## Структура ПО дорожки

Структура программного обеспечения показана на рисунке 1.

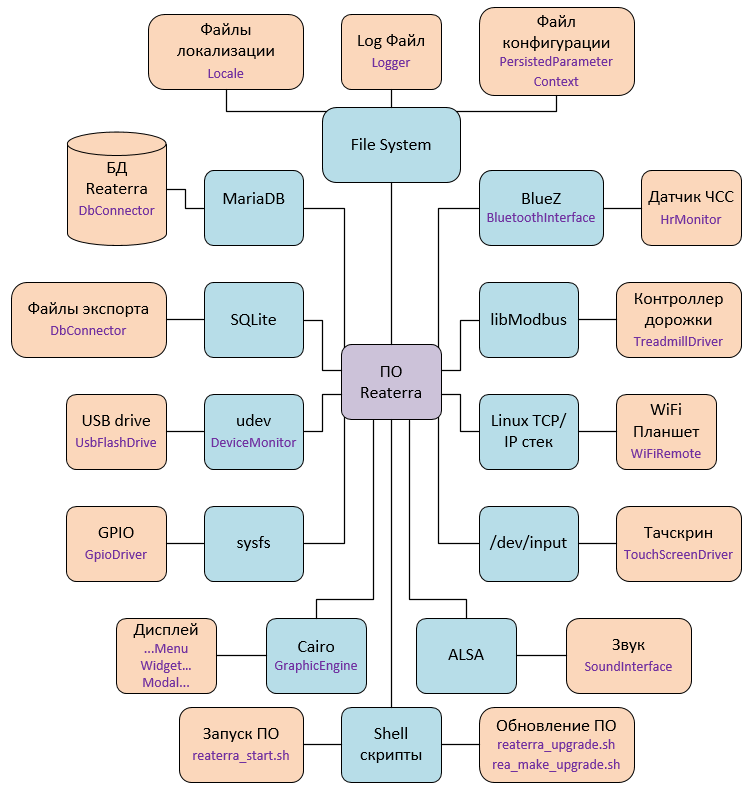


Рисунок 1. Структура ПО Reaterra.

На схеме голубым показаны успользуемые ресурсы, розовым объекты относящиеся к ресурсам. Фиолетовым подписаны названия классов-интерфейсов в объектам или ресурсам. ПО использует следующие внешние ресурсы:

* **Файловая система** для хранения и доступа к файлу настроек reaterra.cfg, лог-файлу reaterra\_log.txt и файлам локализации reaterra\_locale\_XX.txt. Для работы с файлами используется стандарнтый API Linux;
* **Системный** **стек TCP/IP** для связи с планшетом по WiFi.
* **Драйвер** **sysfs** для работы с GPIO микропроцессора - получения информации о состоянии питания системы;
* **Устройство /dev/input** управляемое драйвером тачскрина для получения информации о взаимодействии пользователя с экраном дисплея.
* **Менеджер устройств udev**, предоставляющий сервис получения нотификаций при подключении/отключении USB накопителя;
* **Библиотеки и утилиты ALSA** для воспроизведения звуковых нотификаций;
* **Библиотека BlueZ** для работы с Bluetooth устройствами и конкретно с датчиком ЧСС;
* **Библиотека Cairo** для построения графического интерфейса.
* **Библиотека libModbus** для обмена данными с контроллером дорожки по протоколу MODBUS;
* **СУБД MariaDB** для хранения данных пациентов;
* **СУБД SQLite** для возможности выгрузки данных в формате предыдущей версии дорожки;
* **Shell скрипты** успользуются для запуска ПО и реализации процедуры обновления ПО.

## Структура пользовательского интерфейса дорожки

Пользовательский интерфейс использует концепции окна и набора виджетов находящихся в нем. Структура элементов пользовательского интерфейса показана на рисунке 2.

