

Программный комплекс для создания диспетчерских
информационно-управляющих систем реального
времени
«КОТМИ-2010»

Руководство администратора

версия 1.7
Москва 2009г.



СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	3
1. ТЕХНОЛОГИЯ ОБРАБОТКИ ДАННЫХ	4
1.1. ОБРАБОТКА ТЕЛЕМЕТРИИ	4
1.1.1. Обработка телеизмерений (ТИ).....	4
1.1.2. Обработка телесигналов (ТС).....	6
1.1.3. Расчетные ТИ (псевдо-ТИ)	6
1.1.4. Расчетные ТС (псевдо-ТС).....	7
1.1.5. Обработка телеизмерений интегральных (ТИИ).....	8
1.1.6. Телеуправление.....	8
1.2. АРХИВЫ	9
1.2.1. Общие положения	9
1.2.2. Поля таблицы T_ARCH (описатели архивов).....	9
1.2.3. Описание полей данных архивов	11
1.2.4. Создание нового архива	12
1.3. СОБЫТИЯ.....	13
1.3.1. Общие положения	13
1.3.2. Поля таблиц событий	13
1.3.3. Создание нового события в системе	14
1.4. ОРГАНИЗАЦИЯ РАСЧЕТОВ	15
1.4.1. Общие положения	15
1.4.2. Автоматические расчеты	15
1.4.3. Циклические расчеты.....	16
1.5. ОБМЕН ИНФОРМАЦИЕЙ СО СМЕЖНЫМИ УРОВНЯМИ УПРАВЛЕНИЯ	18
1.5.1. Ретрансляция телеизмерений (ТИ).....	18
1.5.2. Ретрансляция телесигналов (ТС).....	18
1.5.3. Макеты формата ЦДУ и АСКП.....	18
1.6. РЕПЛИКАЦИЯ ИНФОРМАЦИИ ВО ВНЕШНЮЮ СУБД	19
2. УСТАНОВКА И СОПРОВОЖДЕНИЕ.....	21
2.1. СЕРВЕР	21
2.1.1. Системные требования.....	21
2.1.2. Установка	21
2.1.3. Файлы серверной части комплекса КОТМИ-2010	21
2.1.4. Файл инициализации сервера	22
2.1.5. Порядок запуска серверов комплекса КОТМИ-2010	24
2.1.6. Останов основного сервера с выключением питания	24
2.1.7. Останов резервного сервера с выключением питания	25
2.1.8. Сжатие SQL-базы сервера ОИК	25
2.1.9. Резервирование в комплексе	25
2.1.9.1. Режим X.....	25
2.1.9.2. Базовая схема резервирования	26
2.1.9.3. Дополнительная схема резервирования	26
2.1.9.4. Отладочный режим работы сервера.....	27
2.2. МОНИТОР	27
2.3. КЛИЕНТ	30
2.3.1. Инсталляция.....	30
2.3.2. Каталоги на диске.....	31
2.3.3. Файл локальных настроек АРМ (Scada.ini).....	32
2.3.4. Обновление файлов	34
3. НАСТРОЙКА ТАБЛИЦ НСИ	36
3.1. Основные понятия	36
3.2. Модуль администратора (ScdAdm.ModAdm)	36
3.3. Универсальный редактор таблиц НСИ (ScdAdm.ModTbl)	39
3.4. Архивы (ScdAdm.ModTblArch).....	41
3.5. Дорасчеты (ScdAdm.ModTblCalc).....	42
3.6. Древовидные структуры (ScdAdm.ModTree)	44
3.7. Команды щита (ScdAdm.ModTblCmd)	45
4. ПОДГОТОВКА ФОРМ И ДОКУМЕНТОВ	46
4.1. Список форм и дерево документов	46
4.2. Редактор форм	47
5. ГРАФИЧЕСКИЙ ИНСТРУМЕНТАРИЙ «МОДУС»	50
5.1. Инструментарий	50

5.2.	Включение схем в систему	50
5.3.	«Оживление» схем.....	51
5.4.	Пометки на схемах	53
6.	ПОЛЬЗОВАТЕЛИ И ПРАВА ДОСТУПА (ScdAdm.ModTblUsrGr)	57
	ПРИЛОЖЕНИЕ 1	59
1	РЕКОМЕНДАЦИИ ПО ВВОДУ КОМПЛЕКСА В ЭКСПЛУАТАЦИЮ	59
1.1	Настройка нормативно-справочной информации ядра комплекса	59
1.2	Настройка таблиц деревьев	60
1.3	Настройка архивов	60
1.4	Порядок ввода информации для приема ТИ, ТС	60
	ПРИЛОЖЕНИЕ 2	61
1	ОСНОВНЫЕ ФУНКЦИОНАЛЬНЫЕ МОДУЛИ КЛИЕНТСКОЙ ЧАСТИ	61
	ПРИЛОЖЕНИЕ 3	62
1	ЯЗЫК ПРОГРАММИРОВАНИЯ TScript	62
1.1	Общий синтаксис.....	62
1.2	Операции и выражения	63
1.3	Операторы	65
1.4	Функции	67
1.5	Исполняющая подсистема	69
	ПРИЛОЖЕНИЕ 4	70
1	ВСТРОЕННЫЕ ОБЪЕКТЫ ЯЗЫКА TScript.....	70
1.1	IntS. Целое с флагами	70
1.2	DbIS. Вещественное с флагами	70
1.3	Res. Запись	70
1.4	Tbl. Таблица	72
	ПРИЛОЖЕНИЕ 5	74
1	Внешние функции TScript, реализованные на сервере	74
	ПРИЛОЖЕНИЕ 6	90
1	ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ ВНЕШНИХ ФУНКЦИЙ ЯЗЫКА TScript	90
1.1	Функция _ArchCurr.	90
1.2	Функции _GblVarGet, _GblVarSet	91
1.3	Пример работы с базой.	92
	ПРИЛОЖЕНИЕ 7	94
1	ФЛАГИ ДЛЯ АРХИВНЫХ ЗНАЧЕНИЙ	94
1.1	Флаги для аналоговых величин.....	94
1.2	Флаги для дискретных величин	95

ВВЕДЕНИЕ

Оперативный информационный управляющий комплекс (далее **ОИУК**) «КОТМИ-2010» – это SCADA система (Supervisory Control And Data Acquisition - диспетчерское управление и сбор данных.), которая обеспечивает:

- Создание АРМ диспетчера, телемеханика, операторов;
- Прием данных и управление технологическими режимами;
- Межуровневый обмен информацией в системе управления;
- Создание активных отчетов и итоговых документов для руководства различного уровня.

Нормативно-справочная информация (НСИ) комплекса представляет собой систему таблиц-описателей, хранимую в реляционной базе данных.

В качестве SQL-базы опробованы:

- База формата .mdb (MS Access);
- MS SQL Server 2000.

Для хранения данных (телеметрия, данные АСКУЭ и т.д.) может использоваться как SQL-база, так и база данных реального времени собственной разработки.

1. ТЕХНОЛОГИЯ ОБРАБОТКИ ДАННЫХ

1.1. ОБРАБОТКА ТЕЛЕМЕТРИИ

1.1.1. Обработка телеизмерений (ТИ)

Настройка параметров обработки ТИ производится в таблице **T_TI** базы данных системы.

Первичная обработка ТИ:

- По приходу ТИ, в случае контроля по времени необновления, устанавливается счетчик необновления ТИ и счетчик времени пропадания УТМ, из которого поступило ТИ.
- Проверяется признак обработки ТИ (флаг **НСИ- TI_WORK** из таблицы T_TI). Если признак не установлен, то обработка ТИ прекращается.
- ТИ проверяется на физические пределы (если флаг **НСИ TI_CON_PHYS=TRUE**). Физические пределы задаются в полях **TI_CMIN** и **TI_CMAX** таблицы T_TI. Для двухсторонних характеристик проверяется отклонение от нулевой точки (величина допустимого отклонения задается в поле **TI_DIGR_KW**). При контроле на нижний физический предел или на отклонение от нулевой точки при двухсторонней характеристике возможен (если флаг **НСИ TI_CON_TS=TRUE**) дополнительный контроль ТИ по состоянию соответствующего телесигнала (ТС).
- Если зафиксировано нарушение, то при состоянии ТС «ОТКЛ», значение ТИ в квантах равно нулю. Этим устраняется дребезг ТИ при отключенной линии, или отключенном генераторе.
- Проверяется скорость изменения значения ТИ (при флаге **НСИ TI_CON_JAMP=TRUE**). Если изменение значения превосходит допустимую величину из поля **TI_JAMP_KW** (в квантах или миллиамперах), то для архивного значения выставляются флаги (соответствующий флаг-признак и флаг общей недостоверности). Затем, если для данного параметра определен дубль, то действующее значение ТИ заменяется на дублирующее и для архивного значения выставляется флаг перехода на дубль.
- Если скачка не было, но есть проверка на устойчивый скачок (флаг **НСИ TI_CON_UST=TRUE**) и был скачок при предыдущем приеме, то для ТИ выставляется соответствующий флаг. В этом случае, флаг общей недостоверности, не выставляется.
- Выполняется масштабирование ТИ. Предусмотрена обработка измерений от датчиков с ломаной характеристикой (два участка масштабирования).
- Фильтрация ТИ (при флаге **НСИ TI_CON_FLT=TRUE**). Если фильтрация необходима, то необходимо задать коэффициент фильтрации (поле **TI_FILTR**). Формула фильтрации следующая:

$$\text{ValTiFlt} = (\text{kFiltr} * \text{ValTi}) + ((1 - \text{kFiltr}) * \text{ValTiPrev}), \text{ где}$$

ValTiFlt- значение ТИ, с учетом
фильтрации;
kFiltr – коэффициент фильтрации;

ValTi - значение ТИ;
ValTiPrev- значение предыдущего ТИ;

¹ До конца раздела 1.1 далее: “флаг” - предполагается признак для архивного значения; если указано “флаг-НСИ” - предполагается значение соответствующего поля из таблицы T_TI

- Проводится проверка значения ТИ на аварийные (флаг **НСИ ТИ_CON_AVR=TRUE**) и технологические (флаг **НСИ ТИ_CON_T=TRUE**) пределы. При нарушении выставляются соответствующие флаги (см. Приложение).

Вторичная обработка ТИ:

- Проверка ТИ на ручное управление (флаг **НСИ ТИ_CON_ARM=TRUE**). Если ТИ на ручном управлении, то значение ТИ заменяется значением из поля **ТИ_VALUE** таблицы **Т_ТИ** и дальнейшая обработка прекращается.
- Проверка ТИ по состоянию УТМ. Если УТМ выключено или неработоспособно, то выставляются соответствующие флаги для ТИ.
- ТИ проверяется на обновление (флаг **НСИ ТИ_CON_R=TRUE**). Константа счетчика обновления задается в поле **ТИ_TIME**.
- Выставляется флаг общей недоверности ТИ, если в процессе предшествующей обработки был выставлен хотя бы один из следующих флагов (анализ происходит в данной последовательности):
 - 1) Флаг вывода ТИ из обработки.
 - 2) Флаг недоверности из-за отключения УТМ.
 - 3) Флаг недоверности из-за отказа УТМ.
 - 4) Флаг недоверности по обновлению.
 - 5) Флаг недоверности при контроле по кодовому значению.
 - 6) Флаг недоверности по скорости изменения.
- Проверка необходимости замены ТИ на дубль. Замена ТИ на дубль возможна в двух случаях:
 - 1) ТИ недоверно. В этом случае просматриваются последовательно все заданные дублирующие значения (1, 2 затем 3) и первое достоверное - становится действующим значением ТИ.
 - 2) Анализируются последовательно флаги НСИ - признаки принудительного перевода на дубль по команде оператора - **ТИ_SIGN_TI**, **ТИ_SIGN_D2**, **ТИ_SIGN_D3** и в случае, если один из них равен **TRUE**, значение данного дубля становится действующим значением ТИ.
- Контроль допустимого отклонения ТИ, переведенного на дубль, от значения этого дубля. Контролируется значение ТИ (значение из Архива ТИ, не замененных на дубль) на допустимое отклонение в % от указанного дублирующего параметра. При этом переведенное на указанный дубль ТИ вернется на собственное значение (не замененное на дубль) при входе его значения в указанный допуск отклонения от значения Дубля и появится флаг 13: «Значение параметра в допустимом диапазоне отклонения от дубля». При этом снимается общий флаг перехода на значение Дубля (15), но сохраняется признак принудительного перевода на дубль по команде оператора (перевод на значение Дубля 1, Дубля 2 или Дубля 3), который удаляется принудительно оператором путем снятия соответствующего флага НСИ. В случае если этот флаг не снят, а значение параметра ТИ выйдет из допустимого диапазона отклонения от значения дубля, его действующим значением вновь станет значение указанного дубля.
- Контроль допустимого отклонения ТИ от плана (флаги **НСИ ТИ_CON_PL=TRUE** или **ТИ_CON_PL_PLUS=TRUE**). При **ТИ_CON_PL=TRUE** контроль производится на отклонение от плана, а при **ТИ_CON_PL_PLUS=TRUE** на превышение плана. Величина отклонения задается в поле **ТИ_DIGR_PL** в процентах. В случае нарушения выставляется соответствующий флаг.

- При флаге **НСИ TI_CON_ARX=TRUE** производится контроль на наличие событий по данному ТИ. Генерируемые события перечислены в таблице **T_EV_COD**.

1.1.2. Обработка телесигналов (ТС)

Настройка параметров обработки ТС - в таблице **T_TS** базы данных системы.

Первичная обработка ТС:

- По приходу ТС значение, если необходимо (флаг **НСИ TS_INV=TRUE**), инвертируется и восстанавливается счетчик времени пропадания УТМ, из которого поступил данный ТС.
- Проверяется признак обработки ТС (флаг **НСИ TS_WORK** таблицы **T_TS**). Если признак не установлен, то обработка ТС прекращается.
- Фильтрация ТС (флаг **НСИ TS_FILT=TRUE**). Рекомендуются только для циклических УТМ. Фильтрация происходит по схеме 2 из 3. Формула фильтрации следующая:

$ValTsFlt = (ValTs \& OldValTs) | (ValTs \& OldValTsFlt) | (OldValTs \& OldValTsFlt)$, где:

ValTsFlt- значение ТС, с учетом
фильтрации ;
ValTs - значение ТС;

OldValTs - значение предыдущего ТС;
OldValTsFlt - значение предыдущего ТС, с
учетом фильтрации ;

Вторичная обработка ТС:

- Проверка ТС на ручное управление (флаг **НСИ TS_CON_ARM=TRUE**). Если задано ручное управление, то производится замена значения ТС значением из поля **TS_VALUE**.
- Контроль ТС по состоянию УТМ. Если УТМ выключено или неработоспособно, то выставляются соответствующие флаги.
- Выставляется флаг общей недостоверности ТС, если уже выставлен, в процессе предшествующей обработки, хотя бы один из следующих флагов (анализ происходит в соответствующей последовательности):
 - 1) Флаг вывода ТС из обработки.
 - 2) Флаг недостоверности из-за отключения УТМ.
 - 3) Флаг недостоверности из-за отказа УТМ.
- При флаге **НСИ TS_REG=TRUE** производится контроль на наличие событий по данному ТС. Генерируемые события описаны в таблице **T_EV_COD**.

1.1.3. Расчетные ТИ (псевдо-ТИ)

Настройка параметров обработки псевдо-ТИ - в таблице **T_PTI** базы данных системы.

Псевдо-ТИ считаются в темпе поступления входящих в них составляющих (ТИ, данные суточной ведомости и т. д.). Формируется флаг общей недостоверности псевдо-ТИ на основе выполнения функции «логическое ИЛИ» над флагами общей недостоверности всех входящих в расчет параметров. Затем подвергаются следующей обработке:

- Проверка псевдо-ТИ на аварийные (флаг **НСИ PTI_CON_AVR=TRUE**) и технологические (флаг **НСИ PTI_CON_T=TRUE**) пределы.
При нарушении пределов, устанавливаются соответствующие флаги.

- Проверка необходимости замены псевдо-ТИ на дубль. Замена псевдо-ТИ на дубль возможна в двух случаях:
 - 1) Псевдо-ТИ недостоверно. В этом случае просматриваются последовательно все заданные дублирующие значения (1, 2 затем 3) и первое достоверное становится действующим значением.
 - 2) Анализируются последовательно флаги НСИ - признаки принудительного перевода на дубль по команде оператора - **PTI_SIGN_TI**, **PTI_SIGN_D2**, **PTI_SIGN_D3** - и в случае, если один из них равен TRUE, -значение данного дубля становится действующим значением.
- Контроль допустимого отклонения псевдо-ТИ, переведенного на дубль, от значения этого дубля. Контролируется значение псевдо-ТИ (значение из Архива псевдо-ТИ, не замененных на дубль) на допустимое отклонение в % от указанного дублирующего параметра. При этом переведенное на указанный дубль псевдо-ТИ вернется на собственное значение (не замененное на дубль) при входе его значение в указанный допуск отклонения от значения Дубля и появится флаг 13: «Значение параметра в допустимом диапазоне отклонения от дубля». При этом снимается общий флаг перехода на значение Дубля (15), но сохраняется признак принудительного перевода на дубль по команде оператора (перевод на значение Дубля 1, Дубля 2 или Дубля 3), который удаляется принудительно оператором путем снятия соответствующего флага НСИ. В случае если этот флаг не снят, а значение параметра псевдо-ТИ выйдет из допустимого диапазона отклонения от значения дубля, его действующим значением вновь станет значение указанного дубля.
- Контроль допустимого отклонения псевдо-ТИ от плана (флаги НСИ **PTI_CON_PL=TRUE** или **PTI_CON_PL_PLUS=TRUE**). При **PTI_CON_PL=TRUE** контроль производится на отклонение от плана, а при **PTI_CON_PL_PLUS=TRUE** на превышение плана. Величина отклонения задается в поле **PTI_DIGR_PL** в процентах. В случае нарушения выставляется соответствующий флаг.
- При флаге НСИ **PTI_CON_ARX=TRUE** производится контроль на наличие событий по данному псевдо-ТИ. Генерируемые события описаны в таблице **T_EV_COD**.

1.1.4. Расчетные ТС (псевдо-ТС)

Настройка параметров обработки псевдо-ТС - в таблице **T_PTS** базы данных системы.

Псевдо-ТС считаются в темпе поступления входящих в них составляющих ТС. Формируется флаг общей недостоверности псевдо-ТС на основе выполнения функции «логическое ИЛИ» над флагами общей недостоверности всех входящих в расчет параметров. Затем подвергаются следующей обработке:

- Псевдо-ТС инвертируется, если флаг **НСИ PTS_INV=TRUE**.
- Фильтрация псевдо-ТС (флаг **НСИ PTS_FILT=TRUE**). Фильтрация происходит по схеме 2 из 3. Формула фильтрации следующая:

$$\text{ValPtsFlt} = (\text{ValPts} \ \& \ \text{OldValPts}) \mid (\text{ValPts} \ \& \ \text{OldValPtsFlt}) \mid (\text{OldValPts} \ \& \ \text{OldValPtsFlt}),$$
 где:

ValPtsFlt- значение ТС, с учетом
фильтрации ;
ValPts - значение ТС;

OldValPts - значение предыдущего ТС;
OldValPtsFlt - значение предыдущего ТС, с
учетом фильтрации ;

- При флаге **НСИ PTS_REG=TRUE** производится контроль на наличие событий по данному псевдо-ТС. Генерируемые события описаны в таблице **T_EV_COD**.

1.1.5. Обработка телеизмерений интегральных (ТИИ)

Настройка параметров обработки ТИИ - в таблице **T_TII** базы данных системы.

Выполняется следующая обработка:

- Проверяется признак обработки данного ТИИ (флаг **НСИ TII_WORK** таблицы **T_TII**). Если признак не установлен, то обработка данного ТИИ прекращается.
- Выполняется масштабирование ТИИ, если флаг **НСИ TII_SIGN_NOIMP=FALSE**. Масштабный коэффициент задается в поле **TII_IMP**

1.1.6. Телеуправление

Телеуправление позволяет операторам дистанционно управлять оборудованием.

Эта функция доступна через схемы и документы клиентской части.

Оператор выдает команду на исполнение через событие с кодом 901, причем в качестве параметра передается идентификатор соответствующего ТС в базе.

Возможность телеуправления ТС определяется флагом **TS_TU** таблицы **T_TS** базы данных.

Если номер группы и номер в группе для данного ТС не совпадает с командой телеуправления - выставляется признак **TS_TU_READR** в таблице **T_TS**, в этом случае адрес для телеуправления должен формироваться с помощью таблицы **T_TU** базы данных.

1.2. АРХИВЫ

1.2.1. Общие положения

Подсистема архивирования предоставляет услуги по организации хранения, доступа и поиска данных, поисковый образ которых является функцией времени.

Архивирование данных возможно по срезам или по изменениям. Архивирование по изменениям предпочтительнее для быстроменяющихся и нерегулярно поступающих данных (ТИ, ТС, псевдо-ТИ, псевдо-ТС), так как в этом случае:

- Системные ресурсы не тратятся на запись не изменяющихся данных;
- Записываются все переходные изменения в данных, в отличие от систем записи срезов, управляемых по времени, которые могут пропустить критические события, возникающие между срезами.

Описания архивов находятся в таблице **T_ARCH**.

Для каждого архива в базе создается таблица, описывающая поля архива и их типы.

1.2.2. Поля таблицы T_ARCH (описатели архивов)

- **ARCH_ID**. Идентификатор архива (порядковый номер описателя в таблице);
- **ARCH_CON_WORK**. Задействованность архива. Если значение этого поля FALSE, то архив исключается из обработки и запросы к нему не поддерживаются.
- **ARCH_CON_RES**. Если значение этого поля TRUE, то данные, записываемые в этот архив, автоматически реплицируются на остальные сервера многомашинного комплекса.
- **ARCH_OBJ_ID**. Идентификатор таблицы, описывающей поля архива. Данное поле является идентификатором записи в таблице T_OBJ.
- **ARCH_NAME**. Наименование архива.
- **ARCH_CON_SQL_DIR**. Признак записи данных архива непосредственно в таблицу SQL-базы. На данный момент не поддерживается, так как существующие СУБД не обладают приемлемым быстродействием.
- **ARCH_CON_SQL**. Признак того, что весь архив или его часть хранятся в SQL-сервере. Данные пишутся в таблицу SQL-базы, в поле типа BLOB в упакованном виде. Все таблицы для хранения данных в SQL-базе имеют одинаковую структуру, описанную ниже.
- **ARCH_CON_RT**. Признак того, что весь архив или его часть хранятся в базе реального времени.
- **ARCH_CON_CHANGE**. Признак того, что это архив с записью изменений. Для архива с записью изменений раз в заданное время, определяемое скважностью (циклом) архива записывается срез данных, а далее к нему дописываются все изменения. Например, для архива ТИ можно указать цикл архива равным 10 минутам. Это значит, что срез всех данных будет записываться раз в 10 минут, а далее к нему будут дописываться только изменения. Это позволяет значительно экономить дисковое пространство и повышать суммарное быстродействие системы за счет меньшего количества обращений к диску для чтения данных приложениями комплекса.
- **ARCH_CON_MON**. Признак архива с циклом, кратным одному месяцу. Имеется в виду календарный месяц, с учетом разного количества дней в месяце.

Данный признак имеет высший приоритет и перекрывает все остальные данные о цикличности архива.

- **ARCH_CON_WEEK.** Признак архива с циклом, кратным энергетической неделе. Для работы такого архива необходимо ведение энергетического календаря (модуль «Календарь»), в котором должны быть определены энергетические недели. Данный признак имеет приоритет ниже, чем ARCH_CON_MON, но перекрывает все остальные данные о цикличности архива.
- **ARCH_CON_SHIFT.** Признак архива с циклом, кратным смене. Для работы такого архива необходимо ведение энергетического календаря (модуль «Календарь»), в котором должны быть определены смены. Данный признак имеет приоритет ниже, чем ARCH_CON_WEEK, но перекрывает все остальные данные о цикличности архива.
- Данные об общей глубине хранения архива. Поля **ARCH_DEPTH_MON, ARCH_DEPTH_WEEK, ARCH_DEPTH_DAY, ARCH_DEPTH_HOUR, ARCH_DEPTH_MIN, ARCH_DEPTH_SEC.** Общая глубина хранения архива (в секундах) вычисляется по следующей формуле:

$$(ARCH_DEPTH_MON) * 31 * 24 * 60 * 60 + (ARCH_DEPTH_WEEK) * 7 * 24 * 60 * 60 + (ARCH_DEPTH_DAY) * 24 * 60 * 60 + (ARCH_DEPTH_HOUR) * 60 * 60 + (ARCH_DEPTH_MIN) * 60 + (ARCH_DEPTH_SEC).$$
- Данные о глубине хранения архива в базе реального времени. Поля **ARCH_RT_MON, ARCH_RT_WEEK, ARCH_RT_DAY, ARCH_RT_HOUR, ARCH_RT_MIN, ARCH_RT_SEC.**
 Общая глубина хранения архива в базе реального времени в секундах вычисляется по формуле, аналогичной приведенной выше.

Если данные архива хранятся только в базе реального времени, то данные о глубине архива в базе реального времени должны совпадать с данными об общей глубине хранения архива.

Если данные архива хранятся и в SQL-базе и в базе реального времени (т.е. **ARCH_CON_SQL= TRUE** и **ARCH_CON_RT=TRUE**), то самые свежие записи архива хранятся в базе реального времени с глубиной, равной глубине хранения в архиве реального времени. По мере старения записи переписываются в SQL-базу. Глубина хранения в SQL-базе равна общей глубине хранения минус глубина хранения в архиве реального времени.

Если данные архива хранятся только в SQL-базе, то глубина хранения равна общей глубине хранения данных в архиве.

- **ARCH_RT_PATH.** Месторасположение файлов базы реального времени для данного архива, если **ARCH_CON_RT=TRUE**. Указывается путь к файлам. Например: C:\ArchOic\. Данное поле позволяет располагать файлы базы реального времени на разных дисках и в разных папках, если это необходимо.
- **ARCH_NCI_OBJ_ID.** Идентификатор таблицы с НСИ для данного архива в таблице T_OBJ.
- **ARCH_PRM.** Параметры отображения для данного архива. Используется клиентской частью комплекса для хранения ряда параметров. Например: флаги архивов.
- **ARCH_CON_CALC.** Признак архива с расчетными данными (если TRUE). Например псевдо-ТИ, псевдо-ТС и т.д. Если признак равен FALSE, то это архив с атомарными данными. Например ТИ, ТС, ТИИ и т.д.

- **ARCH_CON_WR_CALC.** Признак архива с хранимыми расчетными данными (если TRUE). Если FALSE, то это архив с виртуальными расчетными данными. То есть эти данные рассчитываются и доставляются клиенту в момент запроса, без их сохранения на физический носитель. Данные, рассчитываемые таким образом, должны согласовываться с глубиной хранения данных, по которым проводится расчет. Например, если в расчете используется данное из архива с глубиной хранения одни сутки, то при запросе расчетного параметра за время, более чем одни сутки назад, будет естественно получен неверный результат.
- **ARCH_CALC_PREF.** Если необходимо, чтобы данный архив участвовал в системе автоматических расчетов (т. е. при записи параметра в базу автоматически определяется, в какие расчеты он входит, расчеты выполняются и результаты пишутся в соответствующие архивы), надо в это поле записать имя функции доступа для архива с атомарными величинами (например для часового архива ведомости с данными ТИ записана функция доступа TI) или префикс имени функции для архива с расчетными данными (например для часового архива ведомости с данными ПТИ записан префикс функции PTI). Подробности описаны в «Организация расчетов дополнительных параметров». Для архивов с данными ТИ, ТС, псевдо-ТИ, псевдо-ТС это поле не заполняется, так как соответствующие расчеты проводятся сервером ввода-вывода.
- **ARCH_CON_DTCP.** Признак того, что при записи данных используется время, полученное с низового УТМ. На данный момент не задействован, зарезервирован на будущее.
- **ARCH_CON_SYN.** Признак необходимости синхронизации записей данного архива с основного сервера комплекса при запуске данного сервера.
- **ARCH_SYN_REC_FORW, ARCH_SYN_REC_BACK.**
Если ARCH_CON_SYN=TRUE, то количество записей соответственно вперед и назад, считываемых с основного сервера для синхронизации архива. Например, если для архива ТИ с записью изменений указан цикл архива 10 минут, то в этом случае для того, чтобы синхронизировать данный архив на 1 час, необходимо ARCH_SYN_REC_BACK поставить равным шести. Необходимость синхронизации вперед возникает, например, для архивов с плановыми данными, которые могут быть записаны на несколько дней вперед.

1.2.3. Описание полей данных архивов

В данных архивов возможно использование следующих полей:

- **ID** – идентификатор параметра (тип - целое или длинное целое);
- **DT** – время записи параметра в архив (тип – длинное целое);
- **DTCP** – время фиксации параметра в УТМ для тех УТМ, которые поддерживают эту возможность (тип - длинное целое);
- **VAL** – значение параметра (тип – логический, байт, целое, длинное целое, плавающее одинарной точности, плавающее двойной точности, текстовое поле, MEMO-поле, BLOB). Для типов текстовое поле, MEMO и BLOB корректировка записанных данных не допускается;
- **FLG** – флаги параметра (тип – целое, длинное целое);
- **COUNT** – используется во вспомогательных архивах для хранения количества суммирований данного параметра при расчете, например, среднего или интегрального значения на заданном временном интервале. Тип – длинное целое;

- **IDPER** – индекс текущего периода расчета. Используется во вспомогательных архивах для определения завершения периода расчета, например, при расчете интегральных значений. Тип – длинное целое.

