Номер документа TSS0319

ООО «Компания Семь печатей»

117216, Москва, ул. Феодосийская, д. 1, тел.(факс): (495)) 225 25 31

E-mail: info@sevenseals.ru Web-Page: http://www.sevenseals.ru

Интеграция систем безопасности

Интегратор

ДЛЯ ВЕРСИИ *7* **СКУД** *TSSPROFI*

Программное обеспечение

Часть I
Общее описание

руководство администратора

Москва

2008

Оглавление

1. Общее описание

Программа **Интегратор** является средством организации взаимодействия двух или более внешних систем (программных комплексов), работающих в сфере обеспечения безопасности (контроль доступа, видео наблюдение, охранная и пожарная система).

Внешней системой будем называть систему, функционирующую независимо от других систем и имеющую свой собственный интерфейс и настройки. С каждой внешней системой программа **Интегратор** связывается индивидуальным образом.

Связь программы **Интегратор** с внешними системами осуществляется посредством подключения соответствующих модулей (библиотек).

На данный момент к программе **Интегратор** могут быть подключены следующие модули, посредством которых она связывается с соответствующими внешними системами:

- TSS (модуль системы контроля доступа);
- email (модуль электронной почты);
- ISS (модуль системы видео наблюдения);
- ITV (модуль системы видео наблюдения);
- Sys (модуль системы ретрансляции);
- Shelni (модуль охранной системы);
- r08 (модуль охранной системы Рубеж 08 фирмы Сигма-ИС);
- SMS (модуль передачи SMS-сообщений).

В настоящем документе подробно описана теория конфигурирования системы и работы программы, а также разобраны практические приемы работы с ней, на основе прилагаемых к ПО демонстрационных модулей.

Во второй части документации описывается настройка работы с конкретными внешними системами.

2. Основные принципы работы программы

2.1. Введение

Суть функционирования данной программы состоит в преобразовании события, пришедшего от одной из внешних систем, в команду для другой.

Поэтому, на этапе настройки программного модуля **Интегратор**, в нем необходимо создать словарь перекодировки “событие-команда”.

Для каждого модуля, подключаемого к **Интегратору**, создается свой собственный словарь перекодировки.

На Рис. 1 представлена логическая схема, которая демонстрирует процесс взаимодействия двух внешних систем:

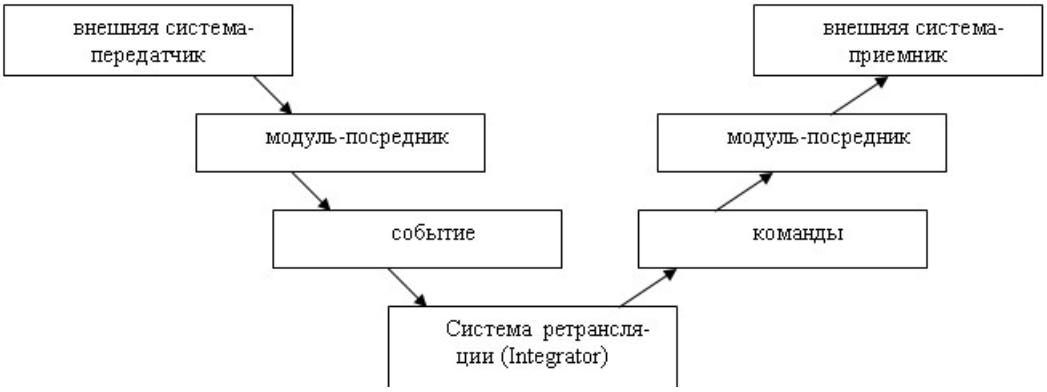


Рис. 1

Когда из внешней системы в **Интегратор** приходит поток данных, то он (**Интегратор)** сравнивает входной поток со всеми масками событий[1], которые в нем определены для модуля-посредника (через который происходит взаимодействие между транслятором и системой-передатчиком). При обнаружении подходящей маски генерируется событие, по которому формируются команды, образующие очередь, согласно которой они должны выполниться на системе-приемнике.

И, наконец, происходит последовательная (с учетом указанных временных задержек) пересылка команд внешней системе-приемнику через модуль, являющийся посредником между системой **Интегратор** и системой-приемником.

Следует учесть, что любая внешняя система может являться:

– Передатчиком;
– Приемником;
– И передатчиком и приемником.

2.2. Терминология

Составим словарь терминов, которые использовались при написании данного руководства:

- **Модуль (Модуль-посредник)** - это модуль, посредством которого осуществляется взаимодействие между системой ретрансляции (**Интегратор**) и одной из внешних систем (приемником или передатчиком).
- Это программное обеспечение, которое создается независимо от всех внешних систем (для каждого программного комплекса в отдельности) и подключается к программе **Интегратор** в виде библиотеки. Модуль обеспечивает связь **Интегратор** с какой-либо внешней системой.
- **Система – передатчик** – это внешняя система, от которой приходит поток данных на модуль- посредник (соответствующий данной системе).
- **Система-приемник** – это внешняя система, которой посылаются посредством соответствующего модуля-посредника команды на выполнение.
- **Маска события** – это строка, с которой сравнивается поток данных, приходящий от внешней системы.

---

[1] См. **п. 2.2**

- **Маска команды** – это выходная строка, которая отправляется для выполнения на систему-приемник.
- **Регулярные выражения** – это широко используемый способ описания шаблонов для поиска текста и проверки соответствия текста шаблону.

2.3. Механизм преобразования строк

Обратите внимание, что данный раздел является ключевым для понимания и последующей настройки всего механизма работы ПО **Интегратор**.

Еще раз остановимся на общих принципах работы программы. Итак, извне приходит поток данных в виде символьной строки. На основе анализа этих данных мы должны решить, во-первых, подлежит ли пришедшая строка преобразованию (т.е. нужно ли по ее приходу формировать команду для выходной системы) и, если подлежит, то, как сформировать необходимую нам команду.

Ниже приведены основные положения, которые касаются процессов формирования маски, констант и переменных. При формировании масок используется язык регулярных выражений (см. документ “**Синтаксис регулярных выражений**”).

2.3.1. Формирование маски

Для разбора входной строки используются маска подстроки любых символов («.*») и набор литералов (заданных символов), которые должны присутствовать в строке. Если пришедшая строка удовлетворяет установленной маске, то **Интегратор** начинает действия по формированию выходной команды, если нет, игнорирует ее.

Например, нам необходимо “отловить” событие от внешней системы и отправить команды, определенные для данного события, другой внешней системе на выполнение.

Пусть строка события должна содержать в себе символьную подстроку SELFCONTROL.

Чтобы на систему-приемник были отправлены на выполнения нужные команды, во-первых, должно быть отловлено соответствующее событие.

Зададим для этого маску события вида . *, SELFCONTROL, .*. После чего любая входная строка будет сравниваться с указанной маской. Если она удовлетворяет маске (т.е. содержит в себе символьную подстроку SELFCONTROL), то Интегратор начинает отправлять команды на внешнюю систему-приемник посредством модуля-посредника, который был задан оператором при задании маски команды.

Для данного события определим следующие маски команды (подробнее см. п. 3.3.2.1):

- PlayWavFile (“c:\ WINDOWS\HELP\Tours\ WindowsMediaPlayer\Audio\Wav\wmpaud5.wav")”)- команда на проигрывание звукового файла;
- Snd("slava@office.sevenseals.ru", "%Time("Windows")%", "SelfControl") – отправка письма на указанный электронный адрес получателя.

Вышеуказанные команды будут отправляться в порядке очереди (в зависимости от приоритета) на систему-приемник, как только с системы-передатчика будет приходить поток данных, удовлетворяющих маске события *, SELFCONTROL, .* .

2.3.2. Переменные и их использование

Для ряда выходных команд необходимо задавать параметры, присутствующие во входном событии. Эти параметры задаются в виде переменных.

Первоначально переменная формируется в маске события. Для примера, зададим маску события в виде выражения “a % x1%b”. В данном выражении переменной будет являться x1. Переменные всегда заключаются между знаками %.

Далее, мы можем использовать заданную переменную при создании выходной команды.

Например:

- Зададим переменную **x1** в маске события: “a % **x1** %b”;
- Зададим на выход маску команды в виде %**x1**%.
- Зададим сроку в виде: “a **Тревога** b”.
- Получим: **Тревога**.

