UML
УНИФИЦИРОВАННЫЙ ЯЗЫК МОДЕЛИРОВАНИЯ

Самоучитель по UML

http://www.e-reading.club/book.php?book=33640

https://sites.google.com/site/anisimovkhv/learning/pris/lecture

https://msdn.microsoft.com/ru-ru/library/dd409376(v=vs.120).aspx

Основа UML


Meyer
Before and after conditions
Harel
Statecharts
Gamma, et al
Frameworks and patterns,
Booch
Booch method
HP Fusion
Operation descriptions and message numbering
Rumbaugh
OMT
UNIFIED MODELING LANGUAGE
Embley
Singleton classes and high-level view
Jacobson
OOSE
Wirfs-Brock
Responsibilities
Shlaer - Mellor
Object lifecycles
Odell
Classification

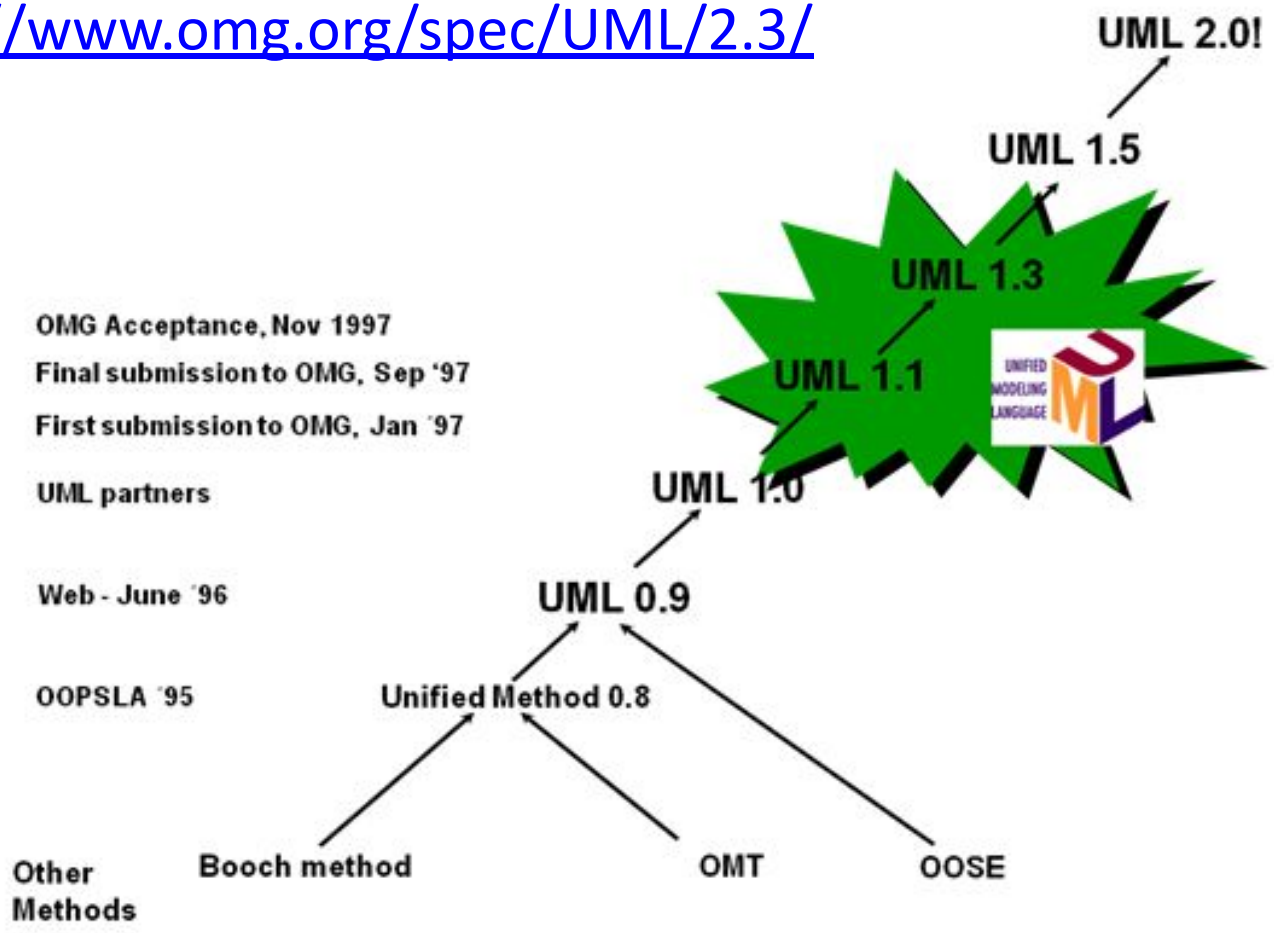
Начало 1995 г.

Grady Booch (Гради Буч)

Jim Rumbaugh (Джим Рамбо) Ivar Jacobson (Ивар Якобсон)

История развития

http://www.omg.org/spec/UML/2.3/



UML

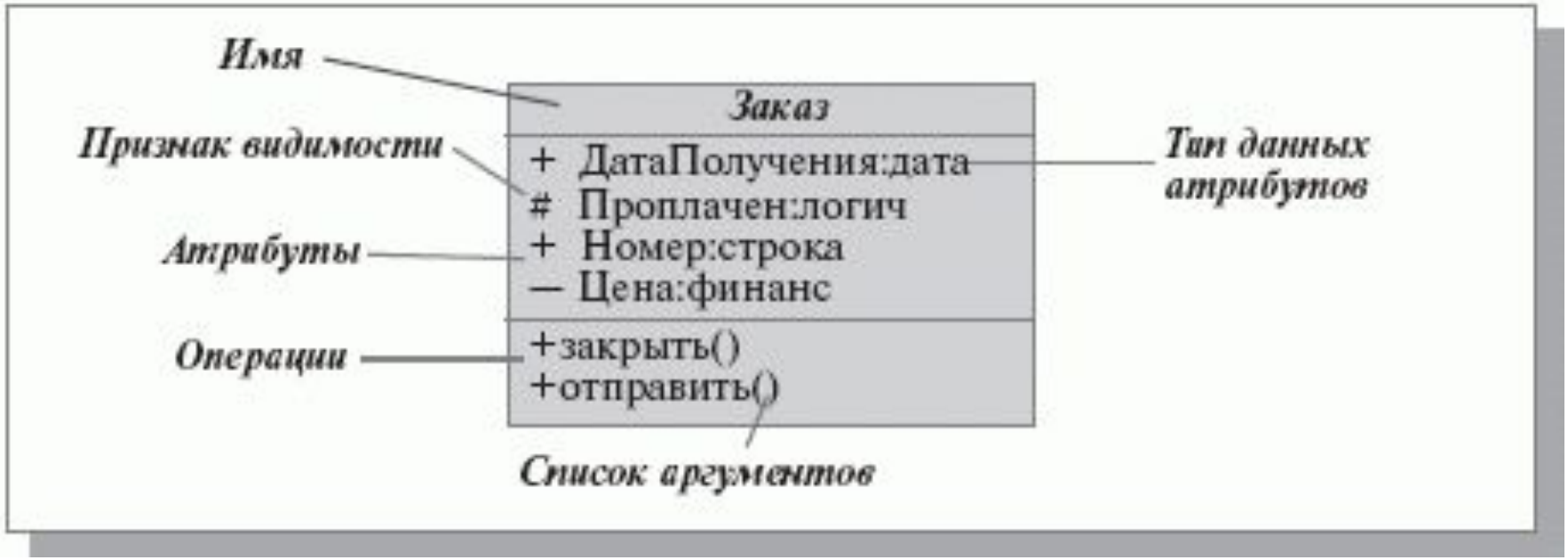
UML представляет собой *объектно-ориентированный язык моделирования*, обладающий следующими основными характеристиками:

- является языком *визуального моделирования*, который обеспечивает разработку *репрезентативных* моделей для организации взаимодействия заказчика и разработчика ИС, различных групп разработчиков ИС;
- содержит механизмы расширения и специализации базовых концепций языка.

Процессе объектно-ориентированного анализа и проектирования

Канонические диаграммы UML

Понятие класс



Синтаксис UML для классов

< признак видимости> <имя атрибута> :
<тип данных> = <значение по умолчанию>

<признак видимости> <имя операции>
<(список аргументов)>

Уровни видимости

- public (общий) — любой внешний *класс*, который "видит" данный, может пользоваться его общими свойствами. Обозначаются знаком " + " перед именем атрибута или операции;
- protected (защищенный) — только любой потомок данного *класса* может пользоваться его защищенными свойствами. Обозначаются знаком " # ";
- private (закрытый) — только данный *класс* может пользоваться этими свойствами. Обозначаются символом " - " .

Области действия

- instance (экземпляр) — у каждого экземпляра *класса* есть собственное значение данного свойства;
- *classifier* (классификатор) — все экземпляры совместно используют общее значение данного свойства (выделяется на диаграммах подчеркиванием).

Кратность классов

- не содержащие ни одного экземпляра — тогда *класс* становится служебным (Abstract);
- содержащие ровно один экземпляр (Singleton);
- содержащие заданное число экземпляров;
- содержащие произвольное число экземпляров.

ДИАГРАММЫ ПРЕЦЕДЕНТОВ

Диаграммы прецедентов

Диаграммы прецедентов (*диаграммы вариантов использования, use case* diagrams) – это обобщенная модель функционирования системы в окружающей среде.

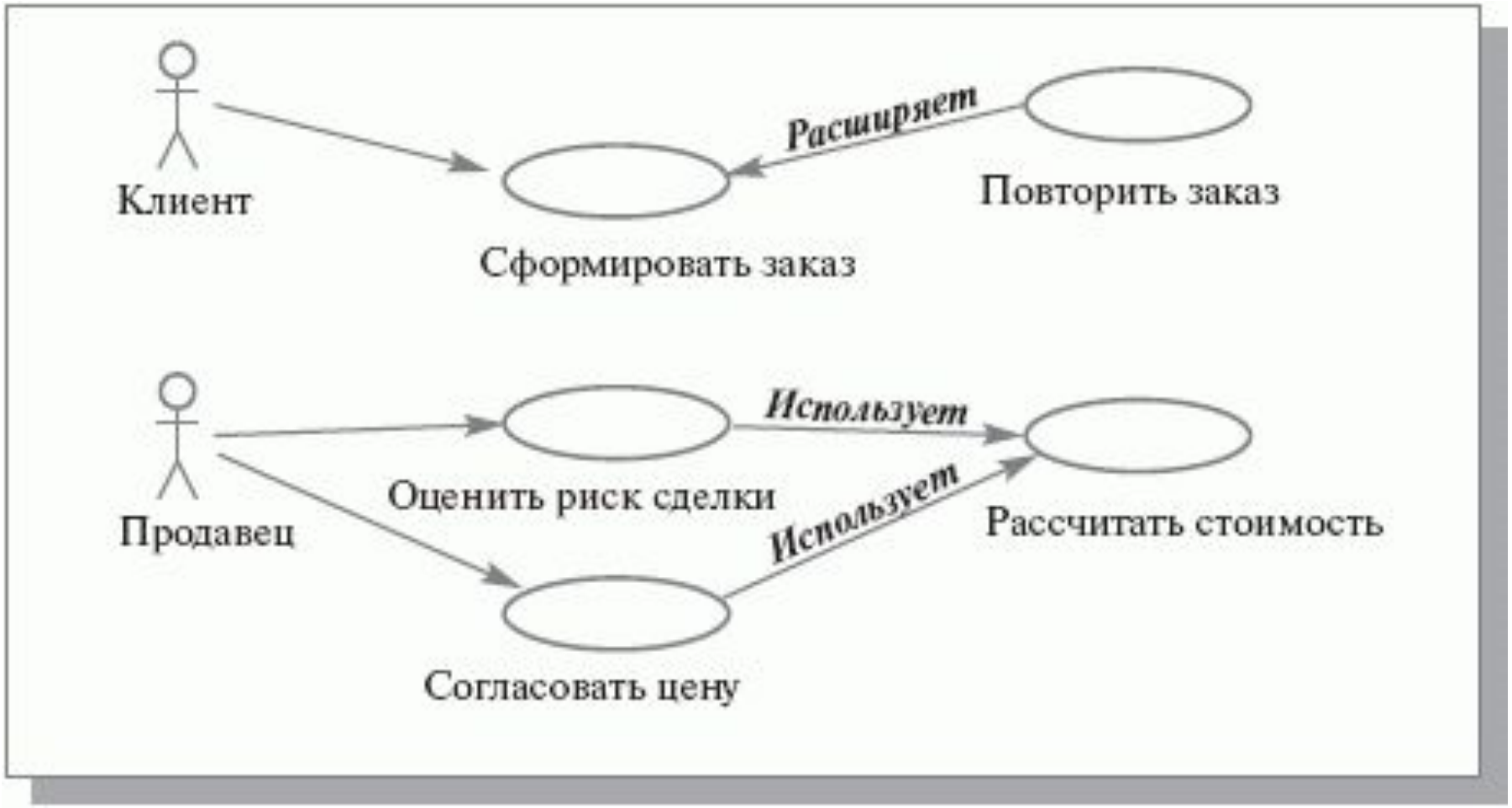
Прецеденты и действующие лица

***Прецедент* (use case)** — это типичное взаимодействие пользователя с системой, которое при этом:

- описывает видимую пользователем функцию,
- может представлять различные уровни детализации,
- обеспечивает достижение конкретной цели, важной для пользователя.

***Действующие лица* (актеры, actors)** используют систему (или используются системой) в данном прецеденте.

Диаграмма прецедентов



Связи типа «расширение» и «использование»

Связь типа "расширение" применяется, когда один *прецедент* подобен другому, но несет несколько большую функциональную нагрузку. Ее следует применять при описании изменений в нормальном поведении системы.

Связь типа "использование" позволяет выделить некий фрагмент поведения системы и включать его в различные прецеденты без повторного описания.

ДИАГРАММА КЛАССОВ

Диаграммы классов

Диаграммы классов (*class diagrams*)
– *логическая модель* базовой структуры системы, отражает статическую структуру системы и связи между ее элементами.


Клиент
:имя
:адрес
Кредитный рейтинг()
Ассоциация
Заказ
:ДатаПолучения
:Проплачен
:Номер
:Цена
закрыть
отправить
Обобщение
Корпоративный клиент
:контактное лицо
:кредитный рейтинг
:кредитный лимит
счет
Частный клиент
:номер кредитной карты
Ассоциация
(агрегация)
Строка заказа
:наименование
:количество
:цена
Зависимость
Товар
:наименование
:цвет
:размер

***Класс* (class)** — абстрактное описание множества однородных объектов, имеющих одинаковые *атрибуты, операции* и отношения с объектами других *классов* .


