ООО «Компания Семь Печатей»

117216, Москва, ул. Феодосийская, д. 1, корп. 6; тел.(факс): (495)225-25-31, (495)020-23-46

Email: 2252531@mail.ru; Web-page: www.sevenseals.ru, www.shop-sevenseals.ru

Система контроля и управления доступом

TSS-OFFICE

TSS-PROFI

***ВЕРСИЯ** 7*

Программное обеспечение

Программы ядра СКУД

руководство администратора

МОСКВА

2018

Оглавление

В документе используются специальные термины и выражения. Для полного понимания информации, изложенной в данном тексте, рекомендуем ознакомиться с глоссарием «TSS0011_Словарь терминов».

1. Принципы работы системы

Для работы Системы Контроля и Управления Доступом (далее СКУД или Система) в полнофункциональном (комплексном) режиме должны быть запущенны следующие программы (сервисы):

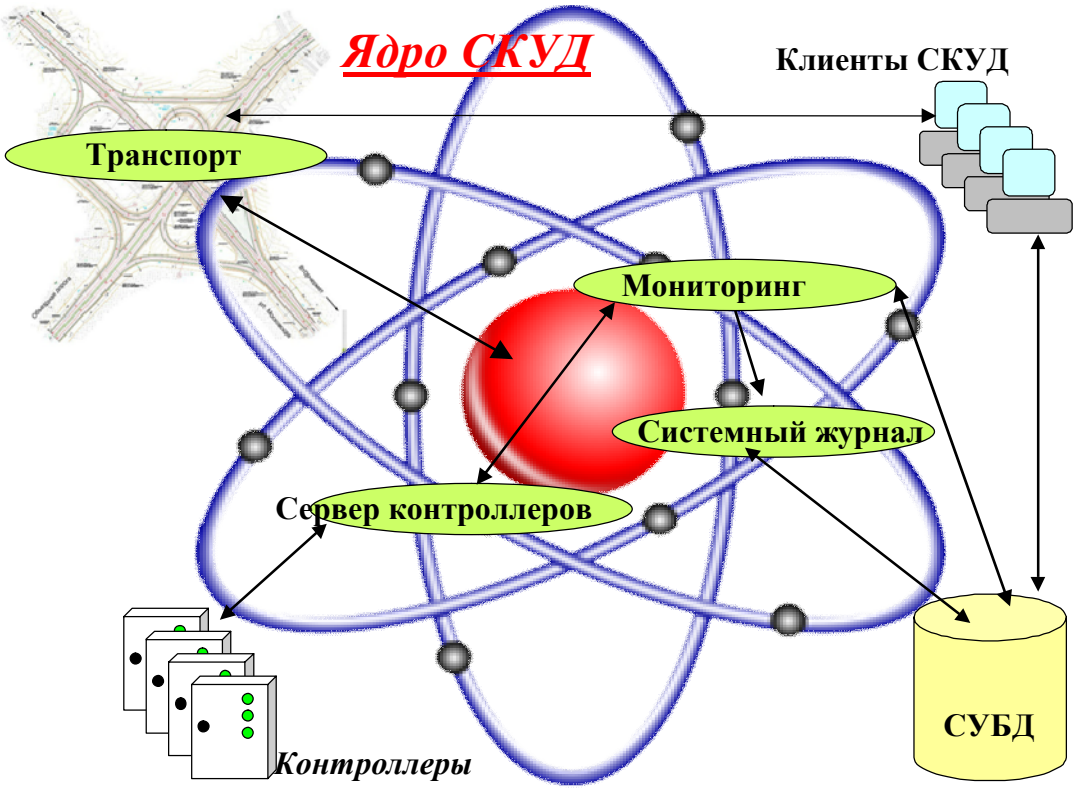
- *Система управления* (ACSGMSServer),
- *Транспорт* (Transsrv),
- *Сервер контроллеров* (ServConts),
- *Мониторинг* (Monitoring),
- *Системный журнал* (WriterLog).

Все указанные программы в совокупности составляют ядро СКУД.

Работа ядра СКУД базируется на двух основных принципах:

1. Разделение функций между программными модулями.
2. Сервисное исполнение всех модулей Ядра.

На рисунке приведена примерная схема взаимодействия основных элементов СКУД TSSProfi.



Служба *Сервер контроллеров* (которая, собственно говоря, является драйвером оборудования СКУД) работает в ОС Linux[1] и Windows[2].

Остальные службы работают под ОС Windows.

Все сервисы ядра являются не интерактивными. Для работы с ними (слежение, управление, настройка) используются соответствующие консольные приложения (программы). Т.о. каждый сервис имеет свое консольное приложение. Исключение составляет служба *Сервер контроллеров* – информацию о ее работоспособности несет консоль службы *Мониторинг*.

Все программы ядра инсталлируются как сервисы во время установки ПО. При установке каждого сервиса создается соответствующая секция в реестре, в которой прописываются значения настроек по умолчанию. Параметры реестра по необходимости могут быть изменены либо в соответствующем консольном приложении, либо в программе *Редактор установок*.

Для работы неотъемлемой части системы СУБД *Fierbird* должно быть установлено соответствующее ПО. В настоящей версии СКУД используется версия Fierbird 2.5 (Firebird-2.5.3.26780_0_Win32.exe). Установка ее выполняется во время установки комплекса СКУД с дистрибутивного диска. Ее работа обеспечивается функционированием двух сервисов (*Firebird Server – DefaultInstance* и *Firebird Guardian Service*) на сервере СКУД. Подробно о работе данной СУБД будет рассказано в документе *Администрирование - Обслуживание баз данных*. Для обозначения СУБД *Fierbird* далее будет использовано сокращение FB.

Все остальные модули СКУД реализованы как программы и являются клиентскими приложениями по отношению к модулям ядра. Еще раз повторим, что для работы СКУД необходимо и достаточно функционирование вышеперечисленных сервисов.

В настоящем документе будет рассказано о работе сервисов ядра системы.

2. Назначение модулей Ядра СКУД

2.1. Программа *Система управления*

ACSGMSServer [3] (acsgmsserver.exe) – сервис, позволяющий администрировать СКУД. Он стартует службы ядра[4], следит за их корректной работой, обеспечивает защиту Системы от несанкционированного доступа.

Сервис должен стартовать автоматически при старте ОС.

Служба *Система управления* имеет консоль (программу) *Консоль системы управления* (ACSGMSServerConsol)

2.2. Программа *Сервер Контроллеров*

ServCont (ServCont.exe) – программа, обеспечивающая работу с контроллерами. Является, по сути, драйвером оборудования.

Служит для установки и поддержки связи с контроллерами, обработки ошибок связи, восстановления связи. Является практически ретрансляторам событий и команд между контроллерами и *Системой принятия решений* (*Мониторингом*).

---

[1] Теоретически драйвер должен работать на всех UNIX-подобных системах. На сегодняшний день он протестирован на следующих клонах Linux: ucLinux, Suise, Debian.

[2] Здесь и далее подразумеваются ОС Windows XP, 2003, Vista, 2008.

[3] ACSGMS – Access Control System General Management System.

[4] За исключением *Сервера контроллеров*.

Работает на любом ПК ЛВС или *Мастер-контроллере TSS* под управлением ОС Linux или Windows. Соединяется с контроллерами СКУД либо через сом-порт данного ПК, либо через ЛВС посредством интерфейсного модуля TSSEthernet.

Не требует никаких настроек. Не хранит никаких данных об оборудовании. Ведет протокол работы. Возможно включение трассировки записи приходящих от контроллеров событий.

Устанавливаются при инсталляции ПО. Физически является программой (servcont.exe), запускаемой сервисом (servconts.exe), который следит за ее работоспособностью.

2.3. Программа *Транспорт*

TSSTransport (Transsrv.exe) – сервис, обеспечивающий сетевое взаимодействие модулей СКУД. Должен функционировать только на Сервере Системы[5]. Имя ПК, на котором работает *Транспорт*, указывается в файле настроек (файлы с расширением .ini) для каждого модуля (файлы настроек конфигурируются автоматически во время инсталляции).

Сервис должен стартовать автоматически при старте ОС.

2.4. Программа *Системный журнал*

WriterLog или *Системный журнал* (WriterLog.exe). Программа, осуществляет запись в Системный журнал СКУД. Именно через нее обращаются к Системному журналу все модули системы. Помните, что данная программа не предназначена для просмотра и работы с журналом. Для этих целей пользуйтесь программой *Дистанционный мониторинг* и системой отчетов.

Тип старта – ручной. Запуск выполняется *Системой управления*.

2.5. Программа *Мониторинг*

Программа предназначена для выполнения алгоритмов СКУД (принятие решения по правам допуска, реализация различных режимов и функций СКУД, работа с охранной сигнализацией).

При загрузке программа *Мониторинг* проверяет наличие файла защиты, а также соответствие параметров лицензии реальной конфигурации системы. В случае несовпадения этих параметров[6] программа прекращает работу.

Программа функционирует только в связке с программами *Сервер контроллеров* и *Системным журналом*.

Необходимо понимать, что все остальные модули СКУД (*Персонал*, *Проходная* и прочие) будут полноценно функционировать только при работающем *Мониторинге*.

Тип старта – ручной. Запуск выполняется *Системой управления*.

На рисунке приведена общая схема взаимодействия модулей СКУД.

[5] Сервером СКУД называется компьютер, на котором исполняются модули Ядра СКУД и на котором находится база. Относительно ЛВС это обычная рабочая станция.

[6] Подлежит проверке число контроллеров, их адреса, количество записей в базе персонала, количество рабочих станций, количество мультимониторингов.

2.6. Мониторинговый и Мультимониторинговый режимы работы

Мультимониторинг – это режим работы системы контроля доступа, в котором программы *Мониторинг* и *Сервер Контроллеров* запускаются на разных рабочих станциях ЛВС.

Мониторинг – режим работы, при котором все модули ядра выполняются на Сервере СКУД.

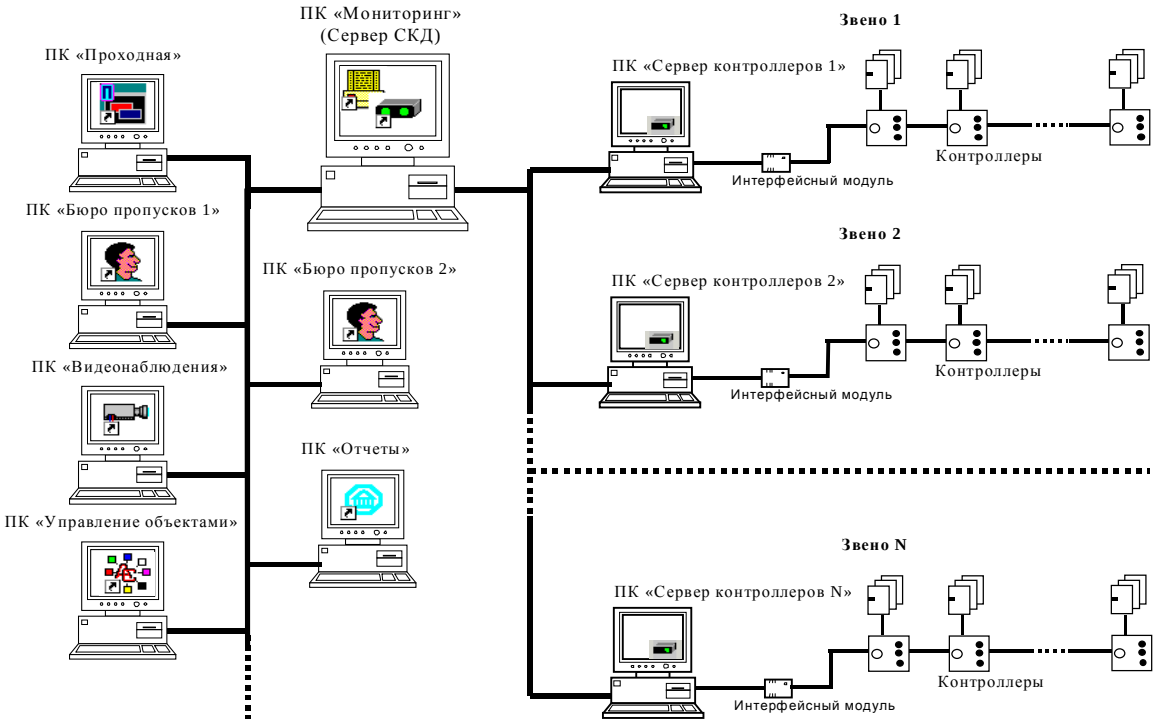
Назначение *Мультимониторинга* – это расширение стандартной системы контроля доступа, позволяющее связать в единое целое ее отдельные звенья.

Отдельным звеном СКУД является некоторое количество контролеров, связанных общей шиной, и заведенных на COM-порт компьютера через интерфейсное устройство Bit 4.3. СКУД в своей стандартной конфигурации может работать либо с одним таким звеном, либо с несколькими звеньями, заведенными на разные сом-порты компьютера Сервера СКУД, а также, включенных в локальную сеть посредством интерфейсного модуля TSSEthernet.

Такая конфигурация СКУД, как *Мультимониторинг* позволяет объединить звенья контроллеров, заведенные на **разные** сетевые ПК, в единую СКУД. Система в такой конфигурации может быть использована в следующих случаях:

- Когда линии контроллеров расположены на разных территориях и связывание их в единую сеть либо недопустимо по техническим характеристикам (удаленность), либо нецелесообразно по экономическим соображениям.
- Когда система состоит из большого (несколько десятков) числа контроллеров и требуется повысить ее надежность путем разгрузки центрального сервера.

Принцип работы *Мультимониторинга* поясняется на рисунке.



Каждый из компьютеров «Сервер контроллеров…» связан со своей линией оборудования СКУД. На каждом из них работает программа обмена с контроллерами (прием событий и управление). Вся информация передается на центральный компьютер системы – *Сервер*

СКУД, на котором запущена программа *Мониторинг*. *Мониторинг* объединяет полученные данные по правилам, установленным при его конфигурировании. Он принимает решения по правам доступа и передает их на ПК управления контроллерами. Те, в свою очередь, «доводят» это решение до контроллеров, т.е. указывают им, разрешить или запретить проход (включать или не включать реле).

В случае разрыва связи *Серверов контроллеров* с *Мониторингом* (выключение программы, обрыв локальной сети) каждый *Сервер контроллеров* переводит свою цепочку контроллеров в автономный режим.

При восстановлении связи с *Мониторингом Сервер контроллеров* автоматически переводит контроллеры в комплексный режим.

При отключении *Сервера контроллеров*, связанное с ним оборудование начинает функционировать в автономном режиме.

Принципы работы остальных модулей системы остаются прежними. Они связаны по локальной сети **только** с ПК *Мониторинга* и видят единую цепь контроллеров.

Использование *Мастер-контроллеров TSS* является, по сути, разновидностью режима *Мультимониторинг*, только в этом случае роль ПК *Сервера контроллеров* выполняет *Мастер-контроллер*.

