РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ПОРТАТИВНОГО СПИРОМЕТРИЧЕСКОГО АППАРАТНО-ПРОГРАММНОГО КОМПЛЕКСА

Н. И. Маничев, И. С. Войтешенко

Белорусский государственный университет
Минск, Беларусь
e-mail: voit@bsu.by

Рассматриваются технологические вопросы проектирования и разработки программного обеспечения инновационного портативного спирометрического устройства.

Ключевые слова: спирометрия; портативное устройство; проблемноориентированное проектирование; кроссплатформенный подход; Xamarin.

DEVELOPMENT OF THE SOFTWARE FOR PORTABLE SPIROMETRY HARDWARE AND SOFTWARE COMPLEX

N. I. Manichev, I. S. Vojteshenko

Belarusian State University
Minsk, Belarus

Consider the technological aspects of designing and software development for innovative portable spirometry device.

Keywords: spirometry; portable device; domain-driven design; cross-platform approach; Xamarin.

ВВЕДЕНИЕ

С развитием интернета вещей возникла и реализуется потребность в небольших портативных устройствах, подключающихся к смартфону или планшету и позволяющих сделать достаточно точные измерения показателей жизнедеятельности организма человека с использованием мобильного приложения. Среди этих устройств можно выделить спирометры [1] – медицинские приборы, позволяющие проводить исследования функции внешнего дыхания, включая измерение объемных и скоростных показателей дыхания. Спирометрия помогает диагностировать различные заболевания (бронхиальная астма, хроническая обструктивная болезнь легких и др.), а также позволяет оценивать состояние аппарата дыхания при профессиональных заболеваниях.

На рынке присутствуют в основном дорогостоящие и не слишком мобильные варианты спирометров. Существуют и портативные устройства [2], но точность измерения с их помощью недостаточно высока.

Настоящая статья посвящена проектированию и разработке с использованием технологии Xamarin программного обеспечения для поддержки портативного спирометрического устройства.

СТРУКТУРА АППАРАТНО-ПРОГРАММНОГО КОМПЛЕКСА, ПРОВЕДЕНИЕ ИЗМЕРЕНИЙ И ОБРАБОТКА РЕЗУЛЬТАТОВ

Измерения производятся с помощью измерительной трубки (трубки Флейша), через которую дышит пациент. Данные, полученные с измерительной трубки, через последовательный порт передаются в аппаратный мост FTDI, который в свою очередь доставляет их в портативное устройство (смартфон, планшет). Полученные данные обрабатываются портативным устройством, а затем отображаются на экране в виде медицинских протоколов в режиме реального времени. Указанные протоколы включают: график зависимости объема (л) и потока (л/с); показатели, характеризующие вентиляцию легких и бронхиальную проходимость; предварительный диагноз, который требует подтверждения врача. Все результаты исследования сохраняются в базе данных, что позволит в дальнейшем иметь возможность оценивать динамику состояния здоровья пациента.

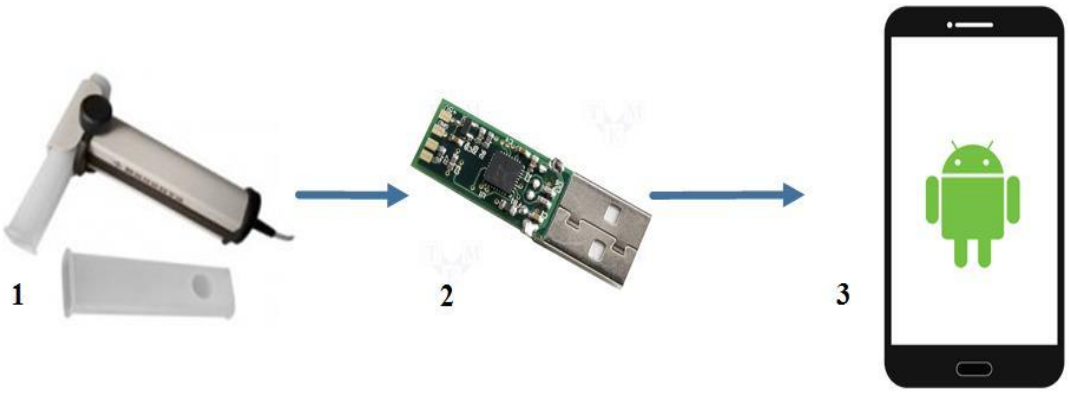

Рис. 1. Структурная схема разрабатываемого портативного спирометрического аппаратно-программного комплекса: *1* – измерительная трубка, *2* – аппаратный мост FTDI, *3* – портативное устройство (смартфон)

FTDI (Future Technology Devices International) — шотландская компания, которая разрабатывает, производит и осуществляет поддержку устройств и соответствующих программных драйверов для преобразования последовательной передачи данных по RS-232 или уровней TTL в сигналы шины USB. Это дает возможность современным компьютерам использовать простой классический RS-232 интерфейс, а разработчикам позволяет не вникать в сложности USB интерфейса [4].

