УТВЕРЖДЕН
ЛЯЮИ.00673-01 90 01 ЛУ

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

«ВИЗУАЛИЗАЦИЯ ТЕХНОЛОГИЧЕСКОГО ПРОЦЕССА SCADA-ЭЛЬБРУС»

Руководство по формированию и ведению БД

ЛЯЮИ.00673-01 90 01

Листов 18

Перв. Применяемость
ЛЯЮИ.00673-01

Литера

Инв. № подп.	Подп. и дата	Взам.инв. №	Инв. № дубл.	Подп. и дата

АННОТАЦИЯ

Данный документ содержит сведения о формировании и ведении базы данных для работы (ПО) «Визуализация технологического процесса SCADA (Supervisory Control And Data Acquisition — диспетчерское управление и сбор данных) Эльбрус» (далее «Визуализация»)

СОДЕРЖАНИЕ

1. Правила подготовки данных 4
1.1. Порядок отбора информации для включения в базу данных 4
1.1 Правила подготовки и кодирования информации 5
1.2 Формы представления информации 6
1.3 Правила заполнения форм 6
1.4 Порядок внесения изменений информации 6
2. Порядок и средства заполнения базы данных 7
2.1. Состав технических средств 7
2.1 Правила, порядок, последовательность и описание процедур, используемых при заполнении базы данных 7
2.2. Перенос данных на машинные носители информации 10
3. Процедуры изменения и контроля базы данных 11
3.1. Состав и последовательность выполнения процедур по контролю и изменению содержания базы данных 11
4. Порядок и средства восстановления базы данных 12
4.1. Описание средств защиты базы от разрушения и несанкционированного доступа 12
4.2. Правила, средства и порядок проведения процедур по копированию и восстановлению базы данных 12
Перечень сокращений 17

1. ПРАВИЛА ПОДГОТОВКИ ДАННЫХ

1.1. Порядок отбора информации для включения в базу данных

Для работы ПО «Визуализация» необходимо сформировать исходные данные для импорта в sql подобную базу (см. рисунок 1).