2.3.3. Формирование констант

Программа позволяет создать список констант. Константы могут использоваться для подстановки в анализируемые строки, часто встречающиеся символьные и шестнадцатеричные значения, а также позволяют оперировать символами, которые нельзя ввести с клавиатуры. Константы создаются в окне **Константы и функции (**вкладка **Константы)** (см. п. 3.2.1).

Например, зададим константу разделителя. Для этого:

1. Создадим константу EOL=ODOA;
2. Подставим в выражение: abc%EOL%def;
3. Получим: abc ODOA def.

2.3.4. Использование функций

Помимо создания списка констант, в программе (в окне **Константы и функции (**закладка **Функции)**) можно просмотреть перечень функций, их описание (см.Рис. 5). Функции[2] также могут быть использованы для подстановки в анализируемые строки. Они отображаются в окне **Константы и функции (**закладка **Функции).**

Например, нам необходимо отправить по электронной почте письмо и в качестве темы передать локальное время в формате Windows. Для этого мы вставим в команду Snd("slava@office.sevenseals.ru", "%**Time("Windows")**%", "SelfControl") имя функции: %Time("Format{hhnnsszzz;Windows}")%.

Перед тем, как использовать ее в данной команде, нам нужно было скопировать ее имя из таблицы (вкладка **Функции)** в буфер обмена, а потом уже вставить его в маску команды.

При получении письма по электронной почте, на место темы (Subject) письма будет подставляться значение времени.

---

[2] Список функций отображается в окне вкладки **Функции,** загружаясь из библиотеки **ifuncs.dll**, файл которой находится в одном каталоге с программой **Integrator**.

3. Описание интерфейса программы Интегратор

Главное окно программы состоит из следующих элементов (см. Рис. 2):

– (**1**)- окно, в котором представлен перечень всех подключенных в данный момент модулей.

– (**2**)- окно, содержание которого зависит от того, какая вкладка[3] на панели (**4**) выбрана пользователем (**Модуль, Трансляция** или **Протокол**).

– (**3**)- управляющее меню, представленное двумя пунктами (**Файл** и **Средства**).

– (**4**)- основные программные вкладки (**Модуль, Трансляция** или **Протокол**), которые пользователь может переключать в ходе работы с программой.

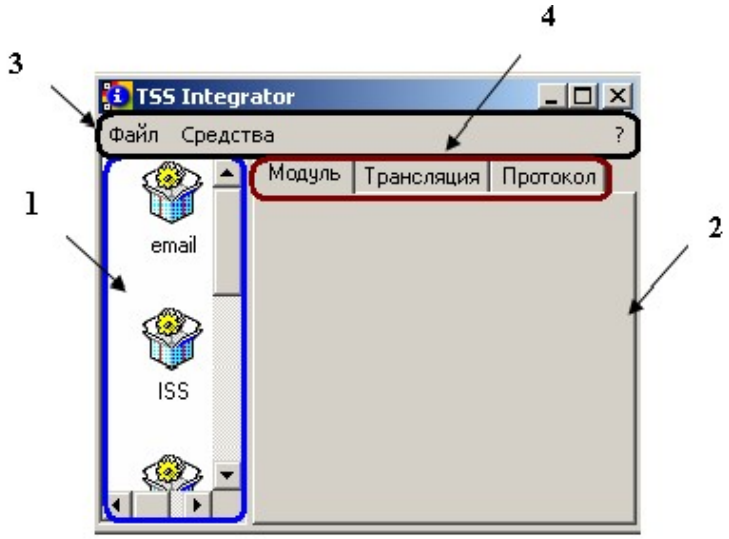


Рис. 2

3.1. Главное меню программы

Главное меню, расположенное на панели (**3**), содержит три пункта:

– Файл

– Средства

– ?

3.1.1. Файл

Пункт меню **Файл** включает в себя подпункты:

- ***Модули*** (включение/отключение модулей)
- ***Выход*** (завершение работы с программой)

[3] Назначение каждой вкладки будет рассмотрено позже.

При выборе подпункта ***Модули*** на экране появляется окно (см.Рис. 3), в котором представлена таблица, состоящая из четырех полей:

4. *Файл* (название библиотеки, соответствующей определенному модулю);
5. *Имя* (имя модуля);
6. *Описание* (описание модуля);
7. *Версия* (версия, присвоенная модулю).

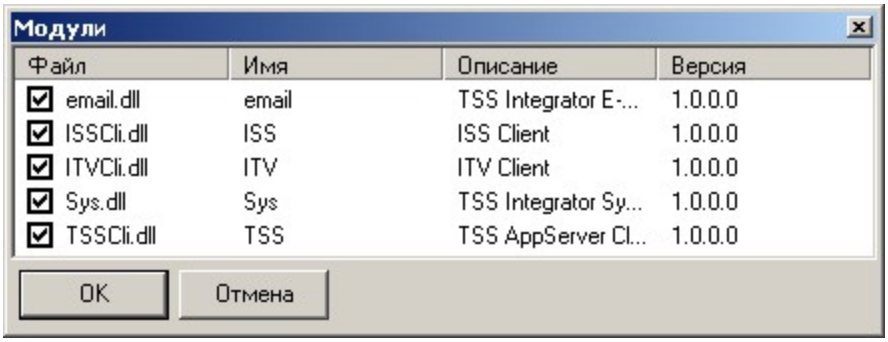


Рис. 3

Щелчком левой кнопкой мыши в опционном окошке, принадлежащем определенному модулю, можно включать (☑) или выключать (☐) данный модуль.

Если модуль отключен, то он не отображается в общем списке модулей в главном окне программы (**1**) (см. Рис. 2).

Кнопка **OK** служит для сохранения внесенных изменений и закрытия окна ***Модули***.

Кнопка **Отмена** – закрывает окно ***Модули***, не сохраняя при этом изменения, сделанные в нем пользователем.

При выборе последнего подпункта ***Выход*** (в меню **Файл**) осуществляется завершение работы с программой (подробнее см. п. 5.2).

3.1.2. Средства

- Пункт меню **Средства** включает в себя следующие подпункты:
- ***Константы и функции*** (задание констант и загрузка функций для дальнейшего их использования в создании масок событий и масок команд);
- ***Опции*** (настройка дополнительных возможностей программы).

Рассмотрим их подробнее.

- При выборе подпункта ***Константы и функции*** на экране появляется окно:

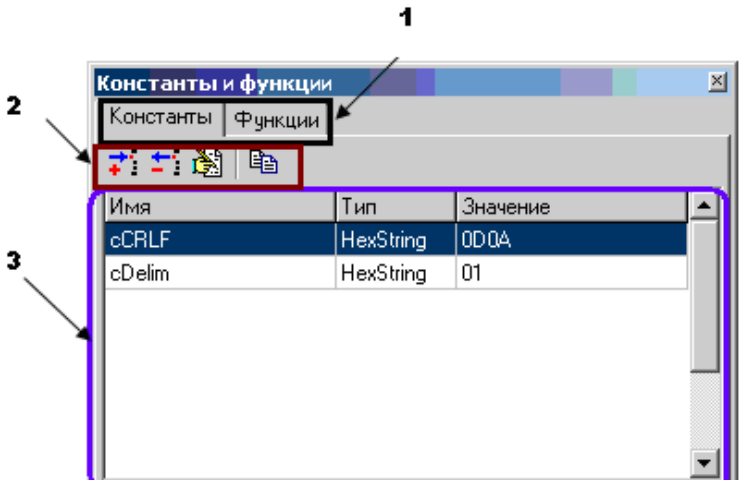


Рис. 4

Панель (**1**) содержит две вкладки:

1. **Константы** (используются в регулярных выражениях при создании масок событий и масок команд);
2. **Функции** (для просмотра системных функций программы и использования их при создании команд).

Когда необходимо задать определенную константу для дальнейшего ее использования в регулярном выражении (при создании масок команд и событий), нужно выбрать вкладку **Константы (**см. Рис. 4**)**.

В окне (**3**) отображена таблица (содержащая все ранее определенные константы) со следующими полями:

– *Имя* (имя константы);

– *Тип* (тип константы)*;*

– Для констант определены два типа:

- String (символьная строка);
- HexString (строка шестнадцатеричных символов).