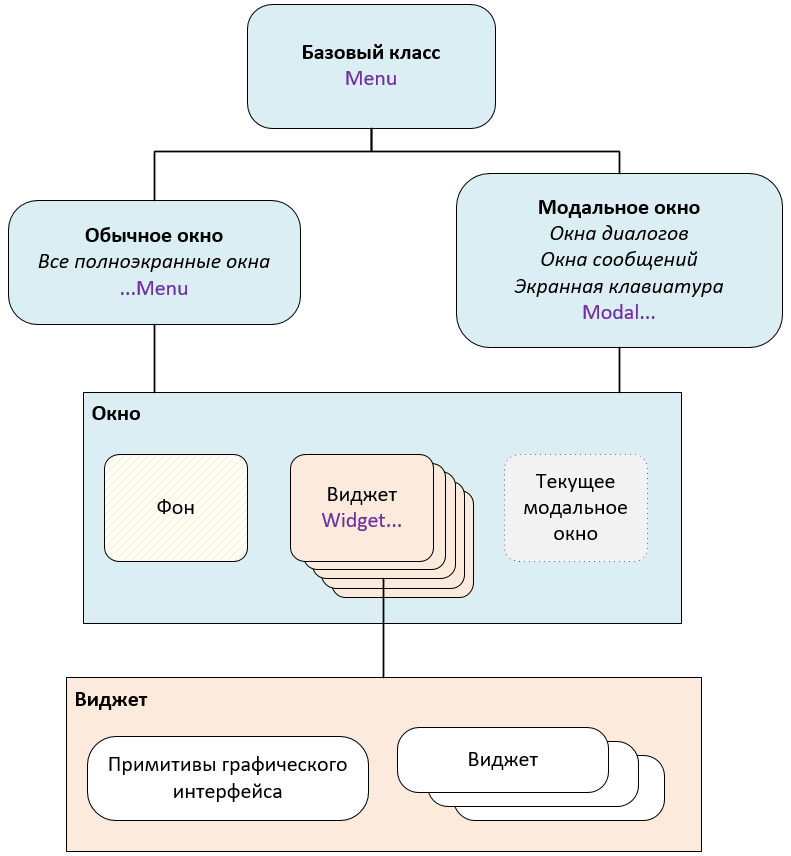


Рисунок 2. Структура элементов пользовательского интерфейса.

Все окна отнаследованы от класса Menu. Окна делятся на обычные(полноэкранные) и модальные. На основе обычных окон построены все экраны меню интерфейса. Модальные окна это окна диалогов, сообщений и экранных клавиатур. Поверх обычного окна может быть открыто только одно модальное окно. Если модальное окно открыто, то элементы интерфейса обычного окна блокируются и не реагируют на события взаимодействия с пользователем.

Окна содержат информацию о своем положении, картинке фона, списке виджетов и наличии открытого модального окна (для обычных окон).

Виджет может отрисовывать свой вид используя примитивы графического движка, а также может включать в себя другие виджеты.

## Система сообщений ПО дорожки

Взаимодействие между компонентами системы построено в основном на использовании системы сообщений. Структура системы сообщений показана на рисунке 3.