1.2.4. Создание нового архива

Для создания нового архива необходимо выполнить следующие действия:

- Создать таблицу с описанием полей архива по правилам пункта 1.2.3 и прописать данную таблицу в **T_OBJ** с соответствующим типом таблицы (тип для архивов - 200);
- Если в таблице **T_ARCH** установить признак **ARCH_CON_SQL**, то данные архива будут храниться в таблице реляционной базы данных, описанной в пункте 3.3 документа «Структура базы данных оперативно-информационного комплекса». Имя таблицы для соответствующего архива должно состоять из имени таблицы, на которую указывает поле **ARCH_OBJ_ID** таблицы **T_ARCH**, плюс «_A». Все действия по созданию нового архива выполняются в настоящее время средствами используемой СУБД.
- Если в таблице **T_ARCH** установить признак **ARCH_CON_RT**, то данные архива будут храниться в базе реального времени. В поле **ARCH_RT_PATH** необходимо указать местоположение файлов базы реального времени для данного архива. Файлы будут созданы автоматически после перезагрузки сервера ОИК. Их будет два: с расширением «.rti» (индексы для файла с данными), и с расширением «.rtb» (данные). Имена файлов будут те же, что и у таблицы, которая определена в поле **ARCH_OBJ_ID** таблицы **T_ARCH**, эта таблица используется для описания формата данных архива.
- Если в таблице **T_ARCH** установить оба признака: **ARCH_CON_SQL** и **ARCH_CON_RT**, то часть архива будет храниться в базе РВ, а часть в реляционной БД в соответствии с установленными характеристиками глубины хранения.
- В **T_ARCH** описать характеристики архива (глубина, цикличность и т. д. – смотри пункт 1.2.2 настоящего документа).

Новый архив будет доступен после перезапуска сервера ОИК.

1.3. СОБЫТИЯ

1.3.1. Общие положения

Все события в комплексе разбиты по категориям.

Описание категорий событий - в таблице **T_EV_CAT** базы данных. Внутри каждой категории может быть описано произвольное количество кодов событий.

Коды событий описываются в таблице **T_EV_COD**. Любой код события может быть выведен из обработки, если поле **EV_COD_WORK** для соответствующего кода события перевести в состояние FALSE. Любой код события может быть объявлен аварийным, то есть требующим квитирования (подтверждения об оповещении), если для него поле **EV_COD_ALARM=TRUE**.

Определены имена полей таблиц событий в системе (пункт 1.3.2). Эти имена используются модулем «События» из АРМ-а клиента. Вы можете вводить дополнительные поля для описания событий, но в этом случае они будут доступны только для программ пользователя, который ввел эти поля.

Генерация события, оповещение подписанных на него клиентов, происходит в момент записи информации в архив событий - таблицу, прописанную в таблице **T_OBJ** базы данных с типом 300.

Таким образом, за счет стандартного интерфейса перечень событий можно легко расширять.

Есть возможность генерировать события без записи их в базу данных. Для этого надо описать в таблице **T_EV_COD** необходимые события. Таблица для хранения событий создавать не над. Генерации события происходит с помощью серверной процедуры «**GEN_EV_NO_WRITE**». В таком событии обязательно должны присутствовать поля **EV_EV_COD_ID** и **EV_ENOBJ_ID**.

1.3.2. Поля таблиц событий

Определены следующие поля:

- **EV_ID** – идентификатор записи в таблице (тип – длинное целое);
- **EV_TIME** – время фиксации события в ОИК (тип – длинное целое);
- **EV_ENOBJ_ID** – идентификатор объекта, на котором произошло событие (тип – длинное целое);
- **EV_EV_COD_ID** – код события. Ссылка на поле **EV_COD_ID** таблицы **T_EV_COD** (тип – длинное целое);
- **EV_PAR_ID** – идентификатор параметра, вызвавшего событие (тип – длинное целое);
- **EV_PAR_NAME** – наименование параметра, вызвавшего событие (тип – текст);
- **EV_KWIT** – признак необходимости квитирования. Если TRUE, то надо квитировать. Выставляется приложением, генерирующим событие, на основе анализа поля **EV_COD_ALARM** таблицы **T_EV_COD**. Тип - логический;
- **EV_USRS_ID** – идентификатор пользователя, генерирующего событие (тип – длинное целое);
- **EV_KWIT_ID** – идентификатор записи в таблице с квитируемым событием. Данное поле присутствует только в таблице **T_EV_KWIT**, в которую записывается протокол квитирования. Тип – длинное целое;

- **EV_KWIT_OBJ_ID** – идентификатор таблицы с квитируемым событием. Данное поле присутствует только в таблице T_EV_KWIT, в которую записывается протокол квотирования. Тип – длинное целое;
- **EV_VALUE** – для хранения некоторого значения. Например, для событий обработки телеизмерений здесь хранится значение ТИ, для события запроса передачи макета – время, за которое необходимо передать макет и т.д. Тип определяется необходимостью;
- **EV_LIM** – для хранения пороговых значений. Например, для событий обработки ТИ здесь величина нарушенного аварийного или технологического предела. Тип числовой;
- **EV_COMMENT** – комментарий к событию. Тип текстовый или MEMO;
- **EV_BLOB** – дополнительная информация к событию в виде поля типа BLOB.

1.3.3. Создание нового события в системе

Все описанное ниже необходимо выполнить с помощью средств используемой реляционной СУБД. Изменения будут доступны после рестарта сервера ОИК.

Порядок действий следующий:

- Если событие не подходит ни к одной категории событий, уже описанных в комплексе, то необходимо:
 - 1) Создать таблицу для хранения событий данной категории с необходимыми полями (пункт 1.3.2);
 - 2) Добавить запись о созданной таблице в таблицу T_OBJ, причем тип вновь созданной таблицы должен быть равен 300.
 - 3) Добавить описание новой категории в таблицу T_EV_CAT базы данных комплекса.
- Добавить запись о новом событии в таблицу T_EV_COD.
- Обеспечить из своего приложения передачу на сервер (т.е. запись в таблицу событий) сообщений о вновь созданных событиях.

1.4. ОРГАНИЗАЦИЯ РАСЧЕТОВ

1.4.1. Общие положения

Все расчеты, проводимые в комплексе, описываются в таблице **T_CALC** базы данных на языке **TScript**. Возможности языка **TScript** расширены с помощью библиотеки внешних функций (описаны в приложении). Расчеты выполняются в трех случаях:

- 1) Автоматически, если в архив записываются параметры, которые участвуют в каких-то расчетах. Результаты расчетов записываются в соответствующие архивы с расчетными параметрами (пункт 1.4.2);
- 2) Через механизм виртуальных расчетных архивов (**ARCH_CON_CALC=TRUE** и **ARCH_CON_WR_CALC=FALSE**).

Для них расчеты выполняются при каждом запросе данных из этого архива. Для этого необходимо в поле **ARCH_CALC_PREF** указать *префикс функции* для данного архива, например **RASCV**. В этом случае при запросе данного с **ID = 1** из этого архива будет запущена функция **RASCV1** и результаты расчета будут возвращены клиенту.

- 3) Циклически. Для чего разработана подсистема циклических расчетов (пункт 1.4.3).

1.4.2. Автоматические расчеты

Для организации автоматических расчетов приняты ряд соглашений, благодаря которым имеется возможность определить, в какие расчеты входит записываемое в архив данное, выполнить эти расчеты и записать результаты в соответствующий архив.

Чтобы эта возможность поддерживалась, необходимо для архива с нерасчетными параметрами (**ARCH_CON_CALC=FALSE**) указать в таблице **T_ARCH** в поле **ARCH_CALC_PREF** имя функции доступа к параметрам данного архива.

Например - **TI**. В этом случае при записи данного в этот архив определяется, в каких расчетах используется функция **TI** с двумя параметрами – **TI** (**IdPar**, **pTime**), где **IdPar** – идентификатор данного, **pTime** – время расчета. Затем эти расчеты выполняются.

Если архивы идентичны по составу данных, то есть для них назначена одна и та же таблица **НСИ**, то можно использовать для них одну и ту же функцию доступа, что позволяет использовать одни и те же описания расчетов для разных архивов.

Например: функция доступа **TI** определяется следующим образом:

```

TI (pId,pTime)
{
    if(CMP(_ArchCurr(), "T_ARCH_PTI")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_TI");
    }
    else
    {
        Result = ROUND(_ReadData(pId,pTime,"T_ARCH_V_H_TI"), 0);
    }
};

```


В этой функции сначала определяется, для какого архива производится расчет (с помощью функции `_ArchCurr`) и после этого производится чтение значения ТИ или из архива ТИ или из архива часовой ведомости с данными ТИ.

Результаты расчета укладываются в соответствующий архив с расчетными данными (`ARCH_CON_CALC=TRUE` и `ARCH_CON_WR_CALC=TRUE`).

Для определения архива с расчетными параметрами необходимо в поле `ARCH_CALC_PREF` указать *префикс функции* для данного архива.

Например: для архива часовой ведомости с данными ПТИ это - `PTI10`. В этом случае, если выполняется расчет, который является функцией с именем `PTI10`, то это значит, что результат будет записан именно в архив часовой ведомости с данными ПТИ и идентификатор записываемого данного – 10.

Если архивы с расчетными данными идентичны по составу данных, то есть для них назначена одна и та же таблица НСИ, то можно использовать для них один и тот же префикс функции, что позволяет использовать одни и те же описания расчетов для разных архивов.

Например, одна и та же функция:

```
PTI10 (pTime)
{
    TI (1, pTime) + TI (2, pTime);
}
```

может использоваться при расчете псевдо-ТИ, данного архива часовой ведомости с псевдо-ТИ, данного архива получасовой ведомости с псевдо-ТИ. Только, естественно, соответствующим образом должна быть скорректирована функция доступа `TI` с тем, чтобы данные для расчетов брались из нужных архивов.

Есть исключение для архивов с ТИ, ТС, псевдо-ТИ и псевдо-ТС. Для них функции доступа и префиксы функций не указываются, так как все необходимые расчеты выполняются в сервере ввода-вывода и затем посчитанные значения подвергаются дополнительной обработке.

1.4.3. Циклические расчеты

Данная подсистема позволяет без написания дополнительных программ решить множество задач. Все, что должно циклически считаться, можно реализовать с помощью этой подсистемы.

Например:

- Интегрирование параметра с подсчетом среднего значения или электроэнергии по завершении периода интегрирования;
- Циклический расчет потерь мощности и электроэнергии в различных элементах электрической сети;
- Циклический контроль заданной совокупности параметров, с последующей выдачей сообщений об изменении их состояния;
- Заполнение архивов данными других архивов с заданной циклическостью;
- и так далее.

Для ввода нового циклического расчета необходимо:

- Создать новую таблицу с НСИ циклических расчетов. В такой таблице должно быть три поля. Существенным являются окончания названий полей (**ID**, **NAME**,

CALC_ID). По ним идентифицируется информация в полях. Остальная часть наименований полей несущественна.

Например:

- 1) **CALC_C_D_1_ID** - идентификатор расчетного параметра (тип – длинное целое);
 - 2) **CALC_C_D_1_NAME** - наименование расчетного параметра (тип - текст);
 - 3) **CALC_C_D_1_CALC_ID** - идентификатор расчета. Ссылка на **T_CALC**. Тип – длинное целое.
- Включить описание таблицы в таблицу описания таблиц базы ОИК (**T_OBJ**).
 - Включить новую настройку в таблицу настроек циклических расчетов **T_CALC_C**. Необходимо правильно описать поля, задающие период расчета, цикл расчета и задержку расчета.
 - В таблице **T_CALC** описать функции, которые надо циклически выполнять и включить их номера в созданную таблицу с НСИ циклических расчетов.

1.5. ОБМЕН ИНФОРМАЦИЕЙ СО СМЕЖНЫМИ УРОВНЯМИ УПРАВЛЕНИЯ

1.5.1. Ретрансляция телеизмерений (ТИ)

Ретранслируемые данные описываются в таблице **T_TIRETR** базы данных.

Если в описании ретранслируемого ТИ в таблице **T_TITETR** поля **TIRETR_MIN_KW** и **TIRETR_MAX_KW** равны нулю, то величина будет передана на ретрансляцию в именованном виде. Это необходимо, например, для выдачи информации на диспетчерский щит. Можно ретранслировать информацию из любых архивов комплекса.

Возможна смена знака ретранслируемого данного (поле **TIRETR_INV=TRUE**).

Количество информационных бит в квантованной величине задается в поле **TIRETR_LONG_INF**. Можно задать формирование знакового разряда (поле **TIRETR_SIGN=TRUE**). Это необходимо, например, при ретрансляции информации в протоколе TM-800A (8 бит информационных + 1 знаковый).

1.5.2. Ретрансляция телесигналов (ТС)

Ретранслируемые данные описываются в таблице **T_TSRETR** базы данных. ТС объединяются в группы, причем в группе должно быть не более 8 ТС.

Наборы цифробуквенной информации (ЦБИ).

Наборы ЦБИ описываются в таблице **T_CBI**, а данные из наборов в таблице **T_CBID** базы данных.

Обменом ЦБИ занимается главный сервер ввода-вывода (IO) комплекса. В таблице **T_CALC** описана функция с именем **MaketCbiWork** (**не корректировать эту функцию**), на вход которой поступает принятый макет. Эта функция определяет тип принятого макета (данные, квитанция, запрос), ищет описатель данного макета в таблице **T_CBI**, определяет номер функции в **T_CALC**, обрабатывающей данный макет и запускает ее.

Передача макетов инициируется оператором или автоматически, в соответствии с описателем макета. Макет на передачу также формируется с помощью соответствующей функции из таблицы **T_CALC** (идентификатор функции прописывается в описателе макета).

1.5.3. Макеты формата ЦДУ и АСКП

Для работы с макетами формата ЦДУ и АСКП служит программа **ScdMail.exe**. Программа работает как одна из серверных задач и автоматически запускается в момент запуска основного сервера комплекса, на котором в таблице **T_SPRG** прописан ее запуск. Она автоматически принимает и обрабатывает (вводит данные в базу комплекса) макеты, в соответствии с описанием макета, хранимым в базе.

Программа использует протокол POP3.

В работе программа использует ini-файл (**ScdMail.ini**), который должен находиться в той же папке, откуда запускается программа. Структура ini-файла:

[Почта] – имя раздела;

Адрес отправителя=user@en.elektra.ru - задействован для будущих применений;

Имя=mail – имя пользователя для подключения к почтовому серверу;

Пароль=mail – пароль пользователя для подключения к почтовому серверу;

Почтовый сервер=172.18.34.1 – IP адрес почтового сервера;

Интервал проверки=60 – цикл проверки почтового ящика в секундах;

[Scada] – имя раздела;

Сервер=SrvScada – имя или IP адрес сервера ОИК, к которому подключается программа;

Порт=1212 – порт сервера ОИК, к которому подключается программа;

Имя пользователя=User – имя пользователя для подключения к серверу ОИК;

Пароль=User – пароль пользователя для подключения к серверу ОИК.

1.6. РЕПЛИКАЦИЯ ИНФОРМАЦИИ ВО ВНЕШНЮЮ СУБД

Для этой цели служит программа **ScdRepl.exe**. Программа работает как одна из серверных задач и автоматически запускается в момент запуска основного сервера комплекса, на котором в таблице T_SPRG прописан ее запуск. Оформлена как консольное приложение.

В работе программа использует ini-файл (**ScdRepl.ini**), который должен находиться в той же папке, откуда запускается программа.

Данная программа позволяет реплицировать в заданную БД SQL таблицы НСИ, события и изменения архивов.

Поддерживается обработка только архивов записываемых циклически (по срезам).

Предварительно, для всех требуемых данных, нужно подготовить принимающие таблицы во внешней СУБД.

Имена таблиц во внешней СУБД должны совпадать с именами таблиц в ОИК.

Имена полей в таблицах должны совпадать с именами полей соответствующих таблиц ОИК.

Наличие некоторых полей обязательно, остальные могут присутствовать по необходимости (см. комментарии в ini-файле).

Программа подключается одновременно к серверу ОИК (раздел [Сервер]), серверу БД SQL ([База данных]) и осуществляет репликацию данных. Если очистить таблицы, то они будут заполнены данными заново.

Структура ini-файла:

[Сервер]

Адрес=192.168.3.246

Порт=1212

Имя пользователя=EVA

Пароль=123

[База данных]

Подключение=Provider=Microsoft.Jet.OLEDB.4.0;Data

Source=D:\Scd\ScdRepl\ScdRepl.mdb;Persist Security Info=False

//**Подключение**=Provider=MSDAORA.1;User ID=ADMIN;Data Source=XXX;Persist Security Info=False

//**Подключение**=Provider=MSDAORA.1;Password=xxx;User ID=xxx;Data Source=xxx;Persist Security Info=True

[Таблицы НСИ]

; Первое поле таблицы НСИ(идентификатор,INT(4)) должно присутствовать в таблице БД

T_TI=T_TI

T_TS=T_TS

[События]

; Должны присутствовать

поля(INT(4)):EV_ID,EV_TIME,EV_ENOBJ_ID,EV_EV_COD_ID

; Если определено поле EV_TIME_(TEXT|DATETIME), то в него пишется WIN-время EV_TIME

T_EV_TI=110,111

T_EV_NCI=601,602,603

T_EV_TS=201,202,203,204,211,212

[Архивы]

; Должны присутствовать поля: ID(INT(4)),DT(INT(4)|DBL),VAL,TC(INT(4))

; Если определены поля TC_(TEXT|DATETIME) и(или) DT_(TEXT|DATETIME), то в них пишется WIN-время

T_ARCH_V_H_TI=T_ARCH_V_H_TI

2. УСТАНОВКА И СОПРОВОЖДЕНИЕ

2.1. СЕРВЕР

2.1.1. Системные требования

- WINDOWS 2003 SERVER (или XP при количестве пользователей < 6);
- Должен быть установлен протокол TCP/IP;
- Для удобства работы с базой данных комплекса рекомендуется установить MICROSOFT ACCESS.

2.1.2. Установка

Дистрибутив серверной части комплекса КОТМИ-2010 поставляется в виде 3 папок со следующим содержанием:

- «**SrvInst**», где содержится инсталляция программного обеспечения серверной части.
- «**Base**», где содержится первичная база данных комплекса **DbScada.mdb**.

Процесс инсталляции состоит из следующих этапов:

- 1) Переписать папку Base на один из жестких дисков компьютера. Необходимо иметь в виду, что в процессе работы комплекса база будет увеличиваться. Если не требуется хранение архивов в SQL-базе (рекомендуется), то под базу необходимо до 300 MB. В противном случае может потребоваться значительно больше. Следует учесть, что разные SQL-базы имеют разные ограничения на максимальный размер. Например, база формата mdb не может быть более 2 GB.
- 2) Запустить программу ScdSrv.exe из папки SrvInst. В процессе инсталляции можно выбрать диск и папку, в которую будет ставиться программное обеспечение. Но в случае смены диска или папки необходимо будет скорректировать с помощью MS ACCESS содержимое поля SPRG_PATH таблицы T_SPRG базы данных, так как все серверные программы находятся в той же папке, где и программное обеспечение сервера.
- 3) Скорректировать файл инициализации серверной части Scada_new.ini, который находится в папке, куда устанавливалось программное обеспечение серверной части, в соответствии с требуемой конфигурацией комплекса.
- 4) Создать на одном из жестких дисков папку ArchOic, в которой будут находиться файлы базы данных реального времени. Средствами MS ACCESS скорректировать содержимое поля ARCH_RT_PATH таблицы T_ARCH, где указаны пути к файлам базы данных реального времени для архивов. Сами файлы базы данных реального времени будут созданы автоматически при первом запуске сервера.

2.1.3. Файлы серверной части комплекса КОТМИ-2010

После завершения инсталляции программного обеспечения серверной части в указанной при инсталляции папке будут следующие файлы:

- **ScdSrv.exe**. Сервер ОИК со следующими функциями:
 - 1) Проверка прав доступа пользователей ОИК;
 - 2) Диспетчеризация доступа к SQL-базе и базе данных реального времени;
 - 3) Оповещение пользователей о событиях;
 - 4) Циклические расчеты;

- 5) Обмен информацией с разными типами центральных приемо-передающих станций (ЦППС) по единому стандартизированному интерфейсу (прием ТИ, ТС, обмен макетам ЦБИ, ретрансляция ТИ, ТС);
 - 6) Обработка принятой информации ТИ, ТС;
 - 7) Расчет псевдо-ТИ, псевдо-ТС;
 - 8) Запись изменившейся информации в базу данных.
- **ScdMail.exe**. Прием макетов по электронной почте и их разбор.
 - **ScdMail.ini**. Файл инициализации для ScdMail.exe. Структура описана выше.
 - **Scada_new.ini**. Файл инициализации сервера ОИК (структура описывается ниже).
 - **ScdCnt.exe**. Служба контроля серверов ОИК. Устанавливается при установке сервера как сервис с именем «KOTMI-NT Control». Служит для управления сервером ОИК на данном компьютере. Выполняет запросы по останову, запуску сервера, а так же выдает различную информацию о работе сервера ОИК и компьютера.

2.1.4. Файл инициализации сервера

Файл инициализации серверной части комплекса называется **Scada_new.ini**. Он находится в папке, в которую устанавливалось программное обеспечение серверной части и состоит из набора разделов.

Каждая строка имеет имя. Любая строка может быть закомментирована, если первый символ строки - «;». Определены следующие имена строк:

Определены три раздела:

А) [Комплексы]. Описываются комплексы, работающие на данном объекте. В рамках каждого комплекса осуществляется синхронизация баз НСИ и реального времени. Несколько комплексов можно иметь, например, для целей отладки, выделив сервера администраторов в отдельный комплекс или выделив в комплекс сервера, обрабатывающие информацию АСКУЭ. В данном разделе определен только один ключ:

- 1) **Комплекс N**, где N – номер комплекса. Значением данного ключа является строка с наименованием комплекса.

Например:

[Комплексы]

Комплекс 1=ОИК энергосистемы

Комплекс 2=АСКУЭ

Комплекс 3=ОИК отладочный

Б) [Сервер N], где N – номер сервера в файле инициализации. В данном разделе определены следующие ключи:

- 1) **Комплекс**. Значением этого ключа является номер комплекса из раздела **[Комплексы]**.
- 2) **Компьютер**. Имя компьютера, на котором работает данный сервер ОИК.
- 3) **IP-адрес**. IP-адрес компьютера, на котором работает данный сервер ОИК. Если на компьютере установлены два сетевых адаптера, то необходимо написать через запятую оба IP-адреса.
- 4) **Порт**. Номер TCP-порта, через который клиенты должны общаться с данным сервером ОИК.

5) **Тип.** Сервера комплекса могут быть настроены на выполнение определенных функций. Для этого используется механизм логических серверов. Выделено четыре типа логических серверов:

- Сервер ввода-вывода (**IO**). Основной сервер ввода-вывода взаимодействует с различными типами ЦППС, получает от них телеметрическую информацию, обрабатывает ее и передает все изменения параметров, а также соответствующие события (нарушения пределов, изменение состояния коммуникационной аппаратуры и т.д.) в базу данных;
- Сервер событий (**ALARM**). Синхронизирует поток событий, поступающих в комплекс. Любое событие, возникающее на любом сервере комплекса, сначала попадает на обработку в основной сервер событий, где ему присваивается идентификатор записи события в соответствующей таблице базы этого сервера, а затем уже реплицируется на остальные сервера комплекса;
- Сервер расчетов (**CALC**). На основном сервере расчетов выполняются все циклические расчеты, задействованные в комплексе. Результаты расчетов реплицируются на остальные сервера комплекса;
- Сервер MMI (**DIALOG**). Все сервера MMI, описанные в комплексе, поддерживают запросы клиентов на оптимальное подключение и в качестве результата выдают имя и IP-адрес сервера MMI, подключение к которому оптимально, с точки зрения производительности.

Сервера комплекса могут быть настроены на выполнение нескольких функций. Например, для случая одномашинного комплекса:

Тип=**DIALOG+CALC+ALARM+IO**.

Для 4-машинного комплекса могут быть выделены 2 сервера типа:

Тип=**CALC+ALARM+IO**

и 2 сервера:

Тип=**DIALOG**.

- 6) **Синхронизация времени.** Значением этого ключа являются «ДА» и «НЕТ». Если значение «ДА», то с данного сервера, если он основной, осуществляется коррекция времени на всех остальных серверах комплекса.
- 7) **Резервный сервер.** Значением этого ключа является номер резервного сервера из файла инициализации. Если значение ключа ноль, то для данного сервера нет резерва. Все сервера комплекса могут попарно резервироваться. Рекомендуется резервировать сервера с одинаковым значением ключа **ТИП**.
- 8) **Тип SQL-базы.** Значением этого ключа являются «MDB», «ODBC». Значение «MDB» задается для базы формата MS ACCESS (тип файла mdb). Для всех остальных (например, MS SQL SERVER) задается тип «ODBC».
- 9) **Строка подключения к SQL-базе.** Для взаимодействия с SQL-сервером сервер ОИК использует механизм OLE DB. Значением этого ключа является описание соответствующего провайдера данных.
- 10) **Количество подключений к SQL-базе.** Значением этого ключа является количество необходимых соединений с SQL-базой для данного сервера ОИК. Все соединения разделяются между всеми клиентами, подключенными к серверу. Не рекомендуется делать значение этого ключа большим, так как каждое соединение требует дополнительных системных ресурсов.

При количестве одновременно работающих клиентов на сервере не более 20, значение этого ключа может быть равно 3.

- 11) **Папка базы РВ.** Значением этого ключа является путь к каталогу, в котором хранятся файлы базы реального времени. Этот ключ перекрывает значение поля ARCH_RT_PATH таблицы T_ARCH. Например:
Папка базы РВ=C:\ArchOic\
- 12) **Автостарт.** Значением этого ключа являются «ДА» и «НЕТ». Если значение «ДА», то при пуске службы KOTMI-NT Control, автоматически запустится данный сервер ОИК. Это позволяет запускать сервер ОИК в момент запуска WINDOWS до входа пользователя в систему.
- 13) **Окно.** Значением этого ключа являются «ДА» и «НЕТ». Если значение этого ключа «НЕТ», то сервер ОИК не будет представлен на панели задач.
- 14) **Пароль.** Значением этого ключа является строка с паролем запуска - останова сервера ОИК. Пароль запрашивается «Монитором серверов Scada» при пуске или останове сервера ОИК. Если ключ «Автостарт=ДА», то пароль не используется. Если сервер запускается из командной строки, то пароль также не используется.
- 15) **Порт TELNET.** Сервер ОИК является и сервером TELNET и общаться с ним можно с помощью стандартного клиента TELNET. Здесь указывается порт, с которым создается сервер TELNET.
- 16) **Максимальный размер LOG-файла.** Значением этого ключа является размер файла ScdSrv.log в байтах. В этом файле фиксируются моменты пуска – останова, смены статуса (основной – резервный), ошибки запуска, ошибки работы данного сервера ОИК.

2.1.5. Порядок запуска серверов комплекса КОТМИ-2010

- 1) Сервера запускать по очереди.
- 2) Включить питание на первом.
- 3) Если включена опция «Автозагрузка», то сервера загружаются автоматически.
- 4) После запуска сервера проверить его работоспособность.
- 5) Включить питание на втором сервере. Дождаться завершения загрузки сервера. Он должен загрузиться как «резервный».
- 6) Если время загрузки любого из серверов становится недопустимо большим (5-10 минут), причем основное время приходится на инициализацию базы реального времени, то необходимо остановить сервер и произвести дефрагментацию диска с базой реального времени.

2.1.6. Останов основного сервера с выключением питания

- 1) С помощью программы «Монитор серверов» выполнить останов основного сервера ОИК. По завершении останова пропадет иконка сервера.
- 2) Резервный сервер ОИК должен изменить свое состояние на «основной».
- 3) Проверить работоспособность сервера ОИК.

Сервер не должен находиться в режиме Х. Если находится, то это означает неправильное положение арбитров-коммутаторов. В этом случае необходимо визуально проконтролировать состояние арбитров-коммутаторов. Возможно они вручную переведены на остановленный сервер. Если арбитры-коммутаторы находятся в положении «автомат», но не перешли на основной сервер, то необходимо с помощью «Диспетчера задач» снять все экземпляры программы KtmCnsl и дождаться перезапуска этой программы. Если и после перезапуска программы состояние арбитров-коммутаторов остается ошибочным, то необходимо попробовать переключить их на данный сервер

вручную. Если и это не получается, то необходимо отменить решение о выключении питания, опять запустить сервер ОИК, перевести его в режим «основной» и выполнить полный рестарт компьютера с резервным сервером ОИК.

Проверить прием и ретрансляцию телеметрии. Это можно сделать с помощью трассировок сервера ОИК, KtmCnsl и KtmSD. **При просмотре трассировок не пользоваться клавишей «Пауза».**

В случае нормальной работы сервера ОИК можно завершить работу выключаемого сервера.

2.1.7. Останов резервного сервера с выключением питания.

- 1) Убедиться в работоспособности основного сервера (см.выше).
- 2) С помощью программы «Монитор серверов» остановить резервный сервер ОИК.
- 3) Завершить работу Windows.

2.1.8. Сжатие SQL-базы сервера ОИК

Рекомендуется периодически (раз в месяц) выполнять сжатие SQL-базы сервера ОИК. На данный момент используется база формата mdb (MS ACCESS). **Сжатие нельзя производить через сеть!** Процедура сжатия:

- 1) Переключить средства диалоговой работы (мышь, клавиатура, дисплей) на нужный сервер.
- 2) Остановить через «Монитор серверов» сервер ОИК.
- 3) Сделать копию файла базы DbScada.mdb
- 4) Открыть его. Запустится программа MS ACCESS.
- 5) В меню «Сервис» выбрать «Служебные программы» и запустить функцию «Сжать и восстановить базу данных». Процесс сжатия можно видеть внизу слева в ProgressBar.
- 6) Если будут сообщения о неисправимых ошибках, то надо попытаться восстановить базу из копии или с другого сервера и повторить процедуру сжатия.

2.1.9. Резервирование в комплексе

2.1.9.1. Режим X

Режим X определяется по иконке сервера в «Мониторе», по log-файлу сервера, по событиям в АРМ клиента, с помощью программы KtmSd и визуально по состоянию арбитров-коммутаторов.

Режим X означает некорректное состояние арбитров-коммутаторов.

Неправильное положение арбитров-коммутаторов может быть по следующим причинам:

- 1) Арбитр-коммутатор не находится в положении «автомат» и переведен вручную на сервер, не являющийся основным.
- 2) Есть незавершенное приложение KtmCnsl.
- 3) Техническая неисправность самого арбитра-коммутатора.