Окружность
center : Point
radius : Integer
Окно
отобразить()
скрыть()
Счет
открытьСчет()
реализовать
резервное
копирование

(а) (б) (в)

Атрибуты

<квантор видимости> <имя атрибута> [кратность] :

<тип атрибута> =

<исходное значение>

{строка-свойство}

Операции

<квантор видимости> <имя операции>

(список параметров):

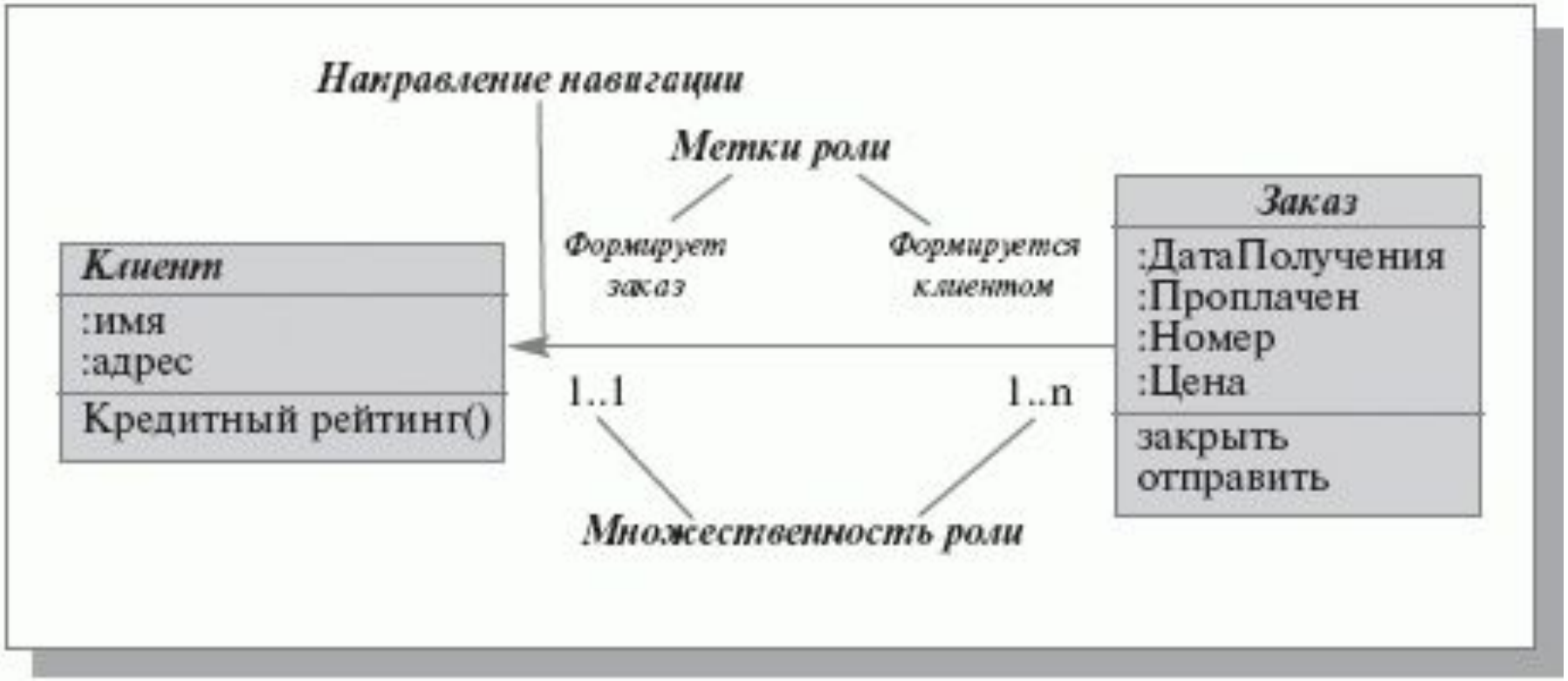
<выражение типа возвращаемого значения>

{строка-свойство}

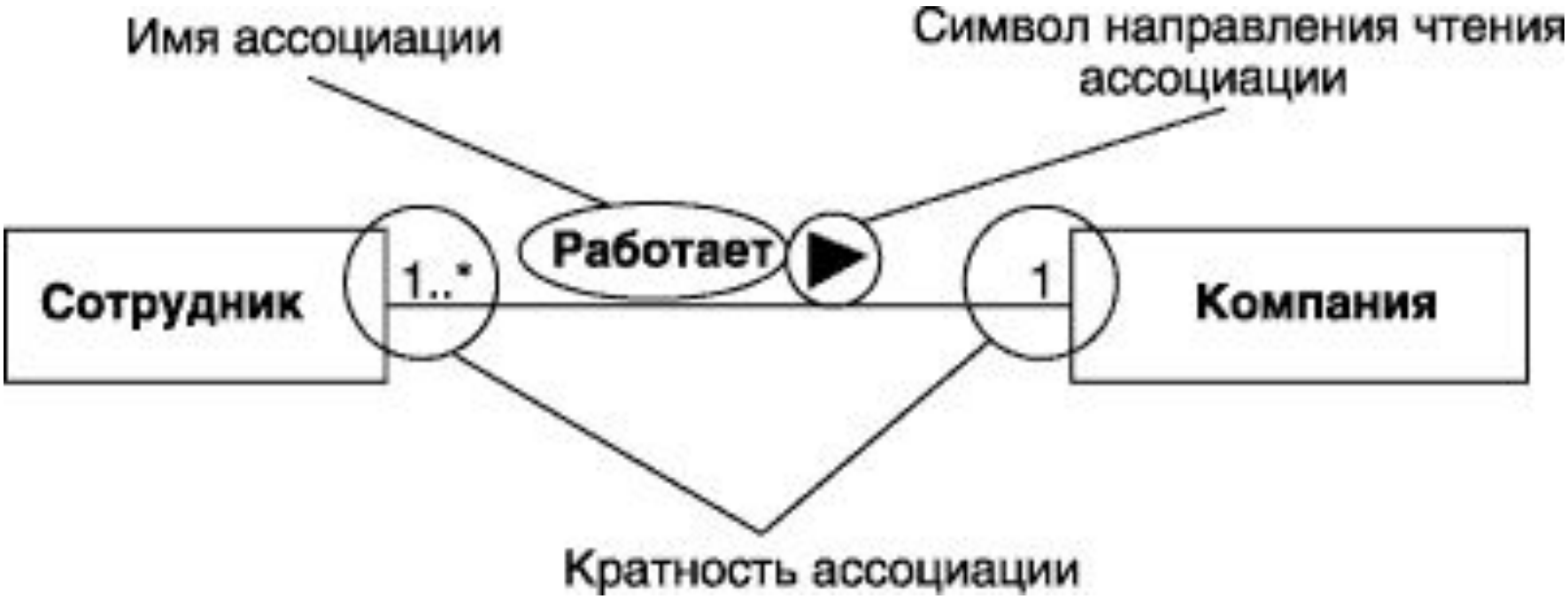
Отношения между классами

- Зависимостью называется отношение использования, согласно которому изменение в спецификации одного элемента (например,*класса* " товар ") может повлиять на использующий его элемент (*класс* " строка заказа "). Часто зависимости показывают, что один *класс* использует другой в качестве аргумента.
- Обобщение — это отношение между общей сущностью (родителем — *класс* " клиент ") и ее конкретным воплощением (потомком — *классы* " корпоративный клиент " или " частный клиент ").
- Ассоциация — это отношение, показывающее, что объекты одного типа неким образом связаны с объектами другого типа ("клиент " может сделать " заказ ").
- Если приходится моделировать отношение типа "часть-целое", то используется специальный тип ассоциации — агрегирование. В такой ассоциации один из *классов* имеет более высокий ранг (целое — *класс* " заказ ") и состоит из нескольких меньших по рангу *классов* (частей — *класс* " строка заказа ").

