Курганский государственный университет
Кафедра программного обеспечения автоматизированных систем

К У Р С Л Е К Ц И Й

по дисциплине

ВВЕДЕНИЕ В ПРОГРАММНУЮ ИНЖЕНЕРИЮ

для студентов направления 231000.62

«Программная инженерия»

Лекция 3.3. Разработка функциональной структуры программной системы.

U.M.L.-диаграммы вариантов использования

План лекции

Введение
1. Структурные модели анализа бизнес-процессов
    1.1. Схема Захмана
    1.2. Диаграммы потоков данных
2. UML-диаграммы вариантов использования
    2.1. Назначение и компоненты UseCase-диаграммы
    2.2. Компонент диаграммы "***Актер***"
    2.3. Компонент диаграммы "***Вариант использования***"
    2.4. Компонент диаграммы "***Интерфейс***"
    2.5. Компонент диаграммы "***Связь***"("Отношение")
        2.5.1. Отношение ассоциации
        2.5.2. Отношение расширения
        2.5.3. Отношение обобщения
        2.5.4. Отношение включения
    2.6. Примеры UseCase-диаграмм
    2.7. Обозначения компонентов UseCase-диаграмм
    2.8. Сценарии вариантов использования
    2.9. Рекомендации и типичные ошибки при разработке UseCase-диаграмм
3. Выявление и анализ требований
Контрольные вопросы и задания

Введение

Разработка требований –обязательная начальная стадия программного проекта (как, впрочем, и любого другого).

Требования к программной системе определяют, какие свойства и характеристики она должна иметь для удовлетворения потребностей пользователей и других заинтересованных лиц.

Для выявления этих потребностей проводится ***анализ предметной области*** или ***моделирование бизнес-процессов,*** в результате которого разработчики должны:

- научиться понимать язык, на котором говорят пользователи;
- выявить цели деятельности пользователей;
- определить набор задач, решаемых пользователями.
- определить набор сущностей, с которыми приходится иметь дело при разработке системы;
- выяснить свойства результатов, которые хотелось бы получить.

Анализом предметной области занимаются ***системные аналитики*** или ***бизнес-аналитики***, которые передают полученные ими знания другим членам команды проекта, сформулировав их на более понятном разработчикам языке.

Для передачи этих знаний обычно служит некоторый набор моделей, оформляемых в виде графических схем и текстовых документов.

В данной лекции будут рассмотрены способы графического представления моделей бизнес-процессов предметной области, разработанные в рамках методологий структурного и объектно-ориентированного анализа.

Структурные модели анализа бизнес-процессов
Схема ***Захмана***

Для систематизации сбора информации о больших организациях и дальнейшей разработки систем, поддерживающих их деятельность, применяется *архитектурная схема предприятия (enterprise architecture framework) - схема Захмана.*

В основе схемы Захмана лежит следующая идея: деятельность даже очень большой организации можно описать, используя ответы на простые вопросы:

- ***Зачем ?***
- ***Кто ?***
- ***Что ?***
- ***Как ?***
- ***Где ?***
- ***Когда?***

Ответы на эти шесть вопросов следует получить, анализируя деятельность предприятия на шести разных уровнях рассмотрения.

Результаты анализа представляются в матричной форме.

Схема Захмана

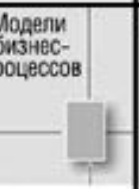
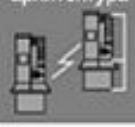
	Мотивация	Люди	Данные	Функции	Место	Время
Контекст	Цели и стратегия бизнеса 	Важные для бизнеса организации 	Вещи, значимые для бизнеса 	Основные бизнес-процессы 	География бизнеса 	События и периоды, важные для бизнеса 
Модель бизнеса	Бизнес-план, частные цели и стратегии 	Модели потоков работ 	Семантические модели  Бизнес-сущности и их связи	Модели бизнес-процессов 	Система логистики 	Базовый график работ 
Системная модель	Модель бизнес-правил 	Архитектура пользовательского интерфейса 	Концептуальная модель данных 	Архитектура приложений 	Архитектура распределенной системы 	Структура обработки событий 
Технологическая модель	Модель правил обработки событий 	Архитектура представления 	Физическая модель данных 	Архитектура программно-аппаратной системы 	Технологическая архитектура 	Структура циклов управления 
Детальное представление	Спецификации правил работы системы 	Спецификации ролей и прав доступа 	Спецификации форматов данных 	Код программных компонентов 	Спецификации архитектуры сети 	Спецификации обработки событий и прерываний 
Работающая организация	Стратегия и тактика	Структура организации	Данные	Выполняемые функции	Географическое расположение и сети	Планы

Схема Захмана

Столбцы матрицы представляют разные аспекты рассмотрения (ответы на шесть перечисленных выше вопросов):

- ***Цели*** организации ***и базовые правила***, по которым она работает (*Зачем ?*).
- ***Персонал***, подразделения и другие элементы организационной структуры, связи между ними (*Кто ?*).
- ***Сущности и данные***, с которыми имеет дело организация (*Что ?*).
- ***Функции и операции*** над данными, выполняемые организацией и различными ее подразделениями (*Как?*).
- ***Географическое распределение*** элементов организации и связи между ее частями (*Где ?*).
- ***Временные характеристики*** и ограничения на деятельность организации, значимые для ее деятельности события (*Когда ?*).

Схема Захмана

Строки матрицы – это различные уровни рассмотрения, из которых при бизнес-моделировании особенно важны три верхних уровня:

- Самый крупный — *уровень организации в целом*, рассматриваемой в ее развитии совместно с окружением, уровень общего планирования ее деятельности. Этот уровень содержит долговременные цели и задачи организации как цельной системы, основные связи организации с внешним миром и основные виды ее деятельности.
- *Уровень бизнеса*, на котором рассматривается внутренняя структура организации во всех аспектах ее деятельности в соответствии с ее основными задачами.
- *Системный уровень*, на котором определяются концептуальные модели всех аспектов организации без привязки к конкретным их воплощениям и реализациям: например, логическая модель данных в виде набора сущностей и связей между ними, логическая архитектура системы автоматизации в виде набора узлов с привязанными к ним функциями и пр.

Диаграммы потоков данных

Диаграммы потоков данных *(**D**ata **F**low **D**iagrams)* уже рассматривались на одной из предыдущих лекций в качестве иллюстрации средств визуального моделирования, разработанных в рамках методологии структурного анализа систем.

DFD-диаграммы использовались (и используются в настоящее время) для графического представления поведения сложных систем и деятельности крупных организаций.

DFD-диаграмма содержат 4 вида графических элементов:

- **процессы**, представляющие собой любые трансформации данных в рамках описываемой системы;
- **хранилища данных;**
- **сущности,** внешние по отношению к системе;
- **потоки данных** между элементами трех предыдущих видов.

Используются несколько систем обозначений для элементов *DFD-диаграмм.*

На следующих слайдах представлены две наиболее известные нотации: нотация *Йордана-ДеМарко* и нотация *Гэйна-Сарсона*, предложенные в 1979 году.

