Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра экономической информатики

ОТЧЕТ

ЛАБОРАТОРНАЯ РАБОТА №2

«Диаграмма последовательности»

Выполнил:

Студент ФКП, гр.814301

Мацаль Н.Ю

Проверил:

Лыщик А.П.

Минск 2020

**Цель:** Изучить принципы построения диаграмм последовательности, состояний, классов и их для своей модели.

**Теоретическая часть:**

**Диаграммы последовательностей используются для уточнения диаграмм прецедентов**, более детального описания логики сценариев использования. Это отличное средство документирования проекта с точки зрения сценариев использования!

Диаграммы последовательностей обычно содержат **объекты**, которые **взаимодействуют в рамках сценария**, **сообщения**, которыми они обмениваются, и **возвращаемые результаты**, связанные с сообщениями. Впрочем, часто возвращаемые результаты обозначают лишь в том случае, если это не очевидно из контекста.

**Объекты** обозначаются прямоугольниками с подчеркнутыми именами (чтобы отличить их от классов).

**Сообщения (вызовы методов)** - линиями со стрелками.

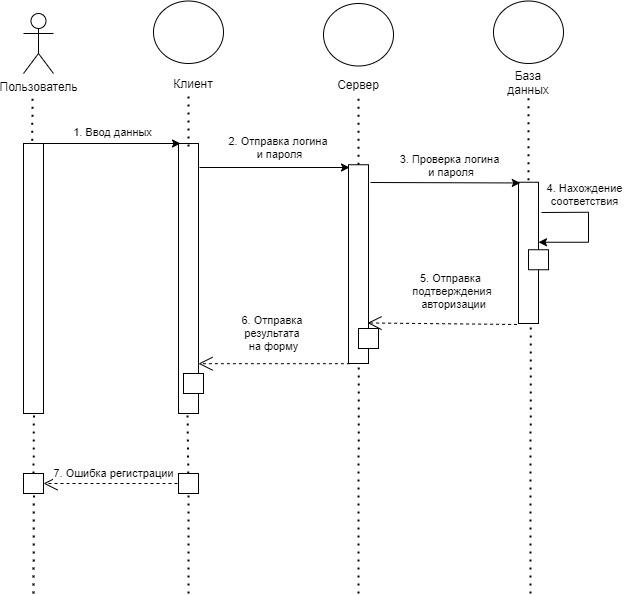
**Возвращаемые результаты** - пунктирными линиями со стрелками.

Прямоугольники на вертикальных линиях под каждым из объектов показывают **“время жизни” (фокус) объектов**. Впрочем, довольно часто их не изображают на диаграмме, все это зависит от индивидуального стиля проектирования.

Как было сказано выше, взаимодействие между актерами отображается при помощи специальных стрелок, передающих управление от отправителя (от кого идёт стрелка) к получателю (тот, к кому направлена стрелка). Стрелки демонстрируют ход сценария и те события, которые происходят во время анализируемого прецедента. Всего существует 5 видов стрелок:

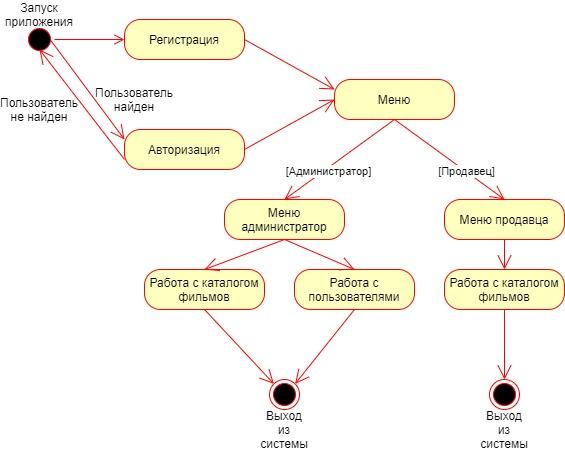
* **Синхронное сообщение** — актер-отправитель передаёт ход управления актеру-получателю, которому необходимо провести в прецеденте некоторое действие. Пока проводимое актером-получателем действие не будет завершено (соответственно, не будет получено ответное сообщение), актер-отправитель теряет возможность производить какие-либо действия. Графически изображается как сплошная линия со стрелкой в виде закрашенного треугольника, после которой идёт прямоугольник, отражающий деятельность объекта, в конце которого находится ответное сообщение.
* **Ответное cообщение** — данное сообщение является ответом на синхронное сообщение. Обычно, содержит какое-либо возвращаемое изначальному актеру-отправителю значение, также возвращающее ему управление (возможность действовать). Графически изображается пунктирной линией с открытой стрелкой.
* **Асинхронное сообщение** — актер-отправитель передаёт ход управления актеру-получателю, которому необходимо провести в прецеденте некоторое действие. Основное отличие от синхронного сообщения состоит в том, что актер-отправитель не теряет возможности совершать другие действия. Графически изображается сплошной линией с открытой стрелкой.
* **Потерянное сообщение** — сообщение без адресата (есть отправитель, нет получателя).
* **Найденное сообщение** — сообщение без отправителя.

