**Основные понятия:**

ПУ — пункт управления.

КП — контролируемый пункт (RTU, remote terminal unit).

ОИК — оперативно информационный комплекс.

ЦППС — центральная приемо-передающая станция (CTS, central transceiver station).

ТС — телесигнал.

ТИ — телеизмерение.

ASDU — блок данных, обслуживаемый прикладным уровнем.

APDU — протокольный блок данных прикладного уровня.

COT — причина передачи (cause of transmission).

GSM-устройство — это устройство приёмо-передачи, использующее сети операторов мобильной связи.

SMS — служба коротких сообщений.

**Введение**

В реальных условиях организация взаимодействия между ПУ и КП для целей обмена телемеханической информацией не всегда бывает возможна с использованием физических линий связи, например в следствии территориальной удаленности КП от некоего географического центра, или же построение физических линий связи становится экономически нецелесообразным.

Так как зоны покрытия базовых станций операторов сотовой связи постоянно расширяются, то в качестве альтернативы проводным линиям связи для обмена телемеханической информации возможно использовать мобильную радиосвязь. В частности, если объем передаваемых данных не велик, то удобной оказывается служба коротких сообщений, описанная в стандарте GSM 03.40.

Максимальная длина одного текстового сообщения в стандарте GSM 140 байт (1120 бит). Таким образом при использовании 7-битной кодировки (латинский алфавит, цифры) возможно отправлять сообщения длиной до 160 символов. Для поддержки других национальных алфавитов (в частности русского) используется 2-байтовая кодировка UCS2. Таким образом, SMS, написанное кириллицей, не может превышать 70 символов. Также возможна отправка SMS, которое состоит из нескольких частей (concatenated SMS), теоретически до 255 частей. Для рационального использования доступного объема SMS необходимо применять компактный и практичный формат обмена телемеханической информации. Текстовый формат сообщений дает возможность представить набор телесигналов и телеизмерений в свободной форме, которую оператор может оценить наглядно.

Одним из достоинств использования SMS технологии, является то что функцию хранения и доставки сообщений берет на себя SMS-центр.

Процесс получения телемеханической информации от КП, поверх существующих сетей сотовой связи средствами SMS, и дальнейшую ее передачу в ЦППС с использованием протокола IEC 60870-5-104 можно разбить на несколько этапов:

*Первый этап*

Получение SMS

Разбор SMS

Cинхронная запись сообщений на диск

*Второй этап*

Обслуживание очереди входящих сообщений

Сборка ТИ и ТС в ASDU в форме ascii-hex

Организация отправки ASDU с применением механизма транзакций

*Третий этап*

Непосредственная передача телемеханической информации в ЦППС средствами протокола IEC 60870-5-104.

**Подробности реализации**

*Список утилит*

iecinit

smsget, smsdrop

quemngr, asdusend, asduconv, sockwrite

iecproxy, ieclink, ginterrog

*Библиотеки вспомогательный функций*

io\_utils

sms\_utils

sms\_parsers

*Список конфигурационных файлов*

iecenv

rtu\_list.conf

cts\_list.conf

*GSM-устройство*

Модем Teleofis RX100-R4.

**Подготовительный этап**

**iecinit**

iecinit — основной скрипт инициализации, запускается как демон с использованием утилиты start-stop-daemon. Отвечает за старт всех дочерних процессов (экспорт переменных окружения?).

iecinit читает конфигурационный файл cts\_list.conf и для каждой записи вида   
cts\_name ip\_address port, запускает отдельный экземпляр программ quemngr и iecproxy в фоне, которые взаимодействуют с конкретной ЦППС. При первом запуске создает структуру каталогов для данных дочерних процессов.

Задает параметры конфигурации GSM-устройства с помощью AT-команд и утилиты stty:

* Разрешает сырой ввод-вывод символьного устройства
* Устанавливает режим кодирования USC2
* Определяет область памяти для хранения SMS
* Переводит GSM-устройство в режим PDU (Protocol Data Unit)
* Запускает фоновый процесс smsget для опроса GSM-устройства на наличие новых SMS.

**Первый этап**

В unix используется принцип «все есть файл». GSM-устройство в unix-подобной операционной системе обычно представлено файлом символьного устройства в каталоге /dev, например /dev/ttyUSB0 или /dev/ttyS0. Для записи и чтения данных из файла устройства используются стандартные средства оболочки:

dev=/dev/ttyUSB0

exec 3<$dev 4>$dev

printf '%s\r\n' "$AT-cmd" >&4

read <&3 reply

Данные команды используются в библиотеке shell-функций io\_utils, предоставляющей более удобный интерфейс для взаимодействия с GSM-устройством.

