СК11.Platform

версия: 11.6.4.
редакция: 7280
дата печати: март, 2022

Программный комплекс СК-11

"Программный комплекс СК-11" – это общее название информационно-технической платформы с изменяемым набором приложений для создания автоматизированных систем оперативно-диспетчерского, технологического и ситуационного управления объектами электроэнергетики. Состав приложений зависит от круга задач, решаемых центром управления, и может меняться в процессе эксплуатации.

Приложения работают с использованием интеграционной серверной платформы СК-11 под управлением ОС Astra Linux с использованием встроенной СУБД PostgreSQL.

В настоящем томе приведено описание приложения "Платформа СК-11" – программа для ЭВМ "СК11.Platform".

СК-11

Авторские, имущественные права и общие положения по использованию документа

Настоящий документ пересматривается на регулярной основе с внесением всех необходимых исправлений и дополнений в следующие выпуски.
Предприняты все меры для того, чтобы содержащаяся здесь информация была максимально актуальной и точной, тем не менее, компания Монитор Электрик не несёт ответственности за ошибки или упущения, а также за любой ущерб, причинённый в результате использования содержащейся здесь информации.
О технических неточностях или опечатках вы можете сообщить в Службу технической поддержки Монитор Электрик. Мы будем рады вашим замечаниям и предложениям.
Содержание данного документа может быть изменено без предварительного уведомления. Перед использованием убедитесь, что это актуальная версия, соответствующая версии использующейся системы. Для получения актуальной версии вы можете обратиться по адресам, указанным на сайте www.monitel.ru.

Названия продуктов и компаний, упомянутые здесь, могут являться торговыми марками соответствующих владельцев.
Продукция, для которой разработана настоящая документация (документ) является сложным прикладным программным обеспечением, которое далее будет именоваться «Программный продукт».
Компания Монитор Электрик оставляет за собой право внесения любых изменений в настоящую документацию.

Гарантия

Компания Монитор Электрик гарантирует устранение выявленных в Программном продукте дефектов. Исправленные версии Программного продукта предоставляются в виде обновления.
Дефектом признаётся отклонение функциональности Программного продукта от соответствующего описания, приведённого в настоящей документации, препятствующее нормальной эксплуатации Программного продукта, при условии соблюдения требований к организации эксплуатации, приведённых в настоящей документации. Допускается несущественное различие фактической функциональности Программного продукта и описания, приведённого в настоящей документации, при условии, что это не влияет значимым образом на процесс эксплуатации.

Правила безопасной эксплуатации и ограничение ответственности

Программный продукт функционирует в составе системы, включающей помимо самого Программного продукта компьютерное аппаратное обеспечение, системное и специальное программное обеспечение, сегменты вычислительной сети – далее совместно именуемые инфраструктурой. Современная инфраструктура, в которой функционирует Программный продукт, включает сложное аппаратное и программное обеспечение, которое может модернизироваться и обновляться независимо от Программного продукта. Поэтому для безопасной и бесперебойной эксплуатации Программного продукта перед вводом его в постоянную эксплуатацию должна быть разработана эксплуатационная документация на систему в целом. Настоящий документ предназначен для облегчения пользователю (эксплуатирующей организации) задачи разработки собственной эксплуатационной документации на систему.

Для повышения безопасности и бесперебойности эксплуатации систем на базе Программного продукта необходимо выполнять следующие основные требования по организации эксплуатации (другие требования и рекомендации могут содержаться в соответствующих разделах документа):

- Реализация и эксплуатация автоматизированных систем, в составе которых функционирует Программный продукт, должны осуществляться на основе проектной документации, при разработке которой проработаны и согласованы с эксплуатирующей организацией все вопросы совместимости и интеграции компонентов, включая Программный продукт.
- Эксплуатация Программного продукта должна проводиться в соответствии с эксплуатационной документацией эксплуатирующей организации, а также рекомендациями Службы технической поддержки Монитор Электрик.

• В эксплуатационной документации должен быть описан механизм взаимодействия специалистов эксплуатирующей организации (администраторы, пользователи) со Службой технической поддержки Монитор Электрик, включая регламент выполнения рекомендаций и подготовки ответов на запросы дополнительной информации Службы технической поддержки Монитор Электрик в ходе штатной эксплуатации и устранения нарушений в работе Программного продукта.
• Запрещено использование нештатных средств, не входящих в состав Программного продукта или не описанных в эксплуатационной документации, в том числе инструментов для внесения изменений в базы данных Программного продукта.
• Аппаратное обеспечение, системное программное обеспечение, внешнее программное обеспечение, взаимодействующее с Программным продуктом или работающее на общей с ним аппаратной платформе, а также другая ИТ-инфраструктура, обеспечивающая работу Программного продукта, должны быть совместимы с эксплуатируемой версией Программного продукта и функционировать без сбоев.
• В соответствии с эксплуатационной документацией и внутренними регламентами эксплуатирующей организации, с определённой периодичностью должны выполняться следующие профилактические мероприятия:

– перезагрузка серверов и клиентских рабочих станций, на которых установлен Программный продукт;
– установка критически важных обновлений системного программного обеспечения, внешнего программного обеспечения, взаимодействующего с Программным продуктом или работающего на общей с ним аппаратной платформе;
–обновление антивирусных БД на серверах и клиентских рабочих станциях, на которых установлен Программный продукт;
– проверка и обеспечение достаточности аппаратных ресурсов;
– проверка журналов операционной системы и Программного продукта на наличие записей об ошибках и устранение причин их возникновения;
– мониторинг корректной работы сетевого оборудования ЛВС, которое участвует в обмене данными между компонентами Программного продукта, а также между Программным продуктом и внешними системами.

• Регламент (периодичность, условия) выполнения профилактических мероприятий определяется эксплуатирующей организацией самостоятельно в зависимости от условий эксплуатации с учётом рекомендаций, приведённых в настоящей документации, и рекомендаций Службы технической поддержки Монитор Электрик при их наличии.
• При использовании Программного продукта для выполнения важных операций, которые могут привести к возникновению значительных убытков или связаны с рисками для жизни и здоровья людей, пользователь Программного продукта должен убедиться в том, что Программный продукт и инфраструктура функционируют в штатном режиме, без сбоев, а после завершения операции – убедиться в том, что она выполнена корректно.
• Все значимые для обеспечения безопасной эксплуатации Программного продукта регламентные операции и профилактические мероприятия, а также факты проверки готовности системы к выполнению важных операций и факты успешного выполнения важных операций должны фиксироваться в оперативном журнале эксплуатации или подтверждаться другим надёжным способом – на усмотрение эксплуатирующей организации. Эксплуатирующая организация должна предоставлять копии и выписки из оперативного журнала эксплуатации по запросу Службы технической поддержки Монитор Электрик.

