Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
Петрозаводский государственный университет
Математический факультет
Кафедра информатики и математического обеспечения

Отчет по дисциплине «Верификация программного обеспечения»

Сервис диагностики сердечных аритмий CardiaCareVerifier

Выполнила:
студентка 6 курса группы 22608
Ю.В. Завьялова

_____
подпись

Лектор:
к.ф–м.н, доцент К. А. Кулаков
Итоговая оценка:

_____
подпись

Петрозаводск
2014

Оглавление

Описание проекта

В данной работе тестируется приложение диагностики сердечных аритмий CardiaCareVerifier, который является частью мобильного сервиса диагностики сердечных аритмий. Приложение написано на Qt+QML+JS.

Основные возможности приложения, которые будут протестированы:

- Получать файлы в формате ATS из файловой системы
- Разбирать файл: выделять заголовок и запись ЭКГ
- Рассчитывать RR– интервалы по записи ЭКГ
- Отображать электрокардиограмму (ЭКГ) с пометками, в местах где обнаружены RR– интервалы

Приложение CardiaCareVerifier планируется использовать в составе web–сервиса диагностики сердечных аритмий CardiaCare. Web–сервис, способный на основе данных кардиограммы получать различные показатели ВСР, используемые в кардиологии, позволит представлять проанализированные данные в удобном для человека виде. На основании этих данных такие специалисты, как физиологи, биофизики, врачи–терапевты и, прежде всего, кардиологи, могут делать выводы о состоянии здоровья пациента. Таким образом, проект направлен на улучшение качества диагностики множества сердечных аритмий и снижению количества случаев внезапной остановки сердца.

Приложение состоит из подсистем в соответствии с рисунком 1. Каждая подсистема представлена в виде класса на языке Qt и будет тестироваться. Список подсистем.

Подробное описание подсистем

Подсистема FileIO - подсистема считывания файла с ЭКГ.

std::vector<double>& read(QString fileName) – получает на входе строку, содержащую путь к файлу ЭКГ в файловой системе, открывает поток для чтения и формирует массив для обработки. Будет проведено блочное и интеграционное тестирование.

Подсистема AliveParser – подсистема, разбирающая содержимое файла в соответствии с входным форматом ATS в соответствии с таблицей 1.

void parseData(std::vector<double>& signal) – Разбирает массив входных данных, проверяет на наличие ЭКГ и сопутствующих данных, заполняет соответствующие поля класса. Будет проведено интеграционное тестирование.

std::vector<double>& readPackageHeader() – считывает заголовок пакета, заполняет соответствующие поля класса. Будет проведено блочное тестирование.

std::vector<double>& readEcgHeader() – считывает заголовок ЭКГ, заполняет соответствующие поля класса. Будет проведено блочное тестирование.

std::vector<double>& readEcgData() – считывает ЭКГ, заполняет соответствующее поле класса. Будет проведено блочное тестирование.

double getLengthECG(std::vector<double>& header) – Считывает длину ЭКГ, заполняет соответствующее поле класса. Будет проведено блочное тестирование.

unsigned getFreqECG(std::vector<double>& header) – Считывает частоту ЭКГ(150 или 300 Гц, заполняет соответствующее поле класса. Будет проведено блочное тестирование.

bool readCheckSum(std::vector<double>& header) – считывает и проверяет контрольную сумму файла ЭКГ, заполняет соответствующее поле класса. Будет проведено блочное тестирование.

Подсистема RPeakDetector – подсистема, выполняющая расчёт RR–интервалов.

void setSignal(unsigned FreqECG, const std::vector<double>& signal) – поиск R–пиков ЭКГ. Будет проведено блочное и интеграционное тестирование.

std::vector<double>& signal() – возвращает массив с ЭКГ. Не тестируется, так как метод только возвращает значение переменной класса.

std::vector<unsigned>& peaks() – возвращает массив RR–интервалов. Не тестируется, так как метод только возвращает значение переменной класса.

Подсистема ECGPainter - подсистема отображения ЭКГ и RR–интервалов.

ECGPainter() – конструктор, инициализирующий отрисовщик. Будет проведено интеграционное, аттестационное тестирование.

drawGrid() – отображает миллиметровую сетку – фон для ЭКГ, исходя из рядов и столбцов. Будет проведено блочное тестирование.

drawSignal (s) – отображение ЭКГ. Будет проведено блочное и интеграционное тестирование.

drawR(s) – отображение RR– интервалов как вертикальные линии. Будет проведено блочное и интеграционное тестирование.