**Smsget**

smsget — программа, периодически опрашивающая GSM-устройство на наличие новых SMS, посредством стандартных AT-команд:

AT+CMGL=4 — получение списка всех доступных SMS

AT+CMGD=i — удаление из памяти устройства SMS с индексом i.

Список SMS представляет собой набор записей вида

+CMGL: 4,1,"",127

07919740430900F0040B919740324334F90008817050104140216C041F044004380431043E04400020003200380020041D043504380441043F044004300432043D043E04410442044C0020043200200437043E043D04350020004E00200031002000320033003A00340039003A00300030002000320036002D00310030002D0032003000310037

Данная SMS записана в формате PDU. Вся служебная информация такая, как адрес отправителя, кодировка, штапм-времени получения сообщения SMS-центром, номер части составной SMS, а так же текст сообщения находятся в теле TPDU (Transfer Protocol Data Unit).

Smsget использует библиотеку shell-функций sms\_utils с помощью, которой разбивает TPDU на поля, декодирует некоторые из них и склеивает отдельные части составной SMS, если таковые имеются.

После разбивки TDPU анализ SMS строится по такому алгоритму:

1. smsget сравнивает телефонный номер отправителя, с номерами из rtu\_list.conf. Если номер отсутствует, сообщение удаляется.

2. Если номер оказался валидным, smsget запускает дочерний фоновый процесс   
smsdrop $dev $comaddr $dcs $scts $data, где

dev — модель устройства, отправившего SMS;

comaddr — общий адрес ASDU;

dcs (data coding scheme) — кодировка текста сообщения UCS2 или GSM;

scts (service centre time stamp) — метка времени получения сообщения SMS-центром;

data — текст SMS в кодировке $dcs.

3. smsget ожидает окончания работы smsdrop, и в зависимости от кода завершения, либо удаляет сообщение из памяти GSM-устройства (код возврата 0 или 2), либо считывает его повторно при следующем опросе (код возврата 1).

4. Если процесс smsdrop завершился успешно, то синхронно записанный им файл db\_archive/$comaddr/tempfile с набором значений телемеханики, процесс smsget копирует в каталог входящей очереди work/drop каждого экземпляра quemngr, который был запущен ранее iecinit, с именем его собственного inode. После этого smsget уведомляет каждый процесс quemngr о наличии нового сообщения в каталоге work/drop активацией FIFO триггера.

Разбор полученного сообщения осуществляет утилита smsdrop. Аргументы вызова функции были описаны выше. Используя указанную кодировку $dcs и функции из библиотеки sms\_utils, smsdrop декодирует текст сообщения $data и метку времени $scts. В зависимости от модели устройства, которое отправило SMS (значения аргумента $dev), smsdrop вызывает определенную функцию анализа текста $data из библиотеки sms\_parsers. Если текст сообщения соответствует формату, smsdrop синхронно записывает результат разбора в файл db\_archive/$comaddr/tempfile. Каждая строка в файле tempfile представляет собой набор полей:

Общий адрес ASDU ($comaddr);

Общий адрес объекта информации;

Идентификатор типа;

Значение объекта информации;

Описатель качества (если таковой используется);

Метка времени.

Пример

130 6 30 0 not\_used 27600:40:14:29:4:3:18

130 1 36 32 not\_used 27600:40:14:29:4:3:18

…

В поле «Описатель качества» вместо not\_used можно задействовать следующий набор набор битов, перечисленных через запятую:

OV — переполнение (Overflow);

SB — подстановки вручную (Substituted);

NT — не актуальное значение (Not topical).

Также smsdrop сохраняет полученный набор значений ТМ, но уже без меток времени, в базу активных значений db\_active в виде набора файлов db\_active/$comaddr/$type/$objaddr, где

type — идентификатор типа;

objaddr — общий адрес объекта информации.

База активных значений db\_active в дальнейшем используется для прикладной процедуры общего опроса, инициируемой со стороны ЦППС.

smsdrop будет иметь следующий код возврата:

0 — разбор и синхронная запись прошли успешно;

1 — синхронная запись сообщения на диск не удалась;

2 — неверный формат сообщения.

**Второй этап**

Чтобы избежать лавинной отправки полученных сообщений в ЦППС, а также реализовать транзакционный механизм используется менеджер очередей quemngr.

