Разработка АИС

Автоматизированная информационная система (АИС) — совокупность программно- аппаратных средств, предназначенных для автоматизации деятельности, связанной с хранением, передачей и обработкой информации.

АИС являются, с одной стороны, разновидностью [информационных систем](https://dic.academic.ru/dic.nsf/ruwiki/101791) (ИС), с другой [автоматизированных систем](https://dic.academic.ru/dic.nsf/ruwiki/16398) (АС), вследствие чего их часто называют ИС или АС.

АИС может быть определена как комплекс автоматизированных информационных технологий, предназначенных для информационного обслуживания организованного непрерывного технологического процесса подготовки и выдачи потребителям научной, управленческой и др. информации, используемой для принятия решений, в соответствии с нуждами для поддержания  эффективной деятельности.

Назначение АИС

Основной причиной создания и развития АИС является необходимость ведения учёта информации о состоянии и динамике объекта, которому посвящена система. На основании информационной картины, создаваемой системой, руководители различного звена могут принимать решения об управляющих воздействиях с целью решения текущих проблем.

Для хорошего и более удобного функционирования почти любой системы, не обязательно информационной, необходимы ведение контроля, анализа работы, внедрение автоматизации для отдельных процессов или системы в целом. Автоматизация позволяет повысить производительность и качество системы, ускорить работу, оптимизировать процессы управления, снизить затраты.

Целью данной учебной практики является проектирование информационной системы аэропорта, которая позволила бы значительно улучшить качество обслуживания клиентов и упростить процессы оформления продаж билетов, вести учет прилета и вылета самолетов, отслеживание багажа, расписание рейсов, управление персоналом и учет финансов.

Для описания предметной области и проектируемой системы будем использовать диаграммы UML. Язык моделирования UML (англ. Unified Modeling Language — унифицированный язык моделирования) — язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур

Осуществление работы аэропорта происходит таким образом:

Диспетчер сообщает общую информацию о передвижении воздушных суден, сотрудник аэропорта, взаимодействуя с пассажиром, вносит его данные и данные его багажа (если он имеется) в базу, далее пассажир переходит к инспектору по досмотру, который проводит досмотр и допускает пассажира до полёта. С графическими данными мы может ознакомиться с рисунком 1.

