

Основные нотации для моделирования бизнес- процессов





Алина Загидуллина

Head of digital products, РЖД-Медицина

- ✳ >4 лет работала в операционном консалтинге в большой четверке (Deloitte, KPMG) с фокусом на проекты по оптимизации бизнес-процессов и разработке программ диджитализации;
- ✳ Делала проекты для различных индустрий, среди которых - ритейл, нефтяная промышленность, телеком, банки и транспорт.
- ✳ Также работала в VK (раньше Mail.ru Group), в отделе аналитики и эффективности, где разрабатывала сценарии развития для таких продуктов как ВКонтакте, GeekBrains, Юла, Delivery Club, Одноклассники и многих других.



План курса

1

Введение в операционную модель

2

Введение в бизнес-процессы

3

Декомпозиция процессов

4

Описание бизнес-процессов

5

Основные нотации описания бизнес-процессов: BPMN

6

Основные нотации описания бизнес-процессов: UML

7

Анализ процессов для выявления проблемных зон

8

Формирование предварительных гипотез по улучшению процессов

9

Детальная подготовка инициатив по оптимизации

10

Планирование и контроль проекта

11

Непрерывный процесс совершенствования



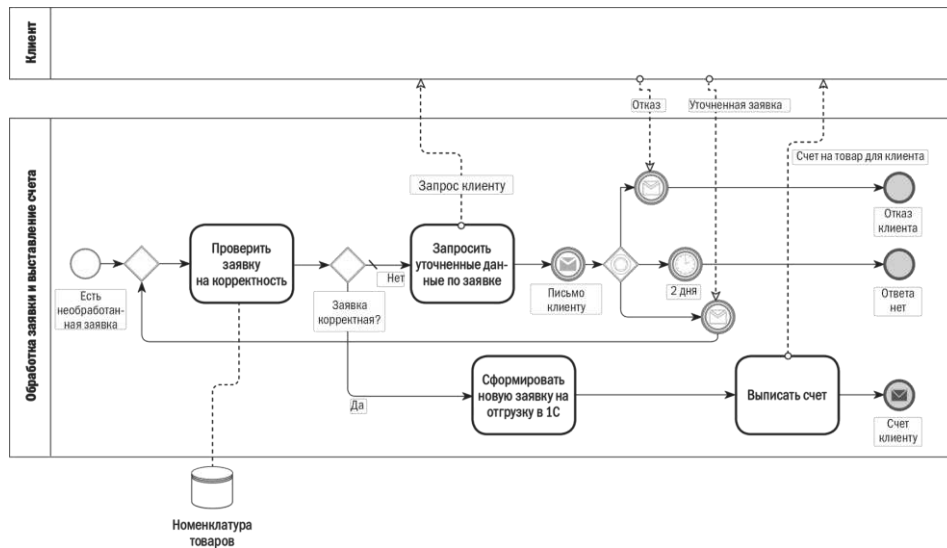
Что будет на уроке сегодня

- 📌 Познакомимся с нотацией UML;
- 📌 Узнаем, как применять данную нотацию, какие у нее ключевые элементы и важные правила использования.



Нотация

- Совокупность графических элементов, которые используются для описания бизнес-процессов компании
- Синтаксис графического языка моделирования
- Правила составления графических моделей, чтобы фиксировать бизнес-процессы для анализа и оптимизации