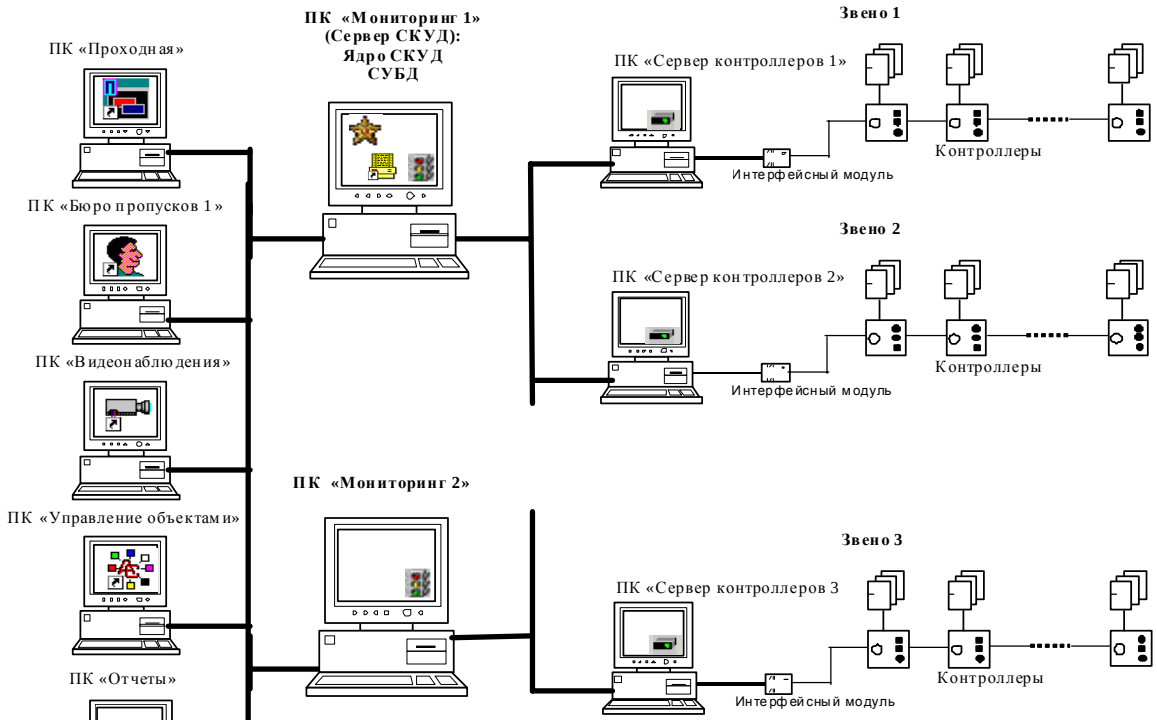
Подчеркнем, что в настоящей версии ПО этот режим конфигурируется точно также как стандартный (т.е. с одним *Сервером контроллеров*). Поэтому далее мы не будем отдельно освещать настройки *Мультимониторинга*. Сам термин оставлен только в связи с лицензированием числа подключаемых к Серверу СКУД ПК с *Серверами контроллеров*.

2.7. Распределенный мониторинг

В текущей версии (7.3) функционирует принципиально новая схема работы сети СКУД, а именно создана возможность включать одновременно несколько *Мониторингов* (т.е. программ, непосредственно исполняющих алгоритмы работы системы). При этом каждый *Мониторинг* связывается (на этапе настройки) со «своими» *Серверами контроллеров*.

Такое построение позволяет значительно разгрузить центральный Сервер системы и ускорить ее работу, особенно при высокой (порядка сотен в секунду) интенсивности проходов.

Принцип построения такой сети показан на рисунке.


ПК «Проходная»
ПК «Бюро пропусков 1»
ПК «Видеонаблюдения»
ПК «Управление объектами»
ПК «Отчеты»
ПК «Мониторинг 1» (Сервер СКУД): Ядро СКУД СУБД
ПК «Мониторинг 2»
Звено 1
ПК «Сервер контроллеров 1»
Интерфейсный модуль
Контроллеры
Звено 2
ПК «Сервер контроллеров 2»
Интерфейсный модуль
Контроллеры
Звено 3
ПК «Сервер контроллеров 3
Интерфейсный модуль
Контроллеры

2.8. Особенности установки модулей ядра

Инсталляция сервисов ядра выполняется автоматически при установке системы с дистрибутивного компакт диска[7].

Всего должно быть установлено пять служб:

- *Транспорт* (Transsrv) – под именем TSSTransport.
- *Система управления* (ACSGMSServer.exe) – под именем TSSACSGMSServer.
- *Сервер контроллеров* (Servconts.exe) – под именем TSSServcont.
- *Мониторинг* (Monitoring.exe) – под именем TSSMonitoring.
- *Системный журнал* (Writerlog.exe) – под именем TSSWriterlog.

Три сервиса имеют тип старта Automatic, т.е. запускаются автоматически при старте операционной системы:

- TSSTransport
- TSSACSGMSServer
- TSSServcont

Таким образом, запуск СКУД в комплексном режиме может быть осуществлен без вмешательства оператора[8], т.к. система стартует до регистрации пользователя.

Кроме собственных консольных приложений, для управления сервисами ядра в процессе эксплуатации комплекса существует утилита *Управление сервисами* (ServiceTool). Сервис транспорта может управляться и настраиваться с помощью программы *Консоль транспорта* (tsfe). Обе программы устанавливаются во время инсталляции СКУД.

2.9. Протоколирование работы модулей ядра

Каждый из указанных модулей во время своей работы фиксирует различные события в собственном файле протокола. Все подобные файлы имеют расширение LOG и хранятся в папке ACS/Log. Имя файла состоит из имени соответствующей программы, даты и времени создания файла. Файл протокола создается при каждом старте приложения. Эти файлы хранят всю информацию о функционировании программы, в том числе и о сбоях в ее работе, в работе других программ комплекса, в работе ОС. При возникновении проблем в работе комплекса данные файлы должны быть переправлены разработчику ПО для детального анализа. При нормальной работе комплекса файлы протоколов должны периодически удаляться администратором СКУД.

[7] Напомним, что инсталляция должна выполняться на профиле пользователя с правами администратора.

[8] При соответствующих настройках.

3. Система управления

3.1. Общее описание

Служба *Система управления* или *Сервер управления* (ACSGMSServer.EXE) располагается в каталоге ACS.

Она выполняет следующие функции:

- Запуск программ Ядра СКУД, а именно:
  - *Мониторинг*
  - *Системный журнал*
- Слежение за работоспособностью программ комплекса.
- Защита ОС от запуска несанкционированных процессов.
- Контроль соблюдения конфигурации СКУД, указанный в файле лицензии (ACS.INI).

Программа стартует автоматически как сервис при запуске ОС.

Список отслеживаемых процессов прописывается в файле настроек ACSGMSServer.INI.

Для управления службой служит программа *Консоль системы управления* (ACSGMSServerConsol.exe). После ее запуска в системном трее появляется иконка программы  . При наличии критических ошибок в работе всей системы в целом, иконка сменяется на  .

Двойной клик мышки на иконке разворачивает основное окно программы. В контекстном меню (правый клик мышью) присутствует единственный пункт, позволяющий закрыть программу.

3.2. Настройка

Вся настроечная информация хранится в текстовом файле ACSGMSServer.ini. В секции [Options] указаны основные параметры, необходимые для работы программы:

ALIAS =@ACS
Указывается имя алиаса, базы данных СКУД.
APPSERVER=TSS
Имя компьютера, на котором работает программа *Транспорт*.
CONNECTNAME=ACSGMSServer
Имя программы для регистрации в Транспорте.
PROGRAMM_KEY=@ACSGMSSERVER
Секция системного реестра с данными для работы программы.
RPCPort=5501
Порт для связи с *Консолью системы управления*.

Далее следуют секции с именами [SERVnn] (где nn – порядковый номер 01, 02 и т.д.), каждая из которых описывает параметры для работы по слежению за сервисом или программой:

RusName=Системный журнал
Русское название службы.
Name=TSSWriterlog
Имя службы.
TypeS=S
Тип (S – сервис, P – программа)
Path=C:\ACS\Writerlog.exe
Расположение модуля на диске.

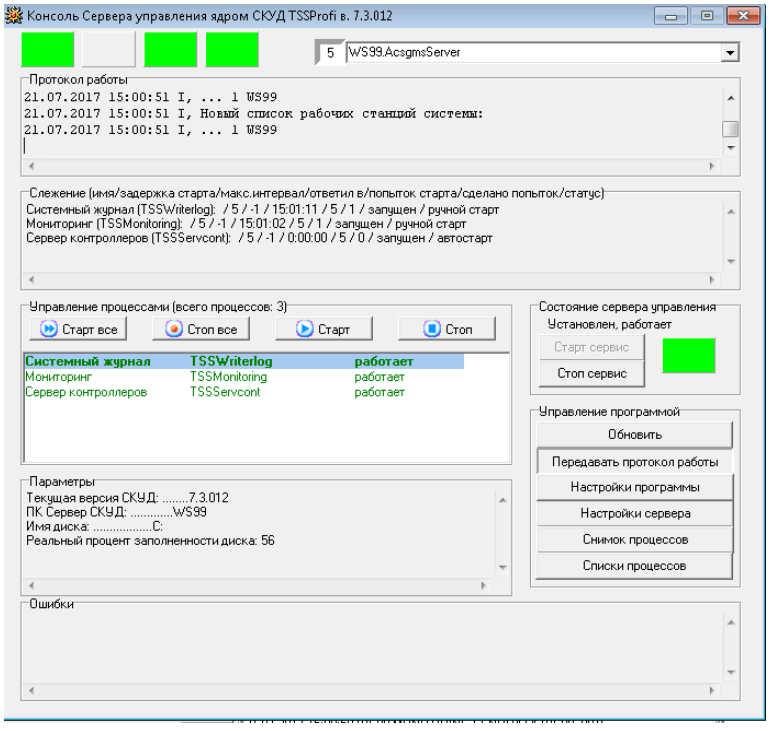
DelayOnStartSystem=3
Задержка старта модуля с момента старта *Системы управления*.
MaxAttemptsStartCounts=2
Максимально допустимое число попыток старта службы.
RebootSystemAfterRestartFault=YES
Действия программы при невозможности стартовать службу (т.е. при достижении максимально допустимого числа попыток старта): NO – ничего не предпринимать, YES - перезагрузить компьютер.
LifeInterval=10
Максимально допустимая задержка прихода события ОК, по которому программа определяет работоспособность отслеживаемого приложения. Если указано значение -1, то слежение за событиями не производится[9].
Critical=YES
Признак критичности неработоспособности данного приложения для функционирования всей системы. Если указано YES, то *Система управления* оповещает о фатальных сбоях в работе СКУД.

3.3. Программа *Консоль системы управления*

Как уже говорилось, служба ядра *Сервер управления* не имеет визуальной формы. Управляется служба (равно как и остальные службы ядра) программой *Консоль системы управления.*

Программа *Консоль системы управления* может работать на любой рабочей станции ЛВС, имеющей связь с Сервером СКУД. Однако возможность старта и остановки службы *Сервер управления* будет доступна только при их работе на одном ПК.

Окно *Консоли системы управления* имеет вид, показанный на рисунке.



В верхней части окна представлена текущая информация о работе комплекса. Четыре панельки определяют (слева направо):

- Работоспособность *Сервера управления*.
- Наличие ошибок в работе *Сервера управления*.
- Состояние связи с *Сервером управления*.
- Наличие критических сбоев в работе комплекса.

Зеленый цвет панельки говорит о нормальном состоянии, красный – о сбое.

Справа представлен раскрывающийся список текущих процессов СКУД.

На панели *Протокол работы* отображается информация о работе сервиса. Те же данные служба управления пишет в свой протокол (файл в папке ACS\LOG с расширением LOG).

[9] Но при этом работоспособность приложения продолжает отслеживаться по его присутствию в списке процессов Windows.

По умолчанию отображение протокола отключено. Для его включения следует нажать клавишу *Передавать протокол работы*.

На панель *Слежение* выводятся данные о настройках каждого сервиса. В строке содержится следующая информация:

- Имя службы.
- Задержка старта (сек.) после запуска *Службы управления*.
- Максимальный интервал ожидания ответа от соответствующей службы (сек.) по истечению которого связь с ней считается потерянной, и начинают предприниматься попытки ее рестарта.
- Время последнего ответа (чч:мм:сс).
- Максимальное число попыток старта службы при ее остановке или потери связи.
- Текущее значение количества попыток старта.
- Статус – состояние (запущен или остановлен), тип старта (ручной, автоматический).

Панель *Управление процессами* отображает текущее состояние отслеживаемых служб и позволяет управлять ими. Набор клавиш выполняет следующие действия (слева направо):

- Стартует все службы.
- Останавливает все службы.
- Стартует выбранную в списке службу.
- Останавливает выбранную в списке службу.

В окне состояния служб отображается:

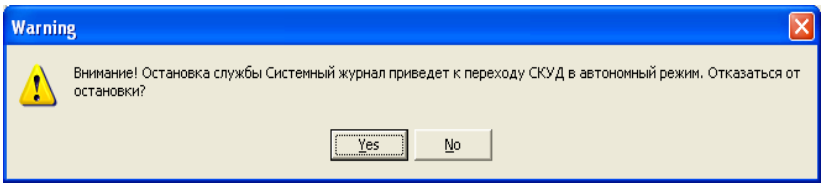
- Русское имя процесса.
- Имя службы.
- Состояние (остановлен, работает, останавливается, стартует).

Панель *Параметры* показывает некоторые информационные данные о работе системы.

Панель *Ошибки* информирует об ошибках в работе *Сервера управления*.

Панель *Состояние Сервера управления* отображает текущее состояние сервиса (установлен, неустановлен, работает, остановлен) и позволяет с помощью одноименных клавиш стартовать или останавливать его.

При остановке любой из служб будет выдано предупредительное сообщение. Внимательно почитайте текст сообщения перед тем, как нажать на клавишу.

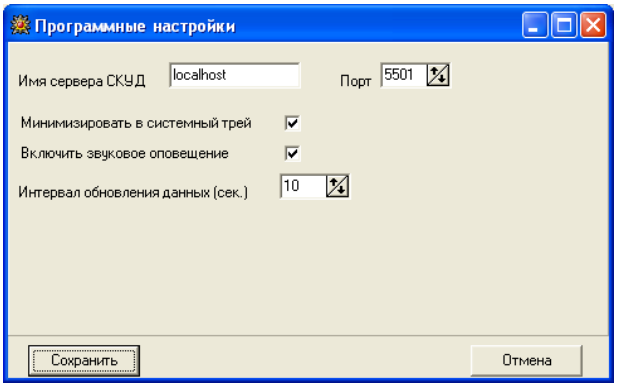


Вся выводимая информация обновляется с частотой 10 секунд. Этот параметр может быть изменен в настройках программы.

