Focused forward

Netcracker

Лекция 1.

Введение. ЖЦПО. Моделирование бизнес-процессов. UML

Web System Design and Development

План лекции

- Регламент курса
- Программное обеспечение (ПО)
- Этапы разработки ПО
- Визуальное моделирование и его средства
- Понятие и применение UML
- Виды диаграмм UML
- Диаграммы вариантов использования (Use Case Diagram)
- Диаграммы классов (Class Diagram)
- Рекомендации по изображению UML диаграмм

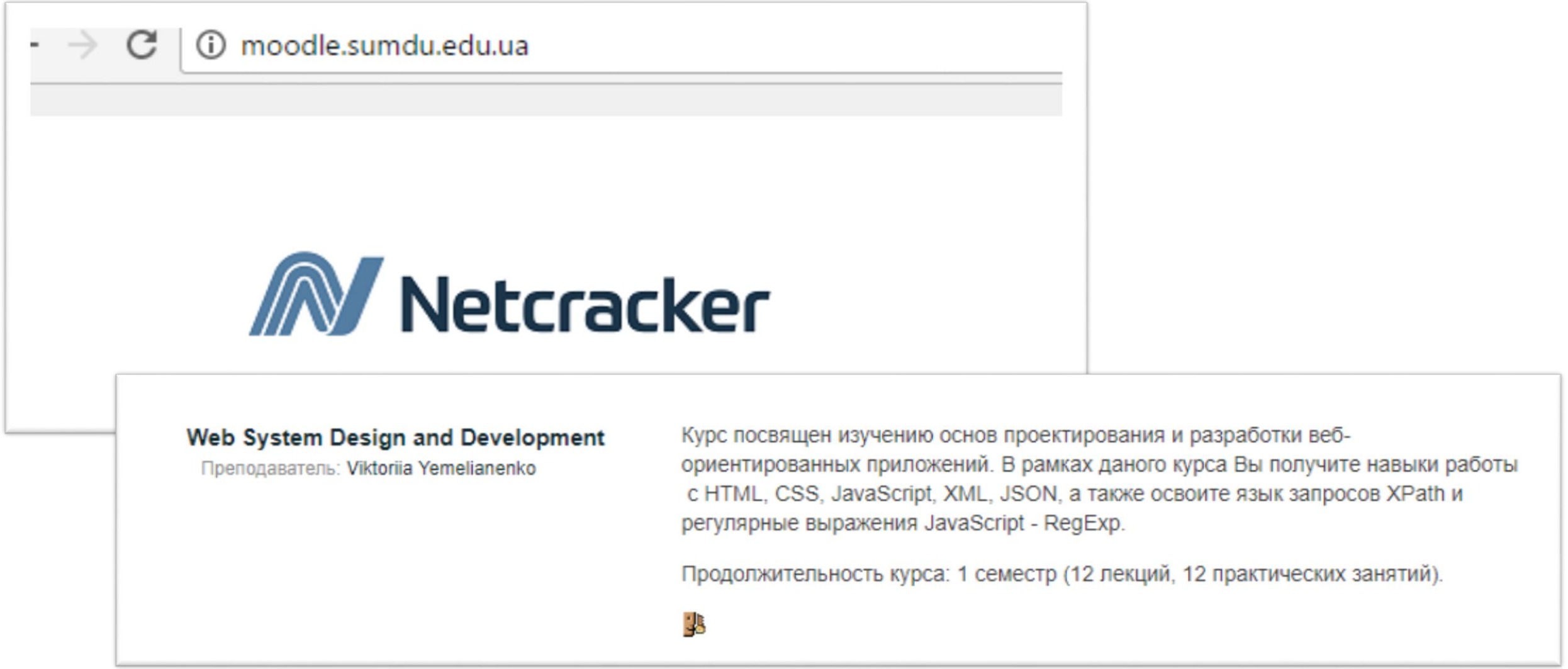
Регламент курса

Продолжительность курса: **1 семестр**

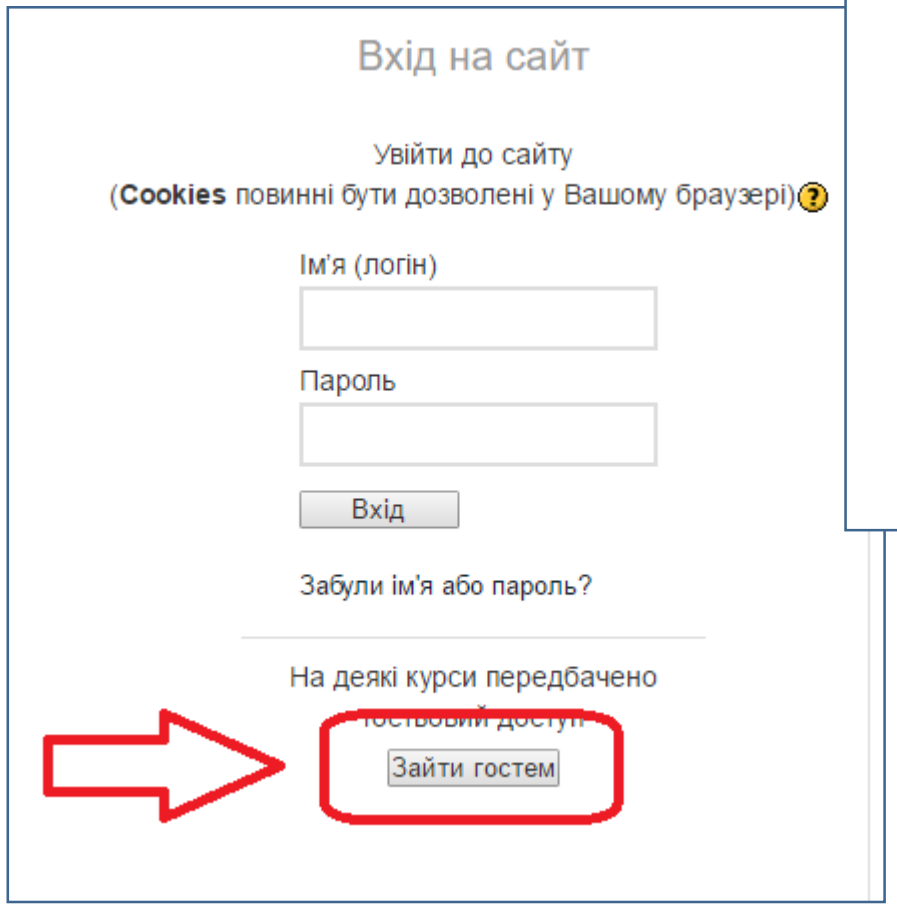
- **12** лекций
- **12** практических занятий
- **2-4** консультации
- **2** модульных контроля
- **1** обязательное домашнее задание

