Пояснительная записка

К программной системе «Система Такси»

По предмету Технологии и разработки прикладных программных систем

Проект подготовили студенты группы Р05-281:

Бондаренко Юрий

Галаев Александр

Галиев Булат

Галкин Тимофей

Евтушенко Данила

Разделы пояснительной записки:

1. [Общие положения](http://www.prj-exp.ru/patterns/pattern_tech_project.php#tech_project1)
   * Наименование системы
   * Цели, назначение и области использования системы
   * Нормативные ссылки
   * Очередность создания системы
2. [Основные технические решения](http://www.prj-exp.ru/patterns/pattern_tech_project.php#tech_project2)
   * [Сведения об обеспечении заданных в техническом задании потребительских характеристик системы, определяющих ее качество](http://www.prj-exp.ru/patterns/pattern_tech_project.php#tech_project25)
   * [Средства разработки](http://www.prj-exp.ru/patterns/pattern_tech_project.php#tech_project29)
3. Заключение

Пояснительная записка к техническому проекту на создание программной системы «Такси»

1. Общие положения

1.1. Наименование системы

1.1.1. Полное наименование системы

Полное наименование – Система обеспечения такси.

1.1.2. Краткое наименование системы

Краткое наименование - СОТ, Система.

1.2. Основания для проведения работ

Работа выполняется на основании договора между НИЯУ МИФИ и Студентами, в рамках которого последние обязаны изучить список вверенных дисциплин.

1.3. Наименование организаций – Заказчика и Разработчика

1.3.1. Заказчик

Заказчик:

Сергиевский.М.В. ...

1.3.2. Разработчик

Разработчик:

Студенты Р05-281: Бондаренко, Галкин, Галаев, Галиев, Евтушенко.

1.4 Цели, назначение и область использования системы

Цель: Моделирование части городской инфраструктуры на примере организации работы Такси.

Назначение: Усвоение Студентами программы курса «Технологии и разработки прикладных программных систем», а так же приобретение в ходе работы необходимых практических навыков.

Область использования: система может быть использована в ходе обучающего процесса.

1.5. Нормативные ссылки

При техническом проектировании использовались следующие нормативно-технические документы:

1. Гост 19.404-79

1.6 Очередность создания системы

**Итерация 1**

**Анализ требований**

Была сформулирована расширенная постановка задачи:

Имеется M машин такси заранее расположенных по городу и L пассажиров.

Число мест в каждом такси 4. Пассажиры распологаются в заранее определённых местах посадки до тех пор, пока не исчерпается их число (программа работает до тех пор, пока система такси не развезёт всех пассажиров).

Такси может подсаживать пассажиров, если это не приводит к изменению маршрута. Такси, высадившее пассажиров, остаётся на пункте, пока не посупит новый заказ ( поступает в случае, если это такси окажется ближайшим к какому-либо пассажиру ).

Задача моделируется, операясь на тот факт, что в «городе» отсутствуют пробки и светофоры, задерживающие такси. ( Такси всё время движется с постоянной скоростью ).

Было решено, что диаграмма прецедентов должна содержать следующие функции:

Для пассажира такси:

-Вызвать ближайшее такси

-Установить конечную точку

Для такси:

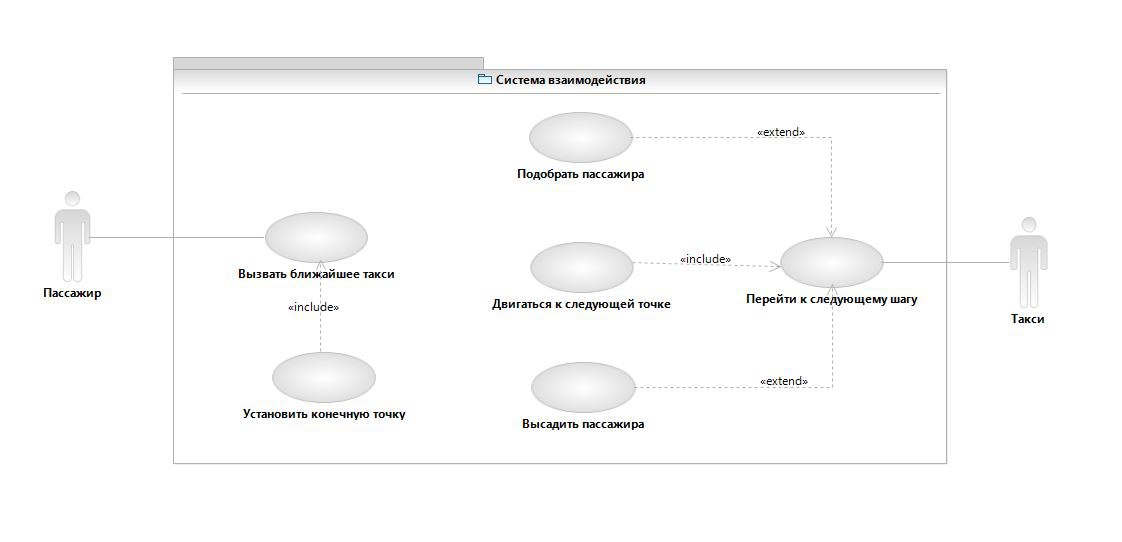
-Перейти к следующему шагу

-Подобрать пассажира

-Двигаться к следующей точке

-Высадить пассажира

На основе уточнённой задачи была построена диаграмма прецедентов.



**Проектирование**

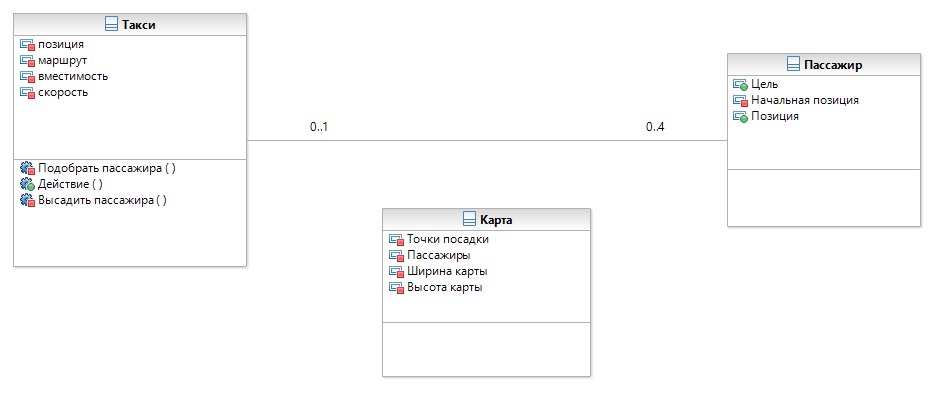
Было решено, что диаграмма классов должна содержать 3 класса, а именно:

-Такси, в котором содержится информация о позиции, маршруте, вместимости и скорости такси

-Пассажир, в котором содержится информация о его позиции, пункте назначения и текущее местоположение

-Карта, состоящая из точек посадки (пункты и связи между ними) и включающая в себя список пассажиров