Панель *Управление программой* содержит ряд клавиш, каждая из которых выполняет следующую функцию:

- *Обновить* – выполняется принудительный запрос всех параметров *Сервера управления*.
- *Передавать протокол работы* – включить/выключить запрос полного протокола работы *Сервера* и вывод его в одноименное окно.
- *Настройка программы* – вывод окна редактирования программных настроек.
- *Настройка сервера* – вывод окна редактирования настроек *Сервера*.
- *Снимок процессов* – создать список текущих процессов Windows для последующего использования в режиме контроля/запрета несанкционированных процессов.
- *Списки процессов* – просмотр текущего состояния режима контроля/запрета несанкционированных процессов.

3.4. Настройка программы



Ряд параметров, используемых программой, может быть изменен в одноименном окне (клавиша *Настройки программы*).

Все редактируемые значения сохраняются в файле настроек (ACSGMSServerConsol.ini).

Имя сервера СКУД – имя ПК, на котором работает ядро СКУД в общем и Сервер управления в частности. Допустимо задавать IP адрес ПК.

Порт – адрес виртуального порта *Сервера управления*. Должен совпадать со значением, указанным в файле настроек *Сервера* в строке RPCPort. По умолчанию – 5501.

Минимизировать в системный трей – при включенной опции нажатие на «-» в заголовке главного окна программы помещает иконку в системный трей. В противном случае минимизированное окно будет располагаться на панели задач.

Включить звуковое оповещение – при включенной опции критические ошибки системы дублируются звуком.

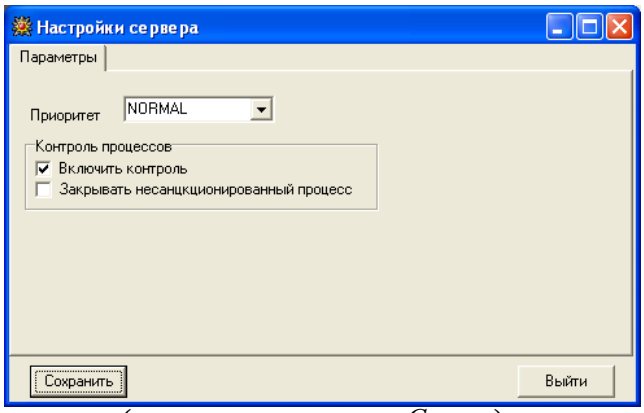
Интервал обновления данных – интервал запроса данных с сервера. По умолчанию – 10 секунд.

Клавиша *Сохранить* осуществляет выход из окна настроек с сохранением данных, клавиша *Отмена* закрывает окно без сохранения.

3.5. Настройка сервера

Некоторые настройки *Сервера управления* могут быть изменены в одноименном окне (клавиша *Настройки сервера*). Обратите внимание, что после редактирования настроек *Сервер* должен быть перезагружен. В связи с этим отметим здесь два момента, связанных с перезагрузкой *Сервера*.

1. Перезагрузка (точнее, последовательное нажатие клавиши *Стоп* и *Старт* на панели *Состояние сервера*) возможно только, если консоль *Сервера* запускается на том же ПК, на котором работает сам *Сервер*.
2. Несмотря на грозное предупреждение, кратковременная остановка службы управления не приведет к сбоям в работе СКУД.



Приоритет – Windows приоритет сервиса. Выбирается из списка.

На панели *Контроль процессов* можно включить одноименный механизм (опция *Включить контроль*). При включенном контроле возможно установить жесткий режим запрета запуска несанкционированных процессов (опция *Закрывать несанкционированный процесс*). Подробно контроль процессов описан в соответствующем разделе.

Клавиша *Сохранить* осуществляет сохранение (точнее, передачу на *Сервер*) настроек, клавиша *Выйти* закрывает окно настроек.

3.6. Запуск программ Ядра СКУД

Как уже было сказано, программа стартует автоматически при старте ОС. При этом она инициирует загрузку следующих программ Ядра СКУД:

- *Мониторинг*
- *Системный журнал*

По умолчанию порядок загрузки указанных модулей таков:

- *Системный журнал* стартует через три секунды.
- *Мониторинг* стартует через пять секунд.

Если при таких временны*х* интервалах во время загрузки СКУД происходят сбои (естественно, не вызванные ошибками установки или конфигурирования Системы), то рекомендуется увеличить значения задержек[10].

Служба ядра СКУД *Сервер контроллеров* стартует в автоматическом режиме и, поскольку до старта *Мониторинга* не выполняет никаких действий, то в диспетчеризации своего старта не нуждается. Тем не менее, она также помещена в список запускаемых служб.

Данный список содержится в файле настройки ACSGMSServer.ini. Его формат приведен в разделе Настройка.

3.7. Слежение за работоспособностью программ комплекса

Служба управления может следить за работоспособностью тех программ комплекса, которые в нем регистрируются. При остановке или зависании любой из этих программ *Система управления* стартует ее заново (как сервис, если это сервис, или как программу, если это программа).

Информация о текущем состоянии отслеживаемых приложений отображается на панели *Слежение* (см. раздел Интерфейс).

Настройка системы слежения выполняется в файле ACSGMSServer.ini. Его формат приведен в разделе Настройка. При остановке или зависании приложения выполняется попытка его перезапуска. Число попыток ограничено параметром MaxAttemptsStartCounts. При достижении этого значения система либо прекращает попытки восстановления, либо, если параметр RebootSystemAfterRestartFault выставлен в YES, перезагружает компьютер.

3.8. Защита ОС и ПО СКУД от запуска несанкционированных программ

Данный уровень защиты выполнен в двух вариантах:

- Фиксация старта не разрешенных приложений.
- Полный запрет на их выполнение.

Для включения режима защиты необходимо выполнить следующие действия:

1. Убедиться в том, что СКУД корректно функционирует.
2. Стартовать на ПК все процессы, к которым имеет право доступа его пользователь (кроме программ ядра СКУД, это возможно будут *Конфигуратор, Персонал, Дистанционный мониторинг, Редактор установок*).

10 Еще раз подчеркнем, что речь идет о старте СКУД, выполняемом во время загрузки ОС. В данном случае, на корректность загрузки модулей ядра СКУД могут влиять особенности конкретного компьютера (параметры его быстродействия, версии установленной ОС).

3. В *Консоли системы управления* нажать клавишу *Настройки сервера*. В появившимся окне на панели *Контроль процессов* включить опцию *Включить контроль*. При необходимости включить опцию *Закрывать несанкционированные процессы*. Нажать клавиши *Сохранить* и *Выйти*.
4. В главном окне программы нажать клавишу *Снимок процессов*.
5. Клавишами *Стоп сервис* и *Старт сервис* перезагрузить *Систему управления*.

Если Вы включили уровень проверки, то программа будет только фиксировать запуск несанкционированных процессов как в своем собственном протоколе (..ACS/LOG/ ACSGMSServer_yyyymmdd_hhmmss.log), так и в *Системном журнале СКУД*. При выборе максимального уровня защиты, любой несанкционированный процесс будет закрываться *Сервером управления.*

Будьте осторожны с сочетанием максимального уровня защиты с заданием пользовательских прав доступа для программы! В этом случае включайте в число разрешенных процессов системную программу работы с сервисами. Помните, что для изменения параметров необходимо остановить сервис. Если Вы все же закрыли себе доступ к изменению настроек и остановки сервиса, стартуйте Windows в безопасном режиме и отключите автоматический старт *Сервера управления.*

3.9. Контроль конфигурации СКУД

Сервер управления контролирует ряд параметров, указанных в файле лицензии ACS.INI, а именно:

- Число рабочих мест.
- Допустимость работы клиентских модулей СКУД.
- Число объектов для режима *Мультимониторинга*.

При запуске отсутствующего в файле лицензии приложения или при старте разрешенного приложения, но при превышении допустимого числа рабочих мест, оно (т.е. приложение) будет закрыто. Аналогично будет закрыта программа *Сервер контроллеров* при превышении разрешенного числа мультимониторингов системы.

4. Сервер контроллеров

4.1. Общее описание

Программа *Сервер контроллеров* (servcont.exe) является, по сути, драйвером оборудования. Исполняемые файлы и библиотеки, необходимые для ее работы, располагаются в папке ..ACS/Servcont.

При работе под ОС Windows программа запускается при старте службы *TSSServcont* (servconts.exe). Сама служба устанавливается при инсталляции комплекса и стартует автоматически при старте ОС.

При работе под ОС Linux работает только программа servcont.exe.

Программа не требует никаких настроек. При своей загрузке не выполняет никаких действий. Принцип ее работы предполагает наличие управляющего клиента, которым (для СКУД TSSProfi) является *Мониторинг*. Только после старта *Мониторинга* согласно его командам *Сервер контроллеров* начинает работу с оборудованием.

Программа не имеет никаких средств отображения. Вся необходимая информация о ее работе сообщается посредством *Консоли Мониторинга*.

Программа ведет собственный протокол своей работы, в который выводится главным образом информация об ошибках. Протоколом является текстовый файл, создаваемый при каждом новом сеансе работы программы. Имя его состоит из имени программы, даты и времени создания файла. Например: servcont20090312130053.txt.

4.2. Диагностика ошибок

При нормальном старте *Сервера контроллеров* в протоколе его работы будет зафиксированы две строчки:

2017-07-21_14:38:34 TSS Servcont v1.23 start.

2017-07-21_14:54:32 Client<127.0.0.1:49173> connected,

где **127.0.0.1:49173** – IP адрес и виртуальный порт ПК Мониторинга.

Все проблемы в работе *Сервера контроллеров* фиксируется в файле протокола. Вот некоторые из наиболее распространенных ошибок:

Ошибки связи с Мониторингом, возникающие вследствие неустойчивого канала и системных настроек Windows и LAN:

2011-03-28_15:17:12 Socket can't receive; sock ec: 10054.

2011-03-28_15:17:12 Client<192.168.0.57:1557> closed?

Где 10054 – ошибка сокета (см. Приложение 1).

Ошибки связи с контроллерами, возникающие из-за указания неверного адреса ком-порта (или IP адреса) или неверных адресов контроллеров.

2011-03-28_15:17:16 Channel<COM1@19200>.Controller<31>: No response in 50 msec

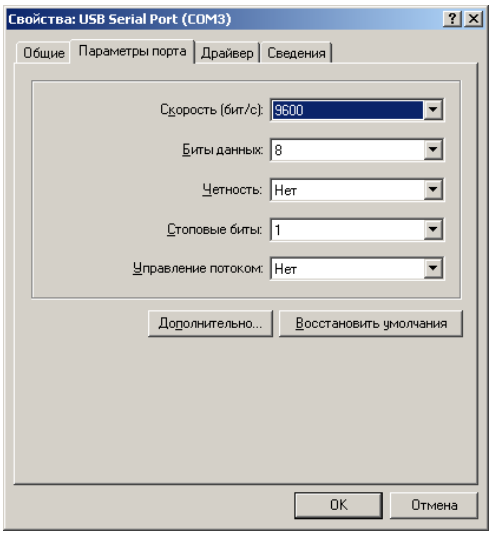
2011-03-28_15:17:36 Channel<COM1@19200>.Controller<31>: No response in 50 msec

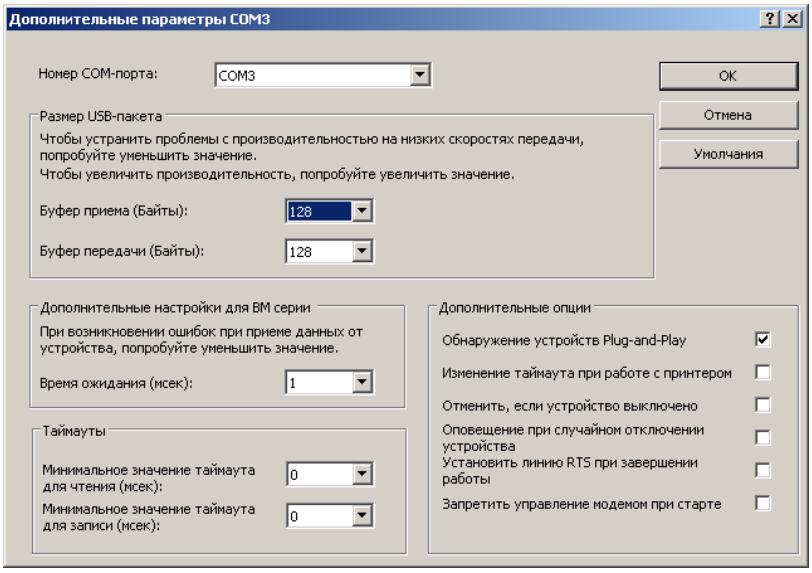
4.3. Настройка подключения контроллеров

4.3.1. USB соединение

При подключении контроллеров через USB, необходимо выполнить следующие настройки.

1. Установить соответствующий драйвер[11]. При этом создаться виртуальный ком-порт.
2. Зайти в диспетчер устройств, найти нужный ком-порт, открыть его *Свойства*. Далее, на закладке *Параметры порта* нажать клавишу *Дополнительно* и, при необходимости, изменить параметры:
   - Буфер приёма (Байты) - 128,
   - Буфер передачи (Байты) - 128,
   - Время ожидания (мсек) – минимальное значение, например, 1).





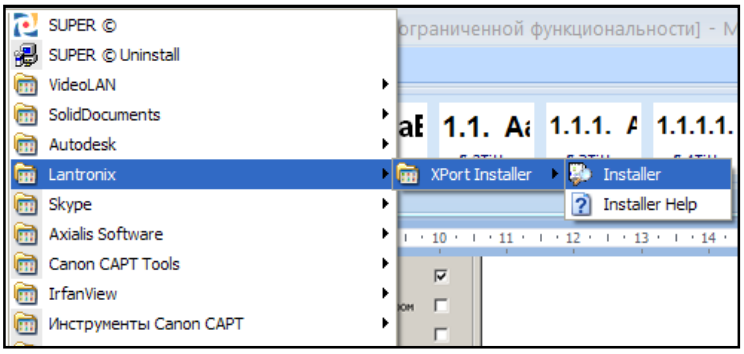
[11] Для USB модели интерфейсного модуля Bit 4.3 это драйвер с дистрибутивного диска TSSProfi. Для произвольного переходника USB-Com – это прилагаемый к нему драйвер.

4.3.2. Подключение через локальную сеть

При подключении цепочки контроллеров посредством интерфейсного модуля TSS-Ethernet прежде всего необходимо выполнить настройки модуля (точнее, преобразователя Ethernet-Com **XPort**). Настройка производится посредством программы Lantronix.