UML



UML



Unified Modeling Language (UML) – это унифицированный язык моделирования

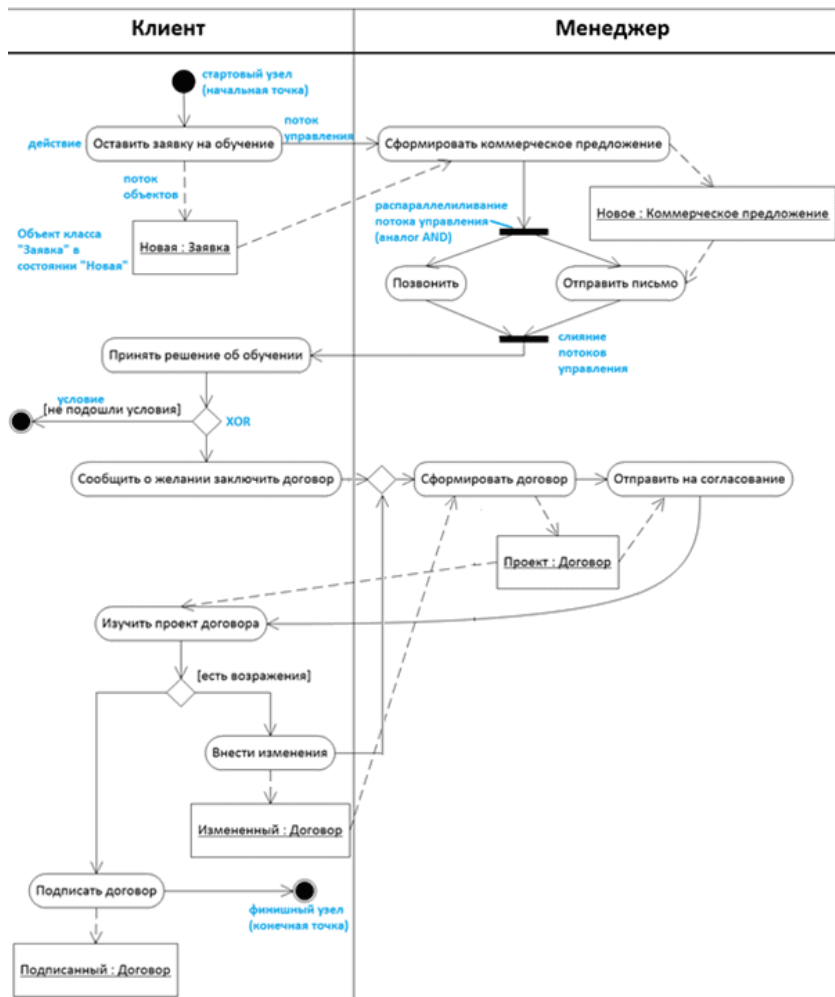
Modeling подразумевает создание модели, описывающей объект. **Unified** (универсальный, единый) — подходит для широкого класса проектируемых программных систем, различных областей приложений, типов организаций, уровней компетентности, размеров проектов, а в нашем случае и для бизнес-процессов.

Нотация использует объектно-ориентированные методы (методология, основанная на представлении программы/процесса в виде совокупности взаимодействующих объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования).



Пример:

Процесс «Подписание договора на обучение» в UML





Типы диаграмм UML



Типы диаграмм UML (1/3)

В языке UML существуют следующие типы диаграмм:

→ **Use-case diagram**

Диаграмма прецедентов: в основе – Actor (исполнитель), который устанавливает логические связи между ролями и прецедентами (вариантами использования).

→ **Class diagram**

Диаграмма классов: представляет собой набор статических и декларативных элементов модели, имеющие общие атрибуты и операции. Диаграмма имеет наиболее полное и развернутое описание связей в программном коде, функциональности и информации об отдельных классах

→ **Activity diagram**

Диаграмма активностей: отображает динамические аспекты поведения и общее представление о работе системы в формате блок-схемы. Диаграмма необходима для описания бизнес-процессов, взаимодействия нескольких систем, логики процедур и потоков работ, особенно при переходе от одной деятельности к другой



Типы диаграмм UML (2/3)

В языке UML существуют следующие типы диаграмм:

→ **Sequence diagram**

Диаграмма последовательности: описывает поведенческие аспекты системы, вид сообщений и уточняет прецедентов. Необходима для отображения взаимодействия объектов в динамике и во времени, подразумевает обмен сообщениями в рамках конкретного сценария

→ **Deployment diagram**

Диаграмма развертывания: отображает графическое представление инфраструктуры, а именно распределение компонентов системы по узлам и маршруты их соединений. Диаграмма организует компоненты и решает второстепенные задачи, связанные с определенным аспектом бизнес-процесса

