Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ

УНИВЕРСИТЕТ»

«Проектирование программного обеспечения»

Отчёт по лабораторной работе №4

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ МОДЕЛИРОВАНИЕ.

UML – ДИАГРАММЫ ПОВЕДЕНИЯ

Цель: Изучение методологии объектно-ориентированного моделирования средствами UML. Ознакомление с основными принципами объектно-ориентированного проектирования программного обеспечения, получение навыков проектирования функциональности информационной системы с применением UML.

Выполнил: Драч К.А.

ФИТ 3 курс 1 группа

Преподаватель: Панченко О.Л.

1. Описание программных средств

Как и в прошлой лабораторной работе, для построения диаграмм будем использовать Draw.io –бесплатное онлайн-приложение для создания диаграмм и схем.

Название: Draw.io

Версия: Веб-приложение

Разработчик: JGraph Ltd.

Адрес: https://www.draw.io/

Режим использования: Онлайн

Доступность на платформах: любой веб-браузер.

Draw.io позволяет создавать различные типы диаграмм, такие как блок-схемы, организационные диаграммы, UML-диаграммы, сетевые диаграммы и многое другое.

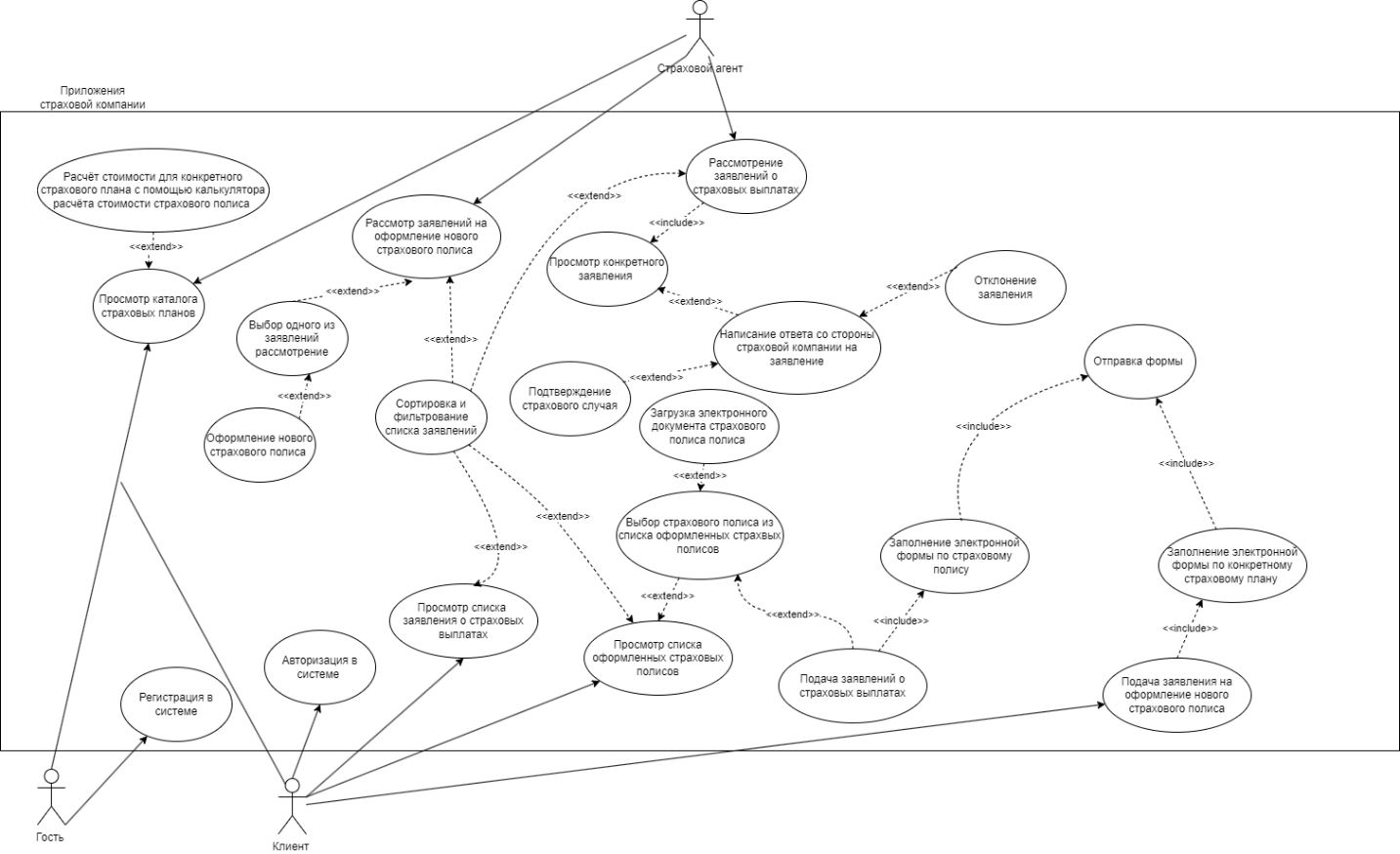
1. Описание практического задания

В соответствии с требованиями к информационной системе разработаем Use-Case-диаграмму описывающая взаимодействие прецедентов с внешними пользователями. Actor — это типовые пользователи (менеджер, операторы и т. д.). Актер представляет собой некоторую роль, которую играет пользователь (или другая система) по отношению к системе.

В подсистеме выделяются такие актёры как гость, клиент, страховой агент. Иерархия актёров выстроена следующим образом: страховой агент находиться на вершине иерархии, за ним следует клиент страховой компании, а за ним следует гость. Каждый уровень иерархии имеет свои уникальные функциональности, соответствующие ролям пользователей.

Для каждого уровня иерархии представлены свои прецеденты. Страховой агент работает с составленными страховыми полисами, рассматривает заявления о страховых случаях, оформляет новые страховые полисы. Клиент страховой компании может оставлять заявления на оформлении страховых полисов, писать заявления о страховых случаях, просматривать их. Гостю доступен каталог возможных страховых планов и калькулятор стоимости страхового полиса.