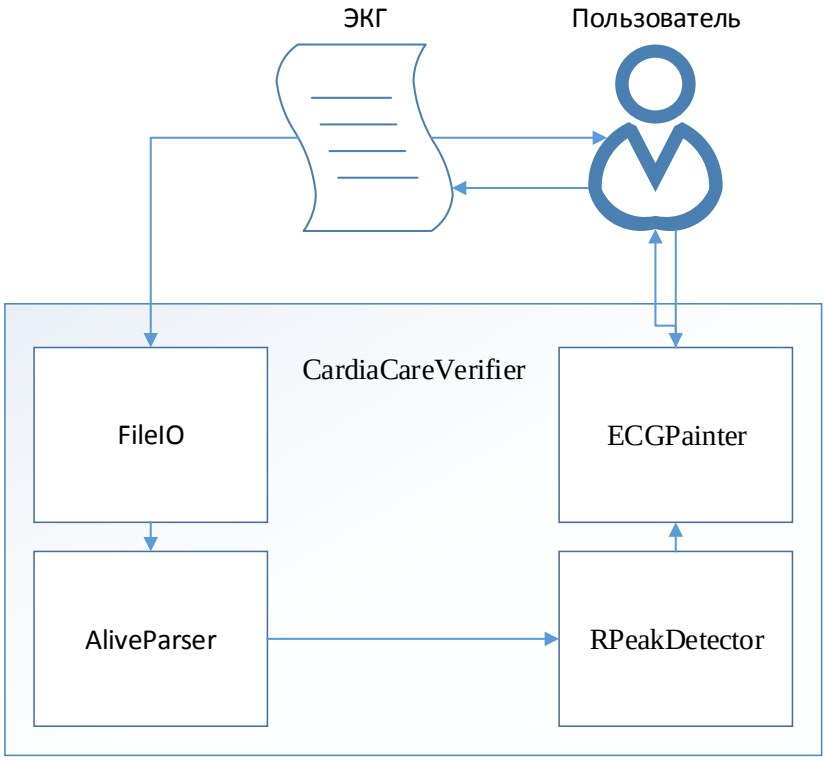

Рисунок 1 - Архитектура приложения.

Packet Structure

Packet Header 6 Bytes	ECG Header 5 Bytes	ECG Data n Bytes	Acc Header 5 Bytes	Acc Data n Bytes	Checksum 1 Byte

Packet Header

Byte	Value	Description
1 - 2	Sync Bytes	Each packet starts with 1st byte 0x00 2nd byte 0xFE
3	Battery Level (%)	0-200 (200 = 100%)
4 - 5	Sequence number Bits 0 – 11	A 12bit number that is incremented for each successive packet.
	Status Bits 12 – 15	Bit 12 – Event Bit 13 – Reserved Bit 14 – Reserved Bit 15 – Reserved
6	Number of data blocks	

ECG Header

Byte	Value	Description
1	ID	0xAA = ECG
2 - 3	Length	Length of ECG header and ECG data in bytes
4	Data Format	0x01 Data Type: 8bit unsigned Sampling rate: 150 samples/s Range: +- 2.66mV 0x02 Data Type: 8bit unsigned Sampling rate: 300 samples/s Range: +- 2.66mV
5	Reserved	Not used

Таблица 2 – Спецификация входного формата ATS.

План тестирования

Приложение состоит из пяти классов, одного файла JS и QML.

Тесты не будут писаться для методов `signal()`, `peaks()` и `categories()`, так как данные методы только возвращает значение переменных соответствующих классов.

Метод `parseData()` выполняет только вызов функции `readPackageHeader()`, `readEcgHeader()`, `readEcgData()`, `readCheckSum()`, для которых будут реализованы блочные тесты, соответственно они также не нуждаются в проведении отдельного тестирования.

Блочное тестирование будет выполняться для методов: `read()`, `readPackageHeader()`, `readEcgHeader()`, `readEcgData()`, `readCheckSum()`, `getLengthECG ()`,`getFreqECG()`,`readCheckSum()`, `setSignal()`, `drawSignal()`, `drawSquare()`, `drawR()`.

Подход к тестированию

В Qt за юнит-тестирование отвечает модуль QTestLib (testlib). Он предоставляет нам набор макросов для тестирования. Поэтому

реализовываются тесты созданием проекта tests в дочерней директории tests основного проекта. В tests лежит класс реализующий тест основного класса.

Интеграционное тестирование

Интеграционное тестирование будет проведено для следующих взаимодействий между методами: `read() и parseData(),readEcgData () и setSignal(),signal () и ECGPainter(),ECGPainter() и drawSignal(),drawSquare(), drawR()`.

Интеграционное тестирование будет проведено над методами класса графического интерфейса приложения `ECGPainter() и drawSignal(),drawSquare(), drawR()`.

Стратегия интеграции

Цель интеграционного тестирования – удостовериться в корректности совместной работы компонент системы.

В данной работе выбран принцип нисходящего тестирования. Он предполагает, что процесс интеграционного тестирования движется следом за разработкой. Сначала тестируют только самый верхний управляющий уровень системы, без модулей более низкого уровня. Затем постепенно с более высокоуровневыми модулями интегрируются более низкоуровневые.

Интеграция методов происходит по схеме взаимодействия в соответствии с рисунком 2. Каждая цифра указывает на номер в последовательности интеграции, а стрелки указывают направление вызовов. Так на первом этапе проверяется взаимодействие методов read и parseData. На втором этапе метод parseData вызывает в себе ряд методов, для которых проведено блочное тестирование, поэтому данный интеграционный тест не понадобится.

