Лекция №3

Объектно-ориентированное моделирование

1. *Подходы к моделированию систем.*
2. *Сущность структурного подхода.*
3. *Базовые принципы структурного подхода*
4. *Сущность объектно-ориентированного подхода.*
5. *Базовые принципы объектно-ориентированного подхода.*

1. Подходы к анализу и проектированию ИС

1. Структурный (функционально-модульный)

- *использует **функциональную декомпозицию**, где структура системы описывается в терминах иерархии ее функций и передачи информации между отдельными функциональными элементами*

2. Объектно-ориентированный

- *использует **объектную декомпозицию**, где структура системы описывается в терминах объектов и связей между ними, а поведение системы описывается в терминах обмена сообщениями между объектами*

1. Сущность структурного подхода

Сущность структурного подхода заключается в его декомпозиции (разбиении) на автоматизируемые функции:

- система разбивается на функциональные подсистемы, которые, в свою очередь, делятся на подфункции, те – на задачи и так далее до конкретных процедур;
- при этом автоматизируемая система сохраняет целостное представление, в котором все составляющие компоненты взаимоувязаны.

при разработке системы «снизу вверх», от отдельных задач ко всей системе, целостность теряется, возникают проблемы при описании информационного взаимодействия отдельных компонентов.

2. Базовые принципы структурного подхода

1. Разделяй и властвуй

- для решения сложной задачи используется ее разбиение на несколько более простых подзадач.

2. Иерархическое упорядочивание

- решение задачи представляется в виде иерархии описаний (функций, потоков данных), где описания вышестоящих уровней детализируются при помощи описаний нижестоящих уровней.

3. Абстрагирование

- при решении задачи рассматриваются только важные ее аспекты, а всё незначительное отбрасывается.

4. Формализация

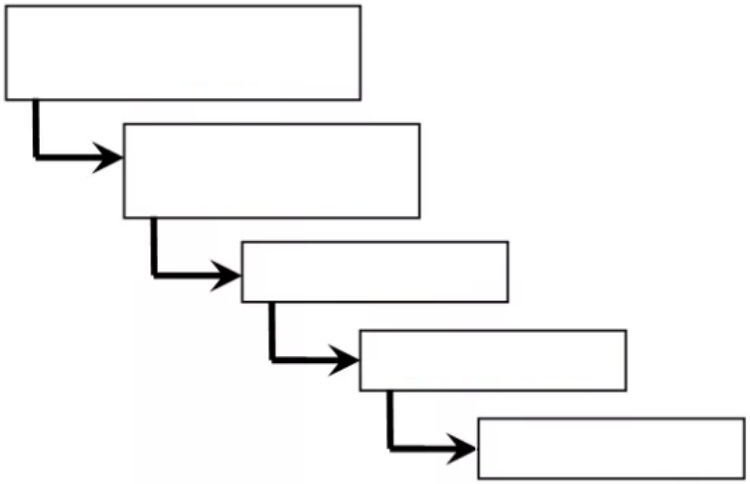
- все принятые решения описываются формально, т.е. применяются формализованные методы описания и моделирования изучаемых и проектируемых процессов.

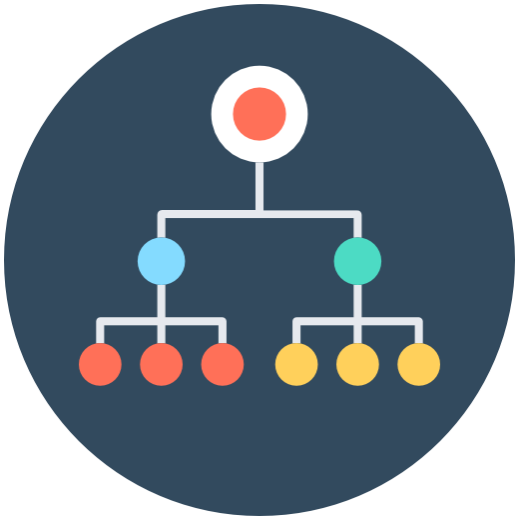
5. Непротиворечивость

- никакая часть системы не может противоречить никакой другой, все должно быть согласованно и находиться в соответствии.

Модель процесса функциональной декомпозиции

Для структурного подхода характерна *каскадная модель жизненного цикла* системы, когда каждый последующий этап решения задачи выполняется после полного завершения предыдущего этапа (является следствием принципа иерархической упорядоченности)





- Система (как и ее описание) представляет собой иерархическую структуру, в которой вышестоящие уровни определяются через нижестоящие.
- *Любую вершину в иерархии можно считать определенной только в том случае, если определены все ее потомки*.

Структурный подход: диаграммы

1. Иллюстрирующие функции, которые система должна выполнять

например, методология SADT (модель IDEF0), BPMN и др.

2. Иллюстрирующие отношения между данными

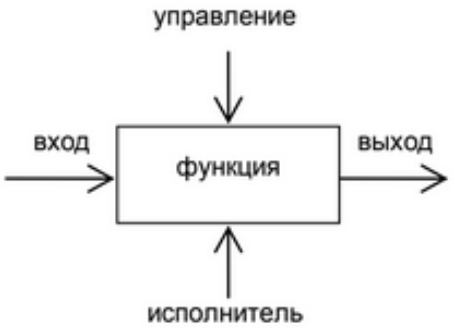
например, DFD - диаграммы потоков данных совместно со спецификациями процессов нижнего уровня (миниспецификациями)

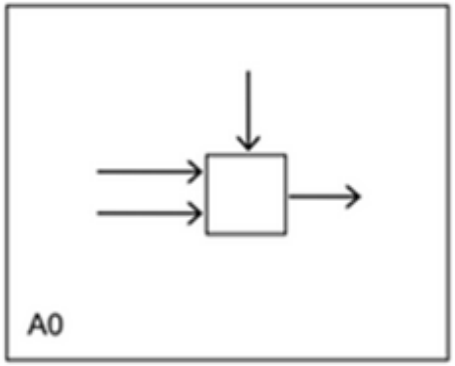
3. Иллюстрирующие динамическое поведение системы

- например, IDEF3 (ARIS EPC) – диаграмма процессов, происходящих в системе, отражающая причинно-следственные связи между ситуациями и событиями

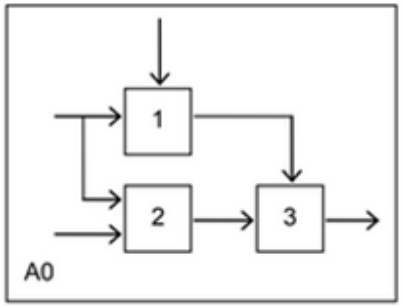
Диаграмма SADT (IDEF0): иерархия

Функциональная модель IDEF0 отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями.

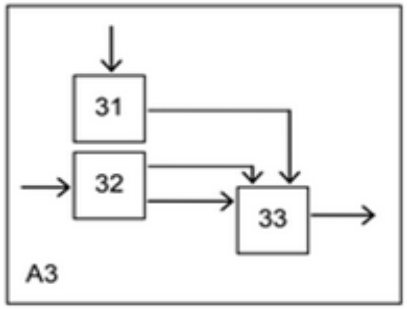




1. Строится корневая диаграмма, на которой моделируемый объект представляется в виде одной функции, для которой определяются все входные параметры, выходные величины, управляющая информация, а также исполнители.

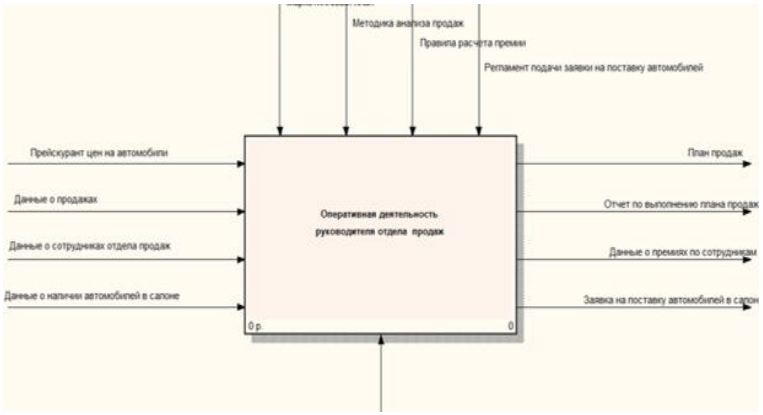


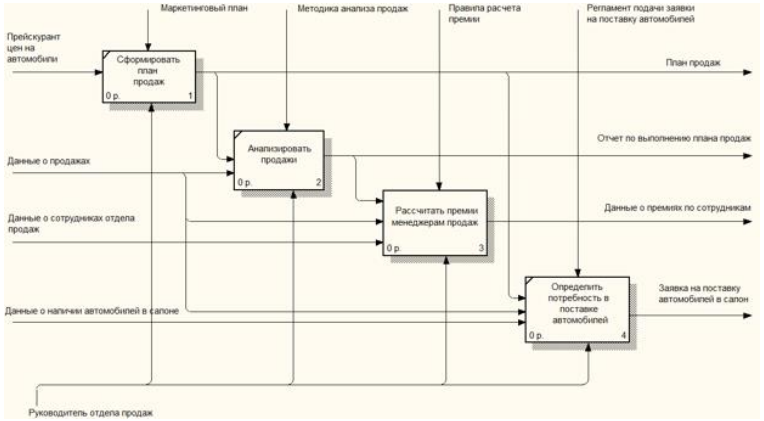
2. Полученная диаграмма детализируется путем разбиения основной функции на несколько подфункций.



