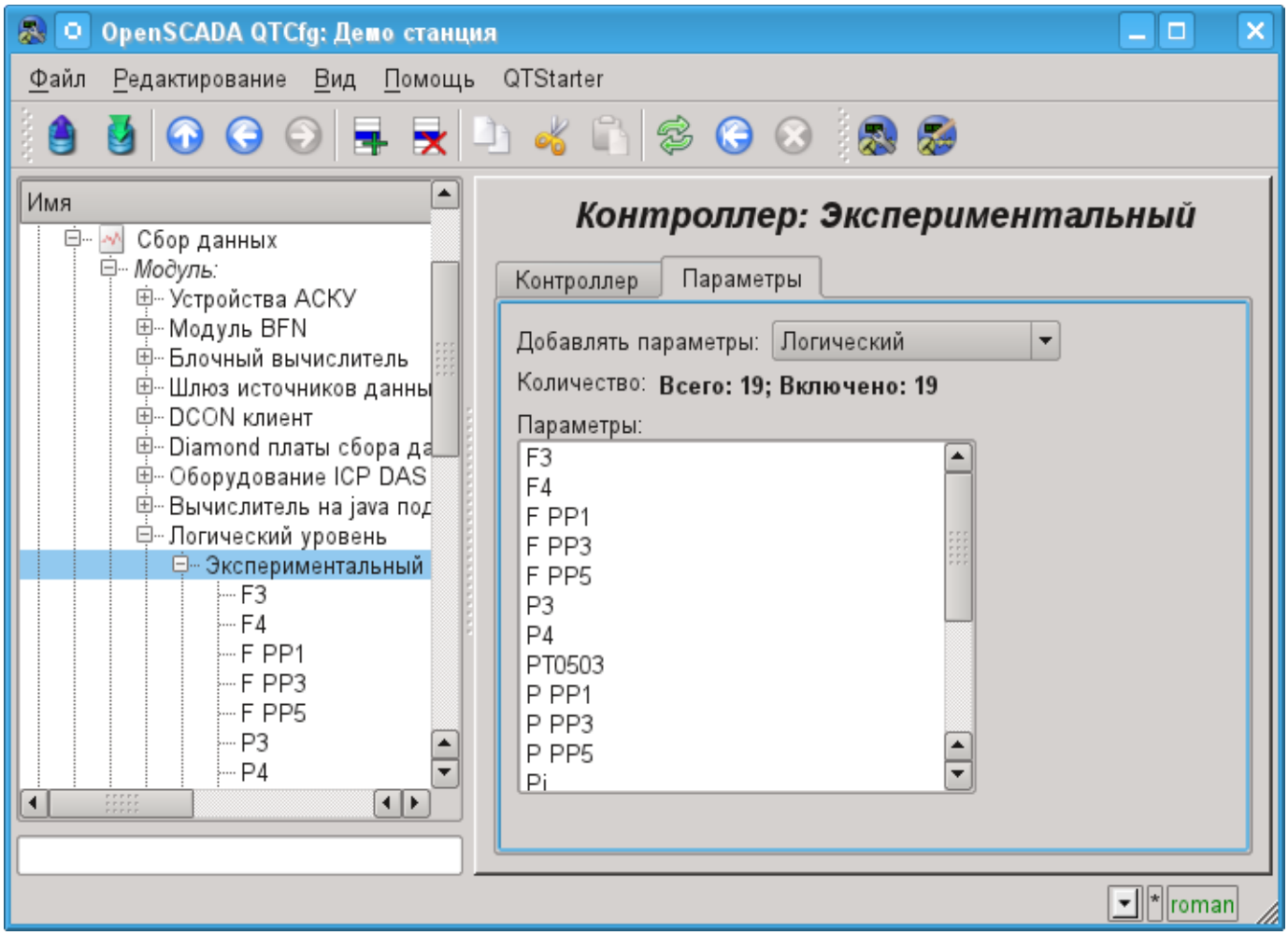

Рис. 4.5j. Вкладка "Параметры" страницы конфигурации контроллера подсистемы "Сбор данных".

Параметры контроллеров подсистемы "Сбор данных" предоставляют конфигурационную страницу с вкладками "Параметр", "Атрибуты", "Архивация" и "Конфигурация шаблона". Вкладка "Конфигурация шаблона" не является стандартной, а присутствует только в параметрах модулей подсистемы "Сбор данных", которые реализуют механизмы работы по шаблону в контексте источника данных, ими обслуживаемого, для логического типа. В данный обзор эта вкладка включена для обеспечения логической завершённости обзора конфигурации шаблонов параметров подсистемы "Сбор данных" как финальный этап — использования.

Вкладка "Параметр" (рис.4.5k) содержит основные настройки в составе:

- Раздел "Состояние" — содержит свойства, характеризующие состояние параметра:
  - *Тип* — указывает тип параметра. Тип выключенного параметра может быть изменён, если доступно несколько типов.
  - *Включен* — состояние параметра "Включен". Включенный параметр используется контроллером для сбора данных.
- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — содержит информацию об идентификаторе параметра.
  - *Имя* — указывает имя параметра.
  - *Описание* — краткое описание параметра и его назначения.
  - *Включать* — указывает на состояние "Включать", в которое переводить параметр при загрузке.
  - *Шаблон параметра* — адрес ранее нами рассмотренного шаблона.

Вкладка "Атрибуты" (рис.4.5l) содержит атрибуты параметра и их значения в соответствии с конфигурацией используемого шаблона и вычисления его программы.

Вкладка "Архивация" (рис.4.5m) содержит таблицу с атрибутами параметра в колонках, и архиваторами в строках. Пользователь имеет возможность установить архивацию нужного атрибута требуемым архиватором просто изменив ячейку на пересечении.

Вкладка "Конфигурация шаблона" (рис.4.5n) содержит конфигурационные поля в соответствии с шаблоном. В примере это групповая связь на внешний параметр. Эту связь можно установить, просто указав путь к параметру, если флаг "Показывать только атрибуты" не установлен, или же установить адреса атрибутов по отдельности, если флаг установлен. Знак "(+)", в конце адреса, сигнализирует об успешной линковке и присутствии целевого объекта.

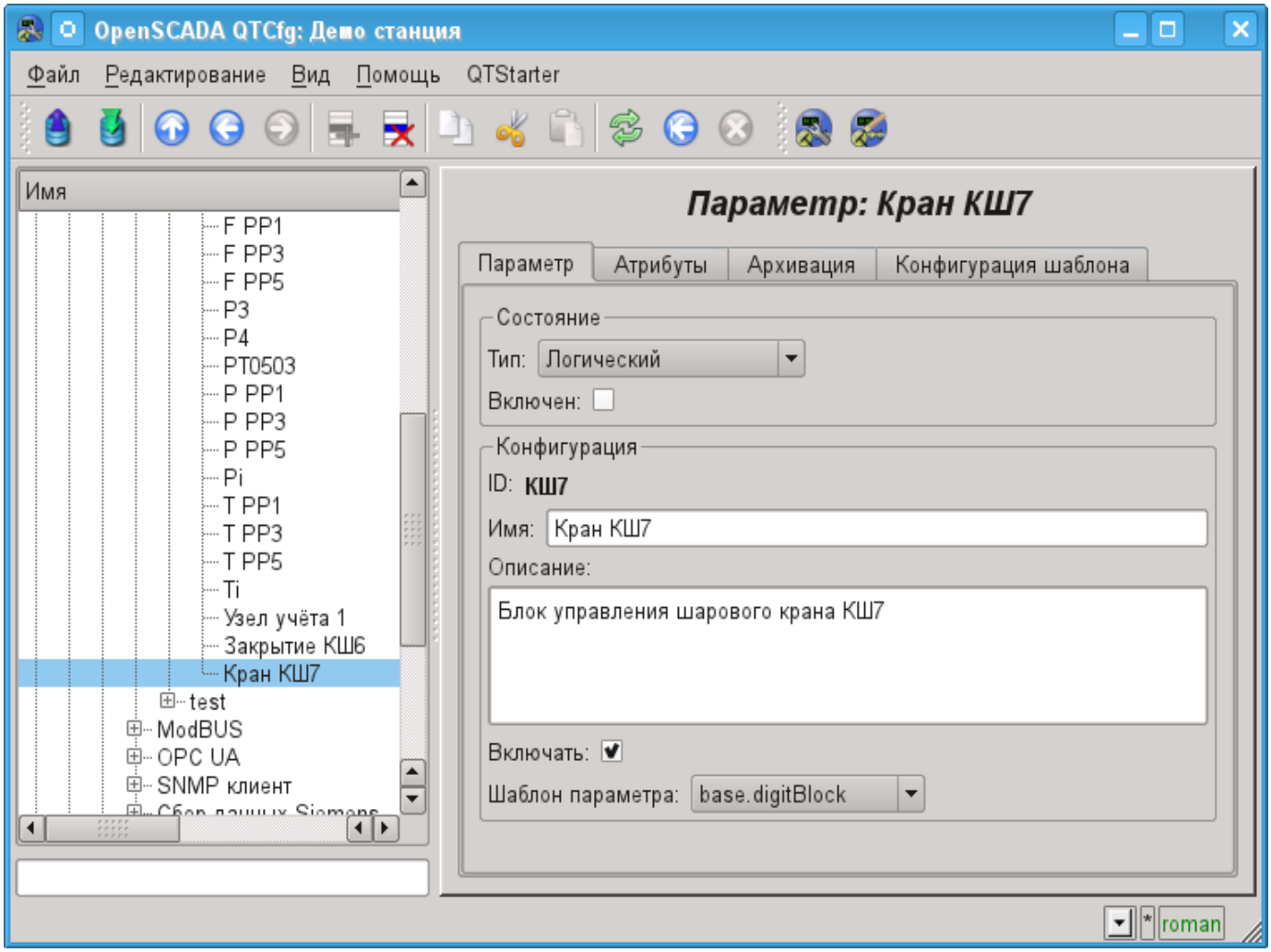

Рис. 4.5k. Главная вкладка конфигурации параметра контроллера подсистемы "Сбор данных".

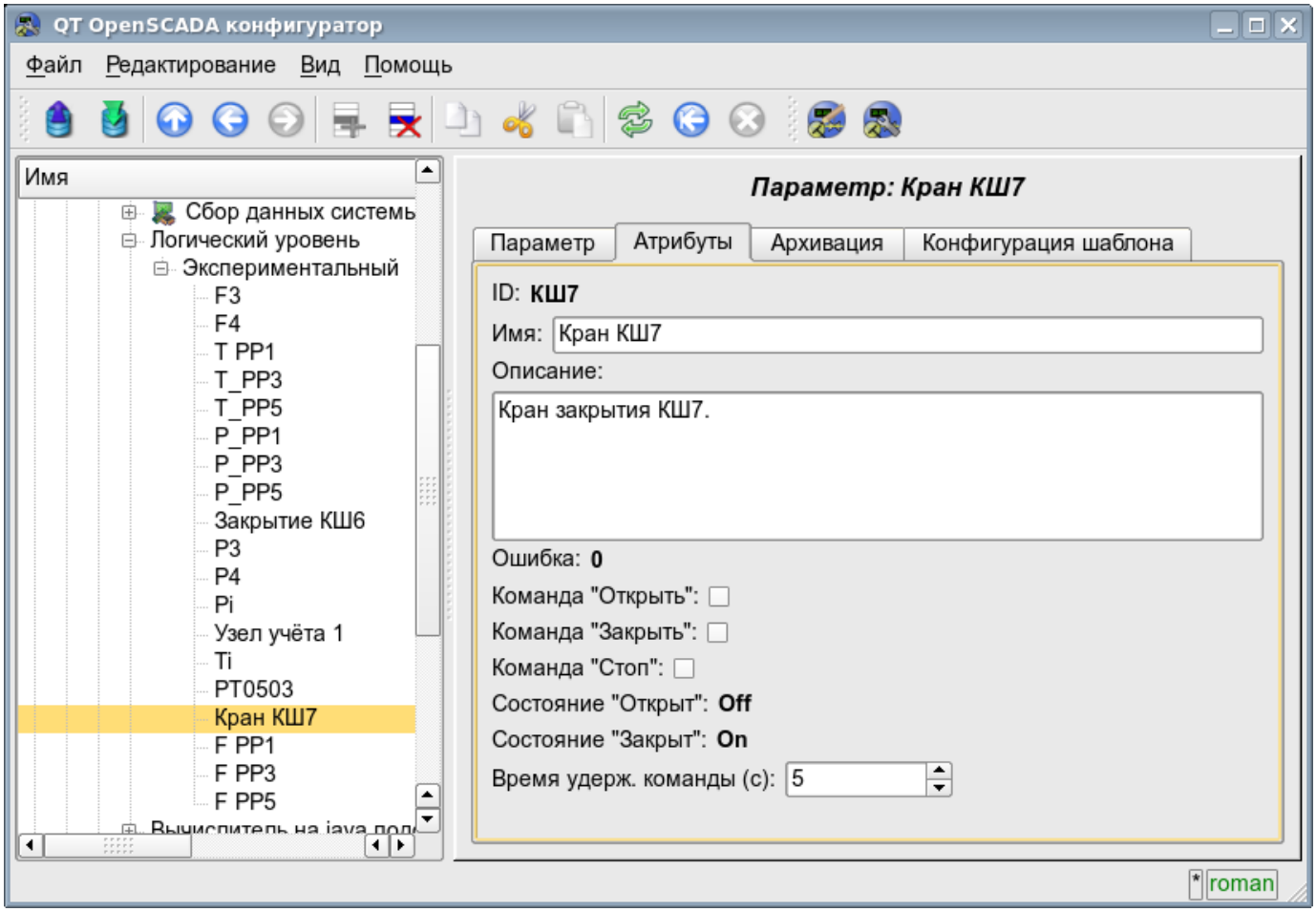

Рис. 4.5l. Вкладка "Атрибуты" параметра контроллера подсистемы "Сбор данных".

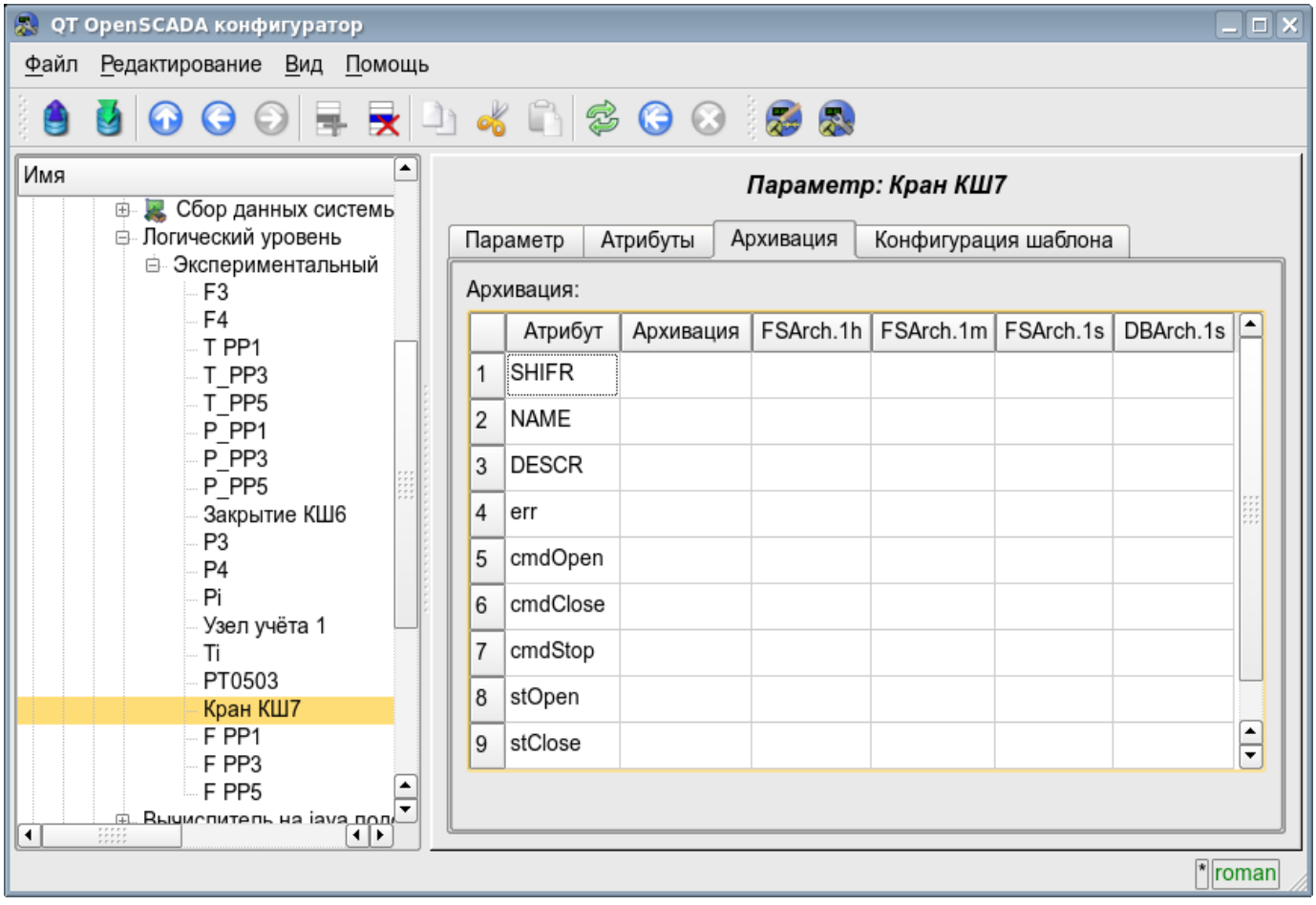

Рис. 4.5m. Вкладка "Архивация" параметра контроллера подсистемы "Сбор данных".