На третьем этапе проверяется соответствующим интеграционным тестом взаимодействие методов setSignal и readEcgData.

На четвертом этапе в методе setSignal заполняются переменные класса, а методы signal и peaks возвращают эти значения. Интеграционный тест не требуется.

Пятый этап интеграции – это взаимодействие подсистем расчета с графическим интерфейсом, для данного типа методов будет проведено аттестационное тестирование.

Шестой этап интеграции – взаимодействие методов отрисовки, методы проверяются на этапе блочного, интеграционного и аттестационного тестирования.

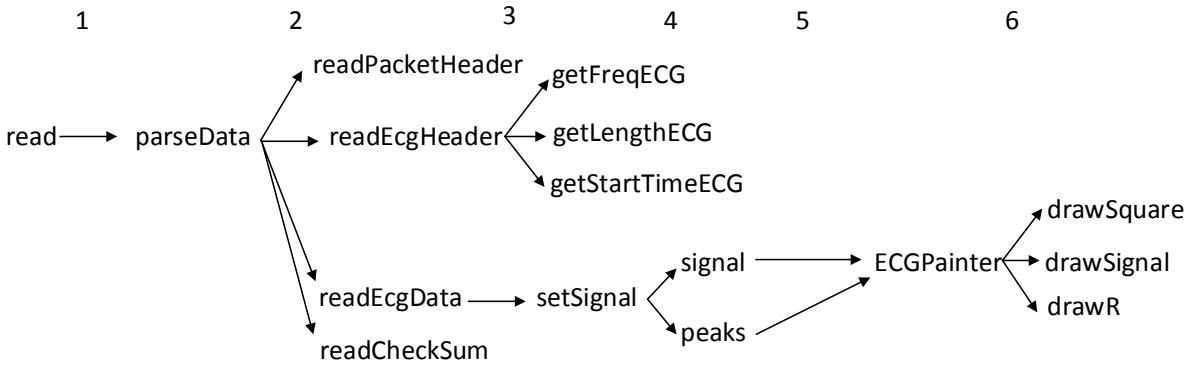

Рисунок 2 – Взаимодействие методов классов.

Выполнение аттестационного тестирования

Аттестация системы будет производится по следующему высокоуровневому сценарию:

1. Запуск приложения
2. Получение окна с полем ввода пути к файлу с ЭКГ
3. Ввод в поле ввода пути к файлу с ЭКГ
4. Получение окна с миллиметровой сеткой, ЭКГ и отметками R-пиков
5. Завершение приложения

Критерий прохождения тестов

Тест считается пройденным, когда выходные данные выполняемого теста совпадают с запланированными данными, описанными в плане тестирования.

Тест считается не пройденным при несовпадении результата или системной ошибке.

Критерий прохождения тестирования

Тестирование считается пройденным, когда набор тестов в совокупности должен обеспечить прохождение каждого метода не менее одного раза. При это тестирование считается успешным при прохождении всех общих тестов и аттестационного тестирования.

Требуемая документация

Тестирование производится по плану, представленному в данном отчет. Результат каждого теста должен заноситься в журнал тестирования, оформленный установленным в данном отчете виде. В случае не прохождения теста ошибку следует занести в журнал ошибок, оформленный установленным в данном отчете виде.

Необходимое оборудование

Для тестирования необходимо выполнение следующих минимальных требований:

- ОС: Windows/Linux/Mac OS X
- CPU: 2.2 GHz
- RAM: 2 GB и более
- ПО: Qt Creator 3.2.1, Qt 5.3.2

Участники тестирования

Тестирование проводится одним тестировщиком по заданному плану тестирования. Ему необходимо владеть Qt 5.3.2 и навыками тестирования. Тестировщик должен быть обеспечен необходимым оборудованием.

Блочные тесты

Тест: 1

Описание: Тест проверяет правильность форматирования массива, представляющего собой файл с ЭКГ для обработки метода std::vector<double>& read(QString fileName).

Входные данные: Строка – путь к файлу с ЭКГ в файловой системе. Файл test_data1.ATS

Косвенные данные: нет.

Ожидаемый результат: Массив inFile типа std::vector<double>, полностью соответствующий массиву в файле test_data1.ATS.

Номер	Входные данные	Тип теста	Результат
1.1	Путь к файлу test_data1.ATS	общий	Массив inFile
1.2	Путь, не соответствующий пути к файлу test_data1.ATS	негативный	Ошибка, исключение ErrorPath
1.3	Путь к файлу test_data1.ATS, нет прав для открытия файла на чтение	негативный	Ошибка, исключение ErrorNoRigts
1.4	Пустая строка	негативный	Ошибка, исключение ErrorNoFile
1.5	Путь к файлу test_data0.ATS, файл пуст	негативный	Ошибка, исключение ErrorFileIsEmpty

Тест: 2

Описание: Тест проверяет правильность форматирования массива, представляющего собой заголовок файла с ЭКГ для обработки метода std::vector<double>& readPackageHeader()

Входные данные: нет. Файлы test_data1.ATS, test_data_2-2.ATS - test_data_2-9.ATS

Косвенные данные: Массив inFile

Ожидаемый результат: Массив PackageHeader типа std::vector<double>, полностью соответствующий массиву в файле test_data1.ATS с 1го по 6й байт.