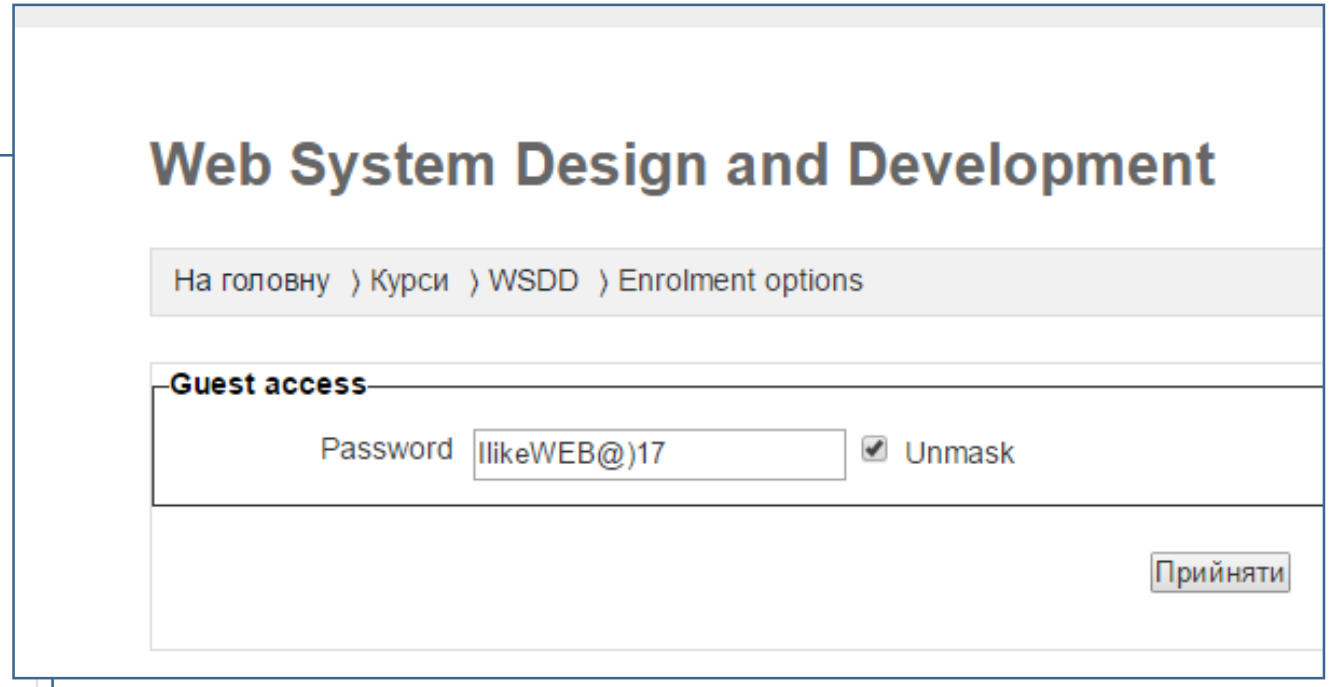
Форма контроля - **экзамен**

Регламент курса



Регламент курса



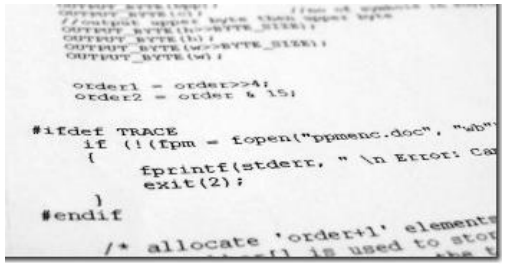


IlikeWEB@)17

Программное обеспечение (ПО)

- **программа** (набор исполняемых инструкций)

interface (GUI / CLI) vs. source code *translation* → (byte code) *translation* → machine code



```
L5 (176)
 ALOAD token
 BIPUSH 76
 INVOKEVIRTUAL String.inde>
 ISTORE startReference
L6 (177)
```

compilation = translation without execution
interpretation = translation with execution

- **данные**
  - переменные
  - константы

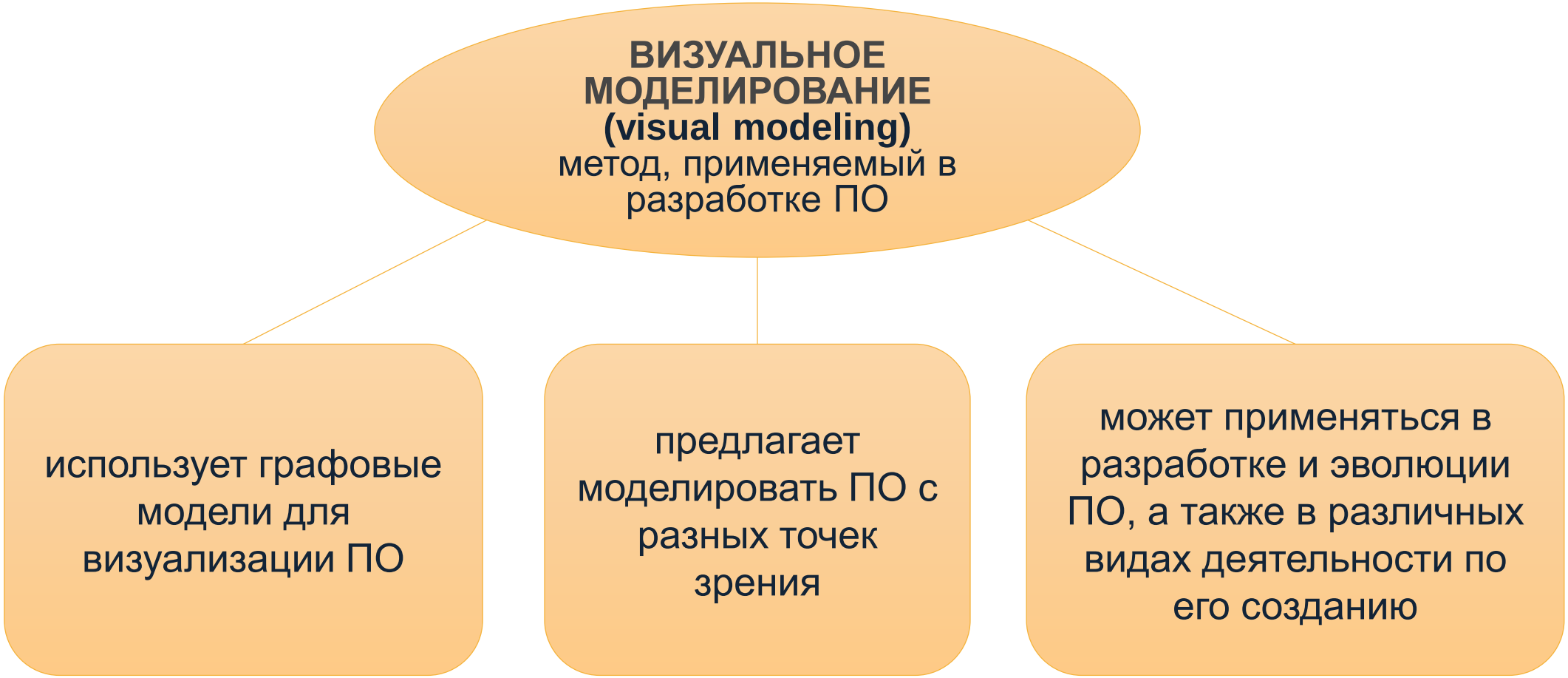
- **документация**
  - требования
  - спецификация
  - мануалы
  - прочее

Программное обеспечение (ПО)

Этапы разработки ПО

Название	Суть этапа
Initiation	Получение согласия заказчика и исполнителя на условия проекта (масштаб, время, стоимость)
Analysis	Детальное определение требований заказчика, high-level design (проектирование)
Design	Детальное проектирование ПО
Implementation	Непосредственно разработка ПО согласно требованиям заказчика и внутренним критериям качества
Testing	Обнаружение и исправление несогласованностей реального и ожидаемого поведения системы
Deployment	Приёмо-сдаточное тестирование и развёртывание ПО на стороне заказчика
Support	Поддержка работы ПО