3. Процесс детализации рекурсивно повторяется для каждой нетривиальной подфункции

Пример реализации структурного подхода в нотации IDEF0





Примеры структурного подхода в нотациях IDEF3 и DFD

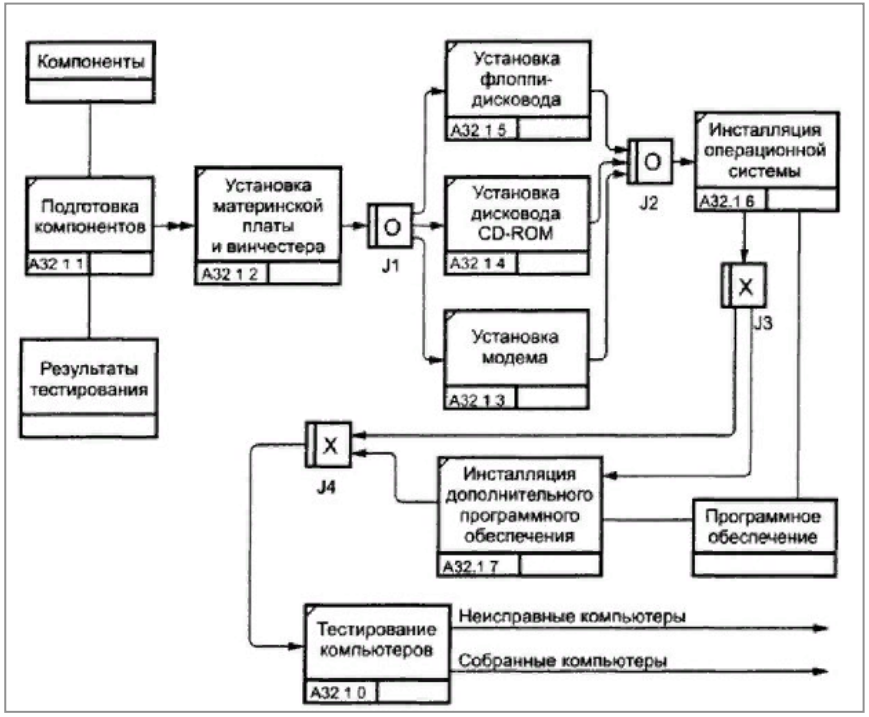
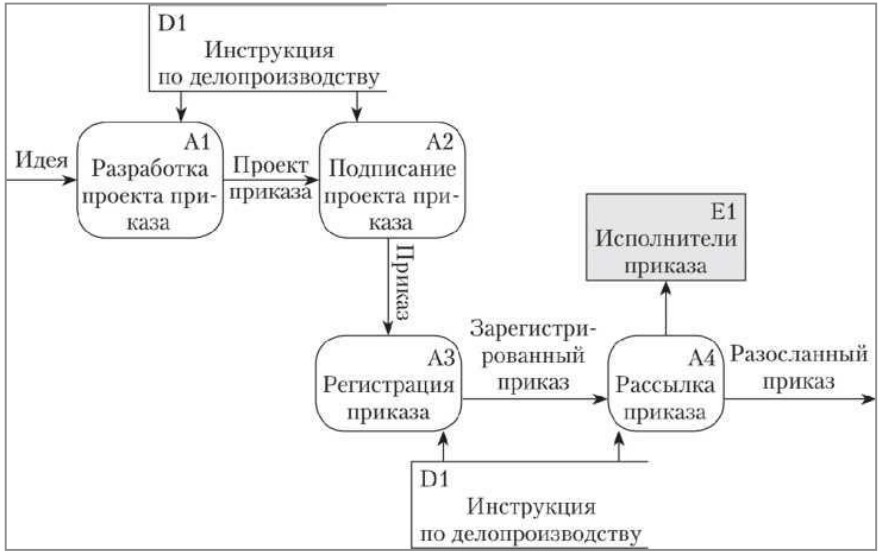
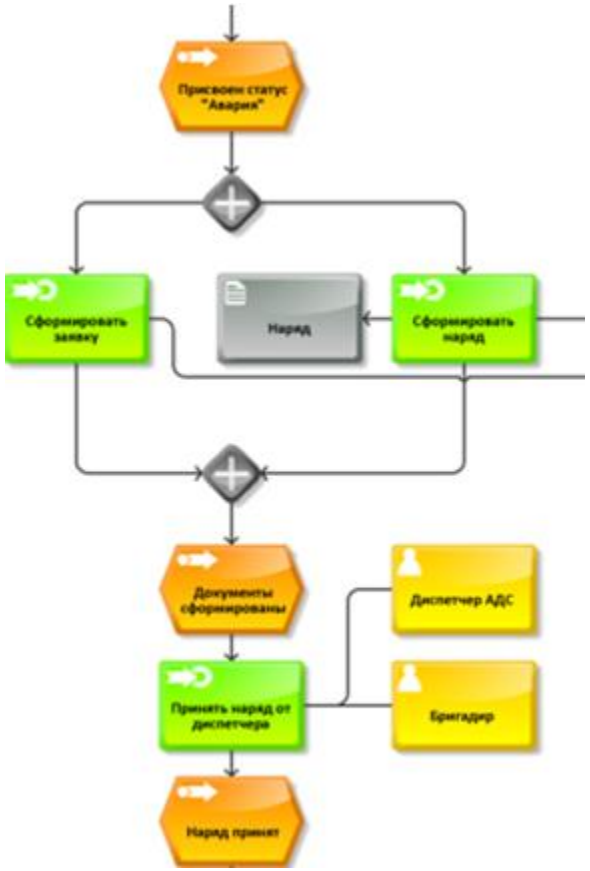


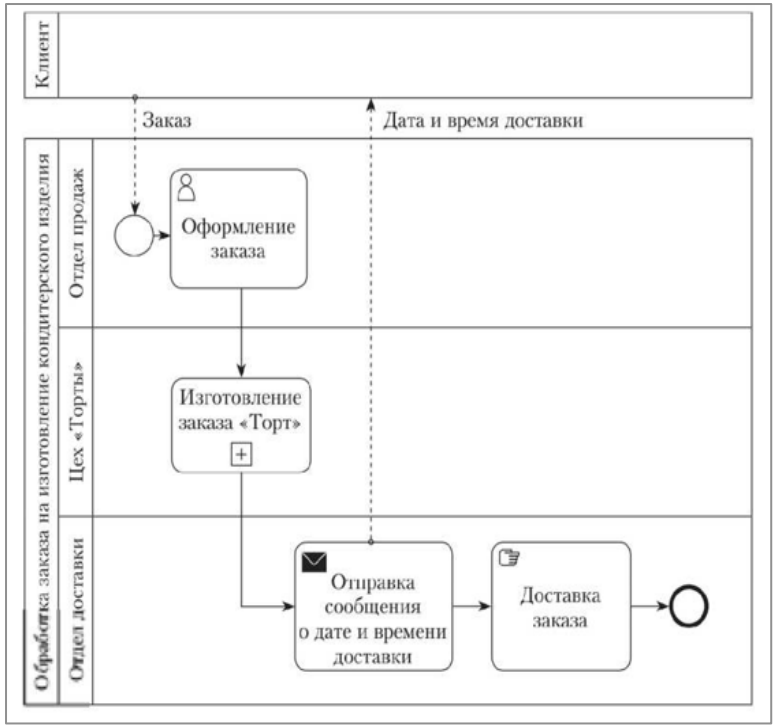
Диаграмма IDEF3
(динамическое поведение системы)



DFD - диаграмма
(отношения между данными)

Примеры структурного подхода в ARIS Express и BPMN



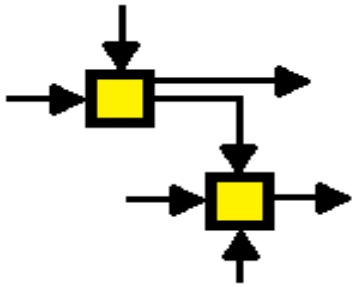


Методология структурного подхода: особенности

В функциональных моделях *(DFD-диаграммах потоков данных, SADT-диаграммах и др.) главными структурными компонентами являются* ***функции*** *(операции, действия, работы), которые на диаграммах связываются между собой потоками объектов.*

для функциональных моделей характерны процедурная строгость декомпозиции ИС и наглядность представления

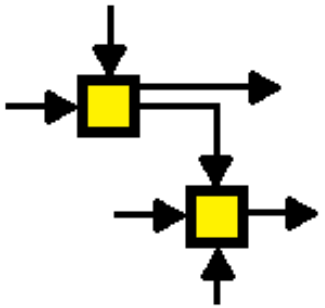
Структурный подход к моделированию



Достоинства:

- *возможность проведения глубокого анализа бизнес-процессов, выявления узких мест: комплексное применение позволяет выявить все возможные рассогласования и неточности;*
- *применение универсальных графических языков моделирования IDEF0, IDEF3, DFD и др. обеспечивает логическую целостность и полноту описания, необходимую для достижения точных и непротиворечивых результатов;*
- *методология проверена временем, широко распространена среди аналитиков и разработчиков;*
- *имеется возможность модульного проектирования ИС*

Структурный подход к моделированию



Недостатки:

- *низкая наглядность для неподготовленных пользователей модели: при увеличении количества уровней представления, анализ и модификация моделей становятся затруднительными;*
- *сложность восприятия иерархически упорядоченной информации;*
- *необходимость следования жёсткой структуре;*
- *процессы и данные существуют отдельно друг от друга;*
- *структура данных находится на втором плане;*
- *не ясны условия выполнения процессов обработки информации, которые динамически могут изменяться.*

4. Сущность объектно-ориентированного подхода

Первые объектно-ориентированные методы появились в начале 80-х гг. XX в., однако основные результаты были получены спустя 10 лет.

- Айвар Джекобсон - *Object-Oriented Software Engineering* (*OOSE*)
- Джим Рембо - *Object Modeling Technique* (*OMT*)
- Грейди Буч - *Booch.*

Эти методы были впоследствии объединены в один, из которого вырос унифицированный язык моделирования *–UML (Unified Modeling Language*).

Объектный подход

- ***Суть объектного подхода*** *заключается в объектной декомпозиции, т.е. система представляется в виде совокупности объектов, которые в процессе взаимодействия обмениваются сообщениями.*
- ***Объект*** *– это самостоятельная, самодостаточная сущность, обладающая состоянием, поведением и семантикой.*

Авторы объектного подхода поставили перед собой три главные ***цели****:*

- *– моделировать системы целиком, от концепции до исполняемого артефакта, с помощью объектно-ориентированных методов;*
- *– решить проблему масштабируемости, которая присуща сложным системам, предназначенным для выполнения ответственных задач;*
- *– создать такой язык моделирования, который может использоваться не только людьми, но и компьютерами.*

4. Базовые принципы объектно-ориентированного подхода

1.Уникальность

- каждый объект имеет свой адрес в памяти, и две объектные переменные считаются равными только в том случае, если они указывают на один и тот же адрес.