Для выполнения поставленной задачи была предложена следующая диаграмма классов



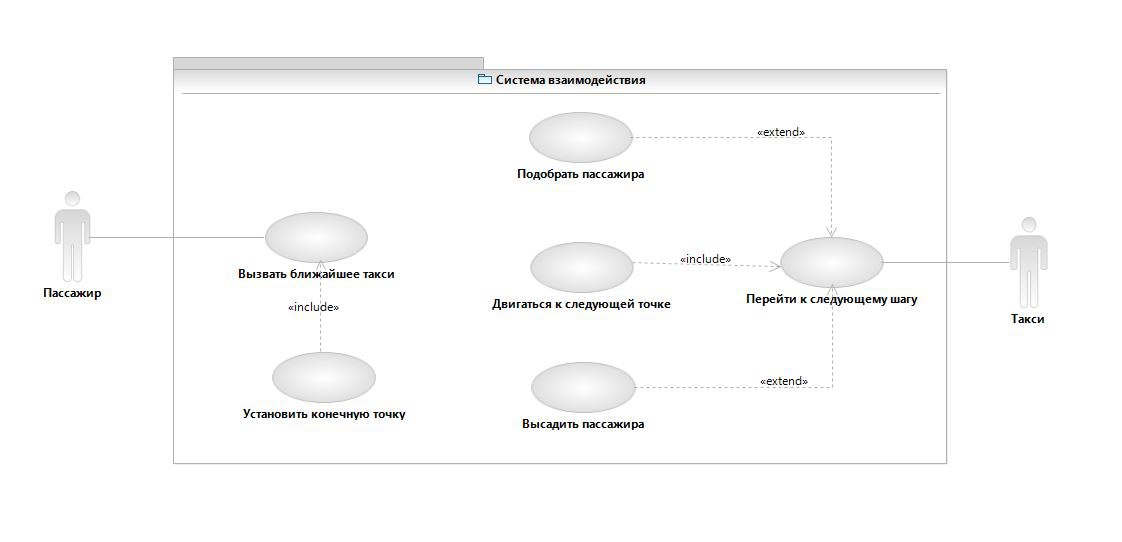
**Программирование**

По полученной диаграмме классов была построена первичная структура кода, соответствующая текущему уровню абстракции. Эта структура не в полной мере отражает функционал, ожидаемый заказчиком от программной системы.

**Итерация 2**

**Анализ требований**

При использовании шаблона проектирования Controller, была введена новая сущность, ответственная за хранение данных и взаимодействие между классами. Поэтому исходная диаграмма прецедентов не претерпела каких-либо изменений.



**Проектирование**

После решения о вводе дополнительного класса «Контроллер», диаграмма классов, представленная в первой итерации была дополнена.

Класс «Контроллер» был создан для реализации хранения данных и взаимодействия между тремя классами, представленными в первой итерации.

Было решено, что класс Контроллер будет содержать следующую информацию:

-Список пассажиров

-Список такси

И осуществлять следующие методы:

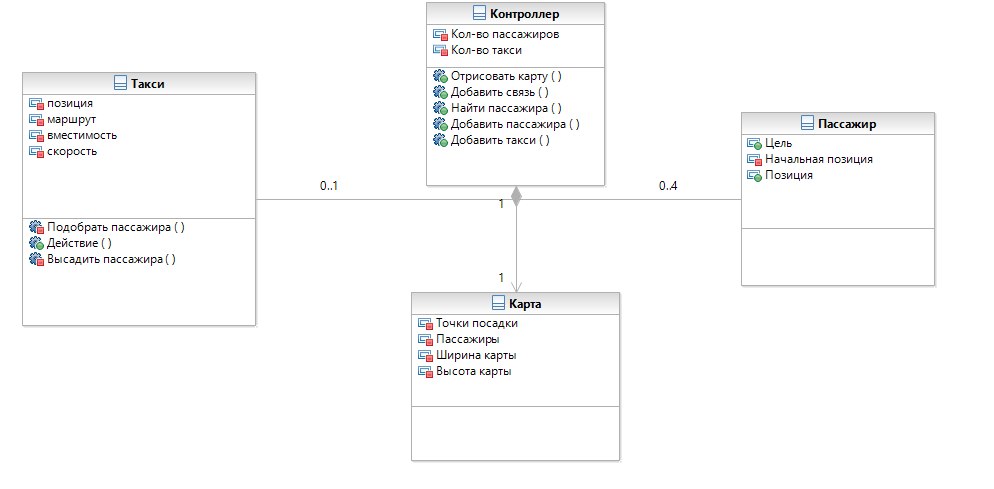
-Отрисовывать карту

-Добавить связь к точке

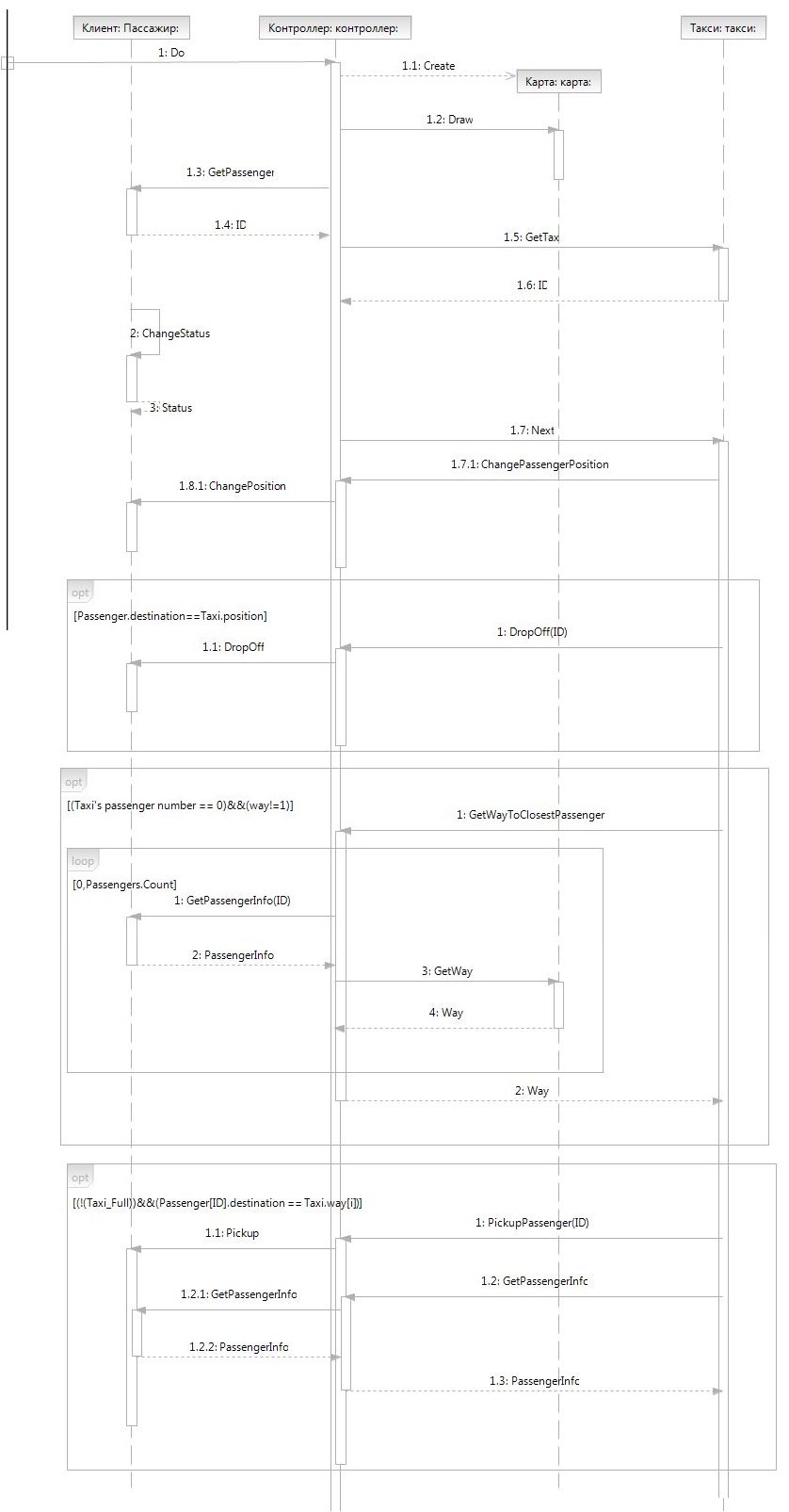
-Найти пассажира

-Добавить пассажира в список

-Добавить такси в список

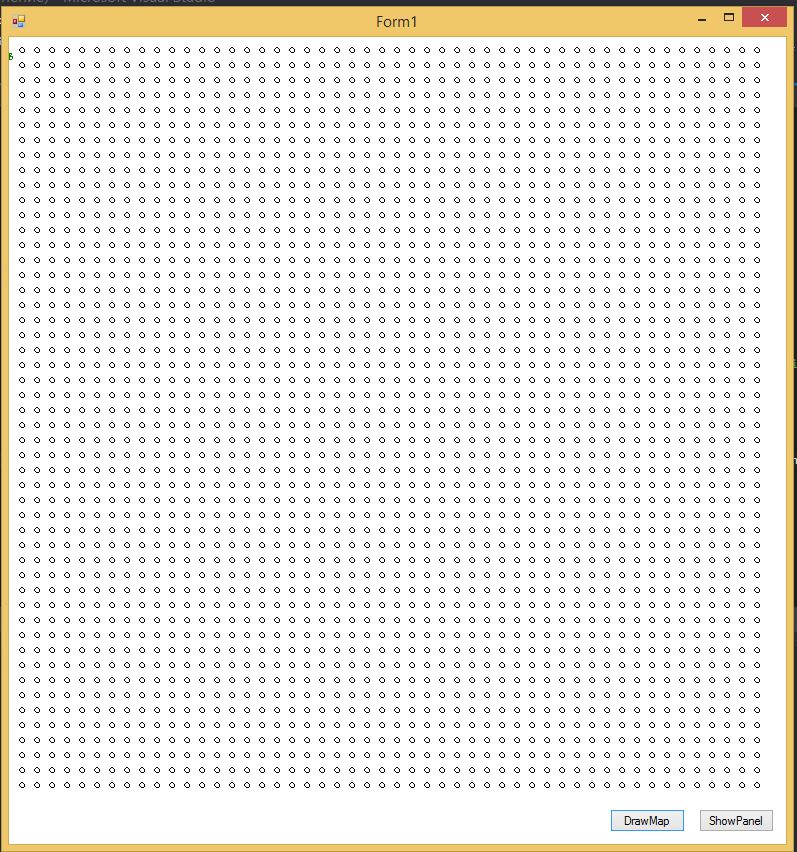


Чтобы более детально описать упорядоченное по времени взаимодействие объектов с последовательностью их появления, была построена диаграмма последовательностей для прототипа, планируемого на данном итерационном шаге разработки программной системы.