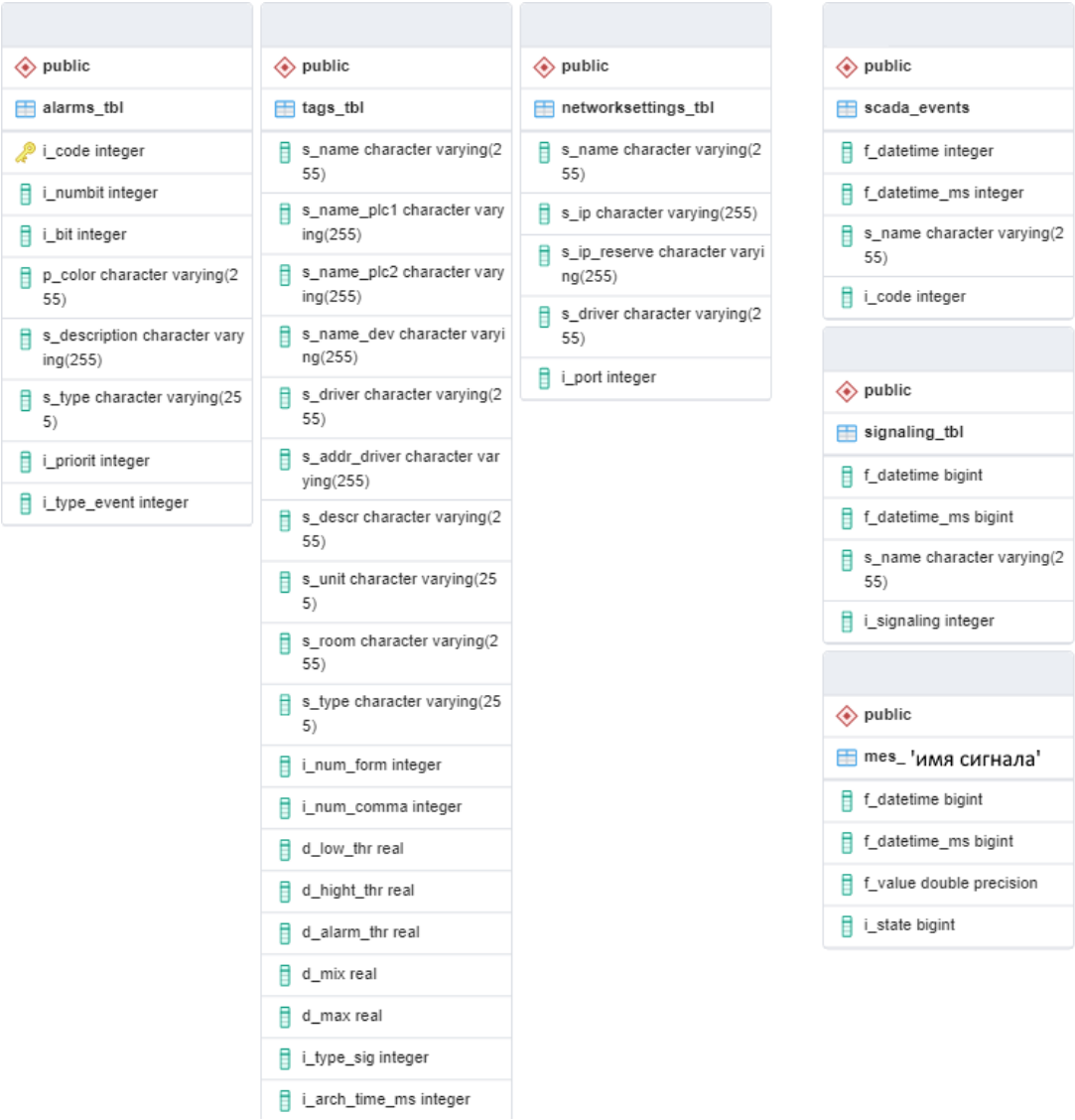

Рисунок 1 – Структура базы данных ПО «Визуализация»

Обязательные для работы ПО таблицы базы данных:

1) signaling_tbl – таблица активных сигнализаций с меткой времени;

2) alarms_tbl – таблица настроек аварийных, предупредительных, технологических и пользовательских сообщений;

3) networksettings_tbl – таблица с tcp/ip настройками tcp/ip серверов для обмена информацией;

4) tags_tbl – основная таблица базы данных сигналов (включающая: имя, описание, тип, адрес и др. свойства);

5) scada_events – таблица для формирования архивов аварийных, предупредительных, технологических и пользовательских сообщений;

6) mes_’имя tag’ – таблицы для хранения архивов измерения. Данные таблицы создаются автоматически при работе службы.

Создание указанных выше таблиц производится путем экспорта DDL/SQL файла из состава установочного комплекта ПО «Визуализация».

Импорт данных в таблицы signaling_tbl, alarms_tbl, networksettings_tbl и tags_tbl производится специальным ПО для sql-подобных СУБД (например, для postgresql - pgadmin).

1.1 Правила подготовки и кодирования информации

Исходные данных для импорта в базу должна быть в виде простых текстовых файлов с использованием разделителей “;” или в виде электронных таблиц формата Microsoft Excel/ Libre Office.

Для таблиц signaling_tbl, alarms_tbl, networksettings_tbl и tags_tbl должен быть создан отдельный файл исходных данных. Имя файла должно иметь следующий вид: *System_’наименование таблицы’_’версия’.’расширение файла’*.

Данные в таблицы scada_events и mes_’имя tag’ производится автоматически службой архивирования SCADA_Service из состава установочного комплекта ПО «Визуализация».

Для измерительных данных в базе данных заведены таблицы измерительных данных mes_’tag_name’ (где tag_name – имя тэга) (см. таблицу 1).

Таблица 1 - Поля таблицы mes_’tag_name’

Название	**Описание**	**Тип данных SQL**	**Обязательность**	**Примечание**
f_datetime	отметка времени	datetime	да	первичный ключ
f_value	текущее измерение	real	нет	-

Название	Описание	Тип данных SQL	Обязательность	Примечание
i_state	код обощенного состояния: • для RADTP — alarm; • для SM1820 и MODBUS TCP — condidtion	int	нет	-

Для таблицы событий в базе данных заведена таблица статусных данных tag_events (см. таблицу 2).

Таблица 2 - Поля таблицы tag_events

Название	Описание	Тип данных SQL	Обязательность	Примечание
f_tag	имя сигнала	varchar(30)	нет	-
f_datetime	отметка времени	datetime	нет	-
i_state	код обощенного состояния: • для RADTP — alarm; • для SM1820 и MODBUS TCP — condidtion	int	нет	-

1.2 Формы представления информации

Информация представляется в виде интерактивных экранных форм, входящих в состав графического пользовательского интерфейса SCADA-Эльбрус.