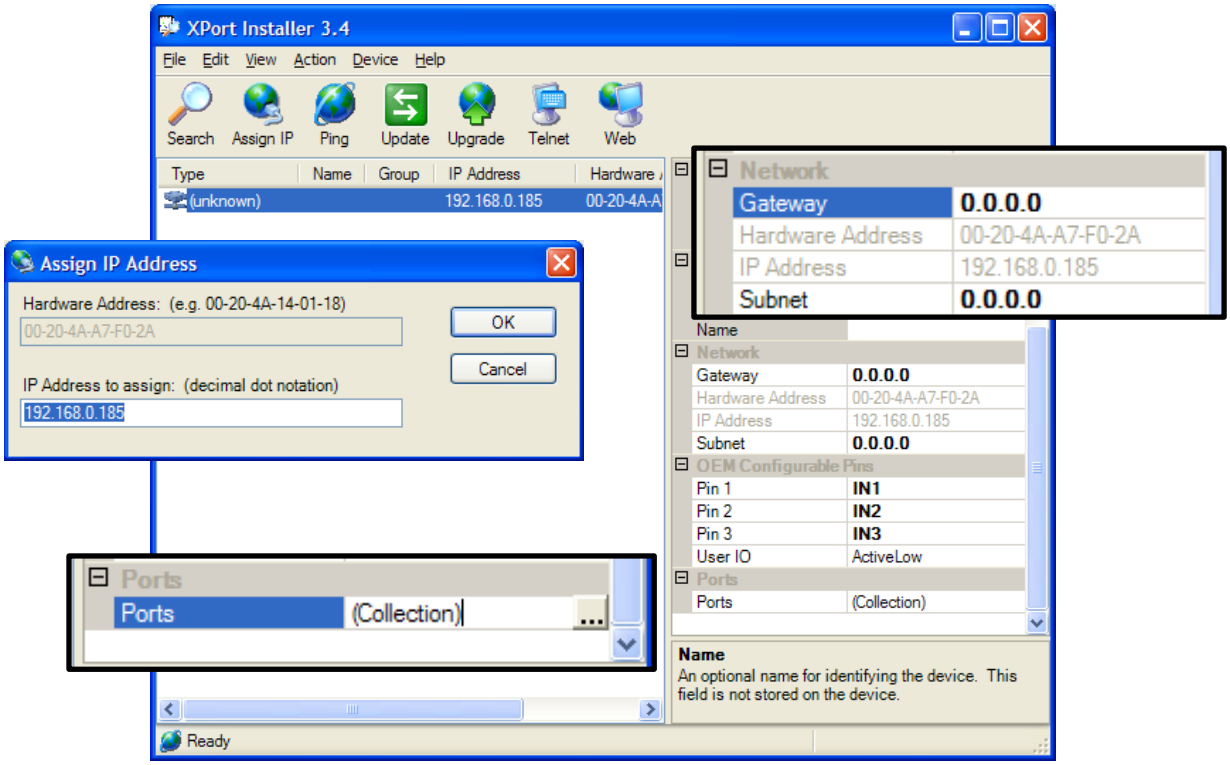
Указанная программа поставляется на дистрибутивном диске TSSProfi. Для ее установки следует запустить программу Setup из папки ...\XPort_Ethernet\InstallerXPort\.

После успешной инсталляции следует стартовать программу *Installer* (как показано на рисунке) и выполнить настройку оборудования, как описано далее.



В главном программном окне, с помощью соответствующих пунктов меню или кнопок на панели инструментов выполнить следующее:

Найти устройство в сети (*Search*). В результате поиска в окне списка устройств появятся все включенные в сеть преобразователи *XPort* с указанием их IP адресов.



Для изменения IP адреса следует выбрать на панели инструментов клавишу *Assign IP*.

Остальные настройки выполняются в дереве на панели свойств, расположенной в правой части программного окна.

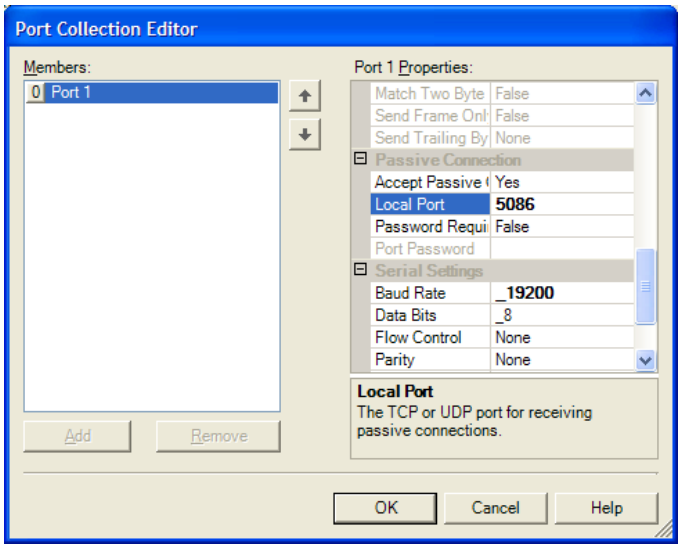
Дополнительные сетевые настройки выполняются в ветке дерева *Network*, как показано на рисунке.

Также следует проконтролировать свойства портов. Соответствующее окно отрывается при нажатии клавиши «…» в строке *Collections* ветки *Ports*.

Рекомендуется проверить адрес локального порта. Он должен быть равен 5086 и доступен в локальной сети. Значение скорости (*Baud rate*) должно составлять 19200.

Предполагается, что межконтроллерная линия подсоединена к модулю TSSEthernet в соответствии с документацией.

Для проверки корректности всех настроек следует воспользоваться тестовой программой *Newtest.exe*.



5. Мониторинг

5.1. Общее описание

Программа *Мониторинг* (Monitoring.exe) предназначена для реализации алгоритмов СКУД (принятие решения по правам допуска, реализация различных режимов и функций СКУД, работа с охранной сигнализацией).

Программа реализована как служба, которая инсталлируется при установке программного комплекса. Старт сервиса инициируется *Системой управления* TSSACSGMSServer.

Настройки службы выполняются следующим образом:

- Редактированием файла начальной загрузки *Monitoring.ini*. Смотрите раздел Описание загрузочных параметров настоящего документа.
- С помощью программы *Консоль Мониторинга* (включение основных режимов работы системы). Смотрите раздел Работа с программой Консоль Мониторинга настоящего документа.
- С помощью программы *Редактор настроек* (тонкая настройка системы). Смотрите документ *Редактирование параметров*.
- С помощью программы *Конфигуратор* (ведение базы данных: описание ПК *Серверов контроллеров*, каналов связи, контроллеров СКУД, оборудования контроллеров, прав доступа и прочее). Смотрите документ *Конфигурирование системы*.

5.2. Описание загрузочных параметров

Перед началом работы необходимо выставить системные параметры, используемые при загрузке программы. Все системные параметры описываются в файле *Monitoring.ini* (содержимое файла выделено курсивом). Не используемые и зарезервированные параметры не описываются.

[Options]

ALIASBASE =@ACS

Указывается имя алиаса, по которому располагается база данных Системы.

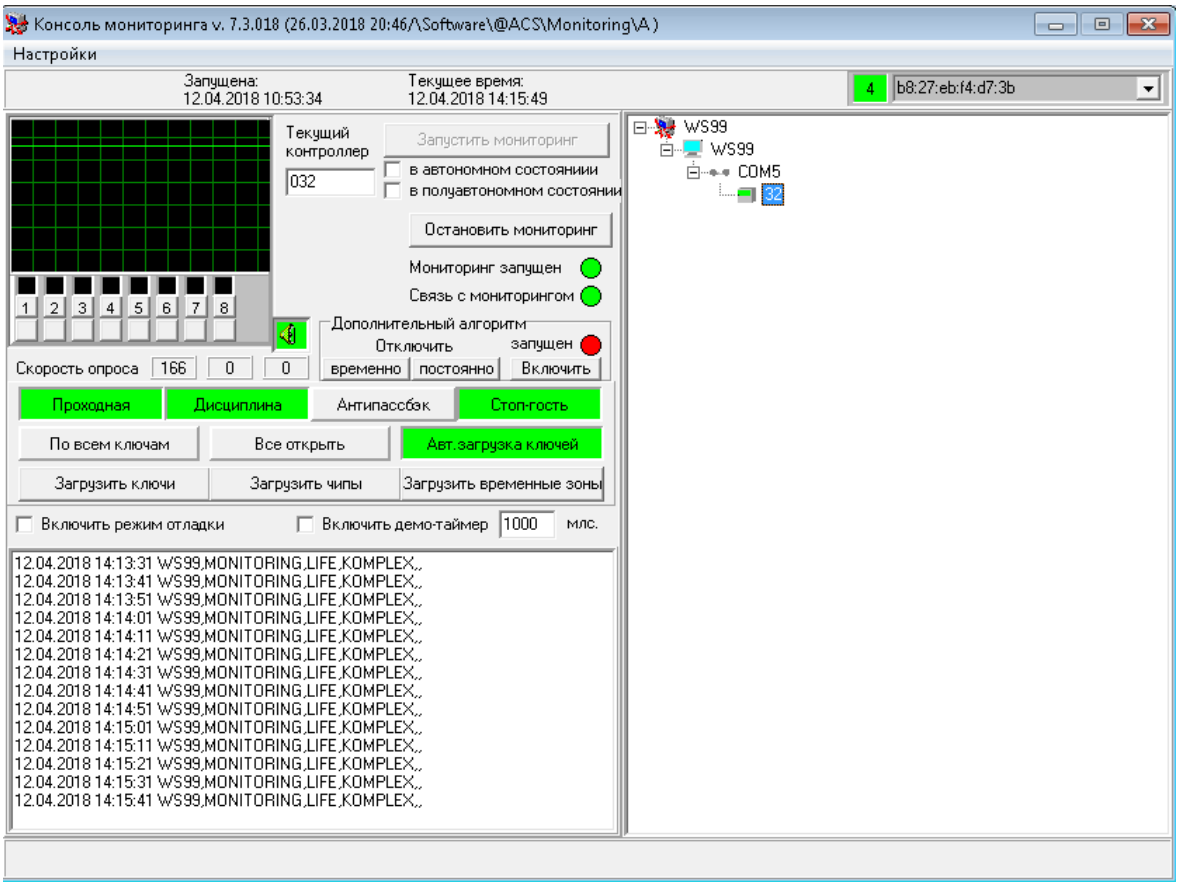
APPSERVER=TSS

Имя компьютера, на котором работает программа *Транспорт*.

Прочие параметры хранятся в системном реестре Windows (HKEY_LOCAL_MACHINE\SOFTWARE\@ACS\Monitoring\A). Настройка их выполняется частично *Консолью Мониторинга*, частично программой *Редактор настроек*.

5.3. Работа с программой *Консоль Мониторинга*

Как уже говорилось, служба ядра *Мониторинг* не имеет визуальной формы. Управляется служба (равно как и остальные службы ядра) программой *Консоль Мониторинга* (MonitoringConsole.exe).



Программа *Консоль Мониторинга* может работать на любой рабочей станции ЛВС, имеющей связь с Сервером СКУД. Однако возможность старта и остановки службы *Мониторинг* будет доступна только при их работе на одном ПК.

Окно *Консоль Мониторинга* имеет вид, показанный на рисунке ниже. На нем расположены следующие элементы отображения и управления.

Информационная панель

Отображается информация о версии и времени создания программы, время с начала сеанса, текущее время, список работающих процессов СКУД.

График

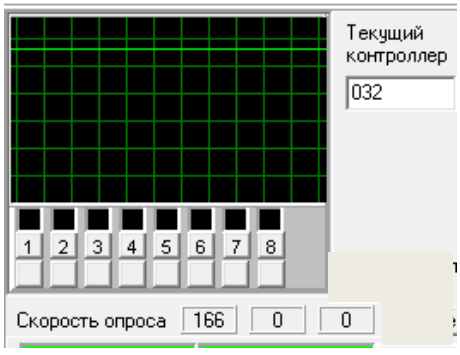


График отображает скорость опроса текущего контроллера. Текущий контроллер выбирается из дерева контроллеров в правом окне двойным кликом левой кнопки мыши. Реальное значение скорости пишется в строке *Скорость опроса*. При нормальной работе скорость для контроллеров, подсоединенных через сомпорт должна быть на уровне 140 – 160 опросов в секунду, для контроллеров на Ethernet канале – 40-60. Более низкая скорость и особенно резкие ее перепады («рванный» график) свидетельствуют о наличии проблем связи с контроллером или работоспособности самого кон-

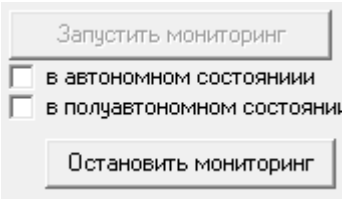
троллера.

Каждый из восьми столбиков под окном графика (по максимальному числу портов контроллера) отмечает последнее прошедшее по порту событие и позволяет управлять реле соответствующего порта.

Верхний ряд окошек цветом показывает последнее событие порта: зеленый – приложена карточка, синий – сработал датчик двери, желтый – нажата кнопка открытия двери.

Кнопка с номером порта включает реле, кнопка под ней – отключает. Время включения реле берется из базы конфигурации (значение, указанное в программе *Конфигуратор* в поле *Время реле*).

Управление Мониторингом



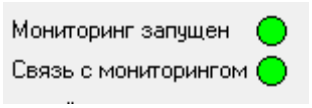
Клавиши *Запустить Мониторинг* и *Остановить Мониторинг* позволяют соответственно стартовать и остановить службу. Остановка *Мониторинга* приводит к мгновенному[12] ходу всей системы в автономный режим.

Обратите внимание, что выполнять указанные действия возможно только в случае, если *Консоль Мониторинга* запущена на одном с *Мониторингом* ПК.

Галка *в автономном состоянии* позволяет перевести работу Мониторинга в автономный режим.

Галка *в полуавтономном состоянии* переводит Мониторинг в режим вычитывания событий при автономной работе системы.

Индикация



На панели два индикатора:

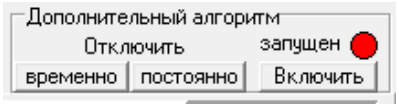
- Служба *Мониторинга* работает.
- Наличие связи Консоли с *Мониторингом*.

Кнопка с динамиком включает голосовые сообщения о работе системы.

Дополнительный алгоритм

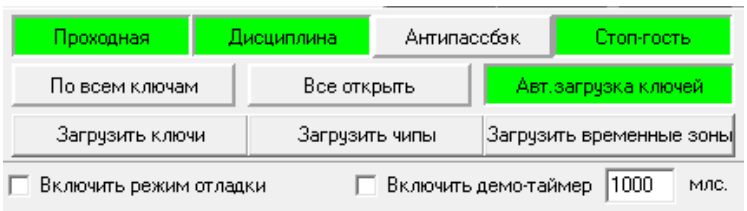
Индикация на панели оповещает о работе дополнительного алгоритма:

- Красный – не запущен либо не инсталлирован
- Зеленый – запущен и работает
- Желтый – запущен, но не работает



Клавиши на панели выполняют функции, соответственные своим названиям.

Управление режимами СКУД



Клавиши на панели управления режимами позволяют включать и отключать контроль следующих прав доступа:

Проходная – режим разрешения прохода во внутренние по-

[12] Точнее, через 3 – 4 секунды.