Компания Монитор Электрик не несёт ответственности за упущенную экономическую выгоду, убытки или претензии третьих лиц, включая любые прямые, косвенные, случайные, специальные, типичные или вытекающие убытки (включая, но не ограничиваясь, утрату возможности использования, потерю данных или прибыли, прекращение деятельности), произошедшие при любой схеме ответственности, возникшие вследствие использования или невозможности использования Программного продукта, даже если о возможности такого ущерба было заявлено.

1. Платформа СК-11

В основу **СК-11** положена общая информационная модель (CIM), выполненная в соответствии со стандартами серий IEC 61970 и 61968.

Связующим звеном является открытая платформа, которая поддерживает широкий набор международных стандартов интеграции и обеспечивает возможность простого подключения и адаптации к Системе комплектов приложений различных производителей.

Сервис-ориентированная архитектура позволяет организовать размещение сервисов по виртуальным узлам и организовать взаимодействие с компонентами системы через интеграционные шины.

Компонентами системы являются программные и технические средства, входящие в состав комплекса.

Платформу **СК-11** составляют следующие подсистемы:

- БДРВ;
- Слой доступа к данным (MAL);
- Слой доступа к журнальным данным (JAL);
- Система управления записью журналов работы модулей;
- Компонент адаптации скриптов;
- Административные блокировки.

В состав информационно-технологической платформы СК-11 входят следующие приложения, реализующие пользовательский доступ к данным:

- Редактор модели – предназначен для наполнения, ведения и работы с данными системы;
- TNA – предназначен для проведения расчёта режимов и анализа состояния электроэнергетических систем;
- MAG Terminal – предназначен для контроля и управления оперативной информацией SCADA СК-11.

Платформа СК-11 позволяет осуществлять удалённый запуск ресурсоёмких клиентских приложений на сервере с использованием технологии *RemoteApp*.

С целью аудита действий пользователей платформа СК-11 содержит журнал безопасности.

Для организации централизованного и унифицированного протоколирования работы Системы используются журналы работы.

1.1. Управление приложениями и сервисами

Сложная автоматизированная система, компоненты которой размещены на различных вычислительных узлах в сети, нуждается в инструменте дистанционного централизованного управления и контроля, позволяющем: осуществлять мониторинг аппаратных и программных компонентов, конфигурировать их, запускать/останавливать сервисы и др. Указанные функции сбора информации и управления системой выполняет **Служба управления задачами СК-11** (`СК-11 Supervisor`) – сервис управления (Супервизор – `Supervisor.exe`), конфигурации и контейнер приложений, которым комплектуются все вычислительные узлы. Пользователи, обладающие необходимым уровнем доступа, осуществляют все вышеперечисленные функции посредством программы "Управление узлами СК" или посредством командлетов PowerShell на платформе *Linux*.

Конфигурация домена СК-11 хранится централизовано в БД "Конфигурация системы". В домене СК-11 описываются все вычислительные узлы, контролируемые ими вычислительные ресурсы, клиентские машины, входящие в домен с указанием их роли.

При старте "Служба управления задачами СК-11" читает из БД "Конфигурация системы" (*SysConfig*) конфигурацию домена, определяет роль своего вычислительного узла и, исходя из неё, применяет конфигурацию: разворачивает и инициализирует описанные в ней вычислительные ресурсы. Служба управления задачами СК-11 автоматически применяет новую конфигурацию в случае её изменения или смены ролей.

Все экземпляры "Службы управления задачами СК-11" в домене постоянно обмениваются состоянием вычислительных узлов между собой, запущенными задачами, предоставляемыми сервисами и т.п. Обмениваются по принципу "каждый с каждым" для поддержания максимальной полноты информации в случае выхода из строя участков локальной корпоративной сети. При вводе в работу нового вычислительного узла или ресурса, а также при отключении или изменении конфигурации, информация автоматически распространяется между супервизорами. Таким образом, подключившись к любой "Службе управления задачами СК-11", можно получить информацию об актуальном состоянии и роли (основной/резервный/недоступный) всех вычислительных узлов и ресурсов домена. Подключившись к конкретной "Службе управления задачами СК-11" с помощью программы "Управления узлами СК" или консольных команд, можно совершить управляющие воздействия согласно правам доступа.

"Служба управления задачами СК-11" предоставляет данные о состоянии узла и по протоколам SNMPv1, SNMPv2C. Можно получить информацию:

- о типе и состоянии вычислительного узла;
- о состоянии запущенных программных ресурсов.

Обеспечение надёжности вычислительных ресурсов

Программные вычислительные ресурсы и серверы, на которых они размещены, могут быть объединены в группы резервирования: Горячего резерва. Групп может быть несколько. Сервер приложений может входить в несколько Групп горячего резерва.

Группа горячего резерва определяет роли серверов (Основной/Резервный), т.е. в группу такого типа могут входить не более двух Серверов приложений.

Для каждой роли в Конфигурации системы определяется перечень программных вычислительных ресурсов. При старте Супервизор сервера, входящего в Группу горячего резерва, вычисляет роль Сервера приложений и соответствующие ей вычислительные ресурсы, которые нужно ввести в работу.

Вычислительный ресурс в Группе горячего резерва может быть привязан как к Основной роли, так и к Резервной или к обоим одновременно (если позволяет его реализация).

Автоматическая реакция на сбои и реконфигурация серверов приложений

Для обеспечения надёжной работы системы с минимальным вмешательством персонала в **Службу управления задачами СК-11** включена функция автоматической реконфигурации ролей вычислительных узлов при различных сбоях. Реконфигурация влечёт остановку одних размещённых на узле ресурсов и запуск других для полного соответствия узла новой роли.

Основной алгоритм базируется на триггерах, которые инициируют процесс реконфигурации. Такими триггерами являются:

- команда оператора на изменение ролей серверов или групп;
- выход из строя или отключение основного сервера в группе;
- разрыв сетевого соединения с основным сервером в группе;
- сбой в работе программных ресурсов, отмеченных в конфигурации как критические.

Служба управления задачами СК-11 по разному обрабатывает вышеперечисленные ситуации. Выполнение того или иного алгоритма зависит как от самого триггера, так и от роли узла на момент наступления события.