2. Классификация

- все *объекты объединяются в классы по принципу сходства структуры, поведения и семантики.*

3. Инкапсуляция

- *совместное «хранение» данных и методов их обработки. В широком смысле инкапсуляция – это скрытие реализации за интерфейсом, т.е. объект обладает внутренней, известной лишь ему структурой и интерфейсом.*

4. Наследование

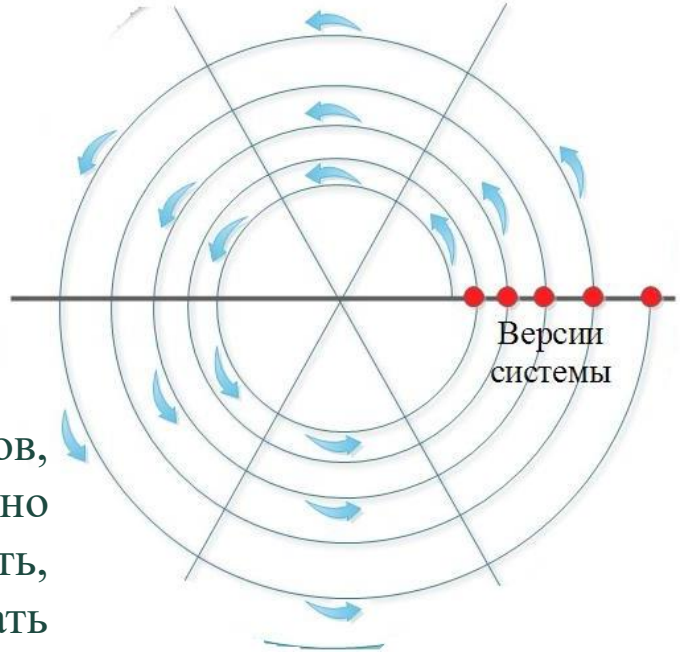
- *классы могут объединяться в иерархии наследования. Структура, поведение и семантика объектов наследуются вниз по иерархии.*

5. Полиморфизм

- *возможность доступа к нескольким реализациям через один интерфейс. Способом реализации полиморфизма в объектно-ориентированных языках программирования является механизм переопределения унаследованных методов.*

Модель процесса объектной декомпозиции

Для объектно-ориентированного подхода характерна *спиральная модель жизненного цикла* системы.



Применяется система объектов, каждый из которых можно анализировать, проектировать, реализовывать и тестировать практически независимо от других.

Имеется возможность формирования итераций, на каждой из которых выполняются все основные этапы разработки для относительно небольшой группы объектов.

Объектно-ориентированный анализ и проектирование

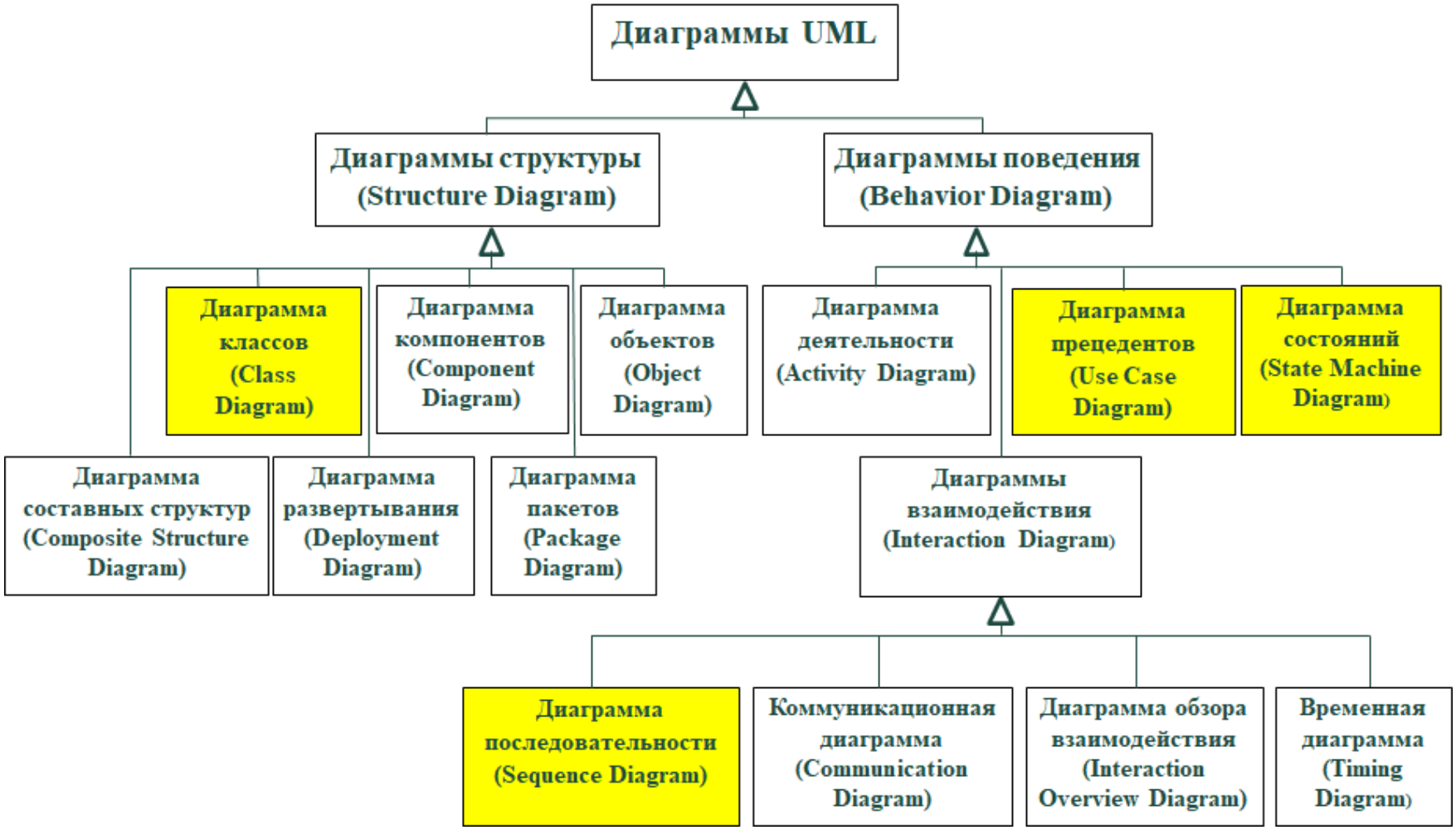
Объектно-ориентированный анализ: *концептуальные модели*

- основное внимание уделяется выделению и описанию понятий предметной области, их структуры и связей между ними.
- *например, в задаче автоматизации библиотеки среди понятий могут присутствовать «библиотека», «издание», «читатель» и др.*

Объектно-ориентированное проектирование: *логические модели*

- основное внимание уделяется определению логических программных объектов, а также их взаимосвязи и взаимодействию между ними.
- объекты характеризуются **атрибутами** и **операциями**.
- *например, класс «издание» может содержать атрибут «название» и операцию «выдать».*

