

Санкт-Петербургский Политехнический Университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Название предмета

Отчет по лабораторной работе

Тема работы

Транспортное расписание

Работу выполнил:

Мальцев М.С.

Группа: 13501/4

Преподаватель:

Вылегжанина К.Д.

Санкт-Петербург
2016

Содержание

1	Транспортное расписание	2
1.1	Задание	2
1.2	Концепция	2
1.3	Минимально работоспособный продукт	2
1.4	Диаграмма прецедентов использования	2
1.5	Диаграмма последовательностей	3
2	Проектирование приложения	3
2.1	Архитектуру приложения	3
2.2	Диаграмма компонентов	4
2.3	Файлы создаваемые в процессе работы приложения	4
2.4	Интерфейс ядра	4
3	Реализация Транспортного расписания	5
3.1	Используемые версии	5
3.2	Основные классы	5
3.3	Скриншоты основных экранов пользовательского интерфейса	6
4	Процесс обеспечения качества и тестирование	7
4.1	Список	7
4.2	Листинг	8
4.3	Частичный листинг	8
4.4	Таблица	8
5	Выводы	8

1 Транспортное расписание

1.1 Задание

Реализовать проект Транспортное расписание

"Транспортное расписание программа позволяющая создать, редактировать и использовать расписание для поездов метрополитена

1.2 Концепция

Программа должна предоставлять обычному пользователю возможность просмотра маршрута поездов, показывать информацию о станций и помогать найти способ проезда до нужной станции. У администратора в отличие от обычного пользователя должны присутствовать права на редактирования данных.

1.3 Минимально работоспособный продукт

Программа, которая позволяет пользователю просмотреть маршруты поездов и информацию о станций