Готовая диаграмма представлена на следующем рисунке 1.

  
Рисунок 1 – Use-Case диаграмма

**Вывод:** в результате лабораторной работы изучены основные принципы объектно-ориентированного проектирования программного обеспечения, получены навыки проектирования функциональности системы с применением UML, а также составлена Use-case диаграмма ресторана, которая демонстрирует функционал и выделяет группы пользователей системы.

**Ответы на теоретические вопросы**

1. Дайте описание понятиям Unified process (UP) и UML.

Унифицированный процесс (Unified Process) – процесс разработки программного обеспечения, который обеспечивает упорядоченный подход к распределению задач и обязанностей в организации-разработчике.

Унифицированный язык визуального моделирования - Unified Modeling Language (UML) — это стандартная нотация визуального моделирования программных систем, принятая консорциумом Object Managing Group осенью 1997г., и на сегодняшний день поддерживаемая многими объектно-ориентированными CASE-продуктами.

1. Перечислите основные диаграммы UML 2.0.

Диаграммы деятельности, классов, взаимодействия, компонентов, составных структур, развертывания, обзора взаимодействия, объектов, пакетов, последовательности, конечных автоматов, временная, прецедентная.

1. Назовите CASE-средства, поддерживающие создание UML диаграмм.

Draw.io, Microsoft Visio, **StarUML**, Creately, **IDEF0 Diagram Software, Erwin**.

1. Укажите назначение диаграммы вариантов использования.

Отображает функции информационной системы, взаимодействие между актерами и функциями.

1. Опишите нотации, которые используются для построения Use-Case диаграммы.

*Актёр* – представляет лицо или систему, которое взаимодействует с системой или подсистемой. Актёры могут быть как внешними (например, пользователи, другие системы), так и внутренними (например, другие подсистемы или компоненты).

*Прецеденты* – описывают функциональность или поведение системы с точки зрения пользователя. Они представляют цели или задачи, которые пользователь или другой актёр может выполнить с помощью системы.

*Подсистема* – является частью системы и содержит группу связанных функций (прецедентов). Подсистемы используются для структурирования системы на более управляемые части.