Свойства ассоциации

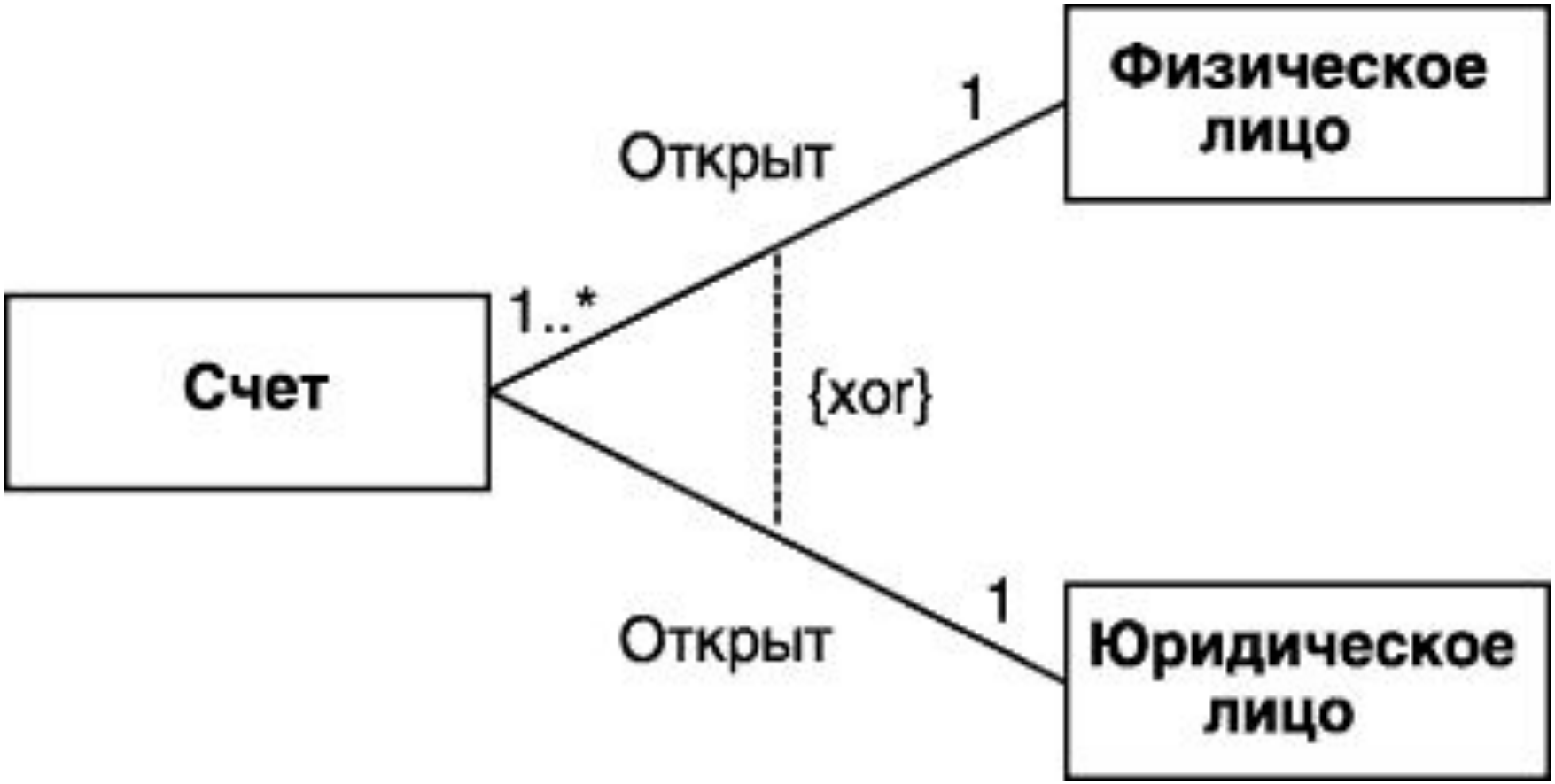


Ненаправленная бинарная ассоциация



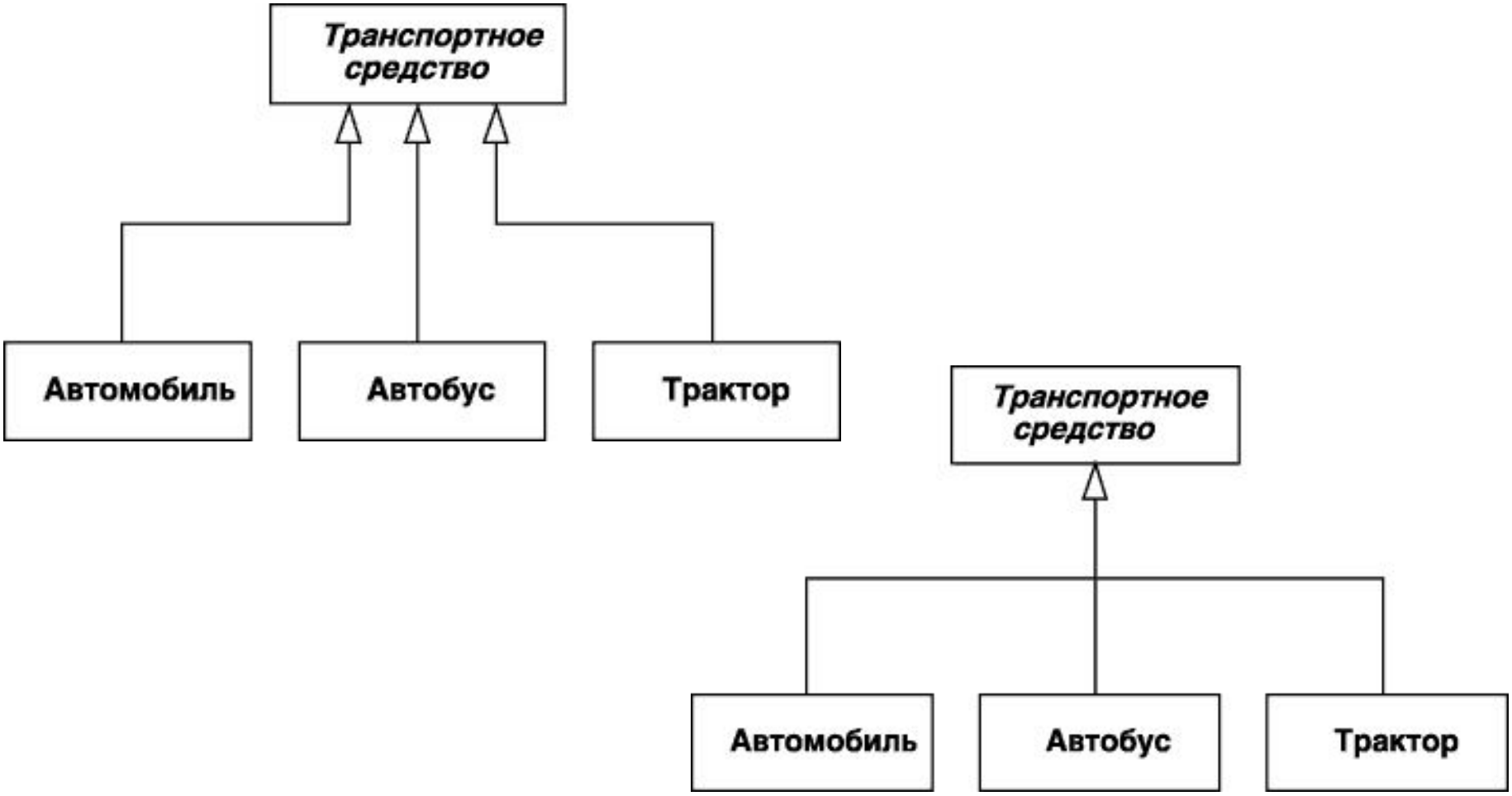
Направленная бинарная ассоциация

Исключающая ассоциация



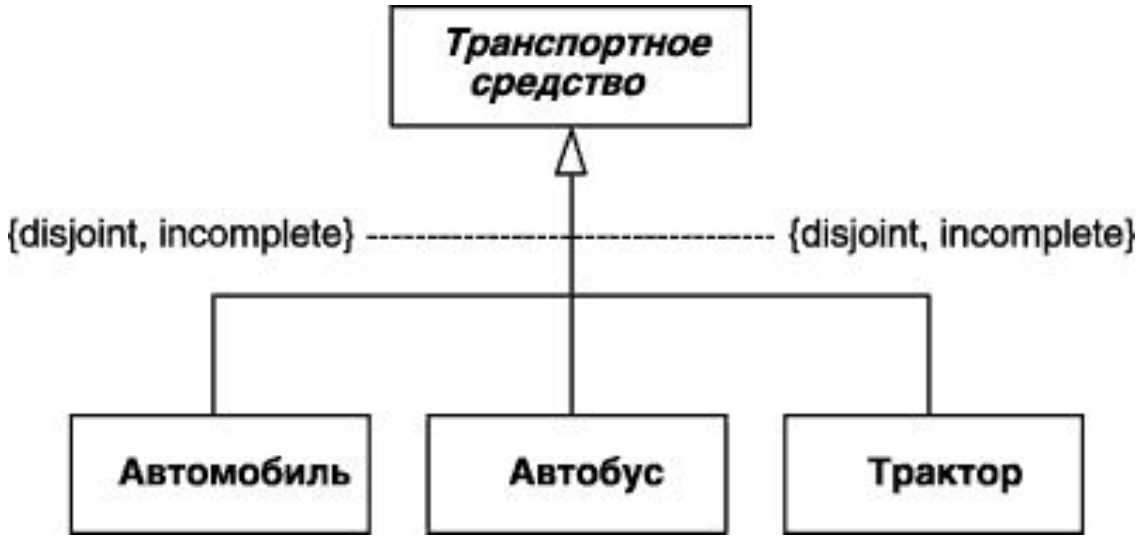
n-арная ассоциация

Отношение обобщения



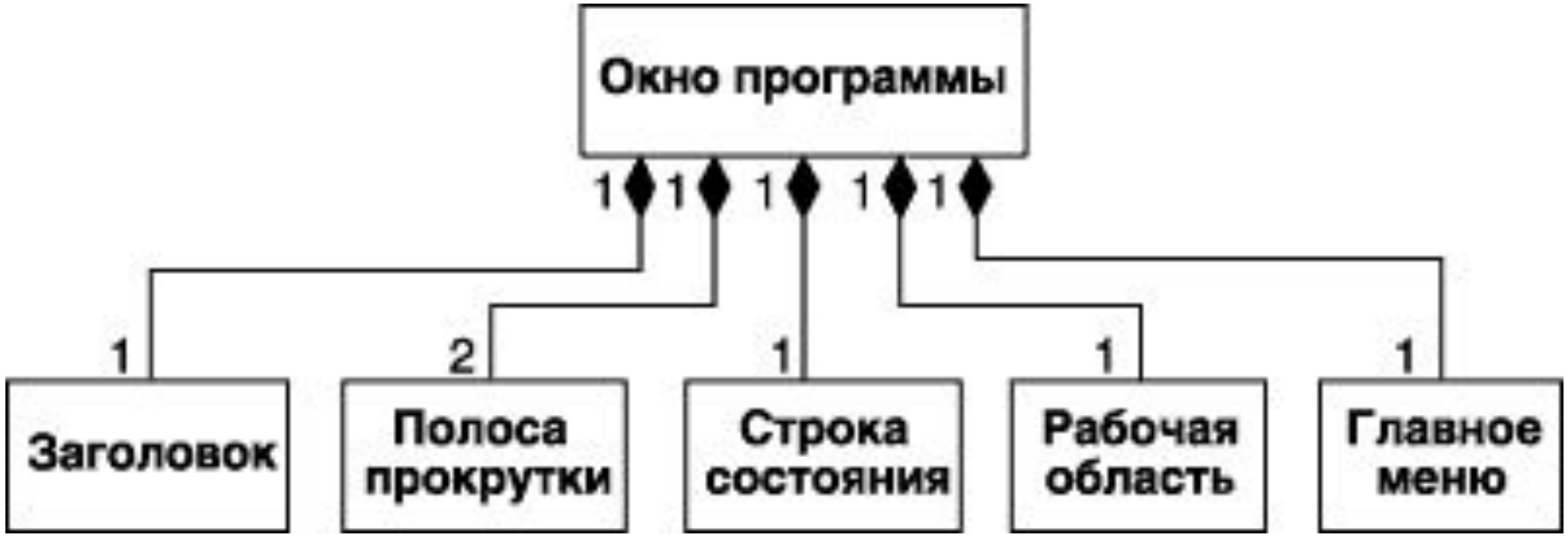
Ограничения отношения обощения

- {complete}
- {*incomplete*}
- {disjoint}
- {*overlapping*}



Отношения агрегации

Отношения композиции



ДИАГРАММЫ ВЗАИМОДЕЙСТВИЯ

Диаграммы взаимодействия

Диаграммы взаимодействия (*interaction* diagrams) – модель процесса обмена сообщениями между объектами, представляется в виде диаграмм последовательностей (*sequence diagrams*) или кооперативных диаграмм (*collaboration* diagrams).

Кооперация

Кооперация (collaboration) — спецификация множества *объектов* отдельных классов, совместно взаимодействующих с целью реализации отдельных вариантов использования в общем контексте моделируемой системы.

Элементы диаграмм кооперации

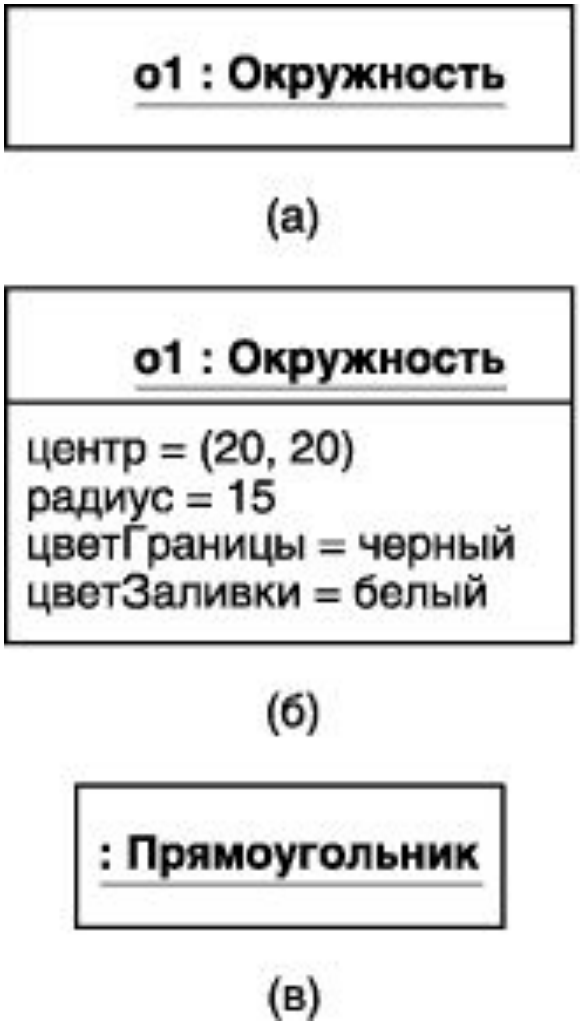
1. Объекты
2. Связи
3. Сообщения

Объект

Объект (object) — сущность с хорошо определенными границами и индивидуальностью, которая инкапсулирует состояние и поведение.

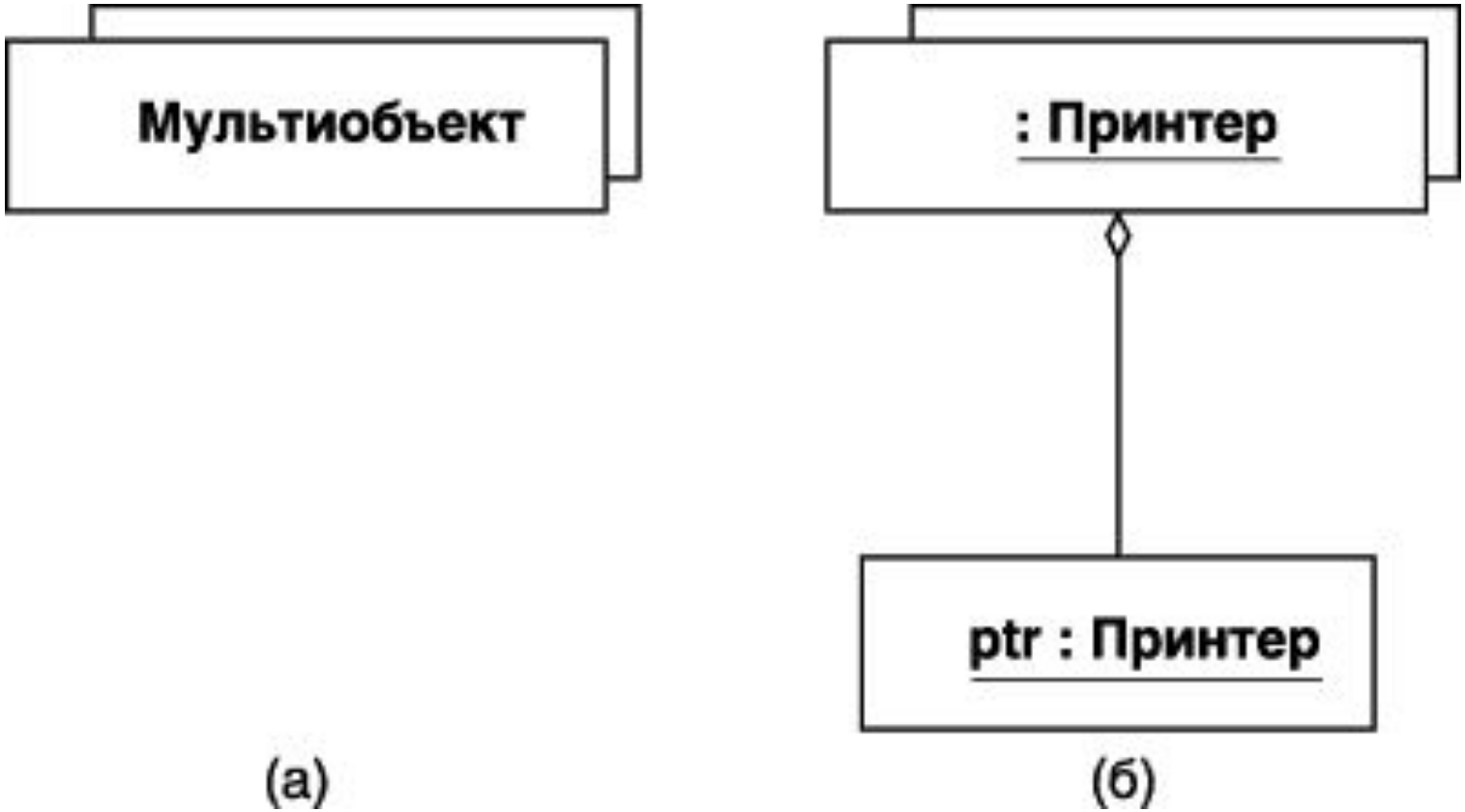
<собственное имя объекта >'/'<Имя роли класса>:<Имя класса >

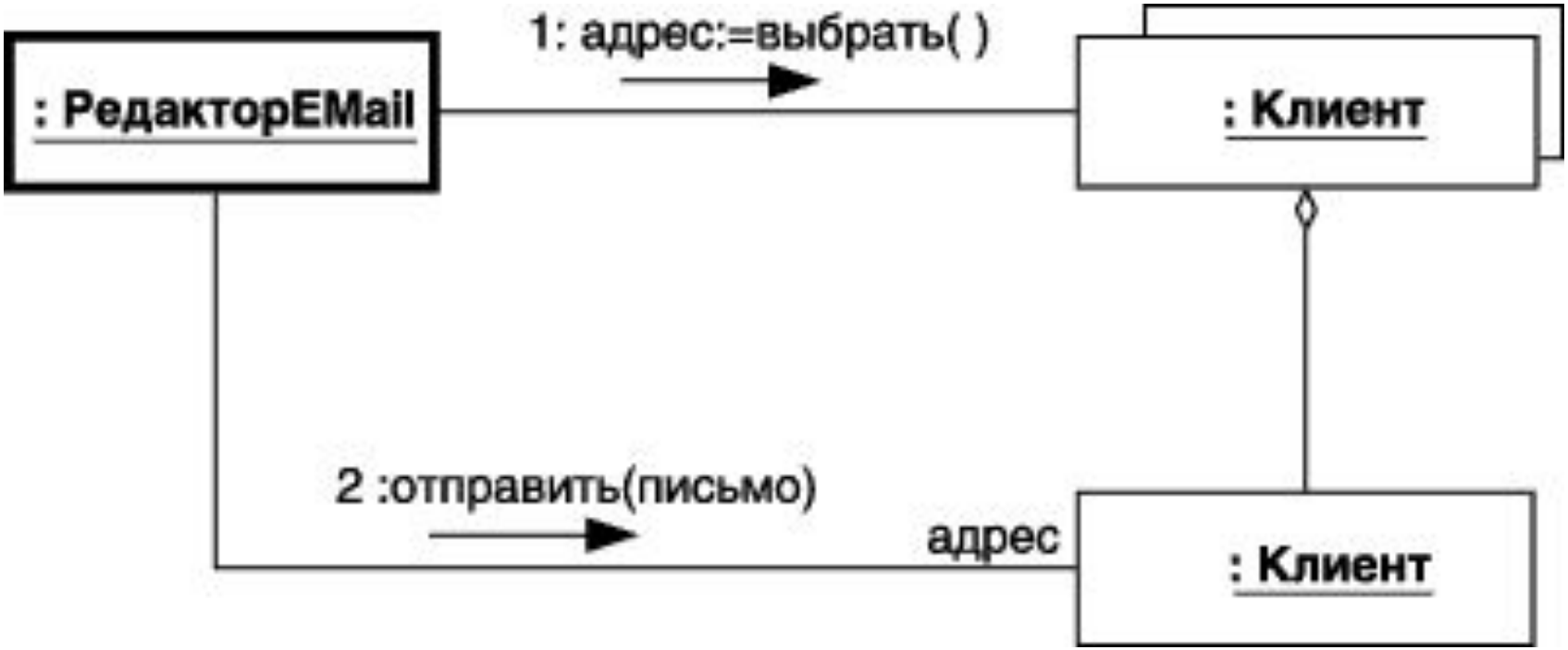
Объекты: анонимные, сироты



Пассивный и активный объект

Мультиобъекты




: РедакторEMail
1: адрес:=выбрать( )
: Клиент
2 :отправить(письмо)
адрес
: Клиент

Составной объект или объект-композит

Связь

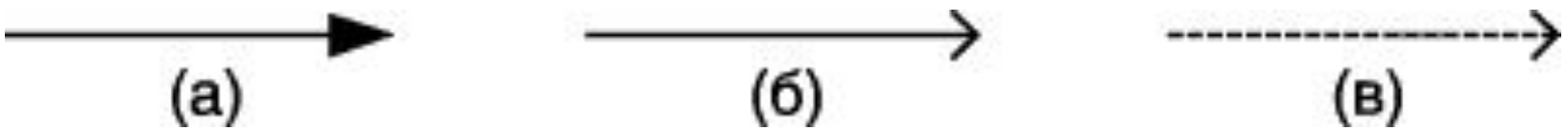
Связь (link) — любое семантическое отношение между некоторой совокупностью *объекто в* .