Первая причина устраняется путем перевода арбитра-коммутатора в положение «автомат».

Для устранения второй причины необходимо воспользоваться «Диспетчером задач» Windows. Надо открыть вкладку «Процессы» и завершить все процессы с именем KtmCnsl.

Третья причина может быть устранена только заменой неисправного арбитра-коммутатора.

2.1.9.2. Базовая схема резервирования

Данная схема встроена в сервер и работает всегда. Для всех серверов одного комплекса обеспечивается синхронизация нормативно-справочной информации и содержимого заданных архивов. Список серверов указывается в файле инициализации (смотри описание файла инициализации сервера). Все сервера попарно резервируются. Первый загруженный сервер является главным. По времени, зафиксированному в поле THIS_TIME_CURR таблицы T_THIS, определяется время простоя запускаемого сервера. Затем запрашиваются все изменения нормативно-справочной информации и архивная информация с главного сервера комплекса, произошедшие за время простоя. После этого сервер определяет свой статус основного или резервного и оповещает о своем статусе другие сервера комплекса. Алгоритм определения главного сервера комплекса следующий:

- а) Определяется сервер в комплексе статус которого основной (PRYMARY), является сервером событий (ALARM) и не является отладочным.
- б) Если поиск по первому критерию не дал результата, то определяется первый сервер, который находится в состоянии готовности к резервированию и не является отладочным.

Смена статуса происходит в случае:

- 1) Остановка основного сервера.
- 2) Сбой в работе основного сервера, приводящий к аварийному завершению его работы.
- 3) По команде оператора с помощью «Монитора серверов» (ScdMon).

После выдачи команды «Изменить статус» контроль правильности перехода выполняется визуально по иконке сервера и по его log-файлу.

Если один из серверов перешел в режим X, то это означает некорректное состояние арбитров-коммутаторов (о режиме X выше).

- а) Перевода основного сервера в отладочный режим (см. далее).
- б) При работе дополнительной схемы резервирования (см. далее).

2.1.9.3. Дополнительная схема резервирования

Для повышения надежности работы комплекса может быть задействована дополнительная схема резервирования. С помощью этой схемы можно контролировать работоспособность устройств комплекса.

В базе должны быть таблицы T_RSRV, T_RSRV_PAR. Если необходим контроль состояния АК-А, то на серверах ввода-вывода должна быть таблица T_SOST_AKA. Структура таблиц в документе «Структура базы данных». Все таблицы д.б. исключены из схемы резервирования (поле OBJ_NO_RES в таблице T_OBJ в состоянии TRUE). Это связано с тем, что:

- 1) состояния контролируемых параметров должны фиксироваться независимо на каждом из серверов пары;
- 2) в случае четырехмашинного комплекса (например, два сервера выполняют функции ЦППС, а два – роль серверов, с которыми непосредственно работают пользователи) списки контролируемых параметров будут для каждой пары свои. Так ЦППС контролируют состояния каналов связи, АК-А, ИВЧ. На другой паре может работать прием макетов и нужно контролировать состояние связи с почтовым сервером.

Сервера резервируются попарно, поэтому таблицы на серверах пары д.б. с одним набором строк. Таблицы должны настраиваться с помощью MS ACCESS без запуска серверов или с остановленным вторым сервером.

Каждая строка таблицы T_RSRV_PAR соответствует одному контролируемому параметру.

Состояние параметра (поле RSRV_PAR_STAT) должно заполняться независимо на основном и резервном сервере, поэтому заполняющие программы должны быть запущены на основном и резервном сервере.

Предлагается в программах, формирующих состояние параметра его номер (т.е. поле RSRV_PAR_ID) задавать в конфигурационных файлах программ. Если номер 0, то программа не формирует состояние параметра.

Схема резервирования и контроль состояния АК-А не будут работать, пока заполняющие программы не пропишут состояния **всех** параметров, задействованных в схеме резервирования или состояния всех АК-А.

Состояния параметров должны обновляться не реже, чем указано в поле RSRV_RENEW таблицы T_RSRV, состояния АК-А не реже 60 сек. Иначе схема резервирования и состояния АК-А опять перейдут в не рабочее состояние.

События, коды которых описываются в RSRV_EVN должны принадлежать к категории 1400 (EV_COD_EV_CAT_ID=1400).

2.1.9.4. Отладочный режим работы сервера

Отладочный режим может быть установлен с помощью «Монитора серверов» для любого сервера комплекса. Сервер, переведенный в отладочный режим становится полностью независимым от других серверов комплекса, т.е. не участвует в зеркалировании базы НСИ и базы реального времени. В то же время на него будет поступать телемеханика. То есть этот режим можно использовать при массовых изменениях НСИ, дорасчетов.

По завершении отладки можно утвердить или отменить изменения, сделанные на отладочном сервере. При утверждении изменений отладочный сервер дает последовательно команду на рестарт всем серверам комплекса. После завершения процедуры рестарта каждый сервер будет синхронизирован по базе с отладочным сервером. При отмене изменений отладочный сервер сам рестартует и по завершении этого процесса будет синхронизирован по базе с другими серверами комплекса.

2.2. МОНИТОР

Программа «Монитор» (ScdMon.exe) служит для наблюдения и управления серверами комплекса и работает совместно со службой контроля серверов ОИК (ScdCnt.exe). Установка «Монитора» включена в установку клиентской подсистемы ОИК.

Программа «Монитор» позволяет организовать наблюдение за всеми серверами комплекса, запускать и останавливать каждый из них, изменять их статус, переводить один из серверов в режим отладки, отменять и утверждать изменения, сделанные на сервере в этом режиме, смотреть оперативную информацию по их работе.

Окно программы включает две основные панели: слева – список серверов комплекса, справа – совокупная информация по выбранному серверу.

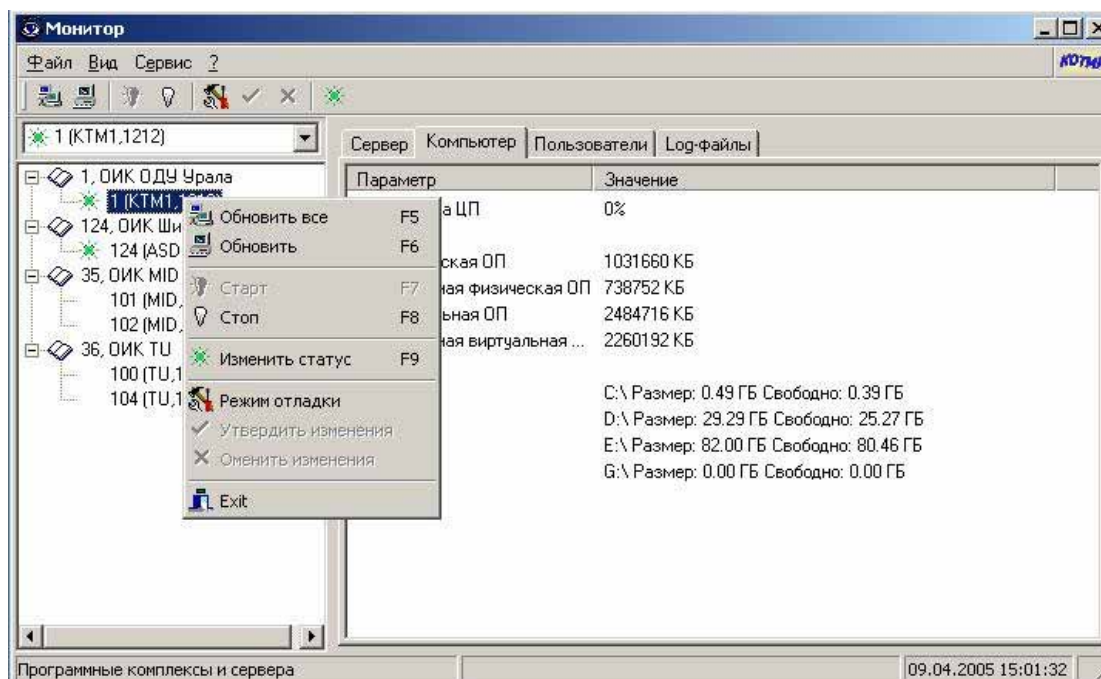


Рисунок 1 – Монитор серверов ОИК

В списке каждый сервер идентифицируется своим номером (см. «службу контроля»). В качестве дополнительной информации в скобках приводятся имя ЭВМ и порт TCP/IP, который данный сервер использует.

Совокупная информация включает в себя основные характеристики сервера, информацию по компьютеру на котором он находится, список подключенных пользователей и Log-файлы, в которых фиксируются основные события возникающие при работе сервера.

Первые две закладки, «Сервер» и «Компьютер», доступны для просмотра всегда. Остальные только если сервер запущен и имя пользователя и пароль, указанные при входе в программу, позволяют «Монитору» подключиться к данному серверу ОИК в качестве клиента.

Запуск (или останов) работы сервера, изменение статуса, перевод в отладочный режим можно осуществить с помощью соответствующих пунктов меню или функциональных кнопок. Любое из этих действий требует дополнительного ввода пароля, прописанного в файле конфигурации соответствующей службы контроля.

Для взаимосвязи со «Службой контроля» программа «Монитор» использует механизм IP-датаграмм и, в частности, возможности групповой рассылки которые он обеспечивает. В основном параметры настройки и конкретизируют нюансы обмена датаграммами.

Параметр	Значение
Групповой адрес монитора:	234.12.12.61
· адрес интерфейса рассылки	
Групповой адрес службы контроля	*
Число пересекаемых маршрутизаторов	0
Адреса прямой рассылки(,)	mid
Период обновления конфигурации(с)	3
Период запроса серверных данных(с)	3

Рисунок 2 – Параметры монитора

«Групповой адрес монитора» - групповой IP-адрес монитора при общении программ мониторов друг с другом (в данной версии не используется).

«Адрес интерфейса рассылки» - можно указать IP-адрес нужного интерфейса на машинах с несколькими сетевыми картами. По умолчанию служба IP выбирает его самостоятельно.

«Групповой адрес службы контроля» - используется монитором для передачи датаграмм службам контроля. Если ваша сеть не поддерживает механизм групповой рассылки, то рекомендуется заменить данный адрес на «*».

«Число пересекаемых маршрутизаторов» - значение данного параметра имеет смысл только если используется механизм групповой рассылки, «Групповой адрес службы контроля» определен и в сети имеется в наличии маршрутизаторы, правильно настроенные для работы с групповой рассылкой. Если «Групповой адрес службы контроля» = «*», то датаграммы маршрутизаторов проходить не будут.

«Адреса прямой рассылки» - если в сети есть маршрутизаторы и (или), по каким либо причинам, не удалось добиться взаимодействия «монитора» и «службы контроля», то список IP-адресов ЭВМ на которых расположены «службы контроля» разрешит эту проблему. Список адресов должен разделяться «,».

«Период обновления конфигурации(с)» - интервал в секундах, через который программа «монитора» опрашивает состояние «служб контроля» в сети.

«Период обновления серверных данных(с)» - если на панели монитора выбран работающий сервер и имя и пароль, указанные при входе, позволяют монитору подключиться к этому серверу, то с этим периодом монитор будет обновлять информацию о пользователях и Log-файле для данного сервера.

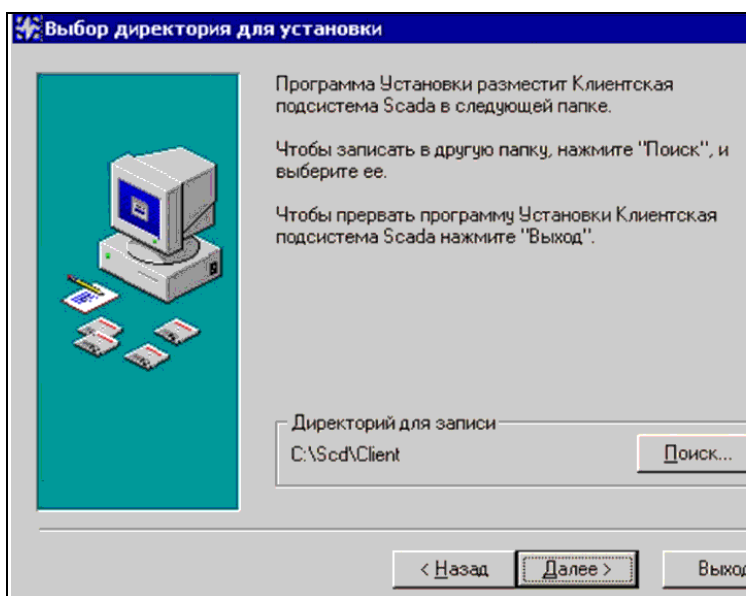
После изменения параметров в окне настройки и сохранении их по кнопке «Ok», новые значения параметров запоминаются, и монитор немедленно перенастраивается на их использование. Параметры хранятся в файле “Scada.ini” клиента и не требуют непосредственной ручной корректировки.

2.3. КЛИЕНТ

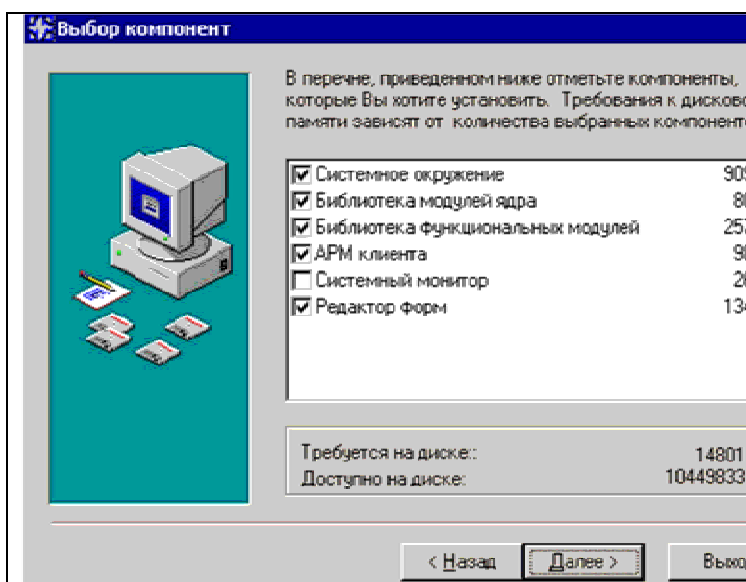
2.3.1. Установка

После установки серверной части комплекса можно приступить к подготовке рабочих мест пользователей ОИК. Весь процесс установки можно разбить на несколько шагов:


- 1) Выбрать место (диск) где будут размещаться файлы клиентской подсистемы, например диск "D".
- 2) Найти и запустить программу установки **ScdCli.EXE**. Пароль для запуска – «Scada». Рекомендуется размещать дистрибутив в разделяемой директории сети (например). [\\Scd\ScadaDst](#).
- 3) Корректно ответить на вопросы, задаваемые программой:



Здесь Вы должны указать директорий для установки. Не рекомендуется изменять наименование директория, а диск можно выбрать любой.



Выбор компонентов. Практически все компоненты, кроме «системного монитора», являются обязательными для полнофункциональной работы клиента. При первой установке убирать что-либо из стандартного списка не рекомендуется.

Определение сервера	
	Наименование сервера <input type="text" value="Сервер ОИК"/>
	Основной сервер (имя ЭВМ или IP-адрес) <input type="text" value="Srv1"/>
	Резервный сервер (имя ЭВМ или IP-адрес) <input type="text" value="Srv2"/>
	Имя пользователя на сервере <input type="text" value="Adm"/>
	<input type="checkbox"/> Запрет обновления системных файлов при запуске
<input type="button" value="Назад"/> <input type="button" value="Далее"/> <input type="button" value="Выход"/>	

Наименование сервера будет фигурировать при входе в АРМ, при выборе сервера. Имена основного и резервного (если есть) серверов – сетевые имена ЭВМ, где они расположены. Вместо имен можно указать IP адреса. Не стоит без достаточных причин запрещать режим «обновления». Это позволит программе АРМ автоматически заменять версии системных и других библиотечных файлов из таблицы T_MOD БД комплекса.

- 4) При необходимости скорректировать файл «Scada.ini», расположенный в директории, выбранном для инсталляции:

В секции [Замена строк] убрать значки комментариев. Указать расположение директория MsOffice (например, &DirMsOffice&=c:\Program Files\MsOffice).

Для успешного входа в систему, имена пользователей, указанные при инсталляции в секции [Пользователи], должны быть прописаны на сервере в таблице T_USRS.

- 5) В ярлыке на рабочем столе «Клиент Scada ARM» добавить параметр – наименование конфигурации АРМа, которая будет загружена при запуске. Отсутствие параметра, соответствует конфигурации «Default».

Например: C:\Scd\Client\ScdArm.exe **Boss**

2.3.2. Каталоги на диске

В процессе инсталляции клиента на диске размещаются следующие файлы:

№	Файл	Описание
Библиотека модулей ядра		
1	Cli\ScdSys.ocx	OLE-интерфейсы для доступа к серверу ОИК, интерпретатору, таблицам в памяти
2	Cli\ScdSys.hlp	Help-файлы библиотеки ScdSys.ocx
3	Cli\ScdSys.cnt	
4	Win\System32\OicMde.dll	Основная библиотека функций для доступа к серверу ОИК, интерпретатор языка TScript, таблицы в ОП
Библиотека модулей администрирования		
5	Cli\ScdAdm.ocx	ActiveX – модули для администрирования комплекса. Основная библиотека
6	Cli\ScdAdm.hlp	Help-файлы библиотеки ScdAdm.ocx
7	Cli\ScdAdm.cnt	
8	Cli\ScdTool.ocx	ActiveX – модули для администрирования комплекса. Дополнительная библиотека
9	Cli\ScdTool.hlp	Help-файлы библиотеки ScdTool.ocx
10	Cli\ScdTool.cnt	
Библиотека основных функциональных модулей		
11	Cli\ScdStd.ocx	Библиотека основных функциональных ActiveX – модулей клиента. Документы, схемы, события и пр.

12	Cli\ScdStd.hlp	Help-файлы библиотеки ScdStd.ocx
13	Cli\ScdStd.cnt	
АРМ клиента		
14	Cli\ScdArm.exe	Программа-оболочка АРМ
15	Cli\ScdArm.hlp	Help-файлы библиотеки АРМ
16	Cli\ScdArm.cnt	
17	Cli\ScdCpy.exe	Программа копирования новых версий файлов ядра системы с возможностью последующего перезапуска АРМ клиента
18	Cli\Scada.ini	Файл локальных настроек для клиентских программ комплекса
19	Cli\NtRus.exe	Русифицирует отображение русских шрифтов схем CorelDraw под NT
20	Cli\Regsvr32.exe	Регистратор OLE-интерфейсов (системная утилита)
Системное окружение		
21	Win\System32*.bpl, midas.dll	BPL-библиотеки среды Delphi 6.0 (Update Pack 2)
Редактор форм		
22	Cli\TxText*.*	Файлы компонента TxText
Примечание: Cli – директорий, где установлен клиент, например C:\Scd\Client; Win\System32 – системный директорий Windows, например C:\WinNT\System32.		

2.3.3. Файл локальных настроек АРМ (Scada.ini)

Файл параметров – Scada.ini, используется программой АРМ-клиента (ScdArm.exe) для получения информации о доступных серверах ОИК, списке пользователей и «синонимах», используемых при конфигурировании и работе АРМ.

Раздел «Сервера» содержит список доступных серверов, к которым может подключиться пользователь с данной ЭВМ. Строка описания сервера состоит из фиксированной части «Сервер» с порядковым номером этой строки в списке. После символа «=» идет описательная часть сервера, состоящая из элементов, разделенных запятыми. Первый элемент – наименование сервера, которое будет отображаться в АРМ-е. Остальные элементы – сетевые имена основного и резервных серверов, которые обеспечивают распределенную работу с выбранной БД. Список сетевых имен рекомендуется начинать с имени основного сервера. Вместо имени сервера допустимо указать его IP адрес. Признак «/lib», размещенный после любого элемента в списке, разрешает обновлять исполняемые модули и библиотеки системы с БД, поддерживаемой этими серверами. Если в одном из компонентов описания сервера, разделенных запятыми встречаются знаки пробел или запятая, то содержимое элемента нужно заключить в двойные кавычки.

По умолчанию, АРМ клиента подключается к серверу, используя порт 1212. Если это не так, то нужный номер порта можно указать сразу после имени сервера, например «Srv3/1232». Номер порта по умолчанию для всего комплекса можно изменить, добавив его сразу после наименования комплекса.

[Сервера]

Сервер0="Сервер ОИК/lib",Srv1,Srv2

Сервер1="Тестовый сервер",Test1/1233

Сервер2="Сервер Буденновск 1/1244/lib",oik_main,172.18.38.3/1245

Сервер3="Сервер Буденновск 2",172.18.38.3,oik_test2

Последнее вхождение=1

Элемент раздела «Последнее вхождение» показывает к какому серверу из списка было сделано последнее, успешное подключение. Данный параметр устанавливается программой автоматически.

Раздел «Пользователи» содержит параметр – «Имена». Данный параметр содержит список из 10 последних имен пользователей, успешно входивших в систему с данной ЭВМ. Имена располагаются в порядке очередности вхождения. Список формируется программой автоматически

Параметры «Оптимальное подключение» и «Восстановление конфигурации» используются для хранения состояния одноименных флажков на заставке входа в программу АРМ-клиента.

[Пользователи]

Имена="Mid,Tu,wert,Sw,Телемеханик,Ковтун,ddd,sv"

Оптимальное подключение=0

Восстановление конфигурации=1

Параметры раздела «Замена строк» используются программой АРМ-клиента при формировании рабочей конфигурации модулей. При этом все вхождения подстрок, расположенные слева от знака «=», заменяются при настройке на значения, расположенные справа.

[Замена строк]

\$DirMsOffice\$=C:\msoff97

\$DirMyFles\$=D:\DataXml

Например, если в конфигурации с некоторой кнопкой связан запуск программы Excel.exe, а расположение директория MsOffice различно на разных ЭВМ, то путем использования в конфигурации подстроки «\$DirMsOffice\$» можно добиться корректного запуска Excel на разных ЭВМ, указав для каждой действительное расположение директория MsOffice.

Помимо замен определенных в данной секции можно использовать следующие предопределенные имена директорий:

- \$DirWin\$ - директорий windows;
- \$DirSys\$ - директорий windows\system32;
- \$DirScd\$ - директорий клиента scada;
- \$DirTmp\$ - директорий временных файлов и данных клиента.
- \$DirLib\$ - аналогичен \$DirTmp\$;

К параметрам раздела «Разное» относятся параметры «Заставки», «Звукового сигнала» и «Схема диспетчерского щита».

С помощью первого можно управлять картинкой-заставкой, появляющейся на экране при входе в систему. В качестве его значения нужно указать имя альтернативного файла-заставки (*.bmp, *.wmf, *.emf, *.jpeg, *.jpg). Если указать пустое значение, то картинка-заставка появляться не будет. В имени файла допустимо использование строк замены, описанных выше.

Файл звукового сигнала предназначен для сигнализации пользователю об аварийном событии, зарегистрированном в системе. По умолчанию, если звуковой сигнал не указан, ищется файл с именем «\$DirTmp\$ \Alarm.wav» и «\$DirScd\$ \Alarm.wav». Если указанный файл существует, то он используется для звукового оповещения. Если же файл не указан или не найден - стандартный «Беер».

Если присутствует пункт "Схема диспетчерского щита=", то АРМ будет загружаться без запроса пароля и сразу выводить на экран заданную схему в полноэкранном режиме.

Поля в строке данных следующие:

- 1) Строка параметров. Та же самая, как описано для переходов и кнопок. конфигураций (см. «Графический инструментальный Модус»).
- 2) Номер сервера из раздела [Сервера], к которому будет произведено подключение.
- 3) Login – пользователя.
- 4) Пароль пользователя.
- 5, 6 - Ширина и высота окна в пикселях.

Если последние параметры не заданы, то полноэкранное окно раскрывается как обычно на полный экран монитора.

[Разное]

;Заставка=

Заставка=\$DirTmp\$\panel.bmp

Звук Alarm=\$DirTmp\$\MyAlarm.wav

Схема диспетчерского щита=""REC=332,ZOOM=58,POSX=58",0,EVA,"124",2304,1728".

Раздел «Монитор» и все его пункты используются программой ScdMon.exe для внутренних нужд и формируются ей автоматически.

2.3.4. Обновление файлов

Механизм обновления файлов предназначен для обновления файлов ядра (OicMde.dll, ScdLib.bpl, ScdSys.ocx и ScdArm.exe), функциональных модулей клиентской части системы (ScdAdm.ocx, ScdTool.ocx, ScdStd.ocx и т.д.), файлов помощи и других файлов (например, таблиц Excel), новые версии которых нужно оперативно распространить на все машины клиентов ОИК.

Все файлы требующие обновления (в том числе и системные) должны быть помещены в таблицы T_MOD БД откуда они будут автоматически тиражироваться на машины клиентов. При этом, передача файлов осуществляется с помощью внутреннего механизма связи ОИК и нет необходимости открывать клиентам дополнительные сетевые ресурсы.

Принцип обновления этих файлов следующий:

*.exe, *.ocx, *.hlp, *.cnt – размещаются в директории ОИК (\$DirScd\$);

*.dll, *.bpl – в системный директорий ОС (\$DirSys\$);

все остальные во временный директорий для текущего комплекса ОИК(\$DirTmp\$).

Первые две группы файлов обновляются, только если для комплекса в файле Scada.ini задан ключ "/lib". Остальные файлы обновляются всегда. Версии файлов определяются по времени последней модификации.

Если обновляются системные файлы (OicMde.dll, ScdSys.ocx, ScdArm.exe), то после копирования этих файлов во временный директорий, автоматически запускается вспомогательная программа ScdCpy.exe, которая переписывает и регистрирует системные файлы и, после обновления, перезапускает АРМ.

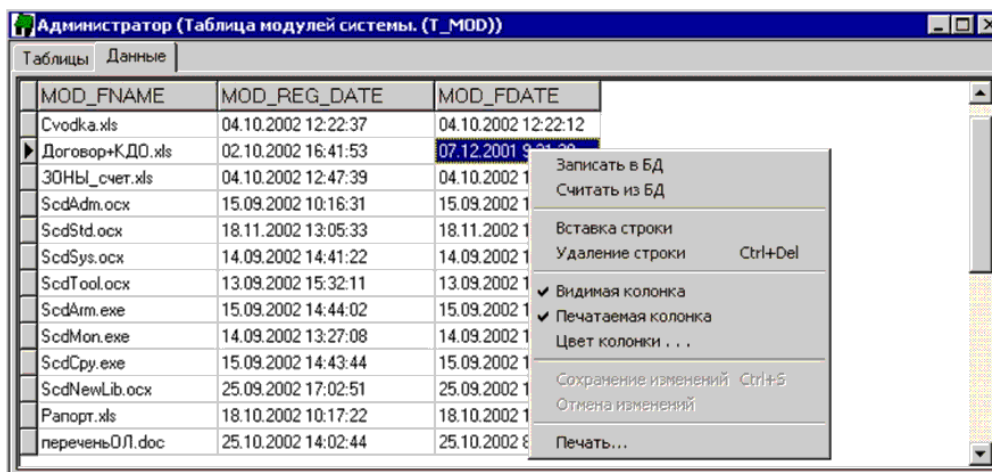


Рисунок 3 – Таблица модулей системы.

Таблица «Т_MOD» располагается обычно в разделе «НСИ ядра Scada», модуля администратора системы. С помощью пункта меню «Вставка строки», можно добавить новый файл в библиотеку, «Записать в базу» - обновить содержимое уже зарегистрированного файла, «Считать из БД» - вручную вытащить файл из БД на диск (при этом регистрация исполняемых файлов не производится)

3. НАСТРОЙКА ТАБЛИЦ НСИ

3.1. Основные понятия

Все данные НСИ хранятся в таблицах БД системы. Каждая таблица содержит подмножество функционально связанных данных, определяющих режим работы той или иной подсистемы.

Редактирование записей таблиц осуществляется в модуле администратора (**ScdAdm.ModAdm**) с помощью универсального редактора таблиц (**ScdAdm.ModTbl**).

Клиентская часть АРМ построена по модульному принципу. Каждый функциональный модуль отвечает за обработку своей части информации. Как механизм, для разработки модулей использовалась OLE-технология Microsoft. Каждый модуль представляет собой ActiveX-объект, размещенный в OCX-библиотеке, который, помимо прочих, реализует обязательный интерфейс IScadaObj.

Принцип модульности позволяет расширять функциональность стандартного редактора таблиц путем задания для нужной таблицы (объекта) индивидуального ActiveX - разработчика.

3.2. Модуль администратора (ScdAdm.ModAdm)

Данный модуль является основным инструментом администратора системы в процессе формирования и настройки таблиц НСИ.

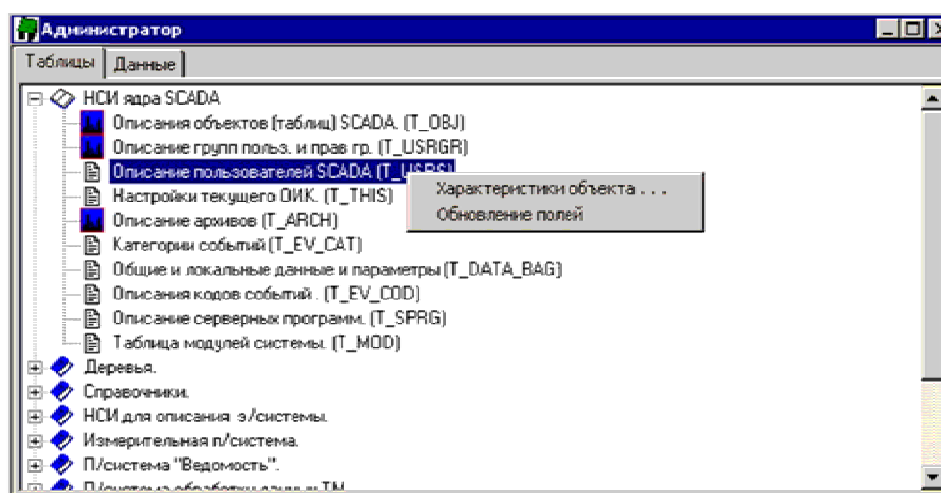


Рисунок 4 – Редактор НСИ

Модуль администратора имеет две панели-закладки: списка таблиц НСИ (Таблицы) и панели редактора (Данные), на которой будет осуществляться процесс просмотра и редактирования записей выбранной таблицы.