Структура рабочего каталога программы:

work/drop — очередь входящих сообщений. Файлы из входящей очереди попадают во внутреннюю, только если их имя совпадает с собственным inode.

work/in — внутренняя очередь сообщений.

work/act — рабочая очередь на отправку. В данной очереди находятся сообщения, отправка которых осуществляется в данный момент.

work/df — очереди отложенных сообщений. Сюда помещаются сообщения для повторной отправки через определенный таймаут. Отложенная очередь представляет собой именованный определенным образом каталог, например 0h0m50s, 0h20m0s, 1h30m25s. Оператор может сам управлять очередями для повторной отправки.

work/fail — каталог сообщений, отправка которых не удалась после определенного числа попыток.

При старте quemngr, сканирует каталоги очередей и восстанавливает связанные с ними структуры данных, а также запускает сопроцесс asdusend, с которым он связан неименованными каналами. При наличии канала до ЦППС начинает немедленную отправку отложенных сообщений, если таковые имеются.

Quemngr сканирует каталог входящих сообщений при каждой активации FIFO триггера. Все новые сообщения из входящей очереди помещаются во внутреннюю, которая уже повторно не сканируется. Quemngr поддерживает структуры данных, при перемещении сообщений из одной очереди в другую. Сообщения передаются по возможности одновременно, но с ограничением на количество (конкуренцию) процессов передачи. Таким образом можно управлять нагрузкой на ЦППС и предотвратить шквал передачи в случае приёма большого количества SMS. Сам quemngr передаёт данные опосредованно. Его основная задача: управление очередью и выдача команд на передачу сообщений.

Если во внутренней очереди work/in имеются сообщения, quemngr переносит их в рабочую очередь work/act и оповещает asdusend, передавая ему на стандартный ввод имя файлы для отправки (inode). На стандартный вывод asdusend асинхронно записывает имя файла и статус. Успешно переданные сообщения quemngr (статус 0) удалит из рабочей очереди, а с теми сообщениями передача, которых не удалась (статус > 1), поступит в соответствии с политикой повторной передачи: либо перенесет в отложенную очередь с отправкой через определенный таймаут, либо сбросит в каталог work/fail для последующего анализа и обработки вручную.

Quemngr устанавливает два обработчика для сигналов SIGHUP и SIGUSR1, которые ему в ходе работы может посылать процесс iecproxy.

Сигнал SIGHUP уведомляет о том, что сетевое соединение с ЦППС потеряно. В данном случае quemngr перестает следить за входящей очередью и прекращает отправку сообщений.

Сигнал SIGUSR1 уведомляет о том, что сетевое соединение с ЦППС установлено. В этом случае все происходит с точностью до наоборот, если имелись отложенные сообщения (например при старте процесса) начинается немедленная отправка, очередь входящих сообщений снова отслеживается.

Asdusend — сопроцесс quemngr, который на каждое передаваемое сообщение из рабочей очереди инициирует процедуру его передачи в ЦППС. Получив на стандартном вводе имя сообщения, asdusend делает следующее:

1. Записывает в ответ на стандартный вывод имя сообщения со статусом 1, уведомляя quemngr о том что сообщение поступило в обработку на отправку и возможна передача следующего сообщения из внутренней очереди.

2. Для каждого сообщения запускает два дочерних процесса asduconv и sockwrite, которые связаны между собой неименованным каналом.

Первый занимается формированием ascii-hex строки ASDU из сигналов ТМ, которые содержит файл рабочей очереди.

Второй отправляет сформированную строку ASDU в UNIX сокет.

3. Записывает статус завершения sockwrite вместе с именем файла на стандартный вывод.

Asduconv — программа, преобразующая набор сигналов ТМ (представление описано выше), в список строк ASDU длиной до 498 символов, в соответствии с форматом описанном в стандарте IEC 60870-5-101 (стандарт или ГОСТ?). Входные данные asduconv получает либо из файла, либо через стандартный ввод. При запуске указывается причина передачи (cos, cause of transmission) через ключ -c:

3 — спорадическая передача;

20 — общий опрос.

Например, следующий набор значений сигналов ТМ с причиной передачи 3  
130 1 30 0 not\_used 27600:40:14:29:4:3:18

130 2 30 1 not\_used 27600:40:14:29:4:3:18

130 1 36 32 not\_used 27600:40:14:29:4:3:18  
будет выглядеть следующим образом в виде строк ASDU  
1E020300820001000000D06B680E9D031202000001D06B680E9D0312

2401030082000100000000004200D06B680E9D0312.

Cформированный список ASDU передается на стандартный вывод.