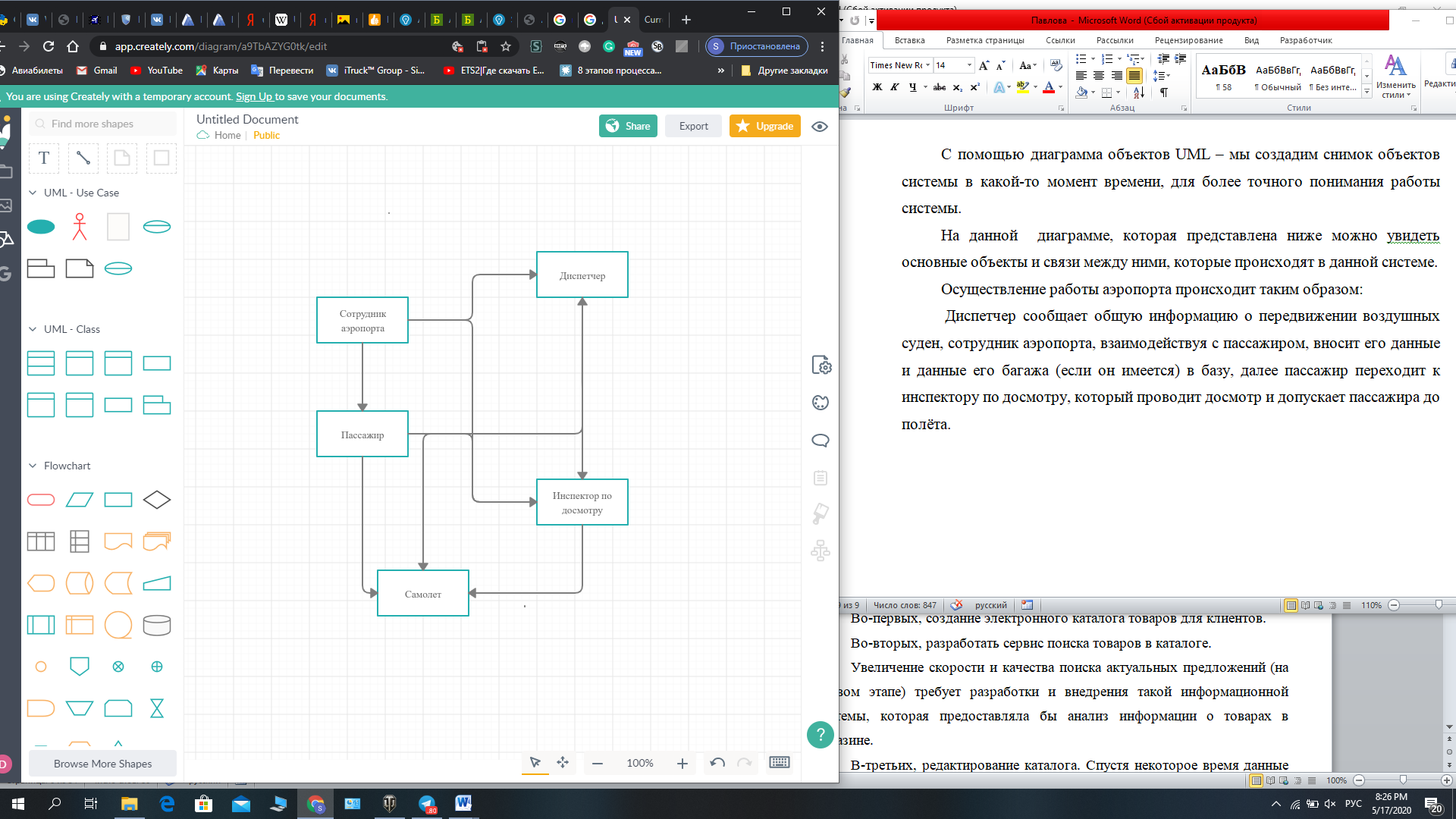


Рисунок 2 – Диаграмма объектов

Разработка требований к ИС

Наименование программного продукта – информационный ресурс для регистрации на рейс онлайн. Данный программный продукт придуман для того, чтобы уменьшить время прохождения пассажирами регистрации на рейс, облегчить и уменьшить работу сотрудникам аэропорта.

Пользователями данного ресурса будут пассажиры, который могут пройти регистрацию на рейс без дополнительно запрашиваемых документов, таких как: виза, доверенность и другие документы, без которых осуществление полёта для данного клиента будет невозможно. Далее, для людей, у которых имеется багаж, также предстоит воспользоваться стандартными методами регистрации.

Требования к данному программному продукту:

Для работы данного приложения необходим интернет, который должен быть бесплатным и работать по всей территории аэропорта.

Вся информация должна быть актуальной на текущий момент времени, каждую минуты должно идти обновление приложения о регистрации на новые рейсы и прочей информации.

Каждые внесенные данные сразу должны сохраняться и поступать в единую базу данных аэропорта.

После завершения всех требований по внесению данных, пассажир может проследовать к терминалу вылета.

Данная предметная область рассматривается только с точки зрения клиентов (пассажиров). Сотрудники аэропорта могут сверять данные, вносимые пассажирами через общую базу, куда приходит вся информация, если в том есть необходимость.

Основой разработки требований является модель системных прецедентов, отражающая выполнение конкретных обязанностей внутренними и внешними исполнителями с использованием информационной системы. Основное назначение данной диаграммы – описание функциональности и поведения системы.

Прецедент – возможность моделируемой системы (часть её функциональности), благодаря которой пользователь может получить конкретный, измеримый и нужный ему результат. Прецедент определяет один из вариантов использования системы и описывает типичный способ взаимодействия пользователя с системой.

Для работы системы информационного ресурса аэропорт на диаграмме вариантов использования были выделены следующие actor:

Клиент/пассажир – это обычные люди, которые нуждаются в перелёте из одной части мира в другую. С помощью приложения могут зарегистрироваться на рейс без взаимодействия с сотрудником аэропорта, если на них не распространяются некоторые ограничения;

Сотрудник аэропорта– это человек, имеющий неограниченный доступ к системе, а также занимающейся внесением информации в базы данных о пассажире, если по каким-либо причинам человек сам не сможет внести свои данные, оказывающий помощь клиенту и контактирующий с диспетчерами и инспекторами по досмотру аэропорта;