→ **Collaboration diagram**

Диаграмма сотрудничества: диаграмма взаимодействия, которая подчеркивает организационную структуру между объектами, которые отправляют и получают сообщения



Типы диаграмм UML (3/3)

В языке UML существуют следующие типы диаграмм:

→ **Object diagram**

Диаграмма объектов: предназначена для демонстрации совокупности моделируемых объектов и связей между ними в фиксированный момент времени

→ **Statechart diagram**

Диаграмма состояний: позволяет описывать поведение системы (демонстрирует поведение одного объекта в течение его жизненного цикла)

⚡ **Мы сконцентрируемся на первых трех, так как именно они наиболее пригодны для описания бизнес-процессов.**



Диаграмма прецедентов



Диаграмма прецедентов



Use-case diagram (диаграмма прецедентов) – это графическое представление всех или части акторов, прецедентов и их взаимодействий в системе или бизнес-процессе

Назначение

Основное назначение диаграммы — описание функциональности и поведения, позволяющее заказчику, конечному пользователю и разработчику совместно обсуждать проектируемую или существующую систему или бизнес-процесс

При моделировании системы с помощью диаграммы прецедентов аналитик стремится:

- чётко отделить систему от её окружения
- определить действующих лиц (акторов), их взаимодействие с системой и ожидаемую функциональность системы
- определить в глоссарии предметной области понятия, относящиеся к детальному описанию функциональности системы (то есть прецедентов)



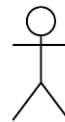
Диаграмма прецедентов

Работа над диаграммой может начинаться с текстового описания, полученного при работе с заказчиком. При этом нефункциональные требования (например, конкретный язык или система программирования) при составлении модели прецедентов опускаются (для них составляется другой документ).

Диаграмма прецедентов использует 3 основных элемента:

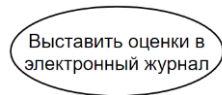
→ **Actor (участник)**

множество логически связанных ролей, исполняемых при взаимодействии с прецедентами или сущностями (система, подсистема или класс). Участником может быть человек, роль человека в системе или другая система, подсистема или класс, которые представляют нечто вне сущности.



→ **Use case (прецедент)**

описание отдельного аспекта поведения системы с точки зрения пользователя. Прецедент не показывает "как" достигается некоторый результат, а только "что" именно выполняется.



→ **System boundary (рамки системы)**

прямоугольник с названием в верхней части и эллипсами (прецедентами) внутри. Часто может быть опущен без потери полезной информации

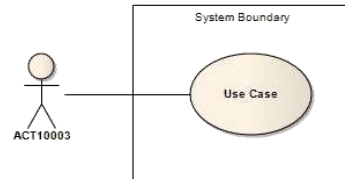




Диаграмма прецедентов: примеры

- ⚡ Рассмотрим классический студенческий пример, в котором есть 2 участника: студент и библиотекарь

Прецеденты для студента: ищет в каталоге, заказывает, работает в читальном зале

Роль библиотекаря: выдача заказа, консультации (рекомендации книг по теме, обучение использованию поисковой системы и заполнению бланков заказа)





Диаграмма прецедентов: примеры

- ⚡ **Второй пример немного сложнее:** видим, что одно и то же лицо может выступать в нескольких ролях. Например, product manager работает над стратегией и больше ничем не занимается, архитектор работает над стратегией и занимается внедрением, build master занимается тремя вещами одновременно, и т.д.