Диаграмма потоков данных в нотации *Йордана-ДеМарко*

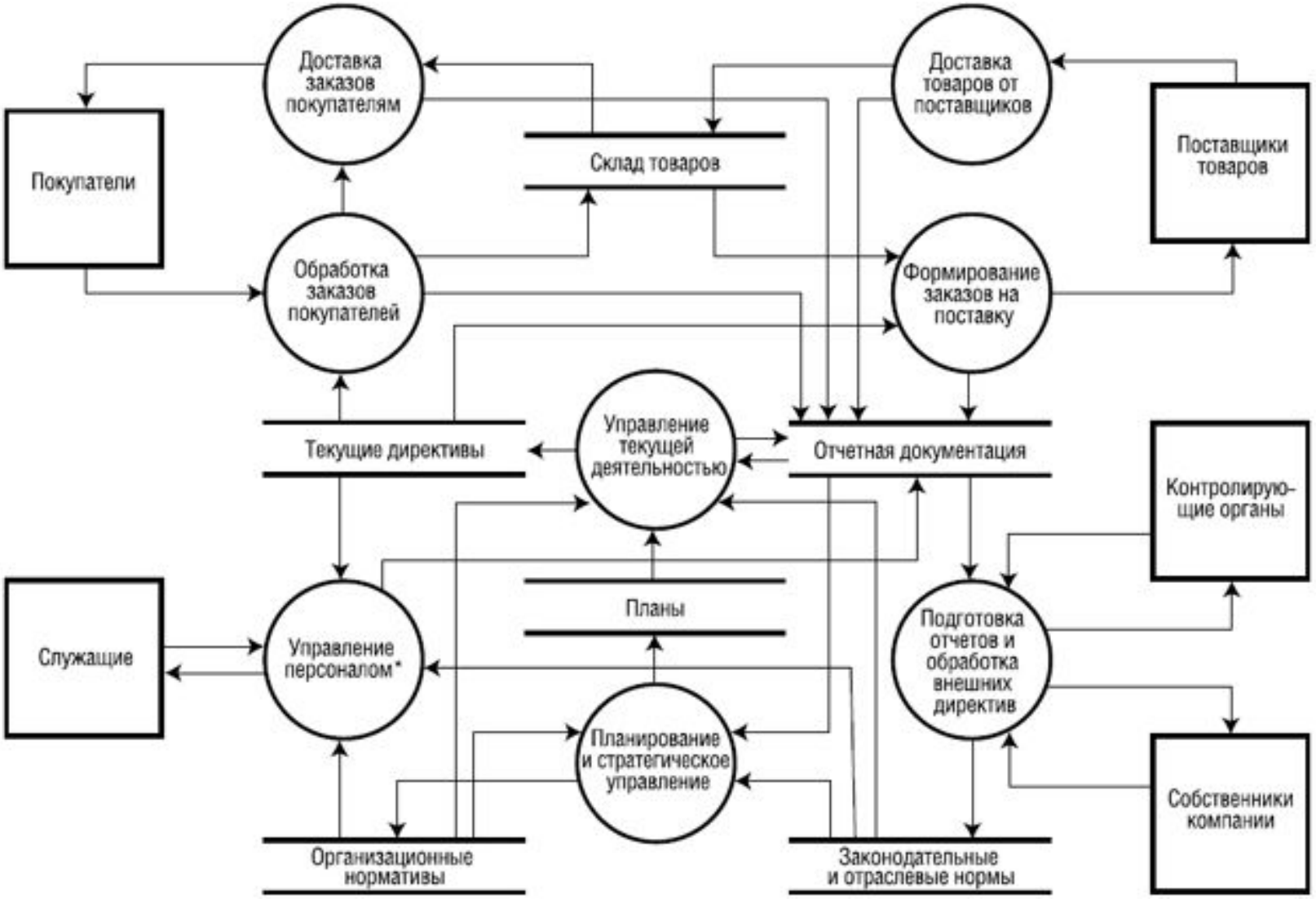


Диаграмма потоков данных в нотации *Гэйна-Сарсона*

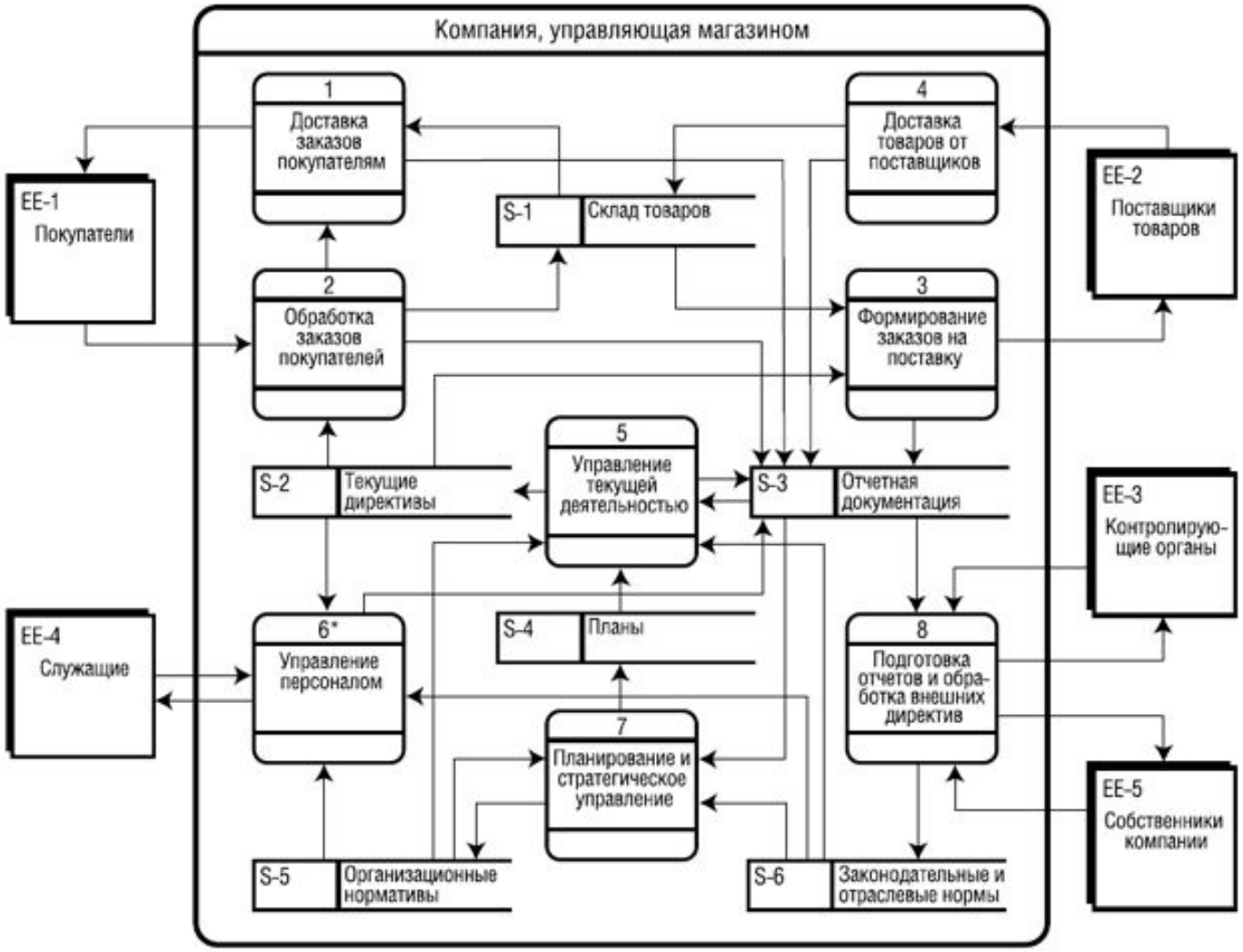
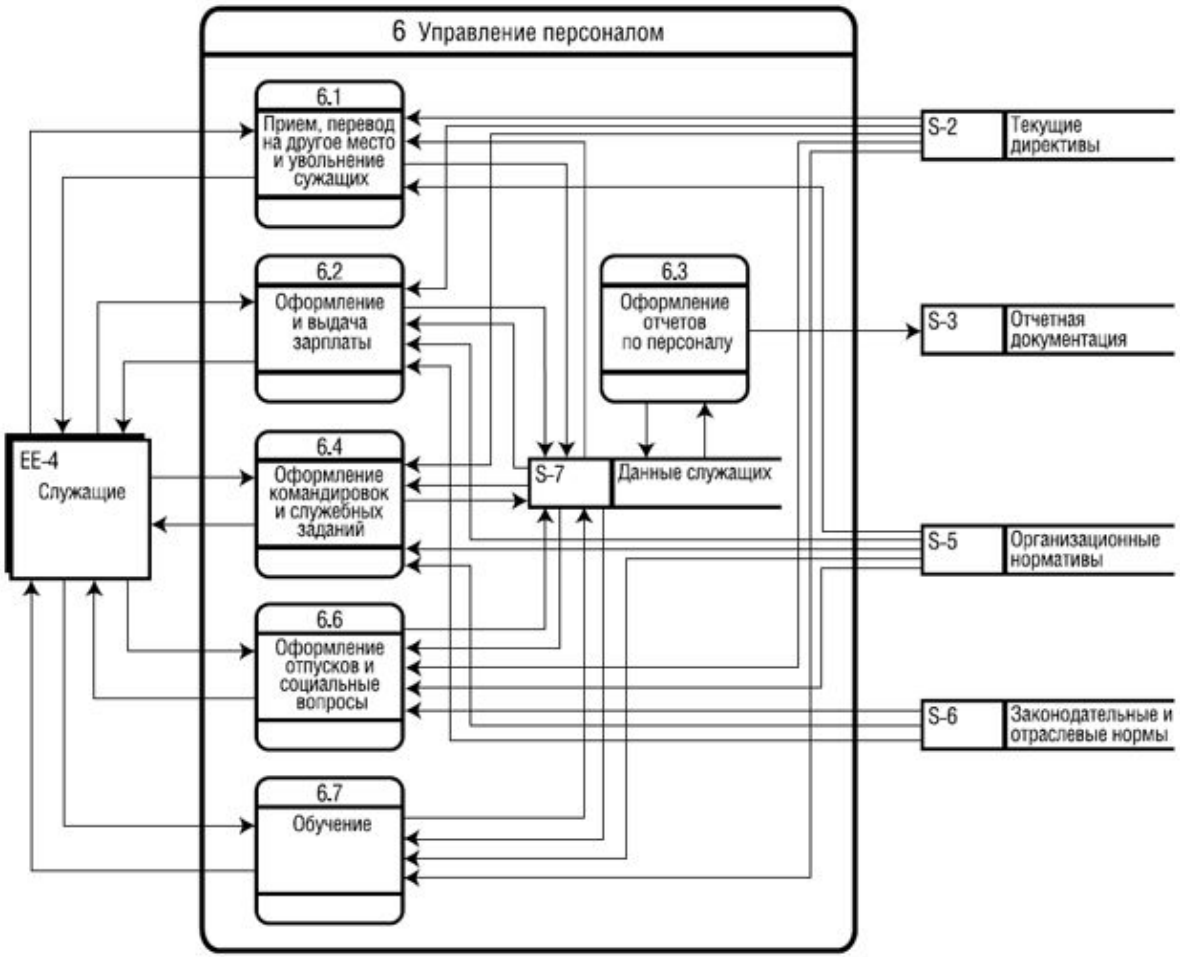


Диаграмма потоков данных в нотации *Гэйна-Сарсона*

Процессы на DFD-диаграммах могут уточняться: для каждого их них может быть разработана отдельная диаграмма, описывающая потоки данных внутри этого процесса. На рисунке показана детализация процесса №6 "*Управление персоналом*"



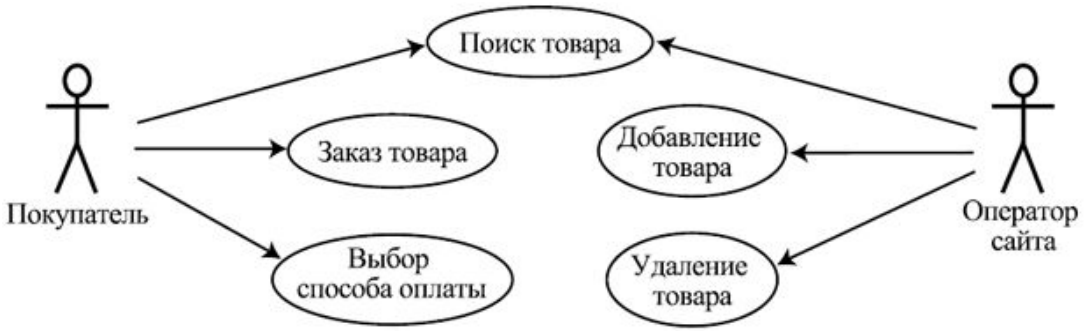
Язык U.M.L. Диаграммы вариантов использования

Диаграмма ***вариантов использования*** (***UseCase*** *diagram* – в русскоязычных источниках называется иногда *диаграммой* ***прецедентов***) – это форма представления концептуальной модели проектируемой системы, которая описывает ее функциональное назначение и в дальнейшем будет детализирована и преобразована в модели логического и физического уровней.

Разработка ***UseCase-***диаграммы преследует следующие цели:

- Определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы.
- Сформулировать общие требования к функциональному поведению системы.
- Разработать исходную концептуальную модель системы для ее последующей детализации в форме моделей логического и физического уровней.
- Подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Проектируемая система представляется на UseCase-диаграмме в виде множества ***актеров***, взаимодействующих с системой с помощью ***вариантов использования***.



Компоненты UseCase-диаграммы

Диаграмма строится базе компонентов следующих типов :

- ***действующие лица*** или ***актеры*** *(actors)* - любые сущности, взаимодействующие с системой извне. Это может быть человек, техническое устройство, программа или любая другая система, которая может служить источником воздействия на моделируемую систему или потребителем информации, генерируемой системой.
- ***варианты использования*** *(use case) - служат для описания сервисов, которые система предоставляет актерам. Каждый вариант использования определяет некоторый набор действий, выполняемых системой при взаимодействии с актером, причем детали реализации такого взаимодействия не раскрываются.*
- ***связи*** или ***отношения*** *(relationships)* - используются для обозначения отношений между компонентами модели.
- ***интерфейсы*** *(interface) - определяют совокупность операций, обеспечивающих выполнение вариантов использования.*
- ***примечания*** (notes) - произвольные текстовые комментарии разработчика, имеющие отношение к компонентам UseCase-диаграммы.