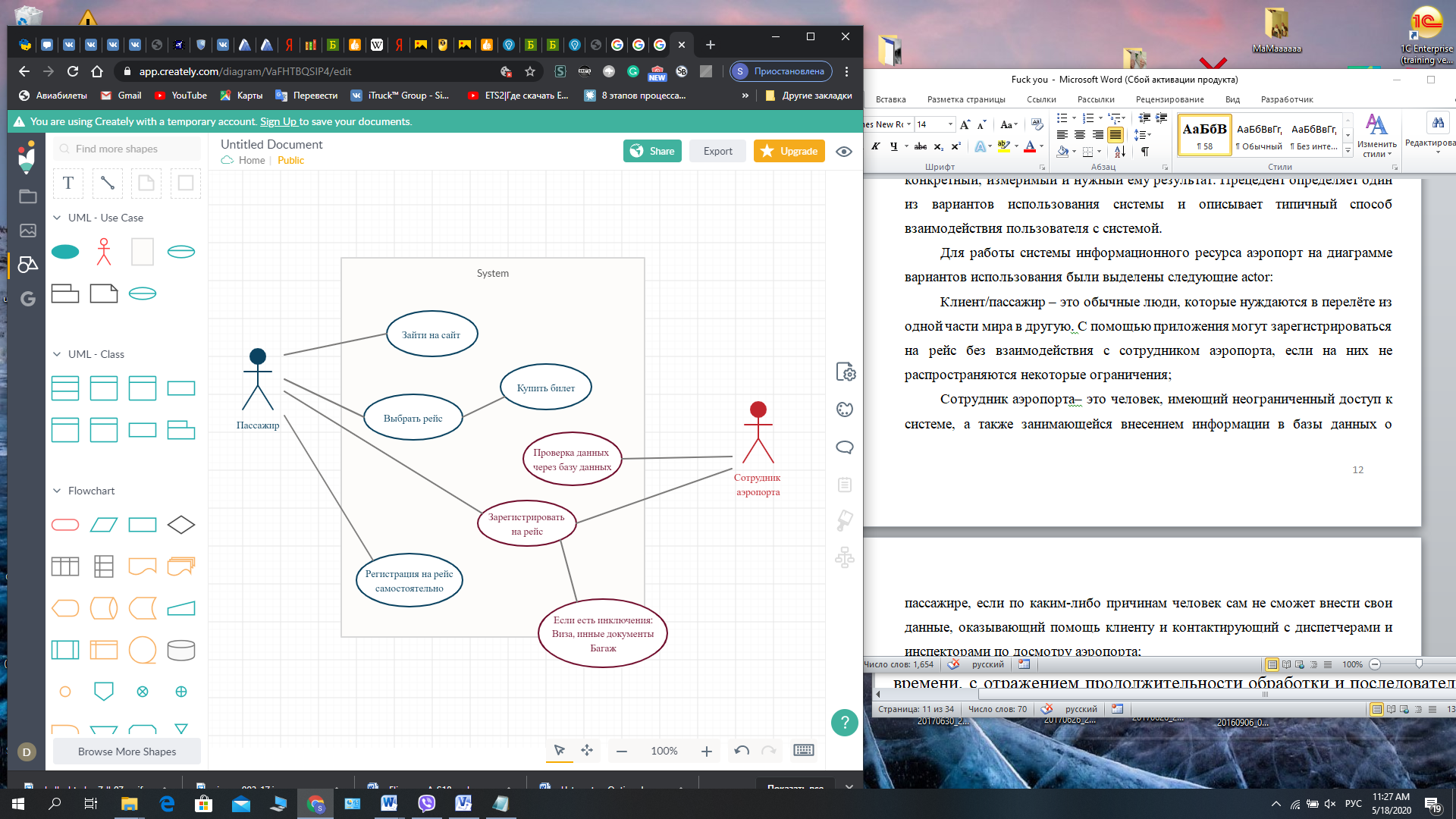


Рисунок 2 представляет модель основных прецедентов системы для покупки билетов и онлайн регистрации на рейс.

Следующая диаграмма последовательностей данной предметной области изображена на рисунке 3. На данной диаграмме показано взаимодействие объектов (обмен между ними сигналами и сообщениями), упорядоченное по времени, с отражением продолжительности обработки и последовательности их проявления.

Основными элементами диаграммы последовательности являются обозначения объектов (прямоугольники с названиями объектов), вертикальные "линии жизни" ("Пассажир", "Сотрудник аэропорта", "Инспектор по досмотру", "Транспортировка багажа"), отображающие течение времени, прямоугольники, отражающие деятельность объекта или исполнение им определенной функции (прямоугольники на пунктирной "линии жизни"), и стрелки, показывающие обмен сигналами или сообщениями между объектами.

Входом в диаграмму последовательностей служит выбор и покупка билета, далее Пассажир, взаимодействуя с системой и Сотрудником аэропорта для дальнейшего прохождения регистрации на рейс. Далее Сотрудник проверяет билет и если есть багаж, то отправляет его на погрузку. После данной процедуры идет выбор сиденья на воздушном судне и выдача посадочного билета.

Потом наступает очередь инспектора, который проверяет билеты и проводит осмотр пассажиров. После удачного прохода всех проверок пассажир идет на посадку в самолет, как это показано на рисунке 3.