1.4 Диаграмма прецедентов использования

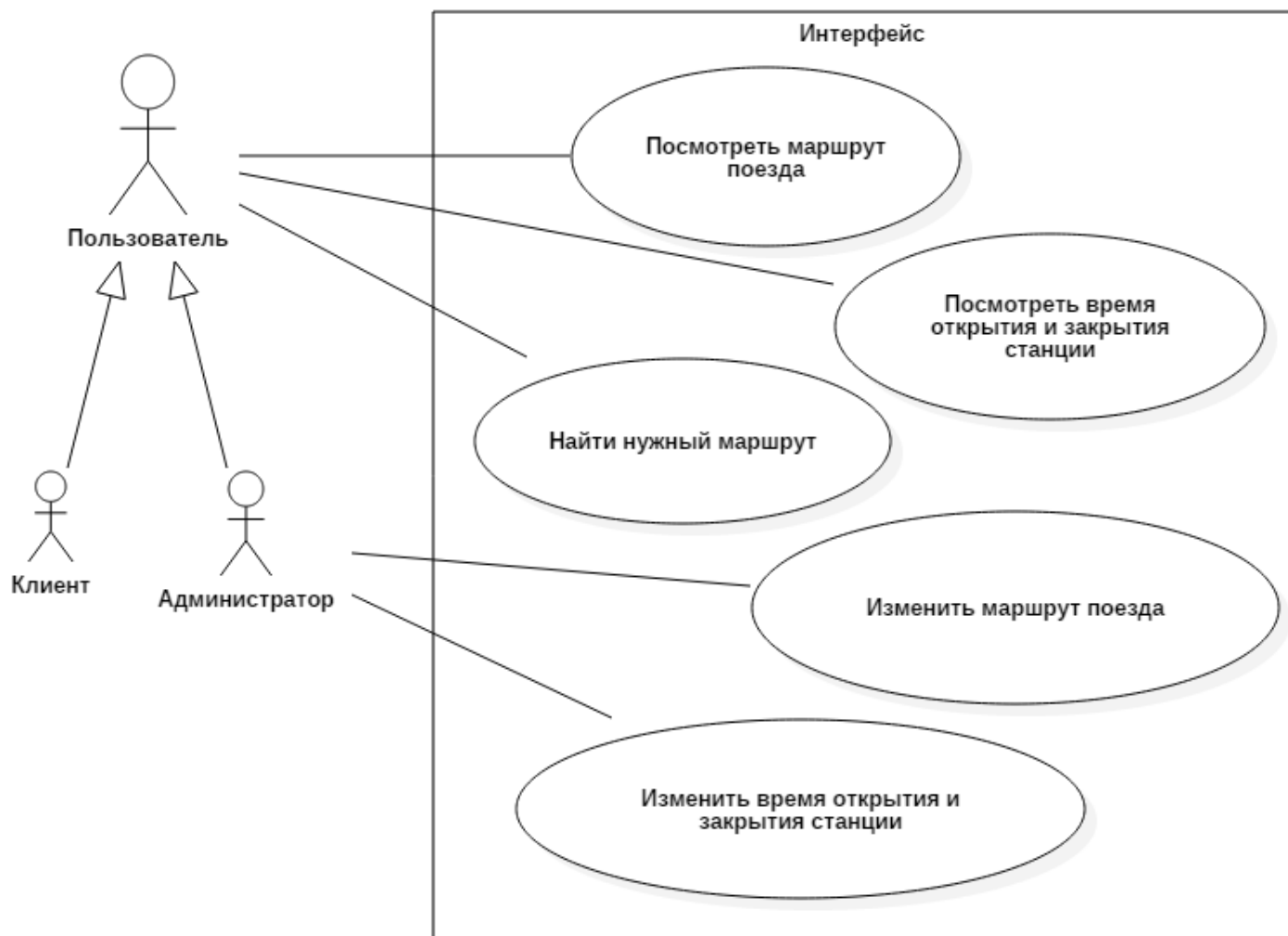


Рис. 1: Диаграмма прецедентов использования

1.5 Диаграмма последовательностей

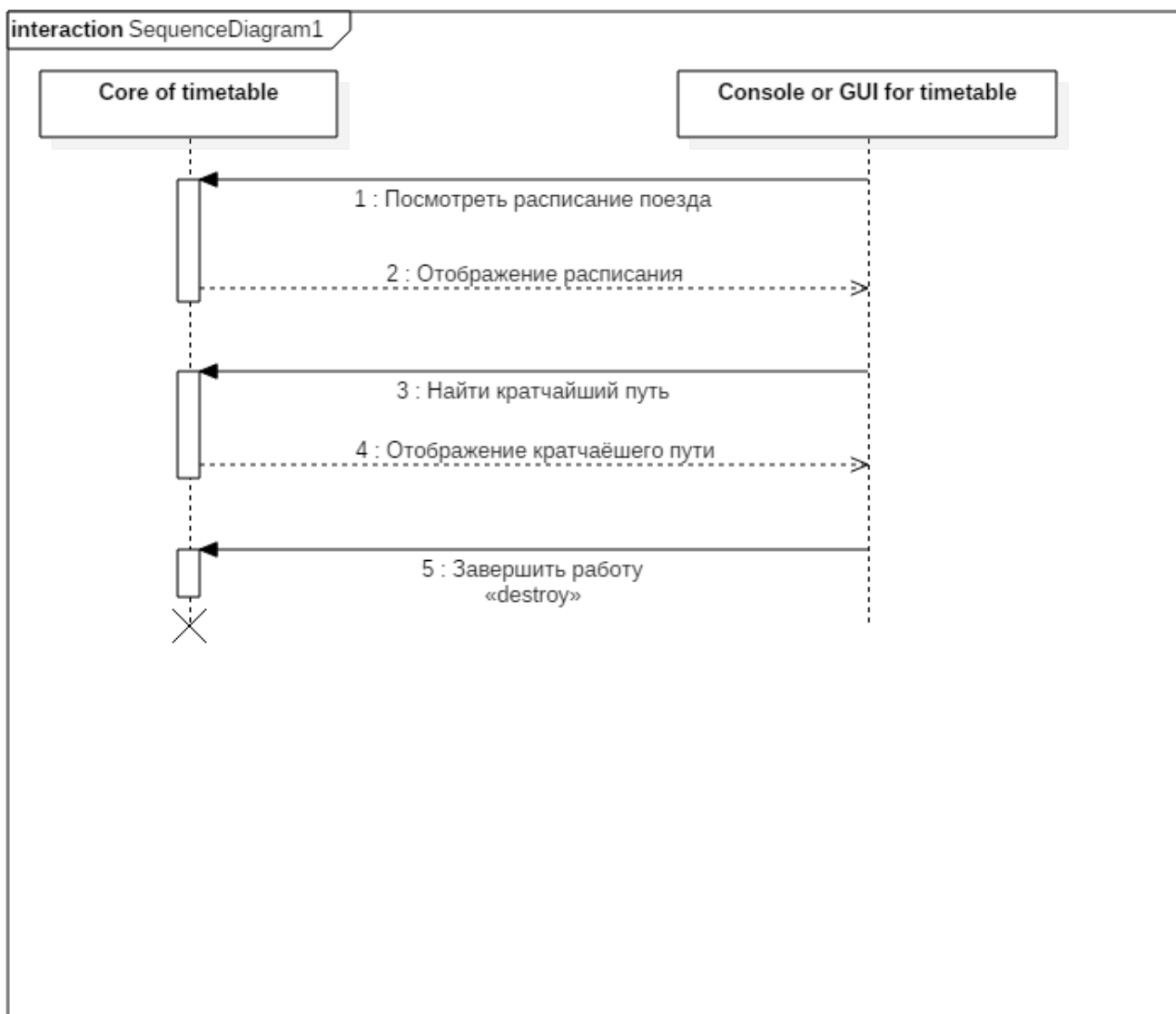


Рис. 2: Диаграмма последовательностей

2 Проектирование приложения

2.1 Архитектуру приложения

Было решено выделить 4 подпроекта:

1. Консольное приложение - подпроект, цель которого предоставить пользователю функциональности ядра с помощью консоли
2. Библиотека - подпроект, содержащий основную бизнес-логику всего проекта
3. Графическое приложение - подпроект, созданный для того, чтобы с помощью графического интерфейса предоставить пользователю функциональности ядра
4. Тесты - подпроект, созданный для того, чтобы тестировать библиотеку, содержащую основную бизнес-логику

2.2 Диаграмма компонентов

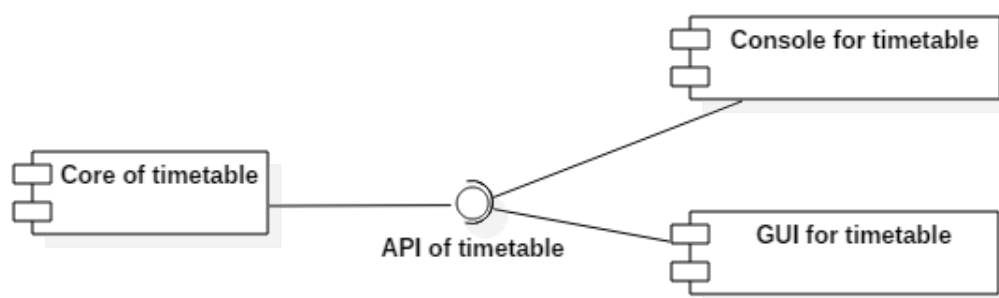


Рис. 3: Диаграмма компонентов

2.3 Файлы создаваемые в процессе работы приложения

Называются файлы могут как угодно пользователю.

Формат данных должен быть следующий:

Для маршрутов: Devyatkino,Grazhdansky Prospekt/Parnas,Prospekt Prosvescheniya

Для станций: Parnas 5.47-0.00/Prospekt Prosvescheniya 5.37-0.40

2.4 Интерфейс ядра

В библиотека предоставляет следующую функциональность:

1. `void putInfoAboutMetro(const std::string &infoAboutRoutes, const std::string &infoAboutStations) noexcept;`

Один из способ передать входные данные, на вход принимаются две строки в определённом формате, которые ядро будет парсить, а потом передаст классам, отвечающим за хранение информации

2. `void loadInfoFromFile(const std::string &name_of_the_file_with_route, const std::string &name_of_the_file_with_station);`

Ввод данных с помощью файлов. В этот метод передаются два параметра, это название файлов, один файл - информация о маршрутах, второй о станциях. Информация, как и в предыдущем методе должна находится в определённом формате

3. `int howManyRoutes() const noexcept;`

Возвращает информацию о том, сколько маршрутов существуетна данный момент

4. `std::vector<std::string> getRoute(const int number_of_the_route);`

Возвращает запрашиваемый маршрут

5. `std::string getInfoAboutStation(const std::string &name_of_the_station);`

Возвращает информацию о запрашиваемой станции

6. `std::string getInfoAboutStation(const int number_of_the_route, const int number_of_the_station);`

Альтернативный способ получения информации о станции

7. `void changeStationInRoute(const int number_of_the_route, const int number_of_the_station, const std::string &new_marking);`