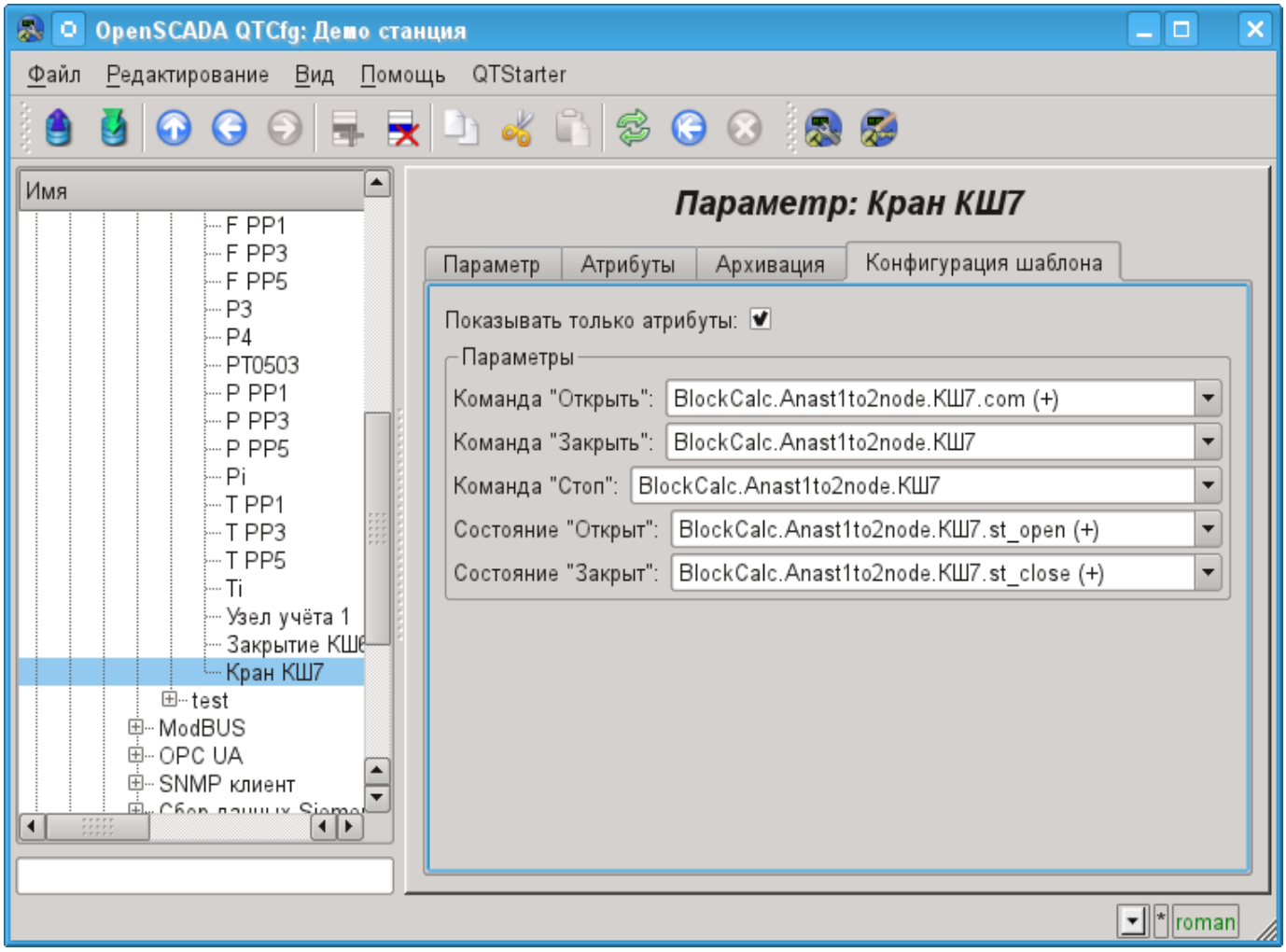

Рис. 4.5п. Вкладка "Конфигурация шаблона" параметра контроллера подсистемы "Сбор данных".

4.6. Подсистема "Архивы"

Подсистема является модульной и содержит иерархию объектов, изображённую на рис.4.6a. Для конфигурации подсистемы предусмотрена корневая страница подсистемы "Архивы", содержащая вкладки "Архив сообщений", "Архивы значений", "Модули" и "Помощь".

Для получения доступа на модификацию объектов этой подсистемы необходимы права пользователя в группе "Archive" или права привилегированного пользователя.

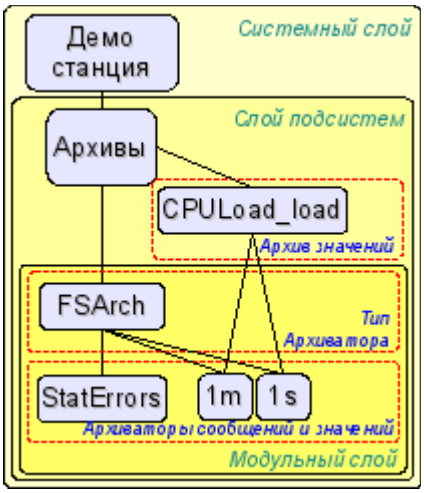

Рис. 4.6a. Иерархическая структура подсистемы "Архивы".

Вкладка "Архив сообщений" (рис.4.6b) содержит конфигурацию архива сообщений и форму запроса сообщений из архива.

Конфигурация архива сообщений представлена полями:

- *Размер буфера сообщений* — указывает на размерность области оперативной памяти зарезервированное на промежуточный буфер сообщений. Сообщения из буфера запрашиваются для просмотра и архивируются архиваторами сообщений.
- *Период архивирования сообщений (с)* — периодичность, с которой архиваторы выбирают сообщения из буфера для их архивирования.

Форма запроса сообщений содержит конфигурационные поля запроса и таблицу результата. Конфигурационные поля запроса:

- *Время* — указывает время запроса.
- *Размер (сек)* — указывает размер или глубину запроса в секундах.
- *Шаблон категории* — указывает категорию запрошенных сообщений. В категории можно указывать элементы выборки по шаблону, а именно символы "*" — для любой подстроки и "?" — для любого символа, а также регулярное выражение заключённое между символами '/' (/mod_(System|LogicLev)/).
- *Уровень* — указывает на минимальный уровень сообщений, т.е. запрос будет обработан для сообщений с уровнем, более или равным указанному.
- *Архиватор* — указывает архиватор сообщений, для которого обрабатывать запрос. Если значение отсутствует, то запрос будет обработан для буфера и всех архиваторов. Если указано <buffer>, то запрос будет обработан только для буфера сообщений.

Таблица результата содержит строки сообщений с колонками:

- *Время* — время сообщения.
- *Категория* — категория сообщения.
- *Уровень* — уровень сообщения.
- *Сообщение* — текст сообщения.

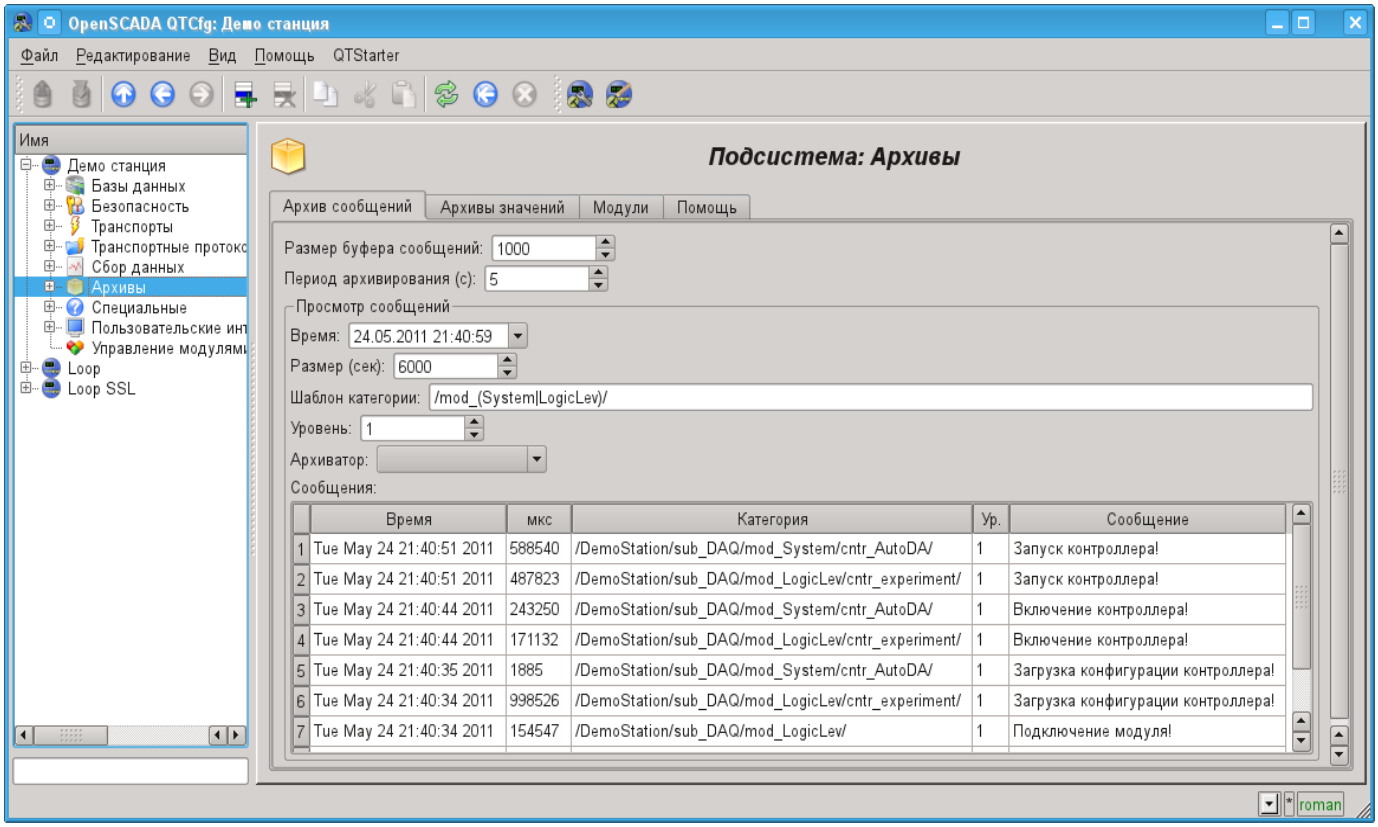

Рис. 4.6b. Вкладка "Архив сообщений" подсистемы "Архивы".

Вкладка "Архивы значений" (рис.4.6c) содержит общую конфигурацию архивирования значений и список архивов значений. В контекстном меню списка значений пользователю предоставляется возможность добавления, удаления и перехода к нужному архиву. Общая конфигурация архивирования представлена полями:

- *Период получения данных (мс)* — указывает периодичность задачи активного архивирования. Фактически, максимальная детализация или минимальный период, активных архивов определяется этим значением.
- *Уровень приоритета задачи получения данных* — устанавливает приоритетность задачи активного архивирования. Используется при планировании задач операционной системой. В случае исполнения станции от имени суперпользователя "root" это поле включает планирование задачи архивирования в режиме реального времени и с указанным приоритетом.

Вкладка "Модули" (рис.4.1b) содержит список модулей подсистемы "Архивы" и идентична для всех модульных подсистем. Вкладка "Помощь" содержит краткую помощь для данной страницы.

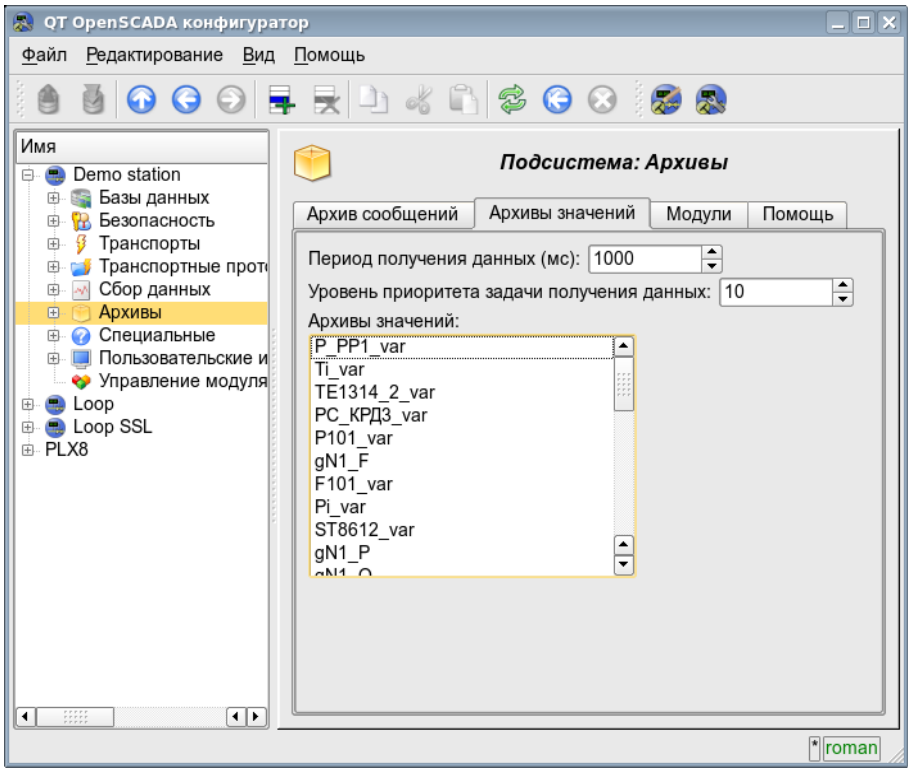

Рис. 4.6c. Вкладка "Архивы значений" подсистемы "Архивы".

Архив значений подсистемы "Архивы" предоставляет конфигурационную страницу с вкладками "Архив", "Архиваторы" и "Значения".

Вкладка "Архив" (рис.4.6d) содержит основные настройки архива в составе:

- Раздел "Состояние" — содержит свойства, характеризующие состояние архива:
  - *Выполняется* — состояние параметра "Выполняется". Исполняющийся архив собирает данные в буфер и обслуживается архиваторами.
  - *БД архива* — адрес БД для хранения данных архива.
- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — информация об идентификаторе архива.
  - *Имя* — указывает имя архива.
  - *Описание* — краткое описание архива и его назначения.
  - *Запускать* — указывает на состояние "Выполняется", в которое переводить архив при загрузке.
  - *Тип значений* — указывает на тип значений, хранящихся в архиве, из списка: "Логический", "Целый", "Вещественный" и "Строка".
  - *Источник* — указывает на тип и адрес источника. Тип источника указывается из списка: "Пассивный", "Пассивный атрибут параметра" или "Активный атрибут параметра". Пассивный архив не имеет ассоциированного источника значений;

данные в такой архив источник передаёт самостоятельно, например, из пользовательских вычислительных процедур посредством внутреннего языка программирования. Типы с атрибутом параметра в поле адреса указывают на параметр подсистемы "Сбор данных" как источник. Пассивный атрибут параметра направляет данные в архив самостоятельно с собственным периодом сбора данных. Активный атрибут параметра опрашивается задачей архивирования этой подсистемы. Фактически все источники реальных данных работают в пассивном и активном режиме архивирования поскольку полученные данные сразу помещают в атрибут параметра, иногда по метке времени. А вот вычислители (DAQ.JavaLikeCalc, DAQ.LogicLev, DAQ.BlockCalc) могут работать только в активном режиме архивирования, поскольку данные в атрибуте параметра обновляются только при их непосредственном запросе и берутся из контекста вычисления. В случае с источниками реальных данных, разница между активным и пассивным режимом архивирования определяется тем, что в пассивном режиме источник может помещать данные в архив по метке времени, а в активном режиме метка времени всегда устанавливается в текущее системное время.

- *Период буфера (сек)* — указывает на периодичность значений в буфере архива.
- *Размер буфера (единиц)* — указывает размерность или глубину буфера архива. Размерность обычно устанавливается в пересчёте на 60 сек периодичности задачи архивирования с запасом.
- *Жесткая сетка времени буфера* — указывает на режим буфера. Режим жёсткой сетки подразумевает резервирование памяти под каждое значение, но без метки времени. Такой режим исключает возможность упаковки смежно-одинаковых значений, но экономит на хранении метки времени. Иначе буфер работает в режиме хранения значения и метки времени и поддерживает упаковку смежно-одинаковых значений.
- *Высокое разрешение времени буфера* — указывает на возможность хранения значений с периодичностью до 1 микросекунды, иначе значения могут храниться с периодичностью до 1 секунды.

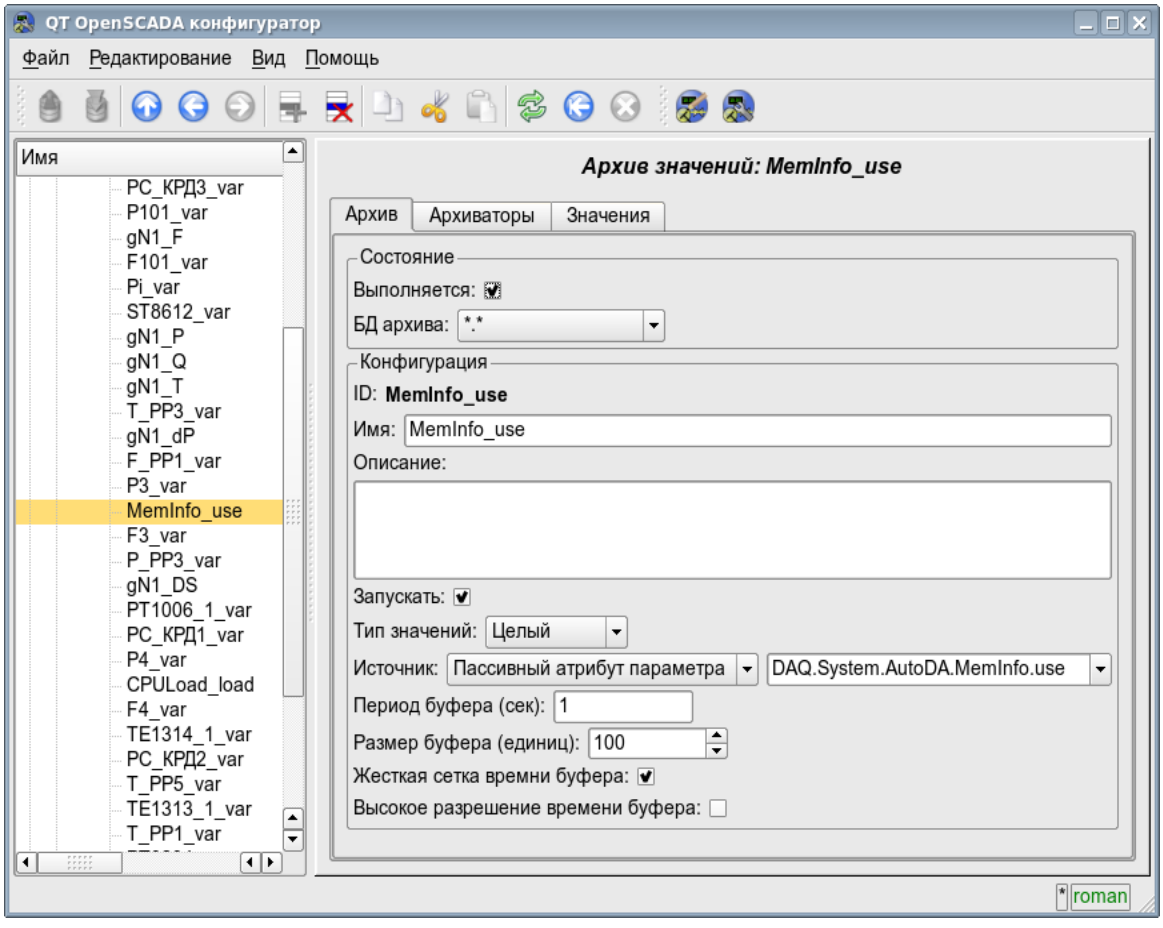

Рис. 4.6d. Основная вкладка конфигурации архива значений подсистемы "Архивы".