Стереотипы связей

- «association» – ассоциация (предполагается по умолчанию, поэтому этот стереотип можно не указывать).
- «parameter» – параметр метода. Соответствующий объект может быть только параметром некоторого метода.
- «local» – локальная переменная метода. Ее область видимости ограничена только соседним объектом.
- «global» – глобальная переменная. Ее область видимости распространяется на всю диаграмму кооперации.
- «self» – рефлексивная связь объекта с самим собой, которая допускает передачу объектом сообщения самому себе. На диаграмме кооперации рефлексивная связь изображается петлей в верхней части прямоугольника объекта.

Сообщения

Сообщение (message) — спецификация передачи информации от одного элемента модели к другому с ожиданием выполнения определенных действий со стороны принимающего элемента.



Сообщения

<Предшествующие сообщения/>

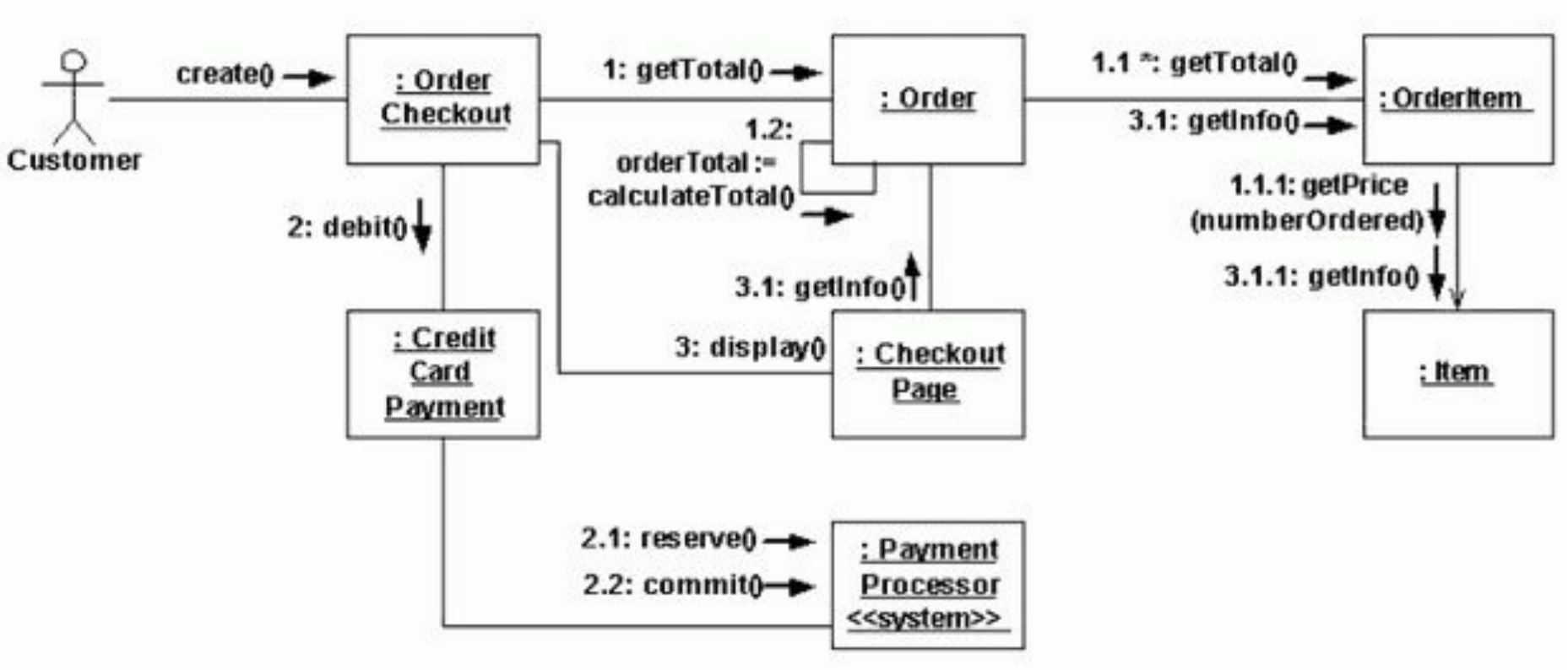
<Выражение последовательности>

<Возвращаемое значение := имя сообщения> <(Список аргументов)>

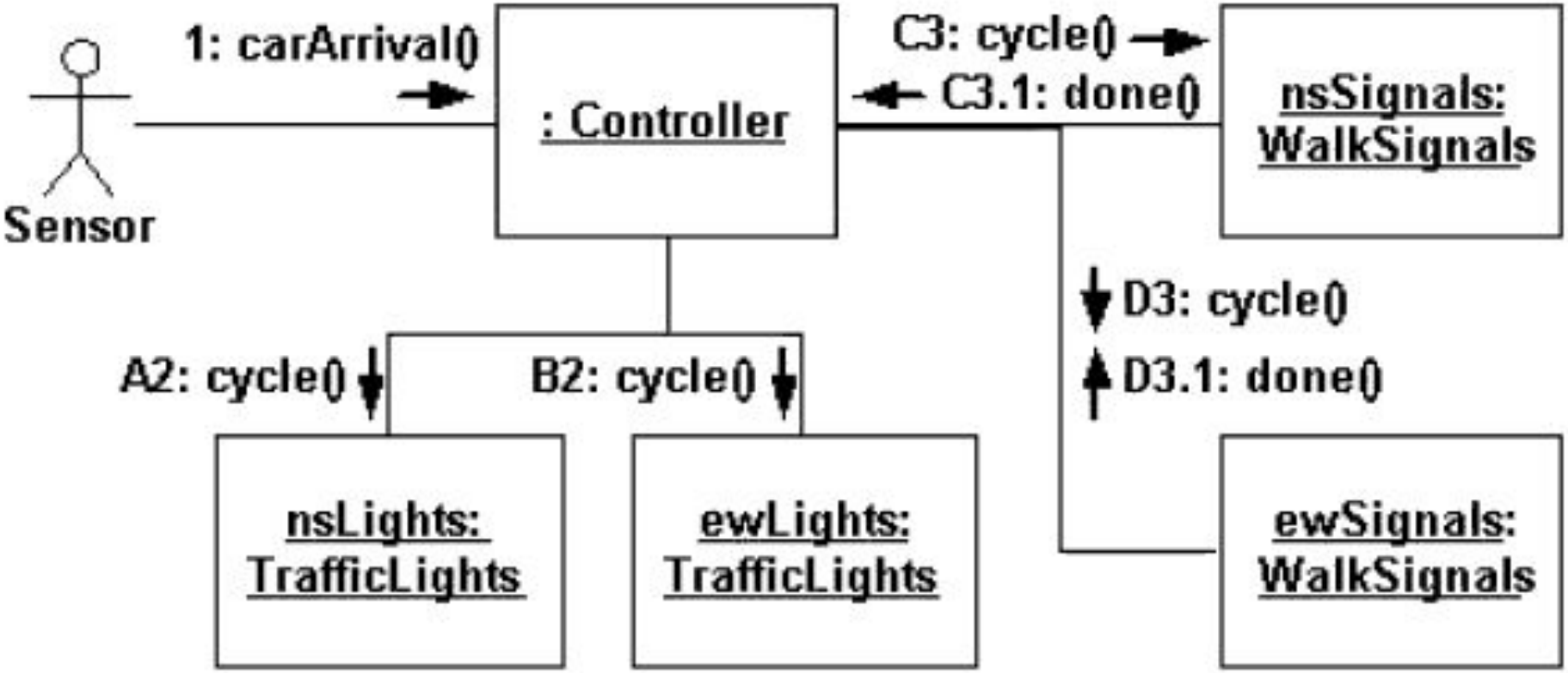
Стереотипы сообщений

- <<call>> (вызвать) – *сообщение*, требующее вызова операции или процедуры объекта-получателя. Если *сообщение* с этим стереотипом рефлексивное, то оно инициирует локальный вызов операции у пославшего это *сообщение объекта*.
- <<return>> (возвратить) – *сообщение*, возвращающее значение выполненной операции или процедуры вызвавшему ее *объекту*. Значение результата может инициировать ветвление потока управления.
- <<create>> (создать) – *сообщение*, требующее создания другого *объекта* для выполнения определенных действий. Созданный *объект* может стать активным (ему передается поток управления), а может остаться пассивным.
- <<destroy>> (уничтожить) – *сообщение* с явным требованием уничтожить соответствующий *объект*. Посылается в том случае, когда необходимо прекратить нежелательные действия со стороны существующего в системе *объекта*, либо когда *объект* больше не нужен и должен освободить задействованные им системные ресурсы.
- <<send>> (послать) – обозначает посылку другому *объекту* сигнала, который асинхронно инициируется одним *объектом* и принимается (перехватывается) другим. Отличие сигнала от *сообщения* заключа ется в том, что сигнал должен быть явно описан в том классе, *объект* которого инициирует его передачу.

Кооперативные диаграммы



Кооперативные диаграммы



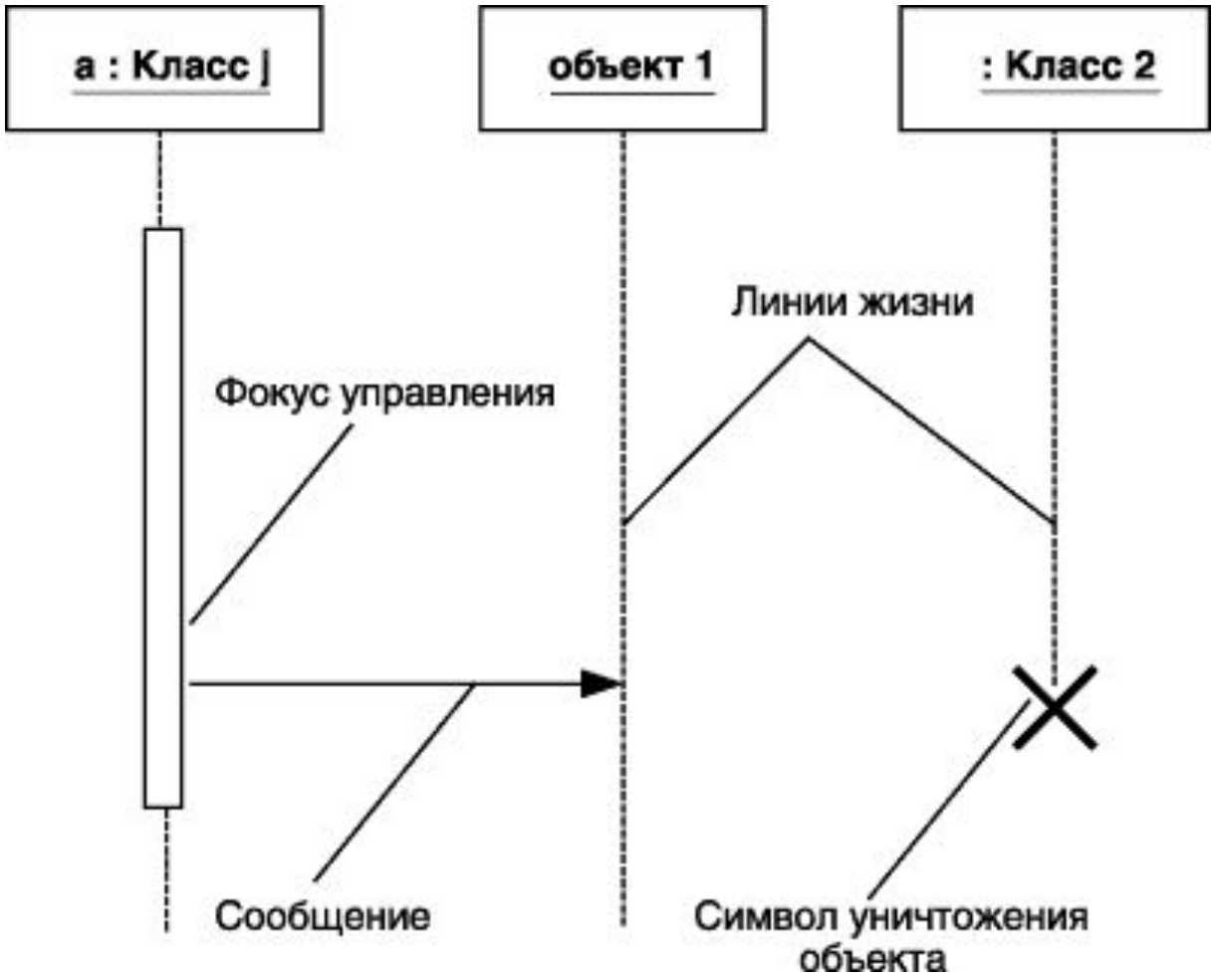
Диаграммы последовательностей

Диаграмма последовательности (sequence diagram) - диаграмма, на которой показаны взаимодействия объектов, упорядоченные по времени их проявления.