Последние два вида стрелок (взаимодействий) используются крайне редко. В основном они используются для демонстрации взаимодействия имеющихся объектов в данном прецеденте с внешними системами.



***Диаграмма состояний*** — это, по существу, диаграмма состояний из теории автоматов со стандартизированными условными обозначениями, которая может определять множество систем от компьютерных программ до бизнес-процессов. Используются следующие условные обозначения:

* Круг, обозначающий начальное состояние.
* Окружность с маленьким кругом внутри, обозначающая конечное состояние (если есть).
* Скруглённый прямоугольник, обозначающий состояние. Верхушка прямоугольника содержит название состояния. В середине может быть горизонтальная линия, под которой записываются активности, происходящие в данном состоянии.
* Стрелка, обозначающая переход. Название события (если есть), вызывающего переход, отмечается рядом со стрелкой. Охраняющее выражение может быть добавлено перед «/» и заключено в квадратные скобки (название\_события[охраняющее\_выражение]), что значит, что это выражение должно быть истинным, чтобы переход имел место. Если при переходе производится какое-то действие, то оно добавляется после «/» (название\_события[охраняющее\_выражение]/действие).
* Толстая горизонтальная линия с либо множеством входящих линий и одной выходящей, либо одной входящей линией и множеством выходящих. Это обозначает объединение и разветвление соответственно.