мещения только после пересечения проходной. По умолчанию выключен.

Дисциплина – режим контроля перемещения во внутренних зонах (группах помещений). По умолчанию выключен.

Антипасбэк (контроль повторного прохода) – режим запрета повторного прохода через проходную. По умолчанию выключен.

Стоп-гость – режим работы системы с гостевыми картами (запрет выхода на проходной и голосовое сообщение для изъятия карты[13]). По умолчанию выключен.

По всем ключам – включает режим прохода по любому ключу, в том числе и не загруженным в базу.

Все открыть – открывает все двери объекта.

Автоматическая загрузка ключей – загрузка текущих прав доступа ключей в контроллеры в автоматическом режиме. По умолчанию включен. Обратите внимание, что речь идет о загрузке прав доступа непосредственно после их создания или правки в программе *Бюро пропусков*. Такая загрузка не приводит к снижению производительности системы в отличие от полной перезагрузки базы ключей (клавиша *Загрузить ключи*). Напомним, что загрузка ключей в контроллеры (а точнее, синхронизация дисковой и контроллерной баз) позволяет системе корректно работать и в автономном режиме[14].

Клавиша *Загрузить ключи* выполняет полную загрузку прав доступа всех ключей в память контроллеров СКУД[15]. Данная операция является достаточно ресурсоемкой как для операционной системы, так и для системы контроля доступа. Поэтому не рекомендуется выполнять ее при интенсивно работающей СКУД.

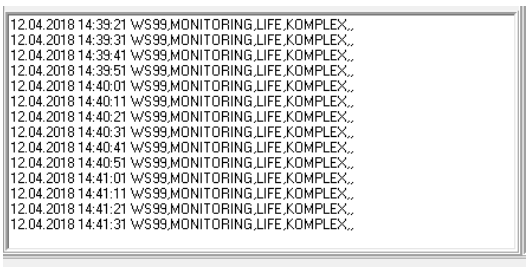
Клавиша *Загрузить чипы* выполняет загрузку охранных чипов (адресов охранных датчиков) в память контроллеров СКУД. Используется только при работе с сигнальными контроллерами марки ТСС.

Клавиша *Загрузить временны**е** зоны* выполняет загрузку расписаний в память контроллеров СКУД. Обратите внимание, что из всех многочисленных временн**ы**х расписаний, поддерживаемых в комплексном режиме работы СКУД, грузятся в контроллер (т.е. работают в автономном режиме) только недельные графики (временн**ы**е зоны). Причем число загруженных расписаний ограниченно первыми 16-ю[16] из созданных программой *Бюро пропусков*.

Галка *включить режим отладки* необходима разработчику для дополнительного анализа работы Мониторинга.

Галка *включить демо-таймер* служит для настройки демо-версии СКУД.

Окно текущих событий СКУД



В окно выводятся текущие события системы, как принимаемые от *Сервера контроллеров*, так и генерируемые *Мониторингом*.

Может быть использовано для целей отладки системы.

Окно структуры СКУД

[13] Здесь и далее речь идет о любом электронном идентификаторе доступа: proximity-карте, карте с магнитной полосой или штрих-кодом, ключом touch memory («таблеткой»).

[14] Т.е. без компьютера Сервера СКУД.

[15] Подробнее смотрите раздел Загрузка ключей.

[16] Напомним, что в памяти контроллера хранится 16 расписаний и 256 т.н. особых (или праздничных) дней.

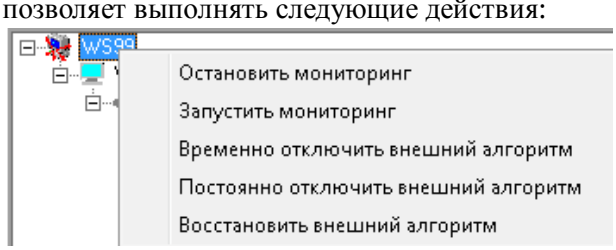
В правом окне отображается в виде дерева структура всей системы контроля доступа. Дерево строится как иерархия вида «*Мониторинг – Сервер контроллеров – Каналы – Контроллеры*». Дерево показывает текущее состояние отдельных компонентов СКУД и позволяет управлять ими.

- Для *Мониторинга* указывается имя ПК (или IP адрес) ПК *Мониторинга*.
- Для *Сервера контроллеров* указывается имя ПК (или IP адрес) драйвера оборудования.
- Для *Канала* – номер сом-порта или IP адрес.
- Для *Контроллера* – адрес контроллера[17].

Голубой цвет значка *Сервера контроллеров* говорит о наличии связи с ним. Красный – об отсутствие связи.

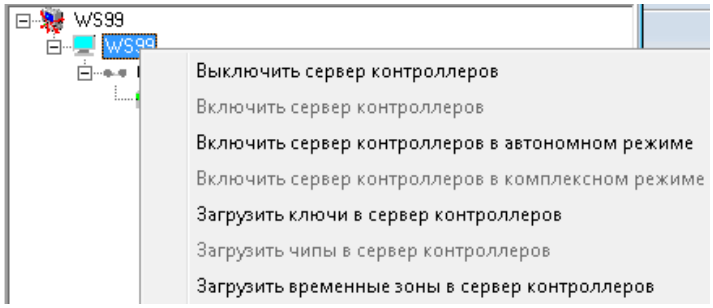
Зеленый цвет значка *Контроллера* говорит о его работе в комплексном режиме. Красный означает, что либо с контроллером нет связи, либо он выключен. Желтый – контроллер находится в автономном режиме с вычитыванием событий[18].

Всплывающее меню на узле *Мониторинг* позволяет выполнять следующие действия:



- Остановить мониторинг – останавливает службу Мониторинга (только при правах администратора);
- Запустить мониторинг – соответственно запускает службу мониторинга;
- Временно отключить внешний алгоритм – временно выключает работу внешнего алгоритма, до перезапуска службы мониторинг;
- Постоянно отключить внешний алгоритм – отключает работу внешнего алгоритма;
- Восстановить внешний алгоритм – восстанавливет работу внешнего алгоритма после отключения.

Всплывающее меню на узле *Сервер контроллеров* позволяет выполнять следующие действия:



- *Выключить Сервер контроллеров* – отсоединиться от выбранного *Сервера контроллеров*, что приведет перевод всех контроллеров данного Сервера в автономный режим.
- *Включить Сервер контроллеров* – восстановить связь с *Сервером контроллеров*, что автоматически переведет все контроллеры данного *Сервера* в комплексный режим.
- *Включить Сервер контроллеров в автономном режиме* – перевести все контроллеры данного *Сервера* в автономный режим. При этом связь с *Сервером* не разрывается, управление СКУД передается внутренней программе контроллеров, события от контроллеров продолжают поступать в *Мониторинг* и записываться в *Системный журнал*.

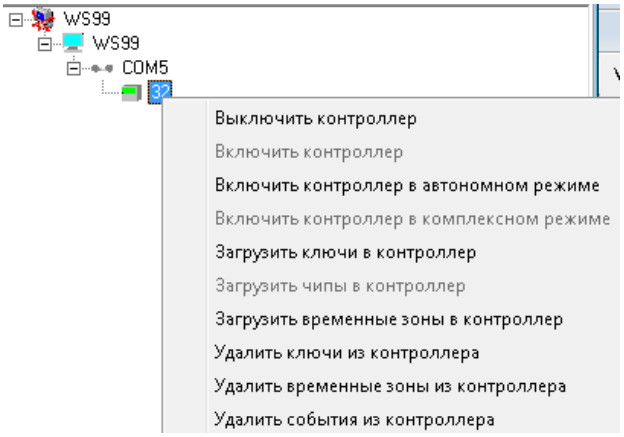
[17] Адрес контроллера должен быть уникален для канала (в отличие от предыдущих версий СКУД).

[18] Обратите внимание – чаще всего желтый цвет контроллера свидетельствует о процессе перехода его из автономного в комплексный режим. При этом, пока контроллер не вычитает все события из своей памяти, *Мониторинг* не переведет его в комплексный режим.

- *Включить Сервер контроллеров в комплексном режиме* – перевести все контроллеры данного *Сервера* в комплексный режим.
- *Загрузить ключи в Сервер контроллеров* – загрузить ключи во все контроллеры данного *Сервера*[19].
- *Загрузить чипы в Сервер контроллеров* – загрузить охранных чипов во все контроллеры данного *Сервера*[20].
- *Загрузить временные зоны* в *Сервер контроллеров* – загрузить расписание во все контроллеры данного *Сервера*[21].

Всплывающее меню на узле *Канал* имеет один единственный пункт *Показать скорость опроса*, при выборе которого включается отображение скорости опроса на графике.

Всплывающее меню на узле *Контроллер* позволяет выполнять следующие действия:



- *Выключить контроллер* – исключение данного контроллера из системы, что приводит к автоматическому переводу его в автономный режим.
- *Включить контроллер* – восстановить связь с контроллером, что автоматически переведет его в комплексный режим.
- *Включить контроллер в автономном режиме* – перевод данного контроллера в автономный режим. При этом связь с ним не разрывается, управление СКУД передается внутренней программе контроллеров, события от него продолжают поступать в *Мониторинг* и записываться в *Системный журнал*.
- *Включить контроллер в комплексном режиме* – перевод контроллера в комплексный режим.
- *Загрузить ключи в Контроллер* – загрузка ключей в данный контроллер.
- *Загрузить чипы в Контроллер* – загрузка охранных чипов ключей в данный контроллер (для охранных контроллеров);
- *Загрузить временные зоны в Контроллер* – загрузка расписания в данный контроллер.
- *Удалить ключи из Контроллера* – очистить базу ключей данного контроллера.
- *Удалить временные зоны из Контроллера* – очистить расписания данного контроллера.
- *Удалить события из Контроллера* – очистить базу событий данного контроллера.

5.4. Работа с *Мониторингом*

5.4.1. Загрузка данных в память контроллера

Прежде всего, следует напомнить, что загрузка данных в память контроллера выполняется для обеспечения автономного режима его работы. В штатном, т.е. комплексном, режиме ни память, ни внутренние алгоритмы программы контроллера не задействуются.

Загрузка данных в контроллер – это, по сути, синхронизация двух баз данных: контроллера и компьютера. Синхронизируются следующие данные:

- Права доступа.

[19] Подробнее смотрите раздел Загрузка прав доступа.

[20] Подробнее смотрите раздел Загрузка чипов.

[21] Подробнее смотрите раздел Загрузка расписания.

- Расписание.
- Коды охранных чипов.

5.4.2. Загрузка прав доступа

При загрузке прав доступа в контроллере прописываются следующие данные:

- Код ключа.
- Код цифровой клавиатуры.
- Пространственные ограничения (разрешенные пункты прохода для данного контроллера).
- Временны**е** ограничения (номер расписания).

Код ключа – это строка из двенадцати символов, представляющая шестнадцатиричное число (код в Wiegand формате). Это код электронной карты, который считывается с нее и передается на контроллер. Данный код не совпадает со строкой символов, которая часто печатается на самой карте.

Код цифровой клавиатуры – строка из четырех шестнадцатиричных символов. Он формируется при использовании в системы специальных считывателей с цифровой клавиатурой (keypad). Хранится в строке кода ключа в старших четырех байтах[22].

Пространственные ограничения прописываются в виде атрибута кода ключа и представляют собой т.н. маску портов – т.е. запрет или разрешение прохода по каждому из восьми портов контроллера.

Номер расписания задается как атрибут ключа в виде десятичного числа от 0 до 15, являющегося, по сути, ссылкой на загруженное в контроллер расписание (временную зону).

Реализуется загрузка прав доступа двумя способами:

- Массовой загрузкой всей базы ключей.
- Загрузкой одного ключа (автоматическая загрузка ключей).

В контроллеры старого типа (201 серии) может производиться только массовая загрузка.

Массовую загрузку рекомендуется применять в следующих случаях:

- Для контроллеров 201 серии.
- Для ускорения процесса автозагрузки ключей при первоначальном создании базы пользователей или при массовом изменении их прав доступа.
- Для автоматической перезагрузки базы ключей в заданное время[23].

В ручном режиме массовая загрузка выполняется клавишей *Загрузить ключи*.

Перед началом загрузки вся база ключей контроллера очищается. Во время ее контроллер на короткое[24] время переводиться в автономный режим.

Автоматическая загрузка ключей – это режим автоматического занесения прав доступа в контроллер при их вводе в карточку в модуле *Бюро пропусков*.

При нажатой на форме *Консоли Мониторинга* кнопке “Авт. загрузка ключей” (включена по умолчанию) будет происходить автоматическое занесение (обновление) кодов ключей в контроллер при их вводе (изменении) в базе *Персонала*. Выполняется этот процесс следующим образом:

---

[22] При 26-ти битной кодировке заняты младшие 8 байт кода ключа из 12.

[23] Время задается программой *Редактор настроек* (закладка *Мониторинг*, параметр *Время полной перезагрузки ключей в контроллеры*). По умолчанию эта функция отключена.

[24] От несколько секунд при загрузке сотен ключей до 2-3 минут при загрузке полной базы (65000 ключей).

- Любое изменения данных по правам доступа сотрудника (изменение кода ключа, маршрута, расписания и пр.) фиксируется в базе данных в буферной таблице[25].
- Периодически[26] *Мониторинг* перечитывает упомянутую таблицу и, если в ней есть запись, передает ее *Серверу контроллеров*.
- *Сервер контроллеров* корректирует данные в памяти контроллера.
- Запись из буферной таблицы удаляется.

Обратите внимание, что при большом числе изменений сведений по персоналу, описанный выше процесс может продолжаться довольно долго. Это приведет к тому, что при внезапном переводе системы в автономный режим некоторые сотрудники не смогут проходить через пункты прохода, поскольку их данные еще не попали в память контроллера. В подобной ситуации целесообразно выполнить вышеописанную полную загрузку ключей. По окончании загрузки буферная таблица очищается.

Подчеркнем, что при выполнении полной загрузки ключей (клавишей *Загрузить ключи*), буферная таблица очищается полностью. Именно поэтому мы рекомендуем после массовых изменений в базе *Персонала*, выполнить полную загрузку ключей, что значительно ускорит процесс занесения кодов в память контроллера.

5.4.3. Загрузка расписания

Загрузка расписания (временн***ы***х зон) выполняется только в ручном режиме клавишей *Загрузить временн**ы**е зоны*. Еще раз напомним, что в контроллер грузятся только недельные расписания (Временн***ы***е зоны, создаваемые в программе *Бюро пропусков*) и только первые шестнадцать расписаний.

В контроллере нет специального режима контроля временн***ы***х ограничений. Они вступают в силу, как только Вы загрузили расписания. Поэтому в автономном режиме может формироваться «неправильный» запрет для прохода ключа по времени в следующих случаях:

- Временн***а***я зона, присвоенная ключу, не загружена в контроллер. Подчеркнем, что вновь созданное расписание в программе *Бюро пропусков* подлежит обязательной ручной загрузке в контроллеры.
- Номер временн***о***й зоны, присвоенной ключу, превышает число 16, почему она в принципе не может быть загружена в контроллер.