Номер	Входные данные	Тип теста	Результат
2.1	inFile с заголовком и ЭКГ	общий	Массив PackageHeader
2.2	inFile с заголовком без ЭКГ	общий	Массив PackageHeader
2.3	inFile с первым байтом 0x00FE и контрольной суммой заголовка	общий	Массив PackageHeader
2.4	inFile с произвольным заголовком	негативный	Ошибка, исключение ErrorNotEcg

2.5	inFile с первым байтом 0x00FE и несовпадающей контрольной суммой заголовка	негативный	ошибка, исключение ErrorNotEcg
2.6	inFile с первым байтом 0x00FE и заголовком короче, чем должен быть	негативный	ошибка, исключение ErrorNotEcg
2.7	inFile с первым байтом не 0x00FE и заголовком короче, чем должен быть	негативный	ошибка, исключение ErrorNotEcg
2.8	Пустой массив inFile	негативный	ошибка, исключение ErrorNotEcg
2.9	Полностью произвольный массив inFile (не формата ATS)	негативный	ошибка, исключение ErrorNotEcg
2.10	Запрет на чтение массива inFile	негативный	ошибка, исключение ErrorNoRights

Тест: 3

Описание: Тест проверяет правильность форматирования массива, представляющего собой заголовок ЭКГ метода std::vector<double>& readEcgHeader()

Входные данные: нет. Файл test_data1.ATS

Косвенные данные: Массив PackageHeader

Ожидаемый результат: Массив EcgHeader типа std::vector<double>, полностью соответствующий массиву в файле test_data1.ATS с 7го по 11й байт.

Номер	Входные данные	Тип теста	Результат
3.1	PackageHeader с заголовком ЭКГ	общий	Массив EcgHeader
3.2	PackageHeader с первым байтом 0xAA и длиной не равной 5 байтам	негативный	ошибка, исключение ErrorNotEcg
3.3	Пустой массив PackageHeader	негативный	ошибка, исключение ErrorNotEcg
3.4	Полностью произвольный массив PackageHeader	негативный	ошибка, исключение ErrorNotEcg

Тест: 4

Описание: Тест проверяет правильность форматирования массива, представляющего собой длину ЭКГ метода double getLengthECG(std::vector<double>& header)

Входные данные: header – заголовок ЭКГ

Косвенные данные: нет

Ожидаемый результат: Число LengthECG – длина ЭКГ, полностью соответствующий значению в файле test_data1.ATS с 7го по 8й байт.

Номер	Входные данные	Тип теста	Результат
4.1	header с первым байтом 0xAA и длиной равной 5 байтам	общий	Число LengthECG
4.2	header с первым байтом не 0xAA и длиной равной 5 байтам	негативный	ошибка, исключение ErrorNotEcg
4.3	header с первым байтом 0xAA и длиной больше	негативный	ошибка, исключение ErrorNotEcg
4.4	header с первым байтом 0xAA и длиной не равной 5 байтам	негативный	ошибка, исключение ErrorNotEcg
4.5	Пустой массив header	негативный	Ошибка, исключение ErrorNoEcgLength

Тест: 5

Описание: Тест проверяет правильность форматирования массива, представляющего собой ЭКГ метода std::vector<double>& readEcgData()

Входные данные: нет.

Косвенные данные: Массив inFile, число LengthECG

Ожидаемый результат: Массив EcgData типа std::vector<double>, полностью соответствующий значению в файле test_data1.ATS с 12го байт длины LengthECG.

Номер	Входные данные	Тип теста	Результат
5.1	inFile с заголовком и ЭКГ	общий	Массив EcgData
5.2	inFile с заголовком без ЭКГ	негативный	Ошибка, исключение

			ErrorNotEcg
5.3	inFile с первым байтом 0x00FE и контрольной суммой заголовка	негативный	Массив EcgData
5.4	inFile с произвольным заголовком	негативный	Ошибка, исключение ErrorNotEcg
5.5	inFile с первым байтом 0x00FE и несовпадающей контрольной суммой заголовка	негативный	Ошибка, исключение ErrorNotEcg
5.6	inFile с первым байтом 0x00FE и заголовком короче, чем должен быть	негативный	Ошибка, исключение ErrorNotEcg
5.7	inFile с первым байтом не 0x00FE и заголовком короче, чем должен быть	негативный	Ошибка, исключение ErrorNotEcg
5.8	Пустой массив inFile	негативный	Ошибка, исключение ErrorNotEcg
5.9	Полностью произвольный массив inFile (не формата ATS)	негативный	Ошибка, исключение ErrorNotEcg
5.10	Запрет на чтение массива inFile	негативный	Ошибка, исключение ErrorNoRights

Тест: 6

Описание: Тест проверяет правильность форматирования массива, представляющего собой частота записи ЭКГ метода unsigned getFreqECG(std::vector<double>& header)

Входные данные: header – заголовок ЭКГ

Косвенные данные: нет

Ожидаемый результат: Число FreqECG– частота записи ЭКГ, полностью соответствующий значению в файле test_data1.ATS 9й байт.