Компоненты UseCase-диаграммы: ***Актеры***

Актер - это сущность, взаимодействующая с системой извне.

Актеры взаимодействует с вариантами использования посредством *ассоциативных связей,* передавая вариантам использования запросы и получая от них соответствующие сервисы.

Кроме этого, с актерами могут быть связаны интерфейсы, которые определяют, каким образом другие элементы модели взаимодействуют с этими актерами.

Актер представляет определенную ***роль*** (группу) пользователей системы или внешних систем, взаимодействующих с проектируемой системой.

Два и более актера могут иметь общие свойства, т. е. одинаково взаимодействовать с одним и тем же множеством вариантов использования. Такая общность свойств и поведения представляется в виде *отношения обобщения* с другим, возможно, абстрактным актером, который моделирует соответствующую общность ролей.

Реально с системой будут взаимодействовать отдельные субъекты или объекты - *экземпляры актеров*, каждый из которых может принадлежать к одной или нескольким ролям.

Актер – это именованный элемент модели, имя актера должно быть кратким и информативным.

Поскольку актер является внешней сущностью по отношению к моделируемой системе, его внутренняя структура в контексте данной диаграммы никак не определяется.

Актер представляется на диаграмме вершиной графа и изображается в виде схематичного человечка, помеченного соответствующим именем

Компоненты UseCase-диаграммы: ***Варианты использования***

Вариант использования – главный компонент UseCase-диаграммы, он служит для описания функционального поведения системы - сервисов, которые система предоставляет актерам.

Каждый вариант использования определяет некоторый сценарий - набор действий, совершаемый системой при взаимодействии с актером, причем детали реализации такого взаимодействия в данной диаграмме не раскрываются.

Предполагается, что вариант использования инициализируется по запросу (сигналу), поступающему от актера, и должен представлять собой законченную последовательность действий. После того, как эти действия будут выполнены, сервис должен прийти в исходное состояние и должен быть готовым к выполнению следующего запроса.

Вариант использования представляется на UseCase-диаграмме вершиной графа и изображается в виде овала, внутри которого записывается имя варианта.

Варианты использования могут быть связаны с одним или несколькими актерами, другими вариантами использования и интерфейсами.

Атрибуты варианта использования

При разработке UseCase-диаграммы для каждого варианта использования должны быть определены следующие атрибуты:

- ***Имя***, ясно говорящее о назначении *варианта использования*.
- ***Описание*** - несколько предложений, описывающих этот *вариант использования*.
- ***Частота*** – показывает, как часто возникает данный *вариант использования*.
- ***Предусловия*** - все условия запуска *варианта использования*.
- ***Постусловия*** - все условия, которые должны быть выполнены после успешного выполнения *варианта использования*.
- ***Основной сценарий*** работы, который используется в большинстве случаев.
- ***Альтернативные сценарии***, используемые иногда: для каждого альтернативного сценария указываются условия его запуска.
- (Необязательно) ***Задействованные актеры***.
- (Необязательно) ***Расширяемые варианты использования***.
- (Необязательно) ***Включаемые варианты использования***.
- (Необязательно) ***Статус***: "в разработке", "готов к проверке", "в процессе проверки", "подтвержден", "отвергнут".
- (Необязательно) ***Допущения об окружении*** и ***ходе работы системы***, использованные при разработке данного варианта.

Компоненты UseCase-диаграммы: ***Интерфейс***

Интерфейсы (interface) в UseCase-диаграммах *определяют совокупность операций, обеспечивающих выполнение сценариев вариантов использования*.

Интерфейсы UseCase-диаграмм не могут содержать ни атрибутов, ни состояний, ни направленных ассоциаций - они содержат только операции без указания особенностей их реализации.

Интерфейс представляется на UseCase-диаграмме вершиной графа и изображается в виде маленького круга, рядом с которым записывается имя интерфейса. В качестве имени может быть существительное или короткий текст, характеризующие соответствующую информацию или сервис: например, «датчик», «видеокамера», «запрос к базе данных», «форма ввода», «устройство подачи звукового сигнала».

Графический символ отдельного интерфейса может соединяться на диаграмме ***сплошной линией*** с тем вариантом использования, который его поддерживает (рисунок а). Сплошная линия в этом случае указывает на тот факт, что связанный с интерфейсом вариант использования должен реализовывать все операции, необходимые для данного интерфейса, а возможно и больше.

Если интерфейс соединен с вариантом использования ***пунктирной линией*** со стрелкой (рисунок б), то это означает, что вариант использования предназначен для спецификации только того сервиса, который необходим для реализации данного интерфейса.

Компоненты UseCase-диаграммы: ***Связь***

Связь используется в UseCase-диаграммах для обозначения различных отношений между компонентами модели.

Связи представляются на UseCase-диаграмме дугами графа и изображаются линиями со стрелками определенного вида, которые могут попарно соединять другие компоненты диаграммы (актеров, варианты использования и интерфейсы) в различных комбинациях.

Связь на UseCase-диаграмме может принадлежать к одному из четырех типов отношений, устанавливаемых между парой компонентов модели:

- Отношение ***ассоциации*** (association relationship)
- Отношение ***расширения*** (extend relationship)
- Отношение ***обобщения*** (generalization relationship)
- Отношение ***включени***я (include relationship).

Типы связей: отношение ***ассоциации***

Отношение ассоциации является одним из фундаментальных понятий в языке UML и используется при построении многих диаграмм.

Применительно к диаграммам вариантов использования ***ассоциативная связь специфицирует особенности взаимодействия актера и варианта использования***.

Ассоциативная связь обозначается на диаграмме ***сплошной линией***, соединяющей актера с вариантом использования, которая может иметь дополнительные условные обозначения:

- ***Имя связи***, указывающее на конкретную роль актера при его взаимодействии с вариантом использования
- ***Кратность связи***, которая определяет количество экземпляров актера и варианта использования, участвующих в связи.
- ***Стрелка***, всегда направленная от актера к варианту использования, что отображает тот факт, что сервис будет активизирован по запросу актера (так как стрелка всегда направлена однозначно, ее часто не показывают).

Типы связей: отношение ***ассоциации***

Кратность (multiplicity) ***ассоциативной связи*** указывается рядом с обозначениями связанных компонентов диаграммы, и характеризует общее количество экземпляров соответствующего компонента, которые могут выступать в качестве элементов данной ассоциации.