Список НСИ содержит все таблицы НСИ, разбитые по категориям. При работе со списком модуль администратора использует следующие системные таблицы:

- «T_OBJ» - описание таблиц НСИ;
- «T_OBJ_T» - список категорий, по которым сгруппированы таблицы;
- «T_FLD» - описание полей в таблицах;
- «T_FLD_T» - допустимые типы полей;
- «T_FLD_F» - дополнительные флаги полей;

Информация, содержащаяся в таблице полей, позволяет сопоставить с физическими полями таблиц БД некоторую дополнительную информацию: русские наименования, расширенные типы и т.п.

Первоначально, записи с описаниями полей попадают в «T_FLD» с помощью команды меню «Обновление полей». Для этого нужно выбрать в списке нужную таблицу и выполнить обновление. Список полей будет создан или обновлен, если он уже существовал. При этом, если происходит обновление, то ранее созданные параметры, по возможности, сохраняются.

Меню «Характеристики объекта...» запускает всплывающее окно, в котором можно изменить характеристики таблицы и ее полей. Окно состоит из двух закладок. На первой собраны характеристики собственно таблицы, на второй – полей.

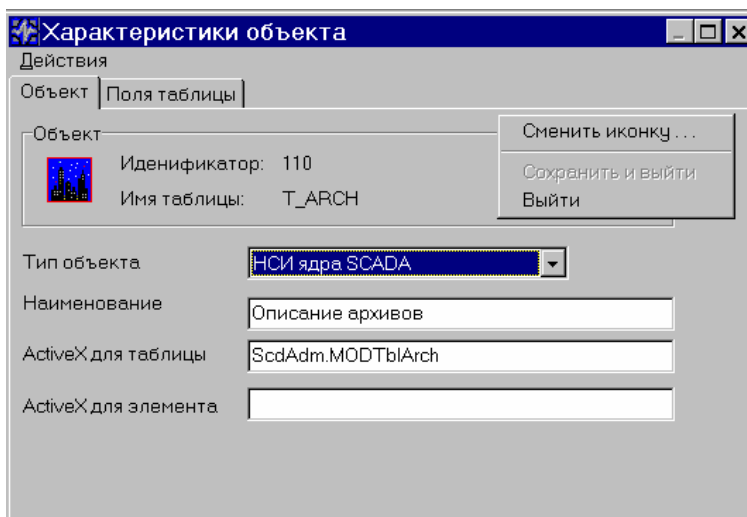


Рисунок 5 – Характеристики таблицы

Рассмотрим поля первой закладки:

- Тип объекта. Фактически это группа, где расположена таблица в модуле администратора. Не стоит менять ее без необходимости.
- Наименование. Русское, пояснительное наименование таблицы.
- ActiveX для таблицы. ActiveX – объект, который будет отвечать за просмотр и редактирование записей таблицы. Он будет появляться на панели «Данные» окна администратора вместо стандартного окна редактора таблиц.
- ActiveX для элемента. ActiveX – объект, который будет отвечать за просмотр и редактирование отдельной записи таблицы. В данной версии эта информация не используется.
- «Сменить иконку...». Позволяет выбрать для таблицы иконку, которая будет отображаться в различных списках при просмотре.

Вторая закладка окна «характеристик объекта» содержит список полей данной таблицы, выбранных из «T_FLD». В каждом поле можно изменить значения «описания» (русского наименования) и, если нужно, «расширенного типа».

Расширенный тип представляет собой вариации на тему, как система может дополнительно интерпретировать стандартный тип БД.

Возможны следующие соответствия:

№	Тип БД	Расширенный тип	Параметры
1	Bool	Bool	Значение по умолчанию
2	SmallInt	SmallInt	По умолчанию, минимальное и максимально возможные значения
3	Integer	Integer	По умолчанию, минимальное и максимально возможные значения
		Время (Unix, sec)	По умолчанию, текущее или заданное
		Счетчик(AutoInc)	Нет
		Ссылка на таблицу. Идентификатор.	Таблица, на идентификаторы записей которой указывает ссылка. Поле, которое будет отображаться. Фильтр, в виде «WHERE (???)» выражения SQL, где «???» - текст фильтра
		Поле флагов. Битовые флаги.	По умолчанию, минимальное и максимально возможные значения
4	Float	Float	По умолчанию, минимальное и максимально возможные значения
5	Double	Double	По умолчанию, минимальное и максимально возможные значения
6	Текст	Текст	По умолчанию
7	Blob	Blob	Нет

Признак «Ссылка на группу» носит частный характер и используется в таблицах «Т_Тl» и «Т_ТS» для группировки при просмотре дерева энергообъектов в модуле ретроспективы.

!!! Если указан расширенный тип –“Ссылка на таблицу”, для обеспечения корректной работы программ корректировки НСИ необходимо, чтобы имя поля соответствовало следующему шаблону:

<префикс описываемой таблицы>_<префикс таблицы, на которую производится ссылка >_”ID”

Пример: Поле таблицы Т_ENOBJ с именем “ENOBJ_ENOBJ_T_ID” – ссылка на таблицу “Т_ENOBJ_Т”

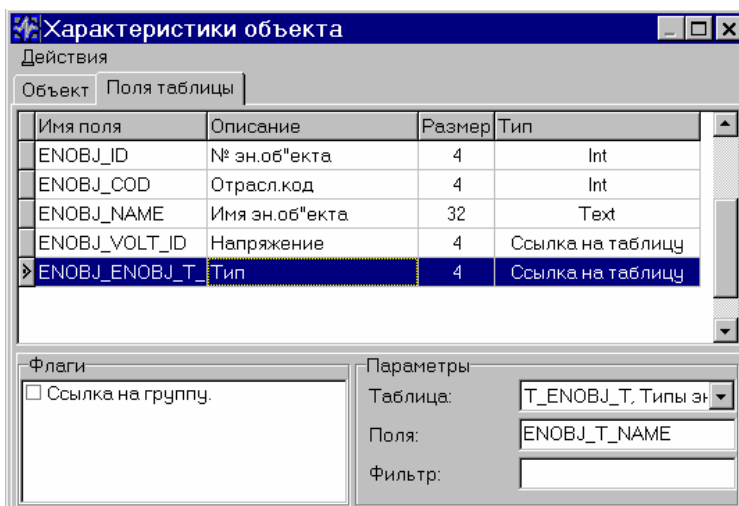


Рисунок 6 – Характеристики полей объекта

Все настройки полей сохраняются в соответствующих записях таблицы «Т_FLD» и используются функциональными модулями комплекса при отображении данных.

Основным потребителем информации о полях является универсальный редактор таблиц НСИ (ScdAdm.modTbl) и модули, наследующие его функциональность.

3.3. Универсальный редактор таблиц НСИ (ScdAdm.ModTbl)

Одной из основных функций модуля администратора является редактирование таблиц БД системы. По умолчанию, для этой цели используется универсальный редактор НСИ - ScdAdm.ModTbl.

Редактируемые данные представляются на экране в традиционной табличной форме. Строка таблицы соответствует записи из таблицы БД, столбцы – ее полям. Основные характеристики столбцов берутся из списка полей сформированного в редакторе объектов администратора «Характеристики объекта...». Поэтому, для таблиц, у которых данный список еще не сформирован, данные отображаться не будут. Выполните для такой таблицы команду «Обновление полей», если нужно отредактируйте имена и типы, затем можете приступить к корректировке данных.

Первоначально, порядок столбцов в таблице соответствует порядку полей в БД. Для удобства просмотра и редактирования данных в таблице можно настроить порядок расположения колонок, цвет, видимость в списке на экране и при печати. Порядок колонок задается с помощью перетаскивания их мышкой за заголовки в верхней строке. Для задания признаков видимости, цвета и печати нужно явным образом кликнуть мышкой или перевести курсор в одну из ячеек нужной колонки и, вызвав по правой клавише всплывающее меню, выбрать требуемую опцию.

Для того, чтобы колонки показывались в цвете или отображалось их желаемое подмножество, дополнительно нужно в главном меню программы «Опции» установить признак «Выделение колонок цветом» и выбрать один из режимов просмотра: «Только видимые колонки», «Колонки для печати» или «Все колонки».

Совокупность настроек и признаков запоминается и восстанавливается для каждой таблицы индивидуально.

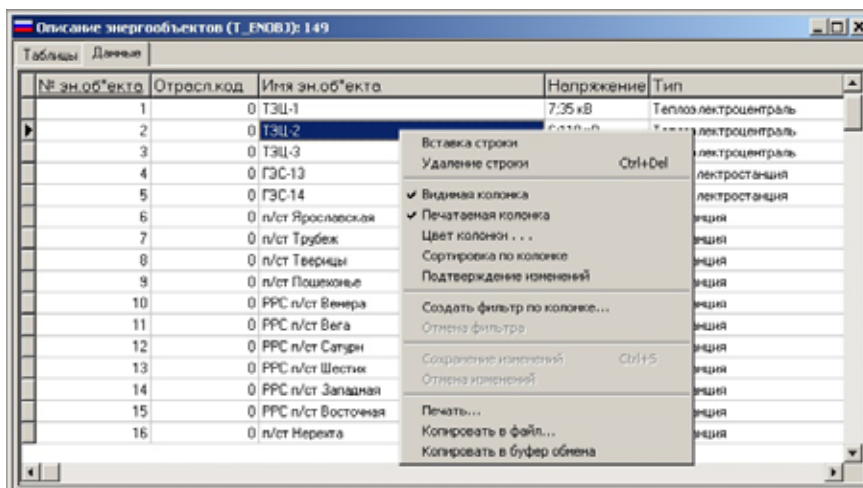


Рисунок 7 – Универсальный редактор таблиц

Для двух типов таблиц (объектов) невозможно сформировать список полей, а, следовательно, получить доступ к данным с помощью универсального редактора. Это «События» и «Архивы». Однако, если Вы разработаете свой редактор (модуль ActiveX) и зарегистрируете его в характеристиках объекта, то специальное редактирование списков событий и архивных значений может производиться с его помощью.

Если Вам позволяют права доступа к выбранной таблице, то с помощью команд меню «Вставка строки» и «Удаление строки» вы можете манипулировать ее записями. Будьте осторожны, допускается множественное выделение и одновременное удаление нескольких записей из редактируемой таблицы. Удалить запись можно также комбинацией клавиш «Ctrl-Del».

Отрасл.код	Имя зн.об"екта	Напряжение	Тип
0	Грачевская	6,110 кВ	Подстанц
0	Заветная	6,110 кВ	Подстанц
0	Промкомплекс	6,110 кВ	Подстанц
0	Ново-Невинном	6,110 кВ	Подстанц
0	Северная	6,110 кВ	Подстанц
0	Благодатное	7,35 кВ	Подстанц
0	ОАО "ПЭС"	5,220 кВ	Предприя
322670	(322670)АО"Став		АО "Энерг
0	III Подъем	6,110 кВ	Подстанц
0	Западная	6,110 кВ	Подстанц
*	0	6,110 кВ	Подстанц
		3,500 кВ	
		4,330 кВ	
		5,220 кВ	
		6,110 кВ	
		7,35 кВ	
		8,10 кВ	
		9,6 кВ	

Рисунок 8 – Добавление записи и редактирование полей

Новая запись, или запись, поля которой были изменены, выделяется слева символом «*». Можно утвердить изменения с помощью «Сохранение изменений» («Ctrl-S») или отказаться, выбрав «Отмена изменений» («Esc»).

С помощью команды «Печать...» можно вывести информацию таблицы на принтер. При этом на печать будут выводиться только колонки с признаком «Печатаемая колонка» в порядке их расположения на экране.

К дополнительным возможностям управления таблицей можно отнести следующие:

- 1) **Сортировка по столбцам.** Выполняется при помощи Рорир-меню или двойным щелчком на заголовке столбца. Заголовок упорядоченной колонки выделяется подчеркиванием. Двойной щелчок по ячейке (0,0) возвращает порядок по умолчанию – по идентификатору.
- 2) **Копирование данных таблицы в файл или буфер обмена Windows.** Копировать можно выделенные строки таблицы или таблицу целиком. Копируются все видимые столбцы. Информация представляется в 2-х текстовых форматах: строки с ячейками разделенными табуляциями и строки с ячейками разделенными запятыми (CSV). Операция копирования выполняется с помощью Рорир-меню.
- 3) **Свойство «Подтверждения изменений».** Если данный флаг установлен, то, перед сохранением изменений, будет запрошено подтверждение на операцию. Данная опция доступна по правой клавише в выпадающем меню и, как другие опции, сохраняется для каждой таблицы отдельно.
- 4) **Фильтр по столбцу.** При запросе нужно перечислить через пробел подстроки, которые могут присутствовать в текстовых значениях поля для выбранного столбца. Задание и отмена фильтра выполняются с помощью команд Рорир-меню.
Если на таблицу наложен фильтр, то об этом теперь сигнализирует маркер в левой верхней ячейке таблицы. В заголовке окна дополнительно выводится количество строк в таблице и, если есть фильтр, кол. строк в фильтре.
- 5) **Фильтр по комбинации битов в текущих значениях архива.** Для таблиц НСИ, по которым ведутся архивы, для текущих значений можно делать оперативные выборки по флагам. Данный фильтр активизируется с помощью команд Рорир-меню. В появившемся окне выбирается нужный архив и комбинация интересующих флагов. При сравнение по флагам используется один из двух критериев - наличие всех выбранных флагов или хотя бы одного из них. Такая фильтрация по значениям осуществляется разово, на момент выполнения запроса.

6) *Функции дополнительного контроля значений в основных таблицах телемеханики:*

- контроль пересечения адресов ТИ (выводятся записи с совпадающими приемными адресами);
- контроль пересечения адресов ТС (выводятся записи с совпадающими приемными адресами);
- контроль пересечения адресов ТИ-ТС (выводятся записи с совпадающими приемными адресами с учетом совпадений из обеих таблиц), контроль пересечения адресов ретранслируемых ТИ (выводятся записи с совпадающими адресами на передачу);
- контроль пересечения адресов ретранслируемых ТС (выводятся записи с совпадающими адресами на передачу);
- контроль пересечения адресов ретранслируемых ТИ-ТС (выводятся записи с совпадающими адресами на передачу с учетом совпадений из обеих таблиц). Данные виды контроля объединены в пункте меню «Дополнительный контроль» и автоматически становятся доступными для соответствующих таблиц НСИ.

3.4. Архивы (ScdAdm.ModTblArch)

Таблица архивов (T_ARCH) содержит перечень архивов системы.

В данной версии не реализован режим автоматического создания и удаления архивов, поэтому запрещено добавление и удаление записей непосредственно из таблицы. Редактирование полей-характеристик архивов осуществляется стандартным образом.

Команда меню «Настройка флагов архива...» запускает редактор, который позволяет настроить характеристики битовых флагов, используемых данным архивом.

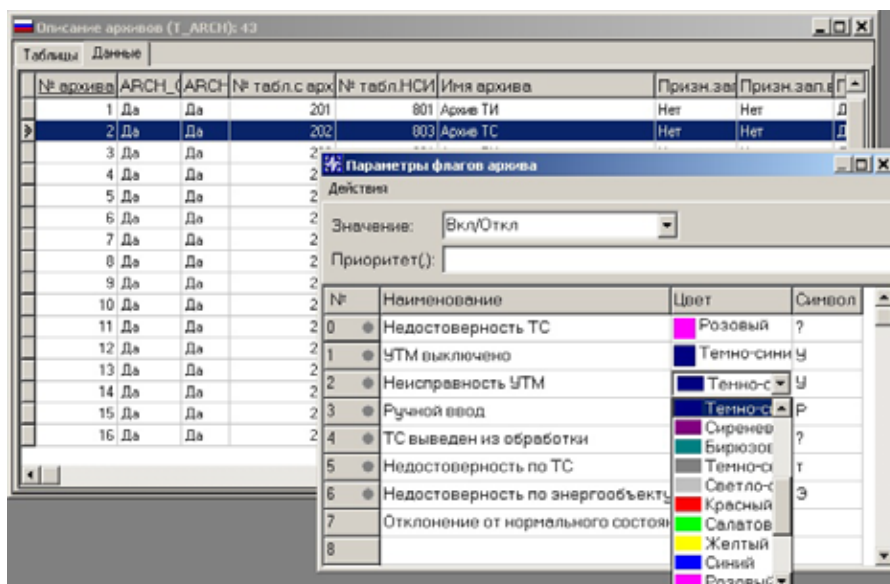


Рисунок 9 – Редактор параметров флагов архива

Для каждого из 32 флагов можно задать 3 параметра:

- 1) наименование – для отображения в других модулях и формах;
- 2) цвет – значение будет выделяться этим цветом (например, в модулях документов и ретроспективы);
- 3) символ – помимо выделения цветом, к значению будет добавлен указанный символ.

Чтобы значения нужных флагов учитывались в отображающих модулях, помимо задания параметров, перечисленных выше, необходимо задействовать эти флаги с помощью команды меню «Использование флага». Значимые флаги отмечаются в таблице в левой колонке темным кружочком. Повторное выполнение команды убирает отметку с флага. Аналогичного результата можно добиться с помощью двойного щелчка мышкой по месту расположения кружочка.

Важно также правильно указать вид отображаемых значений.

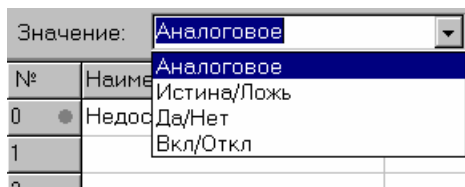


Рисунок 10 – Вид отображаемых значений

Для архивов телеизмерений и других числовых архивов нужно выбрать «Аналоговое» значение. Для архивов телесигналов укажите один из оставшихся вариантов.

В модулях, отображающих архивные данные, в поле флагов для индикации используется первый взведенный флаг из допустимого подмножества. Просмотр флагов начинается с наименьшего порядкового номера (например, 0).

При необходимости, можно изменить порядок анализа флагов в модулях отображения. Для этого перечислите интересующие Вас номера в требуемом порядке через запятую в поле «приоритет».

Перенос настроек между архивами возможен с помощью команд всплывающего меню таблицы «Копировать параметры архива» и «Записать параметры архива».

3.5. Дорасчеты (ScdAdm.ModTblCalc)

Все формулы (функции) дорасчетов размещаются в таблице T_CALC. Каждая функция присутствует в таблице в двух вариантах: исполняемом и отлаживаемом. Первый, основной, используется сервером системы в вычислениях, второй вариант – подготавливается и тестируется системным администратором на машине клиента, перед окончательным утверждением его в качестве основного.

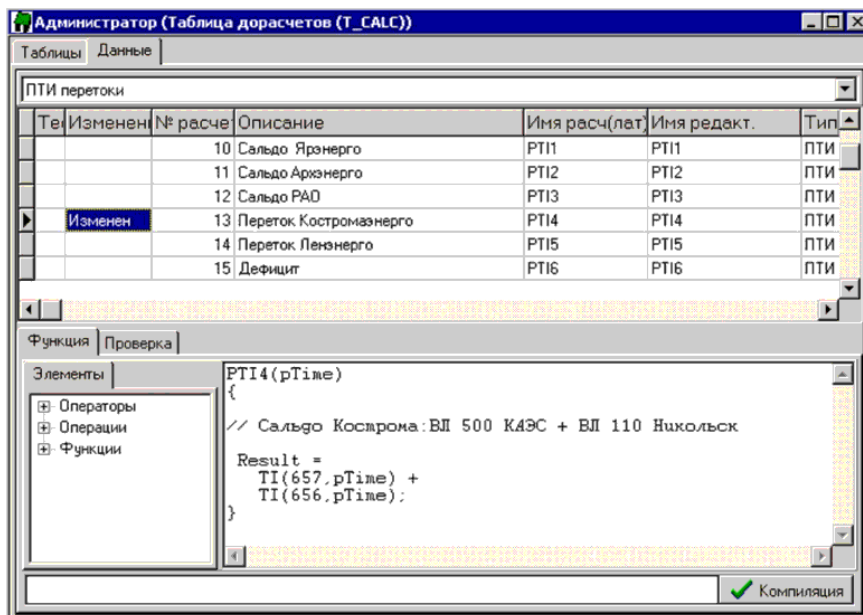


Рисунок 11 – Модуль подготовка дорасчетов

Список дорасчетов можно просматривать целиком или по группам. Выбор группы осуществляется с помощью элемента выпадающего списка в верхней части формы. Сами группы описываются в таблице-справочнике T_CALC_T.

Каждая строка в таблице описывает отдельную формулу (функцию). Поля показывают соответственно:

- «Тест» – Это «виртуальное» поле Оно отсутствует в БД. В нем показывается результат выполнения команд «Компилировать» и «Компилировать все» («!»-ошибка компиляции, «?»- неизвестная ссылка);
- «Изменения» – показывает, что отлаживаемый вариант функции отличается от серверного («Добавлен», «Изменен» или «Удален»). Если есть изменения, то их можно сделать доступными серверу с помощью команд «Утвердить формулу» или «Утвердить все» Можно также отменить сделанные изменения командой «Восстановить исходный текст»;
- «№ расчета» – уникальный идентификатор функции в таблице;
- «Описание» - пояснительное наименование;
- «Имя» - латинское имя функции. Вариант используемый сервером;
- «Имя редактируемое» - латинское имя функции. Отлаживаемый вариант;
- «Кол.парам.» – Количество параметров функции. Вычисляется автоматически. Отлаживаемый вариант;
- «Группа» –Группа дорасчетов;

В нижней части формы, на закладке «функция» расположено еще одно, главное поле. Оно отображает и позволяет редактировать текст функции (отлаживаемый вариант). Язык TScript, используемый при ее написании, подробно описан в приложении.

Имена функций задаются произвольно с учетом одного соглашения. Имена, состоящие из 2-х частей, префикса архива (таблица T_ARCH, поле ARCH_CALC_PREF) и, следующего сразу за ним, номера (идентификатора НСИ) - считаются сервером функциями, с помощью которых можно получить соответствующие архивные значения.

При написании функций могут использоваться ссылки на другие функции (помимо предопределенных в языке), отсутствующие в таблице T_CALC. Эти функции являются «встроенными» для сервера, который самостоятельно обеспечивает реализацию их вызовов. Для наглядности, имена всех встроенных функций начинаются со знака «_» (подчеркивание). Список «серверных» функций приведен в приложении D:

Слева от окна редактора функций расположена панель «дерева элементов». В нем сгруппированы все элементы языка TScript. Основное назначение дерева – памятка. Однако, встав на элемент, можно также перенести его в текст функции командой «Вставка оператора».

Клавиша «Компиляция», в правом нижнем углу, позволяет откомпилировать функцию, проверить правильность ее синтаксиса. При наличии ошибки, сообщение появляется в панели слева от клавиши. Место в тексте, где встретилась ошибка, выделяется маркером.

«Компиляция» не проверяет наличие внешних ссылок, встречающихся в теле функции. Это делает только команда меню «Компилировать». При этом функции, в которых есть ссылки на имена, отсутствующие в таблице T_CALC, отмечаются в таблице знаком «?» (неизвестная ссылка).

Ссылка	Использующие ссылку функции
_ReadDataDiv	RASPLH61, RASPLH62, RASPLH63, RASPLH64, RA...
CV	RASPLH53, RASPLH54, RASPLH55, RASPLH56, RA...
PLH	RASPLH103, RASPLH111, RASPLH12, RASPLH13, R...
RASCV387	RASPLH4
RASPLH1	RASPLH51
RASPLH102	RASPLH52
RASPLH103	RASPLH44
RASPLH107	RASPLH50

Рисунок 12 – Таблица перекрестных ссылок

Рядом с закладкой «функция» расположена закладка «проверка». Она используется для комплексной проверки ссылочной целостности дорасчетов и строится как результат работы команды меню «Компилировать все».

В первой колонке собраны имена всех функций, которые вызываются из других. Во второй – список функций, в которых эта ссылка присутствует.

Новые или измененные дорасчеты, начинают использоваться сервером сразу после их утверждения командами «Утвердить» или «Утвердить все». Поэтому к утверждению следует относиться очень осторожно и тщательно проверять отладочные варианты функций.

3.6. Древовидные структуры (ScdAdm.ModTree)

Древовидные структуры – это специального вида таблицы, позволяющие группировать записи различных таблиц (объектов) НСИ в иерархические (древовидные) списки. Таких таблиц пока немного, всего две:

- 1) T_TREE_DOC – дерево документов. Используется при отображении структуры документов в модулях работы с документами;
- 2) T_TREE_ENOBJ – дерево энергообъектов из таблицы T_ENOBJ.

Первую таблицу не имеет смысла редактировать с помощью универсального редактора «деревьев», т.к. ее специальное редактирование обеспечивается средствами модуля «Документов» (ScdStd.ModDocE), где она используется.

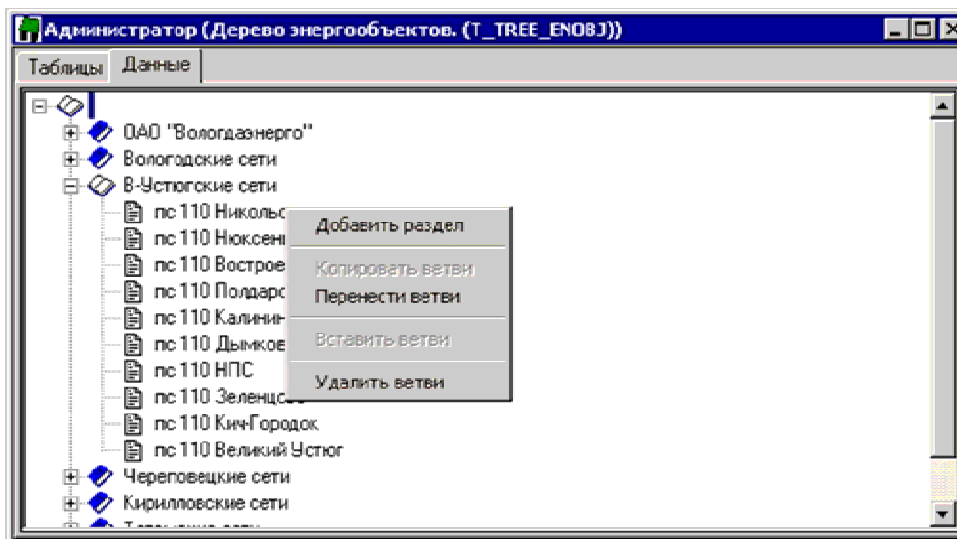


Рисунок13 – Универсальный редактор «деревьев»

Однако, для энергообъектов (T_ENOBJ), это единственный способ создания осмысленной, удобной группировки. «Дерево энергообъектов» используется модулем отображения событий на этапе настройки зоны ответственности пользователя.

Команды конструирования дерева просты и, в целом, напоминают команды, применяемые при построении дерева объектов НСИ в модуле администратора.

С помощью команды «Добавить раздел» создается структура папок. Команды «Копировать ветви», «Перенести ветви» и «Вставить ветви» позволяют наполнить папки объектами из основного списка. Удаление производится командой «Удалить ветви». Один и тот же объект может присутствовать в дереве несколько раз. Если запустить два модуля администратора одновременно то, выветив «Список» энергообъектов на одном и «Дерево» на другом, можно упростить процесс наполнения папок, перетаскивая объекты мышкой

3.7. Команды щита (ScdAdm.ModTblCmd)

Функция данного модуля специфична. Это редактор двоичных команд, которые система может подавать на диспетчерский щит.

Весь список команд расположен в таблице T_CMD. Команды могут показываться по группам или всем списком. Группы должны быть определены в таблице T_CMD_T.

Для каждой записи таблицы команд с помощью специализированного редактора можно сформировать строку двоичных кодов заданной длины.

Длина команды (0..512) указывается во всплывающем окне, открывающемся командой меню «Длина команды». Того же результата можно добиться, щелкнув мышкой по левой верхней ячейке таблицы редактора.

Редактирование самого кода команды может производиться как с помощью шестнадцатиричных значений (в левой части редактора), так и кодов ASCII (в правой).

Команда меню «Выполнить» - отправляет сформированную строку на щит:

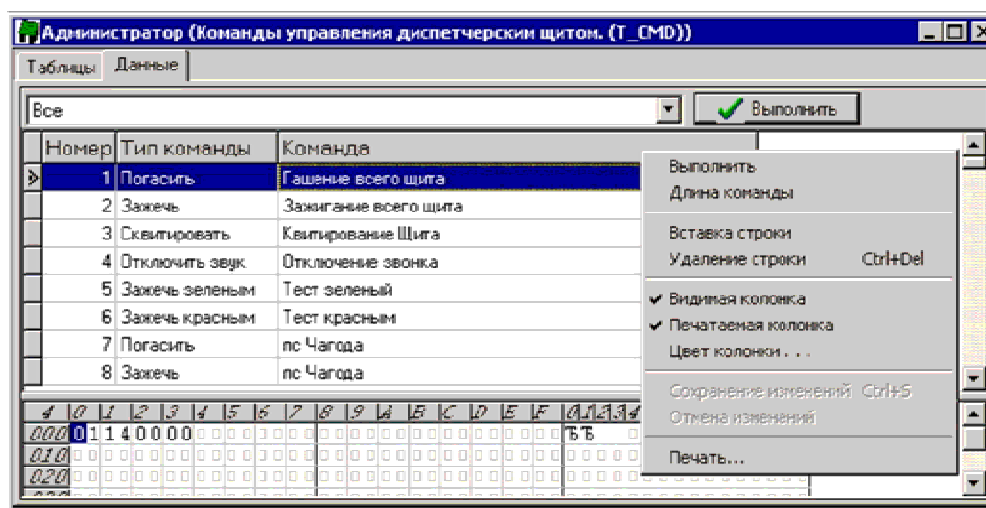



Рисунок 14 – Редактор команд управления щитом

4. ПОДГОТОВКА ФОРМ И ДОКУМЕНТОВ

4.1. Список форм и дерево документов

Основной единицей отображения информации является форма. Формы представляют собой текстовые документы, разрабатываются администратором системы и физически располагаются в таблице T_FRM. Для удобства использования формы группируются в древовидную структуру. Одна и та же форма может присутствовать одновременно в нескольких ветвях дерева

Каждая форма существует в системе в двух вариантах: варианте, используемом пользователями ОИК, и «отлаживаемом» варианте, который подготавливается администратором на стадии разработки или изменения. После завершения отладки, второй вариант формы «утверждается» с помощью команды «Утвердить изменения» и немедленно становится доступным всем конечным пользователям системы. Формы, находящиеся на стадии редактирования, выделяются слева пиктограммой . С помощью команды «Отменить изменения» можно восстановить исходный текст формы.