Виды диаграмм объектно-ориентированного подхода



Примеры статических диаграмм: объектный подход

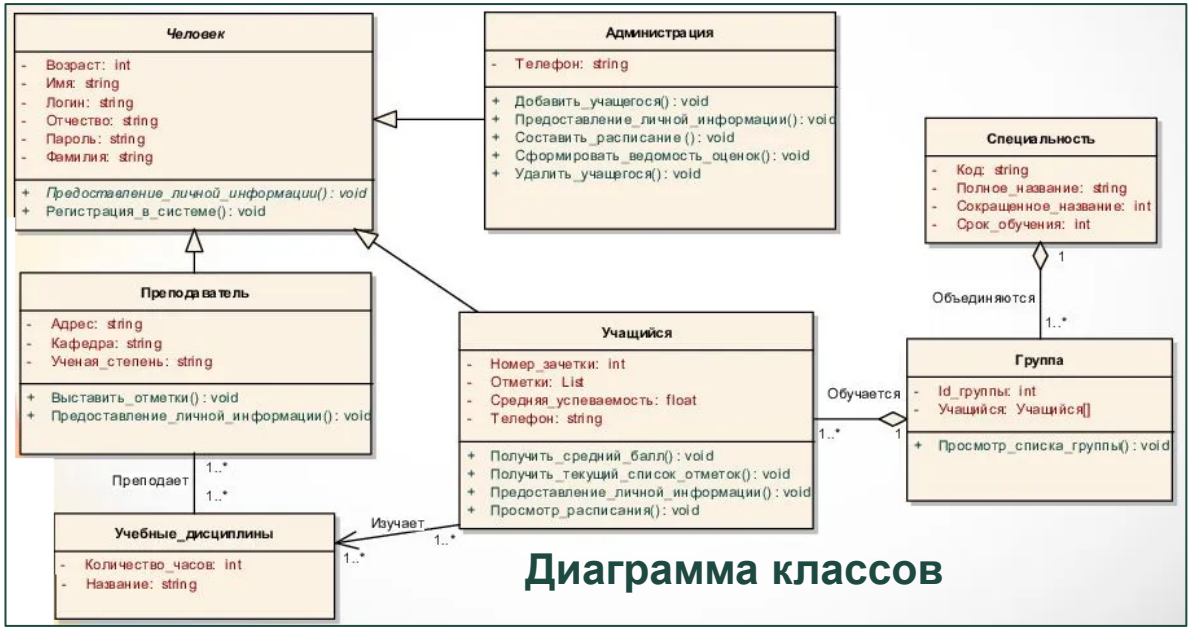


Диаграмма классов

Диаграмма компонентов

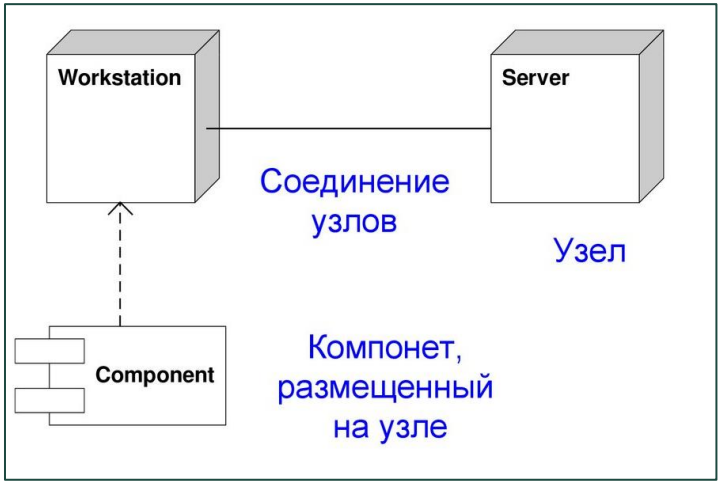


Диаграмма размещения

Примеры диаграмм поведения: объектный подход

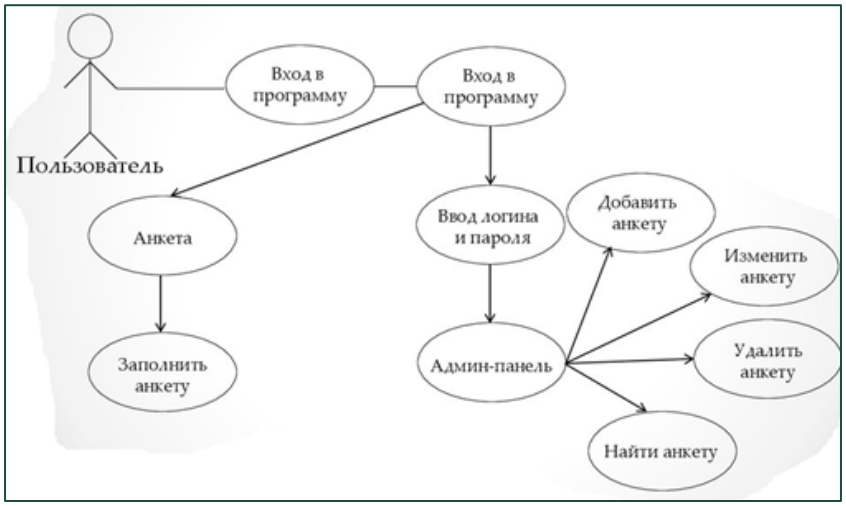


Диаграмма вариантов использования

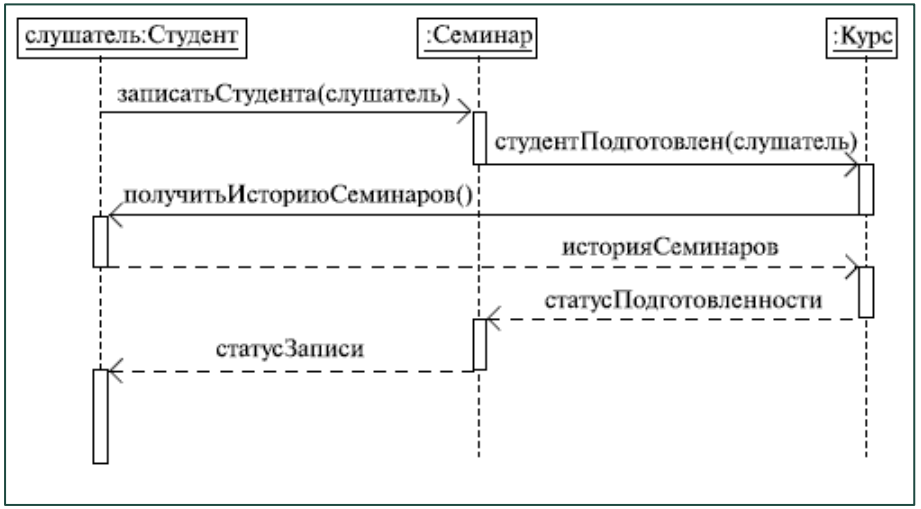


Диаграмма последовательности

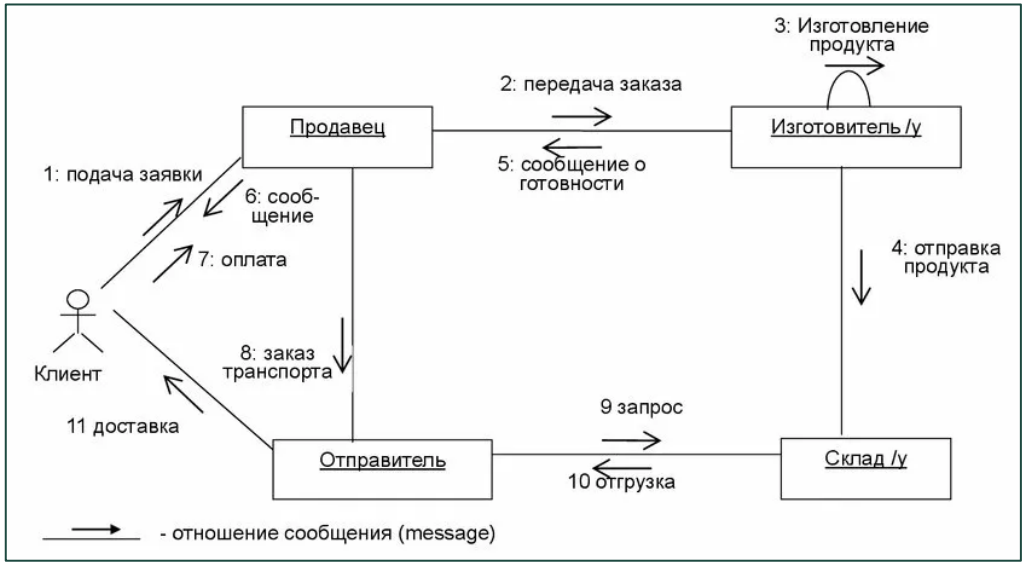


Диаграмма кооперации

Диаграмма последовательности

РТУ МИРЭА, кафедра ППИ

Диаграммы взаимодействия

- ***Диаграммы взаимодействия*** *- модели, описывающие поведение взаимодействующих групп объектов.*
- Как правило, диаграмма взаимодействия охватывает поведение **только одного варианта использования**.
- На диаграмме взаимодействия отображается ряд объектов и те сообщения, которыми они обмениваются между собой в рамках данного варианта использования.

Основные виды диаграмм взаимодействия

диаграммы последовательности (sequence diagrams)

описывают **поведенческие аспекты системы во времени** (*отображают взаимодействие объектов в динамике, т.е. временные аспекты передачи и приемы сообщений)*

кооперативные диаграммы (collaboration diagrams)

описывают **структурные аспекты поведения** системы для спецификации особенностей реализации наиболее значимых операций в системе

Два измерения диаграммы последовательности

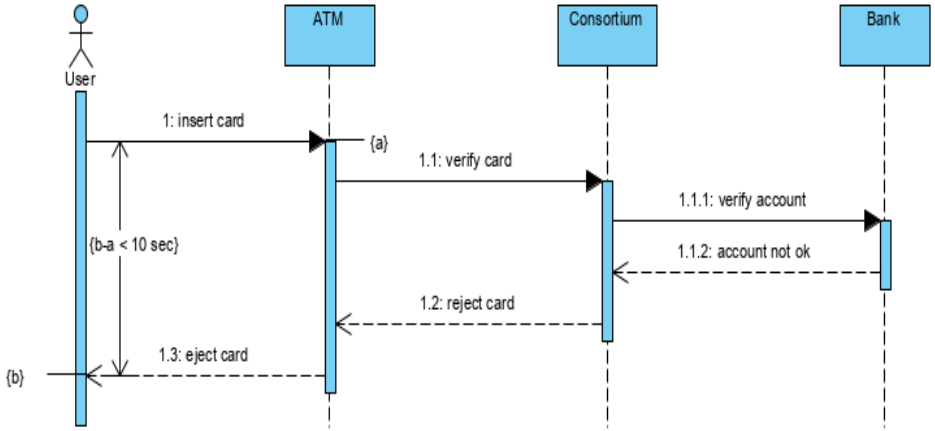
Первое измерение – (объекты с линиями жизни)

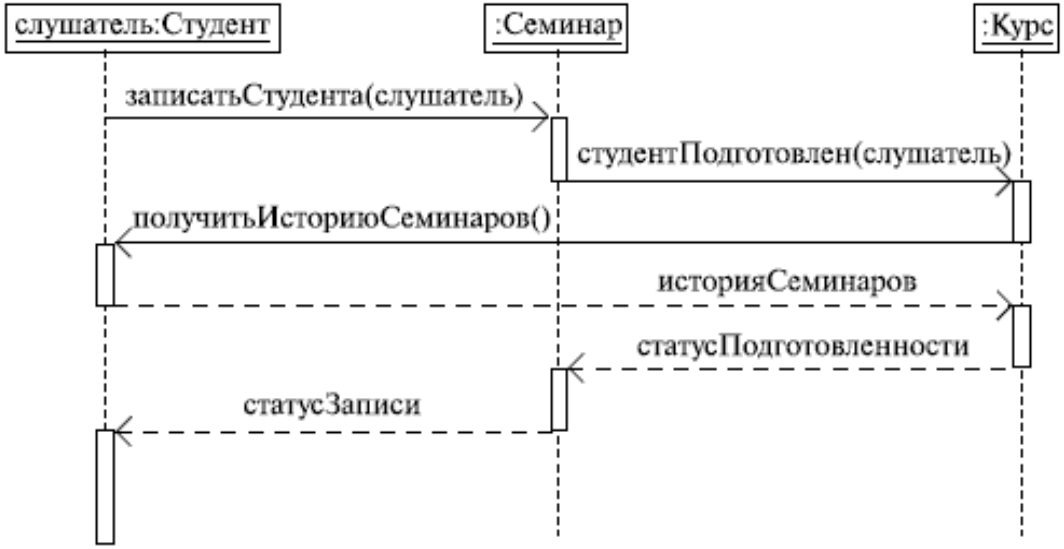
- слева направо в виде вертикальных линий, каждая из которых соответствует линии жизни отдельного участника взаимодействия