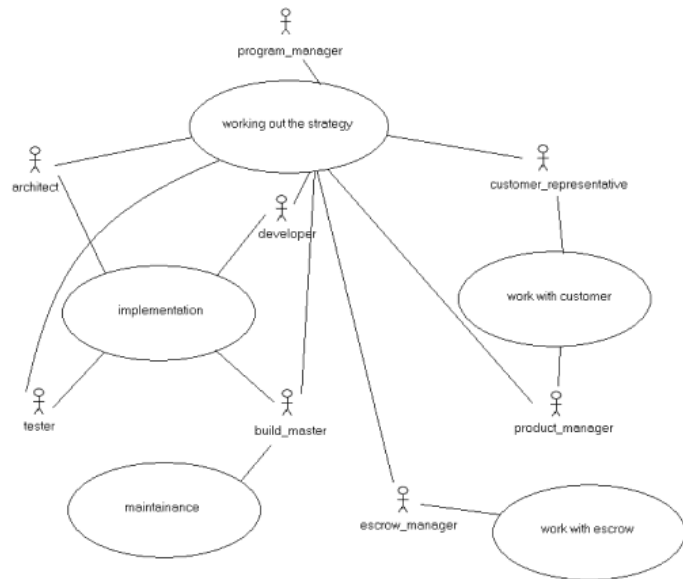




Диаграмма прецедентов: правила

При работе с вариантами использования важно помнить несколько простых правил:

- Каждый прецедент относится как минимум к одному действующему лицу
- Каждый прецедент имеет инициатора
- Каждый прецедент приводит к соответствующему результату



Диаграмма классов



Диаграмма классов

💡 **Class diagram (диаграмма классов) – это** структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей (отношений) между ними

Назначение

Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования



Диаграмма классов: примеры

- ⚡ **Класс (class) — категория вещей, которые имеют общие атрибуты и операции.** Сама диаграмма классов является набором статических, декларативных элементов модели. Она дает нам наиболее полное и развернутое представление о связях в программном коде, функциональности и информации об отдельных классах. Приложения генерируются зачастую именно с диаграммы классов.

- Для класса "студент" есть таблица, содержащая атрибуты: имя, адрес, телефон, e-mail, номер зачетки, средняя успеваемость
- Также показаны связи данной сущности с другими: прохождением курса, какой курс слушает, кто профессор
- В этом примере также добавляются функции, которые могут быть применены к сущности "студент"

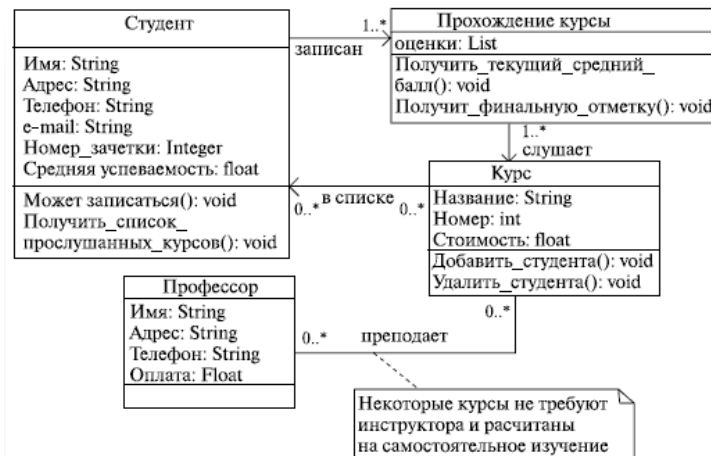




Диаграмма классов: элементы

На диаграмме классы представлены в рамках, содержащих три компонента:

- **Имя класса**
в верхней части, выравнивается по центру и пишется полужирным шрифтом, начинается с заглавной буквы
- **Поля (атрибуты) класса**
выровнены по левому краю и описывают свойства объектов класса. Большинство объектов в классе получают свою индивидуальность из-за различий в их атрибутах и взаимосвязи с другими объектами. Однако, возможны объекты с идентичными значениями атрибутов и взаимосвязей. Т.е. индивидуальность объектов определяется самим фактом их существования, а не различиями в их свойствах. Имя атрибута должно быть уникально в пределах класса. За именем атрибута может следовать его тип и значение по умолчанию.
- **Методы (операции) класса**
выровнены по левому краю.
Операция – функция (действие) или преобразование.
Операция может иметь параметры и возвращать значения.