– *Значение* (значение константы)*;*

Панель (**2**) содержит следующие клавиши управления:

– **Добавить** (добавление новой записи в таблицу);

– **Удалить** (удаление из таблицы выбранной записи);

– **Изменить** (редактирование выбранной записи);

– **Копировать имя константы в буфер обмена** (данную клавишу удобно использовать при создании словаря масок событий, когда необходимо скопировать имя константы и вставить его в регулярное выражение).

Процедуры добавления, удаления и изменения записи будут рассмотрены подробно в п. 3.2.

Теперь перейдем к рассмотрению вкладки **Функции.** Для этого нужно выбрать данную вкладку на панели (**1**). В ней можно просмотреть системные внутренние функции программы **Интегратор** (см. Рис. 5) и их описание, чтобы использовать их для подстановки в анализируемые строки.

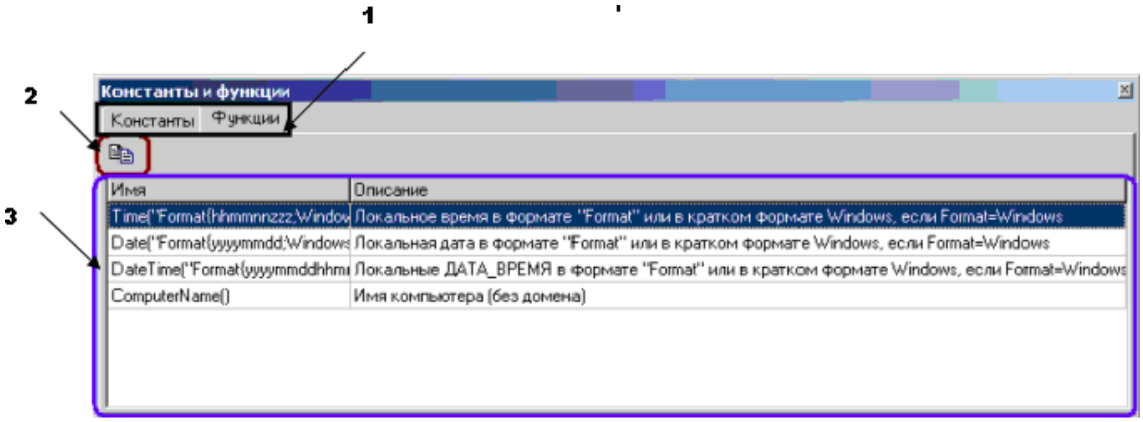


Рис. 5

На управляющей панели (**2**) расположена кнопка:

– **Копировать имя функции в буфер обмена** (данная кнопка упрощает процедуру переноса выбранной функции из таблицы в словарь масок, где она будет использована при формировании соответствующей команды).

Окно (**3**) содержит таблицу с полями (см. Рис. 5):

– *Имя* (имя функции, представленное в соответствующем формате);

Функция **Time** (время) имеет параметр: ("Format{hhnnsszzz;Windows}"). Время может передаваться:

– - в формате Format: час_мин_сек_милисек;

– - или в кратком формате Windows (если Format= Windows).

Функция **Date** (дата) имеет параметр ("Format{yyyymmdd;Windows}"). Дата может передаваться:

– - в формате Format: г_м_д (год) (месяц) (число);

– - или в кратком формате Windows (если Format= Windows).

–

Функция **DateTime** (Дата_Время) имеет параметр ("Format{yyyymmddhhnnsszzz;Windows}"), который может передаваться:

– - в формате Format: год_месяц_день_час_мин_сек_милмсек_;

– - или в кратком формате Windows (если Format= Windows).

– *Описание* (описание функции).

Использование функций подробно рассмотрено в п. 2.3.4.

- Подпункт ***Опции*** будет подробно описан в п. 5.1.

3.1.3. Пункт “?”

Данный пункт содержит справочную информацию **О программе**.

3.2. Работа с константами

3.2.1. Добавление новой константы

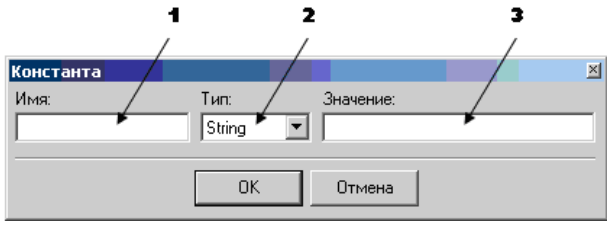


Рис. 6

При нажатии на кнопку **Добавить,** расположенную на панели управления (**2**) (см. Рис. 4), на экране появляется окно (см. Рис. 6):

В поле (**1**) вводится произвольное имя константы. В поле (**2**) из выпадающего списка выбирается тип константы (String или Hex String). В поле (**3**) указывается ее значение.

Кнопка **OK** служит для сохранения внесенных данных и закрытия окна **Константа**.

Кнопка **Отмена** – закрывает окно **Константа**, не сохраняя при этом внесенные данные.

3.2.2. Удаление константы из списка

При нажатии на кнопку **Удалить** (на панели (**2**)) на экране появляется окно для подтверждения удаления выбранной записи.

3.2.3. Редактирование константы

Функция **Изменить** позволяет редактировать выбранную в таблице запись. При этом на экране загружается окно, представленное на Рис. 7.

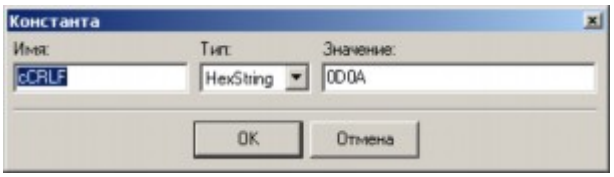


Рис. 7

3.3. Назначение основных вкладок

Программное окно (см. Рис. 2, панель (**4**)) состоит из трех вкладок:

- Модуль
- Трансляция
- Протокол

3.3.1. Вкладка Модуль

Данная вкладка служит для:

- Настройки соединения с внешней системой (путем задания соответствующих параметров).
- Ручного соединения и отсоединения.

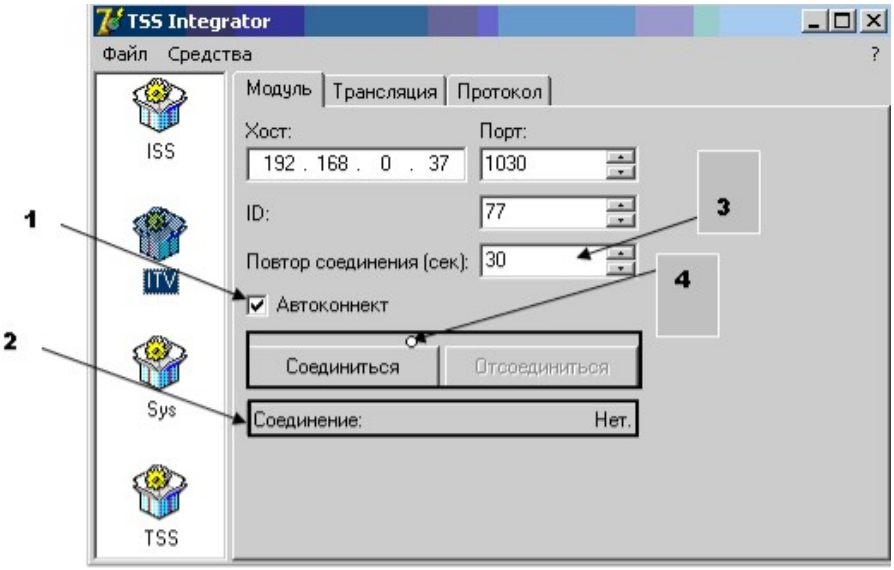


Рис. 8

Параметры соединения для каждого модуля индивидуально будут описаны далее в соответствующих документах.

3.3.2. Вкладка Трансляция

В данной вкладке описывается система трансляции «событие – команда», что является основной работы программы **Интегратор**.

Окно разделено на две панели (**1** и **2**), которые содержат таблицы алгоритмов форматирования и преобразования строк входных (панель **1**) и выходных данных (панель **2**) (см. Рис. 9).

Таблица (**1**) состоит из двух колонок:

– *Активная* (включение/отключение данного события из списка обрабатываемых);

– *Маска события* (смысловой разбор приходящей строки).