В случае срабатывания триггера на изменение роли узла отказоустойчивой группы ресурсов система опирается на следующие правила:

- Группа не должна остаться без Основного узла (master). Если нет другого узла, который мог бы стать основным, то если триггером была ручная команда оператора, она будет отменена. Если триггером являлся сбой узла, то

выполняется пользовательская команда, описанная в конфигурации комплекса СК-11 (например, перезагрузка сервера).

- Резервный узел (slave) становится Основным (master) в случае сбоя Основного или в результате команды оператора.
- Сбой резервного и запасного узлов не приводят к изменению ролей остальных узлов.
- Узел из состояния "Выведен из работы" в другое состояние может перейти только по ручной команде оператора.
- В группе не может быть двух и более узлов с одинаковой ролью. Таким образом, если триггером была ручная команда оператора, то после успешного её исполнения предыдущий исполнитель этой роли понижает свою роль до Резервного (slave) или выводится из работы.

В серверной службе управления задачами СК-11 алгоритм разрешения конфликтов ролей в группе горячего резерва предусматривает, чтобы Основная роль оставалась за сервером, который был назначен им раньше. Для Резервного сервера в группе горячего резерва имеется настраиваемая задержка ожидания восстановления связи с Основным сервером в группе перед повышением роли. Значение задержки по умолчанию – 30 сек., параметр **`RoleAssigmentTimeout`** в конфигурационном файле `Supervisor.config.json`.

В группе горячего резерва "Основная группа" экземпляров серверных ресурсов Супервизор вслед за ролью Основной (master) устанавливает на основном сетевом интерфейсе дополнительный IP-адрес основного узла, для доступа веб-сервисов работающих в SCADA-контуре. Для сетевого интерфейса на резервном узле (slave) дополнительный IP-адрес убирается.

Диагностика состояния вычислительных ресурсов и серверов

Супервизор контролирует состояние программных вычислительных ресурсов на соответствие регламенту запуска, потребления оперативной памяти, дескрипторов ОС, свободного дискового пространства и т.п.

Состояние баз данных контролируется регулярной проверкой доступа, а именно непосредственным подключением к каждой БД. В случае резервирования баз данных путём включения их в группу доступности AlwaysOn, опрашивается состояние синхронизации и доступность базы на каждой реплике SQL-сервера.

Супервизор предоставляет программный интерфейс взаимодействия для того, чтобы серверные приложения сами могли формировать диагностические сообщения, т.е. сигнализировать о наступлении важных событий, информировать о внутренних ошибках, блокирующих состояния, и т.п.

Состояние серверов и компонентов предоставляется внешним системам посредством взаимодействия по протоколу SNMP.

Для регулирования уровня подробности журналов работы серверных приложений СК-11 имеется настраиваемый параметр **`LogerControl`** в конфигурационном файле `Supervisor.config.json`:

- `LogerControl:MaxLogPriority` – определяет максимальный уровень `LogPriority`,
- `LogerControl:MinLogPriority` – определяет минимальный уровень `LogPriority`.

По умолчанию аварийный предел заполнения дискового пространства с журналами – 1 Гб. При достижении установленного значения заполнения дискового пространства происходит снижение уровня подробности ведения журналов.

Управление веб-сервисами и Службой управления задачами СК-11 на выделенных серверах

На платформе Linux управление общим адресов, для выделенных серверов веб-сервисов СК-11, осуществляется системным сервисом *keepalived*.

1.2. БДРВ

База данных реального времени (**БДРВ**) **–** играет роль центрального звена, предоставляющего пользователям, в том числе программным модулям, входящим в состав комплекса, услуги высокопроизводительной обработки запросов доступа к данным реального времени и архивам, включая различные варианты генерации и подписки на изменения с применением фильтров.

БДРВ выполняет роль обработчика запросов пользователей на запись и чтение информации. Обслуживание запросов реального времени БДРВ производит с максимальной скоростью за счёт того, что все актуальные данные она хранит в оперативной памяти. Для обслуживания запросов к историческим данным (относящимся к периодам, отстоящим от текущего момента вперёд или назад по времени) БДРВ использует компонент HIS. Все актуальные данные, которые предполагают сохранение, также передаются от БДРВ в HIS с тем, чтобы попасть в Архив данных.

1.3. Слой доступа к данным (MAL)

Model Access Layer (MAL) – слой доступа к данным. Это центральный компонент, обеспечивающий связь между приложениями и данными. Фактически выступает в роли единого интерфейса доступа к данным из различных приложений.

Для постоянного хранения данных **MAL** использует ObjectDB – объектно-ориентированное хранилище данных, используемое для хранения информационной модели и её объектов. Используется для хранения информации о различных элементах энергосистемы (CIM), а также иных объектов (не CIM), необходимых для обеспечения функциональности системы (к примеру: конфигурации вычислительных узлов и приложений, прав доступа и различных настроек). Хранилище **ObjectDB** поддерживает дерево версий модели, которые наследуются друг от друга. С точки зрения конечного пользователя каждая **версия модели –** это полноценная, самодостаточная модель, являющаяся потомком или предком для другой или других версий моделей.

Контекст – это загруженная в оперативную память MAL из ObjectDB версия модели и совокупность работающих с ней компонентов системы. Данные контекста могут изменяться приложениями. Изменения, внесённые одним из приложений, становятся доступны всем приложениям, работающим с этим контекстом. Контексты делятся на два типа: привязанные к источнику и отключённые от источника. Тип контекста определяется при его создании. Изменения, вносимые в привязанный контекст, сохраняются в источник данных. Изменения, вносимые в отключённый контекст, остаются в памяти и пропадают при его закрытии. Контекст не имеет доступа к данным другого контекста. Таким образом, все контексты изолированы друг от друга.

В состав менеджера контекста входит сервис (Overview Service), предназначенный для получения и сохранения обзорной информации о работе серверных модулей для дальнейшего распространения её подключённым с клиентских рабочих мест программам для отображения в пользовательском интерфейсе (Обзорные формы состояния расчётно-аналитических приложений).

Контекст может быть запущен или остановлен по запросу пользователя. Исключением является Контекст реального времени.

Контекст исследовательский – это контекст, используемый для работы в режиме offline или в режиме исследования, анализа или моделирования. Он предназначен для запуска расчётных задач и приложений.

Актуальная версия модели – версия модели, хранимая в базе и помеченная как актуальная.