Визуальное моделирование и его средства



Визуальное моделирование и его средства

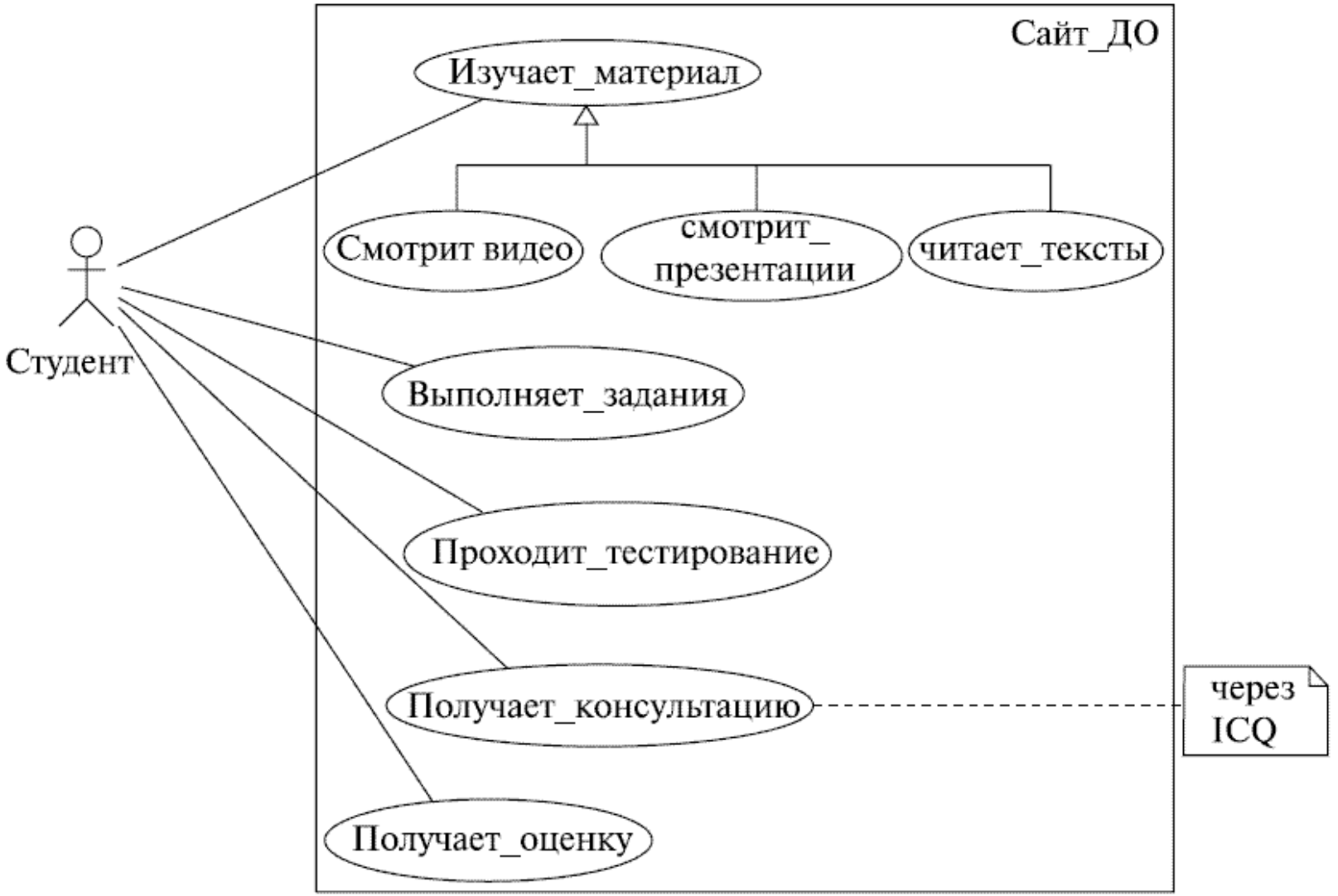
- **Визуальное моделирование применяется на практике с помощью языков, методов и соответствующих программных инструментов.**

UML, BPMN, SADT/IDEF0, IDEF1X, SDL, MSC

RUP/USDP, Use case, SADT

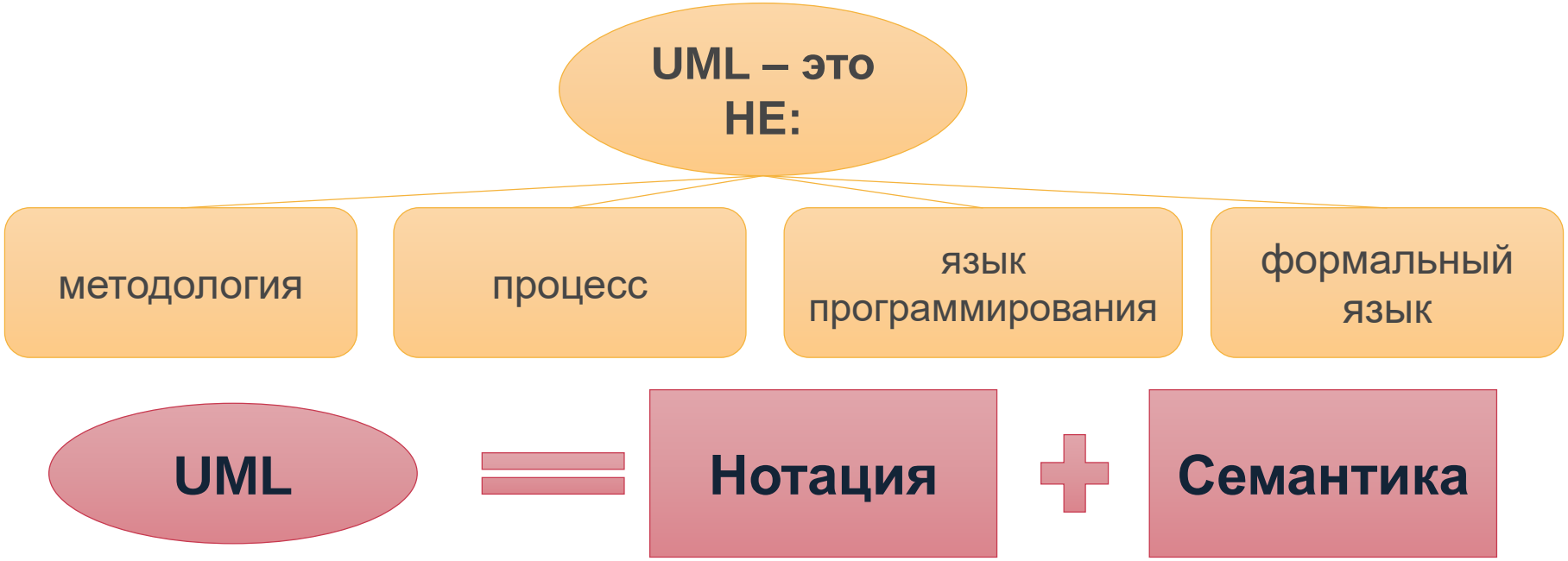
IBM Rational Rose, Borland Together, Telelogic Tau Suit, MS Visio UML Add-on

Понятие и применение UML



Понятие и применение UML

UML (Unified Modeling Language) — унифицированный язык моделирования для описания, визуализации и документирования объектно-ориентированных систем в процессе их анализа и проектирования.



Понятие и применение UML

Легко воспринимаемый и выразительный язык визуального моделирования, специально **предназначенный для разработки и документирования моделей сложных систем** различного целевого назначения.