В случае возникновения подобных проблем рекомендуется либо выполнить загрузку расписаний, либо отменить временн***ы***е ограничения для автономного режима посредством пункта всплывающего меню *Удалить временн**ы**е зоны из контроллера*.

При использовании сложных временн***ы***х ограничений (более 16 расписаний, сменные и индивидуальные графики, дополнительные *права на двери*) во избежание вышеописанных проблем не рекомендуется загружать расписания в контроллеры.

5.4.4. Загрузка чипов

Загрузка адресов чипов производится в память сигнального контроллера марки TSS. Эту операцию необходимо производить после первого запуска *Мониторинга*, а также после каждого переконфигурирования Системы (при добавлении или исключении чипов сигнальных контроллеров). При этом инициируется процесс передачи списка кодов чипов *Серверу контроллеров*. Кнопка неактивна, если в конфигурации не прописаны сигнальные контроллеры.

---

[25] Таблица *Reloadkluch*.

[26] Период задается программой *Редактор настроек* (закладка *Мониторинг*, параметр *Период просмотра базы ключей*). По умолчанию - 30000 мсек.

5.4.5. Режимы работы контроллеров

Система допускает три режима работы с контроллерами СКУД:

Комплексный – контроллер управляется *Мониторингом*. В этом случае он является ретранслятором: передает события (например, код ключа) и исполняет команды (включить реле). При этом обеспечивается полная функциональность системы.

Автономный – контроллер работает самостоятельно без связи с *Мониторингом*. Он использует свою внутреннюю базу ключей, маршрутов и расписаний. События записываются в память, но в систему не передаются. Для *Мониторинга* (и пользовательских программ) данного контроллера и обслуживаемых им пунктов прохода не существует. При этом обеспечивается только основная функциональность: контроль кода ключа, маршрутов и расписаний.

Полуавтономный – промежуточный режим, при котором контроллер принимает решения самостоятельно, но все события передает *Мониторингу*.

Основным режимом в большинстве случаев является комплексный.

Автономный режим считается аварийным. Контроллеры переходят в него автоматически в случае более чем 4-х секундной паузы неактивности *Мониторинга*. *Консоль Мониторинга* также позволяет вручную перевести любой контроллер в автоном (‘выключить контроллер’).

Полуавтономный режим штатно используется системой при переходе из автономного режима в комплексный, а именно, во время вычитывания событий из контроллера. По завершении этой операции контроллер переводится в комплекс.

Эту операцию можно выполнить и вручную из *Консоли Мониторинга*.

Однако в ряде случаев необходимо указывать способ работы с отдельными контроллерами на этапе загрузки СКУД. Так, в случае подключения контроллера через ЛВС (IP канал) из-за загрузки сети возможны значительные задержки времени передачи данных и, как следствие, задержки времени открытия двери на объекте[27].

Для таких контроллеров можно задать режим их работы в составе системы с помощью программы *Конфигуратор*.

[27] Особенно это касается удаленных объектов с низкоскоростными линиями связи.

6. Системный журнал

6.1. Общее описание

Программа ведения протокола событий (*Writerlog.exe*) относится к программам ядра Системы и устанавливается на сервере СКУД.

Программа реализована как служба, которая инсталлируется при установке программного комплекса. Старт сервиса инициируется *Системой управления* TSSACSGMSServer.

Протокол событий является важнейшим элементом СКУД. На основе протокола (*Системного журнала*) выполняется контроль работы Системы в целом, функционирования отдельных подсистем и программных модулей, состояния охраняемых СКУД объектов, передвижения персонала.

На основе *Системного журнала* с помощью стандартных программ отчетов СКУД создаются различные формы протоколов событий. На этих данных также основана работа *Системы учета рабочего времени*[28].

Напомним, что события автономного режима работы сохраняются в памяти контроллеров[29], и после старта ПО СКУД (т.е. перевода Системы в комплексный режим) вычитываются и помещаются в *Системном журнале*.

Описываемая программа также производит архивацию *Системного журнала* (как по достижению определенного количества записей, так и по наступлению заданного времени и даты).

Системный журнал (как текущий, так и его архивы) – это база данных ACS_LOG.FDB. Программа позволяет также вести параллельный протокол событий в формате DBF (syslog.dbf и его архивы), т.е. в формате предыдущей (4) версии ПО[30].

Настройки службы выполняются следующим образом:

- Редактированием файла начальной загрузки *Writerlog.ini*. Смотрите раздел Описание загрузочных параметров.
- С помощью программы *Консоль Системного журнала* (настройка основных параметров). Смотрите раздел Работа с программой Консоль Мониторинга настоящего документа.
- С помощью программы *Редактор настроек* (тонкая настройка системы). Смотрите документ *Редактирование параметров*.

6.2. Описание загрузочных параметров

Перед началом работы необходимо выставить системные параметры, используемые при загрузке программы. Все системные параметры описываются в файле *Writerlog.ini* (содержимое файла выделено курсивом). Не используемые и зарезервированные параметры не описываются.

```
[Options]

ALIASBASE =@ACS_LOG
```

[28] Система учета рабочего времени *TSS2000 TimeKeeper* не входит в базовую поставку и приобретается дополнительно.

[29] Разные типы контроллеров имеют разную память на события. Подробнее об этом смотрите в документе *Общее описание СКУД*.

[30] Возможность записи журнала формата предыдущей версии ПО оставлена для совместимости с пользовательскими программами отчетов, ориентированных на dbase формат таблиц.

Указывается имя алиаса, по которому располагается база данных Системы.

APPSERVER=TSS

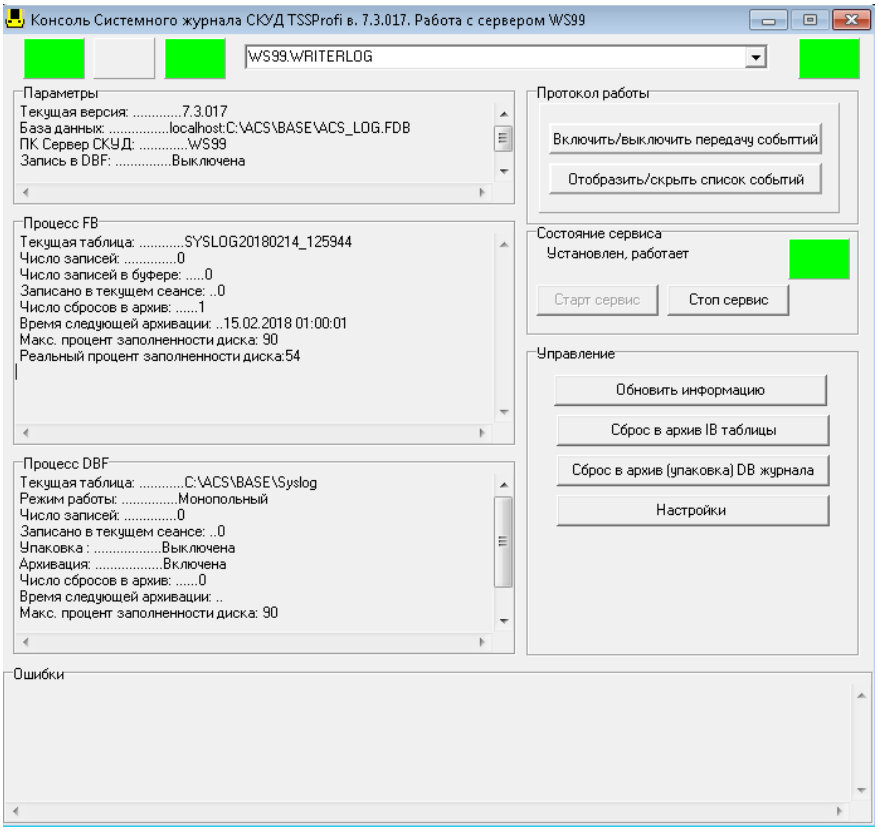
Имя компьютера, на котором работает программа *Транспорт*.

Прочие параметры хранятся в системном реестре Windows (HKEY_LOCAL_MACHINE\SOFTWARE\@ACS\ Writerlog \A). Настройка их выполняется частично *Консолью Мониторинга*, частично программой *Редактор настроек*.

6.3. Работа с программой Консоль Системного журнала

6.3.1. Интерфейс

Как уже говорилось, служба ядра *Системный журнал* не имеет визуальной формы. Управляется служба программой *Консоль Системного журнала* (*WriterlogConsole.exe*).



Программа *Консоль Системного журнала* может работать на любой рабочей станции ЛВС, имеющей связь с Сервером СКУД. Однако возможность старта и остановки службы *Системного журнала* будет доступна только при их работе на одном ПК.

Окно *Консоли Системного журнала* имеет вид, показанный на рисунке. На нем расположены следующие элементы отображения и управления.

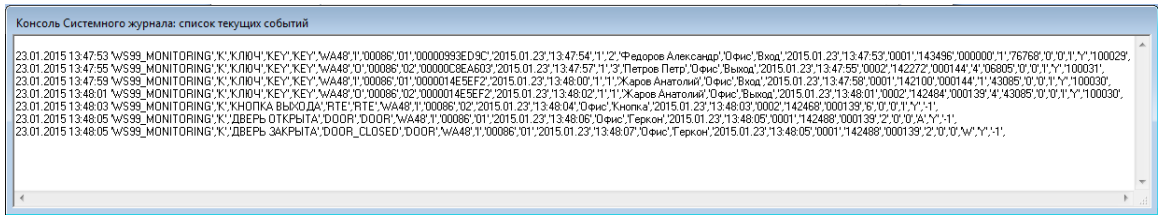
В заголовке программного окна приводится номер версии СКУД и указывается, на каком ПК работает служба *Системный журнал* (как правило, это Сервер СКУД).

В верхней части окна представлена текущая информация о работе комплекса. Четыре панельки определяют (слева направо):

- Работоспособность *Системного журнала*.
- Наличие ошибок в работе *Системного журнала*.
- Состояние связи с *Системным журналом*.
- Состояние процесса репликации: включена или выключена

Зеленый цвет панельки говорит о нормальном состоянии, красный – о сбое.

Клавиши на панели *Протокол работы* позволяют включать или отключать передачу и отображение всех событий системы. При нажатии клавиши *Отобразить…* отображается окно с информацией о событиях СКУД:



На панели *Параметры* приведены некоторые общие данные о СКУД.

Панель *Процесс FB* отображает текущее состояние основной базы *Системного журнала* (т.е. формата Firebird).

Панель *Процесс DBF* отображает текущее состояние опциональной базы *Системного журнала* (т.е. формата Dbase IV).

На панель *Ошибки* выводятся критичные ошибки работы службы.

Панель *Состояние сервиса* показывает текущее состояние службы *Системного журнала*, а также позволяет остановить и запустить его. Данная операция доступна только тогда, когда *Консоль* запускается на том же ПК, на котором работает служба. Обратите внимание, что при работающем сервисе *Система управления* остановка *Системного журнала* достаточно бессмысленна – служба управления автоматически стартует его.

Клавиши панели *Управление* выполняют следующие функции:

Обновить – обновляет текущую информацию панелей *Процесс FB* и *Процесс DBF*.

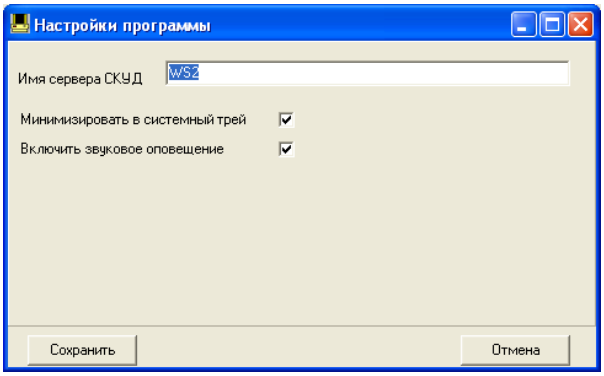
Передавать протокол работы – включение (выключение) трансляции протокола событий.

Сброс в архив IB таблицы – ручное архивирование текущей таблицы журнала (подробнее о системе архивации данных смотрите соответствующий раздел).

Сброс в архив (упаковка) DB журнала – ручное архивирование DBF таблицы.

Настройки программы – редактирование параметров работы *Консоли*.

6.3.2. Программные настройки



В окне *Настройки программы* (одноименная клавиша) доступны для изменения следующие параметры работы *Консоли Системного журнала*:

Имя сервера СКУД – имя ПК Сервера системы или точнее, ПК, на котором работает программа *Системный журнал*.

Минимизировать в системный трей – при включенной опции сворачивание программного окна будет производиться в нижний правый угол рабочего стола Windows.

Включить звуковое оповещение – при включенной опции критические ошибки в работе службы будут сопровождаться звуковым сообщением[31].

[31] Звуковые файлы должны располагаться во вложенной папке Voice.

6.4. Настройка и работа с программой

6.4.1. Настройка

При инсталляции ПО обеспечивается настройка работы *Системного журнала* на запись событий только в формате Firebird в базу ACS_LOG.FDB. При желании вести параллельный протокол событий в dbase формате необходимо включить эту опцию в программе Редактор установок (ParamsEdit) на закладке *Системный журнал*. Помните, что ведение двух протоколов несколько снижает скорость работы программы.

Одним из важных процессов при ведении протокола событий является т.н. архивация Системного журнала. В данном случае под архивацией понимается процедура слежения за размером текущей таблицы журнала, с целью его (размера) ограничения. Этот процесс не следует путать с ведением долговременного архива журнала, которым занимается программа *Менеджер Системного журнала* (SyslogManager).

Настройка параметров архивации главного и параллельного журнала описана ниже.

6.4.2. Архивация FB журнала

При первом старте программы в базе ACS_LOG создается таблица с именем, включающим в себя дату и время создания, которая и становится текущим системным журналом. При выполнении архивации (по числу записей или по времени) текущая таблица закрывается и создается новая по тем же правилам, в которую и продолжается запись событий. Для FB журнала отсутствует понятие монопольного доступа.

При установке комплекса назначаются следующие параметры архивации:

- Сброс в архив производится по понедельникам в 1 час ночи.
- Первый сброс будет выполнен в ближайший понедельник после установки ПО.
- При старте программы после пропущенной архивации будет выполнен принудительный сброс.
- При нарушении процедуры архивации сброс будет произведен по достижении 100000 записей в текущем журнале.

Не рекомендуется хранить в базе *Системного журнала* данные более чем за год. Неиспользуемые для отчетов старые журналы следует или удалять или перемещать в долговременный архив с помощью утилиты *Менеджер системного журнала (SyslogManager)*. Выгруженные из базы данные вы можете переписать на другой носитель информации (компакт диск, сервер) и хранить сколь угодно долго.