1.3 Правила заполнения форм

Правила заполнения форм см. ниже.

1.4 Порядок внесения изменений информации

Сведения о порядке внесения изменений информации изложены в документе «SCADA-Эльбрус. Руководство оператора ЛЯЮИ.00673-01 34 01».

2. ПОРЯДОК И СРЕДСТВА ЗАПОЛНЕНИЯ БАЗЫ ДАННЫХ

2.1. Состав технических средств

В состав технических средств входит два сервера БД: основной (master) и резервный (slave). Все изменения в структуре или содержимом БД отображаются одновременно на обоих серверах. В нормальном («рабочем») режиме клиентские приложения обращаются к основному серверу. При возникновении проблем с основным сервером (потеря связи, поломка) клиентские приложения перенаправляют свои запросы к резервному серверу. Подробное описание о настройке основного и резервного серверов содержится в подразделе 4.2**.**

Конфигурация БД храниться на сервере /database/DB. Конфигурация резервного сервера отличается от основного только наличием файла recovery.conf.

Сервер БД предназначен для сбора информации и долговременного ее хранения. На сервере устанавливается СУБД PostgreSQL.

Информационный обмен данными между сервером и клиентами осуществляется посредством модемных коммутируемых соединений через сеть Ethernet.

Шкаф сервера укомплектован:

1) сервером;

2) источником бесперебойного питания, позволяющим сохранить питание устройств шкафа в случае исчезновения основного питания;

3) модулями защиты интерфейса Ethernet;

4) вспомогательными техническими устройствами для организации разводки кабелей внутри шкафа;

5) выносными блоками расширения портов Ethernet.

2.1 Правила, порядок, последовательность и описание процедур, используемых при заполнении базы данных

2.1.1 SQL-запросы к справочнику оборудований и помещений

- **Создание таблицы equipment_info:**

```
CREATE TABLE public.equipment_info (
        param_name varchar(255) NOT NULL,
        building varchar(255) NULL,
        "level" varchar(255) NULL,
        purpose varchar(255) NULL,
```

```
        room varchar(255) NULL,
        area float4 NULL,
        device varchar(255) NULL,
        "zone" varchar(255) NULL,
        CONSTRAINT equipment_info_check CHECK ((area > (0)::double precision)),
        CONSTRAINT equipment_info_pk PRIMARY KEY (param_name),
        CONSTRAINT equipment_info_device_images_fk FOREIGN KEY (device) REFERENCES
device_images(device_id),
        CONSTRAINT equipment_info_room_images_fk FOREIGN KEY (room) REFERENCES
room_images(room_id) );
```

- **Создание таблицы room_images:**

```
CREATE TABLE public.room_images (
        room_id varchar(255) NOT NULL,
        image bytea NULL,
        CONSTRAINT rooms_img_pk PRIMARY KEY (room_id)
);
```

- **Создание таблицы device_images:**

```
CREATE TABLE public.device_images (
        device_id varchar(255) NOT NULL,
        image bytea NULL,
        CONSTRAINT device_images_pk PRIMARY KEY (device_id)
);
```

- **Получение данных из справочника оборудований и помещений:**

```
SELECT e.building,e.level,e.purpose,e.room,e.area,e.device,e.zone, r.image, d.image
FROM equipment_info AS e
INNER JOIN room_images AS r ON e.room = r.room_id
INNER JOIN device_images AS d ON e.device = d.device_id
WHERE e.param_name = '<имя_параметра>';
```

- **Внесение новых записей в справочник оборудований и помещений:**

```
INSERT INTO equipment_info (building,level,purpose,room,area,device,zone,param_name) VALUES
(?,?,?,?,?,?,?,?);
INSERT INTO room_images (image,room_id) VALUES (?,?);
INSERT INTO device_images (image,device_id) VALUES (?,?);
```

- **Обновление записей в справочнике оборудований и помещений:**

```
UPDATE equipment_info SET building=?,level=?,purpose=?,room=?,area=?,device=?,zone=? WHERE
param_name='<имя_параметра>';
UPDATE room_images SET image=? WHERE room_id='<идентификатор_помещения>';
UPDATE device_images SET image=? WHERE device_id='<идентификатор_оборудования>';
```