Вкладка "Архиваторы" (рис.4.6е) содержит таблицу с конфигурацией процесса обработки данного архива доступными архиваторами. В строках расположены доступные архиваторы, а в колонках параметры:

- *Архиватор* — информация об адресе архиватора.
- *Запущен* — информация о состоянии "Запущен" архиватора.
- *Обработка* — признак обработки данного архива архиватором. Поле доступно для модификации пользователем.
- *Период (с)* — информация о периодичности архиватора.
- *Начало* — дата начала данных архива в архиваторе.
- *Конец* — дата конца данных архива в архиваторе.

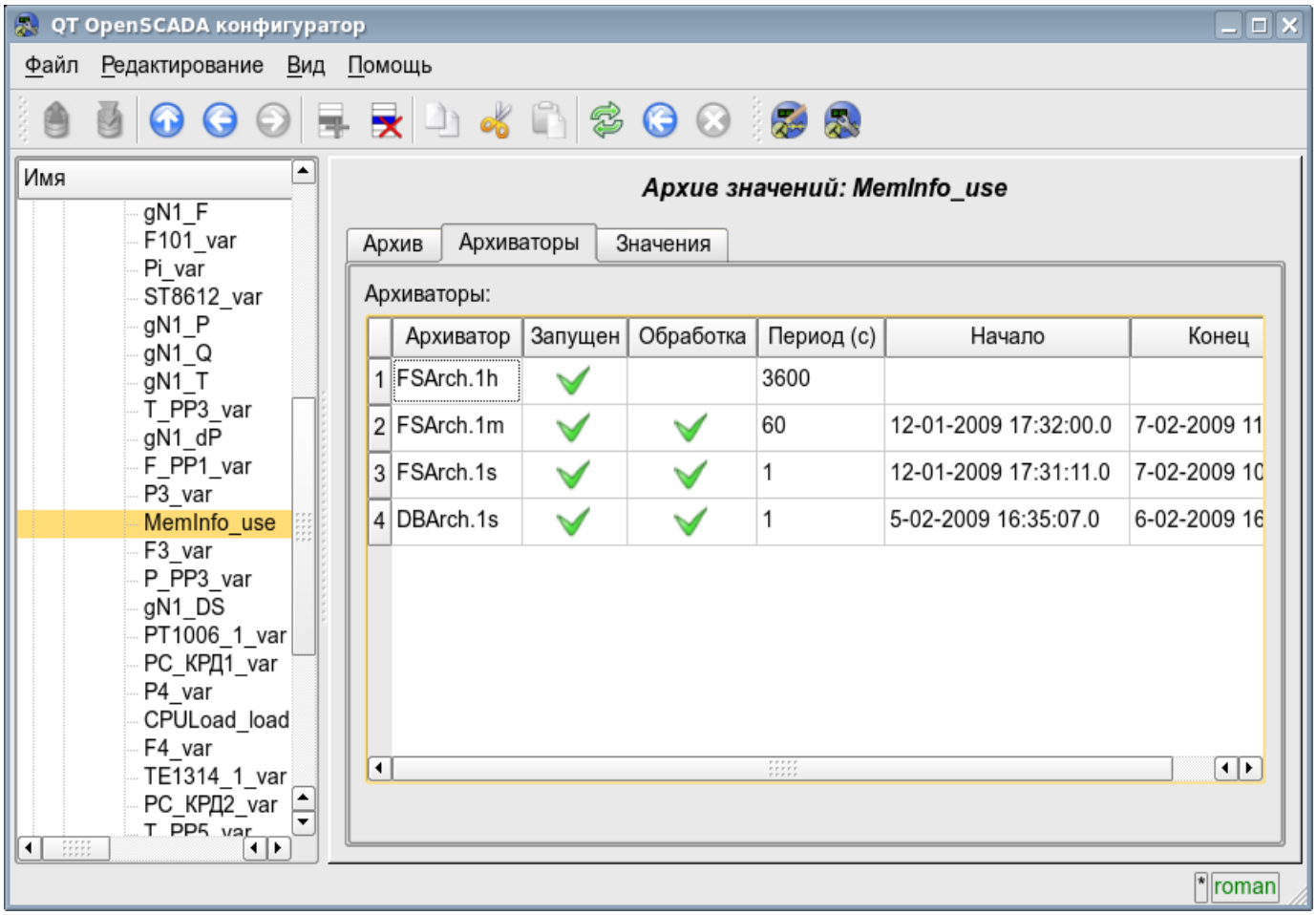

Рис. 4.6е. Вкладка "Архиваторы" архива значений подсистемы "Архивы".

Вкладка "Значения" (рис.4.6f) содержит запрос значений в архиве и результат в виде таблицы значений или изображения тренда. Запрос значений содержит поля:

- *Время* — указывает время запроса. Содержит два поля: поле дата+время и микросекунды.
- *Размер* — указывает размер или глубину запроса в секундах.
- *Архиватор* — указывает архиватор значений для которого обрабатывать запрос. Если значение отсутствует, то запрос будет обработан для буфера и всех архиваторов. Если указано <buffer>, то запрос будет обработан только для буфера архива.
- *Показать график* — указывает на необходимость представления данных архива в виде графика (тренда), иначе результат представляется в таблице, содержащей только время и значение. В случае установки этого поля формируется и отображается график, кроме того появляются дополнительные конфигурационные поля настройки изображения:
  - *Размер изображения* — указывает ширину и высоту формируемого изображения в пикселах.
  - *Шкала значения* — указывает нижнюю и верхнюю границу шкалы значения. Если оба значения установлены в 0 или равны, то шкала будет определяться автоматически в зависимости от значений.

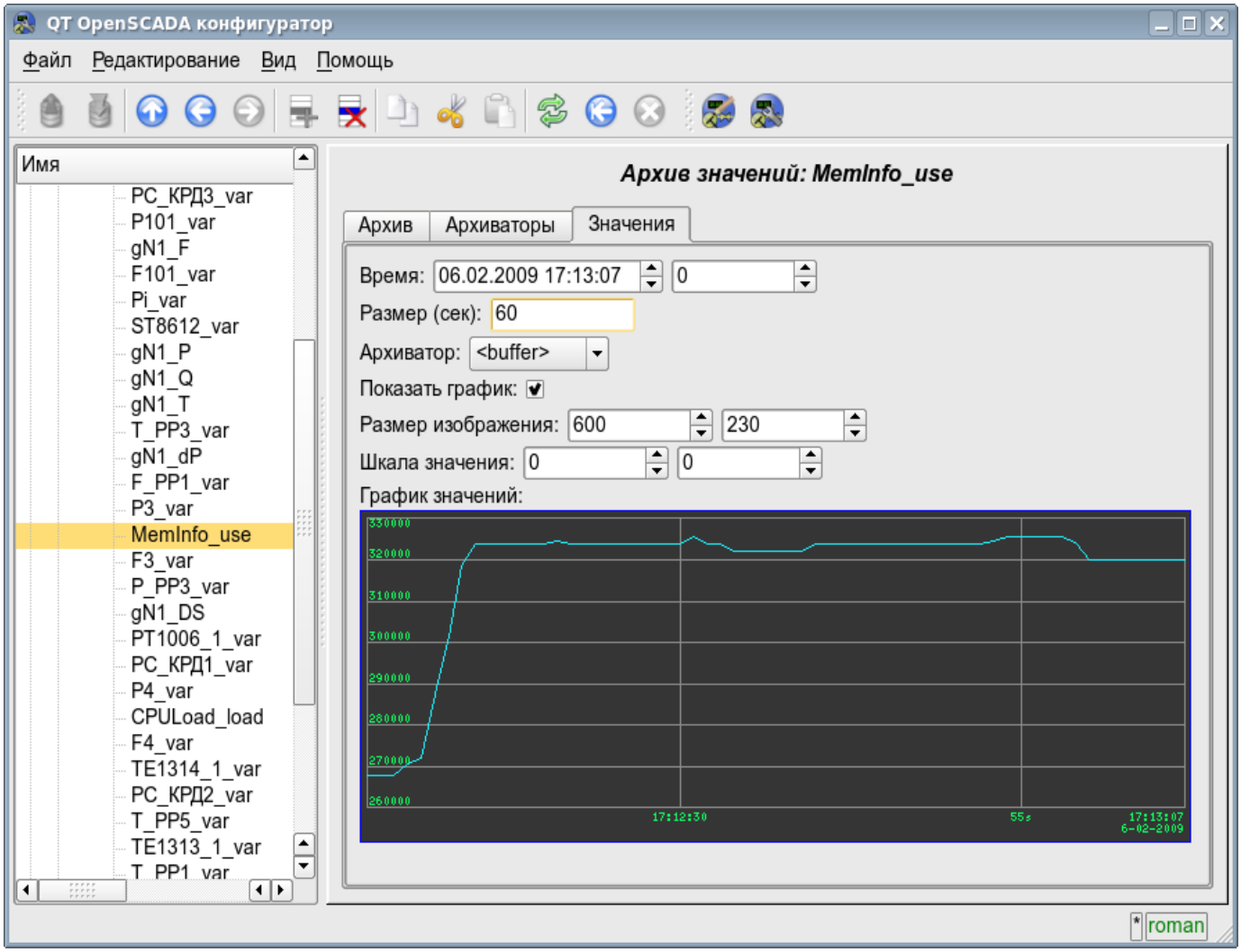

Рис. 4.6f. Вкладка "Значения" архива значений подсистемы "Архивы".

Каждый модуль подсистемы "Архивы" предоставляет конфигурационную страницу с вкладками "Архиваторы" и "Помощь". Вкладка "Архиваторы" (рис.4.6g) содержит списки архиваторов сообщений и значений, зарегистрированных в модуле. В контекстном меню списков пользователю предоставляется возможность добавления, удаления и перехода к нужному контроллеру. Во вкладке "Помощь" содержится информация о модуле подсистемы "Архивы" (рис.4.1d), состав которой идентичен для всех модулей.

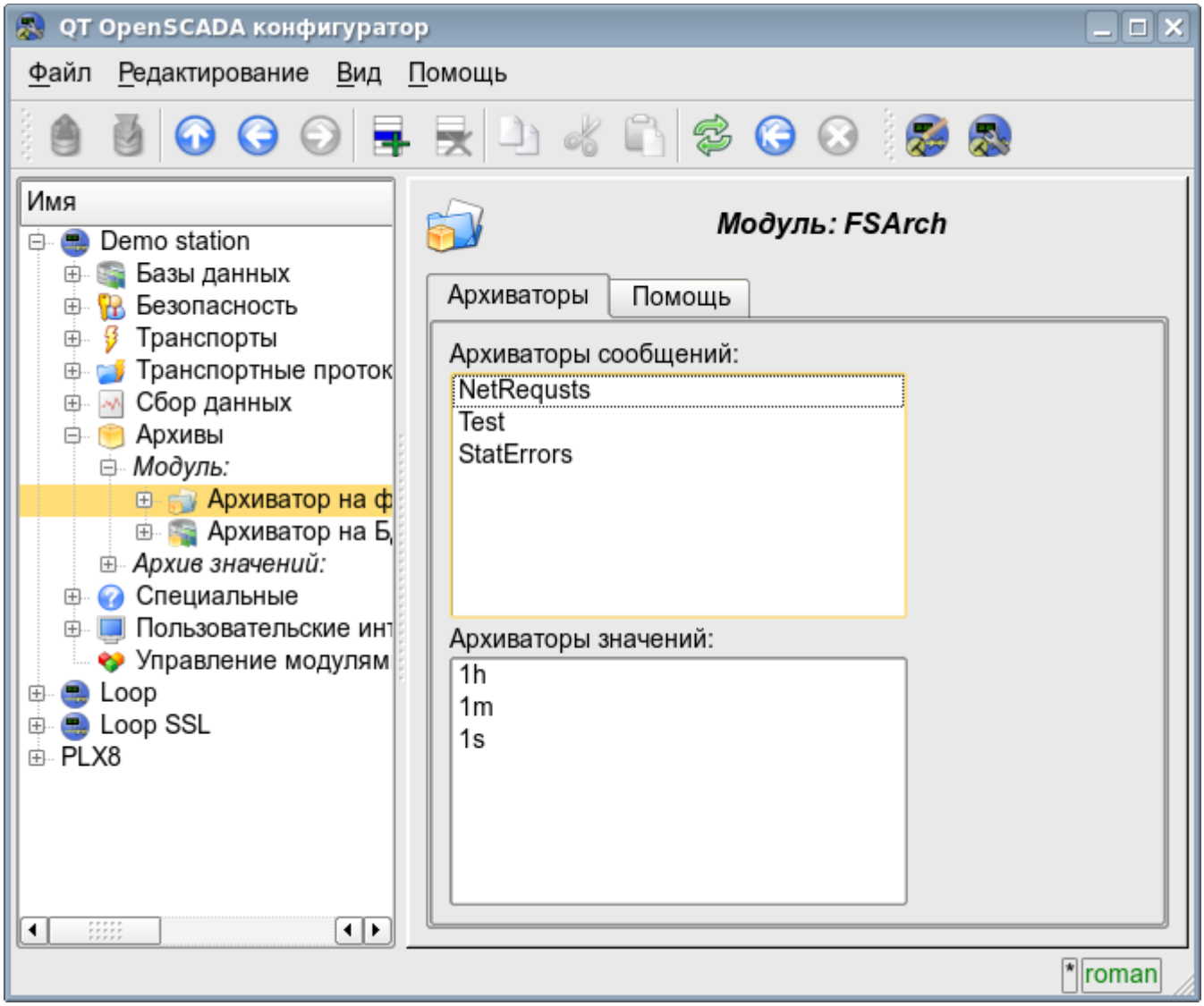

Рис. 4.6g. Вкладка "Архиваторы" модуля подсистемы "Архивы".

Архиваторы сообщений содержат собственную страницу конфигурации с вкладками "Архиватор" и "Сообщения".

Вкладка "Архиватор" (рис.4.6h) содержит основные настройки. Состав этих настроек может несколько отличаться от одного модуля этой подсистемы к другому о чём можно узнать в собственной документации модулей. В качестве примера рассмотрим настройки архиватора сообщений у модуля архива на файловую систему Arch.FSArch Настройки:

- Раздел "Состояние" — содержит свойства характеризующие состояние архиватора:
  - *Выполняется* — состояние архиватора "Выполняется". Исполняющийся архиватор обрабатывает буфер архива сообщений и помещает его данные в своё хранилище, а также обслуживает запросы на доступ к данным в хранилище.
  - *БД архиватора* — адрес БД для хранения данных архиватора.
  - *Конец* — дата+время последних данных в хранилище архиватора.
  - *Начало* — дата+время первых данных в хранилище архиватора.
  - *Размер файлов архиватора (кБ)* — информация о суммарном размере файлов архиватора с данными.

- *Время архивирования (мс)* — время, затраченное на архивирование данных архива сообщений.

- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — информация об идентификаторе архиватора.
  - *Имя* — указывает имя архиватора.
  - *Описание* — краткое описание архиватора и его назначения.
  - *Адрес* — адрес хранилища в специфичном для типа архиватора (модуля) формате. Описание формата записи адреса архиватора, как правило, доступно во всплывающей подсказке этого поля. В примере это относительный путь к директории хранилища.
  - *Уровень сообщений* — указывает на уровень сообщений архиватора. Сообщения с уровнем более или равным указанному обрабатываются архиватором.
  - *Категории сообщений* — список категорий сообщений, разделённый символом ';'. Сообщения, попавшие под шаблоны или регулярные выражения категорий, будут обрабатываться архиватором. В категории можно указывать элементы выборки по шаблону, а именно символы '*' — для любой подстроки и '?' — для любого символа, а также регулярное выражение заключённое между символами '/' (/mod_(System|LogicLev)/).
  - *Запускать* — указывает на состояние "Выполняется", в которое переводить архиватор при загрузке.
- Раздел "Дополнительные опции" — специализированный для модуля раздел, о содержимом которого, можно ознакомиться в документации на модуль.

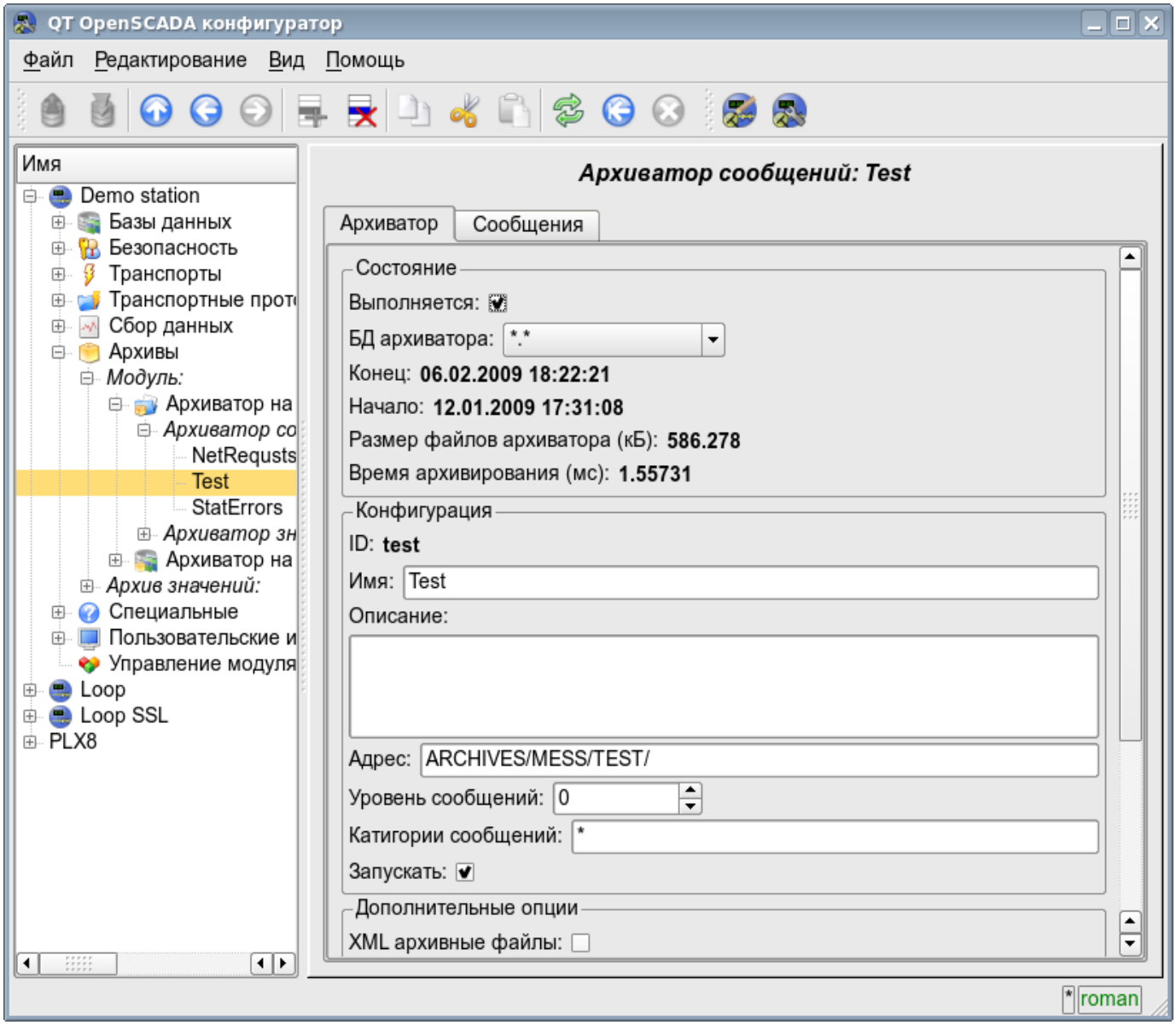

Рис. 4.6h. Главная вкладка конфигурации архиватора сообщений подсистемы "Архивы".