Изменить название станции в маршруте

8. `void addStationInRoute(const int number_of_the_route, const std::string &what_to_add);`
Добавить новую станцию в маршрут
9. `int addRoute() noexcept;`
Добавляет новый маршрут
10. `void deleteStationFromRoute(const int number_of_the_route, const int number_of_the_station);`
Удаляет станцию из маршрута
11. `void deleteRoute(const int number_of_the_route);`
Удаляет маршрут
12. `void addInfoAboutStation(const std::string &name_of_the_station, const std::string &station_description) noexcept;`
Добавление информации о станции
13. `void addInfoAboutStation(const int number_of_the_route, const int number_of_the_station, const std::string &station_description);`
Альтернативный способ добавления информации о станции
14. `void removeInfoAboutStation(const std::string &what_station_to_remove);`
Удаление информации о станции
15. `void removeInfoAboutStation(const int number_of_the_station);`
Альтернативный способ удаления информации о станции
16. `std::vector<std::pair<std::string, std::string>> getAllStationsWhichHaveDescription() noexcept;`
Возвращает станции к которым существует описание
17. `void saveChanges(const std::string &name_of_the_file_with_route, const std::string &name_of_the_file_with_station) noexcept;`
Сохраняет все изменения в файлы

3 Реализация Транспортного расписания

3.1 Используемые версии

- Qt Creator 4.0.0 (opensource)
- Стандарт c++11
- GCC 5.4.2 и GCC 5.6.0
- Операционная система: Windows 10
- cprcheckgui версии 1.7.2

3.2 Основные классы

Из библиотеке хотелось бы упомянуть про следующие классы:

- `CoreOfInfoAboutMetro` - класс, который отвечает за перенаправление задач к подконтрольным ему классам
- `ParsingInfo` - класс, цель которого парсинг информации

- RoutesInfo - класс, отвечающий за обработку информации связанной с маршрутами
- StationsInfo - класс, отвечающий за обработку информации связанной со станциями

3.3 Скриншоты основных экранов пользовательского интерфейса

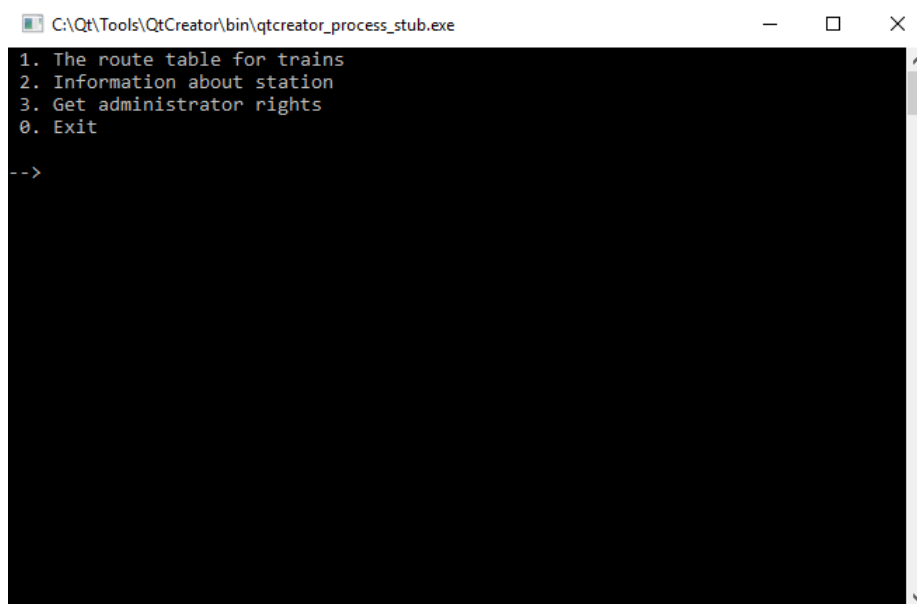
A screenshot of a console window titled "C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe". The window contains a menu with the following options: "1. The route table for trains", "2. Information about station", "3. Get administrator rights", and "0. Exit". Below the menu, there is a prompt "-->" followed by a cursor.