Таблица (**2**) состоит из пяти колонок:

– *Активная* (включение/отключение данной команды из списка отправляемых);

– *Важная* (данное свойство актуально для команд при превышении максимального размера их очереди, см.п. 5.1);

– *Модуль* (имя модуля, соответствующего внешней системе, которой будет послана команда на выполнение);

– *Задержка* (для задания задержки исполнения выходной команды);

– *Маска команды* (указывается способ формирования исходящей строки).

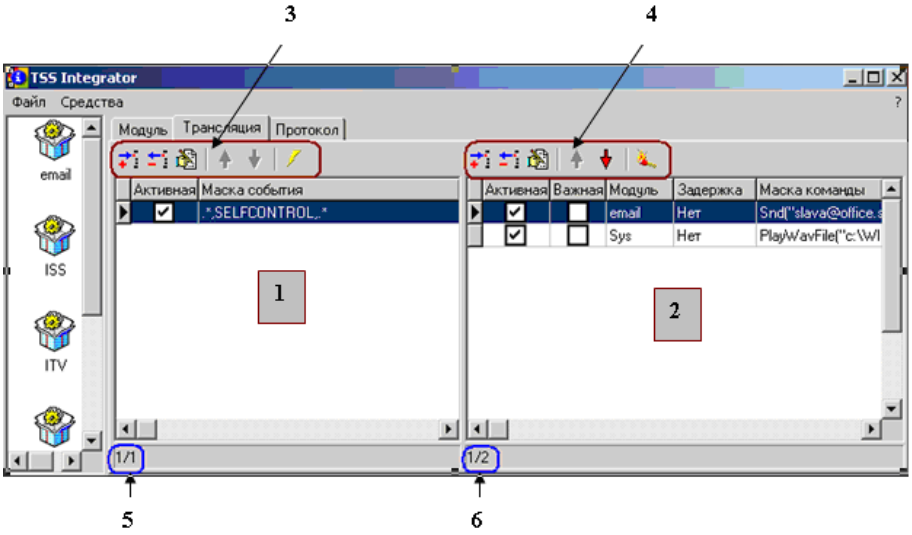


Рис. 9

На панели управления (**3**) располагаются следующие функциональные кнопки (см.Рис. 9):

- **Добавить** (добавление записи в таблицу);
- **Удалить** (удаление выбранной записи из таблицы);
- **Изменить** (редактирование выбранной записи таблицы);
- **Переместить вверх** (перейти на одну запись вверх по таблице);
- **Переместить вниз** (перейти на одну запись вниз по таблице);
- **Эмулировать событие** (эмулировать генерацию события модулем, относящимся к системе-передатчику).

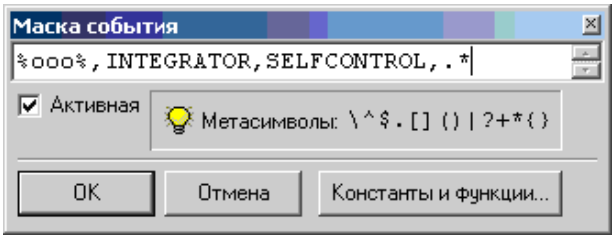
На панели управления (**4**) расположены кнопки, аналогичные кнопкам панели (**3**). Только вместо последней кнопки **Эмулировать событие** - кнопка **Послать команду,** которая используется для отладки.

Панели (**5**) и (**6**) отображают: **(порядковый номер выбранной строки в таблице / общее количество строк в таблице)** (см. Рис. 9).

3.3.2.1. **Добавление записи**

1. **Для маски события**

При нажатии на кнопку **Добавить** на экране появляется окно (см. Рис. 10):



В верхней части данного окна расположено поле, которое предназначено для ввода маски события.

При определении новой маски события, можно включить его в список обрабатываемых. Для этого нужно включить опцию **Активная.**

Рис. 10

Кнопка **OK** служит для сохранения новой записи, внесения ее в таблицу и закрытия окна **Маска события**.

Кнопка **Отмена** – закрывает окно **Маска события**, не занося новую запись в таблицу.

Кнопка **Константы и функции** вызывает окно, изображенное на Рис. 4 (п. 3.1.2).

2. **Для маски команды**

При нажатии на кнопку **Добавить** на экране появляется окно:

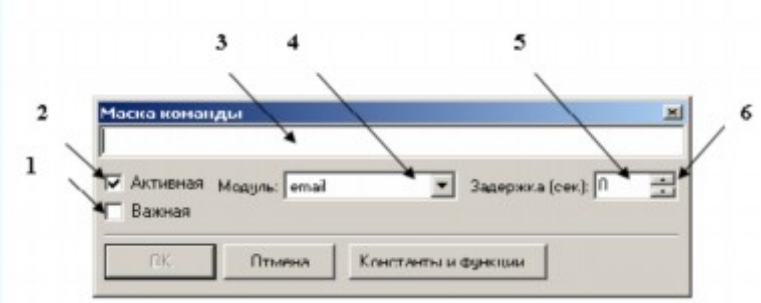


Рис. 11

Поле (**3**) предназначено для ввода маски команды (см.Рис. 11).

Для каждой маски события существует своя очередь команд (т.е. команды выполняются в порядке их следования в очереди). При этом очередь является цик-

личной[4]. Каждая команда в очереди может быть объявлена как **Важная** (для этого включается опция **(1)**). Если количество команд превысило допустимый размер очереди (см.Рис. 16), то команда, которая уже не попадает в этот диапазон, становится на место команды с наименьшим порядковым номером в очереди, если последняя не отмечена как **Важная**.

В поле (**2**) устанавливается признак активности команды (включение/отключение ее из списка обрабатываемых).

В поле (**4**) указывается модуль, внешней системе которого посылается команда для выполнения.

В поле (**5**) с помощью стрелок (**6**) или путем непосредственного ввода с клавиатуры устанавливается временная задержка (в секундах) между генерацией события и выполнением назначенной команды.

Кнопки **OK, Отмена** и **Константы и функции** имеют то же назначение, что и при добавлении в таблицу новой маски события (см., п. **1**).

3.3.2.2. Удаление записи

Процедура удаления записи из таблицы одинакова как для маски события, так и для маски команды.

При нажатии на кнопку **Удалить** (на панелях (**3**) и (**4**), Рис. 9) на экране появляется окно для подтверждения удаления выбранной записи.

3.3.2.3. Редактирование записи

Для маски события

Функция **Изменить** позволяет редактировать выбранную в таблице запись. При этом на экране загружается окно, аналогичное тому, что изображено на Рис. 10.

2. Для маски команды

Процедура редактирования записи для маски команды та же, что и для маски события. Только при этом на экране отображается окно, аналогичное тому, что изображено на Рис. 11.

3.3.2.4. Эмуляция[5] события

Данная функция предназначается для отладки. Пользователь может искусственно сгенерировать событие, удовлетворяющее одной из масок и посмотреть результат своих действий в журнале (вкладка **Протокол**).

---

[4] Т.е. при превышении размера очереди программа возвращается в ее начало.

[5] Эмуляция события - это возможность для пользователя искусственно задать какое-либо событие в целях проверки работы системы.

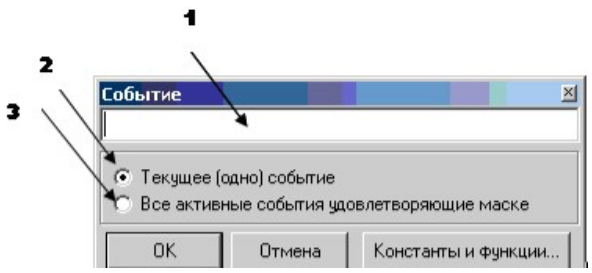


Рис. 12

Поле (**1**) предназначено для отображения события, которое пользователь будет эмулировать.

- Пользователь может эмулировать только текущее (одно) событие, которое им выбрано (выделено) в списке масок событий (см. Рис. 9, таблица (**1**)). Для этого включается опция (**2**).
- Либо все события, удовлетворяющие маске (введенной в поле (**1**)) и отмеченные в списке как **Активные**. Для этого включается опция (**3**).
- Второй вариант наиболее точно отражает реальную работу программы.

Кнопки **OK, Отмена** и **Константы и функции** имеют то же назначение, что и при добавлении в таблицу новой маски события (см.п. 3.3.2.1).

3.3.2.5. **Отправка команды**

При нажатии на кнопку **Послать команду** (см. Рис. 9, управляющая панель (**4**)), никаких изменений на экране не происходит.