**Программирование**

Была визуализированна модель уличной системы. Каждый круг на форме символизирует собой останвочный пункт; зелёный прямоугольник в левом верхнем углу – одна из машин такси, осуществляющих функции транспортировки пассажиров.



**Итерация 3**

**Анализ требований**

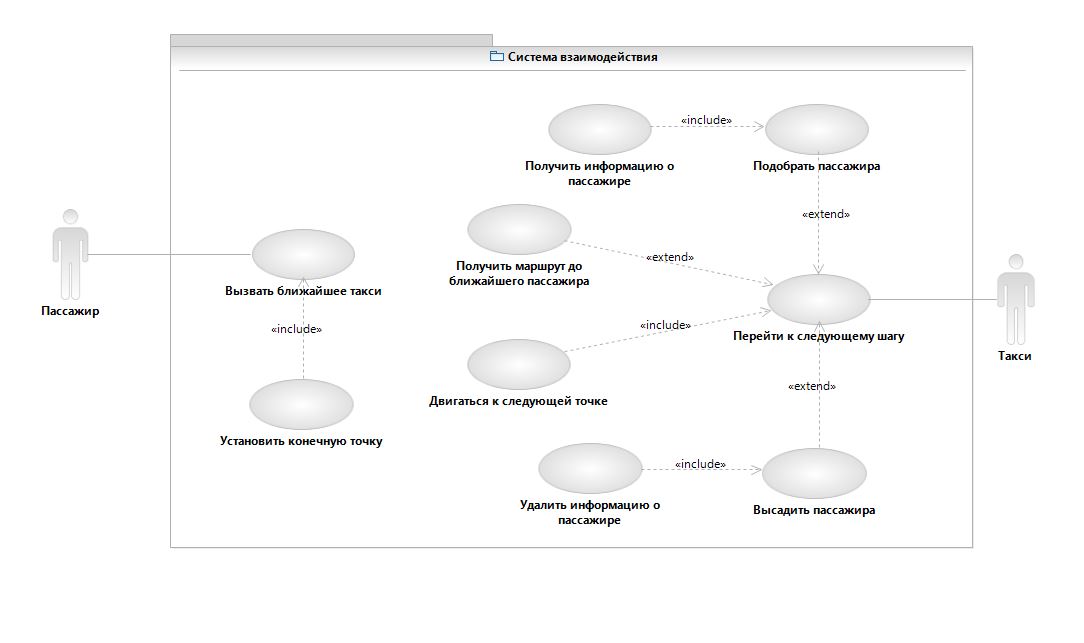
Было решено дополнительно ввести следующий набор функций, оптимизирующих работу программы:

-Удалить информацию о пассажире

-Получить информацию о пассажире

-Получить маршрут до ближайшего пассажира

Новая диаграмма прецедентов имеет следующий вид

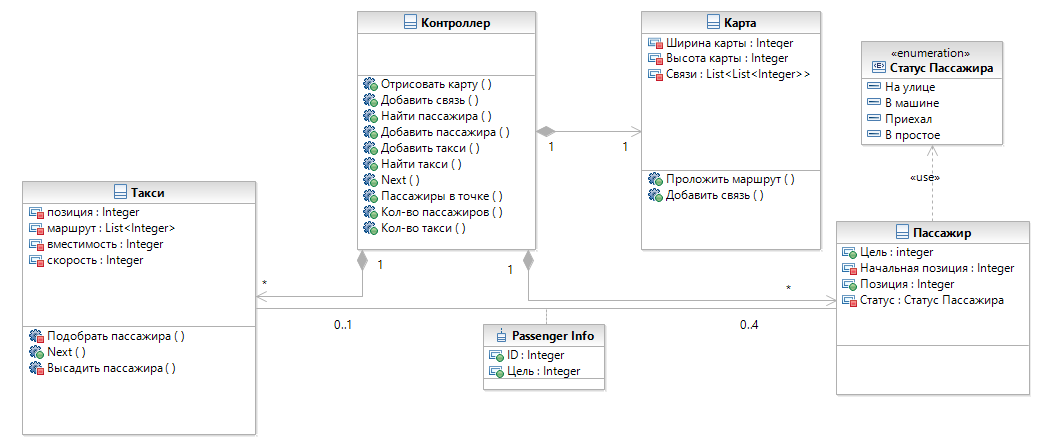
****

**Проектирование**

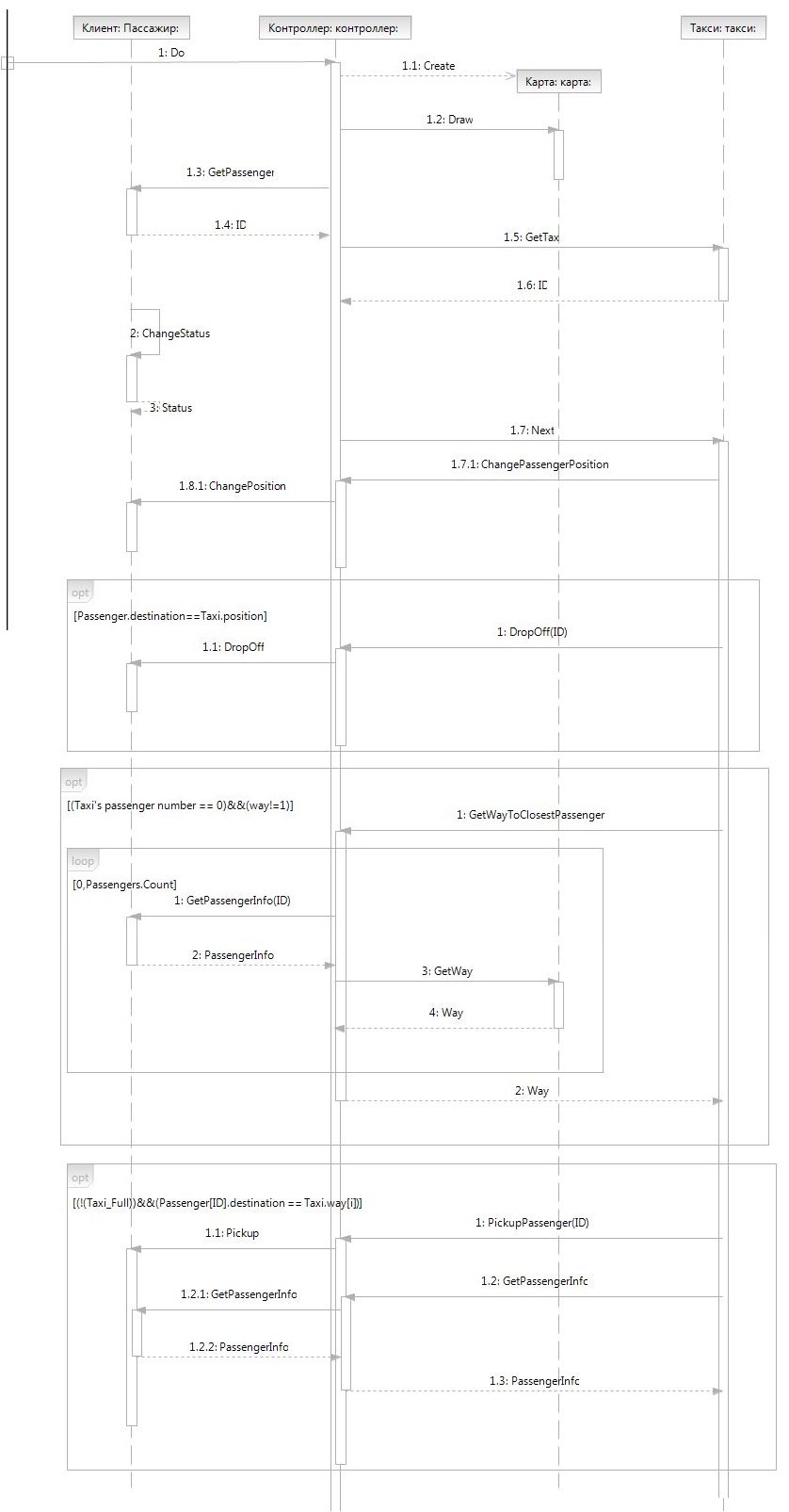
Введение класса ассоциации и перечисления для улучшения качества кода с точки зрения читаемости.