Применительно к UseCase-диаграммам для указания кратности связи используются следующие обозначения:

- ***Целое неотрицательное число***: указывает на кратность, строго равную указанному числу. Например, кратность "1" для актера "Клиент банка" на предыдущем слайде означает, что конкретный кредит не может быть выдан нескольким или неопределенному числу клиентов банка.
- ***Два целых неотрицательных числа, разделенные двумя точками***: обозначает интервал целых чисел, следующих в последовательно возрастающем порядке. Если бы, например, на предыдущем слайде правый конец связи был помечен как "***0..5***", то это означало бы, что клиент банка не может оформить более пяти кредитов.
- "***N..*****"**: обозначает диапазон числовых значений, ограниченный только слева значением ***N*** (символ **"*"** обозначает произвольное конечное целое неотрицательное число, значение которого неизвестно на момент задания соответствующего отношения ассоциации).
- ***Единственный символ*** **"*"** является сокращением записи интервала "***0..****", что означает "***любое целое неотрицательное число***".

Если кратность отношения ассоциации на диаграмме не указана, то по умолчанию принимается ее значение, равное 1.

Типы связей: отношение ***расширения***

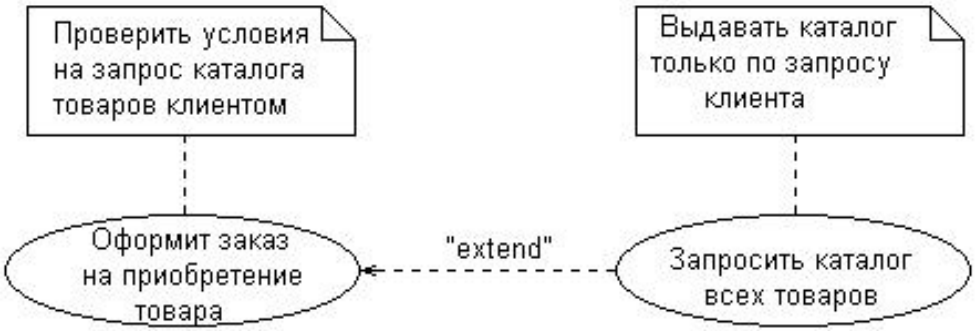
Отношение расширения определяет взаимосвязь между более общим *базовым* вариантом использования и некоторым вариантом использования, *расширяющим* базовый вариант в определенных (исключительных) ситуациях.

Отношение расширения отображает тот факт, что базовый вариант использования может присоединять к своему поведению некоторое дополнительное поведение, определенное для другого варианта использования.

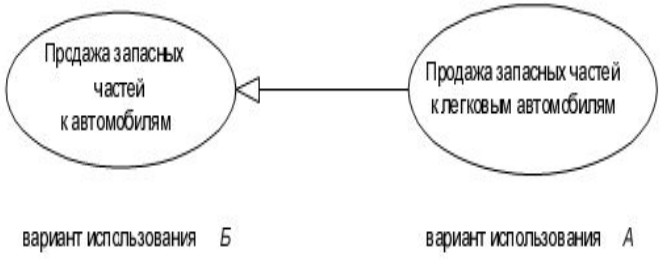
Отношение расширения является направленным и обозначается *пунктирной линией со стрелкой*, направленной от *расширяющего варианта* использования к базовому варианту и помеченной ключевым словом "extend".

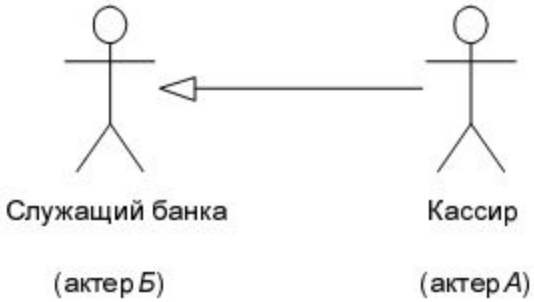
Данное отношение включает в себя некоторое условие, проверяемое в базовом варианте, и указатели на точки расширения в базовом варианте, в которые должно быть помещено соответствующее расширение при выполнении условия (например, условием расширения является запрос от клиента на получение каталога товаров).

Один вариант использования может быть расширением для нескольких базовых вариантов, а также иметь в качестве собственных расширений несколько других вариантов.



Типы связей: отношение ***обобщения***





Отношение обобщения – это связь типа "предок – потомок", которая служит для указания того факта, что *потомок* является специальным случаем (специализацией) своего *предка*.

Потомок наследует все свойства и поведение своего *предка* и участвуют во всех его отношениях.

Потомки могут уточнять или модифицировать наследуемые от своих предков свойства поведения, а также могут наделяться новыми свойствами поведения и могут участвовать в связях, отсутствующих у своих предков.

Отношение обобщения обозначается на UseCase-диаграммах сплошной линией со ***стрелкой в виде незакрашенного треугольника***, направленной от "потомка" к "предку" .

У каждого *предка* может быть несколько *потомков*, однако ***отношение обобщения не является строго иерархическим*** – это означает, что один *потомок* может иметь несколько *предков*, и в этом случае реализуется *множественное наследование потомком свойств и поведения всех его предков*.

Типы связей: отношение ***включения***

Отношение включения устанавливается только между вариантами использования и является *направленным бинарным отношением* в том смысле, что некоторое функциональное поведение, заданное для одного (включаемого) варианта использования, безусловно включается в качестве составного компонента в поведение другого (базового) варианта использования.

Отношение включения используется в тех случаях, когда в нескольких различных вариантах использования обнаруживаются похожие последовательности действий, которые и выделяются в отдельные варианты использования, включаемые в несколько базовых вариантов.

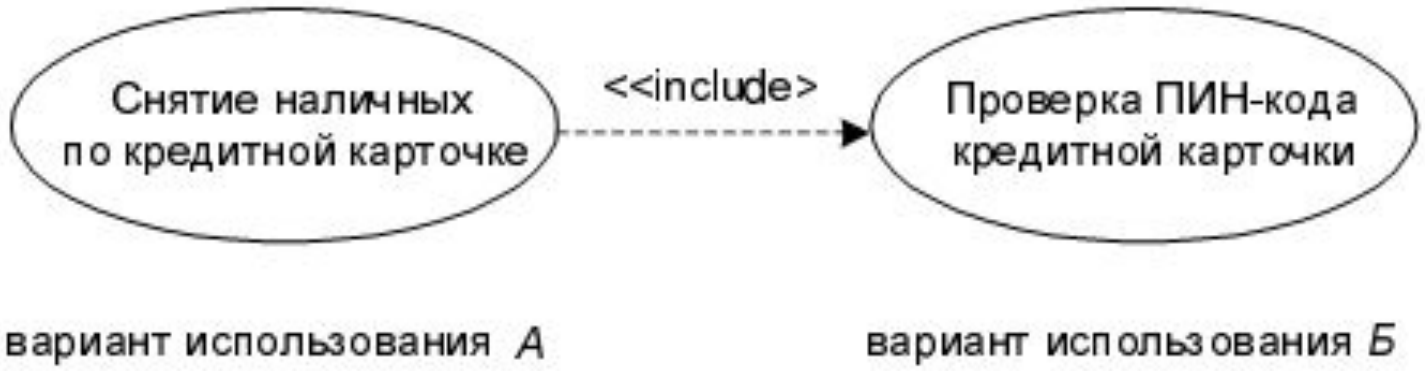
Включаемый вариант использования всегда выполняется по инициативе базового варианта.