Sockwrite — простая программа, принимающая на стандартном вводе текст, который записывает в указанный через ключ -s UNIX сокет. Sockwrite получив от asduconv сформированный список ASDU, отправляет его iecproxy. Дождавшись от iecproxy служебного сообщения, описывающего статус передачи сообщения в ЦППС, завершается:

«<» передача прошла успешно (sockwrite завершается с кодом 0);

«-» передача не удалась (sockwrite завершается c кодом 2).

**Третий этап**

На данном этапе происходит непосредственная отправка сформированных ASDU в ЦППС, а также работа прикладной процедуры общего опроса.

Iecproxy — программа-посредник между менеджером очередей и ЦППС, проксирует запросы на передачу ASDU и уведомляет quemngr с помощью сигналов о наличии или отсутствии сетевого соединения.

В качестве дочернего процесса запускает программу ieclink, реализующую канальный уровень протокола IEC 60870-5-104, с которым связана неименованными каналами. Для взаимодействия с ieclink использует простой служебный протокол.

Сообщения со стороны ieclink:

«+» - соединение с ЦППС установлено;

«-» - соединение с ЦППС потеряно;

«<» - сообщение успешно передано.

Сообщения со стороны iecproxy:

«>» - получен конец сообщения.

4.

Принимает через свой unix-сокет список ASDU в виде ASCII HEX текста, который записывает на стандартный ввод ieclink. Отслеживает статус переданых ASDU.

Уведомляет об отключении или востановлении канала до ЦППС процесс с pid, который она получает через перемунную окружения PID\_NOTIFY (в нашем случае это quemngr). Для этого использует пользовательские скрипты connect\_hook и disconnect\_hook.

Управляет прикладной процедурой общего опроса.

**Ieclink**

Ieclink — программа, реализующая канальный уровень протокола IEC104. В качестве аргументов принимает ip-адрес и порт сервера ЦППС. Устанавливает и поддерживает соединение (c помощью процедуры тестирования) с ЦППС.

Ieclink читает и пишет ASDU на стандартном вводе-выводе в виде текста ASCII HEX (разделителем ASDU служит символ переноса строки). Получив на стандартном вводе список ASDU, который завершается служебным символом «>», ieclink начинает отправку в ЦППС, дожидается передачи всех кадров на канальном уровне с подтверждением и записывает на стандартный вывод статус передачи.

Для уведомления iecproxy о событиях посылает на стандартный вывод короткие служебные сообщения «+» - соединение с ЦППС установлено, «-» - соединение с ЦППС потеряно (передача списка ASDU не удалась), «<» - передача списка ASDU прошла успешно.

**Ginterrog**

Ginterrog — программа, которая запускается iecproxy в качестве дочернего процесса в ответ на активацию прикладной процедуры общего опроса. Ginterrog формирует из базы активных сигналов, расположенных на диске в каталоге db\_active, список ASDU с причиной передачи 20 в виде текста ASCII HEX и записывает его на стандартный вывод.

Основные идентификаторы типов передаваемых сообщений, используемые в текущей реализации M\_SP\_NA\_1 (1), M\_ME\_NC\_1 (13), M\_SP\_TB\_1 (30), M\_ME\_TF\_1 (36).

**Основной сценарий работы для прикладной процедуры «Общий опрос»:**

1. ЦППС посылает команду общего опроса C\_IC\_NA\_1 с COT = 6 «активация». После подключения ЦППС периодически, через определенный тайм аут выполняет процедуру общего опроса или же по запросу оператора.

2. Iecproxy подтверждает прием C\_IC\_NA\_1 на прикладном уровне, дублируя его в направлении ЦППС с COT = 7 «подтверждение активации».

3. Если это первая процедура общего опроса после подключения, то iecproxy дожидается от quemngr отложенных сообщений, которые могли прийти во время отсутствия канала с ЦППС или же служебного сообщения об их отсутствии. Даные ASDU передаются в ЦППС с COT = 3 «спорадически». Статус передачи каждого отложенного сообщения передается quemngr.

4. Далее iecproxy запускает программу ginterrog, которая формирует список ASDU из базы текущих значений с COT = 20 «общий опрос», и начинает его передачу в ЦППС.

5. Закончив передачу последнего ASDU, iecproxy посылает команду C\_IC\_NA\_1 с COT = 10 «завершение активации».

6. Всю работу с канальным уровнем обеспечивает программа ieclink.

**Основной сценарий работы при установленном соединении:**

1. Все вновь пришедшие сообщения, находящиеся в рабочей очереди quemngr передаются в ЦППС с COT = 3 «спорадически».

2. В зависимости от статуса передачи они либо удаляются из рабочей очереди, либо переносятся в отложенную очередь с повторной отправкой через определенный тайм аут.