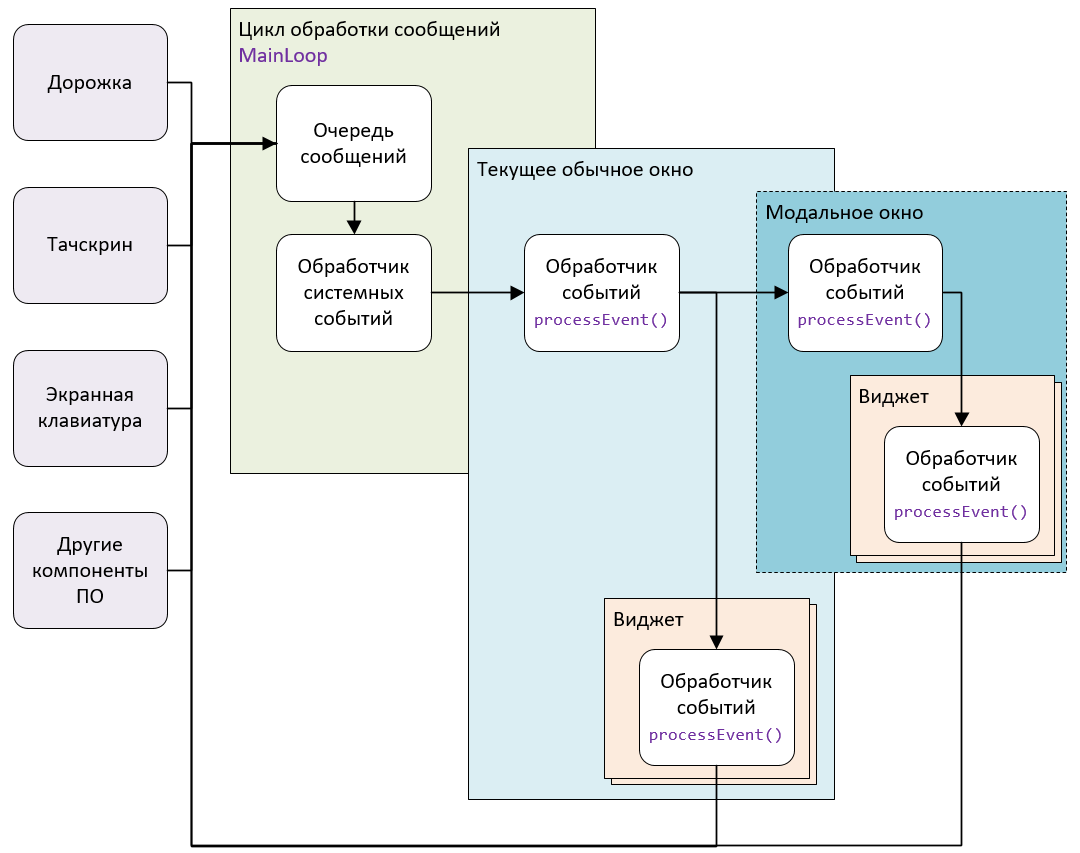


Рисунок 3. Структура системы сообщений.

Цикл обработки сообщений обслуживает очередь сообщений. При наличии сообщения в очереди, сообщение извлекается и передается обработчику системных событий (таких как завершение работы или перезагрузка). Если системный обработчик не смог обработать событие, то он передает его текущему обычному окну вызывая функцию обработчика события окна. Если в обычном окне было активировано модальное окно, то событие передается модальному окну. Обработчик окна может сам обработать событие, а в случае если пришло событие от клавиатуры или тачскрина, то событие передается виджету имеющему фокус ввода или виджету который был активирован(когда координаты события лежат в пределах границ виджета) событием тачскрина.

События поступают в очередь сообщений от различных компонентов ПО, однако основными источниками являются классы, обслуживающие такие компоненты системы как контроллер дорожки и тачскрин. События могут генерировать и сами обработчики событий. Таким образом, к примеру, экранная клавиатура получая событие об отпускании ранее нажатой клавиши сигнализирует о наборе символа.

Также обработчики событий могут запросить смену текущего окна интерфейса.

## Реализация обновления ПО дорожки

Обновление ПО дорожки происходит при включении дорожки с подключенным к ней флеш-диском в корневой директории которого имеется файл обновления upgrade.rea.

Программа обновления реализована в shell скрипте reaterra\_upgrade.sh. Процедура выполнения апгрейда следующая. Вначале происходит монтирование флэш-диска. Если диск был подмонтирован к файловой системе успешно, то проверяется наличие файла апгрейда. Далее создается временный каталог в который распаковывается содержимое файла обновления. Файл обновления должен содержать в себе как минимум три файла:

1. **upgrade.sh** - shell скрипт для выполнения процедуры конкретного апгрейда;
2. **new\_version.txt** - текстовый файл, содержащий номер новой версии ПО;
3. **required\_version.txt** - текстовый файл, содержащий номер версии ПО на которое разрешено проведение данного апгрейда.

Если требуемая версия не соответствует текущей версии апгрейд невозможен.

Также файл обновления может содержать дополнительные файлы, требуемые для проведения конкретного апгрейда.

В простейшем случае дополнительно имееется только исполняемый файл ПО дорожки reaterra, скрипт upgrade.sh выполняет перезапись старого исполняемого файла на новый.

## Структура Андроид программы

Программа клиент состоит из двух частей: сервиса и видимого приложения. Структура программы приведена на рисунке 4.