Элементы диаграмм последовательностей:

1. Объекты
2. Линии жизни
3. Фокус управления
4. Сообщения

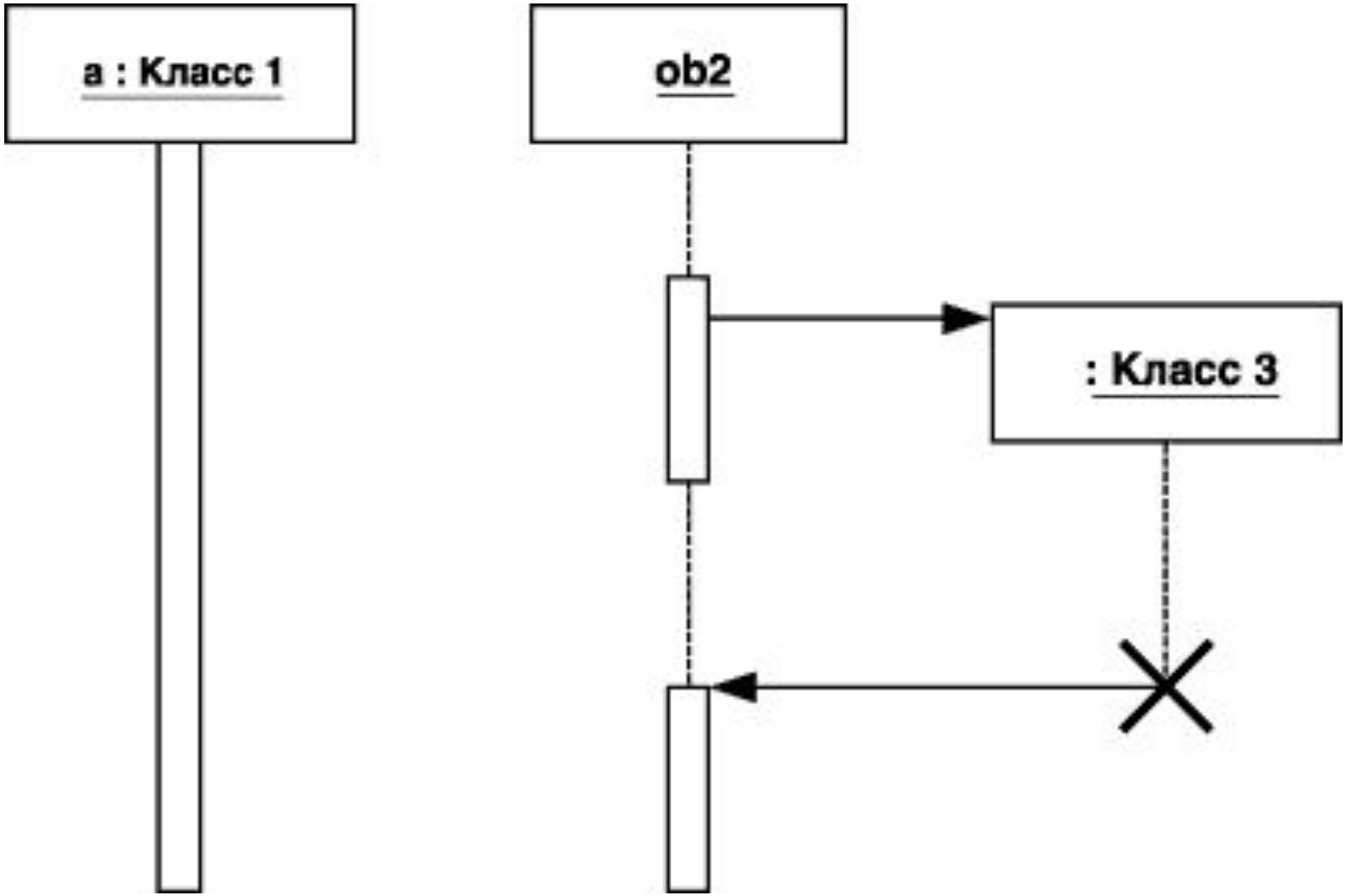
Элементы диаграмм последовательностей:



Линии жизни

Линия жизни объекта (object lifeline) - вертикальная линия на диаграмме последовательности, которая представляет существование объекта в течение определенного периода времени.

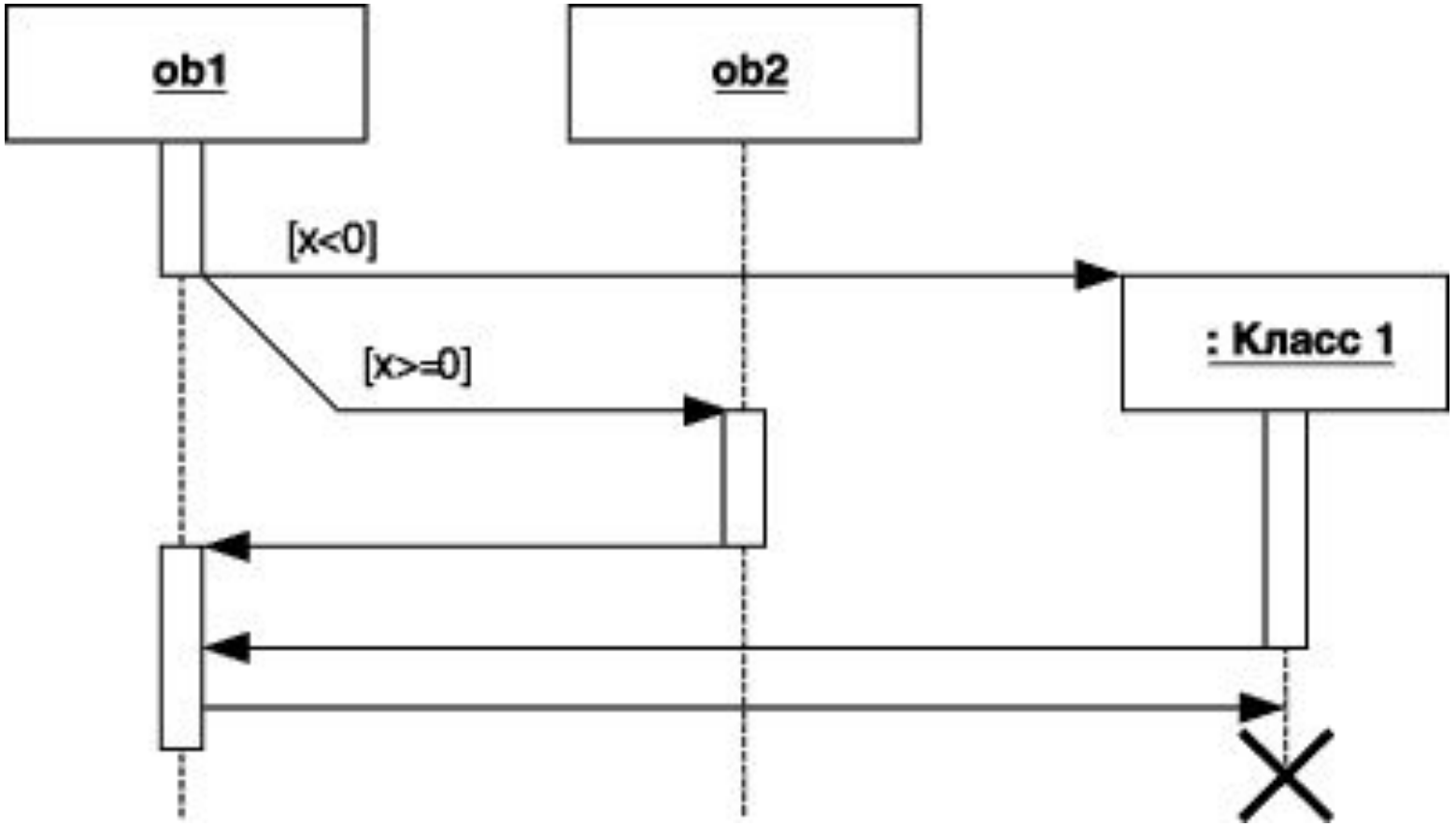
Фокус управления (focus of control) - специальный символ на диаграмме последовательности, указывающий период времени, в течение которого объект выполняет некоторое действие, находясь в активном состоянии.


a : Класс 1
ob2
: Класс 3

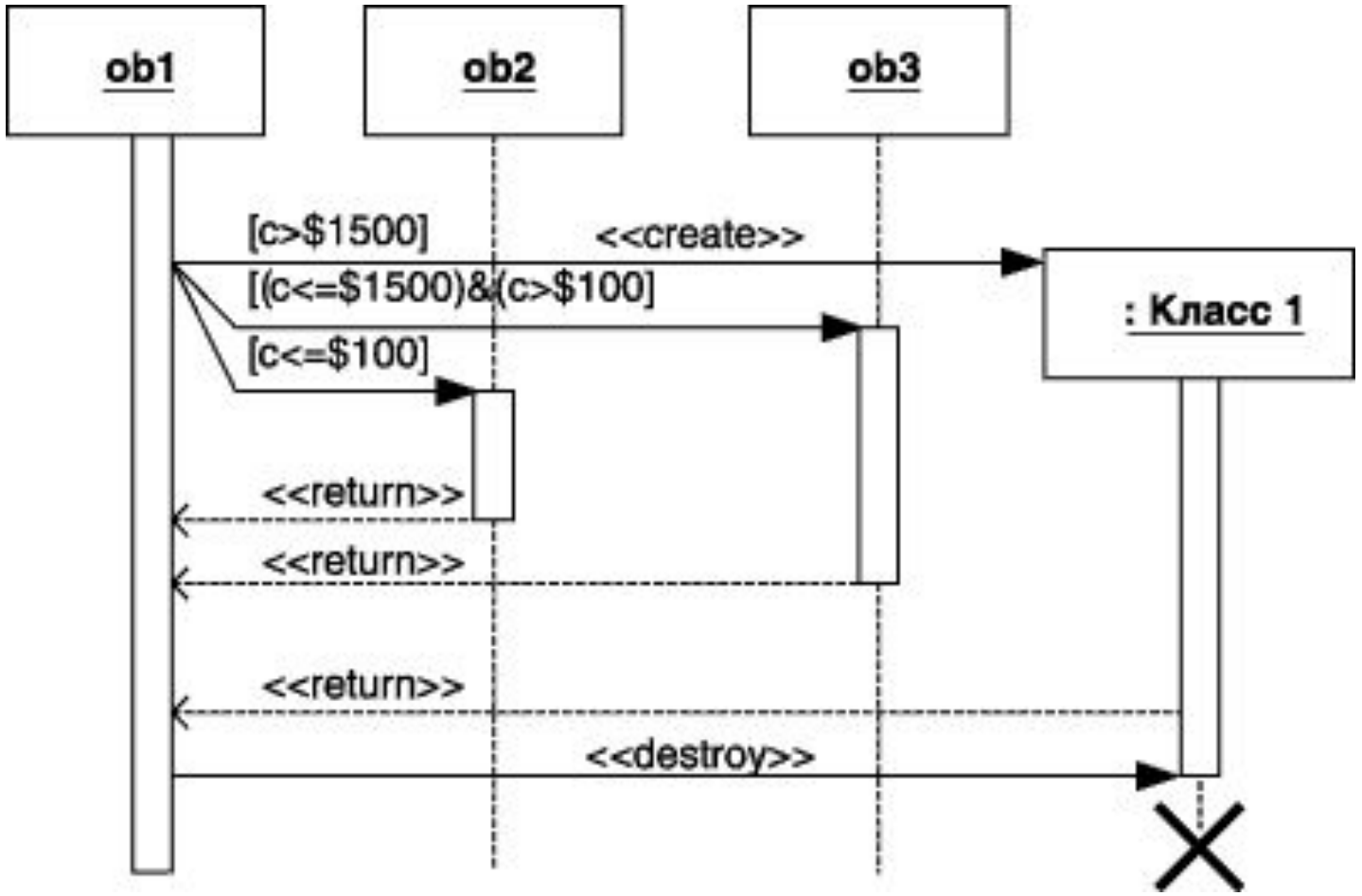
Сообщения


: посетитель Web-сайта
Анонимный актер
a : Класс 1
Рефлексивное сообщение
: Класс 2
Рекурсия