Диаграмма классов: виды связей

→ Ассоциация (association)

представляет собой отношения между экземплярами классов

Каждый конец ассоциации обладает кратностью (синоним – мощностью, ориг. — multiplicity), которая показывает, сколько объектов, расположенных с соответствующего конца ассоциации, может участвовать в данном отношении

В примере на рисунке каждый Товар имеет сколь угодно Записей в накладной, но каждая Запись в накладной обязательно один Товар

Также **на концах ассоциации под кратностью может указываться имя роли**, т.е. какую роль выполняют объекты, находящиеся с данного конца ассоциации

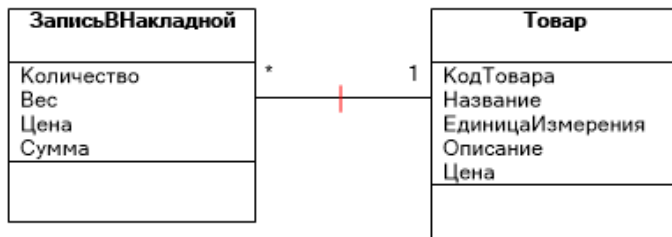




Диаграмма классов: виды связей

→ Агрегация (aggregation)

ассоциация типа «целое-часть»: в UML представляется в виде прямой с ромбом на конце

Ромб на связи указывает, какой класс является агрегирующим (т.е. «состоящим из»); класс с противоположного конца — агрегированным (т.е. те самые «части»).

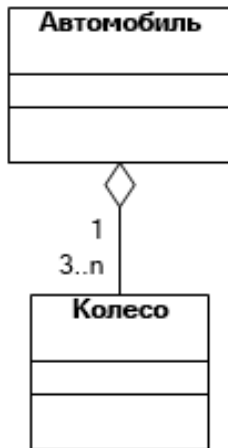




Диаграмма классов: виды связей

→ Композиция (composition)

это такая агрегация, где объекты-части не могут существовать сами по себе и уничтожаются при уничтожении объекта агрегирующего класса

Композиция изображается так же, как ассоциация, **только ромбик закрашен**

Важно понимать разницу между агрегацией и композицией: **при агрегации объекты-части могут существовать сами по себе, а при композиции — нет**

Пример агрегации: автомобиль—колесо, пример композиции: дом—комната





Диаграмма классов: виды связей

→ Наследование (inheritance)

это отношение типа «общее-частное»: позволяет определить такое отношение между классами, когда один класс обладает поведением и структурой ряда других классов

При создании производного класса на основе базового (одного или нескольких) возникает иерархия наследования

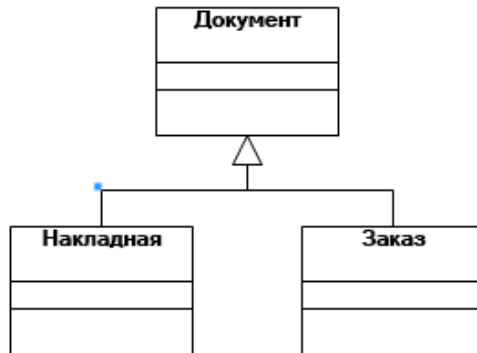




Диаграмма классов: пример построения

- Основной сущностью в системе будет являться **товар**, который хранится на складе
- Понятия товара как некоего описания и товара, лежащего непосредственно на складе, **отличаются друг от друга**: товар, лежащий на складе, кроме того, что связан со складом отношением композиции (агрегация не совсем подходит, поскольку в данной системе товар является товаром, пока он не покинет склад), ещё характеризуется количеством.
- Аналогично следует рассуждать и при рассмотрении отношения Товара и Заказа, Товара и Накладной
- В связи с тем, что Заказ и Накладная в сущности являются документами и имеют сходные атрибуты, они были объединены с помощью общего класса-предка Документ