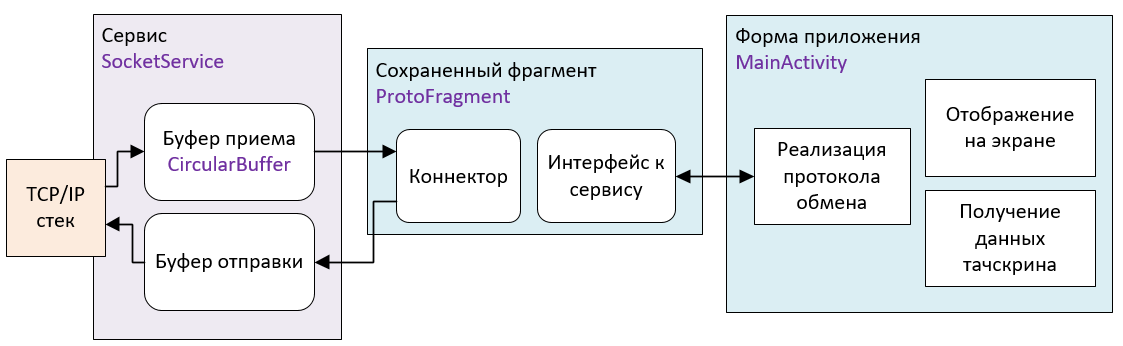


Рисунок 4. Структура Андроид приложения.

Сервис приложения служит для взаимодействия со стеком TCP/IP, через который происходит обмен данными. Сервис содержит реализации входного и выходного буфера данных, а также механизмы установки связи с хостом.

Главная форма приложения взаимодействует с сервисом через сохраняемый фрагмент, в котором находится коннектор к сервису и реализованы интерфейсы для доступа к сервису из формы приложения.

Форма приложения отображает получаемые от хоста картинки на экране устройства, а также отправляет информацию о взаимодействии пользователя с интерфейсом - по сути данные тачпада.

## Протокол обмена Андроид программы и дорожки

Канал связи обмена представляет собой сессию TCP/IP, поэтому вопросы гарантированной и неискаженной доставки решаются внутри стека TCP.

Протокол обмена состоит из двух типов пакетов: пакеты с картинками и пакеты с данными тачскрина.

Пакеты с картинками имеют следующий формат:

|  |  |  |
| --- | --- | --- |
| **№ поля** | **Количество байт** | **Описание** |
| 1 | 8 | Признак начала пакета. Содержит 8 байт со значением 0xFF. |
| 2 | 4 | Размер пакета в байтах вместе с заголовком. |
| 3 | 4 | Координата X картинки. |
| 4 | 4 | Координата Y картинки. |
| 5 | 4 | Ширина картинки в пикселях. |
| 6 | 4 | Высота картинки в пикселях. |
| 7 | 4 | Идентификатор картинки. |

Пакеты с данными тачскрина имеют текстовый формат следующей структуры:

*touch M XXXX YYYY*

где M - символ типа события: D - нажатие, U - отрыв; XXXX - координата X события в формате целого числа с ведущими нулями длиной 4-ре символа; YYYY - координата Y события в формате целого числа с ведущими нулями длиной 4-ре символа.

## Алгоритм работы удаленного управления

На каждое обновление экрана хост вычисляет положение и размер прямоугольника на экране содержащего измененные данные.

Затем определяется место во фрейм-буфере содержащее данный прямоугольник. Полученный массив байтов кодируется в формате PNG, к нему добавляется заголовок содержащий информацию о исходном прямоугольнике и размер получившегося пакета данных. Подготовленный пакет отправляется клиенту по протоколу TCP/IP.

На стороне клиента пакет принимается сервисом и помещается в кольцевой буфер. При успешном декодировании заголовка из буфера извлекается пакет и передается клиенту сервиса - видимой части приложения, текущей активности. Активность преобразовывает картинку из формата PNG в Bitmap и отрисовывает на экране в нужном месте.

Если пользователь андроид приложения нажал на элемент на экране, программа формирует пакет с данными о событии (нажатие или отрыв, координаты) и отправляет сервису. Сервис отправляет пакет на хост.

Хост и клиент проверяют наличие связи по времени между получаемыми пакетами. Если время превышает 5-6 секунд, то считается что произошел обрыв связи. Номинальное отключение клиент выполняет закрывая открытый на хост TCP/IP сокет.

Для поддержки связи в случае если но хосте более двух секунд не было изменений на экране, отправляется пакет содержащий прямоугольник нулевой высоты и ширины - по сути отправляется только заголовок пакета.

Если же на клиенте не было нажатий в течении 2-х секунд, то на хост отправляется heartbit-пакет для поддержания связи.