Если же при настройке дополнительных параметров, в окне под названием ***Опции*** (см. Рис. 16), была включена опция *Переключаться на “Протокол” при эмуляции*, то при отправке команды произойдет автоматический переход во вкладку **Протокол**, в которой можно просмотреть все текущие события.

Когда при формировании выходной команды в ее маске используется переменная (например, %x1%) и ее значение неизвестно, т.е. мы не эмулируем искусственно событие, которое бы совпало с той маской, в которой была сформирована данная переменная (подробнее см. п.2.3.2).

Тогда при нажатии на кнопку **Послать команду** на экране появится окно, в котором будет предложено задать значение переменной, которая использовалась в маске команды (см. Рис. 13):

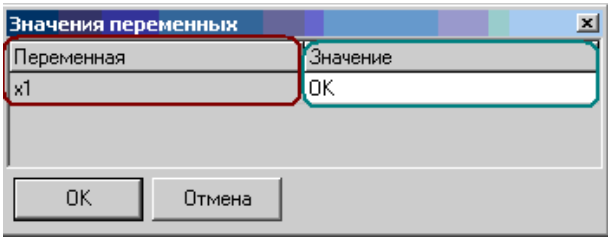


Рис. 13

Кнопка **OK** служит для сохранения введенного значения, закрытия данного окна и отправки команды (маска которой была задана в виде %x1%) на систему-приемник. После этого во вкладке **Протокол** отобразятся строки:

10.11.2004 13:03:52 ITV T: OK

- передача строки OK на систему-приемник посредством модуля ITV.

10.11.2004 13:03:52 ITV O: OK

- строка OK получена системой-приемником посредством модуля ITV.

Кнопка **Отмена** – закрывает окно **Значения переменных**, не сохраняя введенные значения.

3.3.3. Вкладка Протокол

Вкладка **Протокол** выполняет функции журнала, в котором отображаются текущие события.

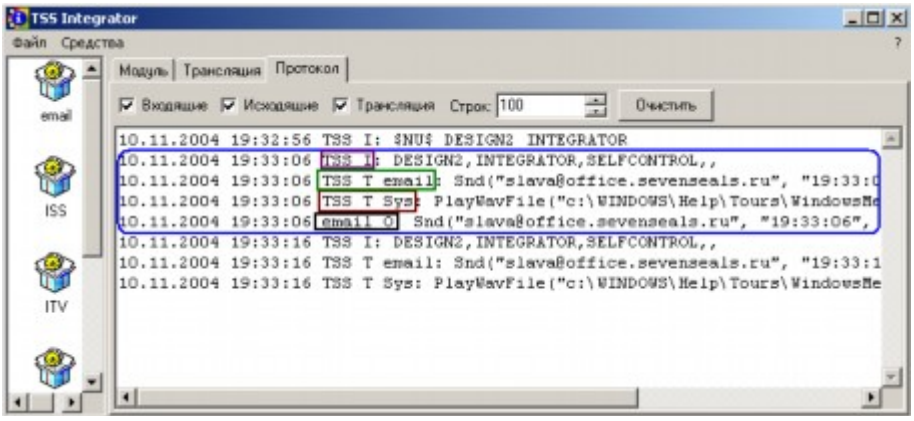


Рис. 14

На панели, расположенной в данной вкладке, можно задать следующие параметры:

– Группы событий, которые заносятся в протокол (входящие, исходящие и трансляция).

В протоколе отображаются события, которые относятся к тем группам, которые отмечены на панели меткой. Помимо отмеченных типов событий, в протокол всегда заносятся события об ошибках выполнения команд и о факте удаления команд, следовавших в очереди за той командой, при выполнении которой произошла ошибка.

– Допустимое количество строк, которое занесется в протокол (устанавливается пользователем путем ввода с клавиатуры или с помощью стрелок).

С помощью кнопки **Очистить (**или при достижении максимального количества строк в протоколе) происходит удаление всех записей из окна протокола.

Чтобы Вы могли прочитать события, которые отображаются во вкладке **Протокол**, в качестве примера расшифруем некоторые из них, выделенные в окне на Рис. 14:

- 10.11.2004 19:33:06 TSS I: DESIGN2,ИНТЕГРАТОР,SELFCONTROL,,
  - 10 ноября 2004 года в 19:33:06 с внешней системы (на которой установлено ПО TSS) на одноименный модуль-посредник поступил (I-In) поток данных.
- 10.11.2004 19:33:06 TSS T email: Snd("slava@office.sevenseals.ru", "19:33:06 ", "SelfControl")
  - 10 ноября 2004 года в 19:33:06 произошла передача (T- translation) команды Snd.
- 10.11.2004 19:33:06 TSS T Sys: PlayWavFile("c:\WINDOWS\Help\Tours\WindowsMediaPlayer\Audio\Wav\wmpaud5.wav")
  - 10 ноября 2004 года в 19:33:06 произошла передача (T- translation) команды для выполнения (PlayWavFile) на систему-приемник посредством модуля Sys.
- 10.11.2004 19:33:06 email O: Snd("slava@office.sevenseals.ru", "19:33:06 ", "SelfControl")
  - 10 ноября 2004 года в 19:33:06 переданная команда Snd была выполнена (т.е. было отправлено письмо по указанному электронному адресу).

Дадим расшифровку кратких обозначений, которые используются во вкладке **Протокол** при отображении записей:

- I (Inn) – входящие;
- T (Translation) – трансляция, передача;
- O (Out) – исходящие;
- E (Error) – ошибка;
- D (Delete) – удаление.

4. Процедура установки ПО

4.1. Требования к ПК, ОС, сети

Требования к операционной системе (ОС):

- ОС на платформе NT;
- Windows 2000;
- Windows XP;
- Windows 2003.

Требования к ПК определяются требованиями, предъявленными к ОС, т.е. характеристики ПК должны удовлетворять таким требованиям, при которых на данный ПК может быть установлена любая из вышеперечисленных ОС.

Требования к сети индивидуальны для каждой внешней системы, с которой связывается **Интегратор** в ходе работы.

4.2. Инсталляция с CD

Для того чтобы установить программу на жесткий диск компьютера, необходимо запустить с диска (который поставляется компанией) программу инсталляции **tss_Интегратор_setup.exe**.

В инсталляционный пакет будут в обязательном порядке входить модули, для иллюстрации работы программы:

- Demo1
- Demo2
- Sys

Данные модули будут автоматически устанавливаться вместе с ПО **Интегратор.**

Что же касается рабочих модулей, они будут предоставляться клиентам за отдельную плату.

4.3. Файловая система

По окончанию инсталляции на жестком диске компьютера будет создан каталог, содержащий следующие файлы:

- Db.udl (в данном файле прописывается путь к базам данных (БД));
- Интегратор.exe (исполняемый файл программы);
- Интегратор.log (системный журнал событий);
- ifuncs.dll (библиотека функций);
- vcl70.bpl (системный модуль (.Borland Package Library));
- rtl70.bpl. (системный модуль (.Borland Package Library));

В зависимости от того, какие модули будут приобретены вместе с ПО **Интегратор,** в программном каталоге будут размещаться файлы соответствующих средств, посредством которых осуществляется связь с внешними системами (например, для системы ITV таким средством будет являться IIDK-интерфейс).

Помимо вышеперечисленных файлов, в каталог должна быть помещена папка Plugins, в которую заносятся файлы библиотек, соответствующие подключенным к **Интегратору** модулям:

- email.dll (модуль электронной почты);
- ISSCli.dll (модуль программного комплекса ISS);
- Sys.dll (модуль программного комплекса Интегратор);
- TSSCli.dll (модуль программного комплекса TSS);
- ITVCli.dll (модуль программного комплекса ITV);
- Shelni.dll (модуль программного комплекса Shelni).

А также, папка Db, в которой находятся файлы баз данных.

На диске, предоставленном фирмой-поставщиком, содержится дополнительный справочный материал под названием *Синтаксис регулярных выражений.doc*. При необходимости, пользователь может подробно изучить данный материал для дальнейшего использования его в работе с ПО **Интегратор,** а именно при формировании масок события и масок команд.

Загрузка программы осуществляется путем запуска исполняемого файла **Интегратор.exe.**

4.4. Особенности установки

Программа защищена от несанкционированного доступа электронным ключом (т.н. HASP).