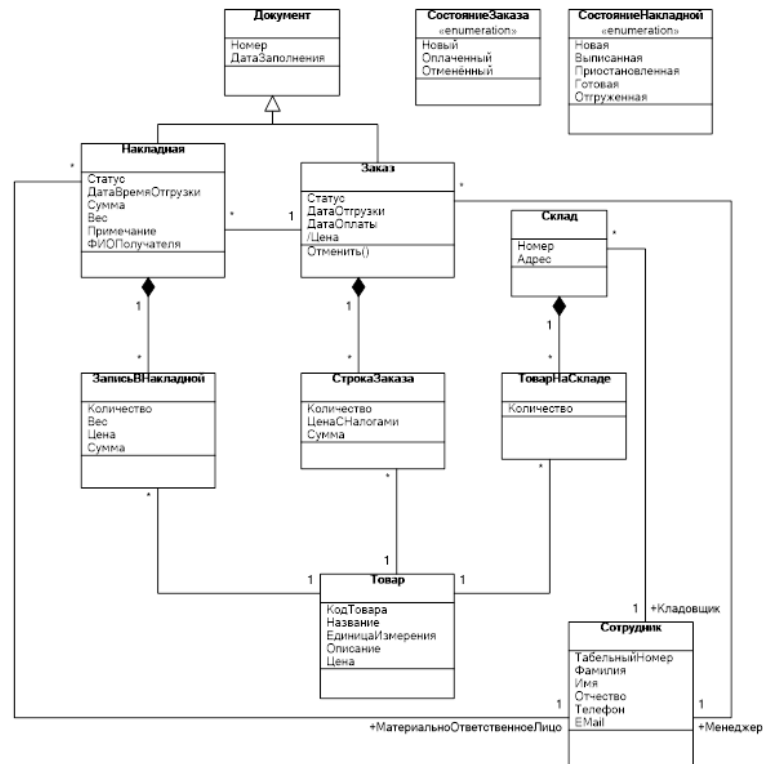




Диаграмма активностей



Диаграмма активностей

💡 **Activity diagram (диаграмма активностей)** – позволяет более детально визуализировать конкретный случай использования. Это поведенческая диаграмма, которая иллюстрирует поток деятельности (бизнес-процесс) через систему.

Назначение

Диаграмма активностей описывает динамические аспекты поведения системы в виде блок-схемы, которая отражает бизнес-процессы, логику процедур и потоки работ — переходы от одной деятельности к другой. По сути, мы рисуем алгоритм действий (логику поведения) системы или взаимодействия нескольких систем