В разрабатываемом приложении используется D2XX FTDI драйвер для платформы Android, который фактически является оберткой над Android USB API и позволяет взаимодействовать с измерительной трубкой при помощи следующих команд: Purge (очистка входного и выходного буферов); Read (чтение данных из порта); Write (запись данных в порт); Open (открытие порта); Close (закрытие порта); SetBaudRate (установка скорости порта); SetDataCharacteristics (установка параметров порта, таких как длина байта, количество стоповых бит и т. д.); SetFlowControl (установка режима аппаратного и программного управления потоком); SetLatency (установка таймера отправки пакетов).

Реализация данных функций и соответствующая библиотека поставляются в зависимости от платформы, что позволяет работать с прибором на различных портативных устройствах.

Внутри трубки Флейша находится металлическая сетка, которая при помощи разности давлений позволяет измерить поток воздуха (л/с). Поток вычисляется при помощи квадратного уравнения, коэффициенты которого получают в процессе калибровки устройства. По потоку вычисляется объем.

Полученный сигнал для уменьшения уровня шума фильтруется при помощи медианного фильтра. Значения отсчетов внутри окна фильтра сортируются в порядке возрастания (убывания); значение, находящееся в середине упорядоченного списка, поступает на выход фильтра. В случае четного числа отсчетов в окне выходное значение фильтра равно среднему значению двух отсчетов в середине упорядоченного списка [5].

Далее приложение вычисляет коэффициенты BTPS. BTPS (Body Temperature and Pressure Saturated) – методика коррекции измеряемых объемов и потоков с помощью учета остывания выдыхаемого воздуха и изменения его влажности. Поправочный коэффициент рассчитывается исходя из предположения, что выдыхаемый пациентом воздух при входе в спирограф охлаждается мгновенно.

Получив откорректированную функцию зависимости объема (л) от потока (л/с), приложение на основании конкретного теста анализирует ее значения, вычисляя ее экстремумы и другие показатели, необходимые для визуализации результатов исследований в виде медицинских протоколов.

ОБОСНОВАНИЕ ВЫБОРА ШАБЛОНОВ ПРОЕКТИРОВАНИЯ И ПРОГРАММНЫХ СРЕД РАЗРАБОТКИ, ОБРАБОТКА ОШИБОК И ТЕСТИРОВАНИЕ

При проектировании и разработке приложения применялись проблемно-ориентированное проектирование, шаблон проектирования MVPVM, кроссплатформенный подход на основе парадигмы аспектно ориентированного программирования, фреймворк Xamarin.

Проблемно ориентированное проектирование. Проблемно (предметно) ориентированное проектирование (DDD, Domain-driven design) — это набор принципов и схем, помогающих разработчикам создавать системы объектов. Подход DDD применяется тогда, когда разработчик не является специалистом в какой-то конкретной области, которая используется в разработке программного продукта.

В соответствии с этим подходом архитектура разрабатываемого приложения состоит из четырех основных модулей:

1. Уровень приложения. Используется для организации основной логики приложения, предоставляет интерфейс ко всем возможностям приложения.
2. Уровень домена (предметной области). Включает бизнес-логику приложения, сущности, объекты, значения, сервисы, которые отражают объекты и связи в предметной области.
3. Уровень инфраструктуры (уровень взаимодействия). Представляет сквозную функциональность приложения, что соответствует аспектно ориентированной парадигме программирования.
4. Уровень представления. Организует пользовательский интерфейс и взаимодействие с ним.

Шаблон проектирования MVPVM. Шаблон проектирования MVPVM состоит из двух частей: кроссплатформенной и специфической для каждой платформы. Первая часть состоит из модели и модели представления. Модель – это фактически данные нашего приложения: сущности, объекты значения, объекты, передающие данные. Модель представления является слоем приложения в разрабатываемом программном обеспечении. Она связывает данные приложения и представление, предоставляя весь необходимый функционал представлению. Платформозависимая часть, состоящая из презентера и представления, отвечает за пользовательский интерфейс. Такой подход позволяет отделить общую часть приложения (данные и бизнес-логику) от специфической для каждой платформы.