Версия модели – представляет собой именованное состояние данных в базе. Версия модели может формально не соответствовать требованиям к хранимой в базе информации.

Актуализация версии модели – последовательность операций, которые должна пройти версия модели для перехода в состояние "Актуальная версия модели".

1.4. Слой доступа к журнальным данным (JAL)

Journal Access Layer (JAL) – слой доступа к журнальным данным. Компонент, обеспечивающий связь между приложениями и журнальными данными Системы. Журнальные данные представляют собой записи экземпляров журнальных классов, описанных в канонической модели. Журнальные классы – это классы модели, описывающие разного рода события, происходящие в Системе, и связанную с ними информацию. Семантика журнальных классов определяется приложениями, использующими их.

Для постоянного хранения журнальных данных используется **Журнальная база данных (БД JournalDB, JDB**) — объектно-ориентированная база данных. Структура JDB определяется журнальными метаданными (JAL metadata), генерируемыми на основе модели журнальных данных, создаваемой в *Enterprise Architect* (EAP).

Совокупность журнальных записей и метаданных экземпляра БД JournalDB образует **Контекст журнальных данных**.

Контексты журнальных данных, используемые в Системе по умолчанию:

- контекст журнальных данных для БД журналов энергосистемы в зависимости от подсистемы;
- `AppData` – контекст журнальных данных для БД журнальных данных приложений;
- `Switching` – контекст журнальных данных БД журналов программ переключений.

Серверные приложения JAL:

- **Служба журнальных данных** (**JalService.dll**) — серверное приложение, выполняющее запросы к JDB. Связь серверного приложения "Служба журнальных данных" с клиентами осуществляется с помощью платформы обмена сообщениями RabbitMQ. Один экземпляр серверного ресурса приложения работает только с одним контекстом журнальных данных (JDB). Количество экземпляров серверного приложения "Служба журнальных данных", запущенных на одном или нескольких серверах, не ограничено;
- **Служба очистки устаревших журнальных данных** (**JalGarbageCollector.dll**) — серверное приложение, выполняющее очистку журналов от устаревших записей по заданным правилам. Правила очистки размещаются в информационной модели "Данные приложений" и в специализированных подключаемых модулях, обеспечивая удаление журнальных записей по сложным правилам с учётом внешних данных. Возможна однократная очистка журналов от устаревших записей с использованием специального набора правил, представленного в файле формата `JSON`. Один экземпляр серверного ресурса обслуживает все контексты журнальных данных домена СК-11;
- **Синхронизация JournalDB** (**JalSync.dll**) — серверное приложение, применяемое для синхронизации журнальных данных между доменами СК-11. Один экземпляр серверного ресурса обслуживает один контекст журнальных данных (JDB)

синхронизируемого домена. Для каждого синхронизируемого контекста (JDB) требуется создавать и настраивать отдельный экземпляр серверного ресурса серверного приложения "Синхронизация JournalDB".

1.5. Обеспечение отказоустойчивости для СУБД PostgreSQL

Обеспечение отказоустойчивости, функции высокой доступности (High Availability), для СУБД *PostgreSQL* осуществляется при помощи совокупности следующих средств:

- Наличие двух серверов (нод) с *PostgreSQL*;
- Виртуальный IP (VIP) для клиентских подключений к *PostgreSQL*;
- Синхронизация (репликация) данных между нодами *PostgreSQL*;
- Наличие третьей, голосующей ноды (whitness) для обеспечения кворума;
- Стек Pacemaker\Corosync, выполняющий автоматическую обработку отказа (auto-failover).

Далее подробно рассмотрены перечисленные средства, а так же сценарий восстановления ноды *PostgreSQL* после сбоя.

Наличие двух серверов с PostgreSQL

Отказоустойчивость *PostgreSQL* обеспечивается наличием двух серверов СУБД, на каждом из которых хранится актуальная полная копия данных.

Один из серверов является основным (master). Именно с ним работают клиенты. Второй сервер (slave), реплика, работает в режиме горячего резерва, синхронно получая от основного (master) все изменения данных.

Поскольку синхронизация изменений происходит в режиме on-line, то в любой момент времени существует "горячий" резервный сервер, который хранит актуальную копию данных и готов, в случае сбоя основного, принять на себя его роль и обслуживать запросы клиентов.

Виртуальный IP (VIP) для клиентских подключений к PostgreSQL

Чтобы избавить клиентов от необходимости переподключаться к той или иной ноде *PostgreSQL* в зависимости от её роли в кластере, существует виртуальный IP-адрес, который мигрирует вслед за ролью основного (master). Клиентские программы получают только этот адрес и используют только его для подключения к PostgreSQL.

Репликация данных между нодами PostgreSQL

Синхронизация данных осуществляется при помощи потоковой репликации, встроенной в *PostgreSQL*, начиная с 9-ой версии. Механизм потоковой репликации основан на журнале предзаписи (WAL).

Write-ahead log (WAL)

Каждый сервер СУБД *PostgreSQL* пишет все изменения сначала в WAL (write-ahead log) – журнал предварительной записи транзакций, и только затем применяет эти изменения к данным в базе. Это позволяет гарантировать целостность данных и отсутствие конфликтов изменений в табличном пространстве.

В случае, если сервер по какой-то причине перезагрузился, сначала проверяет текущий номер транзакции, примененный к табличному пространству (успешно завершенная запись). Затем сервер проверят WAL, и дописывает разницу из WAL в tablespace. Номер транзакции всегда растет монотонно, что исключает конфликты очередности применения. В WAL в основном идёт только запись и она всегда последовательная (чтение бывает лишь эпизодически и тоже последовательное).

Механизм репликации

При репликации осуществляется трансляция потока журнальных записей от основной ноды (master) на резервную ноду (slave). На реплику передаются изменения в виде записей WAL. Это очень эффективный механизм, но он требует, чтобы между серверами была двоичная совместимость (основная версия сервера, операционная система, аппаратная платформа).