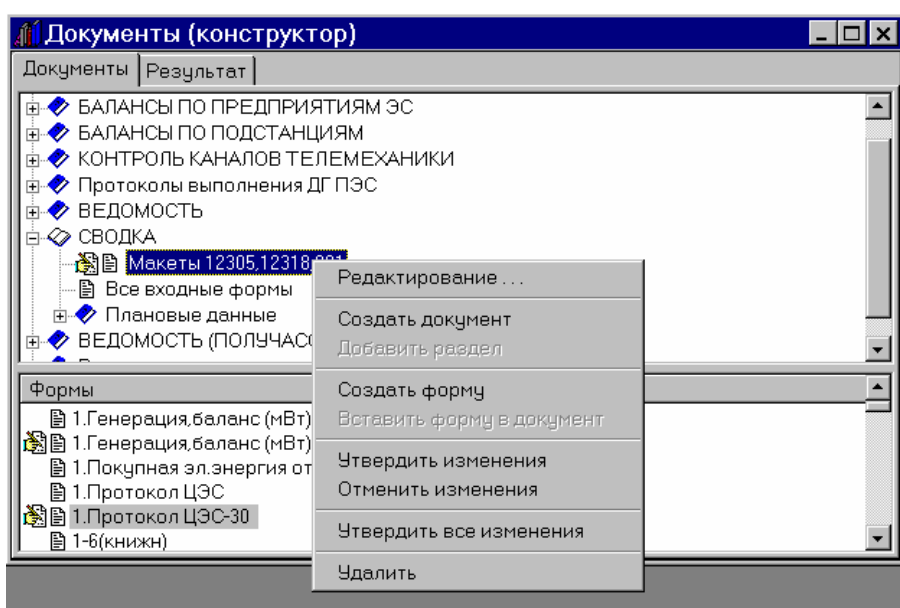


Рисунок 15 – Конструктор документов

Модуль конструктора документов имеет две панели. В нижней панели располагаются в виде списка все формы системы. Они могут просматриваться как в табличной форме, так и списком. Переключение режимов просмотра осуществляется в меню «Опции/Имена списком».

Команда меню «Создать форму» запрашивает имя, создает новую форму с указанным именем и помещает ее в нижний список. Имя формы в списке можно всегда скорректировать, щелкнув мышкой по уже выделенному заголовку.

Верхняя панель содержит «дерево» форм. Каждый узел дерева может содержать в себе другие узлы и произвольное количество форм из нижнего списка. Команда «Создать документ» создает узел первого уровня. Команда «Добавить раздел» создает подраздел для выделенного в данный момент узла дерева.

Разделы и подразделы документов могут наполняться формами с помощью команды «Добавить форму в документ» или с помощью непосредственного перетаскивания мышкой между панелями. При нажатой клавише «Ctrl» перетаскивание на узел осуществляется внутрь узла-приемника формы, по умолчанию форма будет расположена на том же уровне иерархии. Перетаскивание мышкой можно с успехом применять и для манипуляции порядком и вложенностью разделов и форм внутри дерева документов.

Команда «Удалить» удаляет выделенный в настоящее время элемент на одной из панелей модуля. Узел дерева документов можно удалить, только если он пуст.

Удаление формы в дереве документов не приводит к ее физическому удалению из таблицы форм. Форма, в этом случае, удаляется только из дерева.

К удалению формы из списка форм следует относиться с осторожностью. В этом случае форма удаляется из T_FRM и восстановлению не подлежит. Все вхождения такой формы в дерево документов также будут удалены.

4.2. Редактор форм

Если на одной из панелей конструктора документов отмечена форма, то команда «Редактирование...» вызовет для нее редактор, который позволит внести изменения в «отлаживаемый» вариант.

Редактор форм представляет собой универсальный текстовый процессор с довольно широкими возможностями. Он позволяет использовать различные шрифты, выбирать размер и цвет символов, управлять табуляциями, форматировать абзацы, работать с текстовыми таблицами и т.п.

Отличительной особенностью данного редактора, делающей его удобным для отображения большого количества данных телеметрии и архивных значений, является возможность работы с «полями». Поле - это выделенный текстовый фрагмент с набором дополнительных характеристик, позволяющий работать с ним как с независимым объектом отображения данных.

Процесс разработки формы упрощенно можно представить состоящим из двух этапов: наброска общей схемы документа и наполнения ее полями, значения которых будут запрашиваться с сервера и отображаться на форме.

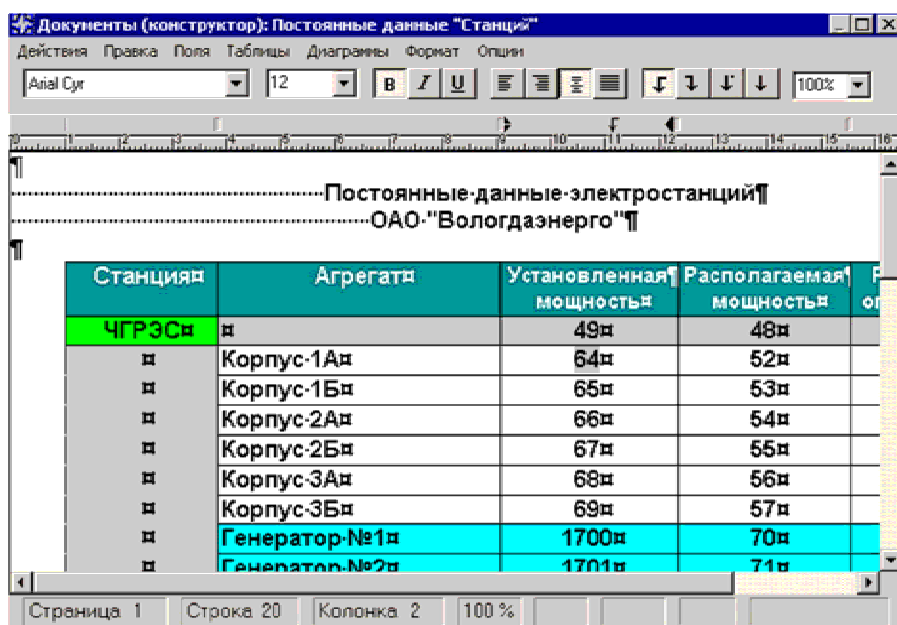


Рисунок 16 – Редактор форм

Рассмотрим процесс разработки более подробно:

- **Выбор формата страницы.** Существуют четыре predetermined формата: А4-книжный, А4-альбомный, А3-книжный и А3-альбомный. Нужный формат задается с помощью соответствующих пунктов меню «Формат».
- **Планирование и набросок общего вида формы.** Расставляем заголовки, таблицы, пояснения и прочую статическую текстовую информацию. При форматировании удобно использовать элементы панели редактора (выбор шрифта, его размера и характеристик, способов выравнивания параграфа и табуляций). Дополнительные, расширенные возможности форматирования можно найти в пунктах меню «Формат» («Параграф...», «Шрифт...», «Цвет текста...» и «Цвет фона...»).
- **Использование текстовых таблиц.** Все команды для работы с таблицами собраны в меню «Таблицы». Команда «Вставить таблицу...» приводит к появлению всплывающего окна, где нужно указать требуемое количество строк и столбцов в новой таблице. Таблица вставляется в текущее положение курсора. В существующей таблице можно добавить или удалить столбец (по месту курсора) или строки таблицы. Добавление новых строк в таблицу возможно с помощью стандартных операций «копирования» в буфер обмена и «вставки» строк в конец таблицы. С помощью команды «Обрамление...» можно настроить вид линий таблицы, заливку ячеек и прочее.
- **Добавление полей.** Поля – динамическая составляющая формы. На данный момент существуют два типа полей: архивов и времени. Первые отображают архивные значения с учетом флагов и настроек выбранного архива. Вторые - время, за которое отображаются данные. Поля добавляются с помощью команд меню «Поля». При добавлении нужно настроить поле, задав нужные для отображения параметры.

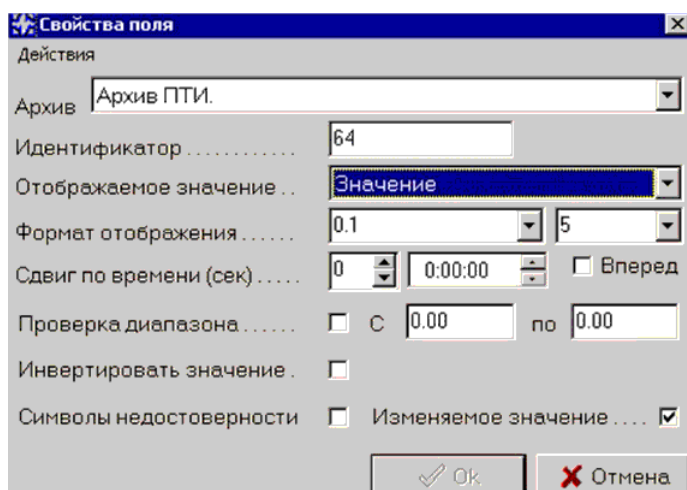


Рисунок 17 – Характеристики поля архива

Для поля архивов это архив и номер НСИ, значение которого будет отображаться на форме. Если выбран флажок «Изменяемое значение», то значение может быть скорректировано пользователем с последующим сохранением его в архиве (ручной ввод). При ручном вводе может быть выбран режим «Проверка диапазона», осуществляющий контроль диапазона вводимых значений. «Формат отображения» вместе с максимальным количеством отображаемых символов определяют длину и вид текстового значения.

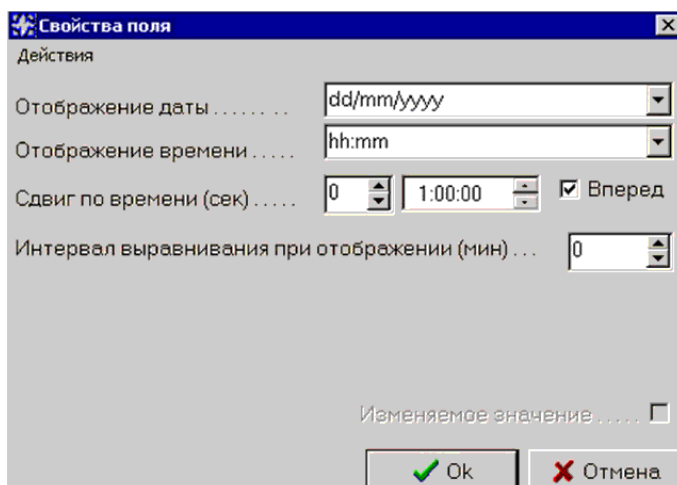


Рисунок 18 – Характеристики поля времени

С помощью поля времени можно отобразить время запроса, за которое отображаются данные формы.

Для обоих типов полей важную роль играет характеристика «Сдвиг по времени». Это сдвиг (день, час, минута, сек) назад (или вперед) по отношению к времени запроса данных для формы. Таким образом, можно считывать значения архивов в нужном временном диапазоне, например, величины в почасовой ведомости.

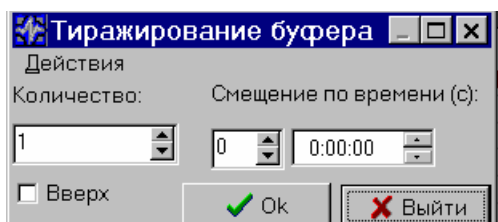


Рисунок 19 – Размножение буфера со смещением по времени

При отображении большого количества полей, использующих сдвиг по времени, удобно применять команду «Правка/Размножить буфер...». Задав направление, количество экземпляров и временной сдвиг можно быстро получить большой массив архивных полей, значения которых будут считаны с периодом, указанным в смещении. Наиболее полезна данная команда при размножении строк текстовых таблиц.

Диаграммы. Помимо таблиц в текст формы с помощью команды «Диаграмма/Добавить диаграмму» можно вставить график. Редактор параметров графика вызывается командой «Свойства диаграммы» и полностью аналогичен редактору диаграмм в модуле ретроспективы (ScdStd.ModRts).

В меню «Опции» можно включить или отказаться от высвечивания в тексте специальных символов, выбрать удобный масштаб отображения формы в редакторе, задать цвет общего фона.

Цвет фона и масштаб отображения запоминаются и применяются при просмотре для каждого документа индивидуально.

При выходе из редактора не забудьте выбрать требование сохранения измененных Вами данных.

5. ГРАФИЧЕСКИЙ ИНСТРУМЕНТАРИЙ «МОДУС»

5.1. Инструментарий

В ранних версиях «КОТМИ-2010» нами применялся упрощенный вариант отображения. На нарисованную «внешними» средствами подложку (метафайл) с помощью программы «Ризограф» наносились активные зоны (ТИ, ТС, ТУ), которые при просмотре оживлялись реальными данными из ОИК.

Другой вариант подготовки схем, с использованием графического редактора «GRIM», не получил широкого распространения, т.к. сам редактор в энергетике не прижился.

С развитием комплекса требования к представлению информации на схемах возрастали, остро стала чувствоваться необходимость в новом, полноценном, самостоятельном механизме подготовки и отображения.

Был выработан ряд требований к приемлемому инструментарию. Это, в первую очередь:

- простота и удобство подготовки «энергетических» схем (причем подготовки преимущественно технологами, а не специалистами в графическом проектировании или программистами);
- требование отображения схем в среде АРМ-клиента и их тесной интеграции с прочими функциональными модулями системы;
- наличие объектной модели и не перегруженного программного интерфейса для эффективного взаимодействия со схемами на этапах оживления и использования;

При выборе рассматривались различные варианты, в том числе и вариант самостоятельной разработки. В итоге было принято решение о целесообразности использования графического инструментария компании «Модус» для подготовки и отображения схем в ОИК «КОТМИ-2010».

В течение десяти лет компания Модус активно работает на рынке программного обеспечения для оперативно-диспетчерских служб предприятий электроэнергетики. За это время компания выпустила линейку программных продуктов и завоевала признание большого количества пользователей. В настоящее ее разработки хорошо известны во всех крупных энергосистемах России, в ЦДУ, в ОДУ, РДУ.

На данный момент в ОИК используются два основных инструмента комплекса «Модус»: графический редактор (Sedit32.exe) и модуль отображения схем ActiveXeme (Htsde2.ocx).

Сложилась следующая технология работы со схемами, состоящая из трех этапов:

- 1) создание схемы в графическом редакторе «Модус»;
- 2) загрузка и оживление созданной схемы в модуле настройки ОИК «КОТМИ-2010»;
- 3) использование схемы в оперативной работе.

При необходимости внесения изменений, «оживленная» схема выгружается в файл на диске, редактируется в редакторе «Модус» и снова загружается (обновляется) в ОИК. При этом все привязки к элементам схемы, созданные ранее при оживлении, автоматически сохраняются.

5.2. Включение схем в систему

Для хранения схем в системе предназначена таблица T_MDS. Каждая запись таблицы содержит одну схему. Схема может быть представлена в двух вариантах. Один из них, базовый, используется в оперативной работе. Другой, настраиваемый, может параллельно редактироваться («оживляться») администратором системы. После внесения требуемых корректив и проверки работоспособности изменения внесенные в схему можно утвердить (или отменить), сделав их, тем самым, доступными в оперативной работе.

Для визуального представления списка схем предназначен модуль ScdMds.ModColl, позволяющий группировать схемы по папкам. Данный модуль хранит «древовидные» коллекций в таблице T_COLL.

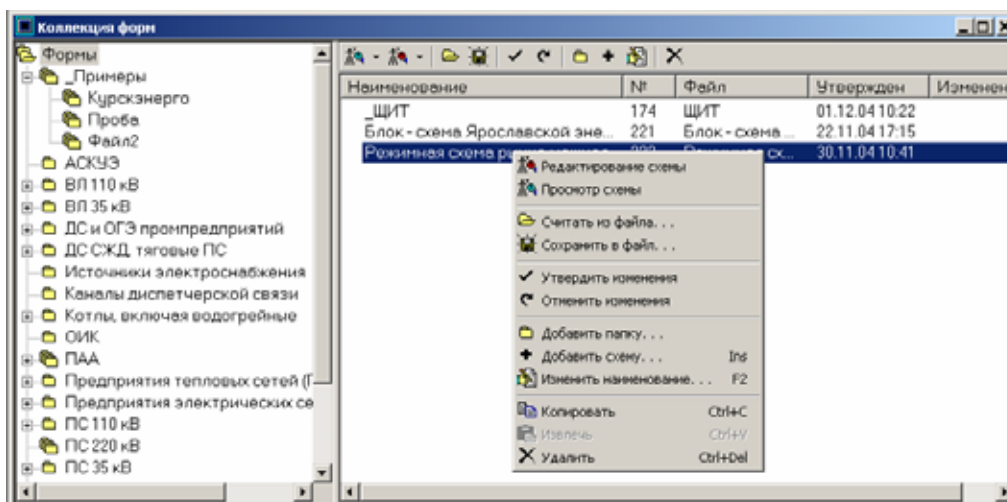


Рисунок 23 – «Дерево» схем

Формирование коллекции схем – процесс интуитивно понятный и простой. Вначале (или по мере необходимости) формируется «дерево» папок. Затем в папки добавляются нужные схемы, ранее подготовленные в редакторе «Модус». Схемы легко перетаскиваются между папками с использованием механизма «Drag and Drop» и операций с буфером обмена Windows. При необходимости могут быть удалены или сохранены в файл на диске.

Схема подготовленная в редакторе «Модус» обычно не обладает информацией о динамических привязках, позволяющих отображать архивные значения, связывать с ними паспорта, осуществлять телеуправление, ручной ввод и другие операции оперативной работы. Данные настройки можно осуществить дополнительно в редакторе «оживления» схем «КОТМИ-2010» (ScdMds.ModEdit).

5.3. «Оживление» схем

Так называемое «оживление» (привязка) представляет собой интерактивный процесс связывания с элементами схем «Модус» данных из архивов ОИК, полей отображения времени, зон переходов и т.п. Настройка схем осуществляется с помощью модуля ScdMds.ModEdit.

На данный момент возможности «оживления» позволяют:

- отображать, и, в специальных случаях, анимировать данные телеметрии. Помимо телеметрической информации на схему могут быть выведены данные из любых архивов, используемых в системе. Например, ведомость, почасовые значения и пр.;
- выполнять телеуправление со схем;
- осуществлять ручной ввод параметров;
- показывать паспорта;
- анализировать различные события и сигнализировать о возникающих аварийных ситуациях;
- осуществлять навигацию и переходы к схемам и другим функциональным модулям системы;
- оперативно показывать дополнительную информацию с помощью всплывающих подсказок (hint-ов).

Прямо в редакторе «оживления» можно оценить полученный результат, щелкнув по иконке «часов» и запустив, тем самым, процесс реального подключения и отображения данных с сервера ОИК.

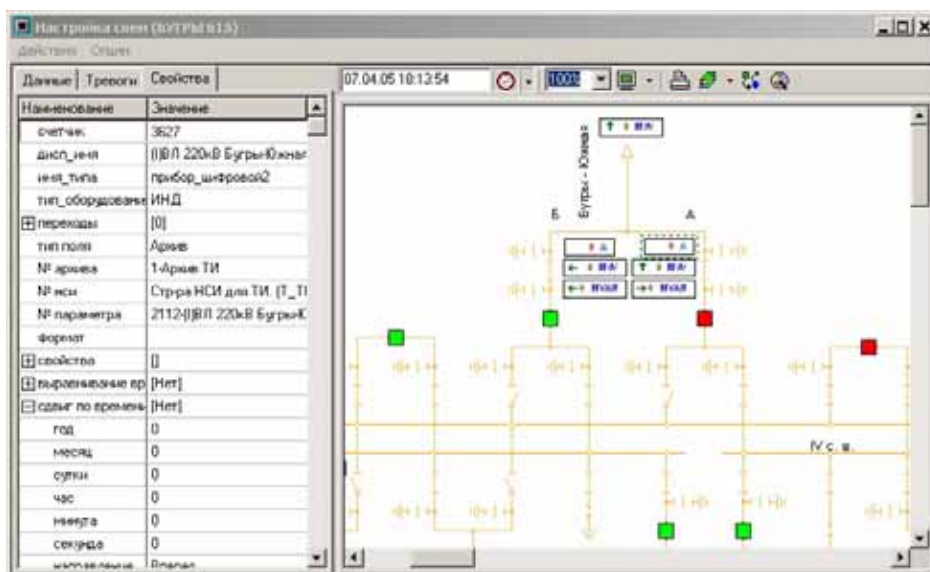


Рисунок 24 – «Оживление» схем

Принцип «настройки» состоит в следующем: с каждым элементом схемы, элемент выбирается мышкой, можно связать информацию о переходах и одно поле (динамическую зону). С помощью «переходов» можно организовать удобную навигацию по схеме (схемам). Поля позволяют организовать динамическую связь с сервером ОИК, запрос, обновление и отображение требуемых данных на элементе схемы.

Все операции по «оживлению» выполняются с помощью окна свойств расположенного слева (закладка «Свойства»).

Для задания «перехода» надо указать его тип (схемы «Модус», формы документов или наборы ретроспективы) и выбрать требуемый документ из выпадающего списка. При этом, в поле «параметры» будет автоматически сформирован список параметров перехода. Для схем «Модус» этот список может быть дополнен вручную. Допустимы следующие элементы:

- TBL (идентификатор таблицы: T_MDS, T_FRM или T_RTS. Обычно формируется автоматически;
- REC (идентификатор записи в таблице, содержащей документ. Обычно формируется автоматически. Синонимы: DOC, RECID, ITEMID);
- PAGE (номер страницы на схеме, 0..n, page=1);
- ITEM (элемент на схеме, счетчик элемента, item=245);
- ZOOM (масштаб отображения в процентах, zoom=33, или 0-обзорный, -1-по ширине, -2-по высоте);
- POSX (смещение ScrollX);
- POSY (смещение ScrollY).

Отдельные элементы в списке должны разделяться запятыми. Аналогичный список параметров можно использовать при настройке конфигураций АРМ для вызова схем «Модус».

С элементы схем имеющими свойства «текст» и(или) «состояние» помимо переходов можно связать поле. На данный момент поддерживаются поля 2-х типов: время и архивное значение.

Время. Для задания поля времени элемент должен иметь свойство «текст». Время форматируется в соответствии с выбранным форматом и выводится как текстовая строка.

Как рассчитать требуемое время? Надо руководствоваться следующими правилами: схема всегда имеет актуальное время отображения. Это либо конкретное время, вручную заданное пользователем, либо текущее - автоматически формирующееся в результате работы режима «постоянного обновления».

И в том, и в другом случае поле времени берет за базовое именно «актуальное» время схемы. Далее, «актуальное» время выравнивается по значению свойства

«Выравнивание». Выровненное время смещается на значение свойства «Сдвиг». Результат форматируется по свойству «Формат» и высвечивается на экране.

Аналогичным образом вычисляется и время запроса для архивных полей.

Заканчивая обсуждение «времени» надо отметить, что сама схема может иметь свойства «выравнивания» и «сдвига» по времени. Эти свойства становятся доступны в редакторе свойств если щелкнуть по пустому месту на схеме. В этом случае «актуальное» время схемы сначала преобразуется централизованно, а затем уже используется полями, согласно выше описанному алгоритму.

Архивы. Для этого типа поля главное правильно выбрать параметр, который должен отображаться. Для этого нужно внимательно задать свойства «№ архива», «№ нси» и «№ параметра». Свойства «№ архива» и «№ нси» взаимосвязаны. Если сначала выбрано НСИ, то список архивов формируется только для него.

Когда параметр выбран, можно подстроить другие свойства поля:

«Формат» для аналоговых значений может задавать количество знаков после запятой, а для логических – синонимы «False, True».

«Свойства» позволяют инвертировать значение перед отображением, добавлять к текстовому значению символ флага, разрешать или запрещать редактирование значений, запрещать вывод в текстового значения в свойство «текст» элемента схемы.

Свойства «Выравнивание» и «Сдвиг» по времени используются для уточнения времени запроса данных с сервера в соответствии с алгоритмом описанным в поле «Время».

Все заданные архивные поля можно найти также на вкладке «Данные». Поля сгруппированы по архивам и могут служить альтернативным средством навигации и управления на схеме.

Все настройки связанные со схемой на этапе «оживления» сохраняются вместе с файлом схемы. Если требуется подкорректировать схему в редакторе «Модус», то надо в модуле «Дерево схем» сохранить «оживленную» схему в файл на диске. Изменить схему в редакторе и загрузить обновленный вариант с помощью команды «Считать из файла». Выполненные ранее настройки «оживления» при этом будут сохранены.

Модуль отображения и «оживления» схем внешне похожи. Отличие только в том, что в модуле отображения отсутствует закладка «Свойства», позволяющая осуществлять настройку элементов привязки.

5.4. Пометки на схемах

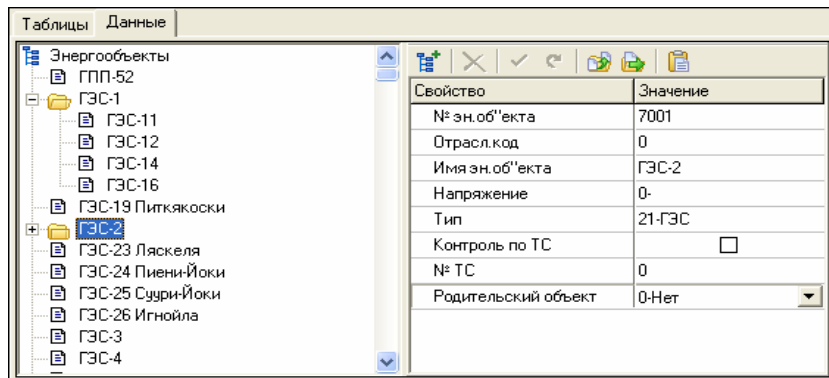
Система диспетчерских пометок позволяет управлять размещением надписей (значков) на схемах. Через эту функцию реализуется механизм контроля состояния оборудования, контроль выдачи команд ТУ, контролируется порядок выполнения работ на объекте.

Описание энергообъектов (оборудования).

Реализована возможность описывать дерево (иерархию) энергообъектов (оборудования). Требуемая детализация определяется решаемыми задачами.

С формирования этого дерева должно начинаться внедрение комплекса. Все обрабатываемые данные в комплексе привязываются к этому дереву.

Для реализации этой функции в таблицу T_ENOBJ было добавлено поле T_ENOBJ_ID – (идентификатор объекта верхнего уровня). Был разработан и добавлен в библиотеку модуль SCDADM.MODENOBJ. Данный компонент позволяет в интерактивном режиме создавать список и описывать иерархические связи между энергообъектами.



Для работы с состоянием оборудования в БД добавлены таблицы:

- 1) T_STATE_T (типы состояния энергообъектов).
- 2) T_EV_STATE (5001, события изменения состояний).
- 3) T_ARCH_STATE (архив состояний энергообъектов).

Пометки

Для работы с пометками в БД ОИК необходимо добавить следующие таблицы:

- T_MARK_C (категория пометок, для группировки).
- T_MARK_T (типы пометок).
- T_CURR_M (архив пометок).
- T_EV_MARK (5002, 5003. События добавления и удаления пометки).

Пометки (значки на схеме «МОДУС») могут быть 2-х типов (T_MARK_C):

- 1) Графические (MARK_C_ID=1). Собственно это и есть встроенные пометки «Модус», используемы для работе в тренажере. Номера этих пометок в таблице T_MARK_T - 1..49. Они соответствуют номерам элементов пометок в «Модусе» и указываются в поле MARK_T_NUM.
- 2) Текстовые (MARK_C_ID=2). Произвольные текстовые плакаты, написание и назначение которых определяется нуждами пользователя при настройке системы. Для текстовых пометок значение поля MARK_T_NUM = 0.

При заполнении таблицы T_MARK_T в поля заносится следующая информация:

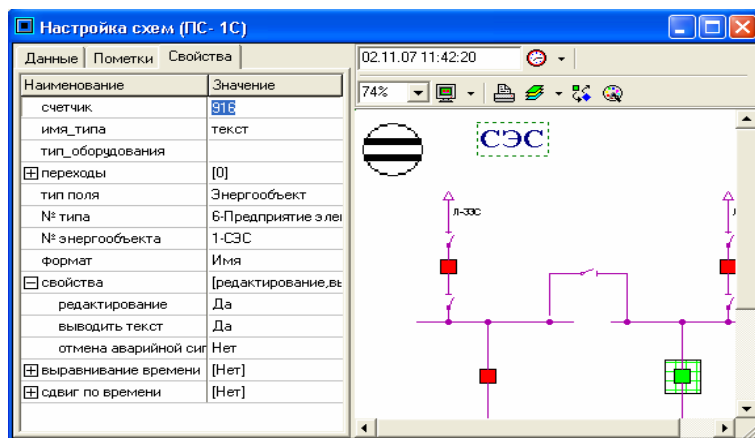
- MARK_T_NAME – наименование пометки. Показывается в списках и подсказках (hint).
- MARK_T_CAPTION – текст на плакате (для текстовых пометок)
- MARK_T_TEXT – шаблон примечания к пометке. Высвечивается по умолчанию при создании пометки и может быть скорректирован пользователем.
- MARK_T_TU – если данный флаг установлен, то для энергообъектов с данной пометкой запрещается (блокируется) операция телеуправления
- MARK_T_STATE_ID – состояние (если задано), которое устанавливается для энергообъекта по выставлению пометки. Состояния выстраиваются по

приоритетам. Поэтому, при наличии нескольких пометок, будет установлено наиболее приоритетное состояние.