Диаграмма активностей: пример

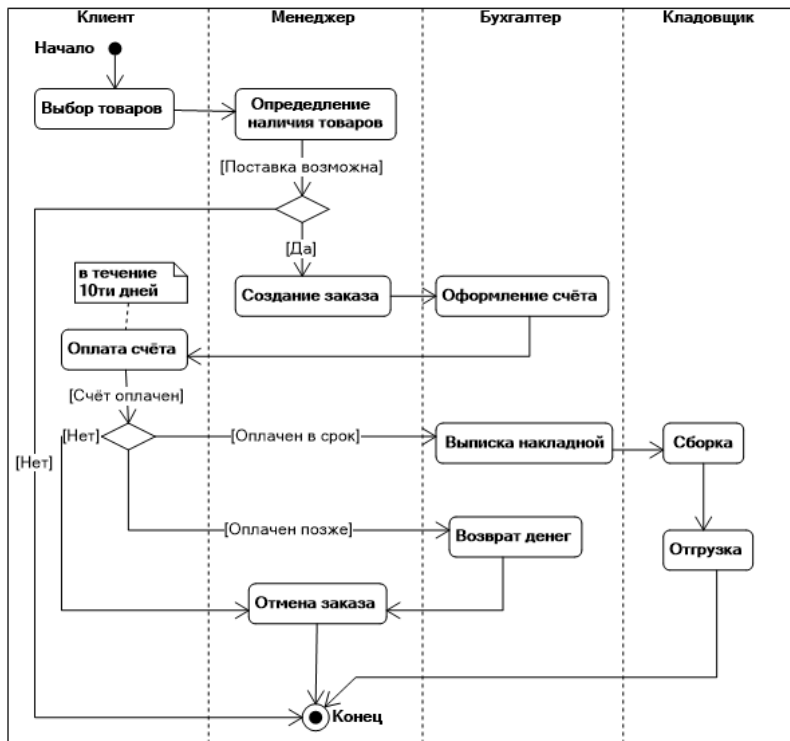












Диаграмма активностей: элементы

Символ	Имя	Использовать
	Пуск/ начальный узел	Используется для представления отправной точки или начального состояния деятельности
	Действие / Состояние действия	Используется для представления деятельности процесса
	Действие	Используется для представления исполняемых подрайонов деятельности
	Поток управления / Край	Используется для представления потока управления от одного действия к другому
	Поток объекта / края управления	Используется для отображения пути движения объектов по активности
	Конечный узел активности	Используется для обозначения конца всех контрольных потоков в рамках деятельности
	Поток конечный узел	Используется для обозначения конца одного потока управления
	Узел принятия решений	Используется для представления условной точки ответвления с одним входом и несколькими выходами







	Узел слияния	Используется для представления слияния потоков. Он имеет несколько входов, но один выход.
	Вилка	Используется для представления потока, который может разветвляться на два и более параллельных потока
	Слияние	Используется для представления двух входов, которые объединяются в один выход
	Отправка сигнала	Используется для представления действия по отправке сигнала на приемную деятельность
	Получение сигнала	Используется для обозначения того, что сигнал получен
	Примечание/ комментарий	Используется для добавления соответствующих комментариев к элементам



Диаграмма активностей: Swimlanes

- ⚡ В Диаграммы активностей Swimlanes (разделы) используются для представления или группирования действий, выполняемых различными действующими лицами в одном потоке

Несколько советов, по использованию Swimlanes:

- Добавить Swimlanes линейных процессов – это упростит чтение схемы
- Не добавлять более 5 Swimlanes
- Располагать Swimlanes в логическом порядке

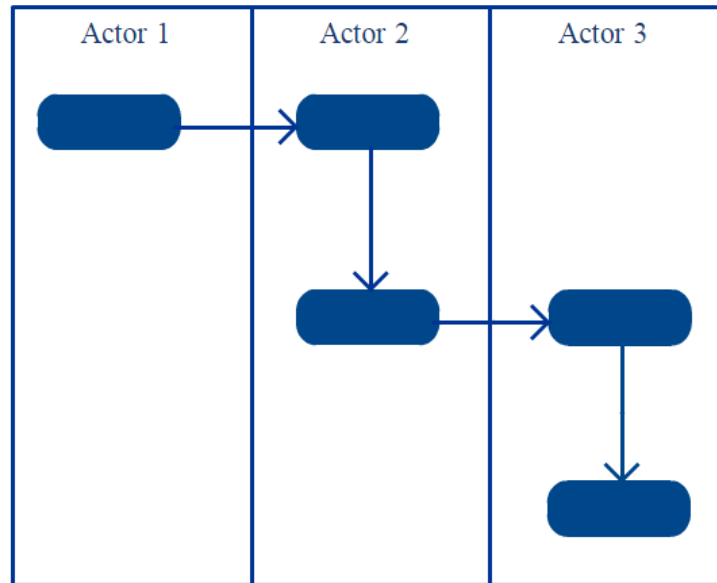
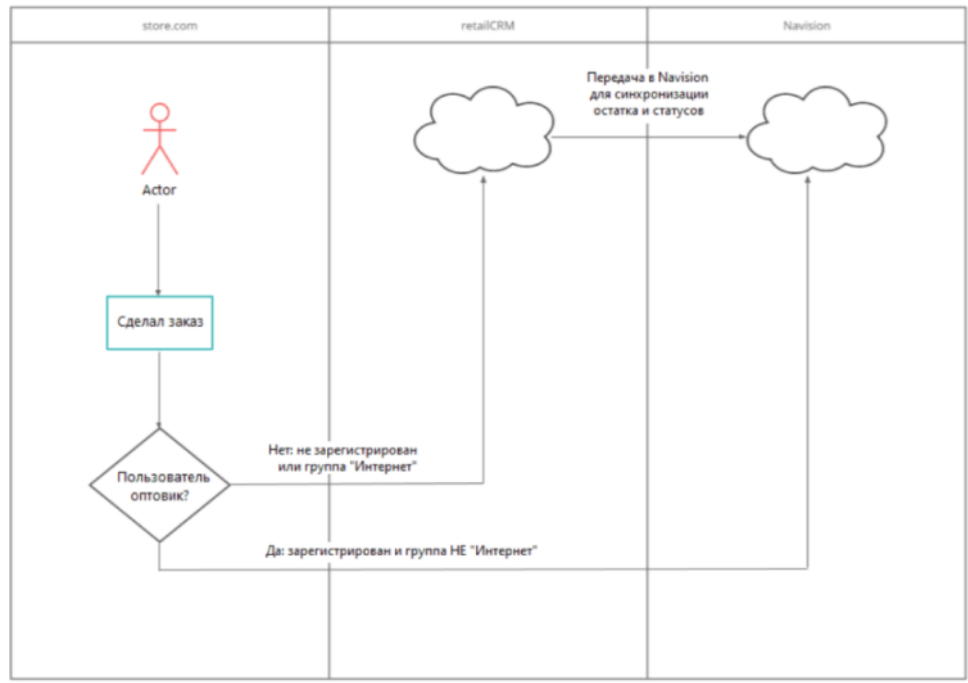