Класс ассоциация Passenger Info содержит следующие поля: ID, Цель.

Перечисление будет отвечать за текущий статус пассажира и уточнять следующую информацию: Находится ли пассажир на улице (или в машине), добрался ли пассажир до своей цели и находится ли пассажир в ожидании вызванной машины в настоящий момент времени.



При дорабатывании кода не изменилась основная структура взаимодействий, поэтому диаграмма последовательностей не изменила своего первначального вида.



**Программирование**

Реализация класса ассоциации и перечисления для улучшения качества кода с точки зрения читаемости.

**Тестирование и отладка**

При тестировании была обнаружена проблема в функции нахождения ближайшего к такси пассажира, использующая, реализованную рекурсивно функцию нахождения маршрута (Многие машины такси ехали за одним и тем же пассажиром). Для исправления ошибки было модифицированно перечисление используемое в классе пассажир.

2. Основные технические решения

2.1. Сведения об обеспечении заданных в техническом задании потребительских характеристик системы, определяющих ее качество

|  |  |
| --- | --- |
| **Требование** | **Подход** |
| Функциональность | Углубленный анализ требований заказчика, взаимодействие с ним на каждом этапе разработки программной системы. |
| Эффективность | Оптимизация объёмов используемых ресурсов путём введенеия и последующего использования динамических структур данных. |
| Практичность | Лёгкость эксплуатации программного обеспечения достигается благодаря узкой группировке и широкой взаимосвязанности реализуемых функций. |
| ... | ... |

2.2. Средства разработки

Для создания ПС использовалось программное обеспечение, включающее IBM Rational Software Architect, интегрированную среду разработки Microsoft Visual Studio. Для работы в Visual Studio использовался язык программирования C# .

1. Заключение

Перед каждым из членов команды задача групповой работы впервые была поставлена при работе именно над этим проектом. Тем не менее, практически сразу после того, как мы узнали, что будем работать над общим проектом, решили, что наиболее подходящий язык для реализации проекта - C#, средство разработки – Microsoft Visual Studio. Именно это средство разработки подходило нам по характеру поставленной задачи. Так же немаловажен тот факт, что именно с этим средством разработки и языком программирования всем участникам проекта приходилось сталкиваться наибольшее число раз. Несмотря на то, что формат разработки подразумевает абсолютно чёткое распределение ролей, каждый участник, которому требовалась какая-либо помощь, получал её от остальных, что очень способствовало грамотной работе. Иногда возникали разногласия вследствие разного видения ситуации и задачи разными участниками проекта. Чаще всего подобные разногласия решались достижением какого-либо компромисса, в остальных случаях помогало взаимодействие с преподавателем, расставлявшее по местам и прояснявшее многие моменты по ходу разработки. Проект был реализован при помощи технологии итеративной разработки, что заметно упрощало работу над ним в целом, а использование такого сервиса контроля версий, как GitHub, позволяло осуществлять работу над проектом даже в те моменты, когда общий сбор и полноценное взаимодействие были невозможны по тем или иным причинам. За время нашей совместной деятельности было много различных хороших и плохих моментов, мы вздорили, мирились, спорили, но это очень сильно помогло научиться понимать друг друга, идти на компромиссы и работать в нужном русле. Каждый получил огромное количество необходимого в нашей дальнейшей профессиональной жизни опыта. В конечном счёте, все получили набор положительных впечатлений от работы в команде, а все тягости и невзгоды, которые легли на наши плечи, померкли перед тем фактом того, что мы смогли многому научиться за время нашей работы.