Вкладка "Сообщения" (рис.4.6i) содержит форму запроса сообщений из архива данного архиватора:

- *Время* — указывает время запроса.
- *Размер (сек)* — указывает размер или глубину запроса, в секундах.
- *Шаблон категории* — указывает категорию запрошенных сообщений. В категории можно указывать элементы выборки по шаблону, а именно символы '*' — для любой подстроки и '?' — для любого символа, а также регулярное выражение заключённое между символами '/' (/mod_(System|LogicLev)/).
- *Уровень* — указывает на минимальный уровень сообщений, т.е. запрос будет обработан для сообщений с уровнем более или равному указанному.

Таблица результата содержит строки сообщений с колонками:

- *Время* — время сообщения.
- *Категория* — категория сообщения.
- *Уровень* — уровень сообщения.
- *Сообщение* — текст сообщения.

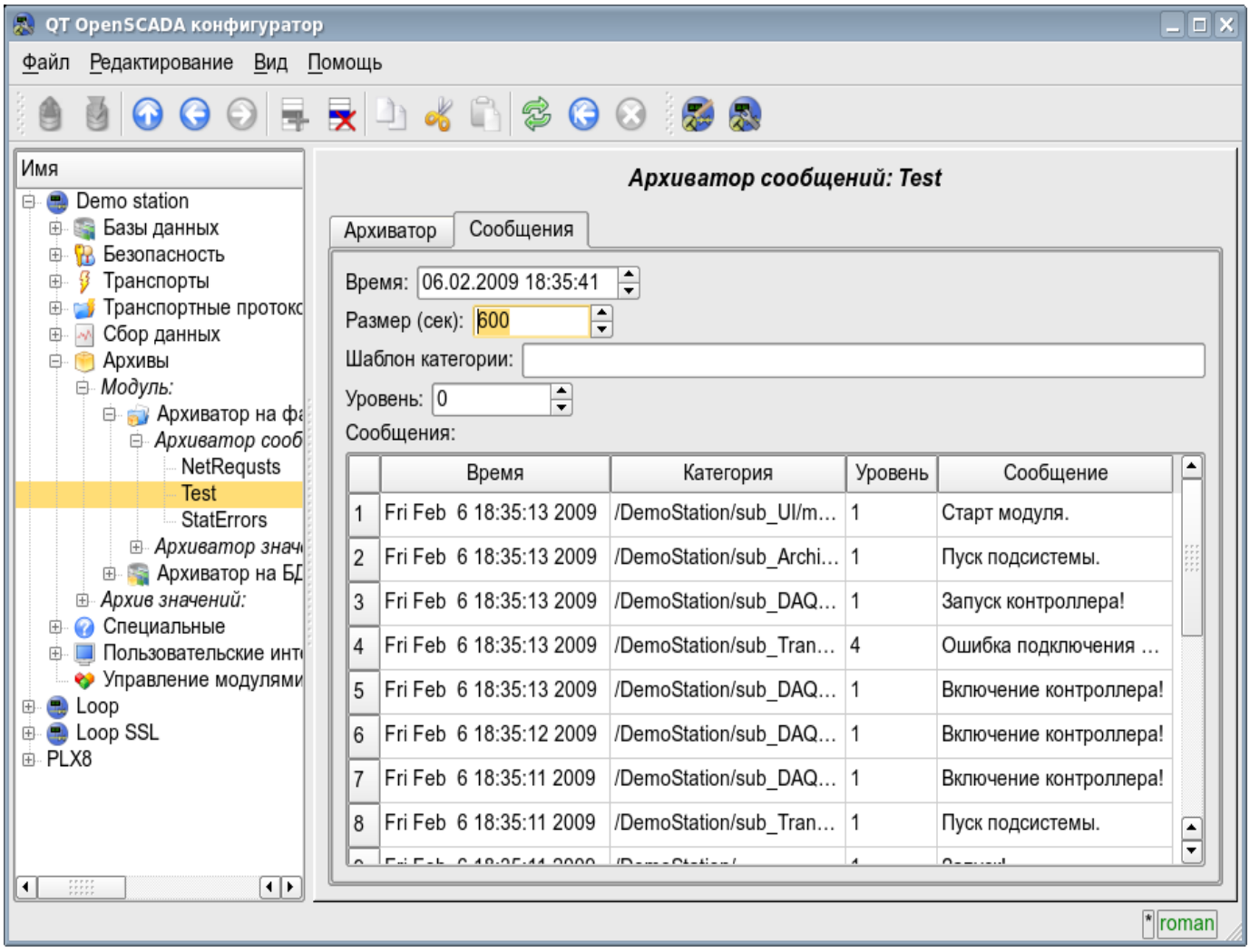

Рис. 4.6i. Вкладка запроса сообщений "Сообщения" архиватора сообщений подсистемы "Архивы".

Архиваторы значений содержат собственную страницу конфигурации с вкладками "Архиватор" и "Архивы".

Вкладка "Архиватор" (рис.4.6j) содержит основные настройки. Состав этих настроек может несколько отличаться от одного модуля этой подсистемы к другому, о чём можно узнать в собственной документации модулей. В качестве примера рассмотрим настройки архиватора значений у модуля архива на файловую систему Arch.FSArch Настройки:

- Раздел "Состояние" — содержит свойства характеризующие состояние архиватора:

- *Выполняется* — состояние архиватора "Выполняется". Исполняющийся архиватор обрабатывает буфера архивов значений и помещает их данные в своё хранилище, а также обслуживает запросы на доступ к данным в хранилище.
- *Время архивирования (мс)* — информация о времени затраченном на архивирование данных буферов архивов. Периодичность архивирования указывается в поле "Период архивирования" раздела "Конфигурация" этой вкладки.
- *БД архиватора* — адрес БД для хранения данных архиватора.

- Раздел "Конфигурация" — непосредственно содержит поля конфигурации:
  - *ID* — информация об идентификаторе архиватора.
  - *Имя* — указывает имя архиватора.
  - *Описание* — краткое описание архиватора и его назначения.
  - *Периодичность значений (сек)* — указывает периодичность значений, которые содержаться в хранилище архиватора.
  - *Период архивирования (сек)* — указывает периодичность задачи архивирования данных буферов архивов. Размерность буферов архивов во временном выражении должна быть не менее, а лучше несколько больше, периодичности задачи архивирования.
  - *Адрес* — адрес хранилища в специфичном для типа архиватора (модуля) формате. Описание формата записи адреса архиватора, как правило, доступно во всплывающей подсказке этого поля. В примере это относительный путь к директории хранилища.
  - *Запускать* — указывает на состояние "Выполняется", в которое переводить архиватор при загрузке.
- Раздел "Дополнительные опции" — специализированный для модуля раздел, о содержимом которого можно ознакомиться в документации на модуль.

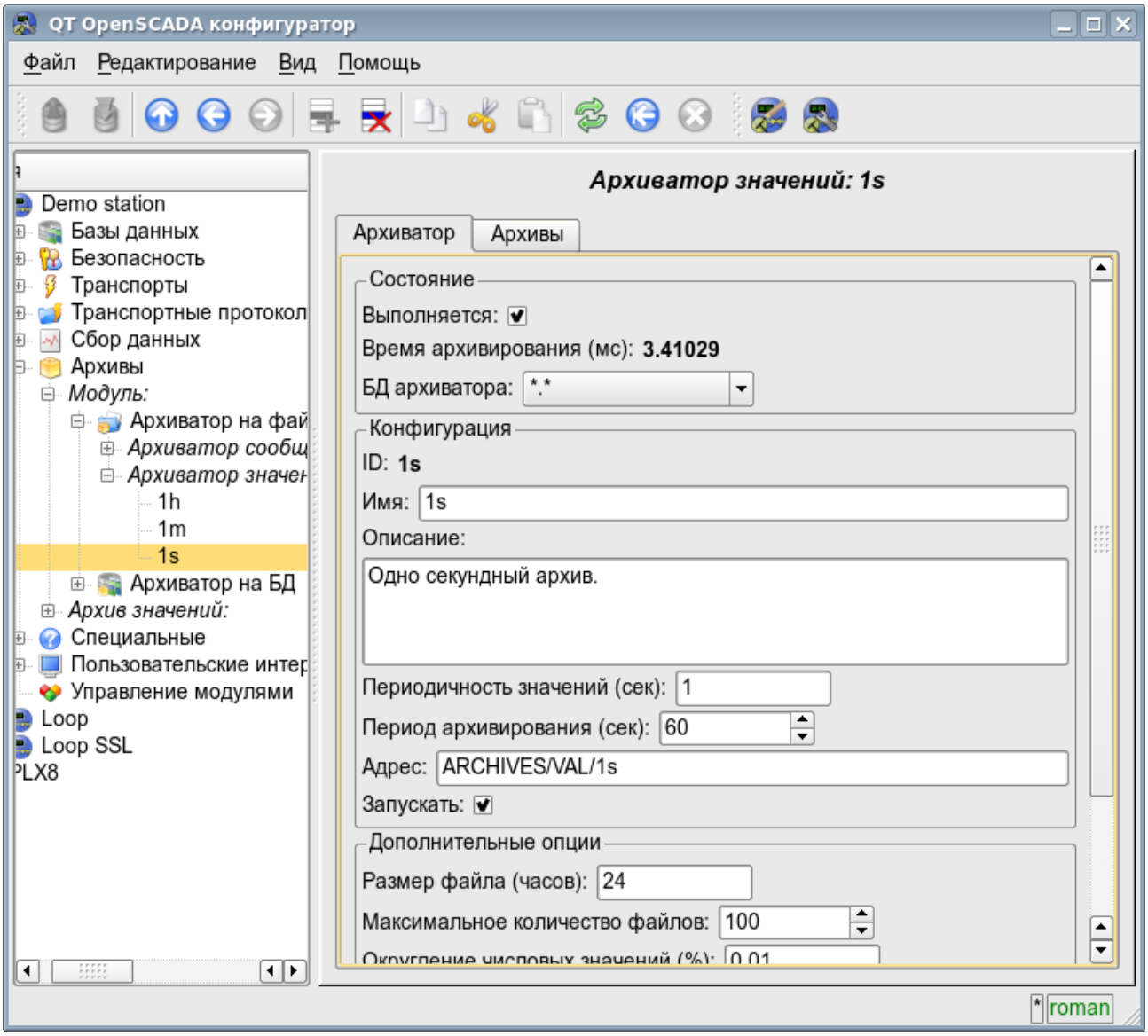

Рис. 4.6j. Главная вкладка конфигурации архиватора значений подсистемы "Архивы".

Вкладка "Архивы" (рис.4.6k) содержит таблицу с информацией об архивах, обрабатываемых архиватором. В строках таблица содержит архивы, а в колонках информацию:

- *Архив* — имя архива.
- *Период (с)* — периодичность архива в секундах.
- *Размер буфера* — размерность буфера в единицах.
- *Размер файлов (Мб)* — специфичное для модуля Arch.FSArch поле с информацией о суммарной размерности файлов хранилища архиватора для архива.

В случае с модулем Arch.FSArch в этой вкладке ещё содержится форма экспорта данных архиватора.

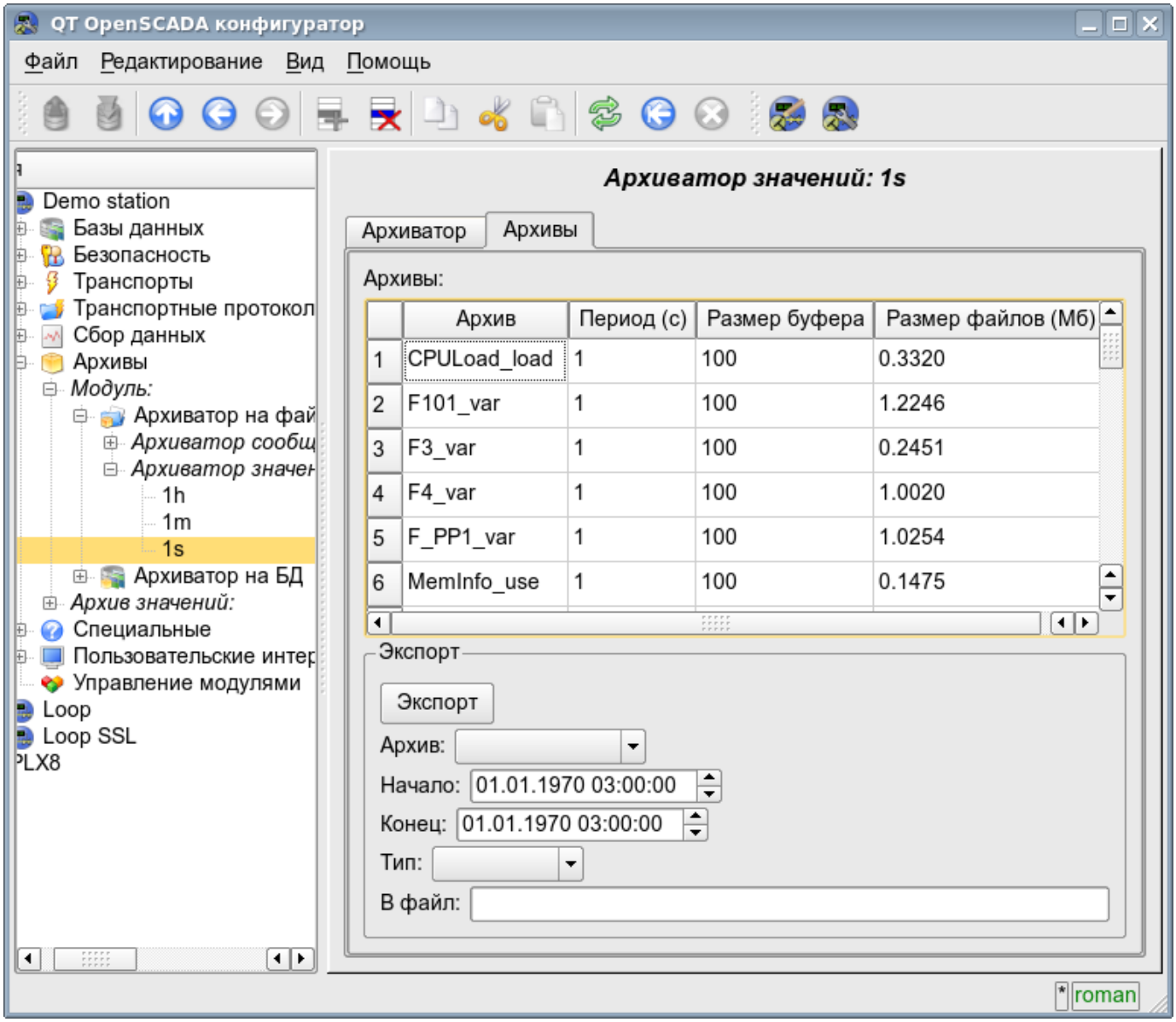

Рис. 4.6k. Вкладка "Архивы" архиватора значений подсистемы "Архивы".

4.7. Подсистема "Пользовательские интерфейсы"

Подсистема является модульной. Для конфигурации подсистемы предусмотрена корневая страница подсистемы "Пользовательские интерфейсы", содержащая вкладки "Модули" и "Помощь". Вкладка "Модули" (рис.4.1b) содержит список модулей подсистемы и идентична для всех модульных подсистем. Вкладка "Помощь" содержит краткую помощь для данной страницы.

Каждый модуль подсистемы "Пользовательские интерфейсы" предоставляет конфигурационную страницу с вкладками "Пользовательский интерфейс" и "Помощь". Вкладка "Пользовательский интерфейс" (рис.4.7a) предоставляет параметр для контроля за состоянием "Выполняется" модуля, а также разделы конфигурации специализированные для модулей этой подсистемы. Во вкладке "Помощь" содержится информация о модуле подсистемы "Пользовательские интерфейсы" (рис.4.1d), состав которой идентичен для всех модулей.

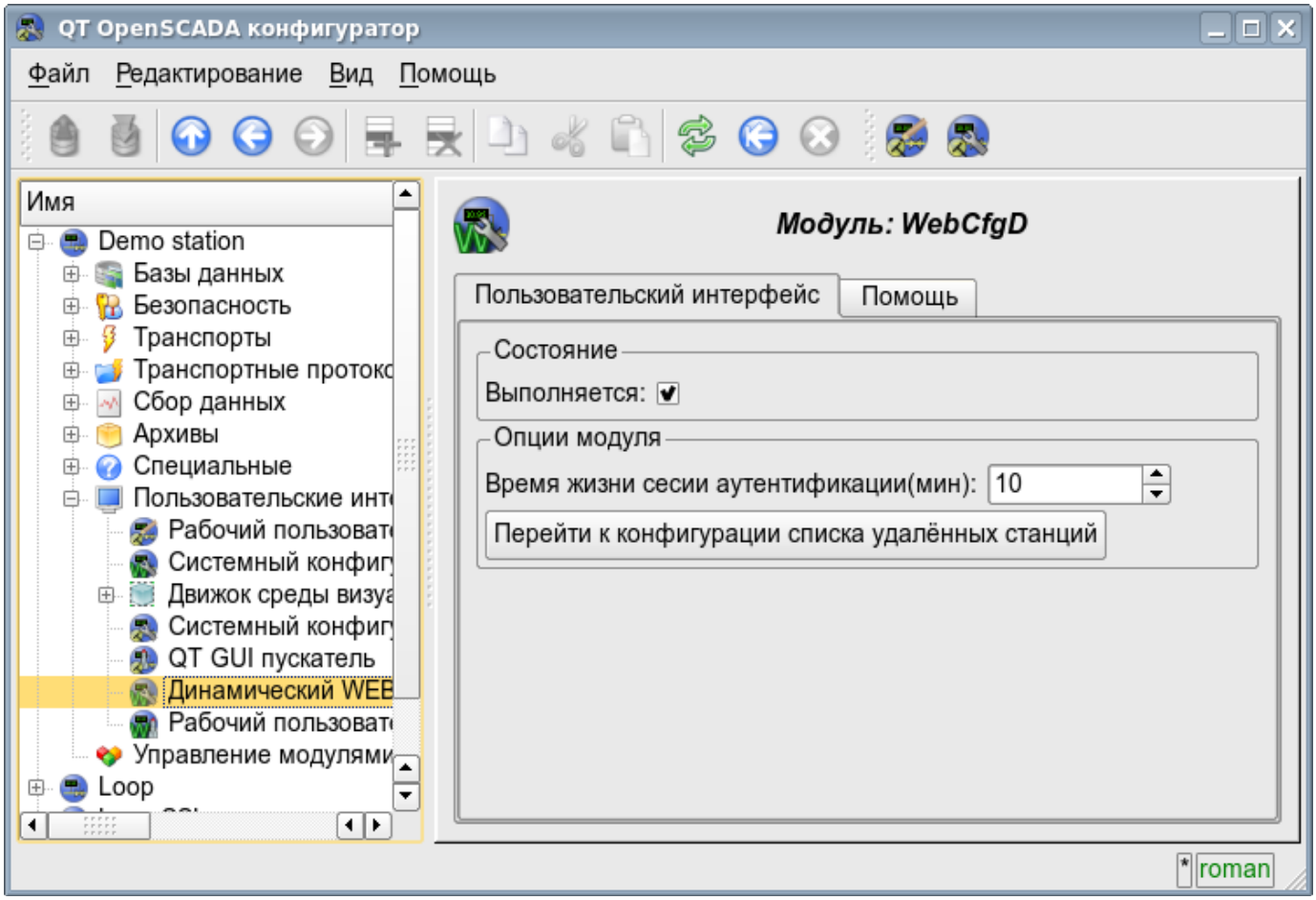

Рис. 4.7a. Вкладка "Пользовательский интерфейс" модуля подсистемы "Пользовательские интерфейсы".