Рис. 4: Меню консольного приложения

Здесь предоставлены основные функциональность предлагаемые пользователю:

- Получить информацию о маршруте
- Получить информацию о станции
- Выход

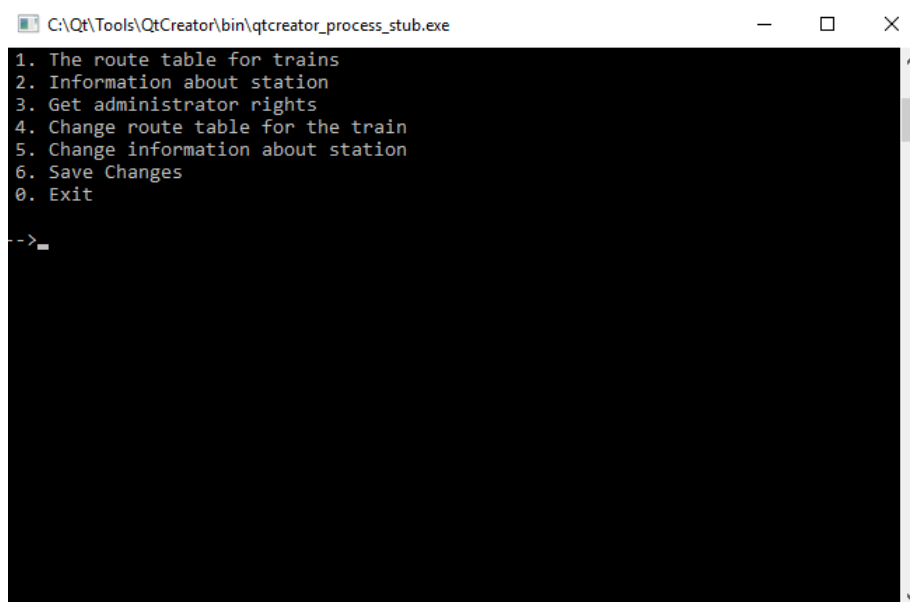
A screenshot of a console window titled "C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe". The window contains an expanded menu with the following options: "1. The route table for trains", "2. Information about station", "3. Get administrator rights", "4. Change route table for the train", "5. Change information about station", "6. Save Changes", and "0. Exit". Below the menu, there is a prompt "-->" followed by a cursor.

Рис. 5: Меню консольного приложения, расширенное для администрирования

Кроме предидущих для администратора становятся доступны функциональности связанные с редактированием:

- Изменить информацию о маршрутах
- Изменить информацию о станциях
- Сохранить изменения

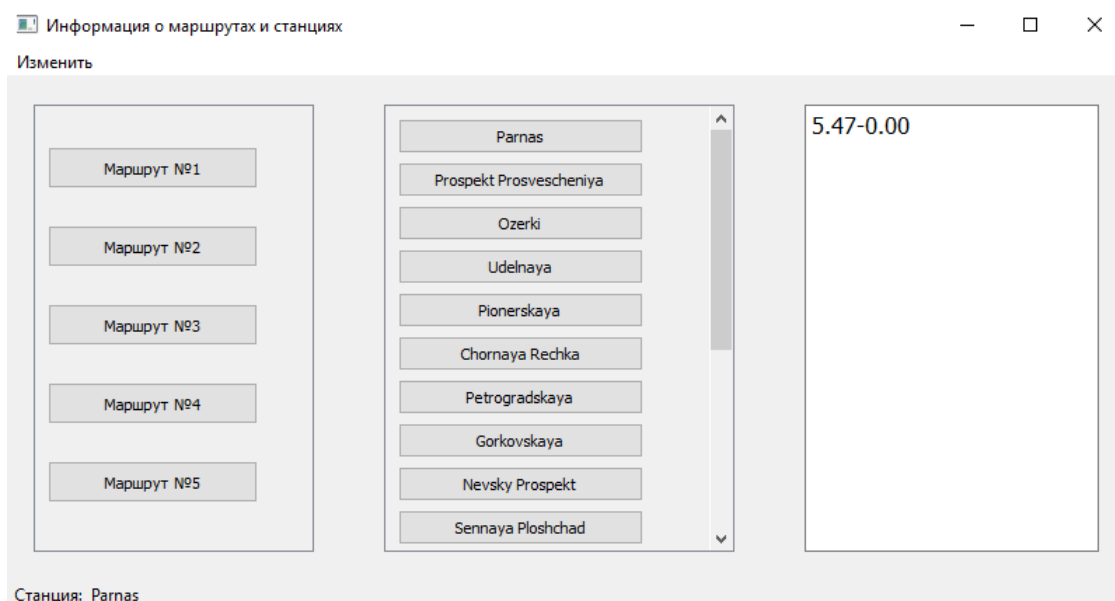


Рис. 6: Главное окно графического приложения

Здесь показано главное окно, в нём можно выбрать маршрут и получить информацию, о станции находящейся в нём