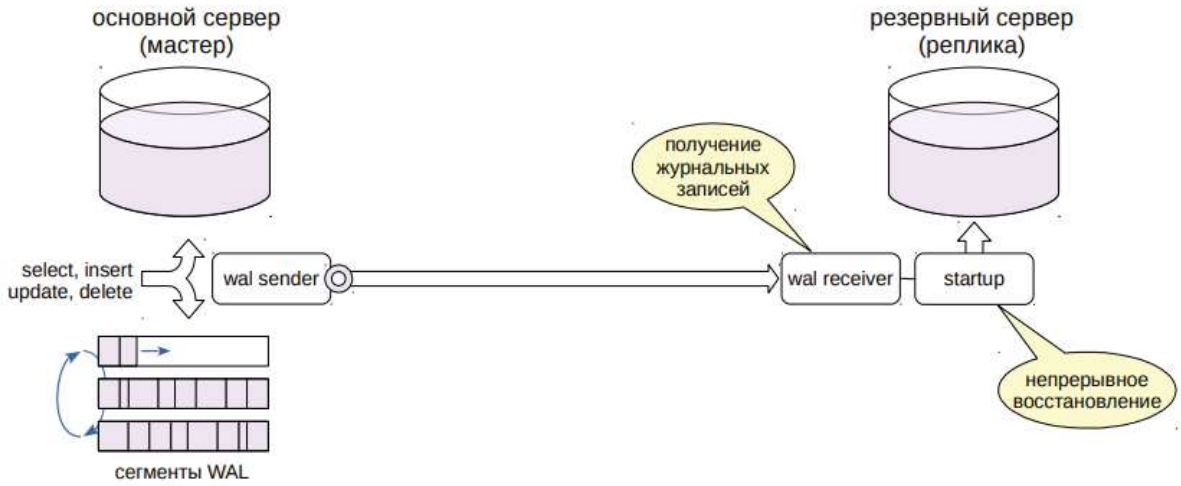


Схема механизма репликации

Настройка репликации сходна настройке физического резервного копирования. Отличие в том, что резервная копия восстанавливается сразу, не дожидаясь поломки основного сервера, работая в режиме постоянного восстановления (standby_mode = on). Постоянно производится чтение и применение новых сегментов WAL, приходящих с основной ноды. Таким образом, реплика постоянно поддерживается в актуальном состоянии. Резервная нода всегда готова стать основной при сбое.

Общий сценарий настройки репликации

1. Создаётся базовая резервная копия (pg_basebackup) на основной ноде (master);
2. Базовая резервная копия разворачивается на резервной ноде (slave);

3. На резервной ноде создаётся управляющий файл `recovery.conf` (`standby_mode = on`);
4. Резервная нода (slave) запускается.

Далее резервная нода (slave) восстанавливает согласованность и продолжает применять поступающие журналы. Доставка — поток WAL по протоколу репликации.

Стек Pacemaker\Corosync, выполняющий автоматическую обработку отказа (auto-failover)

PostgreSQL не предоставляет системное программное обеспечение, необходимое для выявления сбоя на первичном сервере и уведомления резервного сервера СУБД. В качестве такого ПО используется стэк инструментов *Pacemaker\Corosync*.

Pacemaker манипулирует узлами и ресурсами, а именно, выполнять запуск/остановку ресурсов на узлах. Экземпляры *PostgreSQL*, в терминах *Pacemaker*, – это ресурсы, которые расположены на двух узлах и управление которыми программируется по таким правилам, чтобы всегда была основная нода (master) и она была только одна.

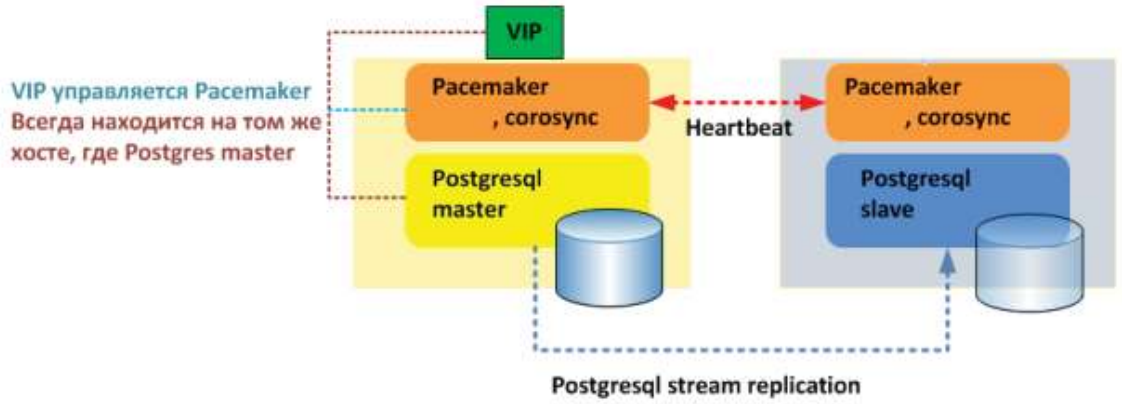


Схема работы ПО Pacemaker\Corosync

Именно *Pacemaker* отвечает за auto-failover: промоутинг (promote) реплики, когда основная нода (master) не отвечает. Представить себе это процесс можно так: *Pacemaker* постоянно опрашивает (ping) основную ноду (master) с целью убедиться в её жизнеспособности. Как только ответ на опрос не приходит на протяжении заданного таймаута, *Pacemaker* считает, что основная нода (master) отсутствует. Он отправляет на резервную ноду (slave) команду стать основной, чтобы соблюсти наличие ресурса *PostgreSQL* в сети. Если позднее бывшая основная нода (master) начинает отвечать, *Pacemaker* обеспечивает понижение её роли (demote) до резервной (slave) либо остановку, тем самым избегая ситуации с двумя основными нодами одновременно – Split brain.

Split-brain и голосующая нода (whitness)

Чтобы успешно обрабатывать сетевые разрывы, чётного количества нод недостаточно. Нужна третья голосующая нода. Рассмотрим следующий сценарий: произошел сетевой разрыв таким образом, что сервера *PostgreSQL* оказались в разных частях сети. В этом

случае для каждой ноды это будет выглядеть так, как будто партнер пропал. При этих условиях *Pacemaker* переведет свою ноду в роль основной (master).