Ветвление потока управления



Ветвление потока управления



Использование стереотипов

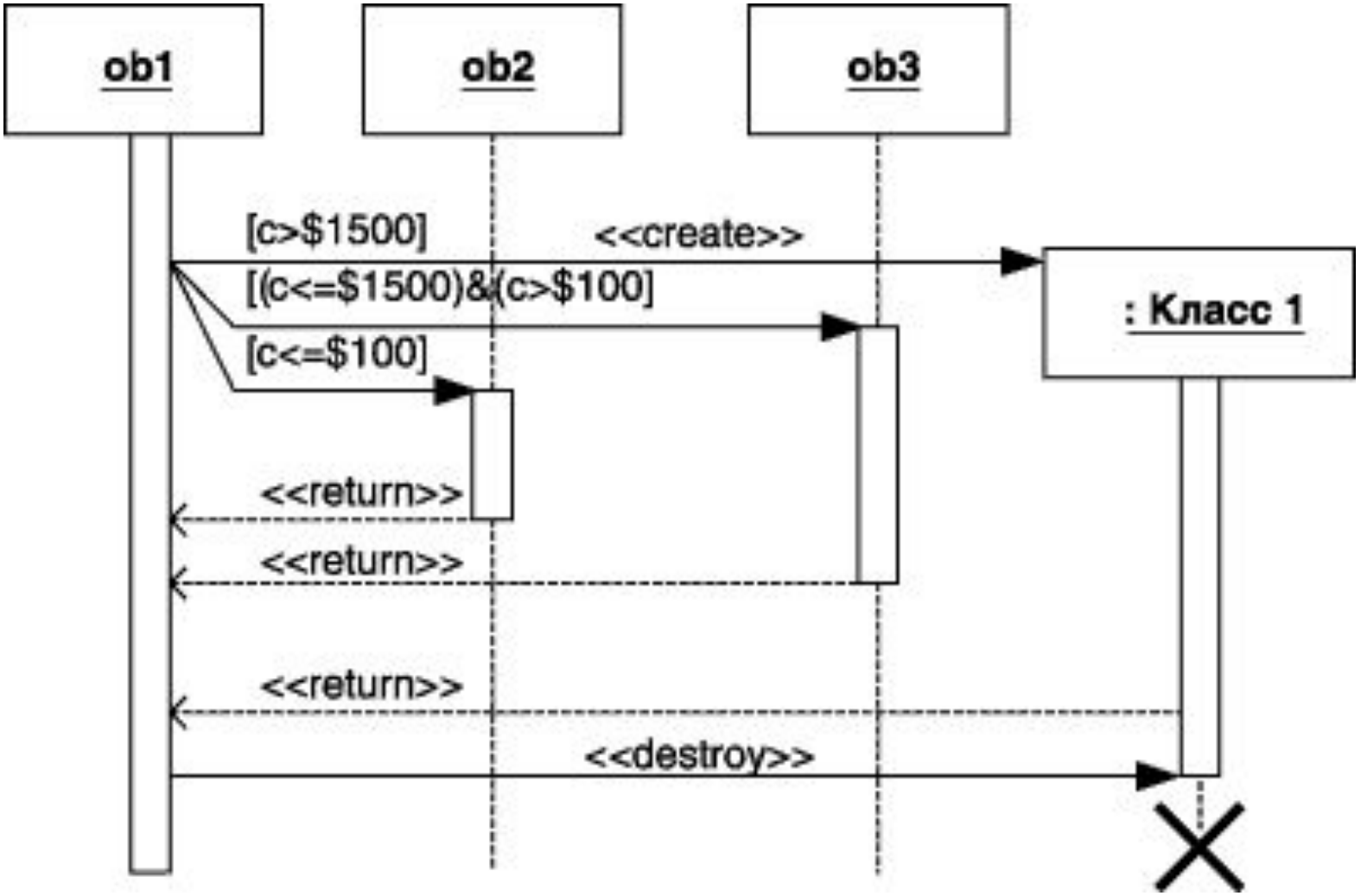
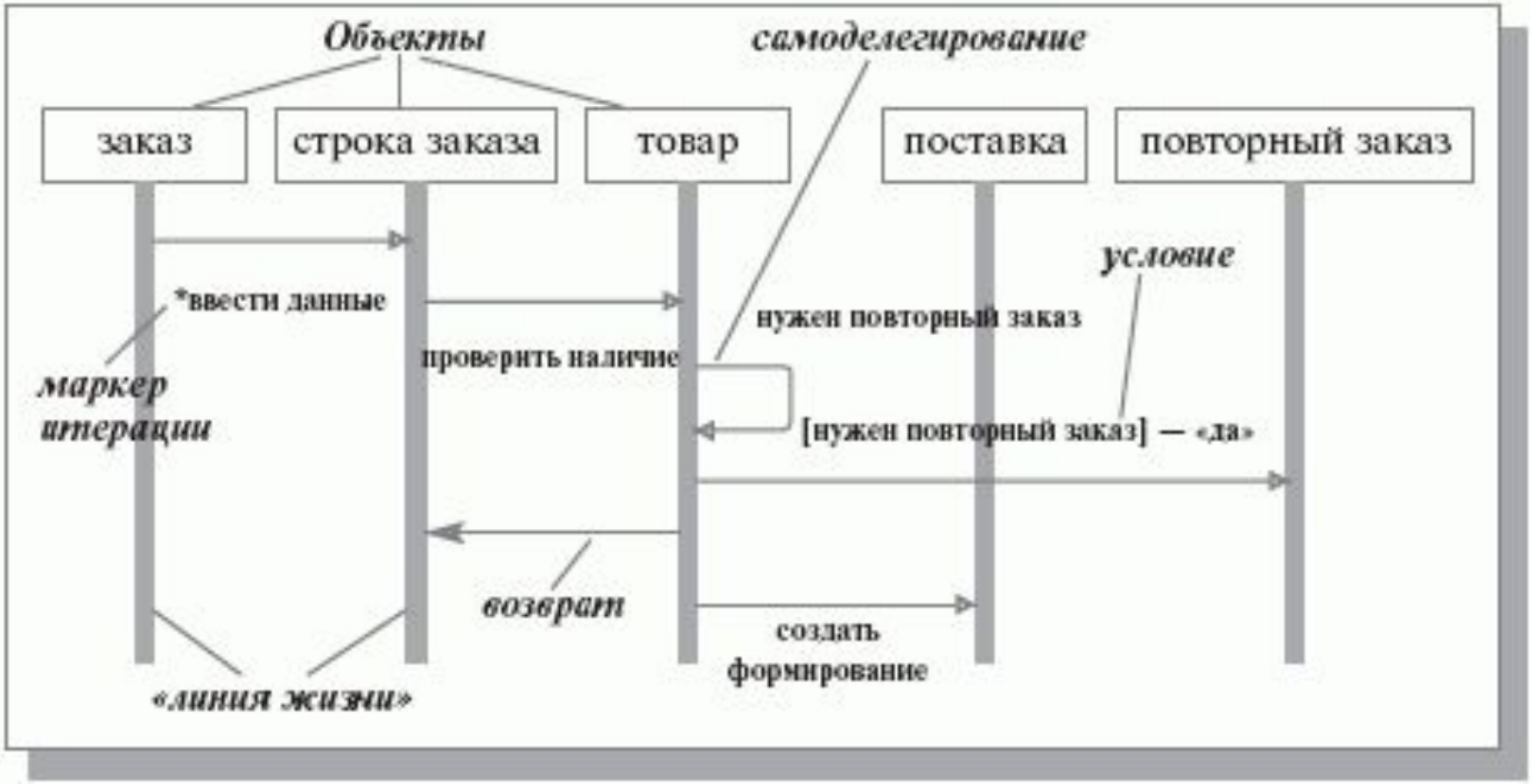


Диаграмма последовательности



ДИАГРАММЫ СОСТОЯНИЙ

Диаграммы состояний

Диаграммы состояний (statechart diagrams) – модель динамического поведения системы и ее компонентов при переходе из одного состояния в другое.

Конечный автомат

Модель для спецификации поведения объекта в форме последовательности его состояний, которые описывают реакцию объекта на внешние события, выполнение объектом действий, а также изменение его отдельных свойств.

Элементы диаграммы состояний

- Состояния
- Переходы


Действие
/получить следующую строку
[не все строки проверены]
/получить
первую строку
проверка строки заказа
выполнить/проверить
строку
Условие
Начало
Поступил
заказ
[Все строки проверены и все
позиции доступны]
[Все строки проверены
и некоторые позиции
отсутствуют на складе]
Функция объекта
Формирование поставки
выполнить/инициировать
поставку
Событие
Позиция получена
[некоторые позиции
отсутствуют на
складе]
Позиция получена
[все позиции
доступны]
Поставка
выполнена
ожидание
Конец
Заказ
поставлен

Состояние

Состояние (state) - условие или ситуация в ходе жизненного цикла объекта, в течение которого он удовлетворяет логическому условию, выполняет определенную деятельность или ожидает события.

Состояния


Имя состояния
Имя состояния
список внутренних действий в данном состоянии
(а)
(б)

Действие

Действие (action) - спецификация выполнимого утверждения, которая образует абстракцию вычислительной процедуры.

<метка действия '/ ' выражение действия>

Метки действия

1. **Входное действие (entry action)** - действие, которое выполняется в момент перехода в данное состояние. Обозначается с помощью ключевого слова - entry.
2. **Действие выхода (exit action)** - действие, производимое при выходе из данного состояния. Обозначается с помощью ключевого слова - метки действия exit.
3. **Внутренняя деятельность (do activity) -** выполнение объектом операций или процедур, которые требуют определенного времени. Обозначается с помощью ключевого слова - метки деятельности do.

Состояние с внутренними действиями

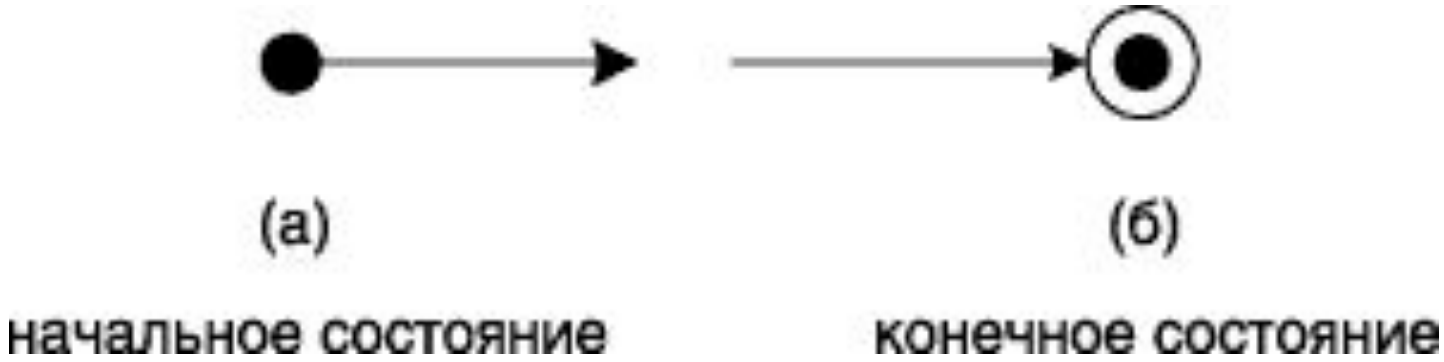
Псевдосостояния

Псевдосостояние (pseudo-state) - вершина в конечном автомате, которая имеет форму состояния, но не обладает поведением.

Начальное состояние (start state) - разновидность псевдосостояния, обозначающее начало выполнения процесса изменения состояний конечного автомата или нахождения моделируемого объекта в составном состоянии.

Конечное состояние (final state) - разновидность псевдосостояния, обозначающее прекращение процесса изменения состояний конечного автомата или нахождения моделируемого объекта в составном состоянии.

Псевдосостояния



Переход

Переход (transition) - отношение между двумя состояниями, которое указывает на то, что объект в первом состоянии (исходном) должен выполнить определенные действия и перейти во второе (целевое) состояние.

Переход

<имя события>'('<список параметров, разделенных запятыми>')'

'['<сторожевое условие>']'

'/'<выражение действия>.

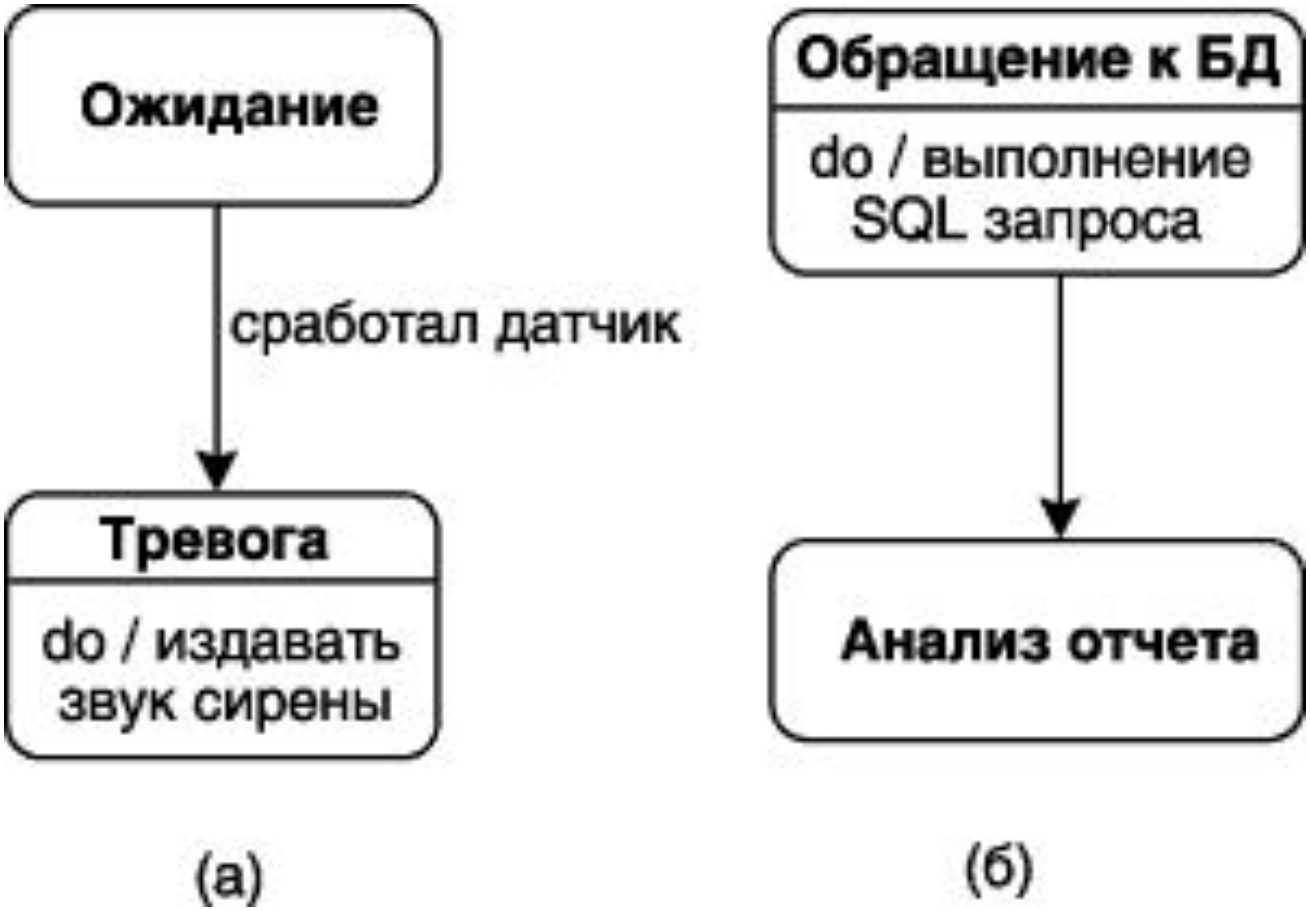
Событие

Событие (event) - спецификация существенных явлений в поведении системы, которые имеют местоположение во времени и пространстве.