Программа **Интегратор** может быть установлена:

- На отдельном компьютере, связанном локальной сетью с компьютерами, на которых установлено ПО взаимодействующих между собой систем[6];
- На том же компьютере, на котором установлено программное обеспечение одной из внешних систем, участвующей во взаимодействии.

Если **Интегратор** установлен на отдельном компьютере, то для его связи с внешней системой нужно указать (при настройке параметров соединения) имя или IP-адрес того компьютера, на котором установлено ПО данной системы.

С некоторыми внешними системами нет возможности установить связь по сети. В данном случае ПО **Интегратор** и ПО внешней системы устанавливаются на одном компьютере.

Если к программе подключен модуль **e-mail** (т.е. в ней предусмотрена функция отправки сообщений на электронную почту получателя), то должен быть предусмотрен выход в Интернет для компьютера, с которого будет осуществляться отправление письма.

5. Настройка ПО

Рядовой пользователь имеет возможность лишь просмотреть настройки системы, но редактировать он их не сможет.

Настройки могут быть изменены пользователем только в том случае, если у него имеются права администратора.

5.1. Задание дополнительных параметров

Для задания дополнительных параметров при настройке работы ПО **Интегратор** нужно выбрать на панели (**3**) главного меню программы пункт **Средства** (см. Рис. 2)**,** а затем подпункт ***Опции.***

---

[6] ПО для каждой из взаимодействующих систем должно устанавливаться на отдельном компьютере.

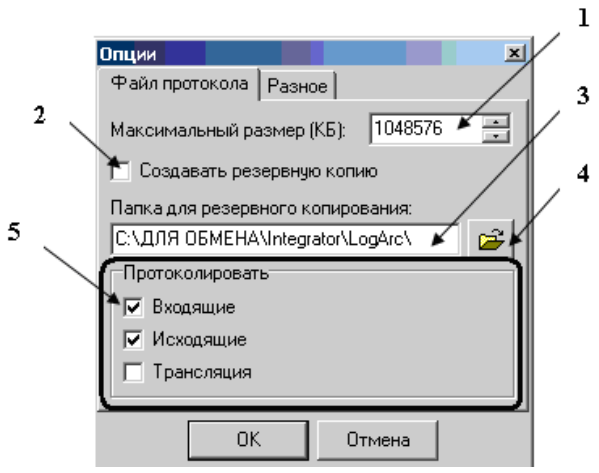


Рис. 15

На Рис. 15 представлено окно для случая, когда на верхней панели выбрана вкладка **Файл протокола**.

В данном окне задаются параметры и настройки для файла протокола[7].

В поле (**1**) вводится с клавиатуры (либо корректируется с помощью стрелок) значение максимального размера файла протокола[8].

При включении опции (**2**) на диске будет создаваться резервная копия, куда будут автоматически сбрасываться все события из файла протокола, когда размер последнего превысит предел, заданный в поле (**1**) (КБ, в килобайтах).

Если же опция (**2**) будет выключена, то файл протокола будет очищаться от записей (без перенесения их в резервный файл) после того, как будет превышен его максимальный размер, заданный в поле (**1**).

В поле (**3**) отображается путь к папке, в которой содержится файл резервной копии. Его можно изменить, нажав кнопку (**4**) и выбрав в окне **Обзор папок** нужную папку.

На панели (**5**) отмечаются (включением соответствующих опций) те типы событий (входящие, исходящие, трансляция), которые будут заноситься в файл протокола (см. Рис. 15). В файл протокола всегда заносятся:

– события об ошибках (**E-Error**), возникающих при выполнении команд;
– события о фактах удаления команд (**D-Delete**), которые следуют в очереди за командами, вызвавшими ошибку при выполнении.

Кнопка **OK** служит для сохранения внесенных изменений и закрытия окна **Опции**.

Кнопка **Отмена** – закрывает окно **Опции**, не сохраняя при этом изменения, сделанные в нем пользователем.

---

[7] Файл протокола (**Integrator.log**) образуется в ходе работы программы и находится в программном каталоге.

[8] Размер файла протокола задается в килобайтах (КБ).

Перейдем к следующей вкладке, под названием **Разное.** После выбора данной вкладки на экране появляется окно (см. Рис. 16):

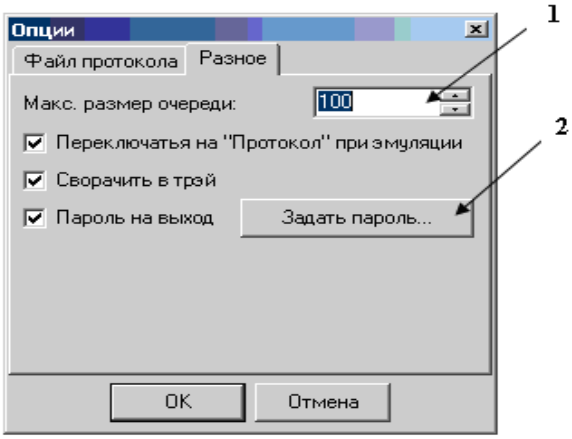


Рис. 16

В данном окне можно задать следующие параметры:

– *Максимальный размер очереди* [9](устанавливается в поле (**1**) путем ввода с клавиатуры либо с помощью стрелок);

– *Переключаться на "Протокол" при эмуляции* (для этого нужно включить данную опцию);

После отправления команды в систему-приемник происходит автоматический переход во вкладку **Протокол** для просмотра событий, которые были занесены в журнал.

– *Сворачивать в трей*[10] (для этого нужно включить данную опцию);

При включении данной опции, после запуска программы ее иконка будет отображаться в трее. И запускаться она будет из трея.

– Пароль на выход.

При включении данной опции следует перейти к заданию пароля нажатием на одноименную кнопку (**2**).

Откроется окно, представленное на Рис. 17 , в котором нужно заполнить поля:

- **Текущий пароль** (ввести уже существующий пароль (на выход) если он был определен ранее в программе, иначе - оставить поле пустым);
- **Новый пароль** (ввести в поле новый пароль);
- **Проверка нового пароля** (подтвердить новый пароль в данном поле).

---

[9] Количество команд в очереди (для каждого события своя очередь команд).

[10] Трей – это системная панель, расположенная в правом нижнем углу экрана компьютера.

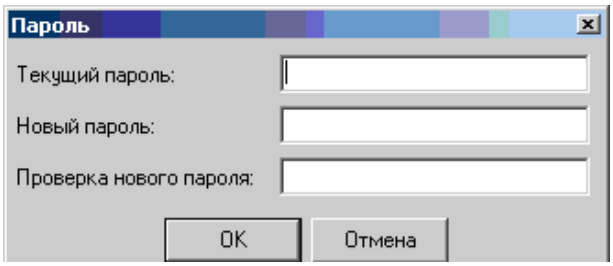


Рис. 17

После заполнения вышеописанных полей, для подтверждения заданных опций и их сохранения следует нажать кнопку **ОК**, для закрытия текущего окна без сохранения введенных параметров - кнопку **Отмена**. В обоих случаях окно **Пароль** закрывается, и происходит возврат к окну **Опции**.

5.2. Выход из программы “TSS Интегратор”

Выход из программы **TSS Интегратор** можно осуществить двумя способами:

1. Без запроса пароля на выходе;
2. С запросом пароля на выходе.

Для того чтобы завершить работу с программой первым способом, Вам не следует устанавливать метку рядом с опцией *Пароль на выход* (**Главное меню программы** -> п. **Средства**-> ***Опции*** -> **Разное**). При этом Вам достаточно будет:

- Выбрать пункт **Файл** главного меню программы;
- В выпадающем меню выбрать пункт **Выход**.

Если же Вы установили вышеупомянутую метку рядом с опцией *Пароль на выход* (см. Рис. 16), то в этом случае Вы уже имеете дело со вторым способом завершения работы с программой.

В данном случае, при выборе пункта **Выход** на экране будет появляться окно (см. Рис. 18), в котором Вам будет предложено ввести пароль (который был ранее задан Вами при установке настроек (см. Рис. 17)).