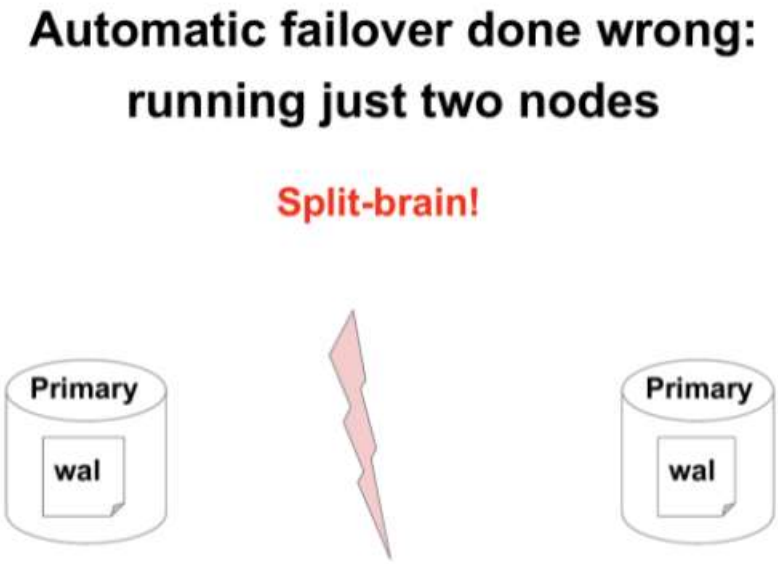


Схема ситуации Split-brain

Если при этом клиенты тоже разделились между частями сети, то данные будут изменены в обоих основных серверах и линии времени разойдутся. Эта ситуация и называется split brain, она неразрешима. Потеря данных неизбежна. Для предотвращения данного сценария, вводится третья нода. *PostgreSQL* на ней нет, она только голосующая. Правила *Pacemaker* настраиваются таким образом, что для кворума необходимы два голоса. Только при кворуме можно переключать роль сервера. Это позволяет избежать Split brain, поскольку при сетевом разрыве либо две ноды из трёх оказываются в одной части сети и тогда они примут решение перевести свой экземпляр *PostgreSQL* в роль основного. Либо, если все три ноды окажутся в разных частях сети, то основным не станет ни одна нода, поскольку ни у кого нет кворума.

В качестве этой третьей голосующей ноды используется файловая папка, расположенная на одном из серверов-приложений (поскольку сервера приложений всегда отделены от серверов БД).

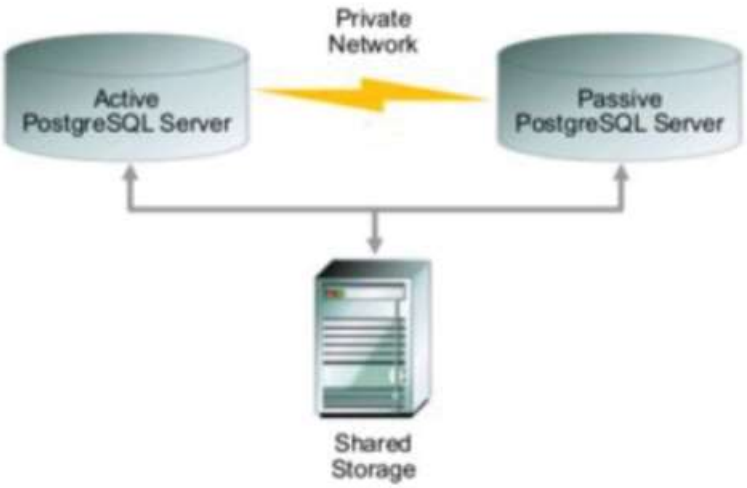


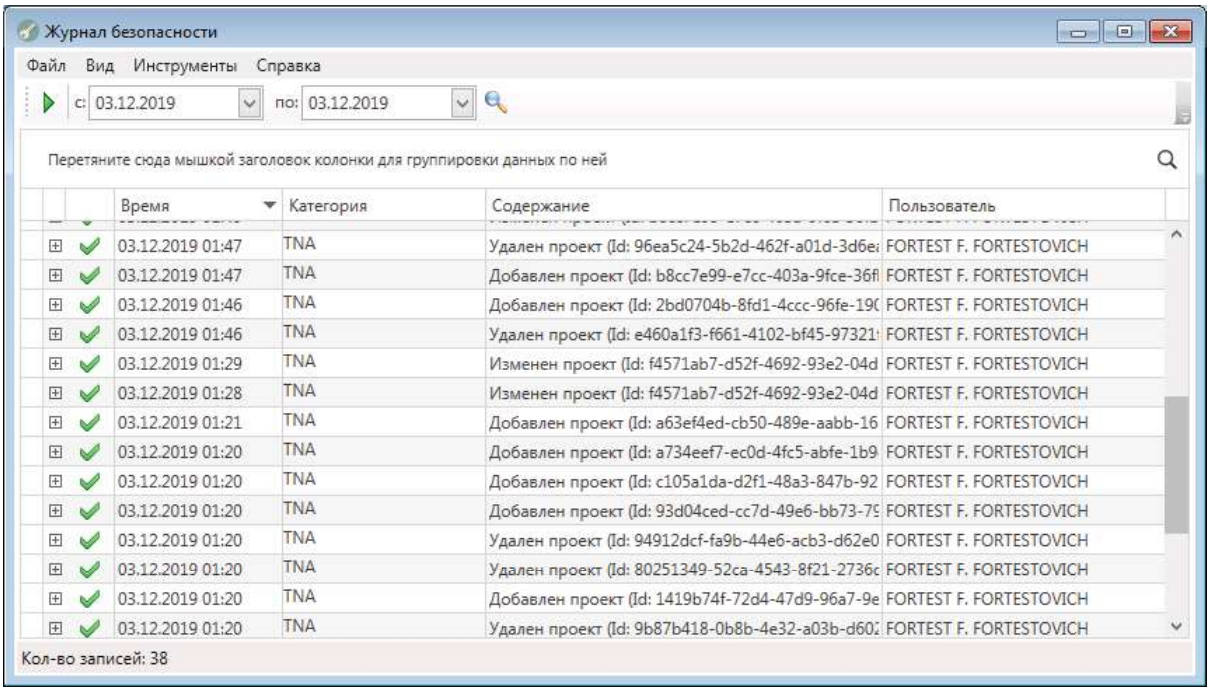
Схема применения голосующей ноды (whitness)

1.6. Журнал безопасности

При работе пользователей с платформой СК-11 ведётся журнал безопасности. Сведения записей журнала безопасности предназначены для аудита действий пользователей, которые важны с точки зрения информационной безопасности.

Фиксация событий безопасности в журнале осуществляется автоматически по категориям действий серверным приложением "Сервис журнала безопасности" (SecurityJournalService.dll).

Клиентское приложение **Журнал безопасности** предназначено для просмотра записей журнала безопасности Системы.