Второе измерение - (время по линии жизни)

- сверху вниз по вертикальным линиям жизней каждого отдельного участника взаимодействия

Пример диаграммы последовательности





Элементы диаграммы последовательности

Объекты - прямоугольник (горизонтальный)

- соответствует **объекту**, участвующему в сценарии в рамках одного варианта использования

Линия жизни - вертикальная пунктирная линия

- соответствует **линии жизни** (фрагменту жизненного цикла) объекта в процессе взаимодействия

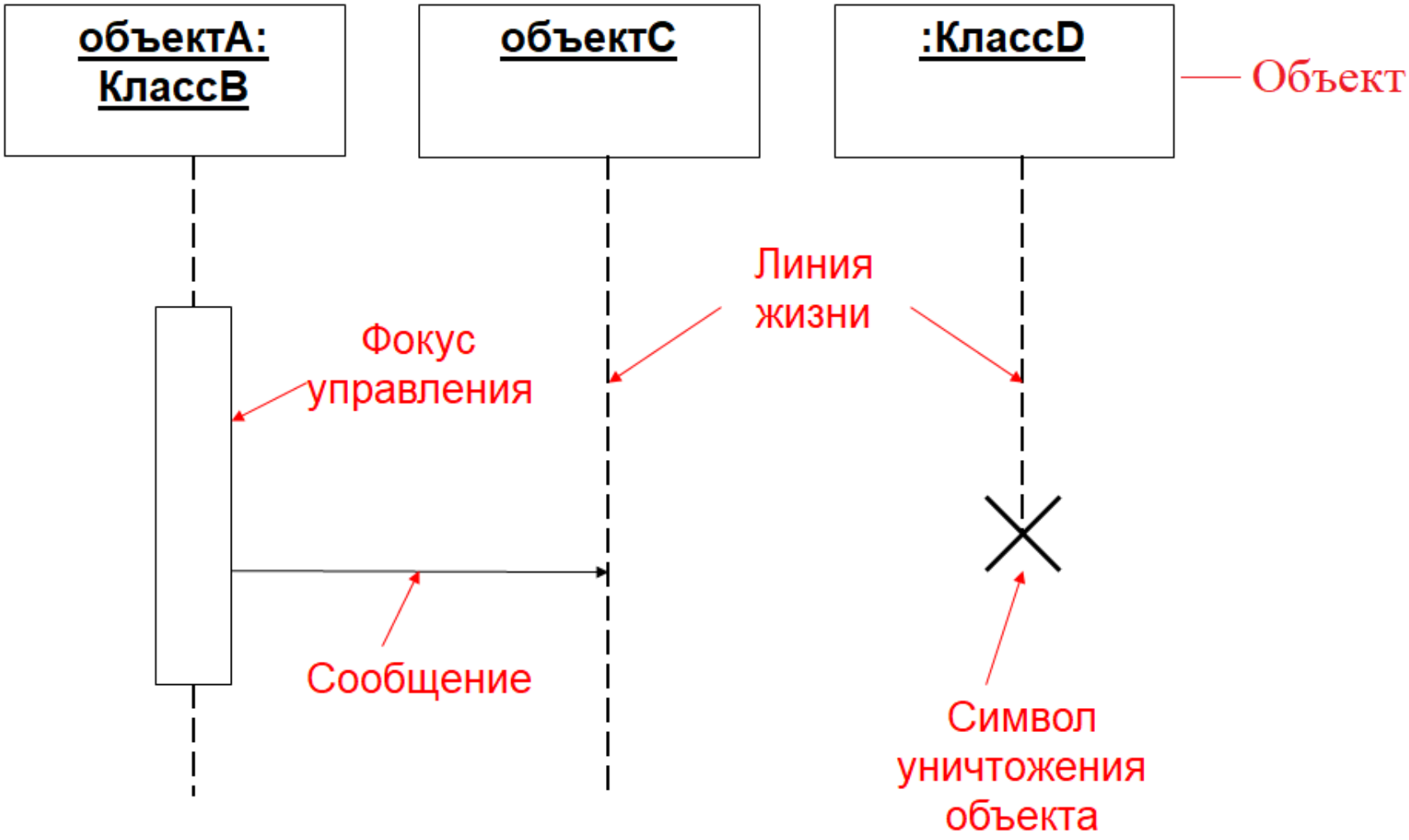
Фокус управления - прямоугольник (вертикально вытянутый)

- соответствует **фокусу управления** (активности объектов)

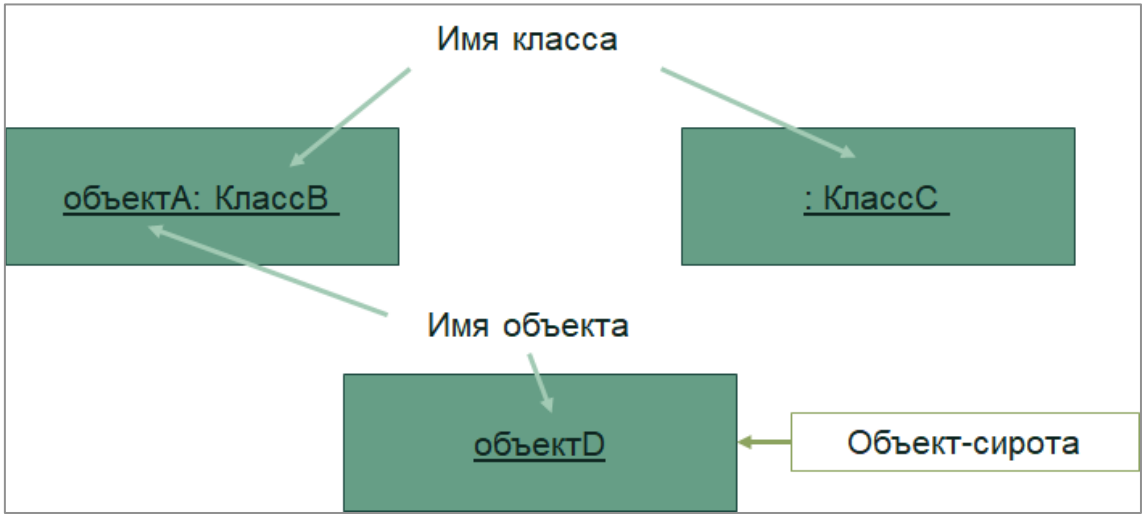
Сообщения - стрелки

- соответствуют **сообщениям** (вызовам методов), передаваемым между действующим лицом и объектом или между объектами для выполнения требуемых функций.

Элементы диаграммы последовательности



Объекты



Каждый объект графически изображается в форме прямоугольника и располагается в верхней части своей линии жизни.

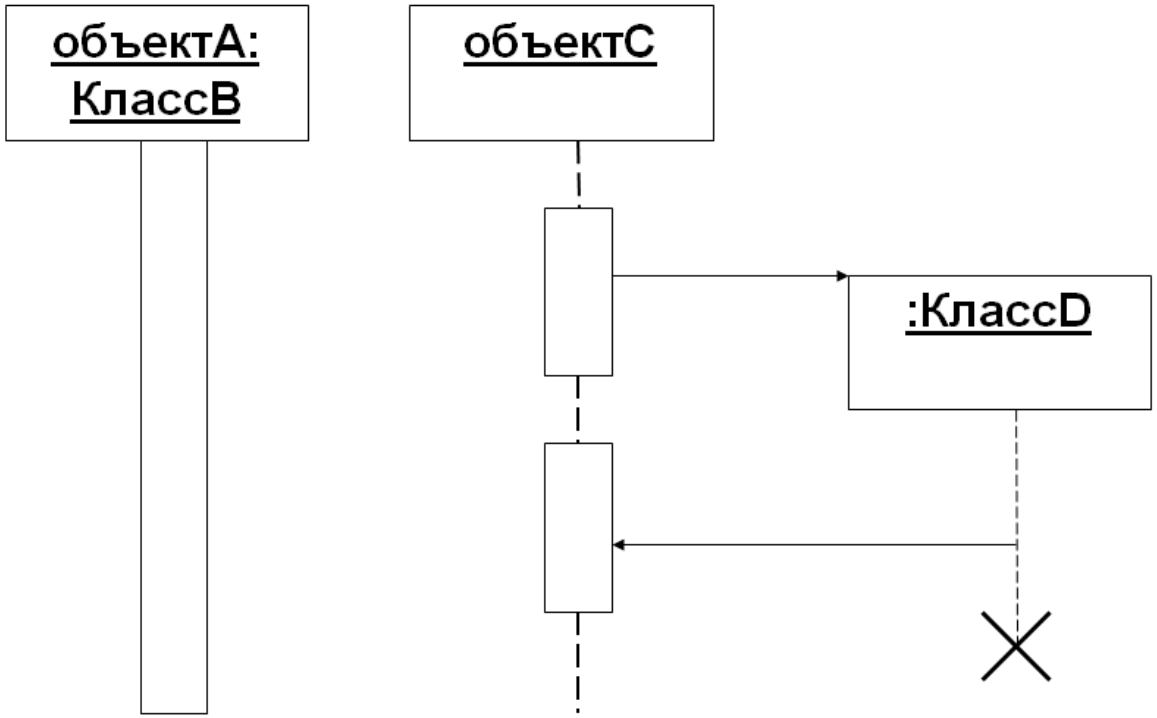
- Внутри прямоугольника записываются **собственное имя объекта со строчной буквы и имя класса, разделенные двоеточием**, при этом вся запись подчеркивается, что является признаком объекта, который, представляет собой **экземпляр класса**.