Настройка параметров выполняется в программе Редактор установок (ParamsEdit) на закладке *Системный журнал*, где указывается периодичность перемещения журнала в архив. Период перемещения зависит от интенсивности заполнения журнала событиями. Оптимальный размер одного журнала (таблицы) – 100 тысяч записей.

Наиболее оптимальный режим архивации – сброс журнала по времени. В зависимости от интенсивности событий периодичность сброса может варьироваться от 1 до 7 и более дней. Назначать архивацию лучше на самый неактивный период работы СКУД, то есть на ночь. При этом следует разносить ее по времени с операциями резервного копирования и восстановления баз, а также с массовой загрузкой кодов ключей в контроллеры.

Значение *Граничное число записей* для базы FB используется как резервный ограничитель ее размера при сбое в системе архивации. В любом случае, он должен быть больше максимального числа событий, накапливающихся в журнале к моменту сброса. Перемещение в архив происходит по первому из возникших событий – либо по дате, либо по числу записей.

6.4.3. Архивация DBF журнала

Под архивацией понимается перемещение текущего *Системного журнала* в архивный каталог. Физически операция выглядит следующим образом:

1. Файл syslog.dbf переименовывается (имя соответствует дате и времени архивации) и перемещается в каталог ARC_SYSLOG.
2. Создается новый файл *Системного журнала.*

Операция переименования может выполняться только в монопольном режиме доступа к файлу.

В стандартном режиме программа должна держать файл *Системного журнала* в монопольном использовании. При этом ни одна программа не может открыть этот файл даже для чтения. Это является необходимым условием для гарантированности его архивации.

На самом деле, в случае разделенного доступа к файлу базы, перед началом архивации программа будет пытаться перевести таблицу в монопольный режим. В случае успеха (т.е. если таблица в данный момент никем не открыта) операция сброса в архив завершится корректно.

При крайней необходимости допускается запуск программы с разделяемым доступом к журналу. При этом становится доступной опция *Разделение*. Помните, что после работы с файлом из внешней программы Вы должны переключить режим доступа на монопольный.

Вы можете использовать Dbase таблицы для создания собственной системы отчетов, при этом, если вы возлагаете архивацию журналов на СКУД, то должны (на момент выполнения архивации) прекращать работу вашей программы (закрывать текущую таблицу Syslog.dbf). Вы можете сами следить за размером таблицы журнала и при вычитывании записи удалять последнюю. При этом рекомендуется включить опцию упаковки в программе Редактор установок (ParamsEdit) на закладке *Системный журнал*. Упаковка будет производиться программой *Системный журнал* с той же периодичностью, которая задана для его архивации. Для упаковки программы также необходимо обеспечить монопольный доступ к журналу.

6.4.4. Проблемы обеспечения целостности протокола событий

Программа записи протокола СКУД является такой же важной ее частью, как остальные программы ядра. При разрыве связи с программой (т.е. при невозможности писать протокол событий) *Сервер контроллеров* переводит Систему в автономный режим работы.

Перед запуском СКУД в рабочую эксплуатацию тщательно проверьте надежную работу программы. Информация о неполадках записывается в файл протокола самой программы (каталог ACS/LOG). При сбоях в работе внимательно просмотрите его.

Мы рекомендуем следующие меры для повышения надежности сохранения данных СКУД:

- Установить время архивации журнала в период минимальной загруженности Системы.
- Стараться, чтобы число записей в журнале не превышало 100000.
- Периодически (желательно **до** архивации) выполнять процедуру резервного копирования базы (Backup-restore).
- Настроить программу *Мониторинг* на запись событий от контроллеров в файл трассировки. При возникновении проблем с сохранностью данных можно будет воспользоваться соответствующей утилитой (*SyslogMaker*) для перенесения «сырых» данных в формат системного журнала.

7. Транспорт

7.1. Принципы работы

Программа *Транспорт* (transsrv) служит для обеспечения межмодульного взаимодействия комплекса СКУД. Взаимодействие может выполняться по двум каналам:

- Socket (т.е. по протоколу TCP\IP),
- Pipe.

Первый способ должен использоваться при расположении клиента и сервера *Транспорта* на разных ПК. Второй предпочтителен, когда и клиент и сервер находятся на одной машине. Однако практика его использования показала, что канал *Pipe* зависит от ограничений работы пользователя ПК с файловой системой. Поэтому применять его рекомендуется только после проверочного прогона системы с этим параметром. Понятно, что *Транспорт* обеспечивает работу клиентов безотносительно их способа подключения.

Установка *Транспорта* выполняется во время инсталляции комплекса. Местонахождение клиента и сервера осуществляется автоматически (т.е. на одном ПК будет выбран канал Pipe, на разных – Socket)

Программа стартует как сервис в автоматическом режиме.

Все программы комплекса СКУД при старте должны зарегистрироваться на *Транспорте* системы сообразно заданным при их настройки параметрам (о чем будет сказано чуть позже). *Транспорт* регистрирует своих клиентов, следит за их работоспособностью (по посылкам «ОК») и передает данные, предназначенные конкретным процессам.

Сам *Транспорт* работает как сервис и не имеет средств отображения. Ряд параметров, необходимых для его работы, задается при инсталляции сервиса, ряд при помощи консольного приложения *Консоль транспорта* (файл tsfe.exe).

Для обеспечения гарантированной работы *Транспорта* служит сервис TSSTransport Guardian (TSSTransGuard), который следит за его корректным функционированием и, при необходимости, рестартует его.

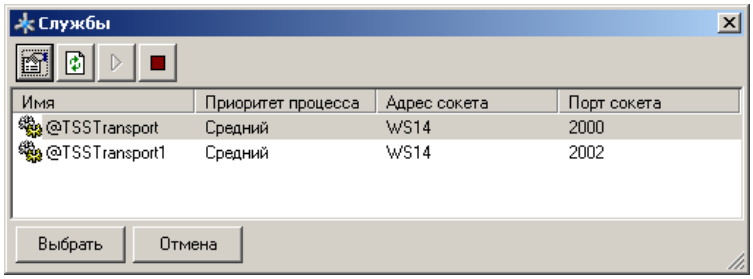
При работе *Транспорт* ведет собственный протокол событий. Вид файла протокола представлен на рисунке. При каждом сеансе работы Транспорта создается новый протокол. Имя файла состоит из префикса TSSTransport, даты и времени его создания, например:

TSSTransport_2006-03-11_11-13-46.362.log.txt

Файл протокола расположен в каталоге ACS\LOG. Как и протоколы программ ядра, эти файлы необходимо периодически удалять.

7.2. Консоль Транспорта

Ряд настроек *Транспорта* позволяет выполнить программа *Консоль транспорта* (tsfe.exe). Она также отображает текущее состояние клиентов *Транспорта*.



При старте программы отображается окно работающих сервисов *Транспорта*.

Информация о сервисах представлена в виде таблицы, в столбцах которой указаны:

- Windows приоритет процесса,
- Имя ПК Транспорта,

- Адрес порта.

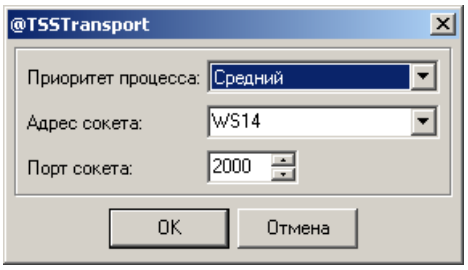
В приведенном на рисунке примере в системе работают два *Транспорта*. Первый обслуживает локальный СКУД (порт 2000). Второй – систему синхронизации (порт 2002).

Обратите внимание, что указаны только параметры сокетного соединения. Параметры подключения по каналу Pipe указываются только на стороне клиентов.

На панели инструментов расположены следующие клавиши:

- Задание свойств.
- Обновление – обновление списка процессов.
- Старт сервиса.
- Остановка сервиса.

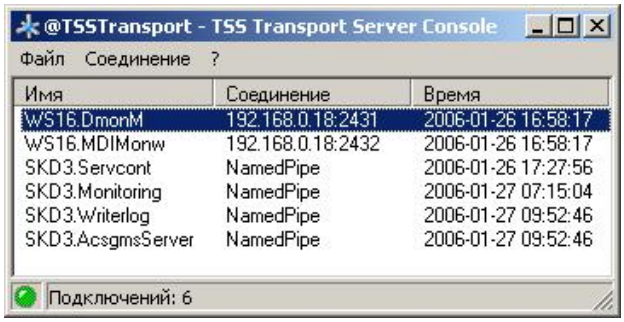
Все упомянутые действия относятся к текущему (выбранному в списке) *Транспорту*.



Окно *Свойства* позволяет переопределить следующие параметры:

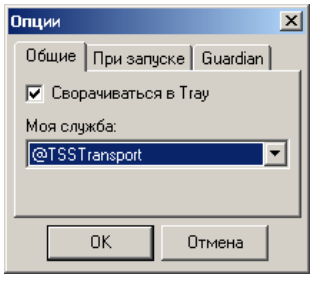
- Системный приоритет сервиса. Значения выбираются из списка.
- Адрес сокета (IP адрес или сетевое имя ПК)
- Порт сокета.

Двойной клик на строчке соответствующего *Транспорта* (или нажатие клавиши *Выбрать*) вызывает окно процессов, зарегистрированных на данном *Транспорте*. Окно имеет следующий вид:



В таблице процессов отображаются:

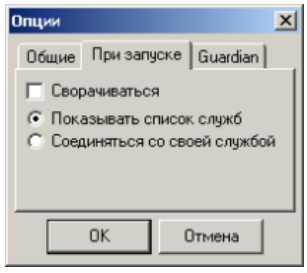
- Имя процесса в нотации <имя ПК>.<имя процесса>.
- Тип соединения. «NamedPipe» для канала Pipe. IP адрес (точнее, IP:port) для канала Socket.
- Дата и время регистрации клиента на *Транспорте*.



Меню *Файл* - *Опции* позволяет вызвать окно настроек для отображения параметров данного *Транспорта*. Окно состоит из трех вкладок.

На первой (*Общие параметры*) указывается:

- Опция *Сворачиваться в Tray*. При ее включении, нажатие на значок Минимизировать окно, последнее будет отображаться в системном трее[32] в виде значка  .
- В поле *Моя служба* можно выбрать из списка запущенных сервисов отличный от исходного *Транспорт*.



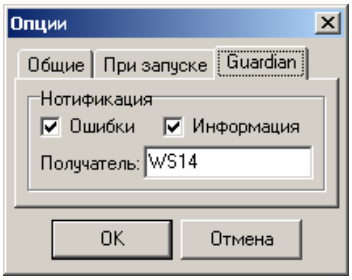
Вкладка *Параметры при запуске* позволяет задать следующие настройки:

- При выборе опции *Сворачиваться*, окно списка *Консоли* сразу свернется с системный трей.
- При выборе типа старта *Показывать список служб Консоль* при старте будет отображать окно служб (приведен-

[32] Т.е. правая часть панели задач Windows. По-русски ее иногда называют «системный лоток».

ное в начале данного раздела). При выборе *Соединяться со своей службой* после старта сразу будет отображаться окно списка процессов.

И, наконец, вкладка *Guardian* отображает настройки службы слежения за соответствующим *Транспортом*, а именно:



- Нотификация – передача по сети только сообщений об ошибках или всей информации. По умолчания протокол событий не ведется.
- Получатель – адресат для передачи сообщений по локальной сети. Можно указывать сетевое имя ПК, либо (для доменной архитектуры) имя пользователя.

7.3. Установка и настройка *Транспорта*

Как уже говорилось, сервис устанавливается автоматически при инсталляции комплекса с дистрибутивного диска. При этом ему задается имя TSSTransport и устанавливается рабочий порт 2000.

Обратите внимание, что все клиенты СКУД (по сокетному каналу) коннектятся к *Транспорту* именно по этому порту.

При установке сервиса вручную (например, при настройке *Системы Синхронизации локальных комплексов СКУД*) следует использовать следующий формат:

Transsrv.exe <список параметров>, где список параметров представляет из себя:

- **/install** – инсталляция, доступны следующие 4 параметра:
  - **/start** – запуск сервиса;
  - **/name**=<имя_сервиса>, где <имя_сервиса> – имя сервиса под которым он будет установлен, эта возможность позволяет установить несколько сервисов на один компьютер, чтобы получить такую возможность нужно: либо переименовать исполняемы файл в той же папке, либо скопировать в другую папку. Лог-файл имеет вид «TssTransport_d_t.log.txt» в случае, если этот параметр не был задан явно и в случае если этот параметр задан, то – «<имя_сервиса>_d_t.log.txt»; здесь
    - d – дата в формате гггг-мм-дд,
    - t – время в формате чч-мм-сс (секунды указаны до тысячной доли).
  - /**port**=<порт_сокета>, где <порт_сокета> – порт сокета на котором сервер будет слушать входящие соединения;
  - /**AutoStart** – если указано, то сервис устанавливается в автозагрузку, иначе вручную.
- /**uninstall** – остановка сервиса (если работает) и деинсталляция;
- /**start** – запуск сервиса;
- /**stop** – остановка сервиса.

Приоритет *Транспорта* всегда рекомендуется устанавливать высоким.

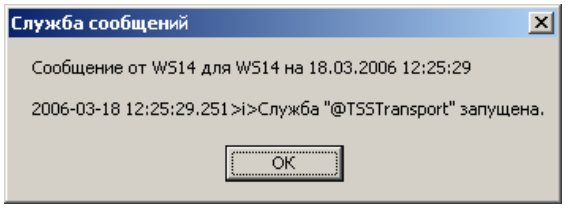
7.4. Сервис слежения за Транспортом (Guardian)

Для гарантированной работы *Трасноспорта* СКУД служит сервис слежения TSSTransport Guardian (модуль transgrd.exe). Он устанавливается и стартует при инсталляции программного комплекса СКУД.

Назначение службы – следить за работоспособностью *Транспорта* системы и при необходимости стартовать его. Необходимые для него работы настройки описаны в предыдущем разделе.

Служба ведет протокол своей работы (файл transgrd.log.txt в каталоге ACS\LOG).

Служба также может рассылать сообщения пользователям локальной сети, например:



7.5. Настройка клиентов *Транспорта*

Еще раз напомним, что по умолчанию все клиенты, работающие на одном с *Транспортом* ПК соединяются с ним по каналу Pipe, на разных – по каналу Socket.

Для программ ядра СКУД параметры соединения с *Транспортом* задаются в соответствующих ключах реестра. Редактируются они программой *Редактор установок*. Изменению подлежат:

- *Имя ПК Сервера* – имя компьютера, на котором функционирует Транспорт.
- *Протокол* – тип соединения (Socket или Pipe).
- *ОК интервал* – интервал посылки сообщения ОК
- *ОК фактор* – множитель для определения полного времени разрыва связи.