Номер	Входные данные	Тип теста	Результат
6.1	header с первым байтом 0xAA и длиной равной 5 байтам	общий	Число FreqECG
6.2	header с первым байтом не 0xAA и длиной равной 5 байтам	негативный	ошибка, исключение ErrorNotEcg
6.3	header с первым байтом	негативный	ошибка,

	0xAA и длиной больше		исключение ErrorNotEcg
	header с первым байтом 0xAA и длиной не равной 5 байтам	негативный	ошибка, исключение ErrorNotEcg
6.4	Пустой массив header	негативный	Ошибка, исключение ErrorNoFreqECG

Тест: 7

Описание: Тест проверяет правильность форматирования массива, представляющего собой частота записи ЭКГ метода bool readCheckSum(std::vector<double>& header)

Входные данные: Массив inFile

Косвенные данные: нет

Ожидаемый результат: true – совпадение контрольной суммы, полностью соответствующий значению последнего байта в файле test_data1.ATS, false – иначе.

Номер	Входные данные	Тип теста	Результат
7.1	inFile с заголовком и ЭКГ	общий	true
7.2	inFile с заголовком без ЭКГ	негативный	true
7.3	inFile с первым байтом 0x00FE и контрольной суммой заголовка	негативный	true
7.4	inFile с произвольным заголовком	негативный	false
7.5	inFile с первым байтом 0x00FE и несовпадающей контрольной суммой заголовка	негативный	false
7.6	inFile с первым байтом 0x00FE и заголовком короче, чем должен быть	негативный	false
7.7	inFile с первым байтом не 0x00FE и заголовком короче, чем должен быть	негативный	false
7.8	Пустой массив inFile	негативный	true
7.9	Полностью произвольный массив inFile (не формата ATS)	негативный	true
7.10	Запрет на чтение массива inFile	негативный	ошибка, исключение ErrorNoRights

Тест: 8

Описание: Тест проверяет правильность нахождения R-пиков метода void setSignal(unsigned FreqECG, const std::vector<double>& signal)

Входные данные: FreqECG – частота записи ЭКГ, signal – ЭКГ из файла test_data1.ATS.

Косвенные данные: нет

Ожидаемый результат: заполнения массива std::vector<unsigned> identified_peaks, полностью соответствующий файлу test_ identified_peaks.ATS.

Номер	Входные данные	Тип теста	Результат
8.1	Signal и FreqECG	общий	Массив identified_peaks
8.2	Signal – пустой и FreqECG	негативный	ошибка
8.3	Signal и FreqECG, не соответсвующий 9му байту в файле test_data1.ATS	негативный	Массив identified_peaks, не соответсвующий test_ identified_peaks.ATS.
8.4	Signal, не соответствующий test_data1.ATS, и FreqECG	негативный	Массив identified_peaks, не соответсвующий test_ identified_peaks.ATS.
8.5	Signal и FreqECG – пустой	негативный	Ошибка, исключение ErrorFregECG
8.6	Signal и FreqECG – 0	негативный	Массив identified_peaks, не соответсвующий test_ identified_peaks.ATS.
8.7	Signal и FreqECG – отрицательный	негативный	Массив identified_peaks, не соответсвующий test_ identified_peaks.ATS.

Тест: 9

Описание: Тест проверяет правильность отрисовки миллиметровой сетки метода void drawGrid().

Входные данные: нет

Косвенные данные: context - контекст отрисовщика

Ожидаемый результат: отображение сетки

Номер	Входные данные	Тип теста	Результат
9.1	Созданный context	общий	отображение сетки
9.2	Не созданный context	негативный	Ошибка, исключение ErrorNoContext

Тест: 10

Описание: Тест проверяет правильность отрисовки ЭКГ миллиметровой сетки метода void drawSignal (s).

Входные данные: s – сигнал ЭКГ

Косвенные данные: context - контекст отрисовщика

Ожидаемый результат: отображение ЭКГ, полностью соответствующий файлу test_ data1.ATS.

Номер	Входные данные	Тип теста	Результат
10.1	s	общий	отображение ЭКГ
10.2	s - пустой	негативный	Ошибка, исключение ErrorNoECG
10.3	s, заполненный произвольными числами	негативный	отображение ЭКГ
10.4	Созданный context	общий	отображение ЭКГ
10.5	Не созданный context	негативный	Ошибка, исключение ErrorNoContext

Тест: 11

Описание: Тест проверяет правильность отрисовки ЭКГ миллиметровой сетки метода void drawR (s).

Входные данные: s – массив индексов, в которых расположены R-пики.

Косвенные данные: context - контекст отрисовщика

Ожидаемый результат: отображение ЭКГ, полностью соответствующий файлу test_ identified_peaks.ATS.