Типы переходов

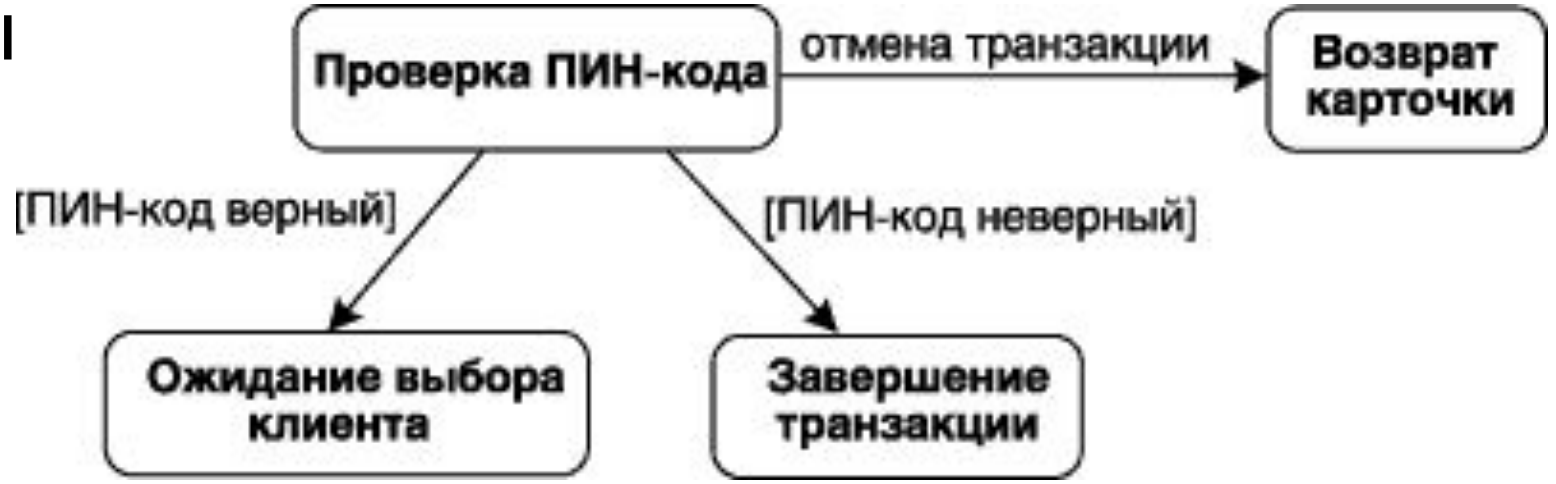
1. Переход называется триггерным, если его специфицирует событие-триггер, связанное с внешними условиями по отношению к рассматриваемому состоянию.
2. Переход называется нетриггерным, если он происходит по завершении выполнения ду-деятельности в данном состоянии.

Триггерный и нетриггерный переходы



Сторожевое условие

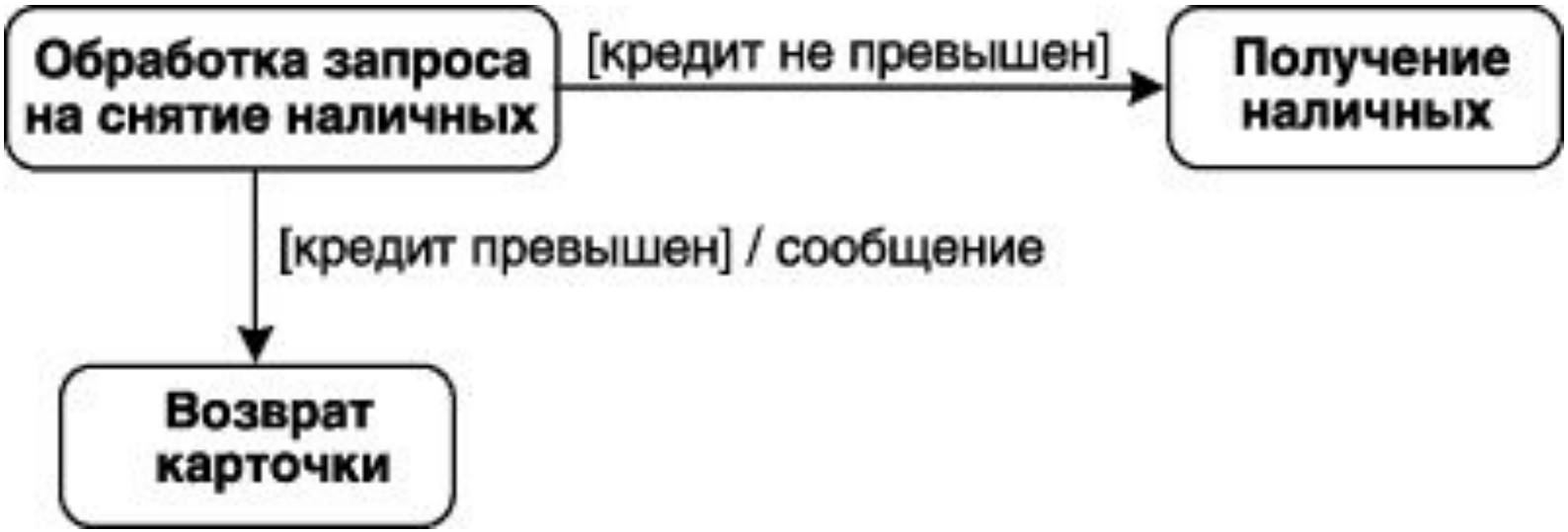
Сторожевое условие (guard condition) - логическое условие, записанное в прямых скобках и представляющее собой булевское вы



Выражение действия

Выражение действия (action expression) представляет собой вызов операции или передачу сообщения, имеет атомарный характер и выполняется сразу после срабатывания соответствующего перехода до начала действий в целевом состоянии.

Выражение действия

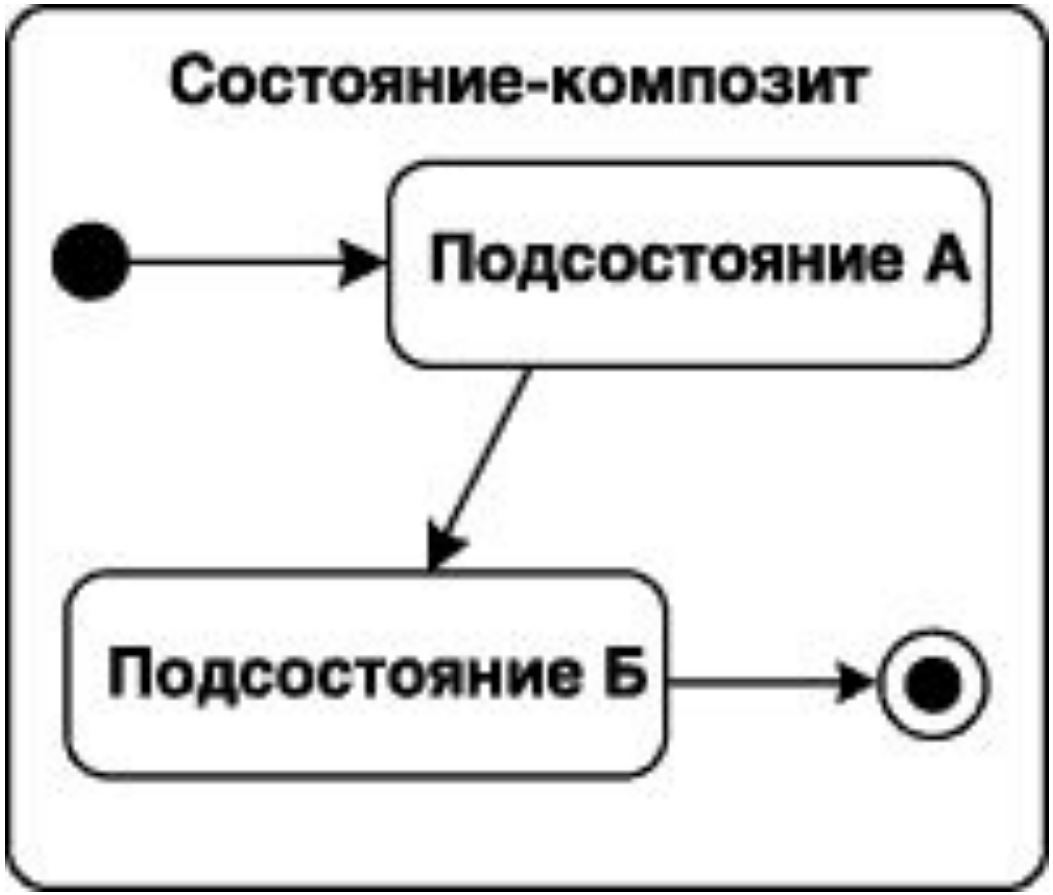



Действие
/получить следующую строку
[не все строки проверены]
/получить
первую строку
проверка строки заказа
выполнить/проверить
строку
Условие
Начало
Поступил
заказ
[Все строки проверены и все
позиции доступны]
[Все строки проверены
и некоторые позиции
отсутствуют на складе]
Функция объекта
Формирование поставки
выполнить/инициировать
поставку
Событие
Позиция получена
[некоторые позиции
отсутствуют на
складе]
Позиция получена
[все позиции
доступны]
Поставка
выполнена
ожидание
Конец
Заказ
поставлен

Составное состояние

Составное состояние, состояние-композит (composite state) - сложное состояние, которое состоит из других вложенных в него состояний.

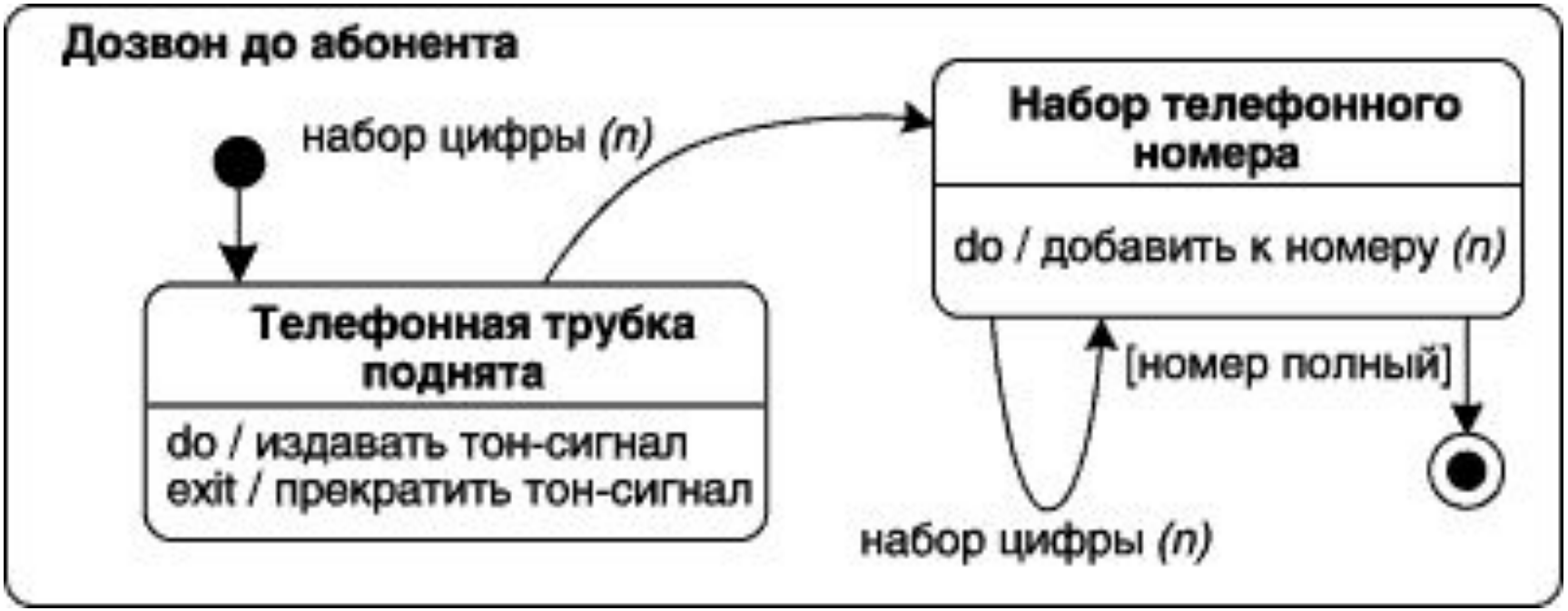
Составное состояние



Последовательные подсостояния

Последовательные подсостояния (sequential substates) - вложенные состояния состояния-композита, в рамках которого в каждый момент времени объект может находиться в одном и только одном подсостоянии.

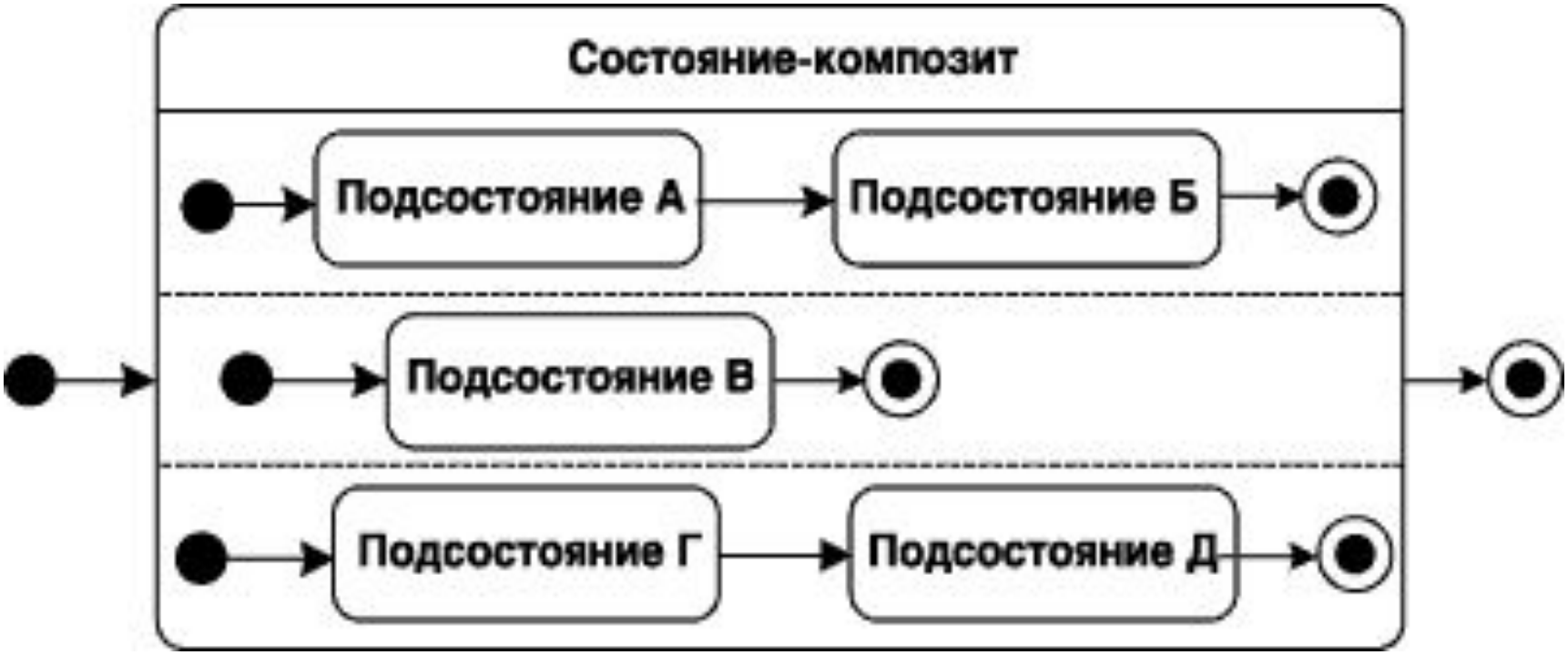
Последовательные подсостояния



Параллельные подсостояния

Параллельные подсостояния (concurrent substates) - вложенные состояния, используемые для спецификации двух и более конечных подавтоматов, которые могут выполняться параллельно внутри составного состояния.