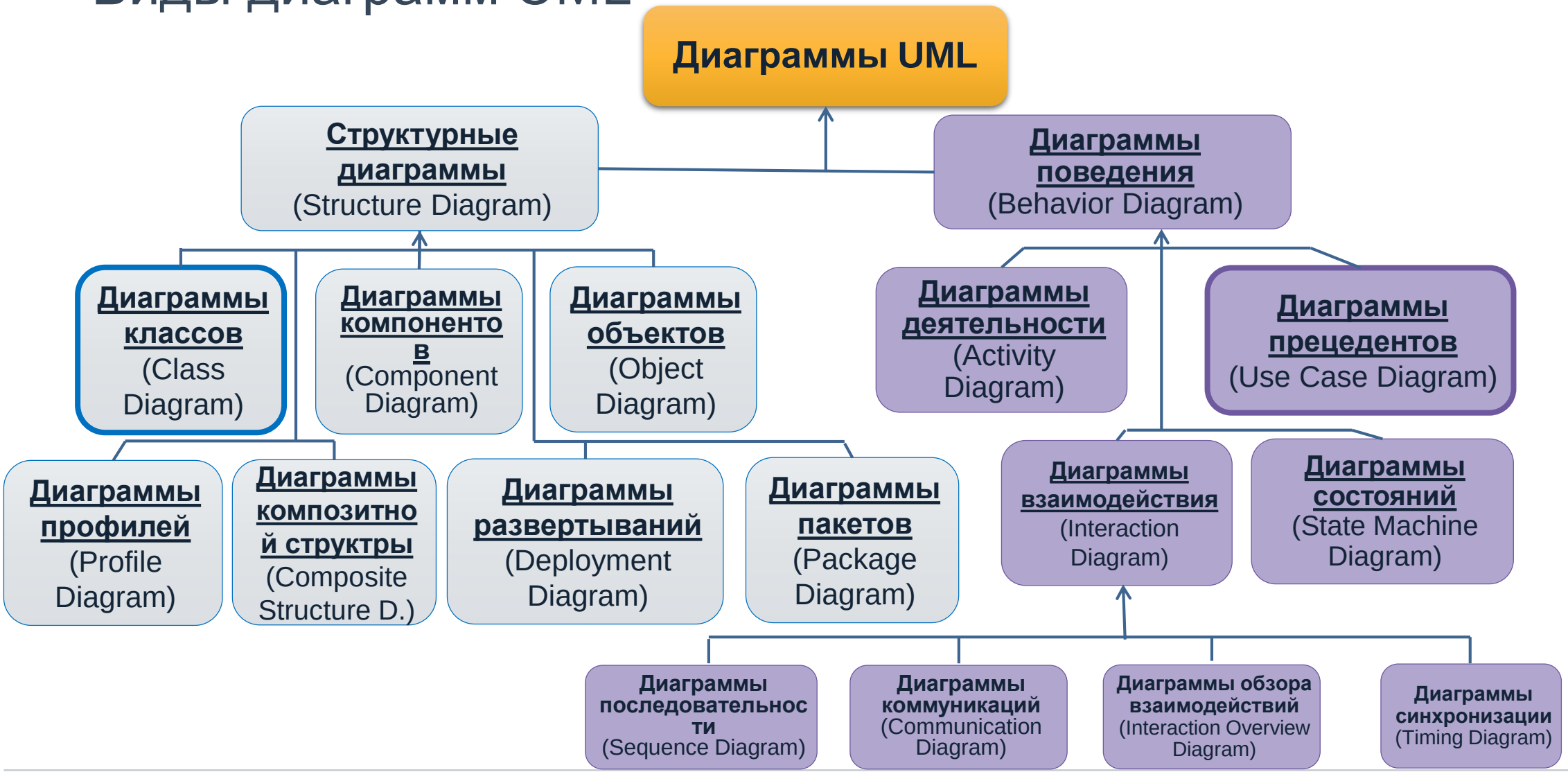
У UML есть **возможность расширения** и специализации для **более точного представления моделей систем в конкретной предметной области.**

Графическое представление моделей в нотации UML **не должно зависеть от конкретных языков программирования и инструментальных средств проектирования.**

Облегчает контроль хода разработки благодаря повышению уровня **абстракции**.

Упрощает взаимодействие между разработчиками ПО и представителями других специальностей.

Виды диаграмм UML



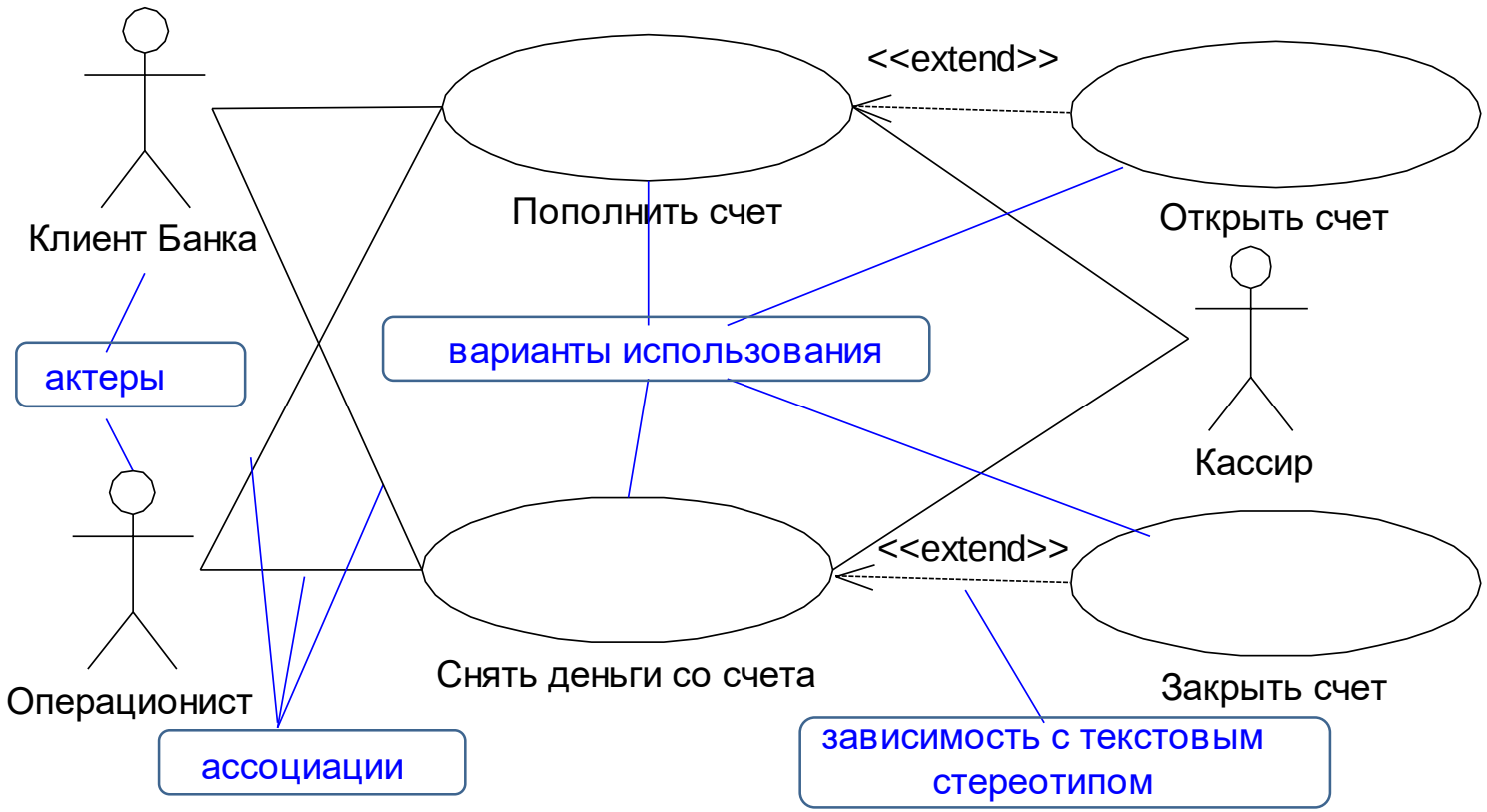
Диаграммы вариантов использования (Use Case D.)

Диаграмма вариантов использования - диаграмма, на которой изображаются варианты использования проектируемой системы,

заключенные в границу системы,

и внешние актеры,

а также определенные отношения между актерами и вариантами использования



Диаграммы вариантов использования (Use Case D.)

Диаграммы вариантов использования
показывают взаимодействия между вариантами использования и действующими лицами, отражая функциональные требования к системе с точки зрения пользователя.

Цель построения **диаграмм вариантов использования** – документирование функциональных требований в самом общем виде.

Описывает **функциональное назначение системы**, т.е. то, что система будет делать в процессе своего функционирования.

Диаграммы вариантов использования (Use Case D.)

Цели построения

1) определить **общие границы** и **контекст** моделируемой предметной области на начальных этапах проектирования;

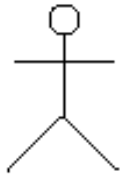
2) сформулировать **общие требования** к функциональному проектированию системы;

3) разработать **исходную концептуальную модель** системы для ее последующей реализации;

4) подготовить **документацию** для взаимодействия *разработчика* системы с ее *заказчиком* и *пользователями*.

Диаграммы вариантов использования (Use Case D.)