Один вариант использования может быть включен в несколько базовых вариантов, а также сам выступать в роли базового варианта по отношению к другим, включаемым в него вариантам.

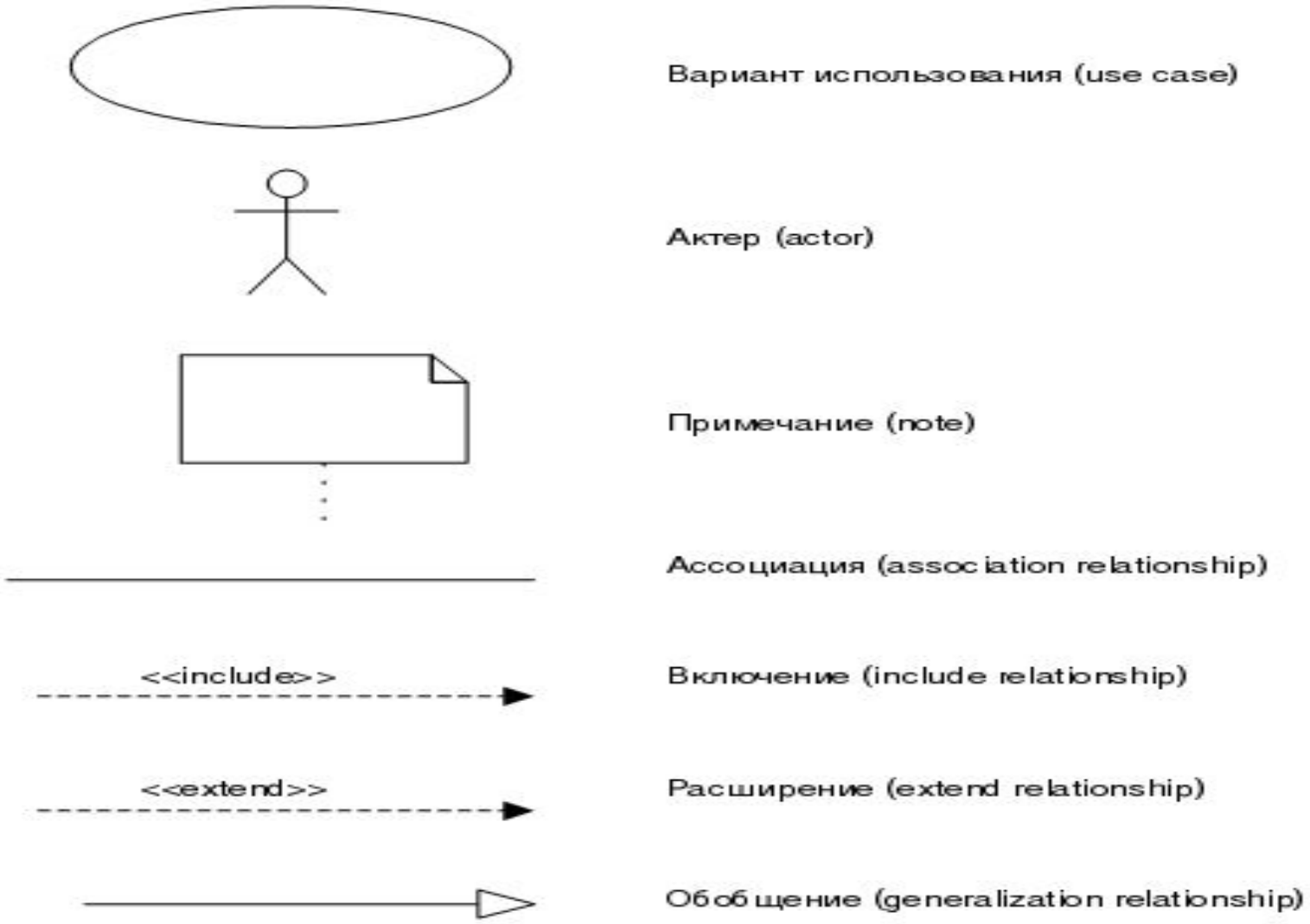
Включаемый вариант использования независим от базового в том смысле, что он предоставляет базовому варианту некоторое инкапсулированное поведение, детали реализации которого скрыты от базового варианта. При этом базовый вариант зависит только от результатов выполнения включаемого в него варианта, но не от его структуры и способа реализации.

Типы связей: отношение ***включения***

Графически отношение включения обозначается ***пунктирной линией со стрелкой***, направленной от базового варианта использования к включаемому и помеченной специальным *стереотипом* - ключевым словом "*include*"