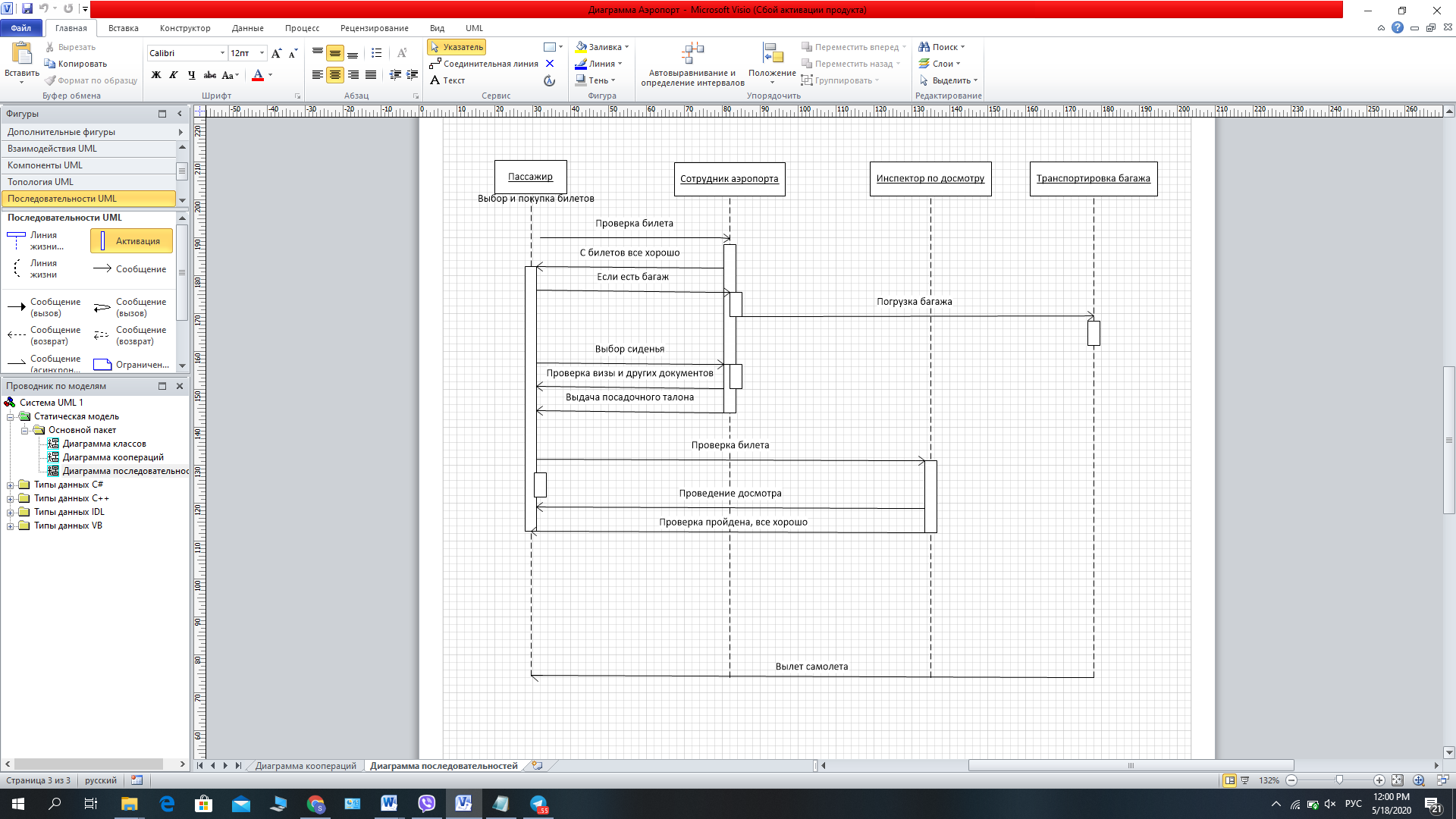


Рисунок 3 – Диаграмма последовательностей для проверки билетов и проведения досмотра перед полетом.

Но с нашей идеей для облегчения обслуживания работы сотрудников данная диаграмма будет выглядеть иначе, в ней не будет взаимодействия между пассажиром и сотрудником аэропорта.

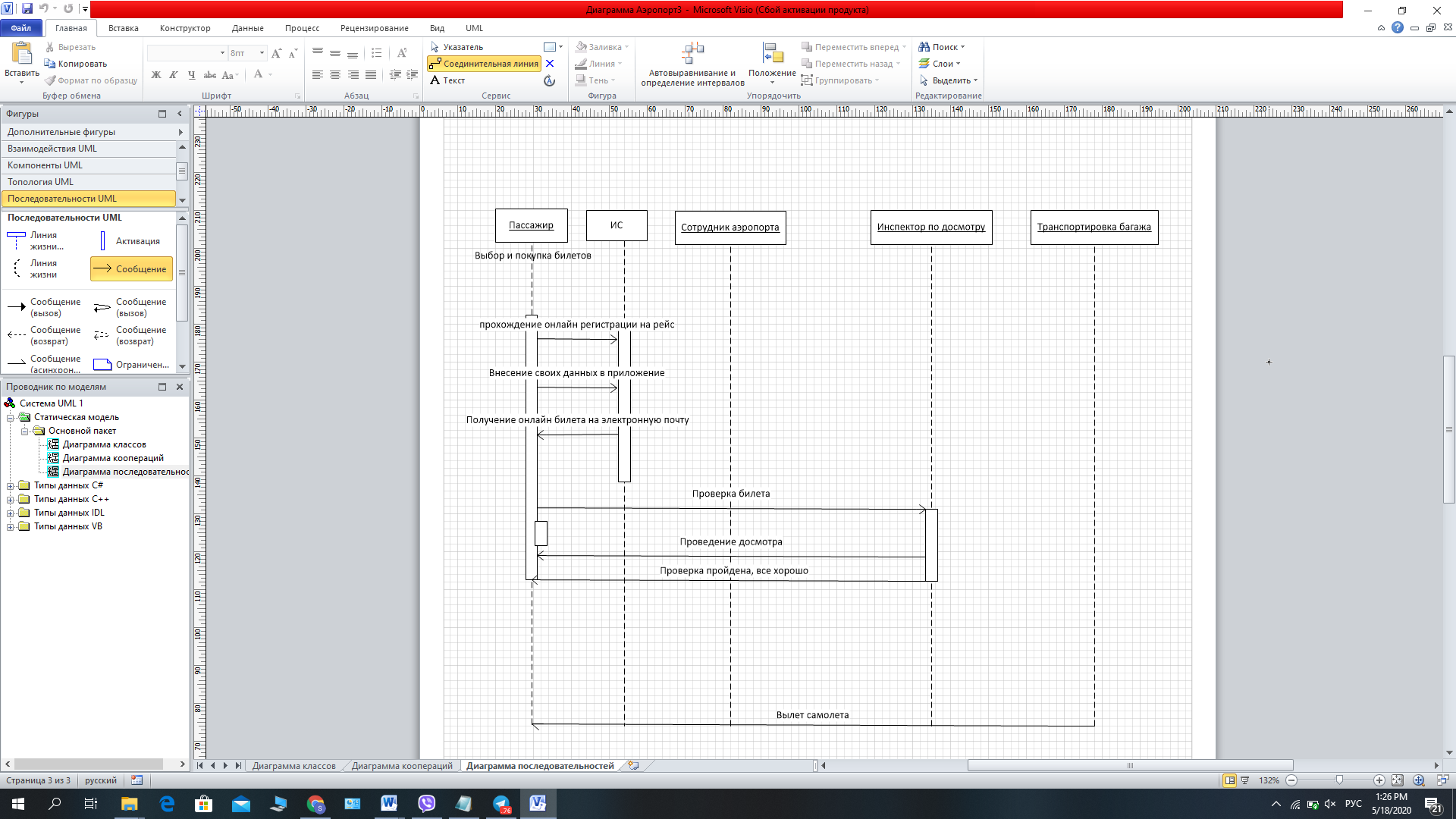


Рисунок 4 – Диаграмма последовательностей для онлайн регистрации пассажира и проведения осмотра перед полётом.

Как мы можем увидеть из диаграмм действие сотрудников аэропорта в регистрации пассажиров не требуется и, соответственно, экономятся средства и время. Так же регистрация проходит без очередей, значит почти не будет занимать много времени у пассажиров.