Удаление записей в справочнике оборудований и помещений:

```
DELETE FROM room_images WHERE room_id='<идентификатор_помещения>';
DELETE FROM device_images WHERE device_id='<идентификатор_оборудования>';
DELETE FROM equipment_info WHERE param_name='<имя_параметра>';
```

2.1.2 SQL-запросы к таблице измерительных, статусных и справочных данных

CREATE TABLE public.TAGS_TBL (S_NAME varchar(255), S_NAME_UP varchar(255), S_DRIVER varchar(255), S_ADDR_DRIVER varchar(255), S_DESCR varchar(255), S_UNIT varchar(255), S_ROOM varchar(255), S_TYPE varchar(255), I_NUM_FORM int, I_NUM_COMMA int, D_LOW_THR real, D_HIGHT_THR real, D_ALARM_THR real, D_MIX real, D_MAX real, I_TYPE_SIG int, I_PRIOR int, I_PORT int, S_IP varchar(255), S_IP_R varchar(255), I_ID_PROC int);

- **Создание таблицы настроек TCP/IP подключений NETWORKSETTINGS_TBL:**

CREATE TABLE public.NETWORKSETTINGS_TBL (S_NAME **varchar**(255), S_IP **varchar**(255), S_IP_RESERVE **varchar**(255), S_DRIVER **varchar**(255), I_PORT **int**, I_ID **int**);

- **SQL-запросы - загрузка данных из таблицы настроек TCP/IP подключений NETWORKSETTINGS_TBL:**

SELECT sNAME, sIP, sIP_RESERVE, sDRIVER, iPORT, iID **FROM** NETWORKSETTINGS_TBL;

- **SQL-запрос – создание таблицы конфигураций событий ALARMS_TBL:**

CREATE TABLE public.ALARMS_TBL (I_CODE **int**, I_NUMBIT **int**, P_COLOR_ON **varchar**(255), P_COLOR_OFF **varchar**(255), S_DESCRIPTION **varchar**(255));

- **SQL-запросы - загрузка данных из таблицы конфигураций событий ALARMS_TBL:**

SELECT I_CODE, I_NUMBIT, P_COLOR_ON, P_COLOR_OFF, S_DESCRIPTION **FROM** ALARMS_TBL

- **Создание таблицы mes_’tag_name’:**

CREATE TABLE mes_'tag_name' (f_datetime datetime, f_value **real**, i_state **int**, **PRIMARY KEY** (f_datetime));

- **Получение данных из таблицы измерительных данных:**

SELECT f_datetime, f_value, i_state **FROM** mes_'tag_name' **WHERE** f_datetime='<отметка_времени>';

- **Внесение новых записей в таблицу измерительных данных:**

INSERT INTO mes_'tag_name' (f_datetime, f_value, i_state) **VALUES** (?, ?, ?);

- **Обновление записей в таблице измерительных данных:**

UPDATE mes_'tag_name' **SET** f_value=?,i_state=? **WHERE** f_datetime='<отметка_времени>';

- **Удаление записей в таблице измерительных данных:**

DELETE FROM mes_'tag_name' **WHERE** f_datetime='<отметка_времени>';

- **Создание таблицы tag_events:**

```
CREATE TABLE tag_events (
        f_tag varchar(30),
        f_datetime datetime,
        i_state int
);
```

- **Получение данных из таблицы измерительных данных:**

SELECT f_tag, f_datetime, i_state **FROM** tag_events **WHERE** f_tag='<имя_тэга>';

- **Внесение новых записей в таблицу измерительных данных:**

INSERT INTO tag_events (f_tag, f_datetime, i_state) **VALUES** (?, ?, ?);

- **Обновление записей в таблице измерительных данных:**

UPDATE tag_events **SET** f_datetime=?,i_state=? **WHERE** f_tag='<имя_тэга>';

- **Удаление записей в таблице измерительных данных:**

DELETE FROM tag_events **WHERE** f_tag='<имя_тэга>';

2.2. Перенос данных на машинные носители информации

Перенос данных на машинные носители информации осуществляется сохранением автономной резервной копии в файл (для последующего сохранения на другом носителе).