4.8. Подсистема "Специальные"

Подсистема является модульной. Для конфигурации подсистемы предусмотрена корневая страница подсистемы "Специальные", содержащая вкладки "Модули" и "Помощь". Вкладка "Модули" (рис.4.1b) содержит список модулей подсистемы и идентична для всех модульных подсистем. Вкладка "Помощь" содержит краткую помощь для данной страницы.

Каждый модуль подсистемы "Специальные" предоставляет конфигурационную страницу с вкладками "Специальный модуль" и "Помощь". Вкладка "Специальный модуль" (рис.4.8a) предоставляет параметр для контроля за состоянием "Выполняется" модуля, а также разделы конфигурации специализированные для модулей этой подсистемы. Во вкладке "Помощь" содержится информация о модуле подсистемы "Специальные" (рис.4.1d), состав которой идентичен для всех модулей.

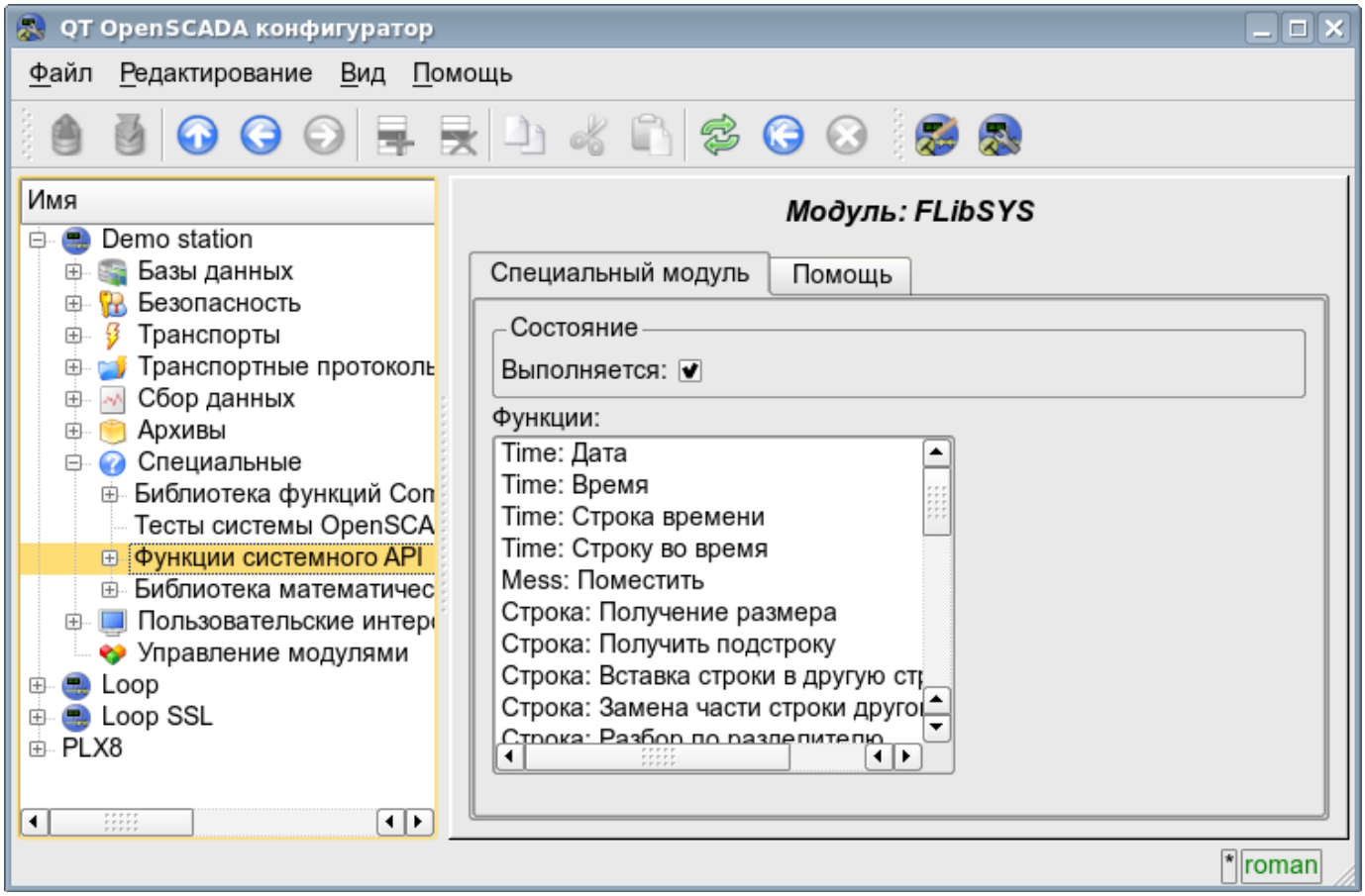

Рис. 4.8a. Вкладка "Специальный модуль" модуля подсистемы "Специальные".

4.9. Подсистема "Управление модулями"

Подсистема не является модульной. Для конфигурации подсистемы предусмотрена страница подсистемы "Управление модулями", содержащая вкладки "Подсистема" и "Помощь". Вкладка "Подсистема" (рис.4.9а) содержит основные настройки подсистемы. Вкладка "Помощь" содержит краткую помощь для данной страницы. Состав вкладки "Подсистема":

- *Путь к разделяемым библиотекам (модулям)* — информация о расположении директории с модулями системы OpenSCADA. Устанавливается параметром <*ModDir*> станции, конфигурационного файла.
- *Разрешённые модули* — информация о списке, разделённом символом ';', модулей, разрешённых для автоматического подключения и обновления. Значение '*' используется для разрешения всех модулей. Устанавливается параметром <*ModAllow*> раздела подсистемы, sub_ModSched, станции, конфигурационного файла.
- *Запрещённые модули* — информация о списке, разделённом символом ';', модулей, запрещённых для автоматического подключения и обновления. Устанавливается параметром <*ModDeny*> раздела подсистемы "sub_ModSched" станции конфигурационного файла. Список запрещённых модулей имеет больший приоритет чем разрешённых.
- *Период проверки модулей (сек)* — указывает на периодичность проверки модулей на факт их обновления. Модули, допустимые для автоматического подключения и обновления, будут автоматически обновлены.
- *Проверить модуль сейчас* — команда выполнить проверку модулей на факт их обновления. Модули, допустимые для автоматического подключения и обновления, будут автоматически обновлены.
- *Разделяемые библиотеки (модули)* — таблица с перечнем разделяемых библиотек с модулями, обнаруженными OpenSCADA. В строках расположены модули, а в колонках информация о них:
  - *Путь* — информация о полном пути к разделяемой библиотеке.
  - *Время* — информация о времени последней модификации файла разделяемой библиотеки.
  - *Модули* — информация о перечне модулей в разделяемой библиотеке.
  - *Включен* — состояние "Включен" разделяемой библиотеки. Привилегированным пользователям предоставляется возможность ручного включения/выключения разделяемых библиотек путём изменения этого поля.

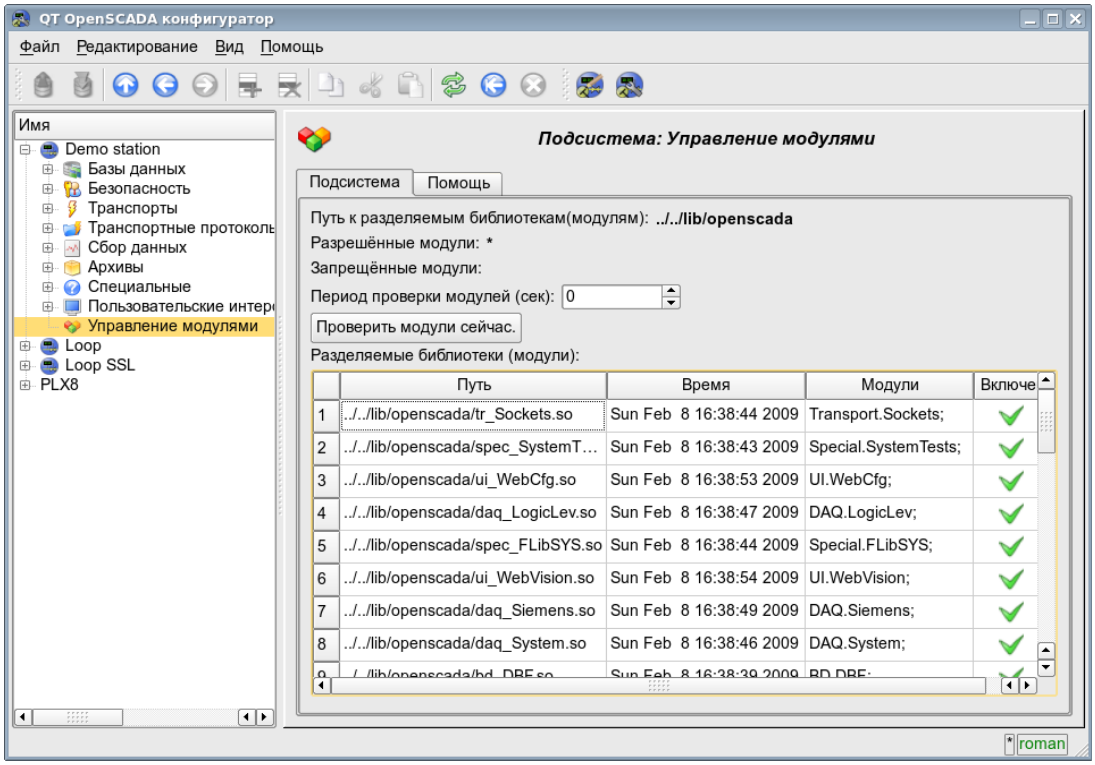

Рис. 4.9а. Главная вкладка конфигурации подсистемы "Управление модулями".

4.10. Конфигурационный файл OpenSCADA и параметры командной строки вызова OpenSCADA

Конфигурационный файл системы OpenSCADA предназначен для хранения системной и общей конфигурации OpenSCADA-станции. Только в конфигурационном файле и через параметры командной строки можно указать часть ключевых системных параметров станции, поэтому знакомство со структурой конфигурационного файла необходимо для специалистов развёртывающих решения на основе OpenSCADA.

Называться конфигурационный файл системы OpenSCADA может по разному, однако принято название oscada.xml и производные от него. Конфигурационный файл обычно указывается, при запуске станции, параметром командной строки *--Config=/home/roman/roman/work/OScadaD/etc/oscada_demo.xml*. Для удобства вызова создаются скрипты запуска станции с нужным конфигурационным файлом, например скрипт (openscada_demo) вызова демонстрационной станции:

```
#!/bin/sh
openscada --Config=/etc/oscada_demo.xml $@
```

Если конфигурационный файл не указан то используется стандартный конфигурационный файл: /etc/oscada.xml .