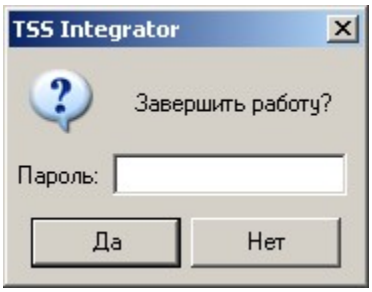


Рис. 18

После того как Вы введете с клавиатуры правильный пароль, для подтверждения выхода из программы, нажмите кнопку **ДА**. Окно программы закроется.

В случае если Вы не захотите завершать работу с программой, нажмите – **НЕТ**. Окно программы останется открытым для дальнейшей работы.

6. Описание работы ПО Интегратор на примере Demo-версии

Рассмотрим на наглядном примере механизм работы ПО **Интегратор**.

1) Опишем процесс взаимодействия двух систем посредством программы **Интегратор.**

Во-первых, к программе **Интегратор** нужно подключить два модуля (**Demo1** и **Demo2**), посредством которых система ретрансляции будет связываться с одноименными внешними системами.

Пусть в данном примере системой-передатчиком будет выступать **Demo1**, а системой-приемником – **Demo2**:

1. В окне (**1**) выберите модуль **Demo1**;
2. На панели (**2**)активизируйте вкладку **Модуль** (см. Рис. 19).

В качестве примера рассмотрим назначение полей, опции и кнопки для *Строки 1* (1-го потока данных).

В поле (**4**) введен искусственно созданный поток данных, который поступит с внешней системы **Demo1** на систему ретрансляции (**Интегратор**) при нажатии на кнопку (**3**) .

Опция **Авто** включается для того, чтобы поток данных, заданный в поле (**4**) поступал на **Интегратор** через интервал времени, который задается в поле (**5**)[11]. Если данная опция выключена, следует пользоваться кнопкой (**3**).

Аналогичные настройки задаются для *Строки 2* (2-го потока данных) и *Строки 3* (3-го потока данных).

Окно (**6**) служит для отображения команд, которые посылаются с **Интегратора** в систему **Demo1** на выполнение (в этом случае, система **Demo1** является приемником).

[11] Временной интервал задается в секундах.

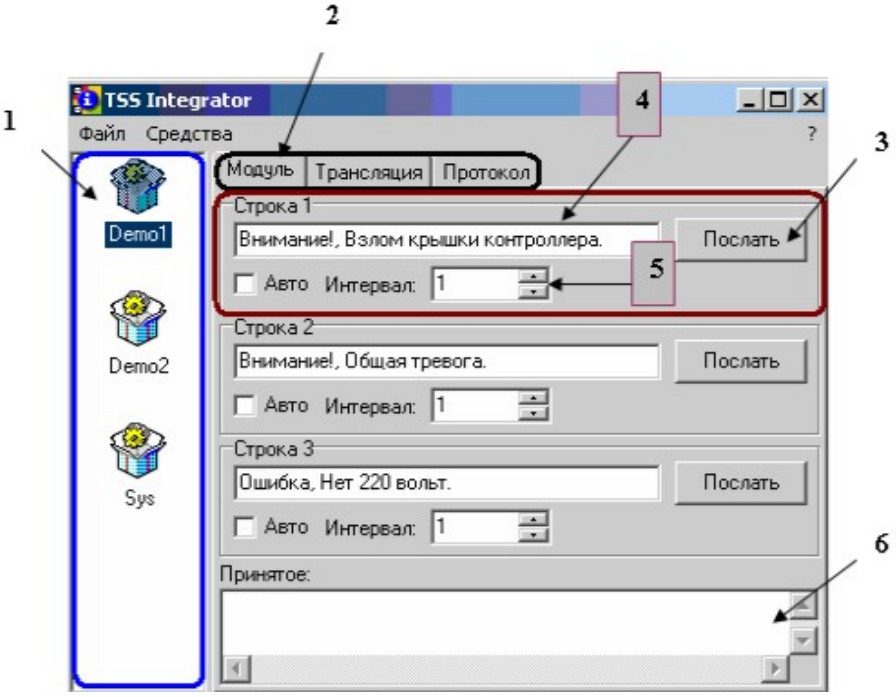


Рис. 19

Вкладка **Модуль** для модуля **Demo2** аналогична вышеописанной (т.е. для **Demo1**).

3. Перейдите во вкладку **Трансляция** (см. Рис. 20).

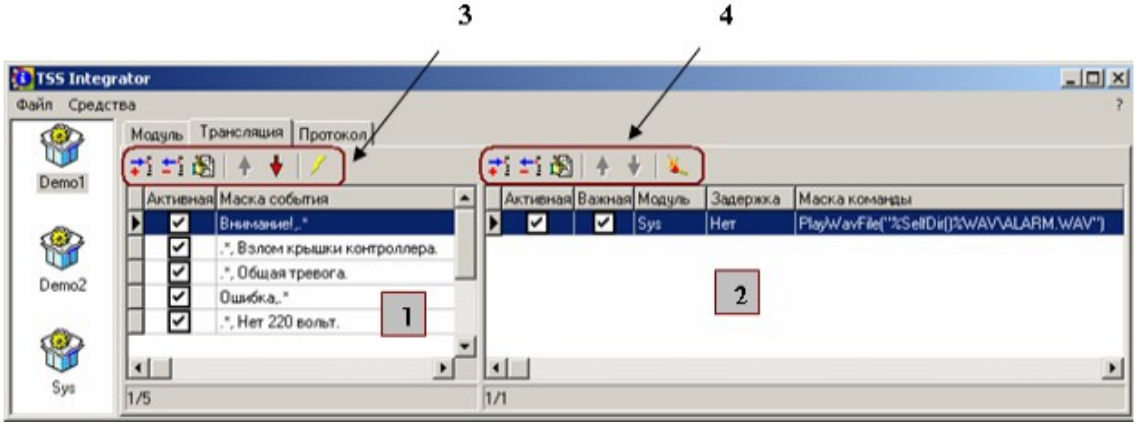


Рис. 20

В окне (**1**) отображается перечень масок событий, в окне (**2**) - перечень масок команд.

Следует напомнить, что каждой маске события соответствует свой набор команд. Для их просмотра следует выбрать в окне (**1**) маску события, и в окне (**2**) отобразится список команд, соответствующих ей.

Например, с системы-передатчика **Demo1** приходит поток данных в виде строки: ***"Внимание! Взлом крышки контроллера"*** на систему ретрансляции. По результатам сравнения данной входной строки со всеми масками события, которые определены в **Интеграторе,** были сгенерированы и отправлены на систему-приемник (посредством модуля-посредника **Sys**) следующие команды:

- PlayWavFile("%SelfDir()%WAV\ALARM.WAV")

– проигрывание звукового файла. Данная команда определена для маски события:

Внимание!,*.

- PlayWavFile("%SelfDir()%WAV\destroyboxc.wav")

– проигрывание звукового файла**.** Данная команда определена для маски события:

.*, Взлом крышки контроллера.

- Exec("CMD", "/C net send %ComputerName()% Включение камеры на запись в серверной. Сообщение о тревоге на пост охраны.", "True", "False")

– выполнение программы, которая присылает на компьютер (имя которого указано в команде) сообщение (см. Рис. 21). Данная команда определена для маски события: **.*, Взлом крышки контроллера**.

–

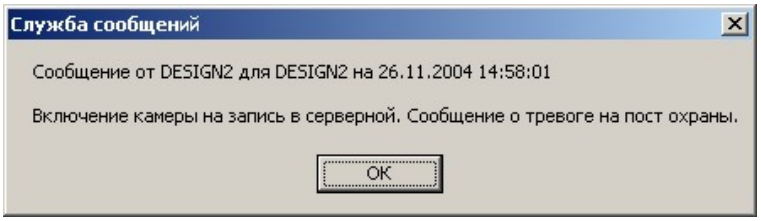


Рис. 21

Чтобы закрыть окно с сообщением, нужно нажать на кнопку **ОК.**

-

-

Все системные функции программы **Интегратор**, которые используются при формировании масок команд, можно просмотреть в окне ***Константы и функции*** (вкладка **Функции**) (см. Рис. 22).