3. ПРОЦЕДУРЫ ИЗМЕНЕНИЯ И КОНТРОЛЯ БАЗЫ ДАННЫХ

3.1. Состав и последовательность выполнения процедур по контролю и изменению содержания базы данных

Детальные сведения о составе и последовательности выполнения процедур по контролю и изменению содержания базы данных изложены в подразделе 2.2.

Контроль информации базы данных осуществляется подсистемой контроля достоверности информации базы данных и подсистемой мониторинга состояний.

Функции контроля выполняются по расписанию подсистемы контроля достоверности или по запросу пользователя. По результатам выполнения функции контроля определяется уровень достоверности данных, заполняется журнал работы подсистемы контроля достоверности.

Подсистема мониторинга состояний включает в себя модули мониторинга функции технологического контроля. По фактам отрицательных решений функции технологического контроля происходит оповещение ответственных пользователей. Подсистема мониторинга состояний обеспечивает общий мониторинг состояния информации в базе данных, определяет реакции SCADA-Эльбрус на некорректные данные БД, отсутствие данных.

Репликации и синхронизация базы данных осуществляется подсистемой обеспечения надежности и защиты информации. В процессе репликации и синхронизации данных используются стандартные средства СУБД PostgreSQL.

4. ПОРЯДОК И СРЕДСТВА ВОССТАНОВЛЕНИЯ БАЗЫ ДАННЫХ

4.1. Описание средств защиты базы от разрушения и несанкционированного доступа

Основными средствами защиты данных являются системные средства аутентификации и шифрования СУБД. Используемая СУБД PostgreSQL гарантирует высокую устойчивость к взлому и сохранность информации.

Таблицы базы данных защищены от несанкционированного доступа средствами, предоставляемыми подсистемой обеспечения надежности и защиты информации. Каждый пользователь имеет доступ только к строго определенному набору информации. Наборы информации определяются ролью пользователя, хранимой моделью защиты информации.

Для работы SCADA-Эльбрус в СУБД PostgreSQL зарегистрированы следующие пользователи с паролем по умолчанию «Qw123Qw» и набором прав, перечисленных в таблице (см. таблицу 3):

Таблица 3 – Права доступа пользователя

имя пользователя	**подключение**	**чтение/запись**	**создание/удаление**	**репликация**
dbuser	+	+	+	-
info	+	+	-	-
postgres	+	+	+	+

П р и м е ч а н и е. Ручное создание/редактирование/удаление директорий и файлов БД средствами ОС Linux (mkdir/rmdir/touch/nano/rm) должно осуществляться только пользователем *postgres*! Например, создание директорий для работы с БД:

$mkdir -m 700 database/DB; $mkdir -m 700 database/TS.

Восстановление таблиц базы данных осуществляется по операторскому запросу. Архивы информации хранятся в течение 3 лет.

4.2. Правила, средства и порядок проведения процедур по копированию и восстановлению базы данных

Операция резервирования и восстановления базы данных включает в себя:

1) создание резервной копии БД;
2) восстановление БД из резервной копии;
3) архивацию резервных копий для экономии дискового пространства;

4) сохранение автономной резервной копии в файл для последующего сохранения на другом носителе;

5) автоматизацию выполнения операции резервирования путем настройки планировщика операционной системы;

6) выборочный экспорт/импорт в файл/из файла для переноса данных между БД.

Для создания резервной копии (backup) БД необходимо выполнить следующую команду из командной строки сервера:

$ pg_basebackup -x -v -P --format=tar -h 127.0.0.1 -p 7005 -U postgres -D /database/backup/data/backup-$(date +%Y-%m-%d)

Для обеспечения высокой доступности и отказоустойчивости БД в SCADA-Эльбрус предусмотрена репликация путём горечого резервирования. Суть репликации состоит в копировании информации из БД основного сервера (master) на резервный (slave). Основной сервер пишет данные постоянно, а резервный вытаскивает изменения и применяет их у себя по мере возможности.

Ниже подробно описаны действия администратора БД для настройки основного и резервного серверов. Используемые сокращения:

- <master_ip> — IP-адрес основного сервера;
- <slave_ip> — IP-адрес резервного сервера;
- <PSQL_VER> — версия СУБД PostgreSQL, установленная на сервере.