**Диаграмма классов** (*class diagram*) — структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

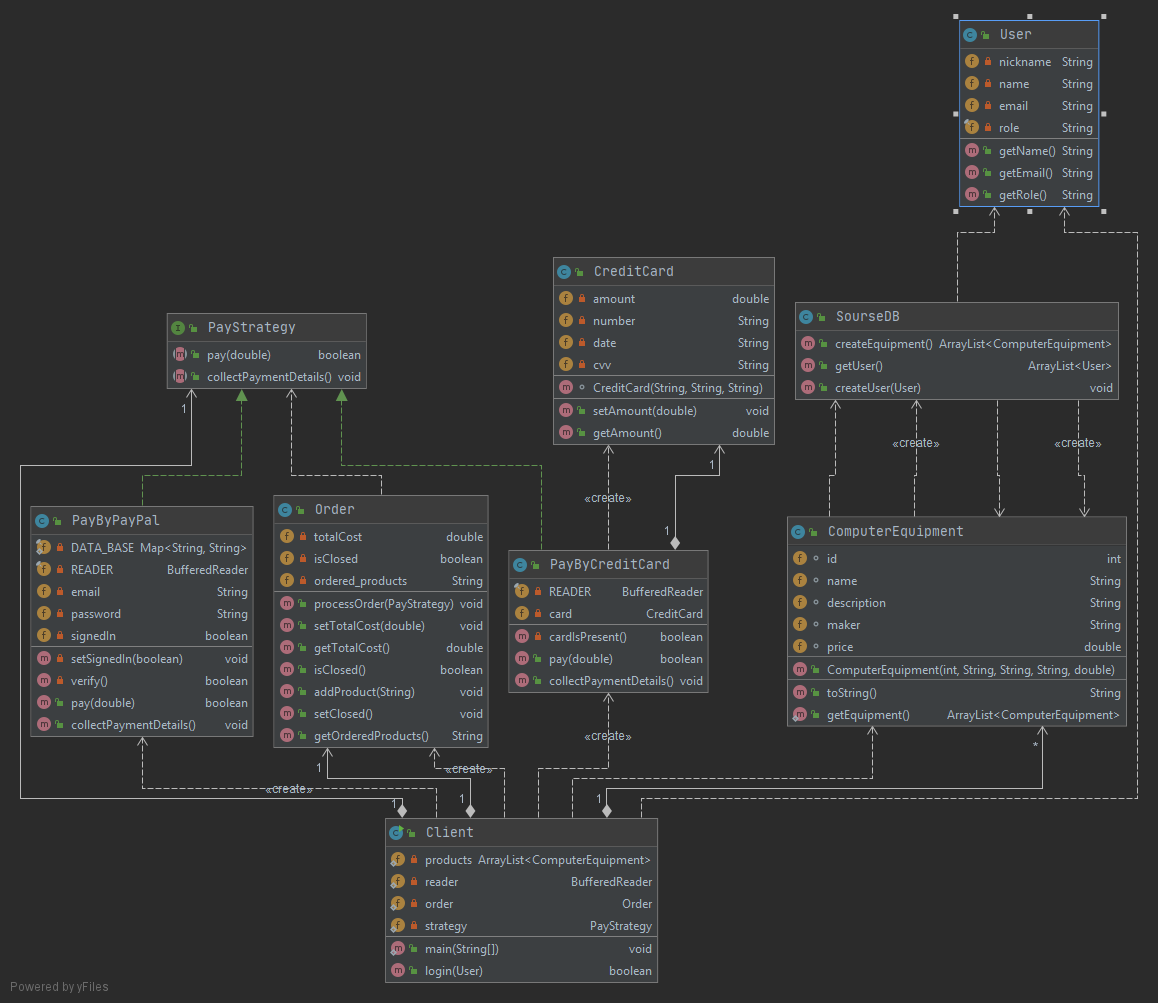
Целью создания диаграммы классов является графическое представление статической структуры декларативных элементов системы (классов, типов и т. п.) Она содержит в себе также некоторые элементы поведения (например — операции), однако их динамика должна быть отражена на диаграммах других видов (диаграммах коммуникаций, диаграммах состояний). Для удобства восприятия диаграмму классов можно также дополнить представлением пакетов, включая вложенные.

При представлении сущностей реального мира разработчику требуется отразить их текущее состояние, их поведение и их взаимные отношения. На каждом этапе осуществляется абстрагирование от маловажных деталей и концепций, которые не относятся к реальности (производительность, инкапсуляция, видимость и т. п.). Классы можно рассматривать с позиции различных уровней. Как правило, их выделяют три основных: аналитический уровень, уровень проектирования и уровень реализации:

* на уровне анализа класс содержит в себе только набросок общих контуров системы и работает как логическая концепция предметной области или программного продукта.
* на уровне проектирования класс отражает основные проектные решения касательно распределения информации и планируемой функциональности, объединяя в себе сведения о состоянии и операциях.
* на уровне реализации класс дорабатывается до такого вида, в каком он максимально удобен для воплощения в выбранной среде разработки; при этом не воспрещается опустить в нём те общие свойства, которые не применяются на выбранном языке программирования.

Диаграмма классов является ключевым элементом в объектно-ориентированном моделировании. На диаграмме классы представлены в рамках, содержащих три компонента:

* В верхней части написано имя класса. Имя класса выравнивается по центру и пишется полужирным шрифтом. Имена классов начинаются с заглавной буквы. Если класс абстрактный — то его имя пишется полужирным курсивом.
* Посередине располагаются поля (атрибуты) класса. Они выровнены по левому краю и начинаются с маленькой буквы.
* Нижняя часть содержит методы класса. Они также выровнены по левому краю и пишутся с маленькой буквы.



**Вывод:** Была разработать диаграмма вариантов использования, мы ознакомились с основными элементами определения, представления, проектирования и моделирования программных систем с помощью UML, программно реализовали схему на языке Java.