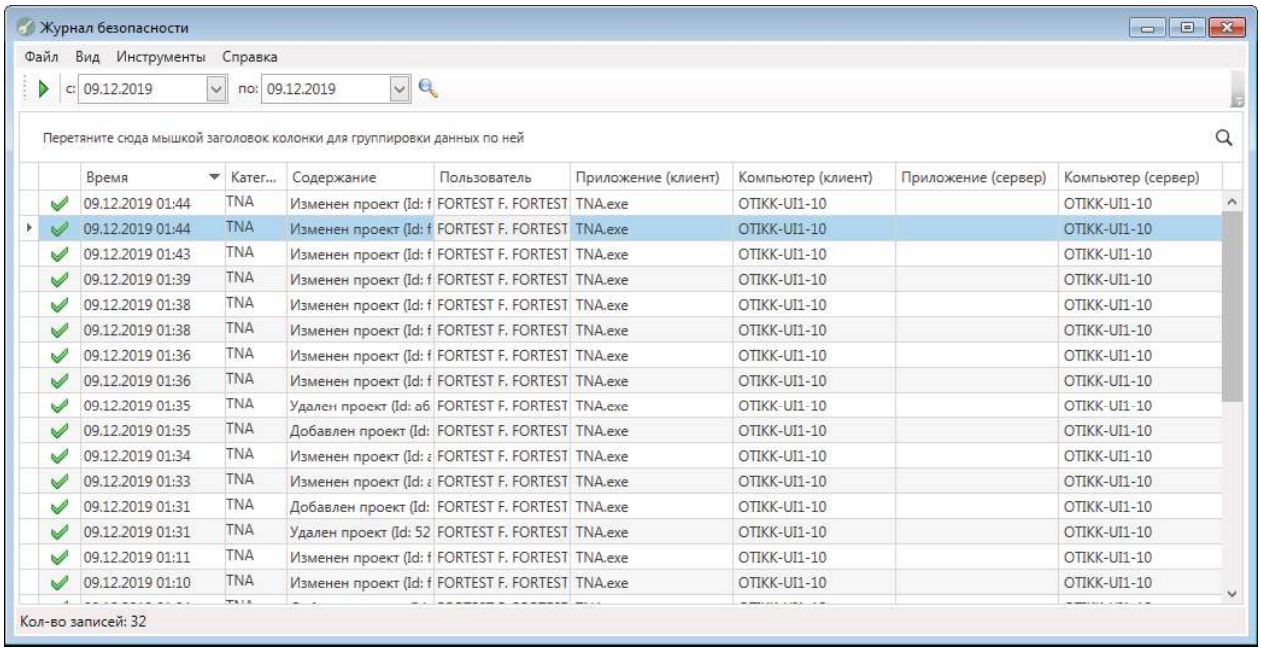


Вид окна приложения "Журнал безопасности"

1.6.1. Интерфейс журнала

Запуск приложения "Журнал безопасности"выполняется посредством команды меню Пуск | Все программы | Monitel | СК-11 | Журнал безопасности.

Запустить приложение также можно с помощью меню приложения "Агент клиента СК-11".



Вид интерфеса приложения "Журнал безопасности"

Окно приложения содержит следующие элементы управления:

- Главное меню;
- Панель инструментов.

В области табличных данных выводится следующая информация о записях журнала безопасности:

- Столбец для отображения статуса действия пользователя. Успешное действие обозначается значком ✔, неуспешное действие обозначается значком ❗;
- Столбец "Время" содержит дату и время зафиксированного действия пользователя;
- Столбец "Категория" содержит наименование контролируемой категории событий безопасности. Категория событий безопасности содержит набор контролируемых действий пользователей для определённой части Системы. В журнале безопасности по умолчанию фиксируются события безопасности следующих категорий:
  - TNA – фиксируются события безопасности, связанные с изменением общих проектов и общих настроек приложения TNA;
  - Доступ к данным ЭЖ – фиксируются события безопасности, связанные с доступом к приложению ОЖ;

- Санкционирование доступа – фиксируются события безопасности, связанные с авторизацией пользователей в Системе.

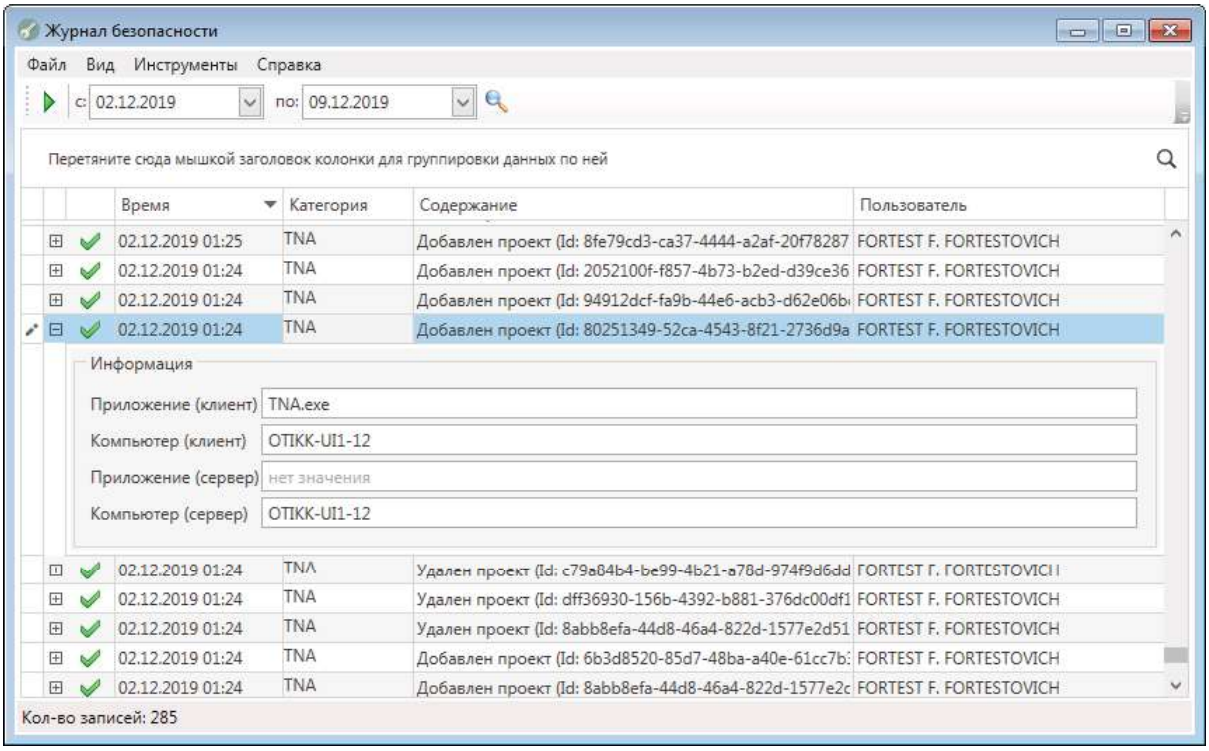
Состав категорий событий безопасности определён производителем ПО. Редактирование состава категорий событий безопасности не предусмотрено.