4.2.1. Настройки основного сервера

П р и м е ч а н и е. Все перечисленные ниже команды нужно выполнять как пользователь *postgres*:

1) Создайте нового пользователя по имени *rep*, специально предназначенного для репликации (заменить <password> на надежный пароль):

$psql -c "CREATE USER rep REPLICATION LOGIN CONNECTION LIMIT 1 ENCRYPTED PASSWORD '<password>';";

2) Откройте каталог конфигураций postgres:

$cd /etc/postgresql/<PSQL_VER>/main;

3) Отредактируйте следующий файл, указав в нём нового пользователя:

$nano pg_hba.conf;

В любой точке файла (только не в конце) поместите следующие строки, которые откроют новому пользователю доступ к этому серверу:

host replication rep <slave_ip>/32 md5

Сохраните и закройте файл.

4) Затем нужно открыть главный конфигурационный файл postgres:

nano postgresql.conf

5) Найдите в нём следующие параметры, раскомментируйте их и измените их значения таким образом:

listen_addresses = 'localhost,<master_ip>'

wal_level = 'hot_standby'

archive_mode = on

archive_command = 'cd .'

max_wal_senders = 1

hot_standby = on

Сохраните и закройте файл.

6) Перезапустите основной сервер, чтобы обновить настройки.

$service postgresql restart

4.2.2. Настройки резервного сервера

П р и м е ч а н и е. Все перечисленные ниже команды нужно выполнять как пользователь *postgres*:

1) Для начала отключите на резервном сервере программу postgresql:

$service postgresql stop

2) Теперь нужно отредактировать файлы postgresql. Откройте каталог настроек:

$cd /etc/postgresql/<PSQL_VER>/main

3) Отредактируйте файл доступа, чтобы ведущий сервер мог подключаться к данному серверу. Это нужно на тот случай, если в дальнейшем понадобится поменять серверы master и slave местами.

$nano pg_hba.conf

Добавьте в файл следующую строку (опять же, только не в конец файла):

host replication rep <master_ip>/32 md5

Сохраните и закройте файл.

4) Откройте конфигурации postgresql:

$nano postgresql.conf

Можно использовать те же опции, что и на основном сервере, изменив только IP-адрес.

listen_addresses = 'localhost,<slave_ip>'

wal_level = 'hot_standby'

archive_mode = on

archive_command = 'cd .'

max_wal_senders = 1

hot_standby = on

Сохраните и закройте файл.

5) Теперь скопируем данные БД с основного сервера:

$psql -c "select pg_start_backup('initial_backup');"

$rsync -cva --inplace --exclude=*pg_xlog* /var/lib/postgresql/<PSQL_VER>/main/ <slave_ip>:/var/lib/postgresql/<PSQL_VER>/main/

$psql -c "select pg_stop_backup();"

Команда rsync может выдать ошибку при изменении файлов сертификата, но это поведение можно считать нормальным.

6) Теперь нужно подготовить файл для восстановления данных на сервере slave. Откройте каталог:

$cd /var/lib/postgresql/<PSQL_VER>/main

7) Создайте в нём файл по имени recovery.conf:

$nano recovery.conf

Внесите в файл следующий код, указав IP-адрес основного сервера и пароль пользователя rep:

standby_mode = 'on'

primary_conninfo = 'host=<master_ip> port=5432 user=rep password=<password>'

trigger_file = '/database/DB/trigger_file'

8) Итак, теперь slave сервер готов к работе. Введите:

$service postgresql start

4.2.3. Переключение с основного на резервный сервер

Для переключения (slave->master) необходимо на резервном сервере от имени пользователя *postgres* выполнить команду:

$touch /database/DB/trigger_file

Это необходимо в случае выхода из строя или технического обслуживания основного сервера. После выполнения данной команды резервный сервер перестанет реплицироваться с основным и станет доступен для операций записи.

Исправив ошибки на основном сервере, можно снова запустить его как master и заново перенастроить вручную slave как описано в подразделе 4.2.2.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

SCADA	Supervisory Control And Data Acquisition — диспетчерское управление и сбор данных
ВК	Видеокадр
БД	База данных
ОС	Операционная система
ПО	Программное обеспечение
ППО	Прикладное программное обеспечение
ПТК	Программно-технический комплекс
АСУ	Автоматизированная система управления
ТС	Технические средства
СУБД	Система управления базами данных

Лист регистрации изменений									
Номера листов (страниц)					Всего листов (страниц) в докум.	№ документа	Входящий № сопроводительного документа и дата	Подпись	Дата
Изм.	Изме-ненных	Замененных	Новых	Аннули-рованных					