Номер	Входные данные	Тип теста	Результат
11.1	s	общий	отображение R-пик
11.2	s - пустой	негативный	Ошибка, исключение ErrorNoECG
11.3	s, заполненный произвольными числами	негативный	отображение R-пик
11.4	Созданный context	общий	отображение сетки
11.5	Не созданный context	негативный	Ошибка, исключение ErrorNoContext

drawR(s) – отображение RR– интервалов как вертикальные линии.

Будет проведено аттестационное тестирование.

Интеграционные тесты

Тест: 12

Описание: Тест проверяет правильность взаимодействия чтения файла и разбора его содержимого между методами std::vector<double>& read(QString fileName) и void parseData(std::vector<double>& signal)

Входные данные: Строка – путь к файлу с ЭКГ в файловой системе. Файл test_data1.ATS

Косвенные данные:

Ожидаемый результат: заполнены переменные: PackageHeader с 1го по 6й байт, ECGHeader с 7го по 11й байт, EcgData с 12го байт длины LengthECG с 7го по 8й байт, FreqECG 9й байт, CheckSum последнего байта в файле test_data1.ATS.

Номер	Входные данные	Тип теста	Результат
12.1	inFile с заголовком и ЭКГ	общий	Массив и числа
12.2	inFile с заголовком без ЭКГ	общий	Числа
12.3	inFile с первым байтом 0x00FE и	общий	Массив и

	контрольной суммой заголовка		числа
12.4	inFile с произвольным заголовком	негативный	Ошибка, исключение ErrorNotEcg
12.5	inFile с первым байтом 0x00FE и несовпадающей контрольной суммой заголовка	негативный	ошибка, исключение ErrorNotEcg
12.6	inFile с первым байтом 0x00FE и заголовком короче, чем должен быть	негативный	ошибка, исключение ErrorNotEcg
12.7	inFile с первым байтом не 0x00FE и заголовком короче, чем должен быть	негативный	ошибка, исключение ErrorNotEcg
12.8	Пустой массив inFile	негативный	ошибка, исключение ErrorNotEcg
12.9	Полностью произвольный массив inFile (не формата ATS)	негативный	ошибка, исключение ErrorNotEcg
12.10	Запрет на чтение массива inFile	негативный	ошибка, исключение ErrorNoRights

Тест: 13

Описание: Тест проверяет правильность взаимодействия получения ЭКГ из файла и поиском R-пиков, методамы и std::vector<double>& readEcgData() и void setSignal (unsigned FreqECG, const std::vector<double>& signal)

Входные данные: нет.

Косвенные данные: Массив inFile, число LengthECG

Ожидаемый результат: заполнен массив R-пиков identified_peaks

Номер	Входные данные	Тип теста	Результат
13.1	inFile с заголовком и ЭКГ	общий	Массив identified_peaks
13.2	inFile с заголовком без ЭКГ	негативный	Ошибка, исключение ErrorNotEcg
13.3	inFile с первым байтом 0x00FE и контрольной суммой заголовка	негативный	Массив identified_peaks, не соответствующий

			файлу
13.4	inFile с произвольным заголовком	негативный	ошибка, исключение ErrorNotEcg
13.5	inFile с первым байтом 0x00FE и несовпадающей контрольной суммой заголовка	негативный	ошибка, исключение ErrorNotEcg
13.6	inFile с первым байтом 0x00FE и заголовком короче, чем должен быть	негативный	ошибка, исключение ErrorNotEcg
13.7	inFile с первым байтом не 0x00FE и заголовком короче, чем должен быть	негативный	ошибка, исключение ErrorNotEcg
13.8	Пустой массив inFile	негативный	ошибка, исключение ErrorNotEcg
13.9	Полностью произвольный массив inFile (не формата ATS)	негативный	ошибка, исключение ErrorNotEcg
13.10	Запрет на чтение массива inFile	негативный	ошибка, исключение ErrorNoRights

Тест: 14

Описание: Тест проверяет правильность взаимодействия конструктора ECGPainter и методом void drawSignal (s).

Входные данные: нет

Косвенные данные: нет

Ожидаемый результат: В окне приложения отображается миллиметровая сетка, ЭКГ.

Номер	Входные данные	Тип теста	Результат
14.1	s	общий	отображение ЭКГ
14.2	s - пустой	негативный	Ошибка, исключение ErrorNoECG
14.3	s, заполненный произвольными числами	негативный	отображение ЭКГ

Тест: 15

Описание: Тест проверяет правильность взаимодействия конструктора ECGPainter и методом void drawR(s).

Входные данные: нет

Косвенные данные: нет

Ожидаемый результат: В окне приложения отображается миллиметровая сетка, R-пики, полностью соответствуют файлу test_ identified_peaks.ATS

Номер	Входные данные	Тип теста	Результат
15.1	s	общий	отображение R-пик
15.2	s - пустой	негативный	Ошибка, исключение ErrorNoECG
15.3	s, заполненный произвольными числами	негативный	отображение R-пик

Аттестационные тесты

Тест: 16

Описание: запуск приложения на эмуляторе

Входные данные: нет

Косвенные данные: нет

Ожидаемый результат: Запуск приложения, Отображение на экране главного меню в портретной ориентации

Тест: 17

Описание: Проверка работы R-пик детектора

Входные данные: Строка – путь к файлу с ЭКГ в файловой системе. Файл test_data1.ATS

Косвенные данные: нет.