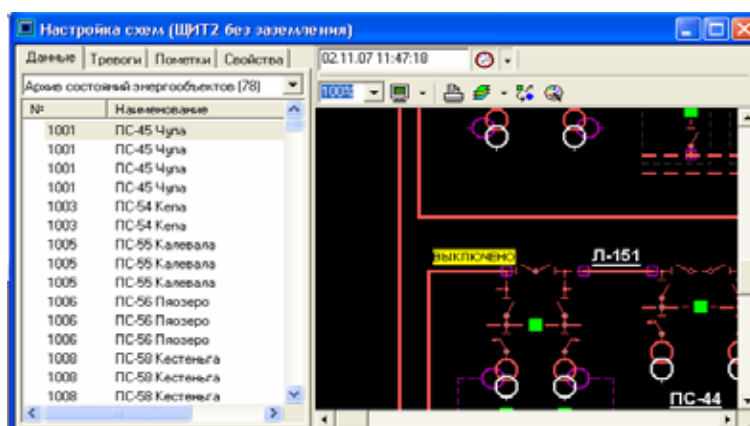
Номер типа пометки (MARK_T_NUM) должен соответствовать номеру пометки в тренажере МОДУС.

Если это поле равно 0, тогда пометка наша текстовая, и то, что написано в поле (MARK_T_CAPTION) будет выводиться на схему.

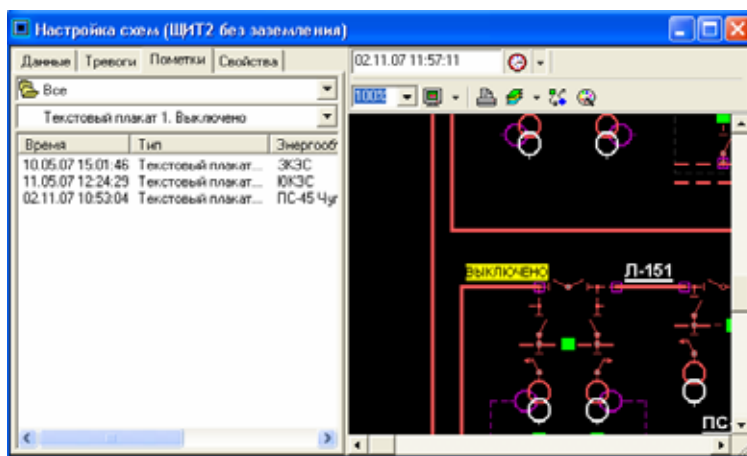
Для работы с пометками на этапе «оживления» должны быть расставлены поля «энергообъектов». Именно для этих полей можно выставлять пометки.



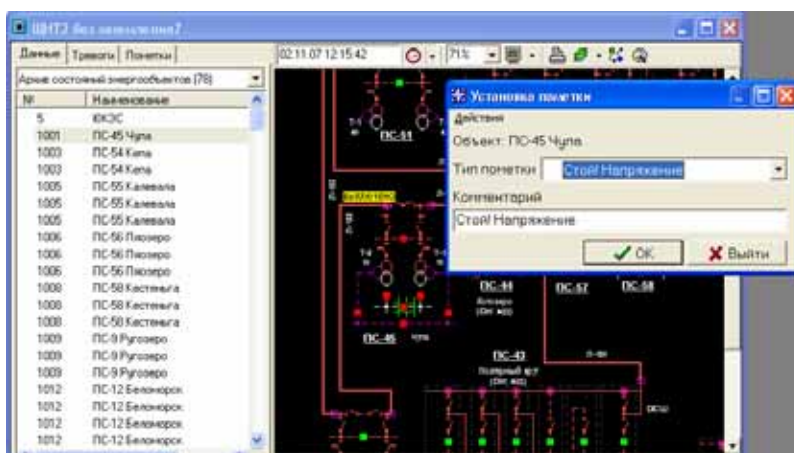
После расстановки полей, существующие привязки можно посмотреть во вкладке «Данные» в списке для «Архива состояний энергообъектов».



Во вкладке «Пометки» показывается список выставленных пометок для энергообъектов присутствующих на схеме. Для «родительских» в списке показываются пометки выставленные к вложенным. Данный список можно фильтровать по энергообъектам и типам пометок.



При двойном щелчке по строке пометки на схеме высвечивается сам элемент пометки, если он присутствует.



Чтобы выставить пометку, нужно отметить нужный энергообъект на схеме и, по правой клавише (контекстное меню), выбрать пункт «Установить пометку...». В появившемся окне выбрать тип пометки, при необходимости, изменить комментарий и нажать клавишу «ОК».

Для удаления пометки, требуется выделить значек, выбрать пункт меню «Удалить пометку» и подтвердить запрошенное действие.

После появления значка пометки на схеме можно изменить ее первоначальное положение и масштаб отображения. Эти действия осуществляются через меню «Положение и размер». Другой способ передвинуть пометку, выделить ее и щелкнуть мышкой по требуемому месту схемы при одновременно нажатой клавише «Ctrl».

Значки пометок можно отметить стилем «Модуса». Для этого в редакторе стилей нужно создать стиль 'ОИК_ПОМЕТКА'.

6. ПОЛЬЗОВАТЕЛИ И ПРАВА ДОСТУПА (ScdAdm.ModTblUsrGr)

Данный модуль является одним из основных инструментов системного администратора. Он позволяет сформировать группы пользователей, авторизовать и ограничить права доступа к информации в рамках этих групп.

Общая схема защиты прав доступа следующая:

- 1) в таблице T_USRGR создаются одна или несколько записей. Одна запись данных описывает одну группу пользователей;
- 2) для каждой группы задаются права доступа к таблицам НСИ, архивам и другим объектам системы. Отдельно определяются права на запуск системных процедур и программных модулей;
- 3) в таблице T_USRS прописываются пользователи системы. Одному пользователю ставится в соответствие одна группа доступа.

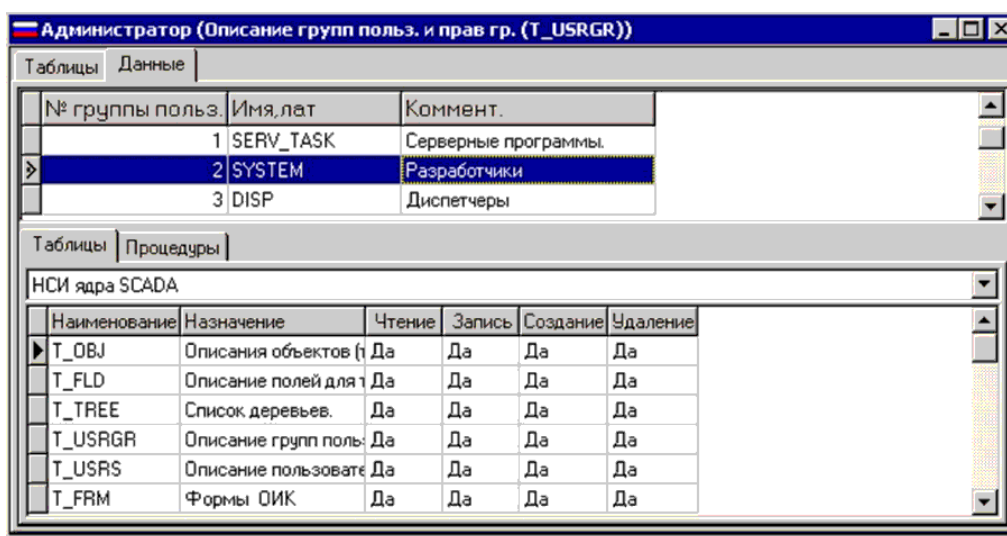


Рисунок 25 – Редактор групп и прав доступа

Для таблиц (объектов) предусмотрены четыре категории прав доступа: на чтение информации, изменение, добавление новых записей и удаление существующих. По умолчанию, изначально разрешен полный доступ к информации. Весь список таблиц для удобства просмотра и модификации разделен на категории, аналогичные категориям объектов в модуле системного администратора.

Аналогичным образом задаются права доступа для серверных процедур (T_PROC_SRV). Для них, правда, предусмотрена только одна категория прав – право на запуск.

Серверных процедур немного и используются они внутри модулей, написанных разработчиками системы Scada. Ограничение доступа к процедурам - это скорее нонсенс, чем правило.

№ группы польз.	Имя, лат	Коммент.
1	SERV_TASK	Серверные программы.
2	SYSTEM	Разработчики
3	DISP	Диспетчер

Таблицы Процедуры	
Наименование	Запуск
READ_ARCH	Да
WRITE_ARCH	Да
CALC_ARCH	Да
TBLFLDINIT	Да
MODADM	Да

Рисунок 26 – Права доступа для модулей и системных процедур

Более интересна другая, так сказать, «недокументированная» возможность. Если в списке процедур разместить имена модулей системы Scada (см. приложение), то запрет доступа к модулю приведет к невозможности его использования в АРМ, даже если он присутствует в конфигурации, с которой этот АРМ запущен. На рисунке, в качестве примера, в последней строке таблицы процедур указано имя модуля администратора.

Специальное имя «ARMCONFIG» предназначено для управления доступом к модулю конфигурации АРМ, вызываемому из программы АРМ клиента.

ПРИЛОЖЕНИЕ 1

1 РЕКОМЕНДАЦИИ ПО ВВОДУ КОМПЛЕКСА В ЭКСПЛУАТАЦИЮ

Перечень необходимой информации:

- 1) Перечень энергообъектов (ОДУ, АО, ПЭС, РЭС, подстанции и т.д.).
- 2) Перечень устройств телемеханики, с которых будет осуществляться сбор информации.
- 3) Списки ТИ, ТС.
- 4) Описания необходимых расчетов.
- 5) Разработать формы для отображения информации.
- 6) Разработать входные и печатные документы ведомости.
- 7) Определить необходимые для отображения в комплексе схемы и возможность использования редакторов Grim или Rizograf.

1.1 Настройка нормативно-справочной информации ядра комплекса.

Первичная настройка производится в базе DbScada.mdb с помощью Microsoft Access.

Корректироваться должны следующие таблицы:

- T_USRGR. Могут быть добавлены новые категории пользователей или скорректированы имена существующих. Не надо трогать или изменять описание категории системных программ (Id категории = 1);
- T_USRS. Может быть добавлен пользователь, под именем которого затем можно будет подключиться к серверу SCADA NT с клиентской части. Описания системных программ (Id категории = 1) изменять не надо;
- Записи из таблиц T_FRM и T_SCH можно удалить все;
- После заполнения таблицы T_ENOBJ (перечень энергообъектов) не забыть внести ID энергообъекта, на котором работает данный комплекс в таблицу T_THIS;
- В таблицах T_EV_CAT и T_EV_COD можно, в соответствии с требованиями заказчика, исправить наименования категорий и кодов событий, признаки принадлежности некоторых кодов к аварийным (то есть требующим квитирования), глубину хранения категорий событий. Удалять записи не рекомендуется. По крайней мере, до полного понимания работы подсистемы, генерации событий и оповещения о них;
- Таблицу T_RTS (описания общих и личных стандартных наборов модуля Ретроспектива) можно почистить;
- В таблице T_DATA_BAG хранятся настройки АРМ-ов комплекса. Их можно не удалять, а только скорректировать в соответствии с требованиями заказчика из модуля Администратора после запуска клиентской части комплекса;
- Из таблицы T_DOC можно удалить все записи или оставить в качестве примера иерархии документов;
- В таблице T_SPRG необходимо отключить на время подготовки базы НСИ запуск всех серверных программ (поле SPRG_NOSTART должно быть в TRUE);
- Удалить все записи в таблицах T_OBJ_R и T_PROC_R. В них определяются права на доступ к таблицам базы данных и права на запуск серверных процедур комплекса для каждой категории пользователей. Если в этих таблицах записей нет, то разрешено всё. После запуска клиентской части можно будет описать все необходимые ограничения на доступ из модуля Администратор;
- Таблицу модулей системы T_MOD можно почистить.

Предполагаем, что имеется в наличии инсталляция клиентской части комплекса с последними версиями модулей.

1.2 Настройка таблиц деревьев

- Таблица T_TREE_DOC (деревья документов) ведется модулем Конструктор документов. Содержимое таблицы также может быть удалено при внедрении комплекса на новом объекте.
- Таблица T_TREE_ENOBJ (дерево энергообъектов) ведется в модуле Администратор. В данное время используется модулем События при выборе подключения к оповещению о событиях на заданном перечне энергообъектов. Дерево строится после заполнения таблицы с описаниями энергообъектов T_ENOBJ.

1.3 Настройка архивов

Первичную настройку также необходимо выполнить с помощью MS Access, так как при первоначальном запуске Серверной части комплекса, в соответствии с настройкой, будут созданы файлы базы реального времени.

Настройка выполняется в таблице T_ARCH в соответствии с описанием полей в документе «Структура БД комплекса КОТМИ-2010». Директорий или директории для хранения файлов базы реального времени должны быть созданы до первого запуска Серверной части комплекса. Не следует удалять или изменять идентификаторы записей для архивов:

- ТИ;
- ТС;
- ПТИ;
- ПТС.

Если в настройке соответствующего архива была указана запись в SQL-базу (поле ARCH_CON_SQL = TRUE), то необходимо почистить соответствующую таблицу архива T_ARCH_...._A.

1.4 Порядок ввода информации для приема ТИ, ТС.

Ввод базы ТМ производится после первичной настройки базы комплекса (см. раздел 2) и инсталляции серверной и клиентской частей комплекса. В момент ввода НСИ рекомендуется отключить автоматический запуск серверных программ комплекса (таблица T_SPRG). Работа с НСИ производится с помощью модуля Администратора клиентской части комплекса.

Рекомендуется следующий порядок ввода:

- 1) Скорректировать справочники (раздел Справочники);
- 2) Описать задействованные ЦППС (таблица T_CIOS);
- 3) Описать обрабатываемые RTU (таблица T_RTU);
- 4) Описать энергообъекты (T_ENOBJ);
- 5) В таблице T_THIS проставить идентификатор энергообъекта, на котором работает данный комплекс;
- 6) Описать задействованные ТИ (T_TI);
- 7) Описать задействованные ТС (T_TS).

После ввода информации, перечисленной выше, необходимо проверить её прием и обработку в комплексе. Проверка правильности приема и обработки ТИ и ТС производится с помощью модуля Ретроспектива.

ПРИЛОЖЕНИЕ 2

1 ОСНОВНЫЕ ФУНКЦИОНАЛЬНЫЕ МОДУЛИ КЛИЕНТСКОЙ ЧАСТИ

- **ScdAdm** - библиотека ActiveX-модулей администратора;
- **ScdAdm.ModAdm** – администратор;
- **ScdAdm.ModTbl** - универсальный просмотр таблиц (внутреннее использование);
- **ScdAdm.ModTblArch** - просмотр таблицы архивов;
- **ScdAdm.ModTree** - универсальный просмотр деревьев (внутреннее использование);
- **ScdAdm.ModTblUsrGr** - группы пользователей и права доступа;
- **ScdAdm.ModTblLib** - таблица (библиотека) общих модулей системы;
- **ScdAdm.ModTblCalc** - таблица дорасчетов;
- **ScdAdm.ModTblCmd** - таблица команд для щита;
- **ScdStd** - стандартные функциональные модули;
- **ScdStd.ModRts** – ретроспектива;
- **ScdStd.ModDoc** – документы;
- **ScdStd.ModDocE** - документы (конструктор);
- **ScdStd.ModSch** – схемы;
- **ScdStd.ModSchE** - схемы (конструктор);
- **ScdStd.ModEvn** - работа с событиями;
- **ModMktCBI** - отправка макетов ЦБИ;
- **ModMktCDU** - отправка макетов ЦДУ;
- **ScdTool** - дополнительная библиотека ActiveX-модулей администратора;
- **ScdTool.ModTbl_Ti** - таблица ТИ;
- **ScdTool.ModTbl_Ts** - таблица ТС;
- **ScdTool.ModTbl_PTi** - параметры ПТИ;
- **ScdTool.ModTbl_PTs** - параметры ПТС;
- **ScdTool.ModTbl_Calen** - энергетический календарь;
- **ScdTool.PspTI** - паспорта ТИ;
- **ScdTool.PspTS** - паспорта ТС;
- **ScdTool.SnapPTM** - срез ТИ-ТС;
- **ScdMds.ModColl** – коллекция («дерево») схем «Модус»;
- **ScdMds.ModView** – отображение схемы «Модус»;
- **ScdMds.ModEdit** – настройка («оживление») схемы «Модус».

ПРИЛОЖЕНИЕ 3

1 ЯЗЫК ПРОГРАММИРОВАНИЯ TScript

1.1 Общий синтаксис

Разделители

Пробелы, символы табуляции, перевода на новую строку и перевода страницы используются как разделители. Вместо одного из таких символов может использоваться любое их количество.

Комментарии

Комментарии начинаются парой символов `/*`, заканчиваются парой символов `*/`. Разрешены везде, где допустимы пробелы.

Символы `//` начинают комментарий, который заканчивается в конце строки, на которой они появились.

ПРИМЕР:

```
FUN1 (Prm1, Prm2) // Пример использования комментариев
{
    var Loc1 /*переменная1*/, Loc2 /*переменная2*/, Loc3;

    /* Тело
       функции */
    . . .
}
```

Идентификаторы

Идентификаторы используются как имена констант, параметров, переменных и функций.

Допустимые символы: цифры 0-9, латинские прописные и строчные буквы a-z, A-Z, символ подчеркивания (`_`), точка (`.`). Первый символ не может быть цифрой или точкой.

Идентификатор может быть произвольной длины.

Регистр при сравнении имен, не учитывается (`"Abs" == "abs"`).

ПРИМЕР: (допустимые идентификаторы)

```
Get_TI
TS.Value
Set_Flags.123
_New.Signum
```

Зарезервированные слова

`const, var, if, else, elseif, while, break, continue, exit, result;`

Типы данных

В выражениях языка используются следующие типы данных:

- Empty – пустой элемент;
- Integer – знаковое целое 4 б ();
- Double – вещественное 8 б ();
- String – строка произвольной длины (может включать 0x0);

- Array – векторный массив, произвольной длины. Каждый элемент массива может иметь любой допустимый тип;
- Object – объект (механизм расширения);
- Производных типов данных нет.

Описания констант, параметров и переменных

Прежде чем имя (идентификатор) может быть использовано в выражении, оно должно быть описано. Идентификатор может представлять собой константу, параметр функции или локальную переменную.

Допускаются константы 3-х основных типов: целые, вещественные и строковые. Значения констант подставляются в выражения на этапе компиляции функций.

Параметры и переменные представляют собой данные, подобные типу VARIANT в языке BASIC, и в процессе вычисления могут принимать значения любых основных типов.

Контроль совместимости типов производится на этапе вычисления.

Все идентификаторы, не являющиеся константами, параметрами, переменными или именами встроенных функций, считаются внешними функциями и связываются на этапе компоновки исполняемого модуля (калькулятора).

ПРИМЕР:

```
FUN2 (Prm1)      // fun2-имя функции, prm1 - параметр
{
    const Const1=1,Const2=2.3,Const3="Строка"; // Константы

    var Loc1; // локальная переменная

    Result = 0; // Result - встроенная переменная-результат
    Loc1 = Int(Prm1); // Int = встроенная функция
    While ( loc1 > 0) // ЦИКЛ-ПОКА
    { // Тело цикла
        Result = result +prm1 + Const1;
        Loc1 = Loc1 - 1;
    }
}
```

1.2 Операции и выражения

Выражения

Выражение состоит из одного или большего числа операндов и символов операций. Значение выражения м.б. передано как параметр.

ПРИМЕР:

```
X = B + 1 + A[10, 12] + B[1][N]
STR(A % 2) + "12345"
[2] = ((Y[1] = 100) * Z) / 2
```

Арифметические операции

- = - присваивание,
- + - сложение,
- - вычитание,
- * - умножение,
- / - деление,

\ - целочисленное деление,

% - деление по модулю.

Операции отношения

Логическое значение ЛОЖЬ представляется целым нулевым значением, а значение ИСТИНА представляется любым ненулевым значением.

= - равенство,

!= - неравенство,

> - больше,

>= - больше или равно,

< - меньше,

<= - меньше или равно.

Логические операции

Логическое значение ЛОЖЬ представляется целым нулевым значением, а значение ИСТИНА представляется любым ненулевым значением.

! - отрицание,

|| - логическое ИЛИ

&& - логическое И

Побитовые операции

Побитовые операции применимы только к целочисленным операндам.

~ - дополнение до единицы

& - побитовое И

| - побитовое ИЛИ

^ - исключающее ИЛИ

Прочие операции

, - операция запятая. Выражения разделенные запятой выполняются слева направо. В качестве результата берется значение последнего выражения.

[x1,x2,..xn] – операция конструктор массива. Выражения, заключенные в квадратные скобки, рассчитываются и из них формируется массив.

ПРИМЕР:

```
Arr = [1, "Строка", 2.33];
```

Приоритеты и порядок выполнения операций

Для каждой группы операций в приведенной ниже таблице приоритеты одинаковы. Чем выше приоритет группы, тем выше она расположена в таблице. Порядок выполнения определяет группировку операций и операндов (слева направо или справа налево), если отсутствуют скобки и операции относятся к одной группе.

Операция	Наименование	Порядок выполнения
()	Вызов функции	Слева направо
!	Логическое отрицание	Справа налево
~	Побитовое отрицание	
-	Изменение знака	
*	Умножение	Справа налево
/	Деление	
\	Целочисленное деление	
%	Деление по модулю	
+	Сложение	Справа налево
-	Вычитание	
<	Меньше, чем	Справа налево
<=	Меньше или равно	
>	Больше, чем	
>=	Больше или равно	
==	Равно	Справа налево
!=	Не равно	
&	Побитовая И	Справа налево
^	Побитовая операция исключающее ИЛИ	Справа налево
	Побитовая операция ИЛИ	Справа налево
&&	Логическая операция И	Слева направо
	Логическая операция ИЛИ	Слева направо
=	Присваивание	Справа налево
,	Операция запятая	Слева направо
[,]	Операция конструктор-массива	Слева направо

1.3 Операторы

Формат и вложенность

Один оператор может занимать одну или более строк. Два или большее количество операторов могут быть расположены на одной строке.

Операторы, управляющие порядком выполнения (if, while), могут быть вложены друг в друга.

Составной оператор

Составной оператор (блок) состоит из одного или большего числа операторов любого типа, заключенных в фигурные скобки ({}). После закрывающей скобки не должно быть точки с запятой (;).

ПРИМЕР:

```
{ X = 1; Y = 2; If (X >= Y) Z = X; else Z = Y; }
```

Оператор – выражение

Любое выражение, заканчивающееся точкой с запятой (;), является оператором.

Оператор присваивания

= - присваивает значение выражения, справа, переменной, параметру или элементу массива слева от знака операции. В выражении м.б. несколько операций присваивания.

Идентификатор = выражение.

Оператор выхода из цикла break

Break –

прекращает выполнение ближайшего вложенного внешнего оператора WHILE. Управление передается оператору, следующему за заканчиваемым.

Оператор описания локальных констант const

const идент1=значение1, идент2=значение2 ..идентN=значениеN;

Данный оператор служит для определения имен локальных констант функции. Он допускается только первым оператором в составном операторе функции.

ПРИМЕР:

```
FUN1
{
    const Const1=1, Const2="12345";
    . . .
}
```

Оператор продолжения цикла continue

Continue –

передает управление в начало ближайшего внешнего оператора WHILE. Этот оператор по действию противоположен оператору break.

Оператор выхода из функции exit

Exit -

Завершает выполнение текущей функции и передает управление в вызывающую. Возвращается результат, сформированный на момент выхода в переменной Result.

Условный оператор if-elseif-else

```
        If (выражение)
        Оператор1
    ElseIf (выражение)
        Оператор2
    Else
        Оператор3
```

Если какое либо условное выражение истинно, то выполняется оператор, следующий за соответствующим if или elseif. Если все выражения ложны, то выполняется else-оператор.

Часть else может опускаться. Частей elseif может быть сколько угодно.

Оператор описания локальных переменных var

var идентификатор1, идентификатор2 .. идентификаторN;

Данный оператор служит для описания имен локальных переменных функции. Он допускается только первым или вторым оператором (после const) в составном операторе функции.

ПРИМЕР:

```
FUN1
{
    var Local1, Local2;
    . . .
}
```

Оператор цикла while

While (выражение)

Оператор

Если выражение истинно, то оператор выполняется до тех пор, пока выражение не станет ложным.

Если выражение ложно, то управление передается следующему оператору.

1.4 Функции

Определение функции.

Функция определяется описанием имени функции, формальных параметров и составного оператора (блока), описывающего выполняемые функцией действия.

Если в теле функции определяются константы, то первым оператором составного оператора функции должен стоять оператор const, описывающий имена и значения этих констант, разделенные запятыми.

Если в теле функции используются локальные переменные (кроме Result)? То следующим оператором составного оператора функции должен стоять оператор var, описывающий имена этих локальных переменных.

ПРИМЕР:

```
FUNMAX (Prm1, Prm2)
{
    const Const1=1, Const2="строка", Const3=1.234;
    var Loc1, Loc2;
    // ...
    if (Prm1 >= Prm2)
        Result = Prm1;
    else
        Result = Prm2;
}
```

Вызов функции

Вызов функции осуществляется по ее имени. Если функция имеет параметры, то значения параметров перечисляются в круглых скобках, через запятую. Количество формальных и фактических параметров должны соответствовать друг другу. Параметры передаются по значению. Каждая функция возвращает результат – значение функции.

Встроенные функции

Данные функции присутствуют в исполняющей системе языка. Названия, назначение, типы аргументов и результата большинства функций соответствуют аналогичным функциям исполняющей библиотеки языка C.

Функция	Описание
Общие	
ABS(X)	Абсолютное значение X
IF(X, Y1, Y2)	Если X=ИСТИНА, то возвращает Y1, иначе Y2
MAX(X, Y)	Наибольшее из X и Y
MIN(X, Y)	Наименьшее из X и Y
ABORT(X)	Аварийное завершение функции с кодом возврата X
TRY(X)	Код завершения выполнения выражения X. Вариант try..catch блока. Если успешно = 0;
Преобразования типов	
TYPE(X)	Тип (0-Empty,1-Int,2-Dbl,3-Str,4-IntS,5-DblS)

INT(X)	Целое
DBL(X)	Вещественное
STR(X)	Строка
ARR(N)	Формирование массива из N элементов
Математические	
SIN(X)	Синус
COS(X)	Косинус
TAN(X)	Тангенс
ATAN(X)	Арктангенс
EXP(X)	Экспонента
LDEXP(X, E)	Экспонента (X-double, мантисса, E-integer, экспонента)
LOG(X)	Логарифм
LOG10(X)	Десятичный логарифм
SQRT(X)	Квадратный корень
POW(X, Y)	Возведение в степень (X**Y)
ROUND(X, N)	Округление X, до N знаков после запятой
Строковые (также для работы с массивами)	
LEN(S) => N LEN(A) => N	Длина строки (без завершающего 0) или массива
MID(S,P,L) => S MID(A,P,L) => A	Получение подстроки или подмассива (S(A)-источник, P-начальная позиция(0..N), L-длина подстроки)
CMP(S1,S2) => N	Сравнение строк (результат (-,0,+))
CMPI(S1,S2) => N	Сравнение строк без учета регистра (результат (-,0,+))
POS(S1,S2) => N	Поиск подстроки S2 в строке S1 (-1 не найдена)
LEFT(S,N) => S LEFT(A,N) => A	Получение подстроки (подмассива) слева
RIGHT(S,N) => S RIGHT(A,N) => A	Получение подстроки (подмассива) справа
CHR(B) => S CHR(AB) => S	Преобразование кода символа в символ Преобразование массива кодов символов в строку символов
ORD(S) => B ORD(S) => AB	Получение кода символа или массива кодов, если строка состоит из нескольких символов
PRINTF(SFmt,AVal) => S	Формирование строки по шаблону и массиву значений. Правила форматирования совпадают с соглашениями принятыми в языке «C» для функции printf. Кроме определения длины аргументов ({h l I64 L})
SCANF(SBuf,SFmt) => AVal	Сканирование строки и выделение массива значений по заданному шаблону. Правила сканирования совпадают с соглашениями принятыми в языке «C» для функции scanf. Кроме определения длины аргументов ({h l I64 L})
Работа со временем(Integer в формате UNIX и Double с миллисекундами)	
TMMSEC(T) => N	Миллисекунды, mseconds after the seconds - [0,999]
TMSEC(T) => N	Секунды, seconds after the minute - [0,59]
TMMIN(T) => N	Минуты, minutes after the hour - [0,59]
TMHOUR(T) => N	Часы, hours since midnight - [0,23]
TMDAY(T) => N	День месяца, day of the month - [1,31]
TMMON(T) => N	Месяц, months since January - [0,11]
TMYEAR(T) => N	Год, years since 1900
TMISDST(T) => N	1-daylight savings time, 0-зимнее, -1-неизвестно
TMWDAY(T) => N	День недели, days since Sunday - [0,6]

TMYPDAY(T) => N	День года, days since January 1 - [0,365]
TMLOCAL(T) => AT	Arr(Year,Mon,Day,Hour,Min,Sec,MSec,IsDst,WDay,Y Day)
TMGM(T) => AT	Arr(Year,Mon,Day,Hour,Min,Sec,MSec,IsDst,WDay,Y Day)
TMMAKE(Y,M,D,H,M,S,Ms) => T	Формирование времени из составляющих
TMMAKEA(AT) => T	Формирование времени из составляющих расположенных в массиве [Year, Mon, Day, Hour, Min, Sec, MSec, [IsDST]]
TMSTR(SFmt,T) => Str	Строковое представление времени. Правила сканирования совпадают с соглашениями принятыми в языке «C» для функции strftime
Взаимодействие с «объектами»	
NEW(OBJTYPE,PARAM)	Создание объекта, конструктор
GET(OBJ, PROP)	Получение значения свойства
PUT(OBJ, PROP, VALUE)	Изменение значения свойства
CALL(OBJ,METHOD,PARAM)	Вызов метода

1.5 Исполняющая подсистема

Для получения результатов функций во время работы программы, группы связанных функций компилируются в Р-код и объединяются вместе в, так называемом, “калькуляторе”. Этот программный компонент позволяет хранить сгенерированный Р-код функций, обеспечивает реализацию ссылок между связанными функциями, исполняет Р-код для получения результатов.