анонимный объект

объект сирота

Линия жизни объекта (object lifeline)

- *- вертикальная линия на диаграмме последовательности, которая представляет существование объекта в течение определенного периода времени.*



изображается пунктирной вертикальной линией, служит для обозначения периода времени, в течение которого объект существует в системе и, следовательно, может потенциально участвовать во всех ее взаимодействиях.

Линия жизни объекта (object lifeline)

- если объект существует в системе постоянно, то и его линия жизни должна продолжаться по всей рабочей области диаграммы последовательности от самой верхней ее части до самой нижней.
- если объект создается в процессе взаимодействия, то его линия жизни начинается с момента его создания.
- отдельные объекты, закончив выполнение своих операций, могут быть уничтожены, для них линия жизни обрывается в момент его уничтожения.
- для обозначения момента уничтожения объекта в языке UML применяется специальный символ в форме латинской буквы "**X**".

Нотации жизненной линии

- *Нотация жизненной линии с символом элемента агента используется в том случае, если конкретная диаграмма последовательности принадлежит случаю использования.*

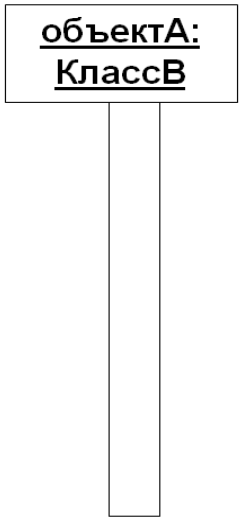
Линия жизни с элементом-субъектом представляет системные данные. Например, в приложении "Обслуживание клиентов" организация-заказчик будет управлять всеми данными, относящимися к клиенту.

Линия жизни с пограничным элементом обозначает системную границу/программный элемент в системе; например, экраны пользовательского интерфейса, шлюзы базы данных или меню, с которыми взаимодействуют пользователи, являются границами.

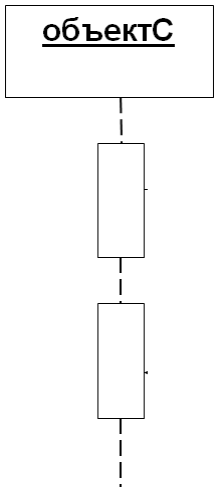
Линия жизни с элементом контроля указывает на контролирующую организацию или менеджера. Он организует и составляет график взаимодействия между границами и субъектами и выступает в качестве посредника между ними.

Фокус управления (focus of control)

Фокус управления - специальный символ на диаграмме последовательности, указывающий период времени, в течение которого объект выполняет некоторое действие, находясь в активном состоянии.



Изображается в форме вытянутого узкого прямоугольника, верхняя сторона которого обозначает начало получения фокуса управления объекта (начало активности), а нижняя сторона - окончание фокуса управления (окончание активности).

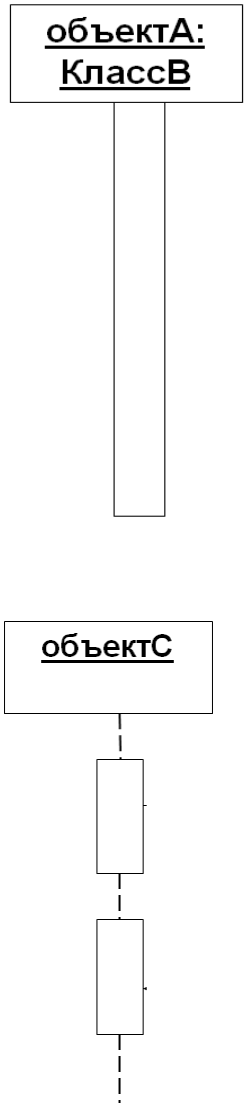


Фокус управления (focus of control)

фокус управления располагается ниже обозначения объекта и может заменять его линию жизни, если на всем ее протяжении он активен.

периоды активности объекта могут чередоваться с периодами его пассивности или ожидания, тогда фокусы управления изменяют свое изображение на линию жизни и наоборот.

получить фокус управления может только объект, у которого в этот момент имеется линия жизни, если же объект был уничтожен, то вновь возникнуть в системе он уже не может, вместо него может быть создан лишь экземпляр этого же класса, который, строго говоря, будет другим объектом.



Сообщения

- Сообщение представляет собой законченный фрагмент информации, который отправляется одним объектом другому;
- Прием сообщения инициирует выполнение определенных действий.

Не все действия связаны с передачей информации и отправкой сообщений.
В UML таковыми считаются:

- вызов операции;
- создание объекта;
- уничтожение объекта;
- возврат значения;
- посылка сигнала.

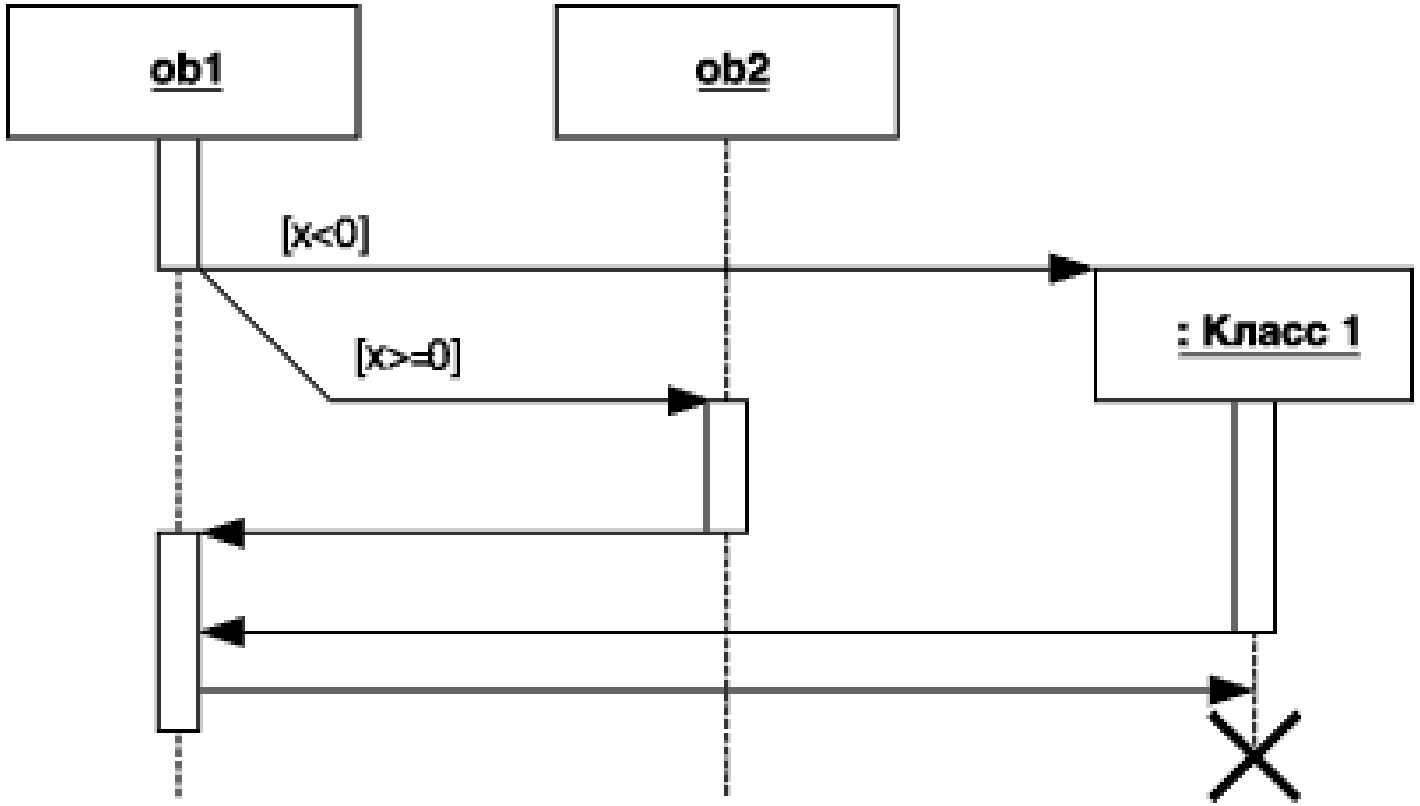
Действие записывается в виде текста над (или рядом со) стрелкой, символизирующей сообщение.
Если действие имеет параметры (вызов операции, создание объекта, посылка сигнала), то аргументы, соответствующие параметрам по числу и типам, записываются справа от имени действия в круглых скобках