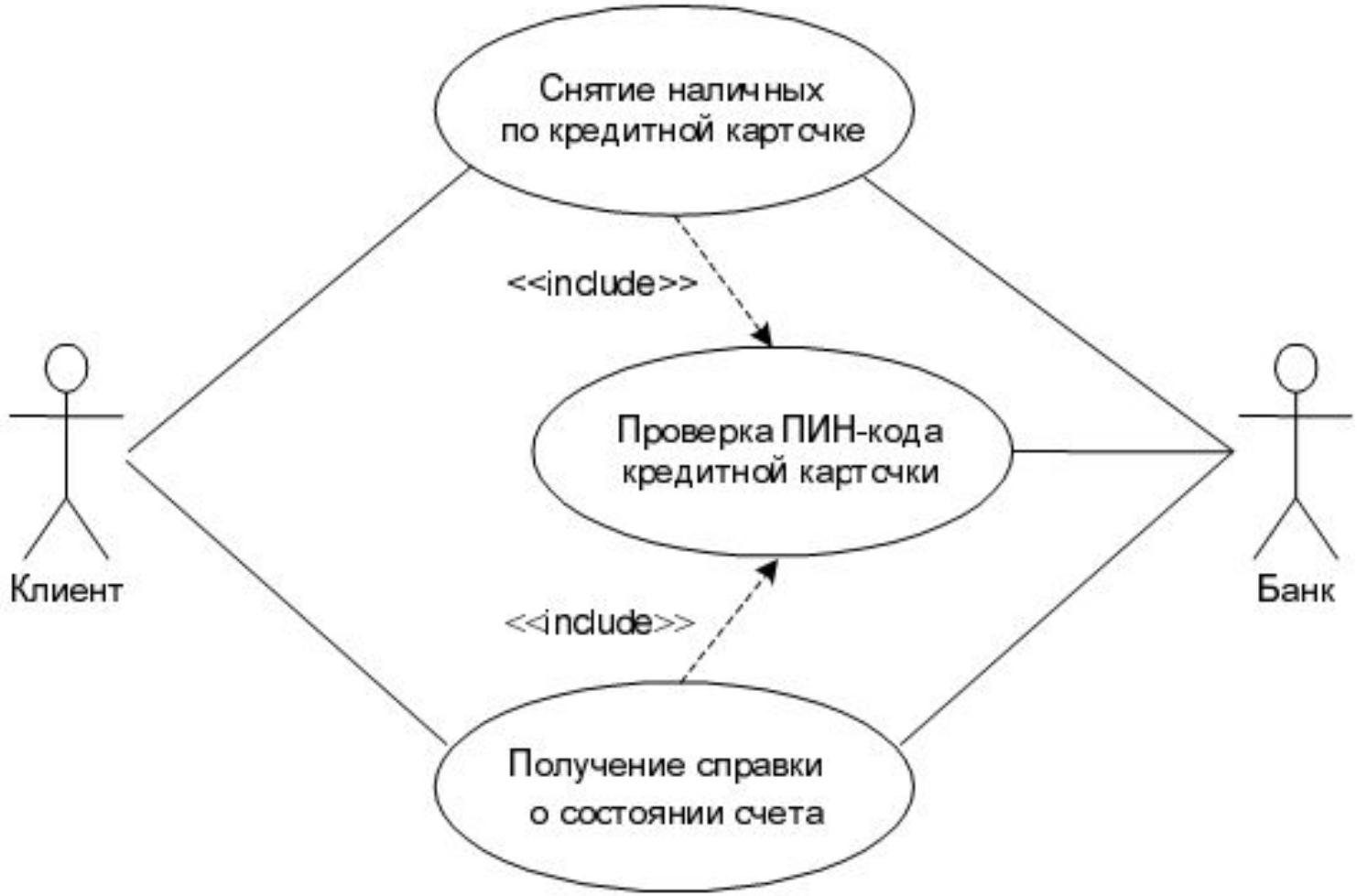


Обозначения компонентов UseCase-диаграммы

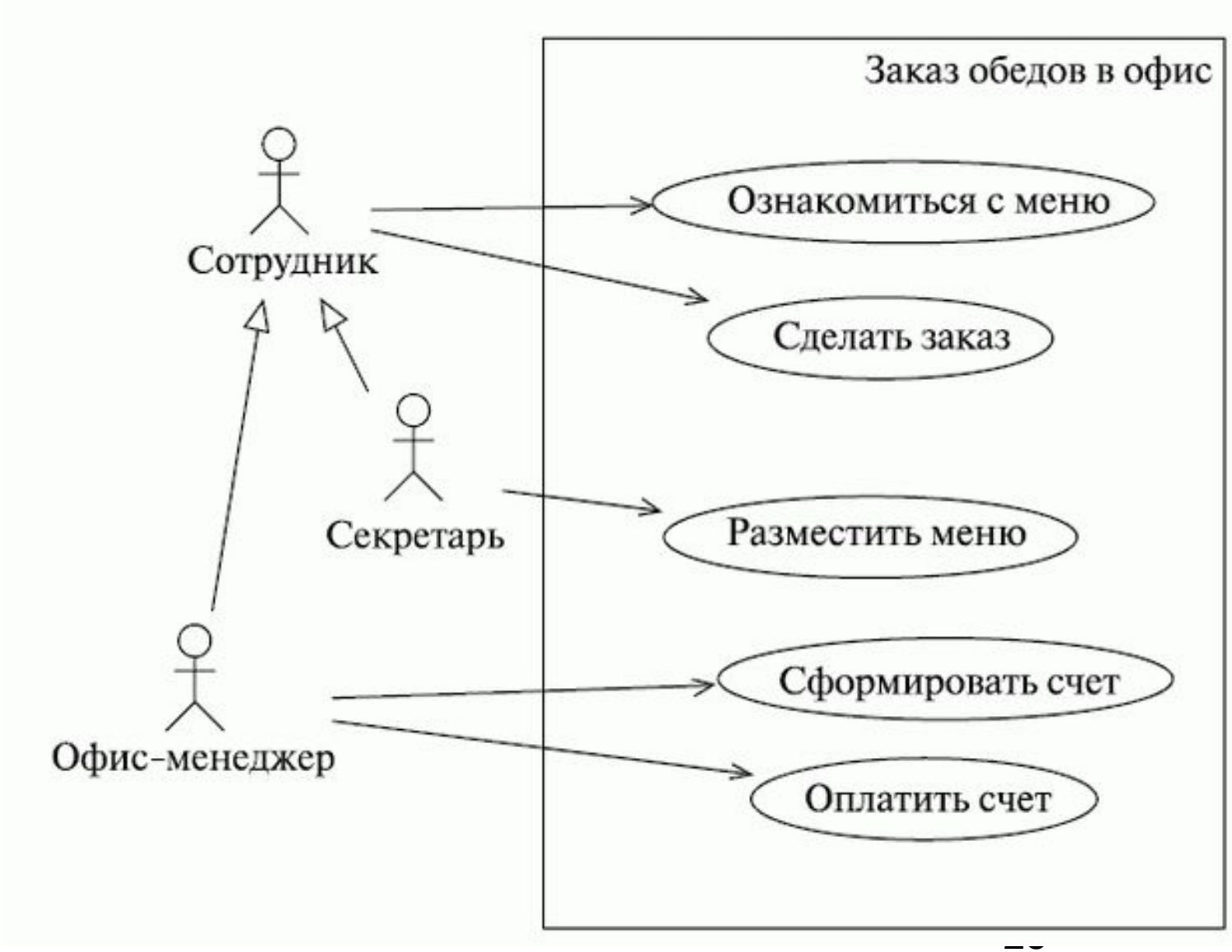


Пример UseCase-диаграммы системы продажи товаров по каталогу

Пример UseCase-диаграммы системы снятия наличных денег со счета через банкомат



Пример UseCase-диаграммы подсистемы оформления заказов обедов



Сценарии вариантов использования

В ряде случаев, когда изобразительных средств UseCase-диаграммы оказывается недостаточно, она может быть дополнена текстовыми *сценариями вариантов использования*, уточняющими детали функционального поведения системы.

При написании сценария важно понимать, что текст сценария не должен заменять собой UseCase-диаграмму – иначе будут потеряны все достоинства визуального представления модели.

Сценарии вариантов использования разрабатываются на базе шаблонов, один из которых, рекомендуемый для применения на начальных этапах концептуального моделирования, приведен ниже.

Шаблон состоит из 4-х разделов и представляется в табличной форме:

1. ***Главный*** раздел
2. Раздел «***Типичный ход событий***»
3. Раздел «***Исключения***»
4. Раздел «***Примечания***»

Пример оформления сценария варианта использования

Сценарий варианта использования "*Снятие наличных денег по кредитной карте*"	
Главный раздел	
Вариант использования	Снятие наличных денег по кредитной карте
Актеры	Клиент, Банк
Цель	Получение запрошенной клиентом денежной суммы наличными
Краткое описание	Клиент запрашивает требуемую сумму. Банкомат обеспечивает доступ к банковскому счету клиента. Банкомат выдает клиенту запрошенную сумму.
Тип	Базовый
Ссылки на другие варианты использования	Включает следующие варианты использования: - идентифицировать кредитную карту - проверить правильность введенного PIN-кода;