Распишем спецификацию для каждого отдельного варианта использования и сделаем диаграмму последовательностей.

Вход в систему.

Наименование: Вход в систему.

Actor, взаимодействующие с прецедентом: Клиент/Пассажир .

Описание: Вход в систему служит для того, чтобы клиент мог в своем личном кабинете осуществлять регистрацию на предстоящий рейс по прибытии в аэропорт.

Предположение: Вход в систему через ввод данных билета обязателен для регистрации на рейс.

Постусловия: если клиент не может войти на сайт, то ему следует зарегистрироваться через стойки регистрации, если же вход выполнен, то пассажир вводит свои данные для дальнейшего прохождения регистрации.

Приоритетный маршрут: зайдя на сайт, пользователь вводить свои данные билета.

Исключительный маршрут: Пользователь не может быть зарегистрирован на рейс без ввода данных билета и данных самого пассажира.

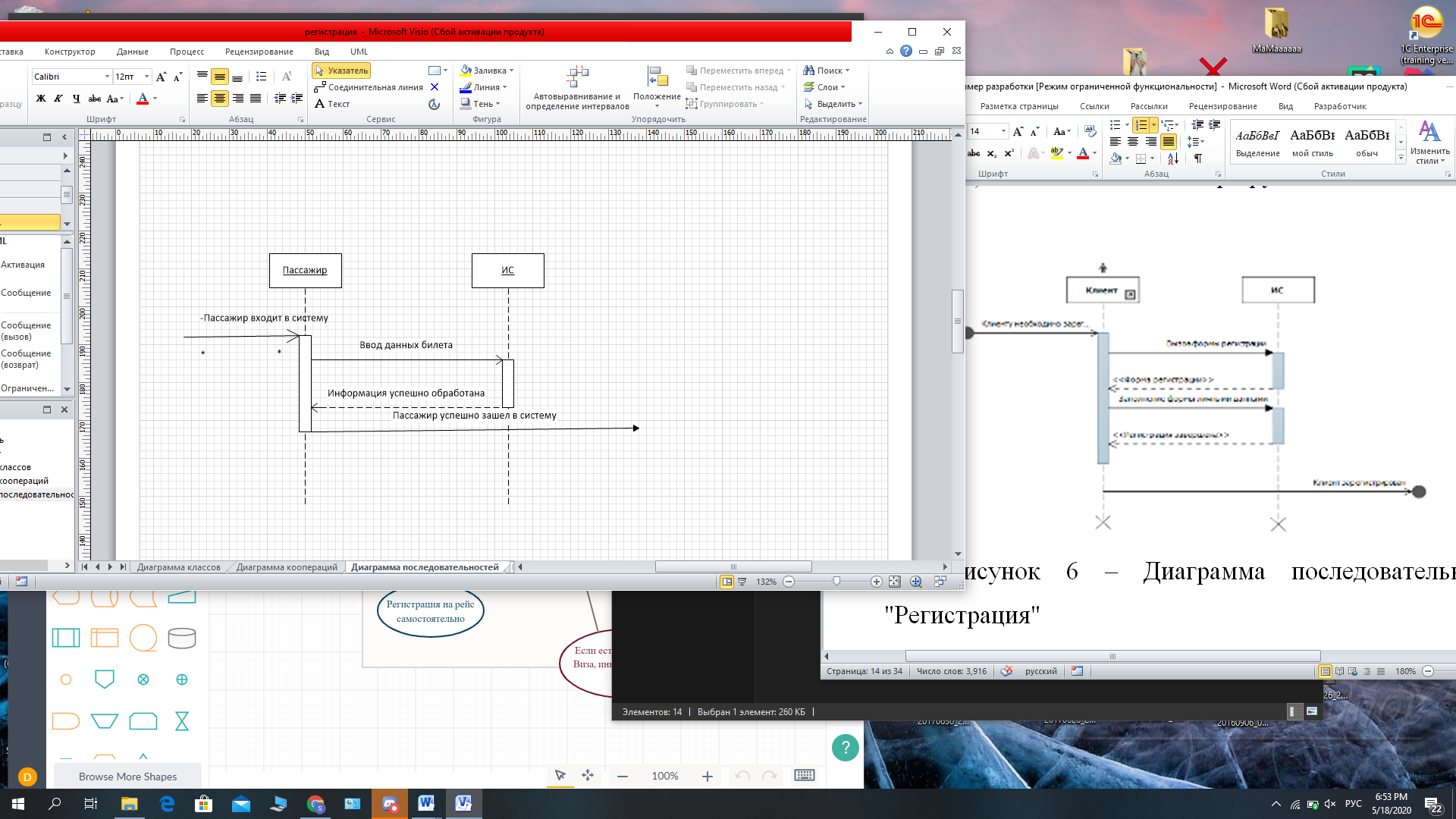


Рисунок 5 – Диаграмма последовательностей для прецедента "Вход в систему"

Регистрация.

Наименование: Регистрация.

Аctor, взаимодействующие с прецедентом: Клиент/Пассажир.

Описание: Регистрация на рейс проходит в процессе внесения данных о пассажире.

Предположение: Регистрация обязательна, для дальнейшего прохождения контроля перед полётом.

Постусловия: После успешной регистрации, пассажир проходит на досмотр, а позже на посадку в самолёт.