Сообщения

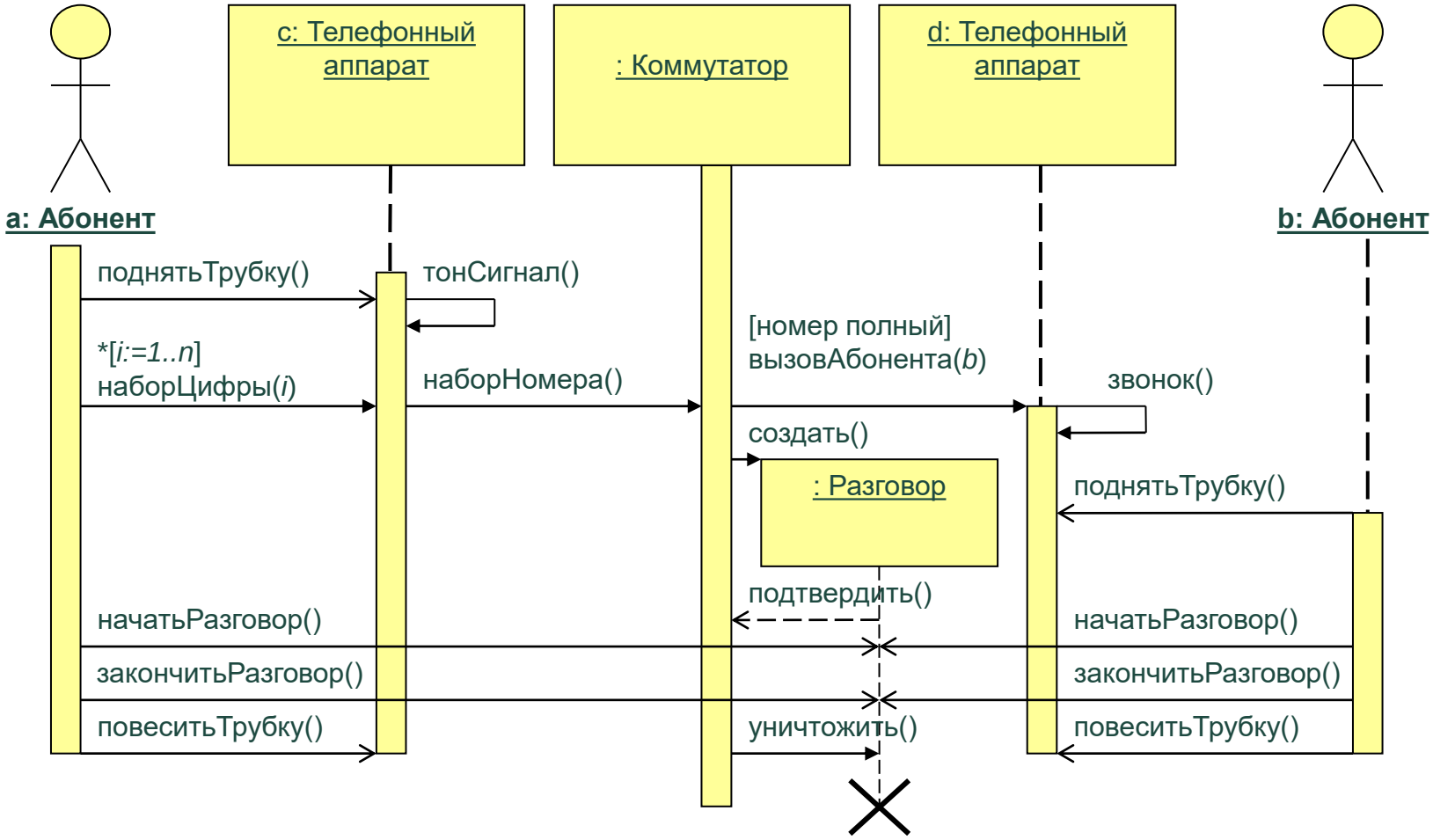
- Взаимодействие между экземплярами моделируется через обмен **сообщениями.**

Найденное сообщение	Синхронное сообщение	Возврат	Асинхронное сообщени	Самовызов
есть получатель, но нет отправителя	для вызова процедур, выполнения операций обозначения отдельных вложенных потоков управления	используется для возврата из вызова процедуры, т.е. указывают на возвращаемые результаты	передается в произвольный момент времени, обычно не сопровождается получением фокуса управления объектом-получателем	сообщение, посланное самому себе

Ветвление потока



Пример диаграммы последовательности



Виды сообщений и из интерпретация

синхронное сообщение:

- *клиент, инициирующий поток сообщений, должен дождаться его завершения, т.е. возврата управления;*

асинхронное сообщение:

- *клиент, инициирующий поток сообщений, не дожидаясь ответа выполняет следующие операции;*

возвращающее сообщение:

- *указывает на возврат значения или управления от получателя обратно к отправителю;*
- *часто отсутствует на диаграммах, поскольку неявно предполагается их существование после окончания процесса выполнения операции.*

рефлексивное сообщение:

- *указывает на сообщение, отправленное самому себе;*
- *начало и конец сообщения соприкасаются с линией жизни или фокусом управления одного и того же объекта.*

Диаграмма кооперации

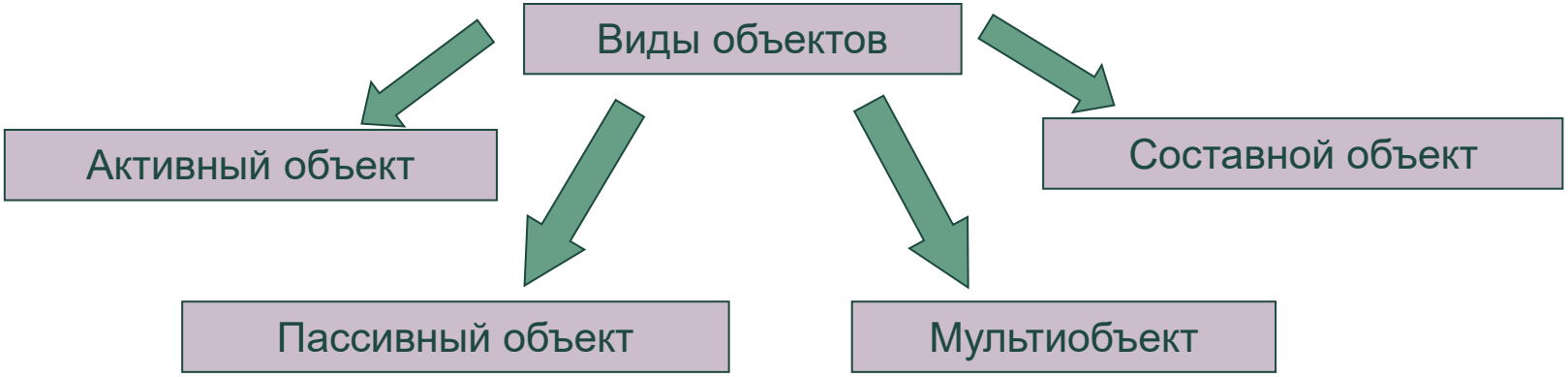
Кооперация (collaboration) служит для обозначения множества взаимодействующих с определенной целью объектов в общем контексте моделируемой системы. Цель самой кооперации состоит в том, чтобы специфицировать особенности реализации отдельных наиболее значимых операций в системе. Кооперация определяет структуру поведения системы в терминах взаимодействия участников этой кооперации.

Кооперация может быть представлена на двух уровнях:

- На уровне спецификации – показывает роли классификаторов и роли ассоциаций в рассматриваемом взаимодействии.
- На уровне примеров – указывает экземпляры и связи, образующие отдельные роли в кооперации.

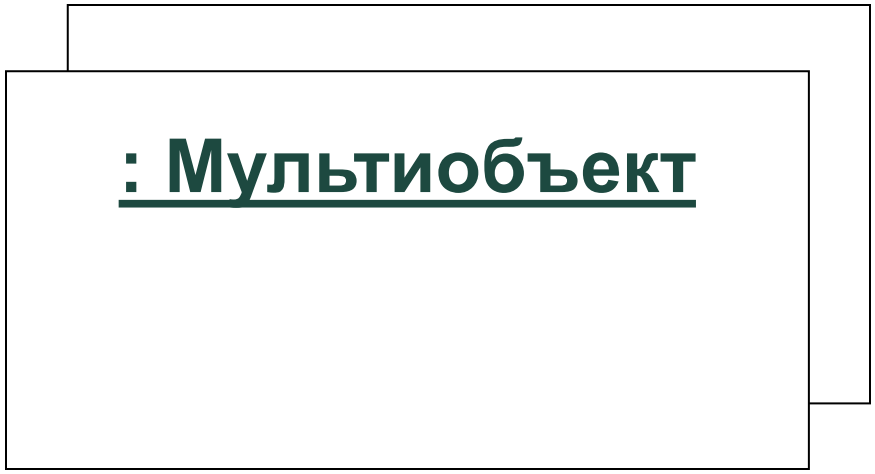
Объекты

- **Объект** является отдельным экземпляром класса, который создается на этапе реализации модели (выполнения программы)



Мультиобъект

- Представляет собой множество объектов, которые могут быть образованы на основе класса.