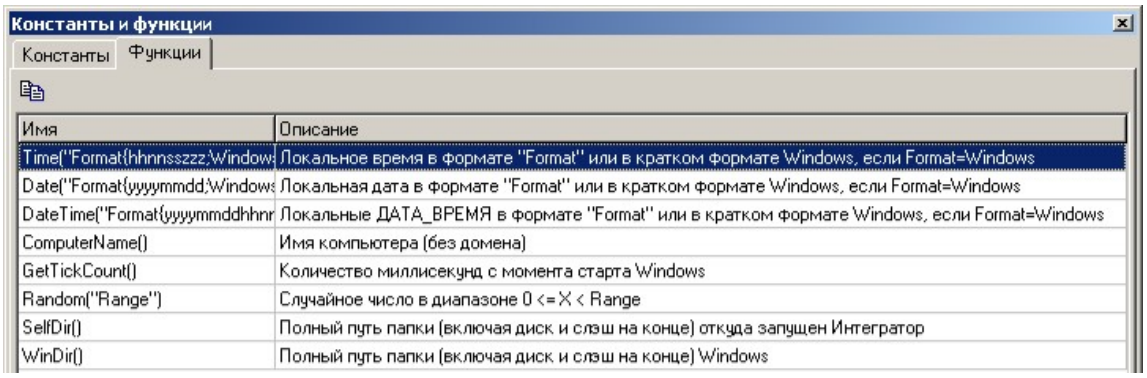


Рис. 22

4. Перейдите во вкладку **Протокол.**

В окне данной вкладки отображаются все записи (входящие (I), исходящие (O), трансляция (T)), т.к. на управляющей панели все опции, соответствующие данным типам событий, включены.

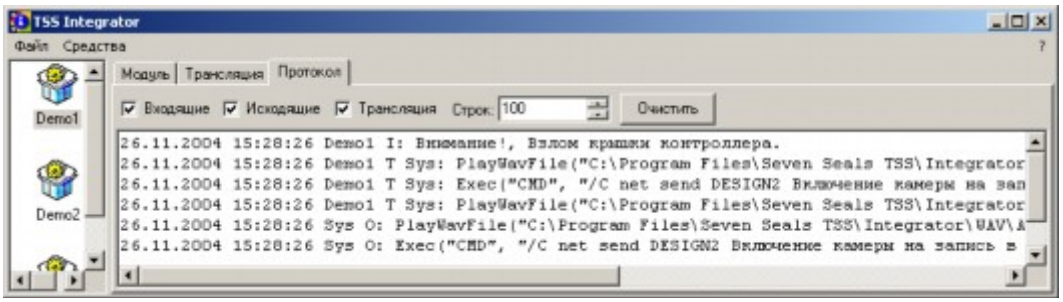


Рис. 23

2) Рассмотрим еще один пример.

Допустим, с системы-передатчика **Demo2** на систему ретрансляции пришел поток данных: “ ***Внимание!, Тревога в помещении "Касса".”***

После сравнения входной строки на предмет соответствия маскам события были сгенерированы и отправлены для выполнения в систему-приемник **Demo1** (посредством одноименного модуля) команды, соответствующие двум маскам события:

- Внимание!,.*

Для данной маски события была определена команда: PlayWavFile("%SelfDir()%WAV\ALARM.WAV"). Данная команда должна быть отправлена системе-приемнику **Demo1** посредством одноименного модуля-посредника.

- `.*,` Тревога в помещении "Касса".

Для данной маски события были определены следующие команды:

– `Exec("CMD", "/C net send %ComputerName()%` Сообщение на пост охраны. Внимание тревога в помещение "Касса". Включить камеры на запись в коридоре и в помещении касса.", "True", "False"). При выполнении данной команды, на компьютер (системы-приемника) должно приходить сообщение:

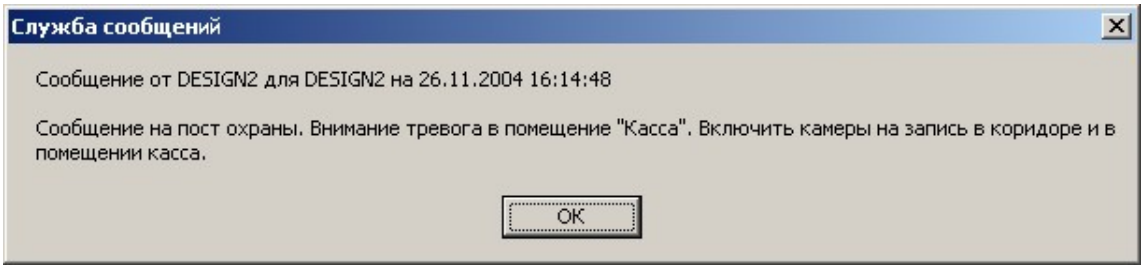


Рис. 24

Чтобы закрыть окно с сообщением, нужно нажать на кнопку **ОК.**

– PlayWavFile("%SelfDir()%WAV\openkassa.wav"). Данная команда должна быть отправлена системе-приемнику **Demo1** посредством одноименного модуля-посредника.

Чтобы убедиться в том, что вышеописанные команды были доставлены посредством модуля **Demo1** одноименной системе-приемнику, нужно выбрать в окне (**1**) модуль **Demo1** (см. Рис. 19) и зайти во вкладку **Модуль**. В окне (**6**) под названием “Принятое” будут отображены две команды (см. Рис. 25):

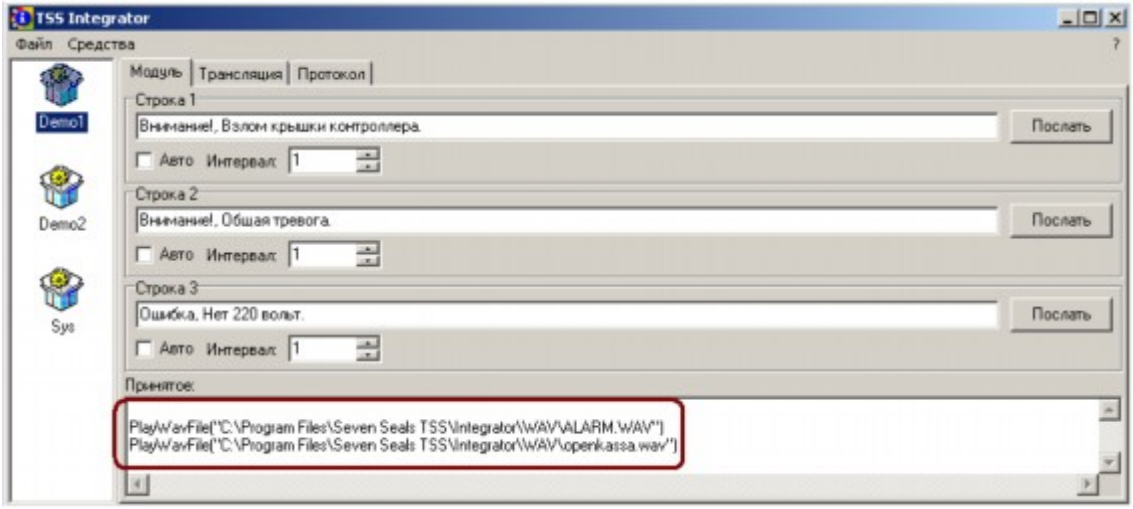


Рис. 25

Аналогичным образом осуществляется взаимодействие посредством программы **Интегратор** между различными внешними системами**.**

7. Приложение

7.1. Команды и функции программы Интегратор

7.1.1. Команды

PlayFile<*Имя файла*> - воспроизвести звуковой файл (расширение WAV). Если в имени файла отсутствует путь, то считается, что он находится в том же каталоге что и файл словаря транслятора;

Exec ("FileName", "Params", "Hide {True;False}", "OnlyIfNotRunning {True;False}") - запускает программу, если OnlyIfNotRunning = True, то запустит только в том случае, если нет запущенного процесса с таким именем файла (без пути).

7.1.2. Функции

Time ("Format{hhnnsszzz;Windows}") – Текущее системное время в формате установленном для Windows;

Date("Format{yyyymmdd;Windows}") - Текущая системная дата в формате установленном для Windows;

DateTime ("Format{yyyymmddhhnnsszzz;Windows}") - Текущая системная дата и время в формате установленном для Windows;

ComputerName () - Имя компьютера (без домена);

GetTickCount () - Количество миллисекунд с момента старта Windows.

Random (“Range”) - Случайное число в диапазоне от 0 до Range.

SelfDir () – Полный путь к папке (включая диск и слэш на конце) откуда запущен **Интегратор.**

WinDir () – Полный путь к папке (включая диск и слэш на конце) Windows.