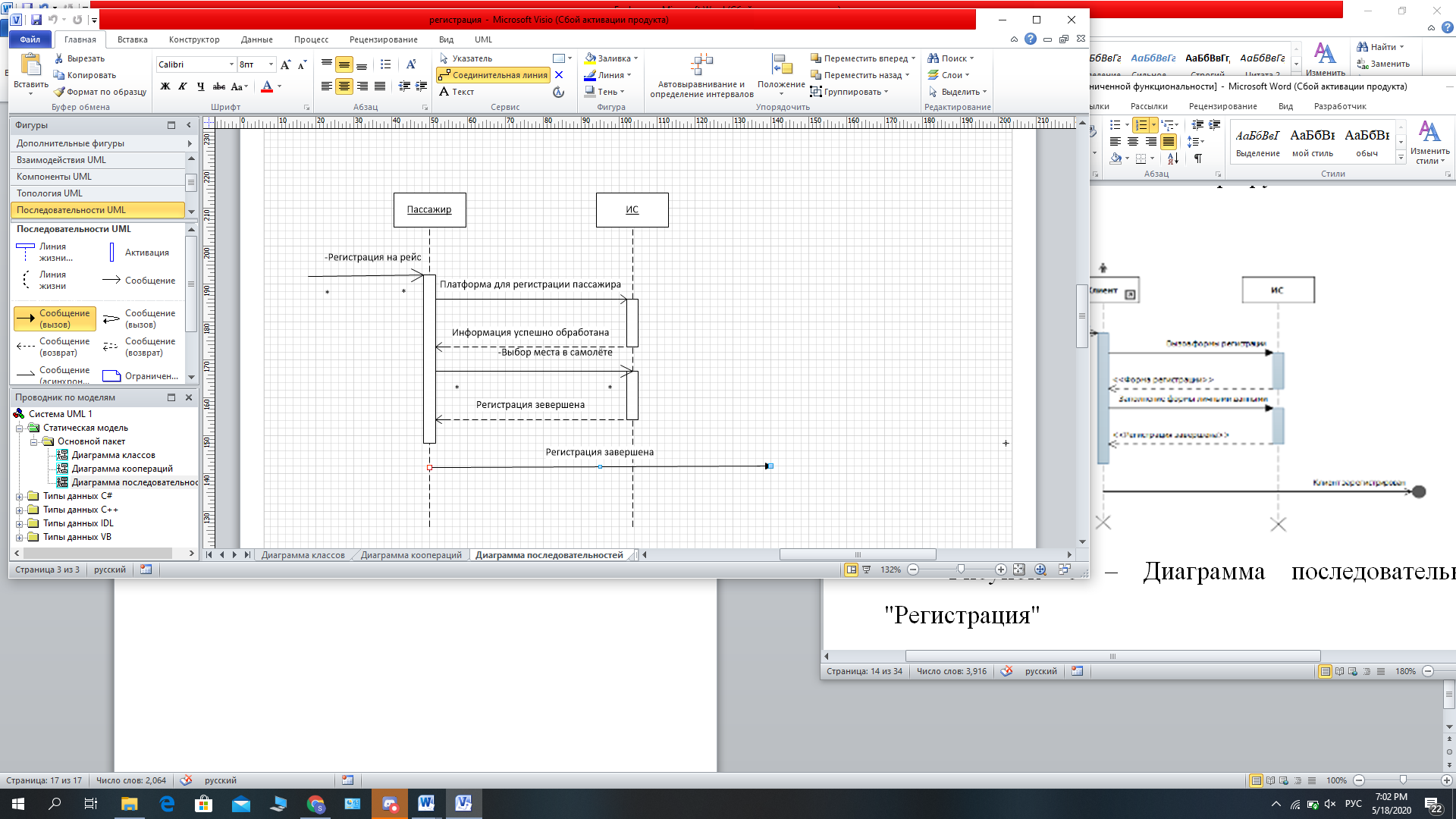


Рисунок 6 – Диаграмма последовательностей для прецедента "Регистрация"

Диаграмма классов описывает типы объектов системы и различного рода статические отношения, которые существуют между ними. На диаграммах классов отображаются также свойства классов, операции классов и ограничения, которые накладываются на связи между объектами.

Данная диаграмма классов системы построена на основе разработанной модели системных прецедентов.

Таблица "Пассажир" содержит такую информацию как ФИО пассажира, данные о билете, электронная почта, контактный телефон, личные данный пассажира.

Таблица "Сотрудник аэропорта" содержит информацию о ФИО сотрудника, авиакомпания, которую представляет сотрудник и терминал, к которому прикреплён сотрудник.

Таблица "Диспетчер" содержит основную информацию о ФИО диспетчера и его должность.

Таблица "Инспектор по досмотру" содержит информацию о ФИО инспектора и его должности.

Таблица "Самолёт" содержит основную информацию о наименовании самолёта, авиакомпании, которой принадлежит самолёт и тип перевозок, совершаемый им.