Допускается использование в калькуляторе, не только функций языка и встроенных функций, но и внешних функций, разработанных пользователем. Данные функции вызываются по адресам, указанным в процессе формирования калькулятора.

В программе допускается существование множества экземпляров компонента “калькулятор”.

ПРИЛОЖЕНИЕ 4

1 ВСТРОЕННЫЕ ОБЪЕКТЫ ЯЗЫКА TScript

1.1 IntS. Целое с флагами

Имя типа: «IntS»

Номер типа по умолчанию: 5 (cv_Obj)

Описание: используется для работы с телеметрией. Целочисленному значению сопутствуют архивные флаги.

№	Функции	Результат	Описание
1	INTS(iVal, iFlag)	Объект IntS	Конструктор объекта IntS
2	VAL(vIntS)	Integer	Целочисленная часть значения
3	SIGN(vIntS)	Integer	Флаговая часть значения

№	Свойство	Тип	Описание
1	“Val”	Integer	Целочисленная часть значения
2	“Sign”	Integer	Флаговая часть значения

1.2 DbIS. Вещественное с флагами

Имя типа: «DbIS»

Номер типа по умолчанию: 6 (cv_Obj+1)

Описание: используется для работы с телеметрией. Вещественному значению сопутствуют архивные флаги.

№	Функции	Результат	Описание
1	DBLS(dVal, iFlag)	Объект DbIS	Конструктор объекта DbIS
2	VAL(vDbIS)	Double	Вещественная часть значения
3	SIGN(vDbIS)	Integer	Флаговая часть значения

№	Свойство	Тип	Описание
1	“Val”	Double	Вещественная часть значения
2	“Sign”	Integer	Флаговая часть значения

1.3 Рес. Запись

Имя типа: «Рес»

Номер типа по умолчанию: 7 (cv_Obj+2)

Описание: объект представляет собой структуру, состоящую из типизированных полей. Данный тип является скриптовым аналогом записи, из которых строятся таблицы в памяти (см. «Руководство программиста»). Может использоваться как самостоятельно, так и в составе таблиц. Табличные объекты будут описаны ниже.

Допустимые типы полей записи:

№	Типы полей	Мнемоника	Описание
1	Bool	B1	Логическое значение (1), 0..1
2	Byte	I1	Байт (1), 0..255
3	Small	I2	Целое (2), -32,768 to 32,767
4	Float	F4	Вещественное (4), -3.4E +/- 38
4	Double	F8	Вещественное (8), 1.7E +/- 308
5	Char	CHn	Строка фиксированной длины
6	Memo	CVn	Строка переменной длины
7	Blob	CB	Строка произвольной длины
8	UnixDT	DT	Время в UNIX-формате, mm-dd-yyuu hh:mm:ss
9	UnixMDT	DM	Время в UNIX-формате с милли- секундами, mm-dd-yyuu hh:mm:ss ms
10	Currency	MN	Денежный тип, – 922337203685477.5808..922337203685477. 5807

Для регистрации нового типа записи формируется текстовая строка следующего содержания:

Имя типа|Количество полей N|Описание]<Cr><Lf>

Имя поля0|Тип поля|Описание]<Cr><Lf>

...

Имя поляN-1|Тип поля|Описание]<Cr><Lf>

Имена типа и поля – произвольный текст длиной не больше 20 символов. Тип поля – мнемоника: B1,I1,I2,I4,F4,F8,CHn,CVn,CB,DT,DM,MN. Описание – необязательный текстовый комментарий.

ПРИМЕР (строка описания типа):

“Новый тип|3\r\nНомер|I4\r\nТекст10\CH10\r\nДанные\CB”

№	Функции	Результат	Описание
1	CRecTypReg(sTypeDef)->TypeId	Integer	Регистрация нового типа записи
2	CRecNew(Rec iTypeId sTypeName)->Rec	Запись	Создание экземпляра записи
3	CRecCopy(Rec)-> Crec	Запись	Копирование (дублирование) записи
4	CrecTypeName(Rec iTypeId)->sTypeName	Строка	Получение имени типа записи
5	CRecTypeId(Rec sTypeName)->iTypeId	Integer	Получение номера регистрации типа записи
6	CRecFlds(Rec sTypeName iTypeId)->iFldNum	Integer	Получение количества полей записи
7	CRecFld(Rec sTypeName iTypeId, iFldIdx sFldName)->{sFldName iFldIdx, iFldType, iFldSize}	Массив	Получение характеристик поля: имя или индекс поля, тип (1..10), размер (в байтах)
8	CRecFldGet(Rec, iFldIdx sFldName)->FldValue	Значение поля	Получение значения поля
9	CRecFldPut(Rec, iFldIdx sFldName, FldValue)	Нет	Изменение значения поля

№	Свойства	Тип	Описание
1	“TypName”	Строка	(R) Наименование типа
2	“TypId”	Integer	(R) Зарегистрированный номер типа
3	“Flds”	Integer	(R) Количество полей
4	“Fld” [iIdx sName]	Массив	(R) Характеристики поля: имя или индекс поля, тип (1..10), размер (в байтах).
5	“FldVal” [iIdx sName]	Integer	(R,W) Значение поля

№	Методы	Параметры	Тип	Описание
1	“TypName”	Rec iTypeId	Строка	Получение имени типа записи
2	“TypId”	Rec sTypeName	Integer	Получение номера регистрации типа записи
3	“Flds”	Rec sTypeName iTypeId	Integer	Получение количества полей записи
4	“Fld”	iFldIdx sFldName	Массив	Получение характеристик поля: имя или индекс поля, тип (1..10), размер (в байтах).
5	“TypReg”	sTypeDef	Integer	Регистрация нового типа записи
6	“RecCopy”		Запись	Копирование (дублирование) записи
7	“RecEdit”			Начало редактирования записи. Таблица.
8	“RecPost”			Сохранение изменений. Таблица
9	“RecDel”			Удаление записи. Таблица.

1.4 Tbl. Таблица

Имя типа: «Tbl»

Номер типа по умолчанию: 8 (cv_Obj+3)

Описание: объект представляет собой таблицу, состоящую из упорядоченных записей. Данный тип является скриптовым аналогом таблицы в оперативной памяти (см. «Руководство программиста»).

№	Функции	Результат	Описание
1	TblNew()->Tbl	Tbl	Создание таблицы
2	TblCopy(Tbl,iOpt)->CTbl	Tbl	Копирование (дублирование) таблицы
3	TblFldAdd(Tbl,sName,iType,iSize)		Добавление поля
4	TblFlds(Tbl)->iFldNum	Integer	Количество полей
5	TblFld(Tbl,iIdx sName) ->{sFldName iFldIdx, iFldType,iFldSize}	Массив	Характеристики поля
6	TblKeyAdd(Tbl,sFlds,iOpt)		Добавление индекса (имена полей через “.”)
7	TblKeyDel(Tbl,iIdx)		Удаление индекса
8	TblKeys(Tbl) -> iCnt	Integer	Количество индексов
9	TblKey(Tbl,iIdx)->{sFlds,iOpt}	Массив	Характеристики индекса
10	TblClearRec(Tbl)		Удаление всех записей
11	TblClearKey(Tbl)		Удаление индексов
12	TblClearAll(Tbl)		Удаление записей, полей и индексов
13	TblRecNew(Tbl)->Rec	Rec	Создание новой, пустой записи
14	TblRecEdit(Rec)->Rec	Rec	Регистрация изменений
15	TblRecPost(Rec)		Фиксация изменений
16	TblRecDel(Rec)		Удаление записи
17	TblRec(Tbl,iKey,iIdx)->Rec	Rec	Получение записи по индексу

18	TblRecIdx(Tbl,iKey,Rec)->iIdx	Integer	Получение индекса записи
19	TblRecCmp(Tbl,iKey,Rec1,Rec2) ->iCmp[-1,0,1]	Integer	Сравнение записей таблиц
20	TblRecSeek(Tbl,iKey,KeyRec KeyVal[...])->[Rec Empty, iIdx]	Массив	Поиск по шаблону или значению (значениям) ключевых полей
21	TblRecSort(Tbl,iKey)		Сортировка по индексу (напр. если idxNonMaintained)
22	TblRecCount(Tbl)->iCnt	Integer	Количество записей в таблице

№	Свойства	Тип	Описание
1	"Flds"	Integer	(R) Количество полей
2	"Fld" [iIdx sName]	Массив	(R) Характеристики поля: имя или индекс поля, тип (1..10), размер (в байтах).
3	"Keys"	Integer	(R) Количество индексов
4	"Key" [iKey]	Массив	(R) Характеристики индекса: имя поля и флаги
5	"Rec" [iKey,iIdx]	Rec	(R) Получение записи по индексу
6	"RecIdx" [iKey]	Integer	(R) Получение индекса записи
7	"RecCount"	Integer	(R) Количество записей в таблице

№	Методы	Параметры	Тип	Описание
1	"TblCopy"		Tbl	Копирование (дублирование) таблицы
2	"FldAdd"	sName,iType,iSize		Добавление поля
3	"KeyAdd"	sFlds,iOpt		Добавление индекса (имена полей через ";")
4	"KeyDel"	iIdx		Удаление индекса
5	"ClearRec"			Удаление всех записей
6	"ClearKey"			Удаление индексов
7	"ClearAll"			Удаление записей, полей и индексов
8	"RecNew"		Rec	Создание новой, пустой записи
9	"RecCmp"	iKey,Rec1,Rec2	Integer	Сравнение записей таблиц (-1, 0, 1)
10	"RecSeek"	iKey,KeyRec KeyVal[...]	Массив [Rec Empty, iIdx]	Поиск по шаблону или значению (значениям) ключевых полей
11	"RecSort"	iKey		Сортировка по индексу (напр. если idxNonMaintained)

ПРИЛОЖЕНИЕ 5

1 Внешние функции TScript, реализованные на сервере

В настоящее время в программное обеспечение серверной части комплекса КОТМИ-2010 встроены следующие функции, расширяющие возможности языка TScript (первым символом наименования внешних функций является символ «_»):

№	Наименование функции	Описание
1	_ArchCurr ()	Определение имени архива, для которого выполняется расчет. Вход: нет. Выход: строка с именем архива или пустая строка при ошибке (str).
2	_SetArchCurr (Str)	Установка имени таблицы архива, для которого выполняется расчет. Вход: Str – имя таблицы архива (str). Выход: нет.
3	_ReadData (Id, Time, Str)	Чтение данных из любых архивов. Вход: Id – идентификатор данного (int); Time – время расчета в секундах в формате UNIX (int); Str – имя таблицы архива (str). Выход: значение данного (int, dbl, ints, dbls).
4	_ReadDataDiv(Id, Time, Str)	Чтение данных- делителей из любых архивов. Вход: Id – идентификатор данного (int); Time – время расчета в секундах в формате UNIX (int); Str – имя таблицы архива (str). Выход: значение данного (int, dbl, ints, dbls). Всегда будет не равно нулю.
5	_WriteData(Id, Time, Str, Val)	Запись данных в любые архивы. Вход: Id – идентификатор данного (int); Time – время расчета в секундах в формате UNIX (int); Str – имя таблицы архива (str); Val – записываемое значение (типы – int, dbl, ints, dbls); Выход: нет.
6	_QuanDayMon (Time)	Определение количества дней в месяце. Вход: Time – время расчета в секундах в формате UNIX (int). Выход: в Result количество дней (int).
7	_QuanWorkDay (Time)	Определение количества рабочих дней на заданный день месяца. Для работы этой функции должен вестись энергетический календарь. Вход: Time – время расчета в секундах в формате UNIX (int). Выход: в Result количество рабочих дней (int).
8	_CurrDayType(Time)	Определение типа дня. Для работы этой функции должен вестись энергетический календарь. Вход: Time – время расчета в секундах в формате UNIX (int). Выход: тип дня (int). Определяется по содержимому таблицы T_DAYT базы данных.

9	<u>_NumTypeDayOfMon</u> (Time, TypeDay)	<p>Определение количества дней заданного типа в месяце. Для работы этой функции должен вестись энергетический календарь.</p> <p>Вход: Time – время расчета (int). TypeDay – тип дня (int0. Определяется по содержимому таблицы T_DAYT базы данных.</p> <p>Выход: количество дней (dbls).</p>
10	<u>_NumHourZoneOfDay</u> (Time, NumZone)	<p>Определение количества часов заданной зоны в сутках. Для работы данной функции должен вестись энергетический календарь. Номера зон в таблице T_ZONE базы данных.</p> <p>Вход: Time – время расчета (int); NumZone – номер зоны (int).</p> <p>Выход: количество часов (dbls).</p>
11	<u>_BegDayOfWeek</u> (Time, NumWeek)	<p>Определение начала заданной энергетической недели в месяце. Для работы данной функции должен вестись энергетический календарь.</p> <p>Вход: Time – время расчета (int); NumWeek – номер недели (int).</p> <p>Выход: день начала (dbls).</p>
12	<u>_EndDayOfWeek</u> (Time, NumWeek)	<p>Определение конца заданной энергетической недели в месяце. Для работы данной функции должен вестись энергетический календарь.</p> <p>Вход: Time – время расчета (int); NumWeek – номер недели (int).</p> <p>Выход: день конца (dbls).</p>
13	<u>_ZoneHour</u> (Time, NumZone)	<p>Определение часов, входящих в заданную зону ФОРЭМа. Для работы этой функции должен вестись энергетический календарь. Номера зон в таблице T_ZONE базы данных.</p> <p>Вход: Time – время расчета (int); NumZone – номер зоны (int).</p> <p>Выход: массив (int) на 24 часа, в котором 0 для часов, не входящих в зону и не 0 для входящих.</p>
14	<u>_ZoneHalfHour</u> (Time, NumZone)	<p>Определение получасов, входящих в заданную зону ФОРЭМа. Для работы этой функции должен вестись энергетический календарь. Номера зон в таблице T_ZONE базы данных.</p> <p>Вход: Time – время расчета (int); NumZone – номер зоны (int).</p> <p>Выход: массив (int) на 48 получасов, в котором 0 для получасов, не входящих в зону и не 0 для входящих.</p>
15	<u>_CalcZone</u> (Time, NumZone, Id, Str)	<p>Расчет электроэнергии для заданной зоны по часам. Для работы этой функции должен вестись энергетический календарь. Номера зон в таблице T_ZONE базы данных.</p> <p>Вход: Time – время расчета; (int) NumZone – номер зоны (int); Id – идентификатор данного, для которого выполняется расчет (int); Str – имя таблицы архива с данным (str).</p> <p>Выход: значение электроэнергии (dbls).</p>

16	<code>_CalcZoneHalfHour (Time, NumZone, Id, Str)</code>	<p>Расчет электроэнергии для заданной зоны по получасам. Для работы этой функции должен вестись энергетический календарь. Номера зон в таблице T_ZONE базы данных.</p> <p>Вход: Time – время расчета (int); NumZone – номер зоны (int); Id – идентификатор данного, для которого выполняется расчет (int); Str – имя таблицы архива с данным (str).</p> <p>Выход: значение электроэнергии (dbls).</p>
17	<code>_ValHour(Time, Hour, Id, Str)</code>	<p>Чтение данного из любого архива за заданный час.</p> <p>Вход: Time – время расчета (int); Hour – час, за который читать данное (int); Id – идентификатор данного (int); Str – имя таблицы архива с данным (str).</p> <p>Выход: значение данного (dbls).</p>
18	<code>_ValRead(Time, Hour, Min, Sec, Id, Str)</code>	<p>Чтение данного за заданное время в пределах суток.</p> <p>Вход: Time – время расчета (int); Hour – час, за который читать данное (int); Min – минута, за которую читать данное (int); Sec – секунда, за которую читать данное (int); Id – идентификатор данного (int); Str – имя таблицы архива с данным (str).</p> <p>Выход: значение данного (dbls).</p>
19	<code>_IntgrVal(Time, Id, Str, Dim)</code>	<p>Суммирование данного за заданные интервалы с циклом 1 час.</p> <p>Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str); Dim – массив (int) с начальными и конечными часами интервалов.</p> <p>Выход: сумма (dbls).</p>
20	<code>_IntgrValHalfHour(Time, Id, Str, Dim)</code>	<p>Суммирование данного за заданные интервалы с циклом 30 минут.</p> <p>Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str); Dim – массив (int) с начальными и конечными часами и минутами интервалов.</p> <p>Выход: сумма (dbls).</p>
21	<code>_MiddlVal(Time, Id, Str, Dim)</code>	<p>Расчет среднего значения данного за заданные интервалы с циклом 1 час в пределах суток.</p> <p>Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str); Dim – массив (int) с начальными и конечными часами интервалов.</p> <p>Выход: сумма (dbls).</p>
22	<code>_MiddlValHalf(Time, Id, Str, Dim)</code>	<p>Расчет среднего значения данного за заданные интервалы с циклом 30 минут в пределах суток.</p> <p>Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str); Dim – массив (int) с начальными и конечными часами и минутами интервалов.</p> <p>Выход: сумма (dbls).</p>

23	<code>_MaxHour(Time, Id, Str)</code>	Определение часа максимума для заданной величины. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: час максимума (dbls).
24	<code>_MaxHalfHour(Time, Id, Str)</code>	Определение часа и получаса максимума для заданной величины. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: массив (dbls) из 2 значений: час и минуты максимума.
25	<code>_MinHour(Time, Id, Str)</code>	Определение часа минимума для заданной величины. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: час минимума (dbls).
26	<code>_MinHalfHour(Time, Id, Str)</code>	Определение часа и получаса минимума для заданной величины. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: массив (dbls) из 2 значений: час и минуты минимума .
27	<code>_EcsPlan(Time, Id, Str)</code>	Экстраполяция плановых данных в пределах часа. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: экстраполированное данное (dbls).
28	<code>_EcsPlanHalfHour(Time, Id, Str)</code>	Экстраполяция плановых данных в пределах получаса. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: экстраполированное данное (dbls).
29	<code>_MiddlPlanNext(Time, Id, Str)</code>	Осреднение плановых данных за час по формуле: Текущ. Час + следующ. Час / 2. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: результат (dbls).
30	<code>_MiddlPlan(Time, Id, Str)</code>	Осреднение плановых данных за час по формуле: Текущ. Час + предыдущ. Час / 2. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: результат (dbls).
31	<code>_MiddlPlanHalfHour(Time, Id, Str)</code>	Расчет средних получасовых значений диспетчерского графика из часовых значений. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: результат (dbls).
32	<code>_MiddlPlanHalfHourNext(Time, Id, Str)</code>	Осреднение плановых данных ведомости за получас. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: результат (dbls).

33	<code>_MiddlValMin(Time, Id, Str)</code>	Расчет среднего значения заданной величины за последнюю минуту. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: результат (dbls).
34	<code>_MiddlValHour(Time, Id, Str)</code>	Расчет среднего значения заданной величины с начала часа. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: результат (dbls).
35	<code>_MiddlValHalfHour(Time, Id, Str)</code>	Расчет среднего значения заданной величины с начала получаса. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: результат (dbls).
36	<code>_MiddlValNoTiHalfHour(Time, Id, Str)</code>	Расчет средних получасовых значений не телеизмеряемых параметров из часовых значений. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива (str). Выход: результат (dbls).
37	<code>_HourOrder(Num, Dim)</code>	Определение значений часов из интервалов по их сквозному порядковому номеру в интервалах. Вход: Num – порядковый номер в интервалах (int); Dim – массив (int) с начальными и конечными часами интервалов. Выход: результат (dbls).
38	<code>_ValHourOrder(Time, Id, Str, Num, Dim)</code>	Чтение данного за заданный час, определяемый по порядковому номеру в интервалах. Вход: Time – время расчета (int); Id – идентификатор данного (int); Str – имя таблицы архива с данным (str); Num – порядковый номер в интервалах (int); Dim – массив (int) с начальными и конечными часами интервалов. Выход: результат (dbls).
39	<code>_IntegrData(Time, Val, IdTune, Id, StrTemp, Str)</code>	Интегрирование данных за произвольный период с произвольным циклом с записью в базу. Используется только в подсистеме циклических расчетов. Подсистемой циклических расчетов в функцию, вызывающую в свою очередь функцию IntegrData, передается в качестве входного параметра идентификатор соответствующей настройки циклических расчетов (поле CALC_C_ID таблицы T_CALC_C). Из настройки используются данные по периоду и циклу расчета. Вход: Time – время расчета (int); Val – значение данного для интегрирования (ints или (dbls)); IdTune – идентификатор соответствующей настройки циклических расчетов (int); Id – идентификатор результата в архиве (int); StrTemp – имя таблицы архива с накоплением данного за период расчета с заданным циклом (str).

		<p>Str – имя таблицы архива с результатом интегрирования (str). Выход: нет.</p>
40	_IntegrDataAfterPer(Time, IdSource, StrSource, IdTune, Id, StrTemp, Str)	<p>Суммирование данных за произвольный период с произвольным циклом после завершения периода с записью в базу. Используется только в подсистеме циклических расчетов. Подсистемой циклических расчетов в функцию, вызывающую в свою очередь функцию IntegrDataAfterPer, передается в качестве входного параметра идентификатор соответствующей настройки циклических расчетов (поле CALC_C_ID таблицы T_CALC_C). Из настройки используются данные по периоду и циклу расчета. Вход: Time – время расчета (int); IdSource – идентификатор в базе данного для интегрирования (int); StrSource – имя таблицы архива с данным для интегрирования (str); IdTune – идентификатор соответствующей настройки циклических расчетов (int); Id – идентификатор результата в архиве (int); StrTemp – имя таблицы архива с накоплением данного за период расчета с заданным циклом (str). Str – имя таблицы архива с результатом интегрирования (str). Выход: нет.</p>
41	_VerifyEndPer(Time, Id, StrTemp, IdTune)	<p>Проверка завершения периода циклического расчета. Используется только в подсистеме циклических расчетов. Подсистемой циклических расчетов в функцию, вызывающую в свою очередь функцию VerifyEndPer, передается в качестве входного параметра идентификатор соответствующей настройки циклических расчетов (поле CALC_C_ID таблицы T_CALC_C). Из настройки используются данные по периоду и циклу расчета. Вход: Time – время расчета (int); Id – идентификатор параметра, для которого проверяется завершение периода расчета, в базе (int); StrTemp – имя служебной таблицы архива для настройки циклических расчетов (str); IdTune – идентификатор соответствующей настройки циклических расчетов (int). Выход: результат (int) проверки в Result и если значение равно 0, то период не завершен.</p>
42	_AkkumDataInCycle(Time, Val, IdTune, Id, StrTemp)	<p>Циклическое накопление данного. Используется только в подсистеме циклических расчетов. Подсистемой циклических расчетов в функцию, вызывающую в свою очередь функцию _AkkumDataInCycle, передается в качестве входного параметра идентификатор соответствующей настройки циклических расчетов (поле CALC_C_ID таблицы T_CALC_C). Из настройки используются данные по периоду и циклу расчета. Вход: Time – время расчета (int); Val – значение данного для накопления (ints или dbls); IdTune – идентификатор соответствующей настройки циклических расчетов (int);</p>

		<p>Id – идентификатор результата накопления в архиве (int);</p> <p>StrTemp – имя таблицы архива с накоплением данного за период расчета с заданным циклом (str).</p> <p>Выход: нет.</p>
43	_AkkumDataAfterPer(Time, IdSource, StrSource, IdTune, Id, StrTemp)	<p>Накопление данного после завершения периода расчета. Используется только в подсистеме циклических расчетов. Подсистемой циклических расчетов в функцию, вызывающую в свою очередь функцию _AkkumDataAfterPer, передается в качестве входного параметра идентификатор соответствующей настройки циклических расчетов (поле CALC_C_ID таблицы T_CALC_C). Из настройки используются данные по периоду и циклу расчета.</p> <p>Вход: Time – время расчета (int);</p> <p>IdSource – идентификатор данного для накопления (int);</p> <p>StrSource – имя архива с данным для накопления (str);</p> <p>IdTune – идентификатор соответствующей настройки циклических расчетов (int);</p> <p>Id – идентификатор результата накопления в архиве (int);</p> <p>StrTemp – имя таблицы архива с накоплением данного за период расчета с заданным циклом (str).</p> <p>Выход: нет.</p>
44	_ReadAkkumData(Time, Id, StrTemp)	<p>Чтение накопленного значения данного из таблицы архива с накоплениями данного (поле VAL). Используется только в подсистеме циклических расчетов.</p> <p>Вход: Time – время расчета (int);</p> <p>Id – идентификатор результата накопления в архиве (int);</p> <p>StrTemp – имя таблицы архива с накоплением данного за период расчета с заданным циклом (str).</p> <p>Выход: результат (dbls).</p>
45	_ReadAkkumCount(Time, Id, StrTemp)	<p>Чтение накопленного значения поля счетчика суммирований из таблицы архива с накоплениями данного (поле COUNT). Используется только в подсистеме циклических расчетов.</p> <p>Вход: Time – время расчета (int);</p> <p>Id – идентификатор результата накопления в архиве (int);</p> <p>StrTemp – имя таблицы архива с накоплением данного за период расчета с заданным циклом (str).</p> <p>Выход: результат (ints).</p>
46	_ClearAkkumDataAndCount(Time, Id, StrTemp)	<p>Сброс значений полей накоплений и счетчика суммирований из таблицы архива с накоплениями данного (поля VAL и COUNT). Используется только в подсистеме циклических расчетов.</p> <p>Вход: Time – время расчета (int);</p> <p>Id – идентификатор результата накопления в архиве (int);</p> <p>StrTemp – имя таблицы архива с накоплением данного за период расчета с заданным циклом (str).</p> <p>Выход: нет.</p>

47	<code>_BackPerTime(Time, IdTune)</code>	<p>Расчет времени на период назад относительно заданного. Используется только в подсистеме циклических расчетов. Подсистемой циклических расчетов в функцию, вызывающую в свою очередь функцию <code>_BackPerTime</code>, передается в качестве входного параметра идентификатор соответствующей настройки циклических расчетов (поле <code>CALC_C_ID</code> таблицы <code>T_CALC_C</code>). Из настройки используются данные по периоду и циклу расчета.</p> <p>Вход: <code>Time</code> – время расчета (int); <code>IdTune</code> – идентификатор соответствующей настройки циклических расчетов (int).</p> <p>Выход: результат (int).</p>
48	<code>_WriteCutArch(Time, ArchSrc, ArchDst)</code>	<p>Запись срезов данных из одного архива в другой. НСИ для архивов совпадают.</p> <p>Вход: <code>Time</code> - время записи (int); <code>ArchSrc</code> - имя таблицы архива, из которого писать (str); <code>ArchDst</code>- имя таблицы архива, в который писать (str).</p> <p>Выход: -.</p>
49	<code>_FillArchByRefData(Time, TblNci, ArchDst)</code>	<p>Запись данных архива при помощи таблицы НСИ. ID записываемых данных совпадает с ID записи в таблице НСИ. В таблице НСИ должны быть поля с наименованиями, в которых должны быть следующие подстроки: “_SIGN_ARCH” – признак данного-архива; “_REC_ID” – идентификатор данного в архиве-источнике; “_OBJ_ID” – идентификатор архива-источника (ссылка на <code>T_ARCH</code>).</p> <p>Вход: <code>Time</code> – время записи (int); <code>TblNci</code> - имя таблицы НСИ (str); <code>ArchDst</code> - имя таблицы архива, в который писать (str).</p> <p>Выход: -.</p>
50	<code>_WorkCbiIn(Time, NumSet, StrMak)</code>	<p>Обработка принятого макета ЦБИ. Определяется номер описателя макета (<code>T_CBI</code>). Затем из описателя определяется номер в <code>T_CALC</code> функции, обрабатывающей данный макет и эта функция запускается на выполнение. В запускаемую функцию передается 3 параметра: время приема (int); номер описателя макета (int); строка с макетом (int).</p> <p>Вход: <code>Time</code> – время приема (int); <code>NumSet</code> – номер набора (int); <code>StrMak</code> – строка с макетом (str).</p> <p>Выход: -.</p>
51	<code>_WorkCbiOut(Time, NumSet)</code>	<p>Обработка принятого запроса на макет ведомости с информацией.</p> <p>Определяется номер описателя макета (<code>T_CBI</code>). Затем из описателя определяется номер в <code>T_CALC</code> функции, обрабатывающей данный макет и эта функция запускается на выполнение. В запускаемую функцию передается 3 параметра: время запроса (int);</p>

		номер описателя макета (int); номер набора (int0. Вход: Time – время приема (int); NumSet – номер набора (int). Выход: -.
52	_ WorkRecIn (Time, NumSet)	Обработка квитанции на переданный макет. Вход: Time – время из квитанции (int); NumSet – номер набора (int). Выход: -.
53	_WriteShortDataCbiToArch(Ti me, IdDescrMak, StrMak, LenHeader, DispBitMass)	Запись данных типа short из принятого макета ЦБИ в архивы с переворотом байтов. Вход: Time – время приема (int); IdDescrMak – идентификатор описателя макета в T_CBI (int); StrMak – строка с макетом (str); LenHeader – длина заголовка в строке с макетом (int); DispBitMass – смещение битового массива наличия данных в макете, если он есть, иначе 0 (int).
54	_WriteLongtDataCbiToArch(Ti me, IdDescrMak, StrMak, LenHeader, DispBitMass)	Запись данных типа long из принятого макета ЦБИ в архивы с переворотом байтов. Вход: Time – время приема (int); IdDescrMak – идентификатор описателя макета в T_CBI (int); StrMak – строка с макетом (str); LenHeader – длина заголовка в строке с макетом (int); DispBitMass – смещение битового массива наличия данных в макете, если он есть, иначе 0 (int).
55	_WriteRecForTrans(IdDescrMak, StrKwit)	Запись на передачу сформированной квитанции о приеме макета. Вход: IdDescrMak – идентификатор описателя макета в T_CBI (int); StrKwit – строка с квитанцией (str).
56	_ReadShortDataCbiFromArch (Time, IdDescrMak, StrHeader, LenHeader, DispBitMass)	Чтение данных из архивов с записью в передаваемый макет ЦБИ в виде short (с переворотом байт). Вход: Time – время чтения (int); IdDescrMak – идентификатор описателя макета в T_CBI (int); StrHeader – строка с заголовком (str); LenHeader – длина строки с заголовком в байтах (int); DispBitMass – смещение битового массива наличия данных в макете, если он есть, иначе 0 (int). Выход: строка с макетом (str).
57	_ReadLongDataCbiFromArch (Time, IdDescrMak, StrHeader, LenHeader, DispBitMass)	Чтение данных из архивов с записью в передаваемый макет ЦБИ в виде long (с переворотом байт). Вход: Time – время чтения (int); IdDescrMak – идентификатор описателя макета в T_CBI (int); StrHeader – строка с заголовком (str); LenHeader – длина строки с заголовком в байтах (int); DispBitMass – смещение битового массива наличия данных в макете, если он есть, иначе 0 (int). Выход: строка с макетом (str).