Как уже говорилось, сервер *Транспорта* определяет наличие на линии клиентов по передаче им условной посылки типа ОК. Если процесс не ответил за условленное время, *Транспорт* исключает его из списка клиентов (кстати, данный параметр является уникальным для каждого процесса – при коннекте процесс передает серверу значение своего ОК интервала). **Под условленным временем понимается произведение параметров *ОК интервал* и *ОК фактор***

Однако иногда сервер может не успеть получить ОК от корректно работающего клиента в обозначенное время. Это происходит по причине загруженности обработкой данных самого процесса, загрузкой процессора другими программами, высоким трафиком ЛВС. Поэтому, на ресурсоемких процессах ядра, работающих на одной с *Транспортом* машине, имеет смысл увеличивать этот интервал до 2-4 секунд. Для клиентов на сетевых ПК этот параметр принудительно установлен в 10 секунд.

Имя ПК транспорта для приложений СКУД (кроме программ ядра) указывается в настроечных файлах (с расширением ini) для каждой программы в строке APPSERVER. Допускается указывать либо IP адрес, либо сетевое имя ПК.

Как уже говорилось, во время работы *Транспорт* ведет протокол событий. В протоколе фиксируются подключения и отключения клиентов, а также возникающие ошибки. Аналогичный протокол ведется и на стороне клиентов. При возникновении проблем в работе соединений рекомендуется внимательно изучить серверный и клиентский протоколы. Коды возможных сокетных ошибок приведены в Приложении 1.

8. Приложение 1 Коды ошибок сокетного соединения

Windows Sockets Error Codes

The following list describes the possible error codes returned by the **WSAGetLastError** function. Errors are listed in alphabetical order by error macro. Some error codes defined in Winsock2.h are not returned from any function—these are not included in this list.

Return code/value	Description
WSAEINTR 10004	*Interrupted function call*. A blocking operation was interrupted by a call to WSACancelBlockingCall.
WSAEACCES 10013	*Permission denied.* An attempt was made to access a socket in a way forbidden by its access permissions. An example is using a broadcast address for **sendto** without broadcast permission being set using **setsockopt**(SO_BROADCAST). Another possible reason for the WSAEACCES error is that when the **bind** function is called (on Windows NT 4 SP4 or later), another application, service, or kernel mode driver is bound to the same address with exclusive access. Such exclusive access is a new feature of Windows NT 4 SP4 and later, and is implemented by using the SO_EXCLUSIVEADDRUSE option.
WSAEFAULT 10014	*Bad address.* The system detected an invalid pointer address in attempting to use a pointer argument of a call. This error occurs if an application passes an invalid pointer value, or if the length of the buffer is too small. For instance, if the length of an argument, which is a **sockaddr** structure, is smaller than the sizeof(sockaddr).
WSAEINVAL 10022	*Invalid argument.* Some invalid argument was supplied (for example, specifying an invalid level to the **setsockopt** function). In some instances, it also refers to the current state of the socket—for instance, calling **accept** on a socket that is not listening.
WSAEMFILE 10024	*Too many open files*. Too many open sockets. Each implementation may have a maximum number of socket handles available, either globally, per process, or per thread.
WSAEWOULDBLOCK 10035	*Resource temporarily unavailable*. This error is returned from operations on non-blocking sockets that cannot be completed immediately, for example **recv** when no data is queued

	to be read from the socket. It is a nonfatal error, and the operation should be retried later. It is normal for WSAEWOULDBLOCK to be reported as the result from calling **connect** on a nonblocking SOCK_STREAM socket, since some time must elapse for the connection to be established.
WSAEINPROGRESS 10036	*Operation now in progress*. A blocking operation is currently executing. Windows Sockets only allows a single blocking operation—per- task or thread—to be outstanding, and if any other function call is made (whether or not it references that or any other socket) the function fails with the WSAEINPROGRESS error.
WSAEALREADY 10037	*Operation already in progress*. An operation was attempted on a nonblocking socket with an operation already in progress—that is, calling **connect** a second time on a nonblocking socket that is already connecting, or canceling an asynchronous request (**WSAAsyncGetXbyY**) that has already been canceled or completed.
WSAENOTSOCK 10038	*Socket operation on nonsocket*. An operation was attempted on something that is not a socket. Either the socket handle parameter did not reference a valid socket, or for **select**, a member of an **fd_set** was not valid.
WSAEDESTADDRREQ 10039	*Destination address required*. A required address was omitted from an operation on a socket. For example, this error is returned if **sendto** is called with the remote address of ADDR_ANY.
WSAEMSGSIZE 10040	*Message too long*. A message sent on a datagram socket was larger than the internal message buffer or some other network limit, or the buffer used to receive a datagram was smaller than the datagram itself.
WSAEPROTOTYPE 10041	*Protocol wrong type for socket*. A protocol was specified in the **socket** function call that does not support the semantics of the socket type requested. For example, the ARPA Internet UDP protocol cannot be specified with a socket type of SOCK_STREAM.
WSAENOPROTOOPT 10042	*Bad protocol option*. An unknown, invalid or unsupported option or level was specified in a **getsockopt** or **setsockopt** call.
WSAEPROTONOSUPPORT	*Protocol not supported*.

10043	The requested protocol has not been configured into the system, or no implementation for it exists. For example, a **socket** call requests a SOCK_DGRAM socket, but specifies a stream protocol.
WSAESOCKTNOSUPPORT 10044	*Socket type not supported.* The support for the specified socket type does not exist in this address family. For example, the optional type SOCK_RAW might be selected in a **socket** call, and the implementation does not support SOCK_RAW sockets at all.
WSAEOPNOTSUPP 10045	*Operation not supported.* The attempted operation is not supported for the type of object referenced. Usually this occurs when a socket descriptor to a socket that cannot support this operation is trying to accept a connection on a datagram socket.
WSAEPFNOSUPPORT 10046	*Protocol family not supported.* The protocol family has not been configured into the system or no implementation for it exists. This message has a slightly different meaning from WSAEAFNOSUPPORT. However, it is interchangeable in most cases, and all Windows Sockets functions that return one of these messages also specify WSAEAFNOSUPPORT.
WSAEAFNOSUPPORT 10047	*Address family not supported by protocol family.* An address incompatible with the requested protocol was used. All sockets are created with an associated address family (that is, AF_INET for Internet Protocols) and a generic protocol type (that is, SOCK_STREAM). This error is returned if an incorrect protocol is explicitly requested in the **socket** call, or if an address of the wrong family is used for a socket, for example, in **sendto**.
WSAEADDRINUSE 10048	*Address already in use.* Typically, only one usage of each socket address (protocol/IP address/port) is permitted. This error occurs if an application attempts to **bind** a socket to an IP address/port that has already been used for an existing socket, or a socket that was not closed properly, or one that is still in the process of closing. For server applications that need to **bind** multiple sockets to the same port number, consider using **setsockopt** (SO_REUSEADDR). Client applications usually need not call **bind** at all— **connect** chooses an unused port automatically. When **bind** is called with a wildcard address (involving ADDR_ANY), a WSAEADDRINUSE error could be delayed until

	the specific address is committed. This could happen with a call to another function later, including **connect**, **listen**, **WSAConnect**, or **WSAJoinLeaf**.
WSAEADDRNOTAVAIL 10049	*Cannot assign requested address*. The requested address is not valid in its context. This normally results from an attempt to **bind** to an address that is not valid for the local computer. This can also result from **connect**, **sendto**, **WSAConnect**, **WSAJoinLeaf**, or **WSASendTo** when the remote address or port is not valid for a remote computer (for example, address or port 0).
WSAENETDOWN 10050	*Network is down*. A socket operation encountered a dead network. This could indicate a serious failure of the network system (that is, the protocol stack that the Windows Sockets DLL runs over), the network interface, or the local network itself.
WSAENETUNREACH 10051	*Network is unreachable*. A socket operation was attempted to an unreachable network. This usually means the local software knows no route to reach the remote host.
WSAENETRESET 10052	*Network dropped connection on reset.* The connection has been broken due to keep-alive activity detecting a failure while the operation was in progress. It can also be returned by **setsockopt** if an attempt is made to set SO_KEEPALIVE on a connection that has already failed.
WSAECONNABORTED 10053	*Software caused connection abort.* An established connection was aborted by the software in your host computer, possibly due to a data transmission time-out or protocol error.
WSAECONNRESET 10054	*Connection reset by peer*. An existing connection was forcibly closed by the remote host. This normally results if the peer application on the remote host is suddenly stopped, the host is rebooted, the host or remote network interface is disabled, or the remote host uses a hard close (see **setsockopt** for more information on the SO_LINGER option on the remote socket). This error may also result if a connection was broken due to keep-alive activity detecting a failure while one or more operations are in progress. Operations that were in progress fail with WSAENETRESET. Subsequent operations fail with WSAECONNRESET.
WSAENOBUFS	*No buffer space available*.

10055	An operation on a socket could not be performed because the system lacked sufficient buffer space or because a queue was full.
WSAEISCONN 10056	*Socket is already connected*. A connect request was made on an already-connected socket. Some implementations also return this error if **sendto** is called on a connected SOCK_DGRAM socket (for SOCK_STREAM sockets, the *to* parameter in **sendto** is ignored) although other implementations treat this as a legal occurrence.
WSAENOTCONN 10057	*Socket is not connected*. A request to send or receive data was disallowed because the socket is not connected and (when sending on a datagram socket using **sendto**) no address was supplied. Any other type of operation might also return this error—for example, **setsockopt** setting SO_KEEPALIVE if the connection has been reset.
WSAESHUTDOWN 10058	*Cannot send after socket shutdown*. A request to send or receive data was disallowed because the socket had already been shut down in that direction with a previous **shutdown** call. By calling **shutdown** a partial close of a socket is requested, which is a signal that sending or receiving, or both have been discontinued.
WSAETIMEDOUT 10060	*Connection timed out.* A connection attempt failed because the connected party did not properly respond after a period of time, or the established connection failed because the connected host has failed to respond.
WSAECONNREFUSED 10061	*Connection refused*. No connection could be made because the target computer actively refused it. This usually results from trying to connect to a service that is inactive on the foreign host—that is, one with no server application running.
WSAEHOSTDOWN 10064	*Host is down*. A socket operation failed because the destination host is down. A socket operation encountered a dead host. Networking activity on the local host has not been initiated. These conditions are more likely to be indicated by the error WSAETIMEDOUT.
WSAEHOSTUNREACH 10065	*No route to host.* A socket operation was attempted to an un-

	reachable host. See WSAENETUNREACH.
WSAEPROCLIM 10067	*Too many processes*. A Windows Sockets implementation may have a limit on the number of applications that can use it simultaneously.**WSAStartup** may fail with this error if the limit has been reached.
WSASYSNOTREADY 10091	*Network subsystem is unavailable*. This error is returned by **WSAStartup** if the Windows Sockets implementation cannot function at this time because the underlying system it uses to provide network services is currently unavailable. Users should check: • That the appropriate Windows Sockets DLL file is in the current path. • That they are not trying to use more than one Windows Sockets implementation simultaneously. If there is more than one Winsock DLL on your system, be sure the first one in the path is appropriate for the network subsystem currently loaded. • The Windows Sockets implementation documentation to be sure all necessary components are currently installed and configured correctly.
WSAVERNOTSUPPORTED 10092	*Winsock.dll version out of range.* The current Windows Sockets implementation does not support the Windows Sockets specification version requested by the application. Check that no old Windows Sockets DLL files are being accessed.
WSANOTINITIALISED 10093	*Successful WSAStartup not yet performed*. Either the application has not called **WSAStartup** or **WSAStartup** failed. The application may be accessing a socket that the current active task does not own (that is, trying to share a socket between tasks), or **WSACleanup** has been called too many times.
WSAEDISCON 10101	*Graceful shutdown in progress.* Returned by **WSARecv** and **WSARecvFrom** to indicate that the remote party has initiated a graceful shutdown sequence.
WSATYPE_NOT_FOUND 10109	*Class type not found.* The specified class was not found.
WSAHOST_NOT_FOUND 11001	*Host not found*. No such host is known. The name is not an official host name or alias, or it cannot be found in the database(s) being queried. This error may also

	be returned for protocol and service queries, and means that the specified name could not be found in the relevant database.
WSATRY_AGAIN 11002	*Nonauthoritative host not found*. This is usually a temporary error during host name resolution and means that the local server did not receive a response from an authoritative server. A retry at some time later may be successful.
WSANO_RECOVERY 11003	*This is a nonrecoverable error*. This indicates that some sort of nonrecoverable error occurred during a database lookup. This may be because the database files (for example, BSD-compatible HOSTS, SERVICES, or PROTOCOLS files) could not be found, or a DNS request was returned by the server with a severe error.
WSANO_DATA 11004	*Valid name, no data record of requested type*. The requested name is valid and was found in the database, but it does not have the correct associated data being resolved for. The usual example for this is a host name-to-address translation attempt (using **gethostbyname** or **WSAAsyncGetHostByName**) which uses the DNS (Domain Name Server). An MX record is returned but no A record—indicating the host itself exists, but is not directly reachable.
WSA_INVALID_HANDLE OS dependent	*Specified event object handle is invalid*. An application attempts to use an event object, but the specified handle is not valid.
WSA_INVALID_PARAMETER OS dependent	*One or more parameters are invalid*. An application used a Windows Sockets function which directly maps to a Windows function. The Windows function is indicating a problem with one or more parameters.
WSA_IO_INCOMPLETE OS dependent	*Overlapped I/O event object not in signaled state*. The application has tried to determine the status of an overlapped operation which is not yet completed. Applications that use **WSAGetOverlappedResult** (with the *fWait* flag set to FALSE) in a polling mode to determine when an overlapped operation has completed, get this error code until the operation is complete.
WSA_IO_PENDING OS dependent	*Overlapped operations will complete later*. The application has initiated an overlapped operation that cannot be completed immediately. A completion indication will be given later when the operation has been completed.

WSA_NOT_ENOUGH_MEMORY OS dependent	*Insufficient memory available*. An application used a Windows Sockets function that directly maps to a Windows function. The Windows function is indicating a lack of required memory resources.
WSA_OPERATION_ABORTED OS dependent	*Overlapped operation aborted*. An overlapped operation was canceled due to the closure of the socket, or the execution of the SIO_FLUSH command in **WSAIoctl**.
WSAINVALIDPROCTABLE OS dependent	*Invalid procedure table from service provider*. A service provider returned a bogus procedure table to Ws2_32.dll. (This is usually caused by one or more of the function pointers being null.)
WSAINVALIDPROVIDER OS dependent	*Invalid service provider version number*. A service provider returned a version number other than 2.0.
WSAPROVIDERFAILEDINIT OS dependent	*Unable to initialize a service provider*. Either a service provider's DLL could not be loaded (**LoadLibrary** failed) or the provider's **WSPStartup**/**NSPStartup** function failed.
WSASYSCALLFAILURE OS dependent	*System call failure*. Generic error code, returned under various conditions. Returned when a system call that should never fail does fail. For example, if a call to **WaitForMultipleEvents** fails or one of the registry functions fails trying to manipulate the protocol/namespace catalogs. Returned when a provider does not return SUCCESS and does not provide an extended error code. Can indicate a service provider implementation error.