Ожидаемый результат: В окне приложения отображается миллиметровая сетка, ЭКГ, R-пики, полностью соответствуют экг – из файла test_data1.ATS, R-пики – из файла test_ identified_peaks.ATS

Номер	Входные данные	Тип теста	Результат
17.1	Путь к файлу test_data1.ATS	общий	Соответствует ожидаемому результату
17.2	Путь, не соответствующий пути к файлу test_data1.ATS	негативный	Ошибка, исключение

			ErrorPath
17.3	Путь к файлу test_data1.ATS, нет прав для открытия файла на чтение	негативный	Ошибка, исключение ErrorNoRigts
17.4	Пустая строка	негативный	Ошибка, исключение ErrorNoFile
17.5	Путь к файлу test_data0.ATS, файл пуст	негативный	Ошибка, исключение ErrorFileIsEmpty

Отчет о проведении тестирования

Результаты тестирования

Дата	Номера выполненных тестов	Количество пройденных тестов	Количество не пройдённых тестов	Номера ошибок
13.01.2015	1.1 – 1.5 2.1 – 2.10	13	2	1, 2
14.01.2015	3.1 – 3.4 4.1 – 4.5 5.1 – 5.10	18	1	3
15.01.2015	6.1 – 6.4 7.1 – 7.10	13	1	4
16.01.2015	8.1 – 8.7 9.1 – 9.2 2.8, 3.8, 5.8, 7.8	13	0	
17.01.2015	10.1 – 10.5 11.1 – 11.5	10	0	
19.01.2015	12.1 – 12.10 13.1 – 13.10 14.1 – 14.3 15.1 – 15.3	26	0	
20.01.2015	16 17.1 – 17.5	6		
Всего:		99	4	

Пример теста

Блочное тестирование метода read() подсистемы FileIO с помощью Qt Test. Модуль тестирования состоит из одного класса Test_Fileio.

test_fileio.h – заголовочный файл:

```
#ifndef TEST_FILEIO_H
#define TEST_FILEIO_H
```

```
#include <QObject>

class Test_Fileio : public QObject
{
    Q_OBJECT
public:
    explicit Test_Fileio(QObject *parent = 0);

    QString test_data1 = "+-0.345+-0.345+…";

private slots:
    void read();

};

#endif // TEST_FILEIO_H
```

test_fileio.c – файл, реализующий тесты для метода read:

```
#include <QTest>
#include "test_fileio.h"
#include "fileio.h"
Test_Fileio::Test_Fileio(QObject *parent) :
    QObject(parent)
{
}
void Test_Fileio::read(){
    FileIO inFile;

    // test 1.1
QCOMPARE(inFile.read("C:/Users/Yulia/Downloads/CardiaCareDVS/CardiaCareDVS/te
st_data1.ATS"), this->test_data1);

    // test 1.2
QCOMPARE(inFile.read("C:/Userd/Yulia/Downloads/CardiaCareDVS/CardiaCareDVS/te
st_data1.ATS"), QString());

    // test 1.3
QCOMPARE(inFile.read("C:/Users/Yulia/Downloads/CardiaCareDVS/CardiaCareDVS/te
st_data_no_rights.ATS"), QString());

    // test 1.4
QCOMPARE(inFile.read("C:/Users/Yulia/Downloads/CardiaCareDVS/CardiaCareDVS/te
st_data0.ATS"), QString());

    // test 1.5
    QCOMPARE(inFile.read(""), QString());
}
```

Тестирование вызывается в mail.c

```
#include <QApplication>
#include <QQmlApplicationEngine>
#include <QtQml>
#include <QTest>
#include <iostream>
#include <cstdlib>
#include <cstdio>
#include "test_fileio.h"
#include "fileio.h"

int main(int argc, char *argv[])
{
```

```
    freopen("testing.log", "w", stdout);

    QApplication app(argc, argv);

    QTest::qExec(new Test_Fileio, argc, argv);

    QQmlApplicationEngine engine;
    qmlRegisterType<FileIO, 1>("FileIO", 1, 0, "FileIO");
    engine.load(QUrl(QStringLiteral("qrc:/main.qml")));
    //engine.

    return app.exec();
}
```

Результат заносится в файл testing.log:

********* Start testing of Test_Fileio *********
Config: Using QtTest library 5.3.2, Qt 5.3.2
PASS   : Test_Fileio::initTestCase()
QDEBUG : Test_Fileio::read() Unable to open the file
QDEBUG : Test_Fileio::read() Unable to open the file
FAIL! : Test_Fileio::read() Compared values are not the same
   Actual (inFile.read("C:/Users/Yulia/Downloads/CardiaCareDVS/CardiaCareDVS/test_data0.ATS")): "+"
   Expected (QString())   : ""
..\CardiaCareDVS\test_fileio.cpp(23) : failure location
PASS   : Test_Fileio::cleanupTestCase()
Totals: 2 passed, 1 failed, 0 skipped
********* Finished testing of Test_Fileio *********

В файле представлены все исключения (QDEBUG), которые были обработаны и ошибки. Подробное описание ошибки см. «Отчет об ошибках» Ошибка 1.