58	<code>_WriteMakForTrans(IdDescrMak, StrMak)</code>	Запись на передачу сформированного макета. Вход: IdDescrMak – идентификатор описателя макета в T_CBI (int); StrMak – строка с макетом (str). Выход: -.
59	<code>_ArrIdDataForArch (Time, StrArch)</code>	Формирование массива идентификаторов данных для среза архива. Вход: Time - время запроса (int); StrArch - имя таблицы архива (str). Выход: - массив (int) идентификаторов данных.
60	<code>_ArrIdDataForArchMsec (Time, StrArch)</code>	Формирование массива идентификаторов данных для среза архива, поддерживающего миллисекунды. Вход: Time - время запроса (int); StrArch - имя таблицы архива (str). Выход: - массив (int) идентификаторов данных.
61	<code>_WriteDataMtxtToArch(Time, IdMakDsc, CodSt, NumSt, Discr, ArrData, ArrSign)</code>	Запись данных из принятого текстового макета в архивы. Вход: Time - время записи первого данного массива (int); IdMakDsc - ID описателя макета (int); CodSt - код строки (str); NumSt - номер в строке (int); Discr - дискретность записи в секундах (int); ArrData - массив с данными (double); ArrSign - массив с признаками (int: 0 - о'к или 1 - no). Выход: - если не 0, то ошибка.
62	<code>_WinToDosStr (Str)</code>	Перекодировка строки WIN -> DOS. Вход: Str – строка в WIN-кодировке. Выход: Строка в DOS-кодировке.
63	<code>_DosToWinStr (Str)</code>	Перекодировка строки DOS -> WIN. Вход: Str – строка в DOS-кодировке. Выход: Строка в WIN-кодировке.
64	<code>_SqlBegWork()</code>	Подготовка работы с SQL-базой. Вход: Нет. Выход: < 0 – ошибка, >= 0 – О'К;
65	<code>_SqlEndWork ()</code>	Завершение сеанса работы с SQL-базой. Вход: Нет. Выход: < 0 – ошибка, >= 0 – О'К;
66	<code>_SqlBlockBeg ()</code>	Начало блока работы с SQL-базой. Начало транзакции. Вход: Нет. Выход: < 0 – ошибка, >= 0 – О'К;

67	_SqlBlockEnd ()	Конец блока работы с SQL-базой. Конец транзакции. Вход: Нет. Выход: < 0 – ошибка, >= 0 – O'K;
68	_CalcSql (StrSql, Prm, CTbl)	Выполнение SQL-запроса SELECT. Вход: StrSql – строка с SQL-запросом; Prm – строка с параметрами запроса; CTbl – таблица, в которую возвращать результат. Таблица предварительно д.б. создана. Выход: < 0 – ошибка, >= 0 – O'K;
69	_CalcRowNew (TblName, StrPar)	Запрос на добавление записи в таблицу SQL-базы. Вход: TblName – строка с именем таблицы; StrName – строка с параметрами запроса; Выход: < 0 – ошибка, >= 0 – O'K;
70	_CalcRowEdit (TblName, StrPar)	Запрос на редактирование записи в таблице SQL-базы. Вход: TblName – строка с именем таблицы; StrName – строка с параметрами запроса (здесь должен быть указан ключ для поиска редактируемой записи в строке вида «KEY=»; Выход: < 0 – ошибка, >= 0 – O'K;
71	_CalcRowDel (TblName, StrPar)	Запрос на удаление записи из таблицы SQL-базы. Вход: TblName – строка с именем таблицы; StrName – строка с параметрами запроса (здесь должен быть указан ключ для поиска удаляемой записи в строке вида «KEY=»; Выход: < 0 – ошибка, >= 0 – O'K;
72	_CalcPost ()	Завершение текущего запроса. Вход: Нет. Выход: < 0 – ошибка, >= 0 – O'K;
73	_CalcFldIsNull (FieldName)	Установка значения поля в таблице SQL-базы в NULL. Вход: FieldName – строка с именем поля. Выход: < 0 – ошибка, >= 0 – O'K;
74	_CalcFldBool (FieldName, Value)	Запись значения в поле типа BOOL. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O'K;
75	_CalcFldByte (FieldName, Value)	Запись значения в поле типа BYTE. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O'K;
76	_CalcFldSmall (FieldName, Value)	Запись значения в поле типа SHORT. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O'K;

77	<code>_CalcFldInt (FieldName, Value)</code>	Запись значения в поле типа INT. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O’K;
78	<code>_CalcFldFloat (FieldName, Value)</code>	Запись значения в поле типа FLOAT. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O’K;
79	<code>_CalcFldDouble (FieldName, Value)</code>	Запись значения в поле типа DOUBLE. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O’K;
80	<code>_CalcFldCurrency (FieldName, Value)</code>	Запись значения в поле типа CURRENCY. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O’K;
81	<code>_CalcFldUnixDT (FieldName, Value)</code>	Запись значения в поле типа UNIXDT. На самом деле происходит запись времени в формате UNIX в поле типа INTEGER. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O’K;
82	<code>_CalcFldUnixMDT (FieldName, Value)</code>	Запись значения в поле типа UNIXMDT. На самом деле происходит запись времени с миллисекундами в поле типа DOUBLE. Вход: FieldName – строка с именем поля; Value – значение. Выход: < 0 – ошибка, >= 0 – O’K;
83	<code>_CalcFldText (FieldName, Value)</code>	Запись значения в поле типа TEXT или MEMO. Вход: FieldName – строка с именем поля; Value – строка. Выход: < 0 – ошибка, >= 0 – O’K;
84	<code>_CalcFldBlob (FieldName, Value)</code>	Запись значения в поле типа BLOB. Вход: FieldName – строка с именем поля; Value – строка. Выход: < 0 – ошибка, >= 0 – O’K;
85	<code>_CalcExec (SqlStr)</code>	Выполнение SQL – запроса. Следует иметь в виду, что изменения в базе, выполненные с помощью этой функции не фиксируются в событиях изменения НСИ. Также не возвращаются результаты выполнения, кроме кода ошибки. Поэтому пользоваться этой функцией надо осторожно. Вход: SqlStr – строка с SQL-запросом (STRING). Выход: -1 - ошибка, >=0 - нет.

86	<code>_WaitForMultipleObjects (HandleArr, TimeOut)</code>	Ожидание срабатывания одного из синхронизирующих объектов (событие, процесс, поток). Вход: HandleName – массив HANDLE синхронизирующих объектов, срабатывание одного из которых ожидается (тип ARR); TimeOut – величина времени ожидания срабатывания в миллисекундах (тип INT). Выход: -1 - ошибка, >=0 - индекс сработавшего объекта, 255 – превышено время ожидания.
87	<code>_WaitForSingleObjects (Handle, TimeOut)</code>	Ожидание срабатывания синхронизирующего объекта (событие, процесс, поток). Вход: Handle – HANDLE синхронизирующего объекта, срабатывание которого ожидается (тип INT); TimeOut – величина времени ожидания срабатывания в миллисекундах (тип INT). Выход: -1 - ошибка, ==0 - сработал объект, 255 – превышено время ожидания.
88	<code>_CreateProcess (ComStr, DirStr, WinHide[, Prior])</code>	Создание процесса (запуск внешней программы). Вход: ComStr – командная строка запуска процесса (тип STRING); DirStr – директорий, с которым работает создаваемый процесс (тип STRING); WinHide – скрывать или минимизировать окно запускаемого процесса. 0 – скрывать, != 0 – минимизировать (тип INT); Prior – необязательный параметр. Если есть, то = 0, то процесс будет запущен с нормальным приоритетом (NORMAL_PRIORITY_CLASS), если != 0, то с низким (IDLE_PRIORITY_CLASS). Тип INT. Выход: -1 - ошибка, >=0 - HANDLE процесса.
89	<code>_GetExitCodeProcess (Handle)</code>	Определение кода завершения ранее запущенного процесса. Вход: Handle - HANDLE процесса. Тип INT. Выход: код завершения процесса. Тип INT.
90	<code>_CloseHandle (Handle)</code>	Закрытие описателя (HANDLE) объекта (процесса, потока, события). Вход: Handle – HANDLE. Тип INT. Выход: -1 – ошибка.
91	<code>_Sleep (Time)</code>	Приостановка выполнения (задержка). Вход: Time – время приостановки в миллисекундах. Тип INT. Выход: нет.
92	<code>_ReadFldVal (FieldName)</code>	Чтение значения конкретного поля из полей события. Применяется в обработчиках событий (функция, имя которой указано в поле EV_COD_NAME_FUNC в таблице T_EV_COD). Вход: FieldName – имя поля. Тип STRING. Выход: Значение поля. Нужно обязательно контролировать тип возвращаемого значения. Например, поле EV_VALUE в

		событии м.б. любого допустимого типа. Если EMPTY, то или ошибка или нет поля.
93	_StatusSrv ()	Запрос статуса своего сервера. Вход: нет. Выход: статус (1 – основной, 2 – резервный, 0 – неопределено)
94	_TypeSrv ()	Запрос типа сервера. Вход: нет. Выход: тип (номера битов: 0x01 - ввод-вывод, 0x02 - диалог, 0x04 - расчет, 0x08 - событий).
95	_NameSrv ()	Запрос имени сервера. Вход: нет. Выход: строка с именем сервера.
96	_ResSrvStat ()	Запрос статуса резервного сервера. Вход: нет. Выход: статус резервного сервера: 0 - не запущен, 1 - основной, 2 - резервный, (-1) – нет.
97	_CreateEvent (EventName)	Создание флага события. Вход: EventName – имя флага события, если пустое, то создается флаг без имени. Тип STRING. Выход: -1 - ошибка, >0 – HANDLE (описатель) флага.
98	_FileOpen (FileName, TypeAccess)	Открытие файла. Вход: FileName – имя файла. Тип STRING. TypeAcces – тип доступа к файлу (STRING). Открывается файл с помощью функции fopen языка С и типы доступа см. в описании этой функции. Выход: 0 - ошибка, > 0 - указатель на открытый файл. Указатель в смысле указателя, возвращаемого функцией fopen. Тип INT.
99	_FileClose (FILE)	Закрытие файла. Вход: FILE – указатель на открытый файл. Выход: -1 - ошибка, >= 0 – нормально.
100	_FileDelete (FilePath)	Удаление файла. Вход: FilePath – путь к удаляемому файлу (STRING). Выход: 0 - нормально, -1 – ошибка.
101	_FileLen (FILE)	Определение длины открытого файла. Вход: FILE – указатель на открытый файл (INT). Выход: -1 - ошибка, >= 0 - длина файла.
102	_FileSeek (FILE, Offset)	Перемещение указателя на заданную позицию в файле относительно его начала. Вход: FILE – указатель на открытый файл (INT); Offset – на сколько байт переместиться (INT). Выход: -1 - ошибка, >= 0 - нормально.

103	_FileRead (FILE, Byte)	Чтение из файла заданного количества байт. Вход: FILE – указатель на открытый файл (INT). Byte – сколько байт прочитать. Выход: -1 и TYPE(Result) != 3- ошибка, TYPE (Result) = 3 (STRING) – строка с данными.
104	_FileWrite (FILE, Buff)	Запись в файл строки с данными. У нас под строкой понимается и последовательность байт. Поэтому можно писать все в соответствии с типом доступа при открытии файла. FILE – указатель на открытый файл (INT). Buff – строка с записываемыми данными. Выход: -1 - ошибка, >= 0 - нормально.
105	_GblVarSet (GblName, Value)	Установка значения глобальной переменной. Глобальная переменная остается доступной даже по завершении создавшей ее функции языка TScript. Если переменной с таким именем нет, то она будет создана, если есть, то ее значение будет заменено. Вход: GblName – имя глобальной переменной (STRING); Value – значение глобальной переменной (любой тип языка TScript). Выход: -1 - ошибка, >= 0 – нормально.
106	_GblVarGet (GblName)	Получение значения глобальной переменной. Вход: GblName – имя глобальной переменной (STRING). Выход: Если тип возвращаемого значения = EMPTY – ошибка.
107	_GblVarDel (GblName)	Удаление глобальной переменной. Вход: GblName – имя глобальной переменной (STRING). Выход: -1 - ошибка, >= 0 - нормально.
108	_ThreadFun (FunName, GblName, SleepTime, GblNameEnd)	Создание потока для выполнения функции языка TScript. Вход: FunName – имя запускаемой функции; GblName – имя глобальной переменной, передаваемой в функцию; SleepTime – величина задержки запуска функции в секундах или 0; GblNameEnd - имя глобальной переменной завершения потока без запуска функции (или пусто). Если имя не пустое, то завершение потока в случае отсутствия переменной с таким именем или не нулевое состояние переменной. Выход: -1 - ошибка, >=0 - HANDLE потока.
109	_UnixTime ()	Получение времени в формате UNIX. Вход: нет. Выход: время в секундах (INT).
110	_SetEvent (Handle)	Установка флага события. Вход: Handle – описатель (HANDLE) флага события. Выход: -1 - ошибка, >=0 – нет.

111	<code>_TerminateProcess (Handle)</code>	Принудительное завершение процесса. Вход: Handle – описатель (HANDLE) процесса. Выход: -1 - ошибка, ≥ 0 - нет.
112	<code>_TerminateThread (Handle)</code>	Принудительное завершение потока. Вход: Handle – описатель (HANDLE) потока. Выход: -1 - ошибка, ≥ 0 - нет.
113	<code>_WriteLogFile (Str)</code>	Запись строки в log – файл сервера. Вход: Str – строка для записи (STRING). Выход: нет.
114	<code>_GetIsDst (Time)</code>	Определение летнего/зимнего времени. Вход: Time – текущее время в формате UNIX. Выход: 0 - зима; 1 - лето; -1 – ошибка.
115	<code>_GetChangeDst (Time)</code>	Определение факта смены лето/зима или зима/лето в данных сутках. Вход: Time – текущее время в формате UNIX. Выход: 0 - нет перехода; 1 - лето/зима; 2 - зима/лето; -1 - ошибка..

ПРИЛОЖЕНИЕ 6

1 ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ ВНЕШНИХ ФУНКЦИЙ ЯЗЫКА TScript.

1.1 Функция _ArchCurr.

Определение архива, для которого производится расчет. Появляется возможность использовать одну формулу для нескольких архивов. Например, ТИ копируются один в один в архивы часовой ведомости, получасовой ведомости и т.д. Поэтому по одной формуле можно выполнять расчеты для этих архивов, но нужно определить, для какого архива производится расчет и подставить нужные данные. Например, функция доступа для ТИ.

```

TI(pId,pTime)
{
    //мгновенные
    if(CMP(_ArchCurr(), "T_ARCH_PTI")==0) // если расчет для архива ПТИ
    {
        Result = _ReadData(pId,pTime,"T_ARCH_TI"); // то данные читать из архива ТИ
    }
    //10 минутные срезы
    elseif(CMP(_ArchCurr(), "T_ARCH_PTI_10MIN")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_TI_10MIN");
    }
    //получасовые средние
    elseif(CMP(_ArchCurr(), "T_ARCH_V30_PTI")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_V30_TI");
    }
    //часовые средние
    elseif(CMP(_ArchCurr(), "T_ARCH_V60_PTI")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_V60_TI");
    }
    //получасовые срезы
    elseif(CMP(_ArchCurr(), "T_ARCH_30M_PTI")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_30M_TI");
    }
    //часовые срезы
    elseif(CMP(_ArchCurr(), "T_ARCH_V_H_PTI")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_V_H_TI");
    }
    //суточные максимумы
    elseif(CMP(_ArchCurr(), "T_ARCH_DMAX_PTI")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_DMAX_TI");
    }
    //суточные минимумы
    elseif(CMP(_ArchCurr(), "T_ARCH_DMIN_PTI")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_DMIN_TI");
    }
    //суточные средние
    elseif(CMP(_ArchCurr(), "T_ARCH_D_PTI")==0)
    {
        Result = _ReadData(pId,pTime,"T_ARCH_D_TI");
    }
    //суточная выработка расчет по ПТИ

```

```

elseif(CMP(_ArchCurr(), "T_ARCH_D_POWER_PTI")==0)
{
    Result = _ReadData(pId,pTime,"T_ARCH_D_POWER_TI");
}
else
{
    Result = _ReadData(pId,pTime,"T_ARCH_TI");
}
}

```

1.2 Функции _GblVarGet, _GblVarSet.

Использование глобальных переменных.

Например, необходимо узнать в расчете псевдо-ТС предыдущее значение ПТС и одного из ТС. Если значение ПТС не изменилось, то завершить функцию. В противном случае, если изменилось значение ТС 1 и он имеет состояние «вкл», то выставить в ПТС флаг 14, если он в состоянии «откл», то флаг 15.

Есть расчет псевдо-ТС.

```

PTS1(pTime1)
{
    var Str1, Str2, Arr1, pTime, WorkVal,
        Val_PTS, Val_PTS_Old, SignPts,
        Val_TS1, Val_TS1_Old,

    pTime =0;

    // Например формула следующая

    Result = TS (1, pTime) | TS (2, pTime);
    Val_PTS = val (Result); // выделяем из результата значение ПТС
    SignPts = sign (Result); // выделяем из результата флаги ПТС

    // Определяем предыдущее значение ПТС 1
    Arr1 = Arr (0); // формирование имени глобальной переменной, в которой хранится
    предыдущее // значение ПТС
    Str2 = "PTS_IGOR_";
    Arr1 = Arr1 + Str2;
    Arr1 = Arr1 + 1; // ID ПТС
    Str1 = printf("%s%d", Arr1); // сформировали имя глобальной переменной, в которой
    хранится старое // значение ПТС. В данном случае это PTS_IGOR_1.
    if (type (_GblVarGet (Str1)) == 0) // проверка наличия переменной со старым значение
    ПТС
    {
        Val_PTS_Old = Val_PTS; // если еще нет (первый запуск), то приравниваем значение к
        текущему
    }
    else
        Val_PTS_Old = _GblVarGet (Str1); // чтение предыдущего значения ПТС из переменной
        _GblVarSet (Str1, Val_PTS); // записываем в переменную текущее значение (на след.
        цикле будет // использоваться как старое)
    // если глобальной переменной с этим именем нет, то она автоматически создается

    if (Val_PTS == Val_PTS_Old) // если значение ПТС не изменилось, то завершить расчет.
    exit;

    // Чтение текущего значения ТС с ID = 1

    WorkVal = TS(1,pTime);
    Val_TS1 = val (WorkVal); // выделение из результата значения ТС

```

```

// Чтение предыдущего значения ТС с ID = 1

Arr1 = Arr (0);
Str2 = "TS_IGOR_";
Arr1 = Arr1 + Str2;
Arr1 = Arr1 + 1; // ID ТС
Str1 = printf("%s%d", Arr1); // сформировано имя глобальной переменной TS_IGOR_1

if (type (_GblVarGet (Str1)) == 0) // если тип возвращаемого значения пустой, то
значение еще не было
                                // установлено
{
    Val_TS1_Old = Val_TS1; // текущее значение равно прошлому
}
else
    Val_TS1_Old = _GblVarGet (Str1);
    _GblVarSet (Str1, Val_TS1);
    if ((Val_TS1 != Val_TS1_Old) && (Val_TS1 != 0)) // если значение изменилось и ТС в
сост. «ВКЛ»
    {
        // Т.е. изменилось значение 1 ТС и он в состоянии "ВКЛ"
        Result = ints (Val_PTS, SignPts | 0x4000); // т.е будет ПТС с установленным 14 флагом
        exit;
    }
    if ((Val_TS1 != Val_TS1_Old) && (Val_TS1 == 0))
    {
        // Т.е. изменилось значение 1 ТС и он в состоянии "ВЫКЛ"
        Result = ints (Val_PTS, SignPts | 0x8000); // т.е будет ПТС с установленным 15 флагом
        exit;
    }
}

```

1.3 Пример работы с базой.

Необходимо по Вашей таблице в SQL-базе выполнить какую-то необходимую Вам обработку.

```

// Предполагаем, что выполняется циклический расчет.
// Есть таблица НСИ с именем T_NCI со следующими полями:
// - NCI_ID - идентификатор записи;
// - NCI_OBJ_ID - идентификатор в T_OBJ таблицы с архивом-источником;
// - NCI_REC_ID - идентификатор данного в архиве-источнике;
// - NCI_SIGN - необходимый Вам признак.

Altai(pTime, IdTune, IdPar)
{
    var i, Tbl_T_NCI, Tbl_T_OBJ, NCI_ID, NCI_OBJ_ID, NCI_REC_ID, NCI_SIGN, Arr1, Str1,
        RecCount, RecCount1, ArchName, Rec;

    Tbl_T_NCI = CTblNew();
    Tbl_T_OBJ = CTblNew();

    // Чтение таблицы с НСИ

    _SqlBegWork ();
    _SqlBlockBeg ();
    _CalcSql ("Select * from T_NCI", "", Tbl_T_NCI);
    _CalcPost ();
    _SqlBlockEnd ();
    _SqlEndWork ();

    RecCount = CTblRecCount(Tbl_T_NCI); // Количество строк в T_NCI

```

```

//_WriteLogFile (Str (RecCount)); // если раскомментарите, то количество записей в log - файле
сервера

    i = 0;
    while (i != RecCount)
    {
        Rec = CTblRec(Tbl_T_NCI, 0, i); // чтение записей таблицы T_NCI
        NCI_ID = CRecFldGet(Rec, "NCI_ID"); // идентификатор записи (в примере совпадает с
IdPar)
        NCI_OBJ_ID = CRecFldGet(Rec, "NCI_OBJ_ID"); // идентификатор таблицы архива-
источника в T_OBJ
        NCI_REC_ID = CRecFldGet(Rec, "NCI_REC_ID"); // идентификатор данного в архиве-
источнике
        NCI_SIGN = CRecFldGet(Rec, "NCI_SIGN"); // необходимый Вам признак
//_WriteLogFile (Str (NCI_SIGN)); // если раскомментарите, то признак в log - файле сервера

// Определение имени таблицы с архивом по ее идентификатору в T_OBJ

        CTblClearAll(Tbl_T_OBJ); // очистка таблицы от прошлого запроса
        _SqlBegWork ();
        _SqlBlockBeg ();
        Arr1 = Arr (0);
        Arr1 = Arr1 + NCI_OBJ_ID;
        Str1 = printf("Select OBJ_NAME_LAT from T_OBJ where OBJ_ID=%d", Arr1);
        _CalcSql (Str1, "", Tbl_T_OBJ);
        _CalcPost ();

        _SqlBlockEnd ();
        _SqlEndWork ();

        RecCount1 = CTblRecCount(Tbl_T_OBJ); // Количество строк в результате запроса
        if (RecCount1 != 1) // должна быть одна строка
        {
            i = i + 1;
            continue;
        }
        Rec = CTblRec(Tbl_T_OBJ, 0, 0); // чтение записи

        ArchName = CRecFldGet(Rec, "OBJ_NAME_LAT"); // имя таблицы с архивом
//_WriteLogFile (ArchName); // если раскомментарите, то имя архива в log - файле сервера

// Далее необходимая Вам обработка. Например...

        if (NCI_SIGN != 0)
        {
            Result = _ReadData(NCI_REC_ID, pTime, ArchName);
            _IntegrData(pTime, Result, IdTune, IdPar, "T_ARCH_TEMPd", "T_ARCH_INTGD");
        }

        i = i + 1;
    }
}

```

ПРИЛОЖЕНИЕ 7

1 ФЛАГИ ДЛЯ АРХИВНЫХ ЗНАЧЕНИЙ

1.1 Флаги для аналоговых величин.

№ флага	Описание флага.
0	Недостоверность параметра (общая).
1	Недостоверное кодовое значение. Используется для телеметрии.
2	Недостоверность по скорости изменения параметра. Используется для телеметрии.
3	Недостоверность по обновлению. Используется для телеметрии.
4	Недостоверность по выключению УТМ. Используется для телеметрии.
5	Недостоверность из-за отказа УТМ. Используется для телеметрии.
6	Недостоверность из-за вывода параметра из обработки.
7	Коррекция значения параметра оператором.
8	Нарушение нижнего технологического предела.
9	Нарушение верхнего технологического предела.
10	Нарушение нижнего аварийного предела.
11	Нарушение верхнего аварийного предела.
12	Недопустимое отклонение величины параметра от плана.
13	Значение в допустимом диапазоне отклонения от дубля. Используется для телеметрии.
14	Устойчивый скачок. Используется для телеметрии.
15	Замена дублирующим значением (общий). Используется для телеметрии.
16	Замена дублирующим значением по дубли 1. Используется для телеметрии.
17	Замена дублирующим значением по дубли 2. Используется для телеметрии.
18	Замена дублирующим значением по дубли 3. Используется для телеметрии.
19	Замена на значение, введенное вручную. Ручной ввод параметра.
20	Автоматическая замена значением дубля
21	Недостоверность по ТС.
22	Недостоверность по энергообъекту.
23	Неактуальное данное (до первого приема).
24	Ручное управление с верхнего уровня.
25	Недостоверность (ММО) (межмашинный обмен по FDST, МЭК 870-5-101, МЭК 870-5-104).
26	Ручное управление (ММО).
27	Замена резервным значением (ММО).
28	Нарушение предела (ММО).
29	Недостоверность из-за вывода из обработки (ММО).
30	Неактуальное данное (ММО).

1.2 Флаги для дискретных величин

№ флага	Описание флага.
0	Недостоверность параметра.
1	Недостоверность по выключению УТМ. Используется для телеметрии.
2	Недостоверность из-за отказа УТМ. Используется для телеметрии.
3	Замена на значение введенное вручную.
4	Недостоверность из-за вывода параметра из обработки.
5	Недостоверность по ТС.
6	Недостоверность по энергообъекту.
7	Отклонение от нормального состояния.
8	Неактуальное значение (до первого приема).
9	Ручное управление с верхнего уровня.
10	Недостоверность (ММО).
11	Ручное управление (ММО).
12	Недостоверность из-за вывода из обработки (ММО).
13	Неактуальное данное (ММО).






















№	Наименование	Цвет	Символ
0	● Недостоверность ТИ	 Розовый	?
1	● Недостоверное кодовое значение	 Розовый	?
2	● Недостоверность по скорости	 Розовый	?
3	● Недостоверность по необновлению	 Розовый	?
4	● УТМ выключено	 Темно-синий	У
5	● Неисправность УТМ	 Темно-синий	У
6	● ТИ выведено из обработки	 Розовый	?
7	● Коррекция ТИ оператором	 Зеленый	К
8	● Нарушение нижн.технолог. предела	 Сиреневый	Т
9	● Нарушение верх.технолог. предела	 Сиреневый	Т
10	● Нарушение нижн.авар. предела	 Красный	А
11	● Нарушение верх.авар.предела.	 Красный	А
12	● Недопустимое отклонение от плана	 Красный	П
13	● Значение в допустимом диапазоне отклонения от дубля	 Оливковый	!
14	● Устойчивый скачок	 Желтый	С
15	● Замена значением дубля	 Бирюзовый	Д
16	● Замена значением Дубля 1	 Бирюзовый	Д
17	● Замена значением Дубля 2	 Бирюзовый	Д
18	● Замена значением Дубля 3	 Бирюзовый	Д
19	● Ручной ввод	 Малиновый	Р
20	● Автоматическая замена значением дубля	 Бирюзовый	Д

Рисунок 27 – Образец настройки флагов Архива ТИ






Значение: Вкл/Откл			
№	Наименование	Цвет	Символ
0	● Недостоверность ТС		Розовый ?
1	● УТМ выключено		Темно-сини У
2	● Неисправность УТМ		Темно-сини У
3	● Ручной ввод		Малиновый Р
4	● ТС выведен из обработки		Розовый ?
5			

Рисунок 28 – Образец настройки флагов Архива ТС






















Значение: Аналоговое			
№	Наименование	Цвет	Символ
0	● Недостоверность		Розовый ?
1	● Недост. код знач. одного из параметров		Розовый ?
2	● Недост. по скорости измен. одного из параметров		Розовый ?
3	● Недост. по необновлению одного из параметров		Розовый ?
4	● Выключено УТМ одного из параметров		Темно-синий
5	● Неисправность УТМ одного из параметров		Темно-синий
6	● Недост. из-за вывода из обработки одного из параметров		Розовый ?
7	● Коррекция значения оператором		Зеленый К
8	● Нарушение нижн.технолог. предела		Сиреневый Т
9	● Нарушение верх.технолог. предела		Сиреневый Т
10	● Нарушение нижн.авар. предела		Красный А
11	● Нарушение верх.авар. предела		Красный А
12	● Недопустимое отклонение от плана		Красный П
13	● Значение в допустимом диапазоне отклонения от дубля		Оливковый !
14	● Устойчивый скачок одного из параметров		Желтый
15	● Замена значением дубля		Бирюзовый Д
16	● Замена значением Дубля 1		Бирюзовый Д
17	● Замена значением Дубля 2		Бирюзовый Д
18	● Замена значением Дубля 3		Бирюзовый Д
19	● Один из парам. на Ручном вводе		Малиновый
20	● Автоматическая замена значением дубля		Бирюзовый Д

Рисунок 29 – Образец настройки флагов Архива ПТИ