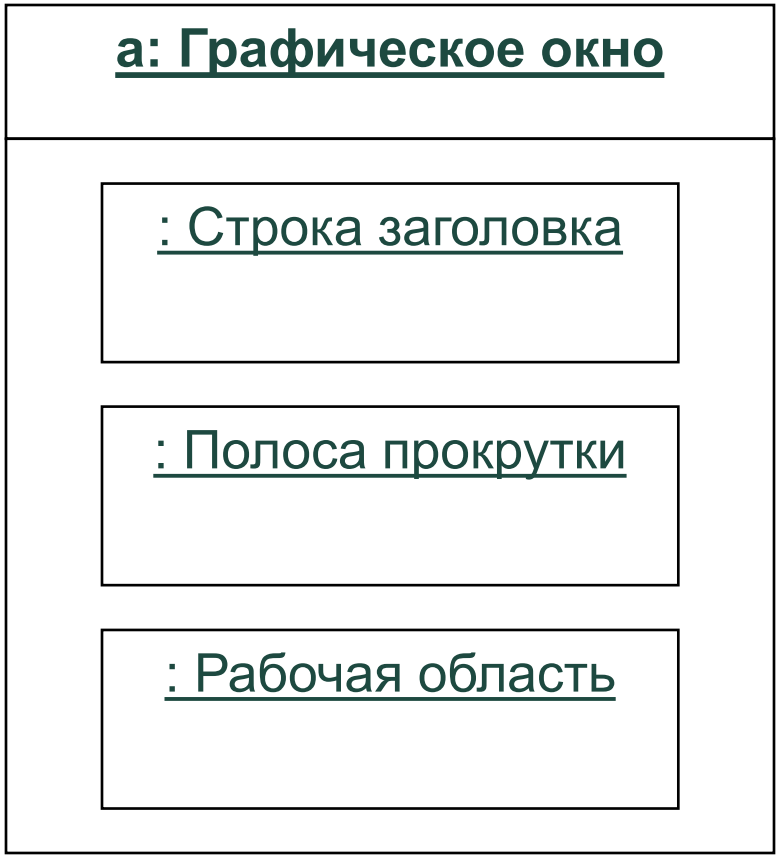


Активный объект

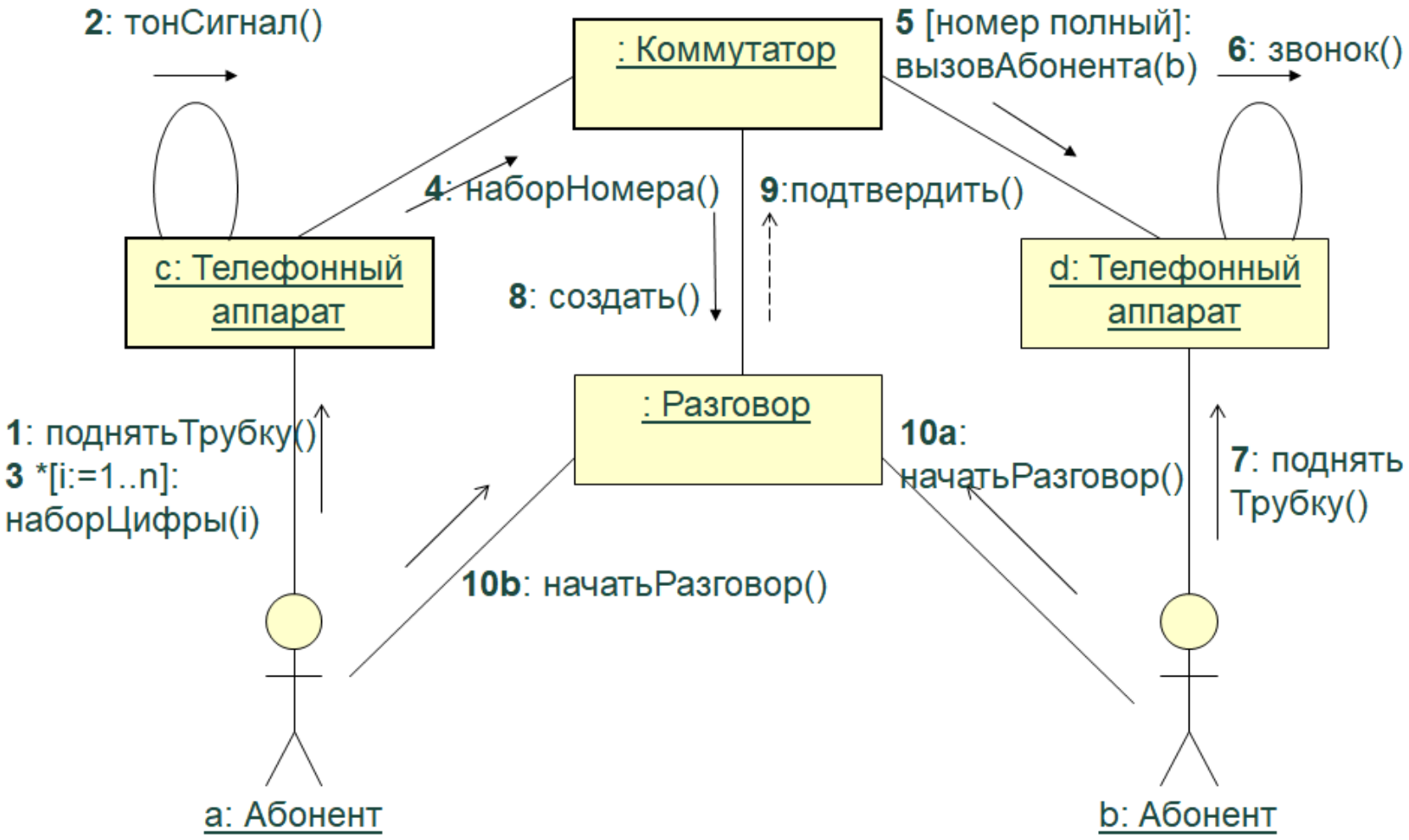
- В контексте языка UML объекты делятся на активные и пассивные.
- *Активный объект* имеет свой собственный поток управления и может инициировать деятельность по управлению другими объектами.

Составной объект

- Предназначен для представления объекта, имеющего **сложную структуру** и внутренние потоки управления.



Пример диаграммы кооперации



Диаграммы последовательности и кооперации: отличие

- На **диаграмме кооперации** изображаются только такие отношения между объектами, которые играют роль информационных каналов при взаимодействии.
- На **диаграмме кооперации** не указывается время в виде дополнительного измерения.
- Таким образом, в диаграмме последовательности делается акцент на временной аспект, в диаграмме кооперации – на статическое взаимодействие объектов системы.

Основные правила разработки диаграмм взаимодействия

1. Для выбранного варианта использования необходимо перенести с диаграммы классов анализа все участвующие в нем классы, а с диаграммы вариантов использования – актеров.

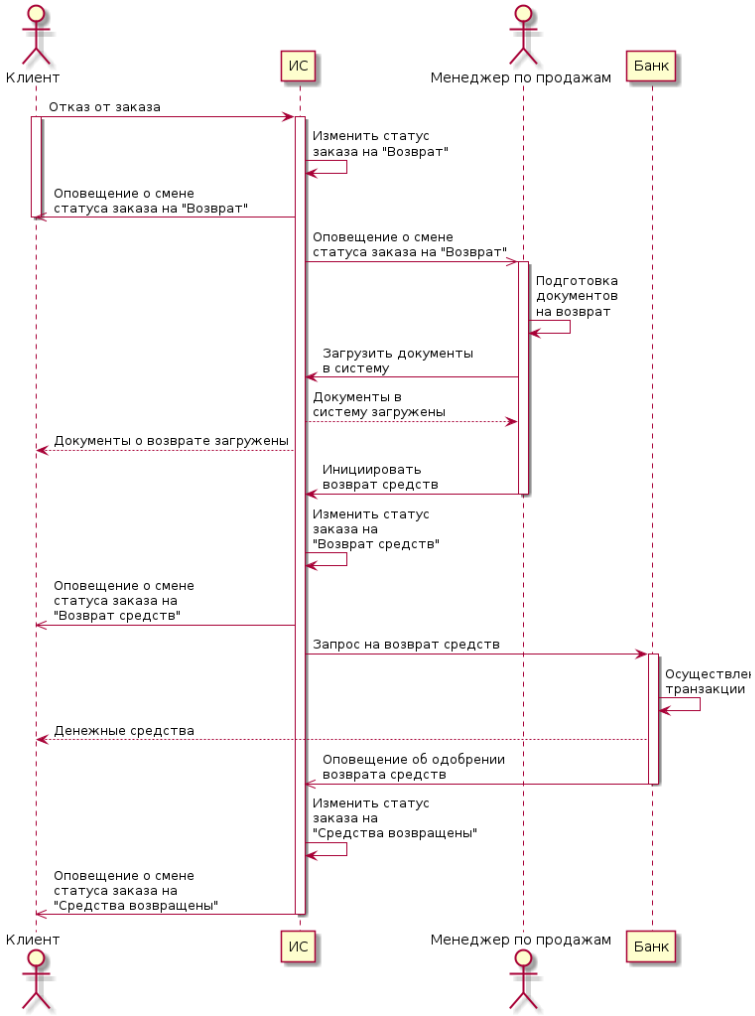
2. На диаграмме последовательности между классами следует отобразить ассоциации, перенесенные с диаграммы классов анализа, а также добавить ассоциации, связывающие актеров с граничными классами.

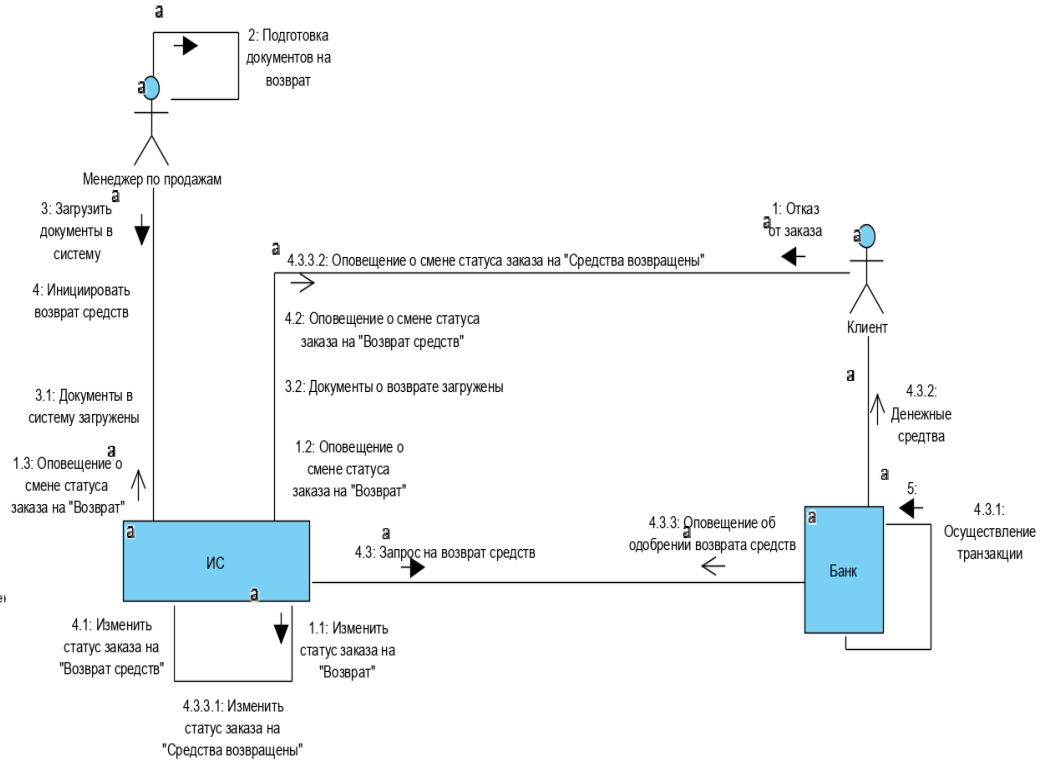
3. На стадии анализа имена сообщениям можно давать произвольно или в виде стереотипов. В дальнейшем (в модели проектирования) имена сообщений должны соответствовать методам классов.

4. Имена сущностей на диаграммах (экземпляры актеров и объекты) должны быть подчеркнуты и обозначены соответствующим образом.

5. На диаграммах последовательности символ уничтожения объектов следует задавать только для тех объектов, которые во время взаимодействия действительно уничтожаются.

Примеры диаграмм: последовательности и кооперации





Спасибо за внимание

РТУ МИРЭА, кафедра ППИ