Основные обозначения

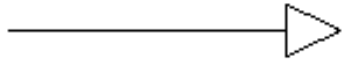


Актер (actor)

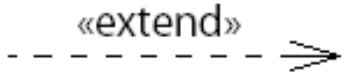
Вариант использования (use case)

Граница системы (system boundry)

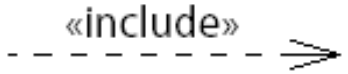
ассоциация



обобщение



расширение



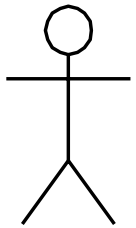
включение

Диаграммы вариантов использования (Use Case D.)

Актер (actor)

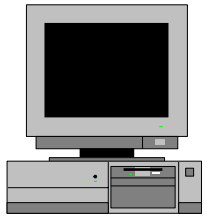
- любая **внешняя по отношению к проектируемой системе сущность**, которая взаимодействует с системой и использует ее функциональные возможности для достижения определенных целей или решения частных задач.

***Примеры актеров*:** кассир, клиент банка, банковский служащий, президент, продавец магазина, менеджер отдела продаж, пассажир авиарейса, водитель автомобиля, администратор гостиницы, сотовый телефон



Клиент банка

<<actor>>
Посетитель Интернет-магазина



Удаленный пользователь

Диаграммы вариантов использования (Use Case D.)

Вариант использования (Use Case)

Определяет последовательность действий, которая должна быть выполнена проектируемой системой при взаимодействии ее с соответствующим актером.

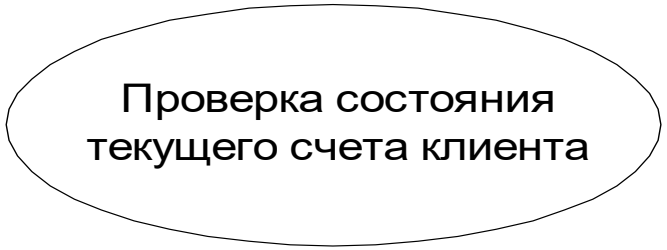
Отвечает на вопрос **«Что должна выполнять система?»**, не отвечая на вопрос «Как она должна выполнять это?».

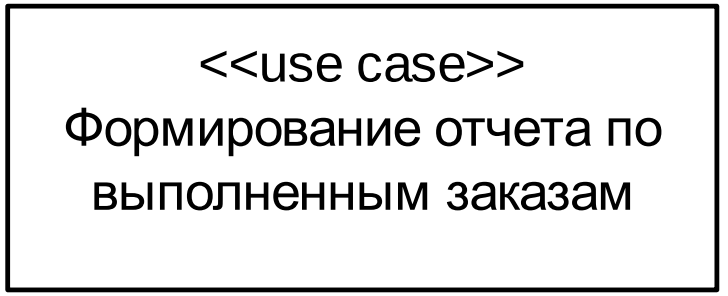
Имена – отглагольное существительное или глагол в неопределенной форме.

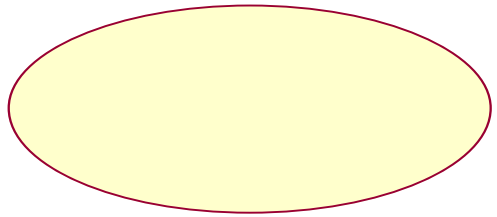
Имя ВИ начинается с большой буквы и обозначается оборотом глагола или существительного, обозначающего действие.

Диаграммы вариантов использования (Use Case D.)

Примеры Use Case:



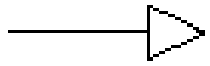




Получение справки о
состоянии счета

Диаграммы вариантов использования (Use Case D.)

Отношения на диаграмме вариантов использования

Relationship	*Function*	*Notation*
ассоциация		
включение	**Специфицирует тот факт, что некоторый вариант использования содержит поведение, определенное в другом варианте использования**	
расширение	**Определяет взаимосвязь одного варианта использования с другим, функциональность или поведение которого задействуется первым не всегда, а только при выполнении некоторых дополнительных условий**	
обобщение	**Предназначено для спецификации того факта, что один элемент модели является специальным или частным случаем другого элемента модели**	

Диаграммы вариантов использования (Use Case D.)

Отношения

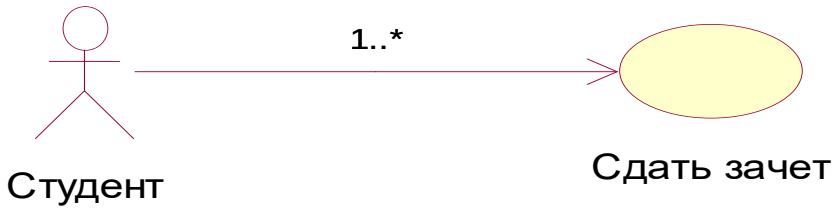
Один актер может взаимодействовать с несколькими вариантами использования и наоборот.

2 варианта использования, определенные для одной и той же сущности, не могут взаимодействовать друг с другом, т.к. любой из них самостоятельно описывает законченный вариант использования этой сущности.

Диаграммы вариантов использования (Use Case D.)

Отношение ассоциации

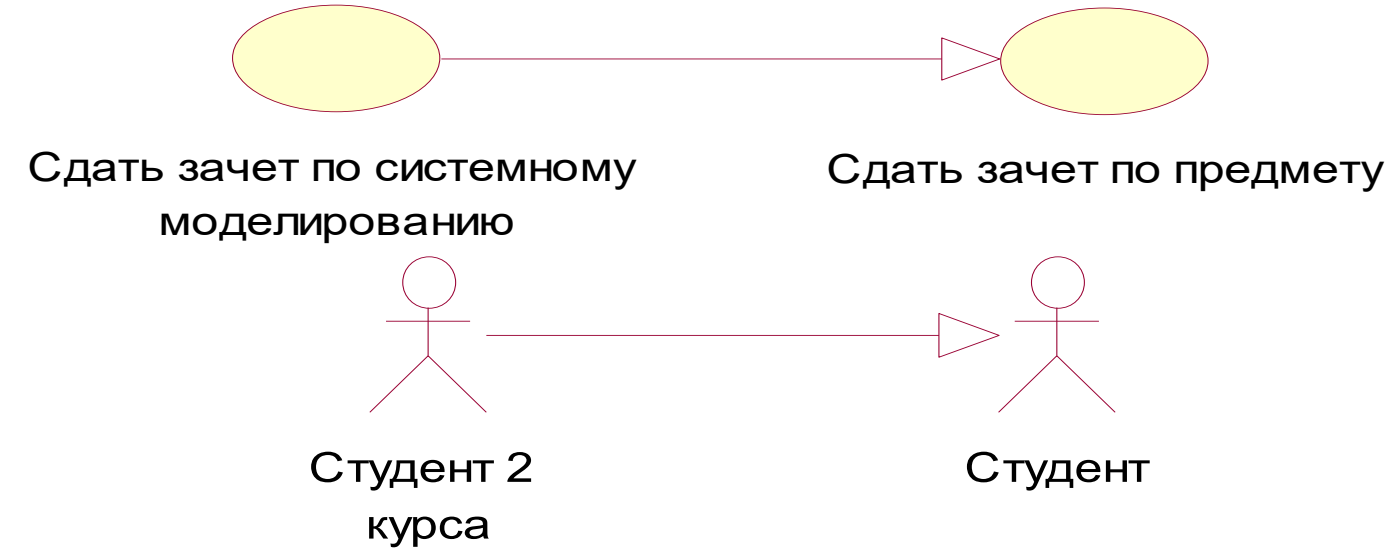
устанавливает, какую конкретную роль играет актер при взаимодействии с экземпляром варианта использования.



Диаграммы вариантов использования (Use Case D.)

Отношение обобщения

Служит для указания того факта, что некоторый вариант использования А может быть обобщен до варианта использования Б (или актер А может быть обобщен до актера Б).