Кроссплатформенный подход. В соответствии с аспектно ориентированной парадигмой программирования в приложении можно выделить сквозную функциональность: логгирование, локализация, обработка ошибок и т. д. Но некоторые сервисы специфичны для каждой платформы, к примеру, сервис показа сообщений выглядит на Android и на Windows по-разному. Для того чтобы в нужный момент использовать правильную версию сервиса, выделяется общий интерфейс, уровень абстракции, а затем подставляется реализация для каждой платформы.

Xamarin – это фреймворк для кроссплатформенной разработки мобильных приложений (iOS, Android, Windows Phone) с использованием языка C#. Xamarin предоставляет отдельные компиляторы для каждой из этих платформ, позволяющие на выходе получать настоящие, нативные приложения, которые могут использовать все возможные аппаратные ресурсы платформы при помощи вызова функций стандартного программного интерфейса.

Обработка ошибок и тестирование. В разрабатываемой системе используется подход отлавливания исключительных ситуаций на всех уровнях приложения, в каждом классе, в каждом методе и оборачивании пойманного исключения в соответствующий наследник базового класса исключения, его логировании и передачи на уровень выше. В результате получаем иерархию исключений, которая не допускает возможности появления необработанной ошибки и, соответственно, не допускает падения приложения. Внутри наследников базового исключения хранится дополнительная информация, которая позволяет получить пользовательское сообщение об ошибке. Исключения передаются вплоть до того слоя приложения, где они успешно обрабатываются.

Проблема тестирования особенно остро стоит при разработке кроссплатформенных приложений. Для оптимизации процесса тестирования используют модульные и интеграционные тесты. В разрабатываемом приложении для написания модульных и интеграционных тестов используется библиотека MSTest, а для упрощения реализации заглушек используется фреймворк Moq. Кроме того, при помощи интеграционных тестов фактически можно протестировать весь функционал приложения, просто заменив презентационный модуль интеграционными тестами, что возможно благодаря выбранной архитектуре.

ЗАКЛЮЧЕНИЕ

В настоящее время рынок портативных устройств в области здравоохранения стремительно растет. В связи с тем, что в текущий момент на рынке спирометрии не существует устройств, обладающих достаточной точностью измерения и возможно-

стью подключения к мобильным устройствам, выход подобного устройства на рынок может иметь успех.

Не так давно возник польский стартап MySpiro [8], который получил финансирование и разработка которого ведется активно, но он пока обладает лишь прототипом подобного устройства. Другие схожие разработки нам неизвестны.

Степень готовности разрабатываемого программного обеспечения на момент написания данной статьи: разработана и протестирована общая часть приложения, включая слой инфраструктуры (взаимодействие с прибором, хранение данных в приборе, логирование и т. д.), доменный слой (сущности, объекты значения, сервисы), слой приложения (фасад приложения), модульные и интеграционные тесты. После разработки пользовательского интерфейса для платформ Android и Windows прибор будет сертифицирован и появится на рынке.

БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1. Спирометрия. Физиологические основы тестирования функции легких [Электронный ресурс]. URL: http://www.spiro.ru/info/osnovi/osnovi.htm (дата доступа: 18.04.2016).
2. Portable spirometers [Electronic resource]. URL: http://www.medicalexpo.com/-medical-manufacturer/portable-spirometer-30494.html (дата доступа: 17.04.2016).
3. Standardization of spirometry / M. R. Miller [et al.] // Eur. Respir. J. 2005. Vol. 26. P. 319–338.
4. FTDI [Electronic resource]. URL: http://www.ftdichip.com/ (дата доступа: 09.06.2016).
5. Медианный фильтр [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/-Медианный_фильтр (дата доступа: 07.06.2016).
6. Xamarin [Electronic resource]. URL: https://xamarin.com/ (дата доступа: 23.05.2016).
7. Эванс Э. Предметно ориентированное проектирование (DDD): структуризация сложных программных систем / Эрик Эванс. М. : Вильямс, 2010.
8. MySpiro [Electronic resource]. URL: http://www.myspiroo.com/ (дата доступа: 12.06.2016).