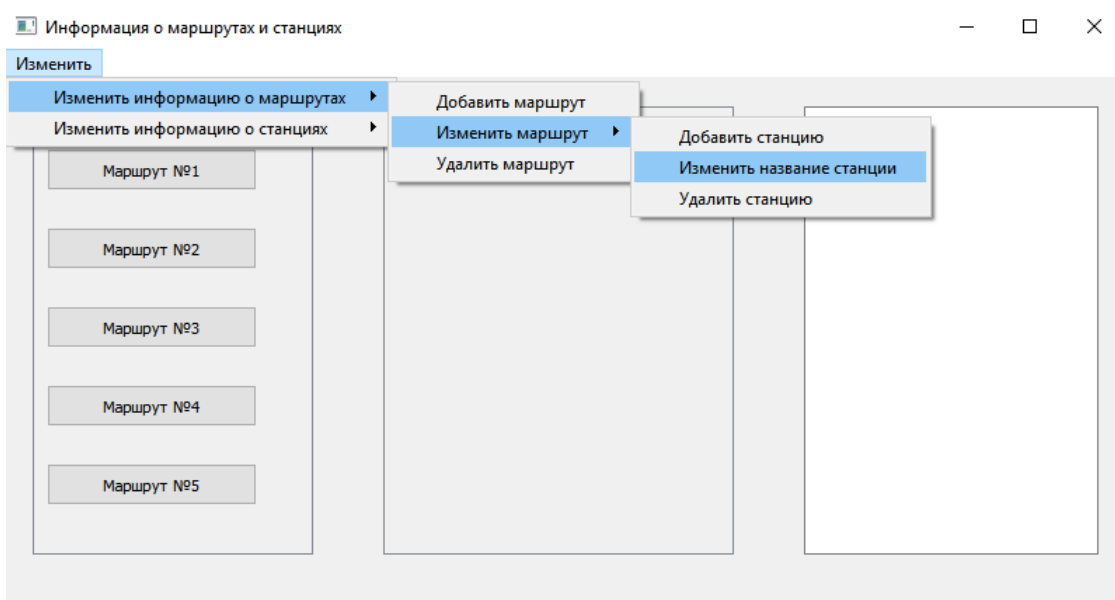


Рис. 7: Открыто меню изменения маршрутов

Здесь продемонстрировано меню редактирования информации о маршрутах и станциях

4 Процесс обеспечения качества и тестирование

Глава – "Процесс обеспечения качества и тестирование". Здесь описываете процесс разработки – количество ревью (и примерное количество замечаний), количество демонстраций (с указанием конкретных

замечаний и комментариями по ним), список использованных утилит (cprcheck, valgrind, gcov и т.д, кто пользовался jenkins – про него тоже) с комментариями как и когда они помогали. Рассказать про автоматические тесты (модульные, функциональные), про тестовые сценарии, которые они покрывают, процент покрытия. Рассказать про тестовые сценарии для ручных тестов.

4.1 Список

- первый элемент списка
- второй элемент списка

4.2 Листинг

Листинг 1: hell_o.c

```
1 #include <stdio.h>
2
3 int main() {
4     puts("Hello ,_Donald_and_Linus");
5     return 0;
6 }
```

Текст без отступа (следует за вставкой)

Новый параграф

Новый параграф с принудительно выключенным отступом

4.3 Частичный листинг

Листинг 2: фрагмент hell_o.c

```
4     puts("Hello ,_Donald_and_Linus");
5     return 0;
```

4.4 Таблица

top left	top right
bot left	bot right

Таблица 1: Название таблицы

5 Выводы

Выводы – своими словами, с душой, от чистого сердца рассказать, чему удалось научиться за семестр, что извлечь. Приложение 1 – лучше бы приложить все листинги Приложение 2 – Примержить pdf, который генерирует доксиджен для кода – как-то это можно сделать (на дженкинсе установлена утилита для такого, без него – есть утилиты и даже онлайн сервис через браузер.