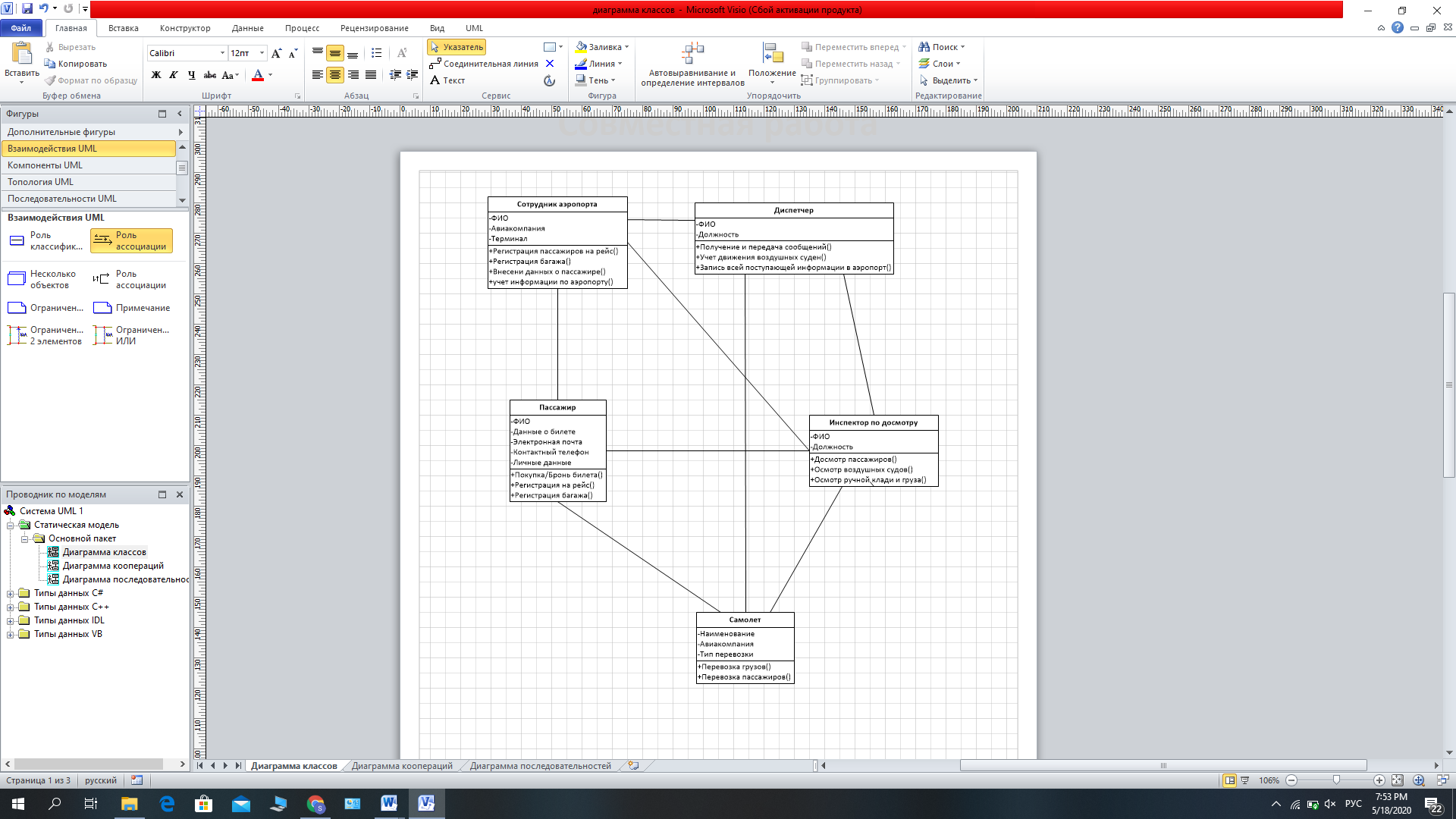


Рисунок 7 – Диаграмма классов системы

Диаграмма компонентов, в отличие от ранее рассмотренных диаграмм, описывает особенности физического представления системы. Данная диаграмма компонентов поможет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код.

Диаграмма компонентов разрабатывается для следующих целей:

- визуализации общей структуры исходного кода программной системы;

- обеспечения многократного использования отдельных фрагментов программного кода;