Отчет об ошибках

Ошибка 1

Краткое описание: ошибка возникла в методе readr тест 2.8 по запуску метода с пустым файлом на входе.

Подробное описание: вместо ожидаемого результат – обработки исключения ErrorFileIsEmpty, был создан пустой массив, так как проверка isEmpty не выполняется.

Ошибка 2

Краткое описание: ошибка возникла в методе readPackageHeader тест 3.8 по запуску метода с пустым массивом на входе.

Подробное описание: вместо ожидаемого результат – обработки исключения ErrorNotEcg, был создан пустой массив, так как проверка isEmpty не выполняется.

Проверяемость: всегда

Ошибка 3

Краткое описание: ошибка возникла в методе readEcgData тест 5.8 по запуску метода с пустым массивом на входе

Подробное описание: вместо ожидаемого результат – обработки исключения ErrorNotEcg, обращение к несуществующей области памяти.

Проверяемость: всегда

Ошибка 4

Краткое описание: ошибка возникла в методе readCheckSum тест 7.8 по запуску метода с пустым массивом на входе

Подробное описание: вместо ожидаемого результат – обработки исключения ErrorNotEcg, обращение к несуществующей области памяти.

Проверяемость: всегда

Покрытие кода тестами

Количество строк кода тестируемых модулей: 1095;

Количество строк кода, покрытых тестами 1085;

Покрытие = 99%

№	Название подсистемы	Количество строк кода	Количество строк кода, покрытых тестами	Процент покрытия
1	FileIO	101	101	100%
2	AliveParser	288	288	100%
3	RPeakDetector	194	184	95%
4	ECGPainter	512	512	100%
Всего:		1095	1085	99%

Список литературы

1. Чуян Е. Н., Бирюкова Е. А., Равагаева М. Ю. Физиологические механизмы вариабельности сердечного ритма / Чуян Е. Н. // Ученые записки Таврического Национального Университета им. В. И. Вернадского. Серия "Биология, химия". — 2008. — Т.21(60). — №3. — 168-89 с.

2. Березный Е. А., Рубин А. М., Утехина Г. А. Практическая кардиоритмография / Березный Е. А. — С-Пб.: Нео. — 2005 г. — 140 с.

3. Кошкин И. В. Аритмии. Методическое пособие / Кошкин И. В. — Набережные Челны: Набережночелнинский медицинский колледж. —2003 г. — 135 с.

4. Авилова Л. И. Longitudinal changes and prognostic significance of cardiovascular autonomic regulation assessed by heart rate variability and analysis of non-linear heart rate dynamics: 2.7. Correlations between different measures of heart rate variability and heart rate dynamics [Электронный ресурс]. — Режим доступа: http://herkules.oulu.fi/ isbn9514272005/html/x307.html (дата обращения: 08.12.2012)

5. Баевский Р. М. и др. Анализ вариабельности сердечного ритма при использовании различных электрокардиографических систем: метод. рекомендации / Р. М. Баевский. — М. — 2002 г. — 53 с.

6. Баевский Р. М., Кириллов О. И., Клецкин С. З. Математический анализ изменений сердечного ритма при стрессе / Баевский Р. М. — М.: Наука. — 1984 г.

7. Bailey J. J., Berson A. S., Garson A. Jr. Recommendations for standardization and specifications in automated electrocardiography.Circulation / Bailey J. J. — 1990 г. — 730-9 с.

8. Машин В.А. Зависимость показателей вариабельности сердечного ритма от средней величины RR-интервалов / Машин В.А. // Российский физиологический журнал им. И.М. Сеченова. — Т. 88. — №7. 2002 г. — 851-55 с.

9. Schwartz PJ, Priori SG. Sympathetic nervous system and cardiac arrhythmias / Schwartz PJ // Zipes DP, Jalife J, eds. Cardiac Electrophysiology. From Cell to Bedside. — Philadelphia: W.B. Saunders. 1990 г. — 330-43 с.

10.Вариабельность сердечного ритма - ВСР :: НИЦ БКБ [Электронный ресурс]. — Режим доступа: http://www.hrv.rcbkb.com/2010/05/klinicheskoe-znachenie-issledovanija_7695.html

11.Нормальные значения параметров вариабельности сердечного ритма [Электронный ресурс]. — Режим доступа: http://www.hrv.ru/standart/prila.html

12.Нормальные значения параметров вариабельности сердечного ритма [Электронный ресурс]. — Режим доступа: http://ru.scribd.com/doc/48803584

13.Glotov I. N., Ovsyannikov S. V., Trenkaev V. N. About distributed database management system based on MariaDB server //Prikladnaya Diskretnaya Matematika. Supplement. – 2012. – №. 5. – С. 104-106.