Пример оформления сценария варианта использования

Раздел "***Типичный ход событий***"	
Действия актеров	Отклик системы
1. Клиент вставляет кредитную карту в приемное устройство банкомата. Исключение №1. Карта недействительна или неправильно вставлена	2. Банкомат проверяет кредитную карту 3. Банкомат предлагает ввести PIN-код
4. Клиент вводит PIN-код Исключение №2. Введен неверный PIN-код	5. Банкомат проверяет введенный PIN-код 6. Банкомат отображает меню
7. Клиент выбирает пункт меню "Снятие наличных денег"	8. Система делает запрос в Банк и выясняет текущее состояние счета Клиента. 9. Банкомат предлагает Клиенту ввести требуемую сумму.
10. Клиент вводит требуемую сумму. 11. Банк проверяет введенную сумму. Исключение №3. Требуемая сумма превышает сумму, разрешенную к выдаче Клиенту.	12. Банкомат выдает наличные деньги. 13. Банкомат изменяет состояние счета Клиента, уменьшив его на выданную сумму. 14. Банкомат печатает чек.
15. Клиент получает деньги и чек.	16. Банкомат предлагает клиенту забрать кредитную карту.
17. Клиент получает свою кредитную карту.	18. Банкомат отображает сообщение о готовности к работе

Пример оформления сценария варианта использования

Раздел "***Исключения***"	
Действия актеров	Отклик системы
<u>Исключение №1</u> Карта недействительна или неправильно вставлена	
	3. Банкомат отображает информацию о негативном результате проверки кредитной карты. 14. Банкомат возвращает Клиенту кредитную карту.
17. Клиент получает свою кредитную карту.	
<u>Исключение №2</u> Введен неверный PIN-код	
	6. Банкомат отображает информацию о неправильном PIN-коде.
4. Клиент вводит новый PIN-код.	
<u>Исключение №3</u> Требуемая сумма превышает сумму, разрешенную к выдаче	
	12. Банкомат отображает информацию о превышении суммы кредита.
10. Клиент вводит другую требуемую сумму.	

Рекомендуемый порядок разработки UseCase-диаграмм

1. Определить главных (базовых) актеров модели – желательно, чтобы их количество не превышало 20.
2. Определить цели всех базовых актеров по отношению к системе.
3. Определить "второстепенных" актеров модели, уточнить их частные цели по отношению к системе, установить *отношения обобщения* между базовыми (предками) и частными (потомками) актерами.
4. Определить базовые варианты использования – желательно, чтобы их общее количество не превышало 50.
5. Определить *включаемые* варианты использования (последовательности действий, представленные в нескольких базовых вариантах) и обозначить стереотипом "include" их связи с соответствующими базовыми вариантами.
6. Для каждого варианта использования:
   - обозначить на диаграмме актеров-участников, определить их интересы;
   - определить предусловия и постусловия использования варианта;
   - написать сценарий успешного выполнения ("типичный ход событий");
   - определить неуспешные ситуации – "исключения";
   - написать сценарии для всех "исключений".
7. Для каждого "исключения" предусмотреть *расширяющий* вариант использования и обозначить стереотипом "extend" его связь с соответствующим базовым вариантом.
8. Проверить диаграмму на отсутствие дублирования актеров и вариантов использования; исключить дубликаты.

Типичные ошибки, допускаемые разработчиками UseCase-диаграмм

1. UseCase-диаграмма излишне детализируется – разработчик пытается отразить в ней детали и логику поведения системы (вместо отображения функционального поведения), что по сути, превращает ее в *диаграмму деятельности.*
2. Модель логически привязывается к существующим техническим устройствам или способам взаимодействия актеров и вариантов использования, что не допустимо на данной стадии разработки системы.
3. Вместо сценария варианта использования формулируются функциональные требования к системе без указания последовательности выполняемых действий.
4. В сценариях описываются только действия актеров и игнорируются отклики системы.
5. В качестве инициаторов действий указывается система, а не актеры, как это должно быть на самом деле.
6. В тексте сценария отсутствует описание исключительных ситуаций и альтернативных последовательностей действий, что может привести к упущениях в определении состава вариантов использования.
7. При именовании вариантов использования и при написании сценариев используются сокращения и специальные термины, привычные для разработчика, но непонятные заказчику и пользователям системы.
8. До того, как описаны сценарии всех вариантов использования, разработчики приступают к спецификации атрибутов и операций соответствующих классов.

Выявление и анализ требований

UseCase-диаграмма дает общее представление о деятельности и целях организаций, в которых будет работать проектируемая программная система, и о функциях системы, реализуемых ею в процессе взаимодействия с пользователями.

На основе информации о функциях системы, представленной на UseCase-диаграмме, разработчик должен ***конкретизировать состав задач***, которые система будет решать, и ***сформулировать требования*** к системе, которые представляют собой детализацию работы этих функций.

Правила работы с *требованиями к ПО* определяются следующими двумя стандартами IEEE:

IEEE 830-1998 Recommended Practice for Software Requirements Specifications (Рекомендуемые методы спецификации требований к ПО).

IEEE 1233-2002 Guide for Developing System Requirements Specifications (Руководство по разработке спецификаций требований к системам), описывающий правила построения требований для программно-аппаратных систем в целом..

Выявление и анализ требований

Вопросы разработки требований к программным системам будут детально рассматриваться в отдельной дисциплине, предусмотренной учебным планом направления подготовки бакалавров 231000.62 – "Программная инженерия" в 6-м и 7-м семестрах.

Контрольные вопросы и задания

1. Перечислите четыре основные задачи, которые решает разработчик программной системы, используя *UseCase*-диаграммы ?
2. Перечислите основные компоненты *UseCase*-диаграмм, укажите назначение каждого из них и приведите примеры их условных обозначений на диаграмме.
3. Перечислите основные и опциональные атрибуты варианта использования, указываемые в спецификации *UseCase*-диаграммы.
4. Определите понятие "*интерфейс*" как компонент *UseCase*-диаграммы.
5. Определите понятие "*связь* (*отношение*)" как компонент *UseCase*-диаграммы.
6. Рассмотрите примеры *UseCase*-диаграмм, приведенных на слайдах №26, №27 и №28:
   - Дополните обозначения *отношений ассоциации* параметрами "*имя связи*" и "*кратность связи*", обоснуйте значения параметра кратности.
   - Приведите примеры отношений *включения* и *расширения* между вариантами использования, показанных на этих диаграммах. В чем разница между этими типами отношений? К каким последствиям (при разработке сценариев вариантов использования и при их программной реализации) приведет факт установления этих типов отношений?
   - Приведите примеры *отношений обобщения*, показанных на этих диаграммах. Могут ли "*потомки*" не участвовать в связях, обозначенных на диаграмме для их "*предков*"? Могут ли "*потомки*" участвовать в связях, в которых не участвуют их "*предки*"?
7. Определите понятие "*сценарий варианта использования*", опишите структуру основных разделов типового шаблона разработки сценариев.
8. Разработайте сценарии вариантов использования, приведенных на слайде №28.
9. Опишите типовой алго37ритм разработки *UseCase*-диаграммы.
10. Перечислите типичные ошибки, допускаемых разработчиками UseCase-диаграмм.
11. Нарисуйте UseCase-диаграмму системы доступа клиента банка к своим банковским счетам с целью снятия наличных денег или пополнения своих счетов через банкомат.
12. Нарисуйте UseCase-диаграмму системы доступа клиента банка к своему банковскому счету с целью перевода безналичных денежных средств со своего счета на другие счета через банкомат.