Структурно конфигурационный файл организован на расширяемом языке разметки текста XML. Следовательно требуется жёсткое соблюдение правил синтаксиса XML. Пример образца типового конфигурационного файла OpenSCADA, с узлами конфигурации большинства компонентов OpenASCADA, приведен ниже:

```
<?xml version="1.0" encoding="UTF-8" ?>
<OpenSCADA>
	<!-- This is the OpenSCADA configuration file. -->
	<station id="DemoStation">
		<!-- Discribe internal parameter for station. Station this only OpenSCADA programm. -->
		<prm id="StName">Demo station</prm>
		<prm id="StName_ru">Демо станция</prm>
		<prm id="StName_uk">Демо станція</prm>
		<prm id="WorkDB">SQLite.GenDB</prm>
		<prm id="Workdir">~/.openscada</prm>
		<prm id="IcoDir">./icons</prm>
		<prm id="ModDir">/usr/lib/openscada</prm>
		<prm id="LogTarget">10</prm>
		<prm id="MessLev">0</prm>
		<prm id="Lang2CodeBase">en</prm>
		<prm id="SaveAtExit">0</prm>
		<prm id="SavePeriod">0</prm>

		<node id="sub_BD">
			<prm id="SYSStPref">0</prm>
			<tbl id="DB">
				<fld ID="GenDB" TYPE="SQLite" NAME="Generic DB" NAME_ru="Основная БД"
					NAME_uk="Основна БД" ADDR="./DEMO/DemoSt.db" CODEPAGE="UTF-8"/>
			</tbl>
		</node>

		<node id="sub_Security">
			<!--
			<tbl id="Security_user">
				<fld
					NAME="root"
					DESCR="Super user"
					DESCR_ru="Супер пользователь"
					DESCR_uk="Супер користувач"
					PASS="openscada"/>
				<fld
					NAME="user"
					DESCR="System user"
					DESCR_ru="Системный пользователь"
					DESCR_uk="Системний користувач"
					PASS=""/>
			</tbl>
			<tbl id="Security_grp">
				<fld
					NAME="root"
					DESCR="Super users groups"
```

```
                        DESCR_ru="Группа суперпользователей"
                        DESCR_uk="Група суперкористувачів"
                        USERS="root;user"/>
        </tbl>-->
</node>

<node id="sub_ModSched">
        <prm id="ModAllow">*</prm>
        <prm id="ModDeny"></prm>
        <prm id="ChkPer">0</prm>
</node>

<node id="sub_Transport">
        <!--
        <tbl id="Transport_in">
                <fld
                        ID="WEB_1"
                        MODULE="Sockets"
                        NAME="Generic WEB interface"
                        NAME_ru="Основной WEB интерфейс"
                        NAME_uk="Основний WEB інтерфейс"
                        DESCRIPT="Generic transport for WEB interface."
                        DESCRIPT_ru="Основной транспорт для WEB интерфейса."
                        DESCRIPT_uk="Основний транспорт для WEB інтерфейсу."
                        ADDR="TCP::10002:0"
                        PROT="HTTP"
                        START="1"/>
                <fld
                        ID="WEB_2"
                        MODULE="Sockets"
                        NAME="Reserve WEB interface"
                        NAME_ru="Резервный WEB интерфейс"
                        NAME_uk="Резервний WEB інтерфейс"
                        DESCRIPT="Reserve transport for WEB interface."
                        DESCRIPT_ru="Резервный транспорт для WEB интерфейса."
                        DESCRIPT_uk="Резервний транспорт для WEB інтерфейсу."
                        ADDR="TCP::10004:0"
                        PROT="HTTP"
                        START="1"/>
        </tbl>
        <tbl id="Transport_out">
                <fld
                        ID="testModBus"
                        MODULE="Sockets"
                        NAME="Test ModBus"
                        NAME_ru="Тест ModBus"
                        NAME_uk="Тест ModBus"
                        DESCRIPT="Data exchange by protocol ModBus test."
                        DESCRIPT_ru="Тест обмена по протоколу ModBus."
                        DESCRIPT_uk="Тест обміну за протоколом ModBus."
                        ADDR="TCP:localhost:10502"
                        START="1"/>
        </tbl>-->
</node>

<node id="sub_DAQ">
        <!--
        <tbl id="tmplib">
                <fld ID="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                        DESCR="" DESCR_ru="" DESCR_uk="" DB="tmplib_test2"/>
        </tbl>
        <tbl id="tmplib_test2">
                <fld ID="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                        DESCR="" DESCR_ru="" DESCR_uk="" DB="test2"
                        PROGRAM="JavaLikeCalc.JavaScript&#010;cnt=5*i"/>
        </tbl>
        <tbl id="tmplib_test2_io">
                <fld TMPL_ID="test2" ID="i" NAME="I" NAME_ru="I" NAME_uk="I"
                        TYPE="4" FLAGS="160" VALUE="" POS="0"/>
                <fld TMPL_ID="test2" ID="cnt" NAME="Cnt" NAME_ru="Cnt" NAME_uk="Cnt"
                        TYPE="4" FLAGS="32" VALUE="" POS="0"/>
        </tbl>-->

        <node id="mod_LogicLev">
                <!--
                <tbl id="DAQ">
                        <fld
                                ID="test2"
                                NAME="Test 2"
                                NAME_ru="Тест 2"
                                NAME_uk="Тест 2"
                                DESCR=""
                                DESCR_ru=""
```

```
                        DESCR_uk=""
                        ENABLE="1"
                        START="1"
                        PRM_BD="test2prm"
                        PERIOD="1000"
                        PRIOR="0"/>
        </tbl>
        <tbl id="test2prm">
                <fld SHIFR="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                        DESCR="" DESCR_ru="" DESCR_uk="" EN="1" MODE="2"
                        PRM="test2.test2"/>
        </tbl>-->
</node>

<node id="mod_System">
        <!--
        <tbl id="DAQ">
                <fld
                        ID="DataOS"
                        NAME="Data OS"
                        NAME_ru="Данные ОС"
                        NAME_uk="Дані ОС"
                        DESCR="Data of services and subsystems OS."
                        DESCR_ru="Данные сервисов и подсистем ОС."
                        DESCR_uk="Дані сервісів та підсистем ОС."
                        ENABLE="1"
                        START="1"
                        AUTO_FILL="0"
                        PRM_BD="DataOSprm"
                        PERIOD="1000" PRIOR="0"/>
        </tbl>
        <tbl id="DataOSprm">
                <fld SHIFR="CPU" NAME="CPU load" NAME_ru="Нагрузка CPU"
                        NAME_uk="Навантаження CPU" DESCR="" DESCR_ru="" DESCR_uk=""
                        EN="1" TYPE="CPU" SUBT="gen"/>
                <fld SHIFR="MEM" NAME="Memory" NAME_ru="Память" NAME_uk="Пам\'ять"
                        DESCR="" DESCR_ru="" DESCR_uk="" EN="1" TYPE="MEM"/>
        </tbl> -->
</node>

<node id="mod_DiamondBoards">
        <!--
        <tbl id="DAQ">
                <fld ID="Athena" NAME="Athena board" NAME_ru="Плата Athena"
                        NAME_uk="Плата Athena" DESCR="" DESCR_ru="" DESCR_uk=""
                        ENABLE="1" START="0" BOARD="25" PRM_BD_A="AthenaAnPrm"
                        PRM_BD_D="AthenaDigPrm" ADDR="640" INT="5" DIO_CFG="0"
                        ADMODE="0" ADRANGE="0" ADPOLAR="0" ADGAIN="0"
                        ADCONVRATE="1000"/>
        </tbl>
        <tbl id="AthenaAnPrm">
                <fld SHIFR="ai0" NAME="AI 0" NAME_ru="AI 0" NAME_uk="AI 0"
                        DESCR="" DESCR_ru="" DESCR_uk=""
                        EN="0" TYPE="0" CNL="0" GAIN="0"/>
        </tbl>
        <tbl id="AthenaDigPrm">
                <fld SHIFR="di0" NAME="DI 0" NAME_ru="DI 0" NAME_uk="DI 0"
                        DESCR="" DESCR_ru="" DESCR_uk=""
                        EN="0" TYPE="0" PORT="0" CNL="0"/>
        </tbl> -->
</node>

<node id="mod_BlockCalc">
        <!--
        <tbl id="DAQ">
                <fld ID="Model" NAME="Model" NAME_ru="Модель" NAME_uk="Модель"
                        DESCR="" DESCR_ru="" DESCR_uk="" ENABLE="1" START="1"
                        PRM_BD="Model_prm" BLOCK_SH="Model_blcks"
                        PERIOD="1000" PRIOR="0" PER_DB="0" ITER="1"/>
        </tbl>
        <tbl id="Model_blcks">
                <fld ID="Klap" NAME="Klapan" NAME_ru="Клапан" NAME_uk="Клапан"
                        DESCR="" DESCR_ru="" DESCR_uk=""
                        FUNC="DAQ.JavaLikeCalc.lib_techApp.klap" EN="1" PROC="1"/>
        </tbl>
        <tbl id="Model_blcks_io">
                <fld BLK_ID="Klap" ID="l_kl1" TLNK="0" LNK="" VAL="50"/>
                <fld BLK_ID="Klap" ID="l_kl2" TLNK="0" LNK="" VAL="20"/>
        </tbl>
        <tbl id="Model_prm">
                <fld SHIFR="l_kl" NAME="Klap lev" NAME_ru="Полож. клапана"
                        NAME_uk="Полож. клапана" DESCR="" DESCR_ru="" DESCR_uk=""
                        EN="1" BLK="Klap" IO="l_kl1"/>
```

```
        </tbl> -->
</node>

<node id="mod_JavaLikeCalc">
        <!--
        <tbl id="DAQ">
                <fld ID="CalcTest" NAME="Calc Test" NAME_ru="Тест вычисл."
                        NAME_uk="Тест обчисл." DESCR="" DESCR_ru="" DESCR_uk=""
                        ENABLE="1" START="1" PRM_BD="Cal FUNC="TemplFunc.d_alarm"
                        PERIOD="1000" PRIOR="0" PER_DB="0" ITER="1"/>
        </tbl>
        <tbl id="CalcTest_val">
                <fld ID="in" VAL="0"/>
                <fld ID="alrm" VAL=""/>
                <fld ID="alrm_md" VAL="1"/>
                <fld ID="alrm_mess" VAL="Error present."/>
        </tbl>
        <tbl id="CalcTest_prm">
                <fld SHIFR="alrm" NAME="Alarm" NAME_ru="Авария" NAME_uk="Аварія"
                        DESCR="" DESCR_ru="" DESCR_uk="" EN="1" FLD="alrm"/>
        </tbl>
        <tbl id="lib">
                <fld ID="TemplFunc" NAME="" NAME_ru="" NAME_uk="" DESCR="" ESCR_ru=""
                        DESCR_uk="" DB="lib_TemplFunc"/>
        </tbl>
        <tbl id="lib_TemplFunc">
                <fld ID="d_alarm" NAME="Digit alarm" NAME_ru="Авария по дискр."
                        NAME_uk="Аварія за дискр" DESCR=""
                        FORMULA="alrm=(in==alrm_md)?&quot;1:&quot;
                                +alrm_mess:&quot;0&quot;;"/>
        </tbl>
        <tbl id="lib_TemplFunc_io">
                <fld F_ID="d_alarm" ID="in" NAME="Input" NAME_ru="Вход" NAME_uk="Вхід"
                        TYPE="3" MODE="0" DEF="" HIDE="0" POS="0"/>
                <fld F_ID="d_alarm" ID="alrm" NAME="Alarm" NAME_ru="Авария"
                        NAME_uk="Аварія" TYPE="0" MODE="1" DEF="" HIDE="0" POS="1"/>
                <fld F_ID="d_alarm" ID="alrm_md" NAME="Alarm mode"
                        NAME_ru="Режим аварии" NAME_uk="Режим аварії" TYPE="3"
                        MODE="0" DEF="" HIDE="0" POS="2"/>
                <fld F_ID="d_alarm" ID="alrm_mess" NAME="Alarm message"
                        NAME_ru="Сообщ. аварии" NAME_uk="Повід. аварії" TYPE="0"
                        MODE="0" DEF="" HIDE="0" POS="3"/>
        </tbl>-->
</node>

<node id="mod_Siemens">
        <!--
        <tbl id="DAQ">
                <fld ID="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                        DESCR="" DESCR_ru="" DESCR_uk="" ENABLE="1" START="1"
                        PRM_BD="test2prm" PERIOD="1000" PRIOR="0" CIF_DEV="0" ADDR="5"
                        ASINC_WR="0"/>
        </tbl>
        <tbl id="test2prm">
                <fld SHIFR="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                        DESCR="" DESCR_ru="" DESCR_uk="" EN="1" TMPL="S7.ai_man"/>
        </tbl>-->
</node>

<node id="mod_SNMP">
        <!--
        <tbl id="DAQ">
                <fld ID="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                        DESCR="" DESCR_ru="" DESCR_uk="" ENABLE="1" START="1"
                        PRM_BD="test2prm" PERIOD="1000" PRIOR="0" ADDR="localhost"
                        COMM="public" PATTR_LIM="20"/>
        </tbl>
        <tbl id="test2prm">
                <fld SHIFR="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                        DESCR="" DESCR_ru="" DESCR_uk="" EN="1" OID_LS="system"/>
        </tbl>-->
</node>

<node id="mod_ModBus">
        <!--
        <tbl id="DAQ">
                <fld ID="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                        DESCR="" DESCR_ru="" DESCR_uk="" ENABLE="1" START="1"
                        PRM_BD="test2prm" PERIOD="1000" PRIOR="0" TRANSP="Sockets"
                        ADDR="exlar.diya.org" NODE="1"/>
        </tbl>
        <tbl id="test2prm">
                <fld SHIFR="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
```

```
                                DESCR="" DESCR_ru="" DESCR_uk=""
                                EN="1" ATTR_LS="321:0:tst:Test"/>
                        </tbl>-->
                </node>

                <node id="mod_Transporter">
                        <!--
                        <tbl id="DAQ">
                                <fld ID="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                                        DESCR="" DESCR_ru="" DESCR_uk="" ENABLE="1" START="1"
                                        PRM_BD="test2prm" PERIOD="1000" PRIOR="0" SYNCPER="60"
                                        STATIONS="loop" CNTRPRM="System.AutoDA"/>
                        </tbl>-->
                </node>
        </node>

        <node id="sub_Archive">
                <prm id="MessBufSize">1000</prm>
                <prm id="MessPeriod">5</prm>
                <prm id="ValPeriod">1000</prm>
                <prm id="ValPriority">10</prm>
                <!--
                <tbl id="Archive_mess_proc">
                        <fld
                                ID="StatErrors"
                                MODUL="FSArch"
                                NAME="Errors"
                                NAME_ru="Ошибки"
                                NAME_uk="Помилки"
                                DESCR="Local errors\' archive"
                                DESCR_ru="Архив локальных ощибок"
                                DESCR_uk="Архів локальних помилок"
                                START="1"
                                CATEG="/DemoStation*"
                                LEVEL="4"
                                ADDR="ARCHIVES/MESS/stError/"
                                FSArchMSize="300"
                                FSArchNFiles="10"
                                FSArchTmSize="30"
                                FSArchXML="1"
                                FSArchPackTm="10"
                                FSArchTm="60"/>
                        <fld
                                ID="NetRequsts"
                                MODUL="FSArch"
                                NAME="Net requests"
                                NAME_ru="Сетевые запросы"
                                NAME_uk="Мережеві запити"
                                DESCR="Requests to server through transport Sockets."
                                DESCR_ru="Запросы к серверу через транспорт Sockets."
                                DESCR_uk="Запити до сервера через транспорт Sockets."
                                START="1"
                                CATEG="/DemoStation/Transport/Sockets*"
                                LEVEL="1"
                                ADDR="ARCHIVES/MESS/Net/"
                                FSArchMSize="300"
                                FSArchNFiles="10"
                                FSArchTmSize="30"
                                FSArchXML="1"
                                FSArchPackTm="10"
                                FSArchTm="60"/>
                </tbl>
                <tbl id="Archive_val_proc">
                        <fld
                                ID="1h"
                                MODUL="FSArch"
                                NAME="1hour"
                                NAME_ru="1час"
                                NAME_uk="1год"
                                DESCR="Averaging for hour"
                                DESCR_ru="Усреднение за час"
                                DESCR_uk="Усереднення за годину"
                                START="1"
                                ADDR="ARCHIVES/VAL/1h/"
                                V_PER="360"
                                A_PER="60"
                                FSArchTmSize="8640"
                                FSArchNFiles="10"
                                FSArchRound="0.1"
                                FSArchPackTm="10"
                                FSArchTm="60"/>
                </tbl>
                <tbl id="Archive_val">
```

```
                <fld
                        ID="test1"
                        NAME="Test 1"
                        NAME_ru="Тест 1"
                        NAME_uk="Тест 1"
                        DESCR="Test 1"
                        DESCR_ru="Тест 1"
                        DESCR_uk="Тест 1"
                        START="1"
                        VTYPE="1"
                        BPER="1"
                        BSIZE="200"
                        BHGRD="1"
                        BHRES="0"
                        SrcMode="0"
                        Source=""
                        ArchS=""/>
        </tbl>-->
</node>

<node id="sub_Protocol">
</node>

<node id="sub_UI">
        <node id="mod_QTStarter">
                <prm id="StartMod">QTCfg</prm>
        </node>
        <node id="mod_WebCfg">
                <prm id="SessTimeLife">20</prm>
        </node>
        <node id="mod_VCAEngine">
                <!--
                <tbl id="LIB">
                        <fld ID="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                                DESCR="" DESCR_ru="" DESCR_uk="" DB_TBL="wlib_test2" ICO=""
                                USER="root" GRP="UI" PERMIT="436"/>
                </tbl>
                <tbl id="wlib_test2">
                        <fld ID="test2" ICO="" PARENT="/wlb_originals/wdg_Box" PROC=""
                                PROC_ru="" PROC_uk="" PROC_PER="-1" USER="root" GRP="UI"
                                PERMIT="436"/>
                </tbl> <tbl id="wlib_test2_io">
                        <fld IDW="test2" ID="name" IO_VAL="Test 2" IO_VAL_ru="Тест 2"
                                IO_VAL_uk="Тест 2" SELF_FLG="" CFG_TMPL="" CFG_TMPL_ru=""
                                CFG_TMPL_uk="" CFG_VAL=""/>
                        <fld IDW="test2" ID="dscr" IO_VAL="Test module 2"
                                IO_VAL_ru="Тест модуля 2" IO_VAL_uk="Тест модуля 2"
                                SELF_FLG="" CFG_TMPL="" CFG_TMPL_ru="" CFG_TMPL_uk=""
                                CFG_VAL=""/>
                </tbl>
                <tbl id="PRJ">
                        <fld ID="test2" NAME="Test 2" NAME_ru="Тест 2" NAME_uk="Тест 2"
                                DESCR="" DESCR_ru="" DESCR_uk="" DB_TBL="prj_test2" ICO=""
                                USER="root" GRP="UI" PER </tbl> <tbl id="prj_test2">
                        <fld OWNER="/test2" ID="pg1" ICO="" PARENT="/wlb_originals/wdg_Box"
                                PROC="" PROC_ru="" PROC_uk="" PROC_PER="-1" USER="root"
                                GRP="UI" PERMIT="436" FLGS="1"/>
                        <fld OWNER="/test2/pg1" ID="pg2" ICO=""
                                PARENT="/wlb_originals/wdg_Box" PROC="" PROC_ru="" PROC_uk=""
                                PROC_PER="-1" USER="root" GRP="UI" PERMIT="436" FLGS="0"/>
                </tbl>
                <tbl id="prj_test2_incl">
                        <fld IDW="/prj_test2/pg_pg1" ID="wdg1"
                                PARENT="/wlb_originals/wdg_Box"/>
                </tbl>-->
        </node>
</node>

<node id="sub_Special">
        <node id="mod_SystemTests">
                <prm id="PARAM" on="0" per="5" name="LogicLev.experiment.F3"/>
                <prm id="XML" on="0" per="10" file="/etc/oscada.xml"/> <prm id="MESS" on="0"
                  per="10" categ="" arhtor="DBArch.test3"/>
                <prm id="SOAttDet" on="0" per="20" name="../../lib/openscada/daq_LogicLev.so"
                  full="1"/>
                <prm id="Val" on="0" per="1" name="LogicLev.experiment.F3.var" arch_len="5"
                  arch_per="1000000"/>
                <prm id="Val" on="0" per="1" name="System.AutoDA.CPULoad.load" arch_len="10"
                  arch_per="1000000"/>
                <prm id="BD" on="0" per="10" type="MySQL"
                  bd="server.diya.org;roman;123456;oscadaTest"
                  table="test" size="1000"/>
                <prm id="BD" on="0" per="10" type="DBF" bd="./DATA/DBF" table="test.dbf"
```

```
                    size="1000"/>
                <prm id="BD" on="0" per="10" type="SQLite" bd="./DATA/test.db" table="test"
                    size="1000"/>
                <prm id="BD" on="0" per="10" type="FireBird"
                    bd="server.diya.org:/var/tmp/test.fdb;roman;123456"
                    table="test" size="1000"/>
                <prm id="TrOut" on="0" per="1" addr="TCP:127.0.0.1:10001" type="Sockets"
                    req="time"/>
                <prm id="TrOut" on="0" per="1" addr="UDP:127.0.0.1:10001" type="Sockets"
                    req="time"/>
                <prm id="TrOut" on="0" per="1" addr="UNIX:./oscada" type="Sockets"
                    req="time"/>
                <prm id="TrOut" on="0" per="1" addr="UDP:127.0.0.1:daytime" type="Sockets"
                    req="time"/>
                <prm id="Func" on="0" per="10"/> <prm id="SysContrLang" on="0" per="10"
                    path="/Archive/FSArch/mess_StatErrors/%2fprm%2fst"/>
                <prm id="ValBuf" on="0" per="5"/> <prm id="Archive" on="0" per="30"
                    arch="test1" period="1000000"/>
                <prm id="Base64Code" on="0" per="10"/>
            </node>
        </node>
    </station>
</OpenSCADA>
```

Рассмотрим детальнее структуру конфигурационного файла. Один конфигурационный файл может содержать конфигурацию нескольких станций в секциях *<station id="DemoStation"/>*. Атрибутом указывается идентификатор станции. Использование той или иной секции станции, при вызове, указывается параметром командной строки *--Station=DemoStation*. Секция станции непосредственно содержит параметры станции и секции подсистем. Параметры конфигурации секции записываются в виде *<prm id="StName">Demo station</prm>*. Где в атрибуте *<id>* указывается идентификатор атрибута, а в теле тега указывается значение параметра "Demo station". Перечень доступных параметров и их описание для станции и всех остальных секций можно получить в консоли, посредством вызова OpenSCADA с параметром --help или во вкладках "Помощь" страниц компонентов конфигурационных файлов OpenSCADA (рис.4.10а).

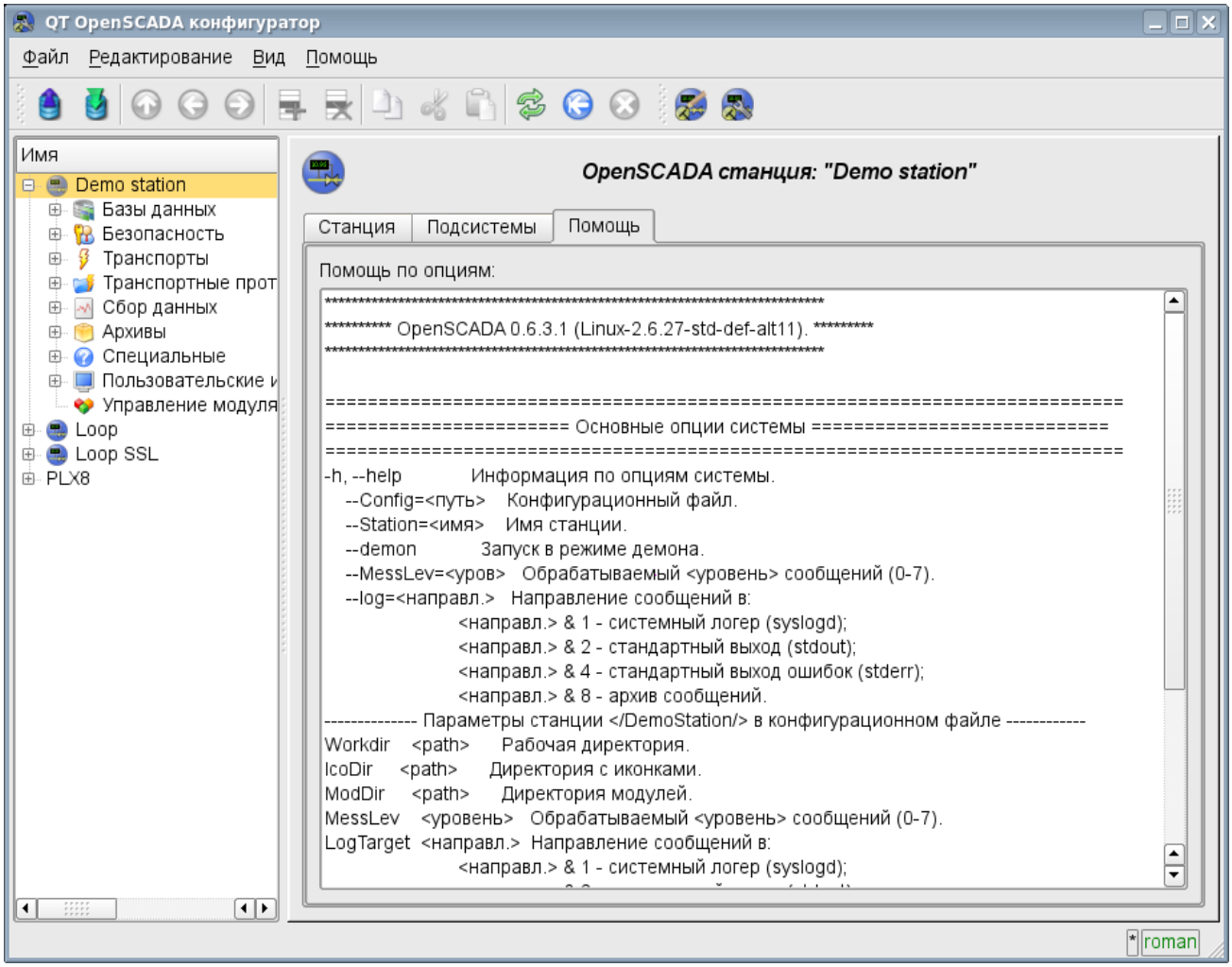

Рис. 4.10а. Вкладка "Помощь" компонента OpenSCADA.