Диаграмма активностей: пример для интернет-магазина

Диаграмма активностей для сайта магазина максимально доступно объясняет, какие есть интеграции в системе:

- Актор (в нашем случае — покупатель), зашедший на сайт, делает заказ
- Далее у нас происходит разветвление: проверяем, является ли пользователь оптовиком
- Если он не зарегистрирован в системе и не оптовик, заказ отправляется в retailCRM, если пользователь зарегистрирован, его заказ попадает в Navision. При этом между retailCRM и Navision происходит синхронизация остатка и статусов.





Плюсы и минусы UML



«+» Плюсы UML

- **С помощью UML можно описать ситуацию или ключевую задачу с различных точек зрения и аспектов поведения системы**
- **UML диаграммы простые в восприятии:** прочитать схему и ознакомиться с ее синтаксисом сможет даже работник, не имеющих специальных знаний в области программирования и написании бизнес-моделей (речь идет о стандартных схемах с 20-40 условными обозначениями)
- **UML минимизирует процент возможных ошибок при создании бизнес-процесса.** Например, она исключает несогласованность параметров дополнительных программ или изменение основных атрибутов.
- **Любой этап бизнес-процесса может быть использован повторно** в уже существующем или новом проекте организации

«-» Минусы UML

- Многие программисты используют современную версию нотации UML 2.0, **которая характеризуется высокой избыточностью языка**, содержит множество диаграмм и конструкций, которые не всегда важны при создании модели бизнес-процесса.
- **Применение объектно-ориентированного подхода требует наличия знаний о предметной области и методах анализа на языке программирования.** Крупные проекты могут включать свыше 100 условных обозначений. В таком случае в команде должны быть специалисты, владеющие определенным уровнем квалификации и умеющие отходить от традиционных подходов к работе.

Нотация UML



🔍 Необходимо описать бизнес-процесс используя нотацию UML (1 процесс на выбор):

- Отправка посылки Почтой России
- Поиск работы
- *Свой вариант*





Итоги урока

-  Подробно познакомились с нотацией UML;
-  Узнали, как применять данную нотацию, какие у нее ключевые элементы и важные правила использования.



Вопросы?

Вопросы?



Вопросы?





На следующем уроке

- 📌 Узнаем, как находить зоны для развития в процессе;
- 📌 Поговорим про то, какие у процесса есть критерии эффективности.



Спасибо за внимание!