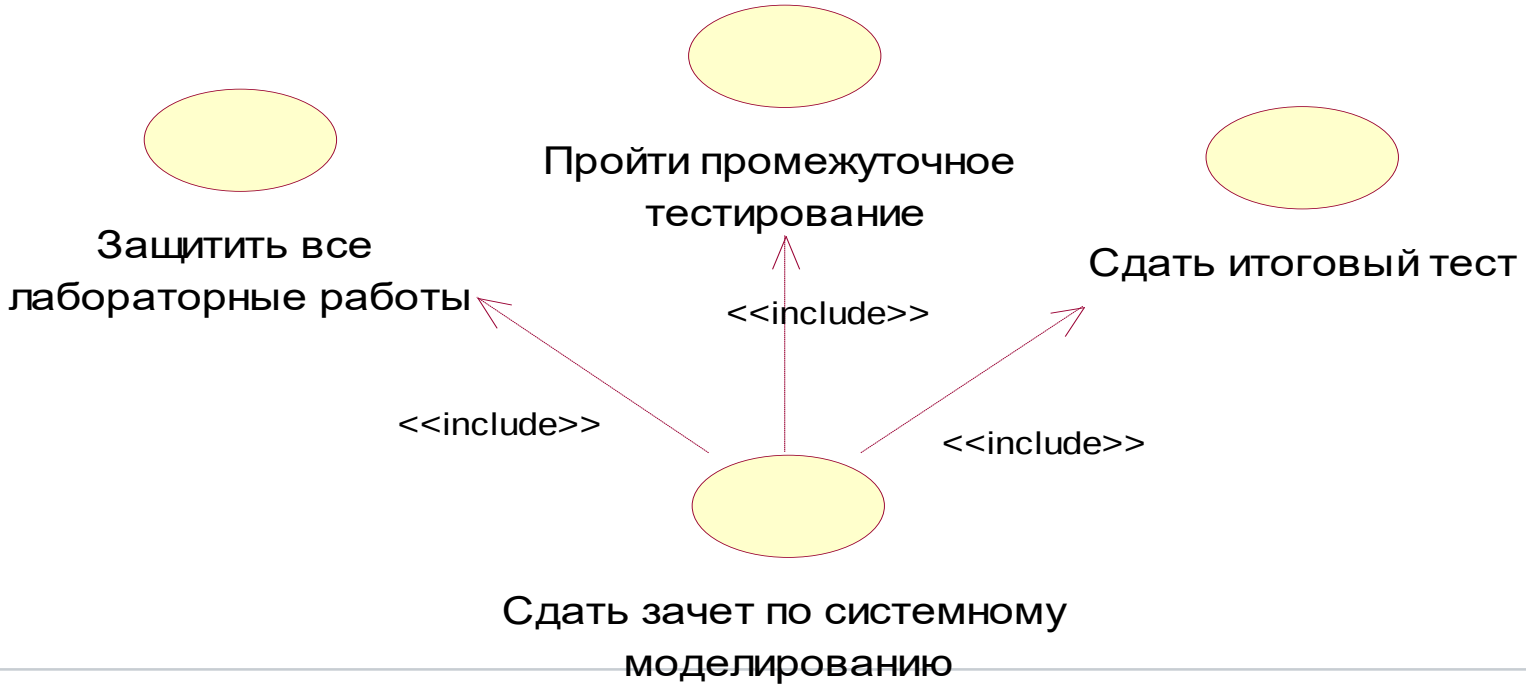


Стрелка указывает в сторону родительского ВИ (актера)

Диаграммы вариантов использования (Use Case D.)

Отношение включения

Указывает, что некоторое заданное поведение для одного варианта использования включается в качестве составного компонента в последовательность поведения другого варианта использования



Диаграммы вариантов использования (Use Case D.)

Отношение расширения

Определяет взаимосвязь базового варианта использования с некоторым другим вариантом использования, функциональное поведение которого задействуется базовым не всегда, а только при выполнении некоторых дополнительных условий.

Сдать зачет

Взять индивидуальную ведомость

Стрелка указывает на базовый вариант использования!

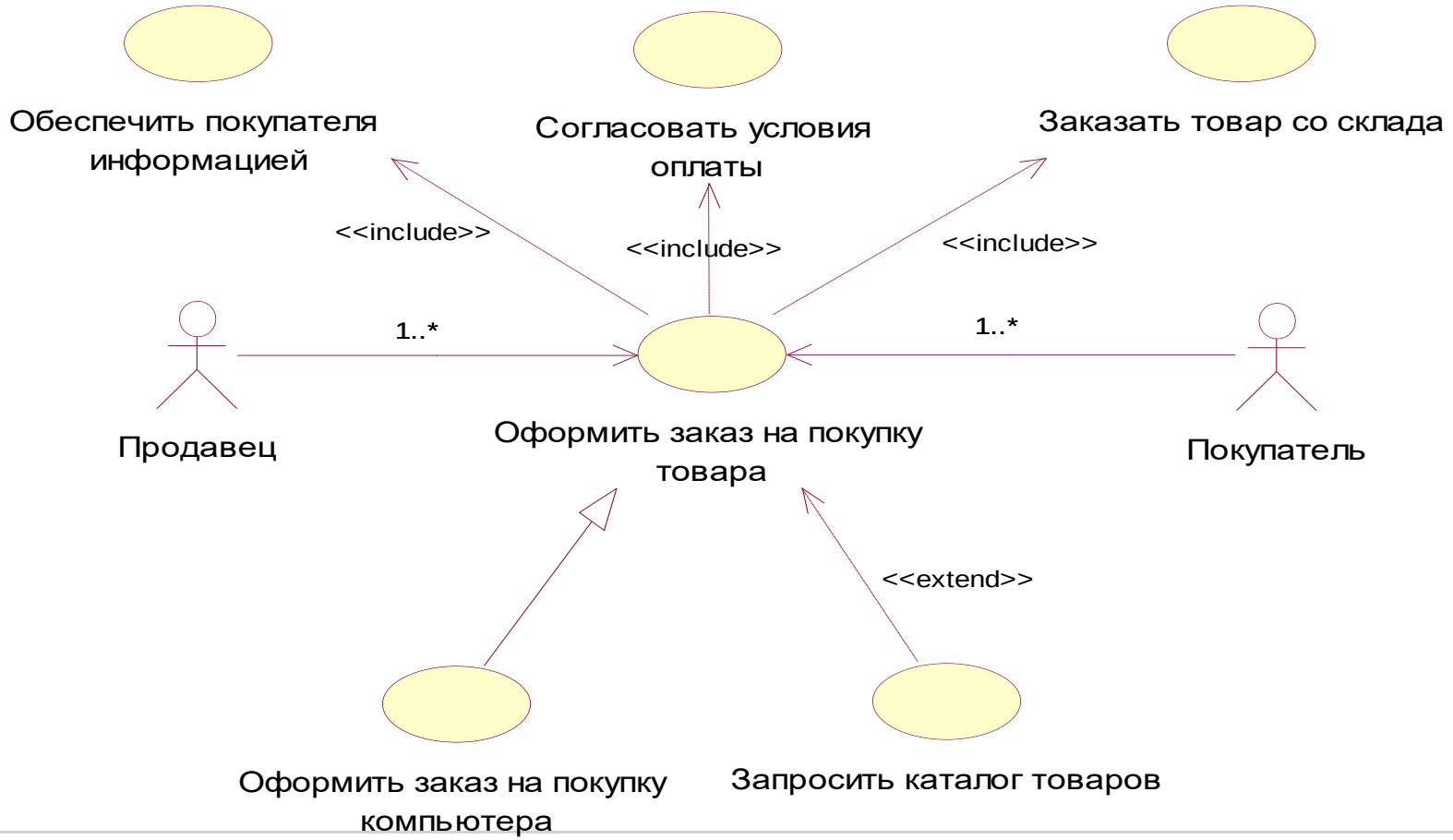
Диаграммы вариантов использования (Use Case D.)

Достоинства модели вариантов использования :

- Определяет пользователей и границы системы;
- Определяет системный интерфейс;
- Удобна для общения пользователей с разработчиками;
- Используется для написания тестов;
- Является основой для написания пользовательской документации;
- Хорошо вписывается в любые методы проектирования (как объектно-ориентированные, так и структурные).