Результат вызова команды: **# ./openscada_demo --help**

```
**************************************************************************
********** OpenSCADA 0.6.4.1 (Linux-2.6.30-std-def-alt15). *********
**************************************************************************
==========================================================================
====================== Основные опции системы ============================
==========================================================================
-h, --help            Информация по опциям системы.
    --Config=<путь>   Конфигурационный файл.
    --Station=<имя>   Имя станции.
    --demon           Запуск в режиме демона.
    --MessLev=<уров>  Обрабатываемый <уровень> сообщений (0-7).
    --log=<направл.>  Направление сообщений в:
                      <направл.> & 1 - системный логер (syslogd);
                      <направл.> & 2 - стандартный выход (stdout);
                      <направл.> & 4 - стандартный выход ошибок (stderr);
                      <направл.> & 8 - архив сообщений.
-------------- Параметры станции </EmptySt/> в конфигурационном файле ------------
StName     <имя>       Имя станции.
WorkDB     <Тип.Имя>   Рабочая БД (тип и имя).
Workdir    <path>      Рабочая директория.
IcoDir     <path>      Директория с иконками.
ModDir     <path>      Директория модулей.
MessLev    <уровень>   Обрабатываемый <уровень> сообщений (0-7).
LogTarget <направл.>   Направление сообщений в:
                       <направл.> & 1 - системный логер (syslogd);
                       <направл.> & 2 - стандартный выход (stdout);
                       <направл.> & 4 - стандартный выход ошибок (stderr);
                       <направл.> & 8 - архив сообщений.
Lang2CodeBase <язык>   Базовый язык для перевода текстовых переменных, два символа.
SaveAtExit <true>      Сохранять систему при выходе.
SavePeriod <сек>       Период сохранения системы.

=================== Подсистема "Управление модулями" ====================
    --ModPath=<путь> Путь к модулям (/var/os/modules/).
--------- Параметры секции </DemoStation/sub_ModSched/> в конфигурационном файле ---------
ModPath    <путь>      Путь к разделяемым библиотекам (модулям).
ModAllow   <список>    Список разделяемых библиотек допустимых для автоматической загрузки, подключения и
                       запуска (bd_DBF.so;daq_JavaLikeCalc.so).
                       Использовать значение '*' для разрешения всех модулей.
ModDeny    <список>    Список разделяемых библиотек запрещённых для автоматической загрузки, подключения и
                       запуска (bd_DBF.so;daq_JavaLikeCalc.so).
ChkPer     <сек>       Период поиска новых разделяемых библиотек (модулей).

========================== Опции подсистемы "БД" ==========================
-------------- Параметры станции </DemoStation/sub_BD/> в конфигурационном файле ------------
SYSStPref    <1>   Использовать идентификатор станции в общей (SYS) таблице.

==================== Опции подсистемы "Безопасности" ====================

====================== Опции подсистемы "Транспорты" ====================

============== Опции подсистемы "Транспортные протоколы" ================

======================= Опции модуля <Protocol:HTTP> ============================
------- Параметры модульной секции </DemoStation/sub_Protocol/mod_HTTP/> в конфигурационном файле ---------
AuthTime  <мин>        Время жизни аутентификации, минут (по умолчанию 10).

=================== Опции подсистемы "Сбор данных" ======================
--------------- Параметры секции </DemoStation/sub_DAQ/> в конфигурационном файле --------------
RdStLevel  <уров>      Уровень текущей станции в схеме резервирования.
RdTaskPer  <с>         Периодичность вызова задачи обслуживания резервирования.
RdRestConnTm <с>       Интервал времени восстановления соединения с "мёртвой" резервной станцией.
RdRestDtTm <час>       Глубина восстановления данных архива из резервной станции, при включении.
RdStList   <список>    Список резервных станций, разделённых символом ';' (st1;st2).

==================== Опции подсистемы "Архивы" ======================
---------- Параметры секции </DemoStation/sub_Archive/> в конфигурационном файле --------
MessBufSize <ед.>      Размер буфера сообщений.
MessPeriod <сек>       Период архивирования сообщений.
ValPeriod <мсек>       Период архивирования значений.
ValPriority <уровень> Уровень приоритета задачи значений.
MaxReqMess <ед.>       Максимальное количество запрашиваемых сообщений.
MaxReqVals <ед.>       Максимальное количество запрашиваемых значений.

==================== Опции подсистемы "Специальные" ====================

========================= Опции модуля <Special:SystemTests> =========================
-------- Параметры модульной секции </DemoStation/sub_Special/mod_SystemTests/> в конфигурационном файле
--------
Общие опции всех тестов:
```

```
  id           идентификатор теста;
  on           флаг включения теста;
  per          период повторения (сек).
      *** Опции тестов ***
1) Param       Тест DAQ параметров. Вычитывает атрибуты и конфигурационные поля параметра.
  1:name       Адрес DAQ параметра
2) XML  Тест разбора файла XML. Парсит и отображает структуру указанного файла.
  1:file       XML файл
3)  Mess  Тест  архива  сообщений.  Периодически  вычитывает  новые  сообщения  из  архива,  для  указанного
архиватора.
  1:arhtor     Архиватор
  2:categ      Шаблон категории сообщения
  3:depth      Глубина сообщения (с)
4) SOAttach    Тест подключения/отключения модулей.
  1:name       Путь к модулю
  2:mode       Режим (1-подключ.;-1-отключ.;0-изменение)
  3:full       Полное подключение(при старте)
5) Val  Тест значений атрибута параметра.
Выполняет периодический опрос последнего значения указанного атрибута, а также опрос архива на указанную
глубину.
  1:name       Путь к атрибуту параметра
  2:arch_len   Глубина запроса к архиву значений (с)
  3:arch_per   Период запроса к архиву значений (мкс)
6) DB   Полный тест БД. Выполняет:
  - создание/открытие БД;
  - создание/открытие таблицы;
  - создание множества записей (строк) предопределённой структуры;
  - модификация множества записей;
  - получение и проверка значений множества записей;
  - модификация структуры записи и таблицы;
  - удаление записей;
  - закрытие/удаление таблицы;
  - закрытие/удаление БД.
  1:type       Тип БД
  2:addr       Адрес БД
  3:table      Таблица БД
  4:size       Количество записей
7) TrOut       Тест выходных и/или входных транспортов.
Выполняет тестирование исходящего транспорта путём отправления запроса к указанному входящему транспорту.
  1:addr       Адрес
  2:type       Модуль транспорта
  3:req Текст запроса
8) SysContrLang Тест языка управления системой.
Производит запрос элементов языка посредством полного пути.
Полный путь к элементу языка имеет вид </Archive/%2fbd%2fm_per>.
Полный путь состоит из двух вложенных путей.
Первый </d_Archive/> это путь к узлу дерева контроля.
Второй </bd/m_per> это путь к конкретному элементу узла.
  1:path       Путь к элементу языка
9) ValBuf          Тесты буфера значений. Содержит 13 тестов всех аспектов буфера значений (подсистема
"Архивы").
10) Archive    Тесты размещения в архиве значений.
Содержит 7(8) тестов архиватора значений на проверку корректности функционирования последовательного
механизма упаковки.
  1:arch       Архив значений
  2:period     Период значений (мкс)
11) Base64Code  Тесты кодирования Mime Base64 алгоритмом.

============= Опции подсистемы "Пользовательские интерфейсы" ============

======================== Опции модуля <UI:Vision> ============================
------- Параметры модульной секции </DemoStation/sub_UI/mod_Vision/> в конфигурационном файле ---------
StartUser    <польз>  Стартовый, безпарольный, пользователь.
RunPrjs      <список> Перечень запускаемых при старте проектов.
RunTimeUpdt  <mode>   Режим обновления динамики в RunTime (0 - адаптивное периодическое обновление всех
                      виджетов, 1 - обновление только изменённых виджетов).
VCAstation   <id>     Станция с движком СВУ ('.' - локальная).

======================== Опции модуля <UI:VCAEngine> ============================
    --VCADBClearForce Принудительная очистка БД СВУ от данных API 1.

======================== Опции модуля <UI:QTCfg> ============================
------- Параметры модульной секции </DemoStation/sub_UI/mod_QTCfg/> в конфигурационном файле ---------
StartPath    <path>   Стартовый путь конфигуратора.
StartUser    <user>   Стартовый, безпарольный, пользователь.

======================== Опции модуля <UI:QTStarter> ============================
------- Параметры модульной секции </DemoStation/sub_UI/mod_QTStarter/> в конфигурационном файле ---------
StartMod     <модули> Список запускаемых модулей (разделитель - ';');

======================== Опции модуля <UI:WebVision> ============================
------- Параметры модульной секции </DemoStation/sub_UI/mod_WebVision/> в конфигурационном файле ---------
SessTimeLife <время>  Время жизни сессии, минуты (по умолчанию 10).
```

Секции подсистем (<*node id="sub_DAQ" />*) содержат параметры подсистемы, секции модулей и секции таблиц отражения данных баз данных в конфигурационном файле. Секции модулей (<*node id="mod_DiamondBoards" />*) содержат индивидуальные параметры модулей и секции таблиц отражения данных баз данных в конфигурационном файле.

Секции таблиц отражения данных баз данных предназначены для размещения в конфигурационном файле записей таблиц БД для компонентов OpenSCADA. Рассмотрим таблицу входящих транспортов "Transport_in" подсистемы транспорты (<*node id="sub_Transport"*>) из примера конфигурационного файла выше. Таблица содержит две записи с полями: ID, MODULE, NAME, DESCRIPT, ADDR, PROT, START. После загрузки с такой секцией и вообще без БД в подсистеме "Транспорты" модуля "Sockets" появятся два входных транспорта. Форматы структур таблиц основных компонентов включены в демонстрационные конфигурационные файлы. За деталями структуры БД нужно обращаться к документации соответствующих модулей.

5. Общесистемное API пользовательского программирования.

API пользовательского программирования представляет собой дерево объектов системы OpenSCADA, каждый объект которого может представлять собственный перечень свойств и функций. Свойства и функции объектов могут использоваться пользователем в процедурах на языках пользовательского программирования OpenSCADA. Точкой входа для доступа к объектам системы OpenSCADA из языка пользовательского программирования JavaLikeCalc является зарезервированное слово "SYS" корневого объекта OpenSCADA. Например, для доступа к функции исходящего транспорта нужно записать: **SYS.Transport.Serial.out_ModBus.messIO(mess);**.

API объектов, предоставляемых модулями, описывается в собственной документации модуля.

5.1. Общесистемные пользовательские объекты.

Абстрактный объект представляет собой ассоциативный контейнер свойств и функций. Свойства могут содержать как данные четырёх базовых типов, так и другие объекты. Доступ к свойствам объекта обычно осуществляется посредством записи имён свойств через точку к объекту *<obj.prop>*, а также посредством заключения имени свойства в квадратные скобки *<obj["prop"]>*. Очевидно, что первый механизм статичен, а второй позволяет указывать имя свойства через переменную. Базовое определение объекта не содержит функций. Операции копирования объекта на самом деле делают ссылку на исходный объект. При удалении объекта осуществляется уменьшения счётчика ссылок, а при достижении счётчика ссылок нуля объект удаляется физически.

Разные компоненты могут доопределять базовый объект особыми свойствами и функциями. Стандартным расширением объекта является массив "Array".

Объект Array

Особенностью массива является то, что он работает со свойствами, как с индексами, и полное их именование бессмысленно, а значит доступен механизм обращения только заключением индекса в квадратные скобки *<arr[1]>*. Массив хранит свойства в собственном контейнере одномерного массива. Цифровые свойства массива используются для доступа непосредственно к массиву, а символьные работают как свойства объекта.

Массив предоставляет специальное свойство "length" для получения размера массива *<var = arr.length;>*. Также массив предоставляет следующие функции:

- *string join(string sep = ",")*, *string toString(string sep = ",")*, *string valueOf(string sep = ",")* — Возвращает строку с элементами массива разделёнными *<sep>* или символом ','.
- *Array concat(Array arr);* — Добавляет к исходному массиву элементы массива *<arr>*. Возвращает исходный массив с изменениями.
- *int push(ElTp var, ...);* — Помещает элемент(ы) *<var>* в конец массива, как в стек. Возвращает новый размер массива.
- *ElTp pop( );* — Удаление последнего элемента массива и возврат его значения, как из стека.
- *Array reverse( );* — Изменение порядка расположения элементов массива. Возвращается исходный массив с изменениями.
- *ElTp shift( );* — Сдвиг массива в верх. При этом первый элемент массива удаляется, а его значение возвращается.
- *int unshift(ElTp var, ...);* — Задвигает элемент(ы) *<var>* в массив. Первый элемент в 0, второй в 1 и т.д.
- *Array slice(int beg, int end);* — Возвращает фрагмент массива от *<beg>* к *<end>* (исключая). Если значение начала или конца отрицательно, то отсчёт ведётся с конца массива. Если конец не указан, то концом является конец массива.
- *Array splice(int beg, int remN, ElTp val1, ElTp val2, ...);* — Вставляет, удаляет или заменяет элементы массива. Возвращает массив удалённых элементов. В первую очередь

осуществляется удаление элементов с позиции <*beg*> и количеством <*remN*>, а затем вставляются значения <*val1*> и т.д., начиная с позиции <*beg*>.

- *Array sort( );* — Сортировка элементов массива в лексикографическом порядке.

Объект RegExp

Объект работы с регулярными выражениями, основан на библиотеке PCRE. При глобальном поиске устанавливается атрибут объекта "lastIndex", что позволяет продолжить поиск при следующем вызове функции. В случае неудачного поиска атрибут "lastIndex" сбрасывается в ноль.

В качестве аргументов создания объекта передаётся строка с текстом регулярного выражения и флаги в виде строки символов:

- 'g' — режим глобального поиска;
- 'i' — режим регистронезависимого поиска;
- 'm' — режим многострочного поиска;
- 'p' — тестирование выражения по обычному шаблону с ключевыми символами: '?', '*' и '\'.

Свойства объекта:

- *source* — Исходный шаблон регулярного выражения, только чтение.
- *global* — Признак глобального поиска, только чтение.
- *ignoreCase* — Признак игнорировать регистр символов при поиске, только чтение.
- *multiline* — Признак многострочного поиска, только чтение.
- *lastIndex* — Индекс символа за подстрокой последнего поиска. Используется в глобальном режиме для продолжения сканирования, при следующем вызове.

Функции объекта:

- *Array exec(string val);* — Вызов поиска по строке <*val*>. Возвращает найденную подстроку (0) и подвыражения (>0) в массиве. Устанавливает атрибут массива "index" в позицию найденной подстроки. Устанавливает атрибут массива "input" в значение исходной строки.

```
var re = new RegExp("(\\d\\d)[-/](\\d\\d)[-/](\\d\\d(?:\\d\\d)?)","");
var rez = re.exec("12/30/1969");
var month = rez[1];
var day = rez[2];
var year = rez[3];
```

- *bool test(string val);* — Возвращает "true" если подстрока найдена в <*val*>.

```
var re = new RegExp("(\\d\\d)[-/](\\d\\d)[-/](\\d\\d(?:\\d\\d)?)","");
var OK = re.test("12/30/1969");
```

Объект XMLNodeObj

Функции:

- *string name( )* — Имя узла, XML-тега.
- *string text( )* — Текст узла, содержимое XML-тега.
- *string attr(string id)* — Значение атрибута узла <*id*>.
- *XMLNodeObj setName(string vl)* — Установка имени узла в <*vl*>. Возвращает текущий узел.
- *XMLNodeObj setText(string vl)* — Установка текста узла в <*vl*>. Возвращает текущий узел.
- *XMLNodeObj setAttr(string id, string vl)* — Установка атрибута <*id*> в значение <*vl*>. Возвращает текущий узел.
- *int childSize( )* — Количество вложенных узлов.
- *XMLNodeObj childAdd(ElTp no = XMLNodeObj); XMLNodeObj childAdd(string no)* — Добавление объекта <*no*> как вложенного. <*no*> может быть как непосредственно объектом-результатом функции *SYS.XMLNode()*, так и строкой с именем нового тега. Возвращается вложенный узел.
- *XMLNodeObj childIns(int id, ElTp no = XMLNodeObj); XMLNodeObj childIns(int id, string no)* — Вставка объекта <*no*> как вложенного в позицию <*id*>. <*no*> может быть как непосредственно объектом-результатом функции *SYS.XMLNode()*, так и строкой с именем нового тега. Возвращается вложенный узел.

- *XMLNodeObj childDel(int id)* — Удаление вложенного узла в позиции <*id*>. Возвращает текущий узел.
- *XMLNodeObj childGet(int id)* — Получение вложенного узла в позиции <*id*>.
- *XMLNodeObj parent()* — Получить родительский узел.
- *string load(string str, bool file = false, bool full = false, string cp = "UTF-8")* — Загрузка XML из строки <*str*> или из файла с путём в <*str*> если <*file*> "true", с кодировкой <*cp*>. <*full*> — для полной загрузки, с блоками текста и комментариями в специальных узлах.
- *string save(int opt = 0, string path = "", string cp = "UTF-8")* — Сохранение дерева XML в строку или в файл <*path*> с параметрами форматирования <*opt*> и кодировкой <*cp*>. Возвращает текст XML или код ошибки. Предусмотрены следующие опции форматирования <*opt*>:
  - 0x01 — прерывать строку перед открывающим тегом;
  - 0x02 — прерывать строку после открывающего тега;
  - 0x04 — прерывать строку после закрывающего тега;
  - 0x08 — прерывать строку после текста;
  - 0x10 — прерывать строку после инструкции;
  - 0x1E — прерывать строку после всех;
  - 0x20 — вставлять стандартный XML-заголовок;
  - 0x40 — вставлять стандартный XHTML-заголовок.
- *XMLNodeObj getElementBy(string val, string attr = "id")* — получить элемент из дерева по атрибуту <*attr*> со значением <*val*>.