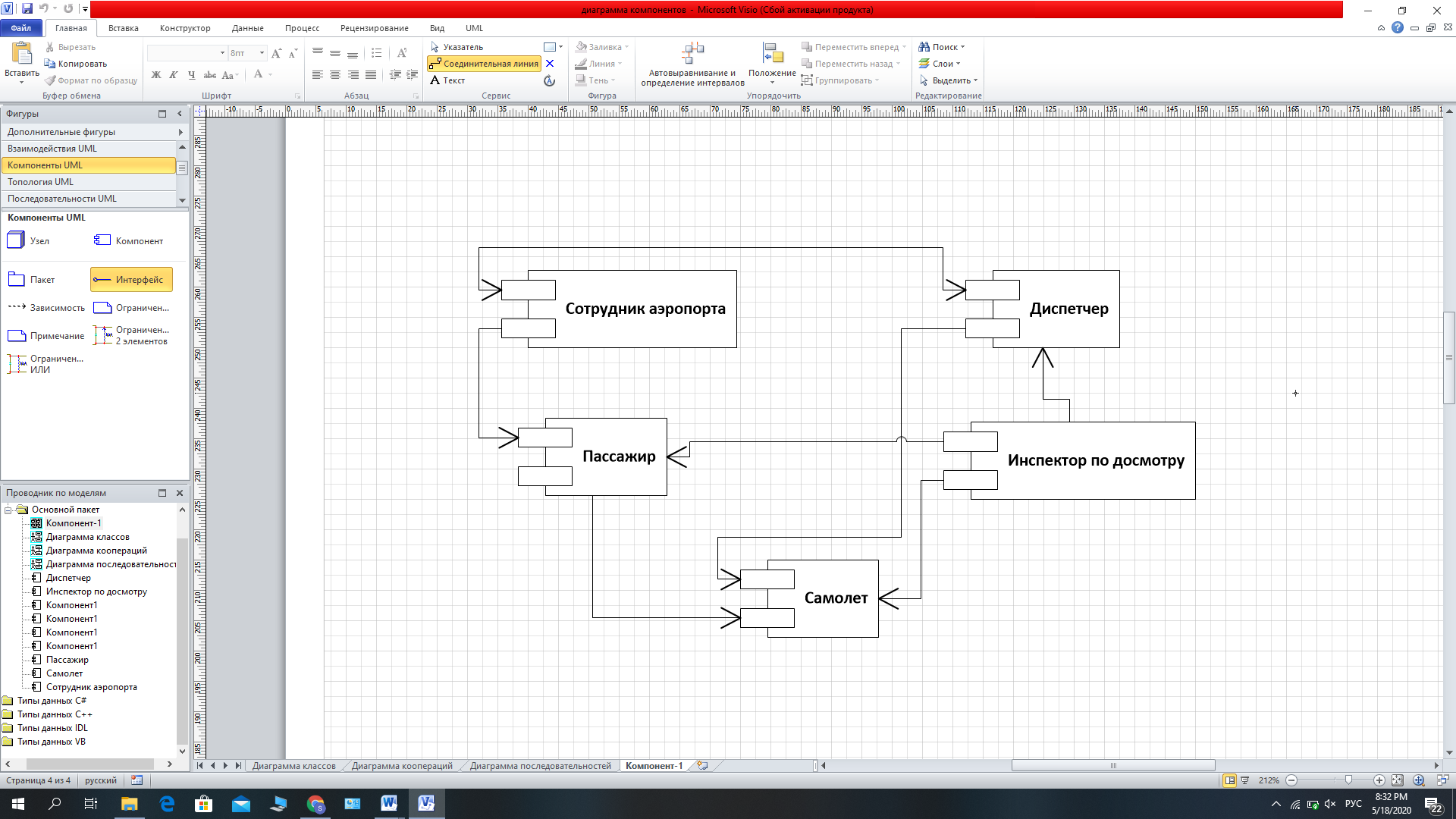


Рисунок 8 – Диаграмма компонентов

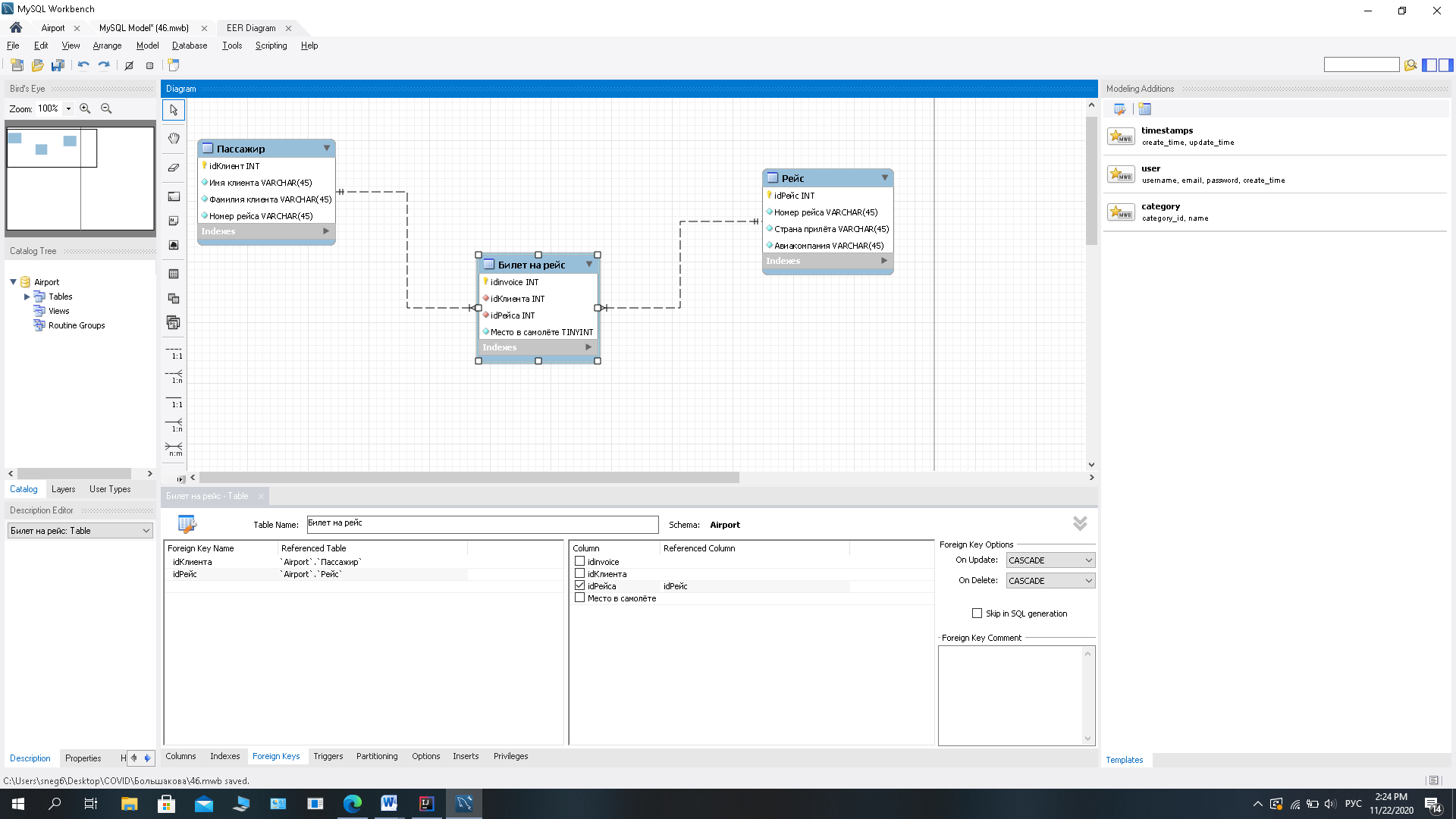


Рисунок 9 – Запрос о рейсе