Диаграммы вариантов использования (Use Case D.)

ДВИ процесса оформления заказа на покупку товара

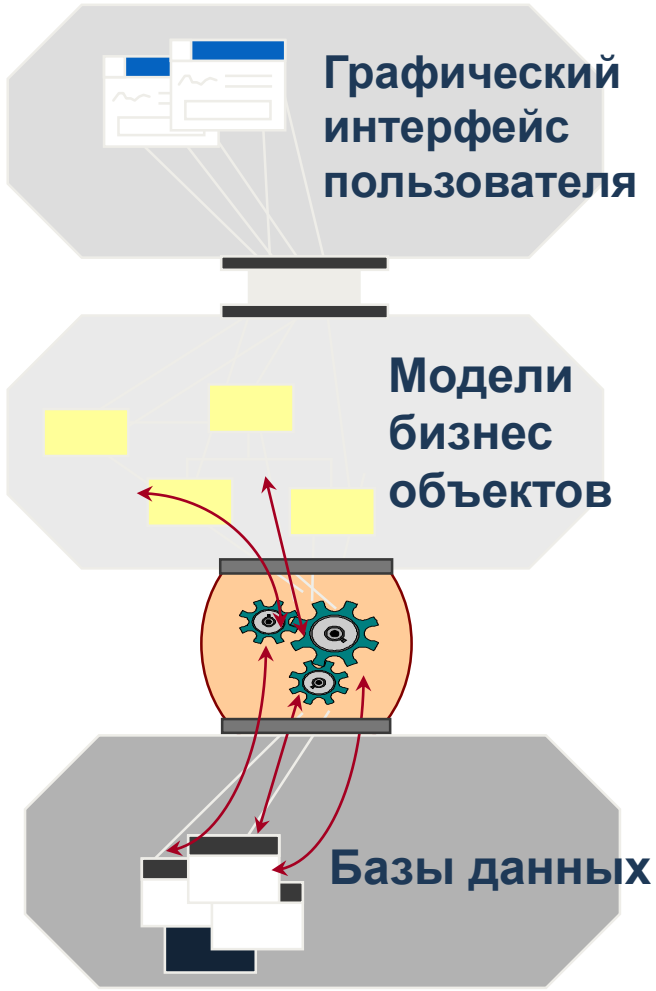


Диаграммы классов (Class Diagram)

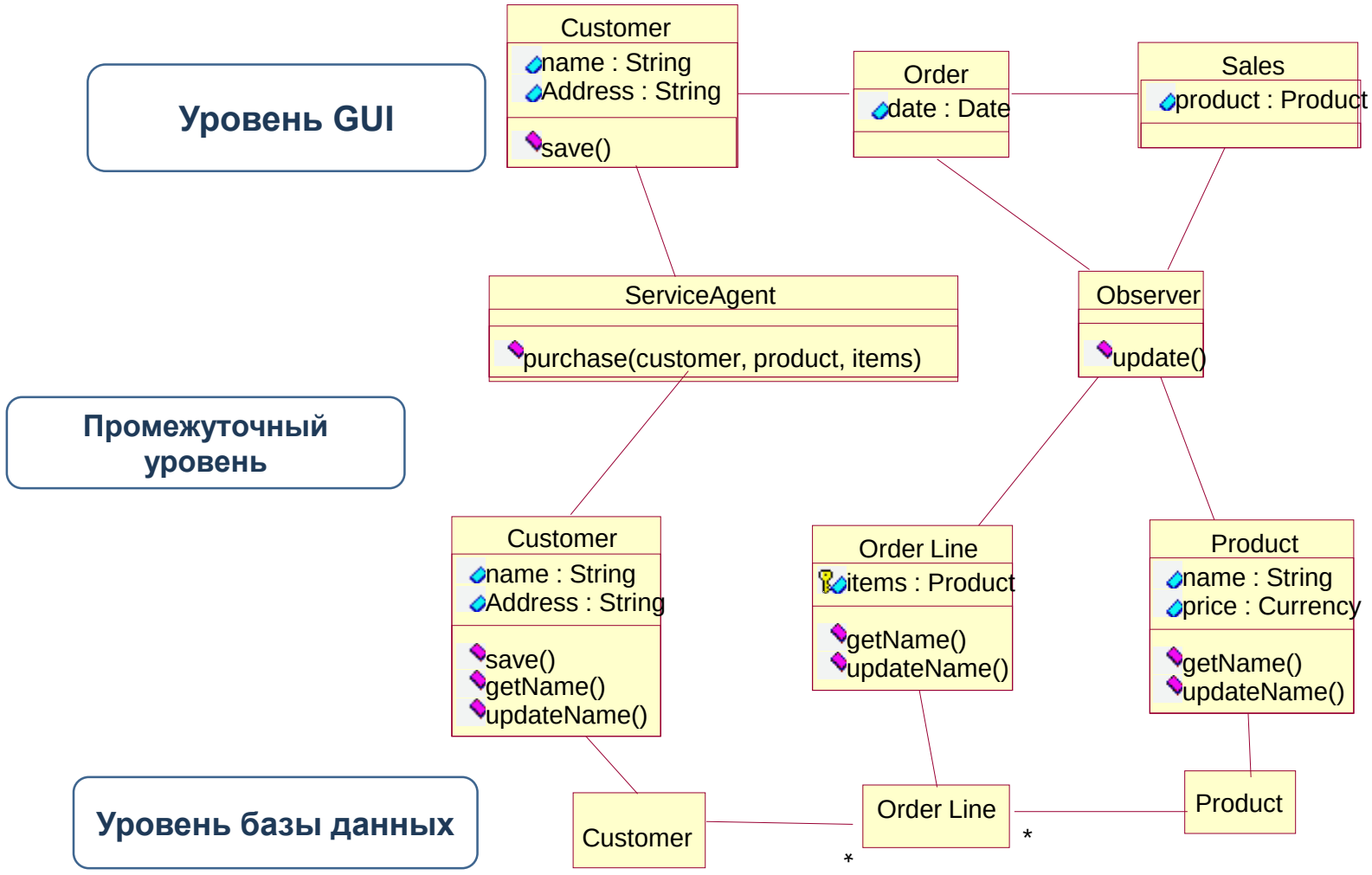
Программная система

Проблема:

Как представить архитектуру программной системы?