5.2. Система (SYS)

Функции объекта:

- *string system(string cmd, bool noPipe = false);* — осуществляет вызов консольных команд <*cmd*> ОС с возвратом результата по каналу. Если <*noPipe*> установлен то возвращается код возврата вызова и возможен запуск программ в фоне ("sleep 5 &"). Функция открывает широкие возможности пользователю OpenSCADA путём вызова любых системных программ, утилит и скриптов, а также получения посредством них доступа к огромному объёму системных данных. Например команда "ls -l" вернёт детализированное содержимое рабочей директории.
- *string fileRead(string file);* — Возвращает содержимое файла <*file*> в строке.
- *int fileWrite(string file, string str, bool append = false);* — Записывает строку <*str*> в файл <*file*>, удаляя присутствующий файл или добавляя в него <*append*>. Возвращает количество записанных байт.
- *int message(string cat, int level, string mess);* — формирование системного сообщения <*mess*> с категорией <*cat*>, уровнем <*level*> (-7...7). Отрицательное значение уровня формирует нарушения (Alarm).
- *int messDebug(string cat, string mess); int messInfo(string cat, string mess); int messNote(string cat, string mess); int messWarning(string cat, string mess); int messErr(string cat, string mess); int messCrit(string cat, string mess); int messAlert(string cat, string mess); int messEmerg(string cat, string mess);* — формирование системного сообщения <*mess*> с категорией <*cat*> и соответствующим уровнем.
- *XMLNodeObj XMLNode(string name = "");* — создание объекта узла XML с именем <*name*>.
- *string cntrReq(XMLNodeObj req, string stat = "");* — запрос интерфейса управления к системе посредством XML. Обычный запрос записывается в виде <get path="/OPath/%2felem"/>. При указании станции осуществляется запрос к внешней станции.

```
//Get the station identifier
req = SYS.XMLNode("get").setAttr("path","/%2fgen%2fid");
SYS.cntrReq(req);
idSt = req.text();
```

- *string sleep(int tm, int ntm = 0);* — усыпить поток исполнения на <*tm*> секунд и <*ntm*> наносекунд. Функция добавлено только для полноты и крайне не рекомендована к использованию, особенно в процедурах пользовательского интерфейса поскольку это приведет к блокированию интерфейса.

- *int time(int usec);* — возвращает абсолютное время в секундах от эпохи 1.1.1970 и микросекундах, если <*usec*> указан.
- *int localtime(int fullsec, int sec, int min, int hour, int mday, int month, int year, int wday, int yday, int isdst);* — возвращает полную дату в секундах (sec), минутах (min), часах (hour), днях месяца (mday), месяце (month), годе (year), днях недели (wday), днях в году (yday) и признак летнего времени (isdst), исходя из абсолютного времени в секундах <*fullsec*> от эпохи 1.1.1970.
- *string strftime(int sec, string form = "%Y-%m-%d %H:%M:%S");* — Преобразует абсолютное время <*sec*> в строку нужного формата <*form*>. Запись формата соответствует POSIX-функции strftime.
- *int strptime(string str, string form = "%Y-%m-%d %H:%M:%S");* — Возвращает время в секундах от эпохи 1.1.1970, исходя из строковой записи времени <*str*>, в соответствии с указанным шаблоном <*form*>. Например, шаблону "%Y-%m-%d %H:%M:%S" соответствует время "2006-08-08 11:21:55". Описание формата шаблона можно получить из документации на POSIX-функцию "strptime".
- *int cron(string cronreq, int base = 0);* — Возвращает время, спланированное в формате стандарта Cron <*cronreq*>, начиная от базового времени <*base*> или от текущего, если базовое не указано.
- *string strFromCharCode(int char1, int char2, int char3, ...);* — Создание строки из кодов символов char1, char2 ... charN.
- *string strCodeConv(string src, string fromCP, string toCP);* — Кодирование текста <*src*> из кодировки <*fromCP*> в <*toCP*>. Если кодировка опущена, то используется внутренняя.

5.3. Любой объект (TCntrNode) дерева OpenSCADA (SYS.*)

Функции объекта:

- *TArrayObj nodeList(string grp = "", string path = "");* — Получение списка дочерних узлов для группы <*grp*> и узла по пути <*path*>. Если <*grp*> пуста то возвращаются узлы всех групп.
- *TCntrNodeObj nodeAt(string path, string sep="");* — Подключение к узлу <*path*> в дереве объектов OpenSCADA. Если указывается разделитель в <*sep*> то путь обрабатывается как строка с разделителем.
- *TCntrNodeObj nodePrev();* — Получить предыдущий, родительский, узел.
- *string nodePath(string sep = "", bool from_root = true);* — Получение пути к текущему узлу, в дереве объектов OpenSCADA. Один символ разделителя указывается в <*sep*> для получения пути через разделитель, например, "DAQ.ModBus.PLC1.P1.var", иначе "/DAQ/ModBus/PLC1/P1/var". <*from_root*> указывает на необходимость формировать путь от корня и без указания идентификатора станции.

5.4. Подсистема "Безопасность" (SYS.Security)

Функции объекта подсистемы (SYS.Security):

- *int access(string user, int mode, string owner, string group, int access)* — Проверка для пользователя <*user*> доступа к ресурсу который принадлежит <*owner*> и <*group*> с доступом <*access*> для режим <*mode*>:
  user — пользователь для проверки доступа;
  mode — режим доступа (4-R, 2-W, 1-X);
  owner — владелец ресурса;
  group — группа ресурса;
  access — режим доступа к ресурсу (RWXRWXRWX — 0777).

Функции объекта пользователя (SYS.Security["usr_User"]) и группы (SYS.Security["grp_Group"]):

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.

5.5. Подсистема "БД" (SYS.BD)

Функции объекта БД (SYS.BD["TypeDB"]["DB"]):

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.
- *Array SQLReq(string req);* — Формирование SQL-запроса к БД.

```
DBTbl=SYS.BD.MySQL.GenDB.SQLReq("SELECT * from DB;");
for(var i_rw = 0; i_rw < DBTbl.length; i_rw++)
{
  var rec = "";
  for( var i_fld = 0; i_fld < DBTbl[i_rw].length; i_fld++ )
    rec += DBTbl[i_rw][i_fld]+"\t";
  SYS.messDebug("TEST DB","Row "+i_rw+": "+rec);
  //> Get column value by name
  if(i_rw) SYS.messDebug("TEST DB: ","Row "+i_rw+": 'NAME'"+DBTbl[i_rw]
["NAME"]);
}
```

Функции объекта Таблицы (SYS.BD["TypeDB"]["DB"]["Table"]):

- *XMLNodeObj fieldStruct();* — Получение структуры таблицы в виде XML узла "field" с дочерними узлами-колонками **<RowId type="real" len="10.2" key="1" def="Default value">{Value}</RowId>**, где:
  - {RowId} — идентификатор колонки;
  - {Value} — значение колонки;
  - type — тип значения колонки: *str* — строка, *int* — целое, *real* — вещественное и *bool* — логическое;
  - len — размер значения колонки, в знаках;
  - key — признак того, что колонка является ключом, и поиск осуществляется по его значению;
  - def — значение колонки по умолчанию.
- *string fieldSeek(int row, XMLNodeObj fld);* — Запрос поля <*row*> таблицы. Если поле получено то возвращается "1" иначе "0". В случае ошибки возвращается "0:Error".
- *string fieldGet(XMLNodeObj fld);* — Запрос значений поля. В случае ошибки возвращается "0:Error".

```
req = SYS.XMLNode("field");
req.childAdd("user").setAttr("type","str").setAttr("key","1").setText("roo
t");
req.childAdd("id").setAttr("type","str").setAttr("key","1").setText("/Lang
2CodeBase");
req.childAdd("val").setAttr("type","str");
SYS.BD.MySQL.GenDB.SYS.fieldGet(req);
SYS.messDebug("TEST DB","Value: "+req.childGet(2).text());
```

- *string fieldSet(XMLNodeObj fld);* — Установка поля. В случае ошибки возвращается "0:Error".
- *string fieldDel(XMLNodeObj fld);* — Удаление поля. В случае ошибки возвращается "0:Error".

5.6. Подсистема "Сбор данных" (SYS.DAQ)

Функции объекта подсистемы (SYS.DAQ):

- *bool funcCall(string progLang, TVarObj args, string prog);* — вызов текста функции <*prog*> с аргументами в объекте <*args*> для языка программирования <*progLang*>. Возвращает "true" при корректном вызове.

```
var args = new Object();
args.y = 0;
args.x = 123;
SYS.DAQ.funcCall("JavaLikeCalc.JavaScript",args,"y=2*x;");
SYS.messDebug("TEST Calc","TEST Calc rezult: "+args.y);
```

Функции объекта контроллера (SYS.DAQ["Modul"]["Controller"]):

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.
- *string name()* — имя контроллера.
- *string descr()* — описание контроллера.
- *string status()* — статус контроллера.
- *bool alarmSet(string mess, int lev = -5, string prm = "")* — установка/снятие нарушения <*mess*> с уровнем <*lev*> (отрицательный для установки иначе снятие), для параметра <*prm*>. Функция формирует нарушение с категорией: **al{ModId}:{CntrId}[.{PrmId}]**, где:
  - *ModId* — идентификатор модуля;
  - *CntrId* — идентификатор контроллера;
  - *PrmId* — идентификатор параметра, из аргумента <*prm*>.
- *bool enable(bool newSt = EVAL)* — получение состояния "Включен" или изменение его назначением атрибута <*newSt*>.
- *bool start(bool newSt = EVAL)* — получение состояния "Запущен" или изменение его назначением атрибута <*newSt*>.

Функции объекта параметра контроллера (SYS.DAQ["Modul"]["Controller"]["Parameter"]):

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.

Функции объекта атрибута параметра контроллера (SYS.DAQ["Modul"]["Controller"]["Parameter"]["Attribute"]):

- *ElTp get(int tm = 0, int utm = 0, bool sys = false)* — запрос значения атрибута на время <*tm:utm*> и признаком системного доступа <*sys*>.
- *bool set(ElTp val, int tm = 0, int utm = 0, bool sys = false)* — запись значения <*val*> в атрибут с меткой времени <*tm:utm*> и признаком системного доступа <*sys*>.
- *TCntrNodeObj arch()* — получение объекта архива, связанного с этим атрибутом. В случае отсутствия связанного архива возвращается "false".
- *string descr()* — описание атрибута.
- *int time(int utm)* — время последнего значения в секундах и микросекундах в <*utm*>.
- *int len()* — длина поля.
- *int dec()* — разрешение для вещественного.
- *int flg()* — флаги поля.
- *string def()* — значение по умолчанию.
- *string values()* — список допустимых значений или диапазон.
- *string selNames()* — список имён допустимых значений.
- *string reserve()* — резервное свойство значения.

Функции объекта библиотеки шаблона (SYS.DAQ[tmplb_Lib"]) и шаблона (SYS.DAQ[tmplb_Lib"]["Tmpl"]) параметра контроллера:

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.

5.6.1. Модуль DAQ.JavaLikeCalc

Объект "Библиотека функций" (SYS.DAQ.JavaLikeCalc["lib_Lfunc"])

- *ElTp {funcID}(ElTp prm1, ...)* — вызов функции библиотеки *{funcID}*. Возвращает результат вызываемой функции.

Объект "Пользовательская функция" (SYS.DAQ.JavaLikeCalc["lib_Lfunc"]["func"])

- *ElTp call(ElTp prm1, ...)* — вызов данной функции с параметрами <*prm{N}*>. Возвращает результат вызываемой функции.

5.6.2. Модуль DAQ.ModBus

Объект "Контроллер" (this.nodePrev())

- *string messIO(string pdu)* — отправка PDU <*pdu*> через транспорт объекта контроллера посредством ModBus протокола. PDU результата запроса помещается вместо запроса в <*pdu*>, а ошибка возвращается в результате функции.

5.7. Подсистема "Архивы" (SYS.Archive)

Функции объекта подсистемы:

- *Array messGet(int btm, int etm, string cat = "", int lev = 0, string arch = "");* — запрос системных сообщений за время от <*btm*> до <*etm*> для категории <*cat*>, уровня <*lev*> и архиватора <*arch*>. Возвращается массив объектов сообщений со свойствами:
  - *tm* — время сообщения, секунды;
  - *utm* — время сообщения, микросекунды;
  - *categ* — категория сообщения;
  - *level* — уровень сообщения;
  - *mess* — текст сообщения.
- *bool messPut(int tm, int utm, string cat, int lev, string mess);* — запись сообщения <*mess*> с категорией <*cat*>, уровнем <*lev*> (-7...7) и временем <*tm*>.<*utm*> в архив и/или список нарушений.

Функции объекта архиватора сообщений (SYS.Archive["mod_Modul"]["mess_Archivator"]):

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.
- *bool status()* — получение статуса архиватора.
- *int end()* — поучение времени окончания данных архиватора.
- *int begin()* — поучение времени начала данных архиватора.

Функции объекта архива (SYS.Archive["va_Archive"]) и архиватора значений (SYS.Archive["val_Modul"]["val_Archivator"]):

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.

5.8. Подсистема "Транспорты" (SYS.Transport)

Функции объекта входящего транспорта (SYS.Transport["Modul"]["in_Transp"]):

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.

Функции объекта исходящего транспорта (SYS.Transport["Modul"]["out_Transp"]):

- *ElTp cfg(string nm)* — получение значения конфигурационного поля <*nm*> объекта.
- *ElTp cfgSet(string nm, ElTp val)* — установка конфигурационного поля <*nm*> объекта в значение <*val*>.
- *string messIO(string mess, real timeOut = 0);* — отправка сообщения <*mess*> через транспорт с таймаутом ожидания <*timeOut*> (в секундах). В случае нулевого таймаута это время берётся из настроек исходящего транспорта.

```
rez=SYS.Transport.Serial.out_ttyUSB0.messIO(SYS.strFromCharCode(0x4B,0x00,
0x37,0x40),0.2);
while(true)
{
  trez = SYS.Transport.Serial.out_ttyUSB0.messIO("");
  if( !trez.length ) break;
  rez+=trez;
}
```

- *int messIO(XMLNodeObj req, string prt);* — отправка запроса <*req*> к протоколу <*prt*> для осуществления сеанса связи через транспорт посредством протокола.

```
req = SYS.XMLNode("TCP");
req.setAttr("id","test").setAttr("reqTm",500).setAttr("node",1).setAttr("re
qTry",2).setText(SYS.strFromCharCode(0x03,0x00,0x00,0x00,0x05));
SYS.Transport.Sockets.out_testModBus.messIO(req,"ModBus");
test = Special.FLibSYS.strDec4Bin(req.text());
```

5.9. Подсистема "Пользовательские интерфейсы" (SYS.UI)

5.9.1. Модуль UI.VCAEngine

Объект "Сеанс" (this.ownerSess())

- *string user( )* — текущий пользователь сеанса.
- *string alrmSndPlay( )* — путь виджета для которого на данный момент воспроизводится сообщение о нарушении.
- *int alrmQuittance(int quit_tmpl, string wpath = "")* — квитирование нарушений <*wpath*> с шаблоном <*quit_tmpl*>. Если <*wpath*> пустая строка то производится глобальное квитирование.

Объект "Виджет" (this)

- *TCntrNodeObj ownerSess( )* — получить объект сеанса данного виджета.
- *TCntrNodeObj ownerPage( )* — получить объект родительской страницы данного виджета.
- *TCntrNodeObj ownerWdg(bool base = false)* — получить родительский виджет данного виджета. При указании <*base*> будет возвращены и объекты страниц.
- *TCntrNodeObj wdgAdd(string wid, string wname, string parent)* — добавление виджета <*wid*> с именем <*wname*> на основе библиотечного виджета <*parent*>.

```
//Добавить новый виджет на основе виджета текстового примитива
nw = this.wdgAdd("nw", "Новый виджет", "/wlb_originals/wdg_Text");
nw.attrSet("geomX", 50).attrSet("geomY", 50);
```

- *bool wdgDel(string wid)* — удаление виджета <*wid*>.
- *TCntrNodeObj wdgAt(string wid, bool byPath = false)* — подключение к дочернему виджету или глобальному посредством пути <*byPath*>. В случае глобального подключения можно использовать абсолютный или относительный путь к виджету. Точкой отсчёта абсолютного адреса выступает объект корня модуля "VCAEngine", а значит первым элементом абсолютного адреса является идентификатор сеанса, который опускается. Относительный адрес берёт отсчёт от текущего виджета. Специальным элементом относительного адреса является элемент вышестоящего узла "..".
- *bool attrPresent(string attr)* — проверка атрибута виджета <*attr*> на факт присутствия.
- *ElTp attr(string attr)* — получение значения атрибута виджета <*attr*>. Для отсутствующих атрибутов возвращается пустая строка.
- *TCntrNodeObj attrSet(string attr, ElTp vl)* — установка атрибута виджета <*attr*> в значение <*vl*>. Возвращается текущий объект для конкатенации функций установки.
- *string link(string attr, bool prm = false)* — получение ссылки для атрибута виджета <*attr*>. При установке <*prm*> запрашивается ссылка для группы атрибутов (параметр), представленных указанным атрибутом.
- *string linkSet(string attr, string vl, bool prm)* — установка ссылки для атрибута виджета <*attr*>. При установке <*prm*> осуществляется установка ссылка для группы атрибутов (параметр), представленных указанным атрибутом.

```
//Установить ссылку восьмого тренда параметром
this.linkSet("el8.name", "prm:/LogicLev/experiment/Pi", true);
```

Объект "Виджет", примитива "Документ" (this)

- *string getArhDoc(integer nDoc)* — получить текст документа архива на глубине <*nDoc*> (0-{aSize-1}).

5.10. Подсистема "Специальные" (SYS.Special)

5.10.1. Модуль Special.FLibSYS

Объект "Библиотека функций" (SYS.Special.FLibSYS)

- *ElTp {funcID}(ElTp prm1, ...)* — вызов функции библиотеки *{funcID}*. Возвращает результат вызываемой функции.

Объект "Пользовательская функция" (SYS.Special.FLibSYS["funcID"])

- *ElTp call(ElTp prm1, ...)* — вызов данной функции с параметрами <*prm{N}*>. Возвращает результат вызываемой функции.

5.10.2. Модуль Special.FLibMath

Объект "Библиотека функций" (SYS.Special.FLibMath)

- *ElTp {funcID}(ElTp prm1, ...)* — вызов функции библиотеки *{funcID}*. Возвращает результат вызываемой функции.

Объект "Пользовательская функция" (SYS.Special.FLibMath["funcID"])

- *ElTp call(ElTp prm1, ...)* — вызов данной функции с параметрами <*prm{N}*>. Возвращает результат вызываемой функции.

5.10.3. Модуль Special.FLibComplex1

Объект "Библиотека функций" (SYS.Special.FLibComplex1)

- *ElTp {funcID}(ElTp prm1, ...)* — вызов функции библиотеки *{funcID}*. Возвращает результат вызываемой функции.

Объект "Пользовательская функция" (SYS.Special.FLibComplex1["funcID"])

- *ElTp call(ElTp prm1, ...)* — вызов данной функции с параметрами <*prm{N}*>. Возвращает результат вызываемой функции.