- Столбец "Содержание" содержит сведения о действии пользователя;
- Столбец "Пользователь" содержит наименование пользователя, выполнившего зафиксированное действие;
- Столбец "Приложение (клиент)" содержит наименование приложения, в котором произошло действие;
- Столбец "Компьютер (клиент)" содержит наименование компьютера, на котором произошло действие;
- Столбец "Приложение (сервер)" содержит наименование серверного приложения, направляющего сведения для записи в журнал безопасности;
- Столбец "Компьютер (сервер)" содержит наименование компьютера, на котором работает серверное приложение, производящее запись в журнал безопасности.

По умолчанию информация в области табличных данных отсортирована по времени внесения записей (столбец "Время"), более новые записи выводятся в начале таблицы.

В строке состояния приложения, расположенной внизу окна, выводится количество выведенных записей.

При переходе в компактный вид табличной области окна приложения автоматически скрываются столбцы "Приложение (клиент)", "Компьютер (клиент)", "Приложение (сервер)", "Компьютер (сервер)". Для записи становится доступна форма с информацией, содержащейся в указанных столбцах. Отобразить форму можно раскрыв дочернюю для записи строку с помощью щелчка ЛКМ на элементе ⊞. Скрыть дочернюю для записи строку можно щелчком ЛКМ на элементе ⊟.



Отображение формы с информацией для записи в компактном виде

1.7. Журналы работы

Для организации централизованного и унифицированного протоколирования работы приложений предназначены модуль управления записью журналов работы (LogServer) и разделяемые библиотеки.

В процессе работы каждый компонент ведёт журнал работы с помощью разделяемой библиотеки. Модуль управления записью журналов работы перехватывает все сообщения компонентов сервера, которые необходимо записать в файл, кэширует их и записывает централизованно, уменьшая количество обращений к жёсткому диску. При остановленном модуле запись производится в файлы напрямую. Такой механизм позволяет:

- увеличить производительность системы, так как уменьшается количество обращений к дисковой подсистеме;
- гарантировать сохранение сообщений, сгенерированных приложением, даже в случае его аварийного завершения.

При конфигурации подсистемы есть возможность настроить период времени, в течение которого производится накопление сообщений в буфере, а также максимальный объём буфера.

При записи сообщений приложение указывает уровни важности для них. Подсистема позволяет во время работы приложения менять уровень подробности – сообщения с важностью ниже этого уровня сохраняться в журнал работы не будут.

Журналы работ приложений – это текстовые файлы, которые могут просматриваться штатными средствами операционной системы.

Кроме того, в рамках подсистемы имеется приложение, предоставляющее возможность удалённого мониторинга журнала работ в реальном времени. Удалённый мониторинг журналов позволяет настраивать фильтрацию передаваемых сообщений на стороне сервера, в том числе и по уровню важности сообщений, при этом уровень важности сообщений, передаваемых в задачу мониторинга, может быть как выше, так и ниже уровня подробности.

1.8. Компонент адаптации скриптов

Достаточно часто в компонентах, реализующих различную функциональность, требуется предоставлять возможность гибкого конфигурирования логики работы, предоставляя пользователям на рабочих местах возможность описывать алгоритмы, реализующие специфичную для конкретного контекста логику.

В виду того что бизнес-процессы у различных пользователей и организаций могут существенно отличаться и на этапе разработки невозможно учесть всю эту специфику, необходимо предоставлять достаточно гибкое средство для решения подобных проблем. Компонент адаптации скриптов нацелен на решение данной задачи посредством предоставления пользователям возможности написания скриптов на языке *C#*. Компонент адаптации скриптов представляет определённые базовые функции по проверке, компиляции и выполнению скриптов.

Компонент адаптации скриптов широко используется в следующих модулях:

- **Подсистема проверки данных –** для предоставления более гибкого и удобного подхода к описанию правил проверок корректности данных. При помощи удобного редактора пользователь создаёт скрипт-правило, реализующее ту или иную логику проверки данных.
- TNA и Редактор модели **–** для предоставления пользователям возможностей по автоматизации своих действий во время работы с данными программными продуктами (см. рис. ниже).
- Модуль анализа нарушений ограничений **–** для предоставления пользователям возможностей описывать проверки, которые нельзя описать с помощью базовых конструкций, предоставляемых модулем.

1.9. Серверное приложение "Административные блокировки"

Серверное приложение Административные блокировки предназначено для установления административных блокировок, которые могут быть использованы при синхронизации различных бизнес-процессов СК-11. Блокировки могут быть установлены по уникальному идентификатору объекта (в этом случае будут заблокированы все нижележащие объекты в дереве) или по любому названию. Установленные блокировки являются срочными. То есть при установлении должно быть передано количество секунд, в течение которых будет действовать блокировка. При необходимости её можно будет продлить.

Предоставляются следующие методы:

- AcquireLock – установить/продлить блокировку. Требуется передать идентификатор объекта либо имя блокировки, длительность в секундах и квитанцию (тикет) предыдущей блокировки, если таковая была вызвана.
- IsObjectLocked – проверяет, была ли установлена блокировка на указанный объект.
- ReleaseLock – освобождает блокировку, требуется передать квитанцию (тикет) блокировки.
- GetLockOwner – возвращает имя пользователя, который установил указанную блокировку.

Механизм административной блокировки используется при редактировании графических схем в совместно изменяемой пользователями версии модели. При переходе к редактированию графической схемы в соответствующем редакторе, схема автоматически блокируется для редактирования другими пользователями. Пользователю, при попытке перехода в режим редактирования заблокированной схеме, выводится соответствующее сообщение. Блокировка снимается после закрытия графического редактора пользователем, осуществлявшем редактирование, как при сохранении изменений в схеме, так и без сохранения изменений в схеме.

Механизм административной блокировки используется в дистанционном управлении. В случае административной блокировки операция ДУ не доступна другим пользователям для объекта блокировки. Административная блокировка снимается с объекта по истечении таймаута (60 секунд).