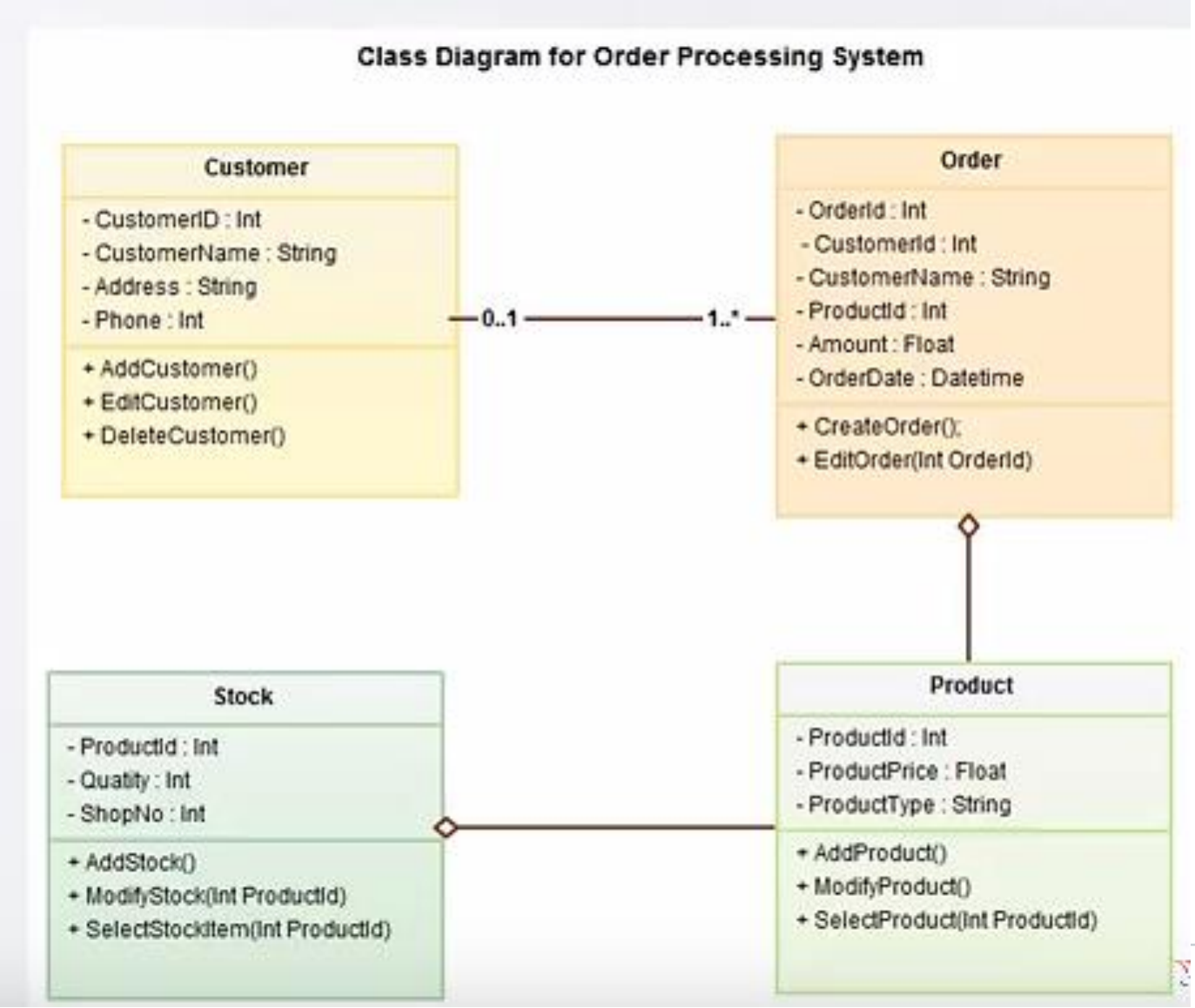


Диаграммы классов (Class Diagram)



Архитектура программной системы в нотации UML

Диаграммы классов (Class Diagram)



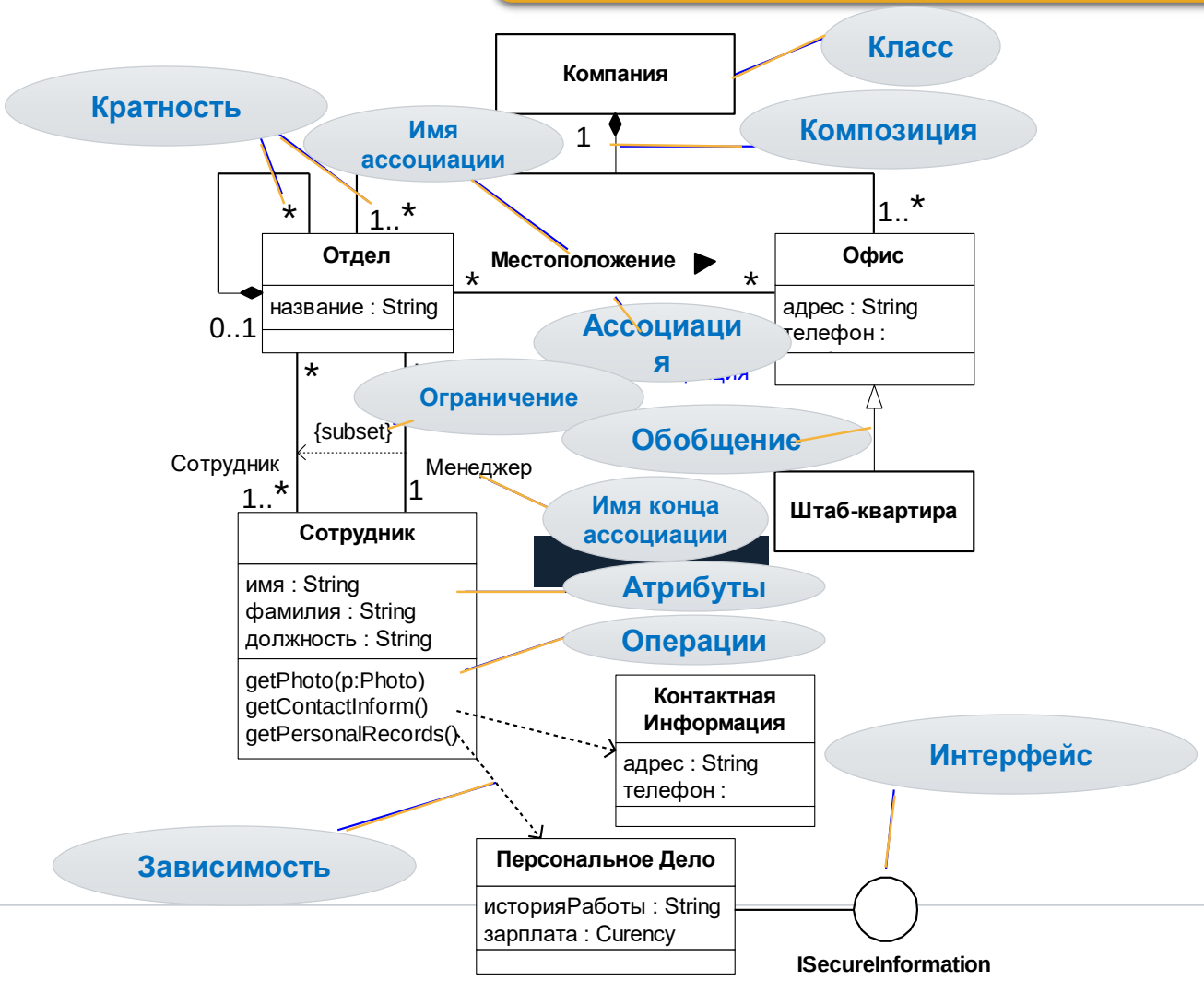
Диаграммы классов (Class Diagram)

Диаграмма классов — основная логическая модель проектируемой системы

- ***Диаграмма классов (class diagram)*** — диаграмма, предназначенная **для представления модели статической структуры программной системы** в терминологии классов объектно-ориентированного программирования.
- Диаграмма классов представляет собой **граф, вершинами или узлами которого являются элементы типа “классификатор”,** которые **связаны различными типами структурных отношений**.
- ***Классификатор*** *(classifier)* – специальное понятие, предназначенное для классификации экземпляров, которые имеют общие характеристики.

Диаграммы классов (Class Diagram)

Основные обозначения на диаграмме



Рекомендации по изображению UML диаграмм

Общие рекомендации по изображению диаграмм в нотации языка UML

- Каждая диаграмма должна служить **законченным представлением** соответствующего **фрагмента моделируемой предметной области**.
- Все **сущности** на диаграмме модели должны быть **одного концептуального уровня**.
- **Вся информация** о сущностях должна быть **явно представлена** на диаграммах.
- Диаграммы **не должны содержать противоречивой информации**.
- Диаграммы **не следует перегружать текстовой информацией**.
- Каждая диаграмма должна быть **самодостаточной для правильной интерпретации** всех ее элементов и **понимания семантики** всех используемых графических символов.

Материалы для самостоятельного изучения

1. https://en.wikipedia.org/wiki/Unified_Modeling_Language
2. http://www.uml-diagrams.org/
3. https://creately.com – удобный онлайн сервис для создания UML диаграмм
4. http://www.codemanship.co.uk/parlezuml/tutorials/usecases/usecases_intro.pdf
5. https://habrahabr.ru/post/150041/ - статья о UML диаграммах классов
6. https://www.smartdraw.com/uml-diagram/
7. https://www.google.com/

Q&A

Thank You

Focused forward

Netcracker