Параллельные подсостояния



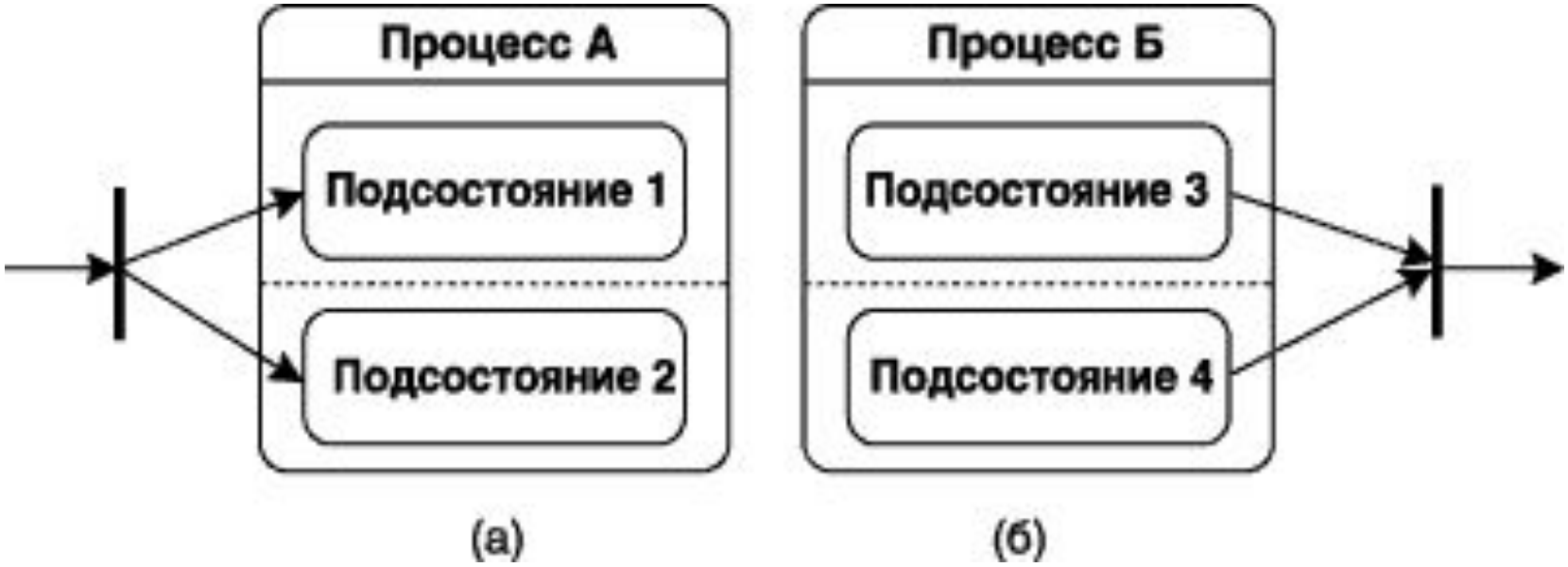
Составное состояние со скрытой внутренней структурой

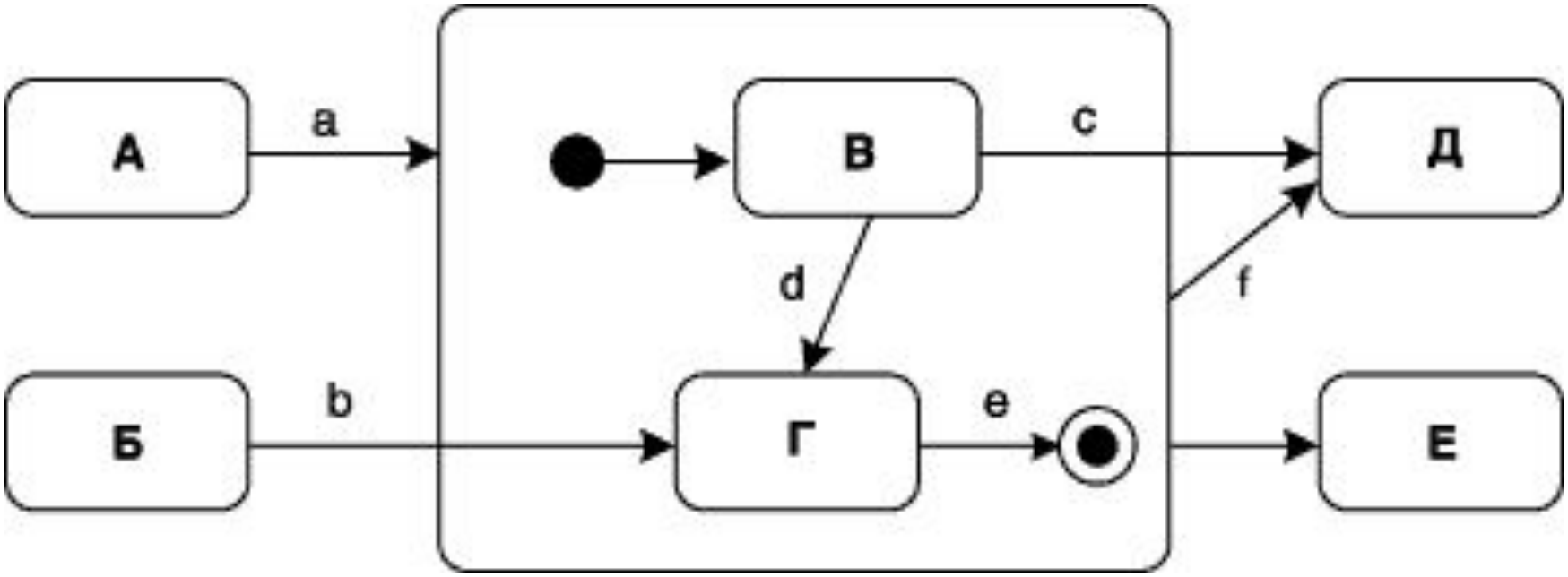
Параллельный переход

Параллельный переход – переход, который явно показывает ситуацию, когда переход может иметь несколько исходных состояний или целевых состояний.

Параллельные переходы разделения (fork) и слияния (join)

Параллельный переход




А
a
В
c
Д
d
f
Б
b
Г
e
Е

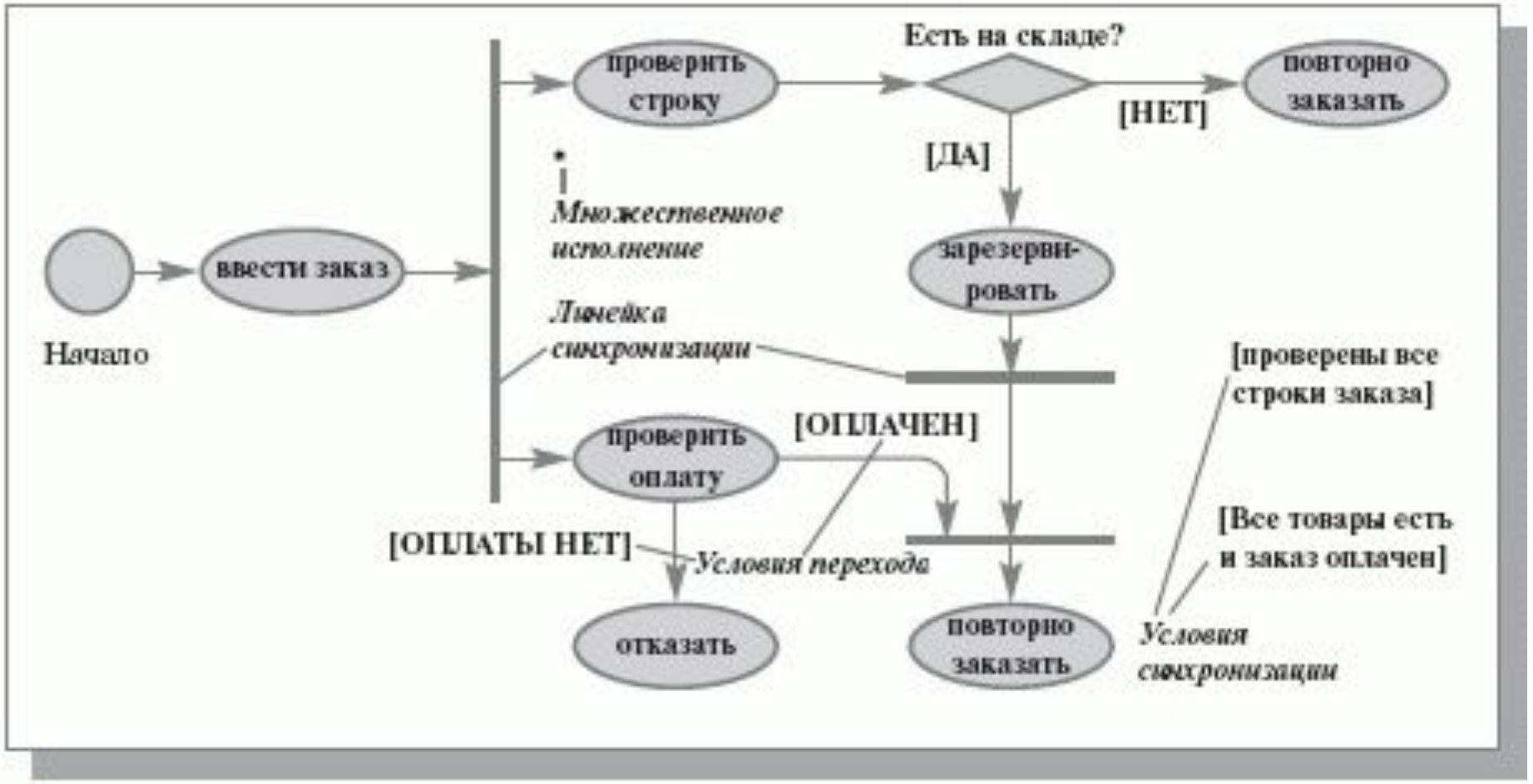
ДИАГРАММЫ ДЕЯТЕЛЬНОСТИ

Диаграммы деятельностей

Диаграммы деятельности (*activity diagrams*) – диаграмма, на которой показано разложение некоторой деятельности на её составные части.

Под **деятельностью** понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов — вложенных видов деятельности и отдельных действий , соединённых между собой переходами, которые идут от выходов одного узла ко входам другого.

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.


Начало
ввести заказ
проверить строку
Есть на складе?
повторно заказать
[НЕТ]
[ДА]
Множественное исполнение
зарезерви-ровать
Линейка синхронизации
проверить оплату
[ОПЛАЧЕН]
[ОПЛАТЫ НЕТ]
Условия перехода
отказать
повторно заказать
[проверены все строки заказа]
[Все товары есть и заказ оплачен]
Условия синхронизации

Состояния деятельности

Состояние деятельности (activity state) - состояние в графе деятельности, которое служит для представления процедурной последовательности действий, требующих определенного времени.

Состояние действия

Состояние действия **(action state)** - специальный случай состояния с некоторым входным действием и, по крайней мере, одним выходящим из состояния переходом.

Состояние деятельности и действия

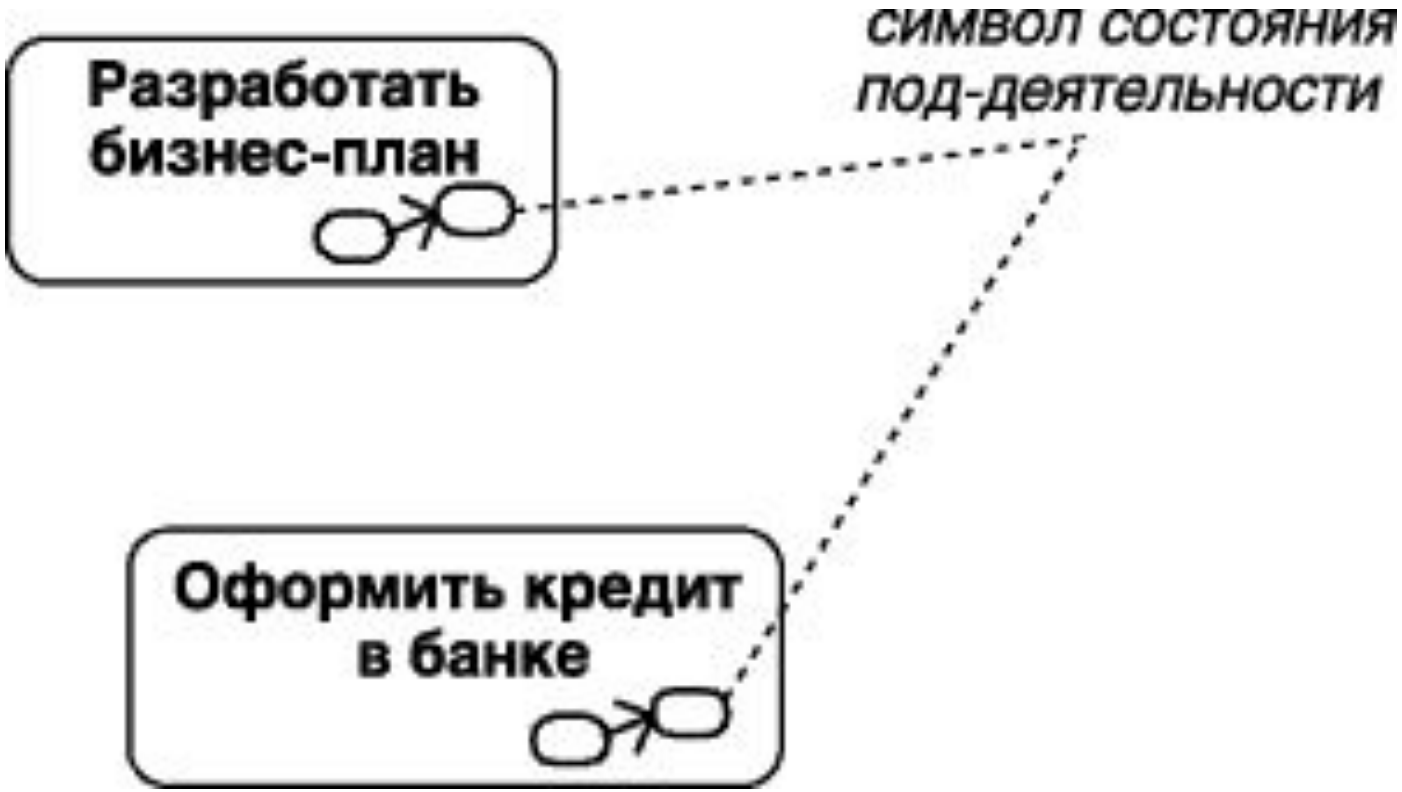
(а) простая деятельность

(б) выражение

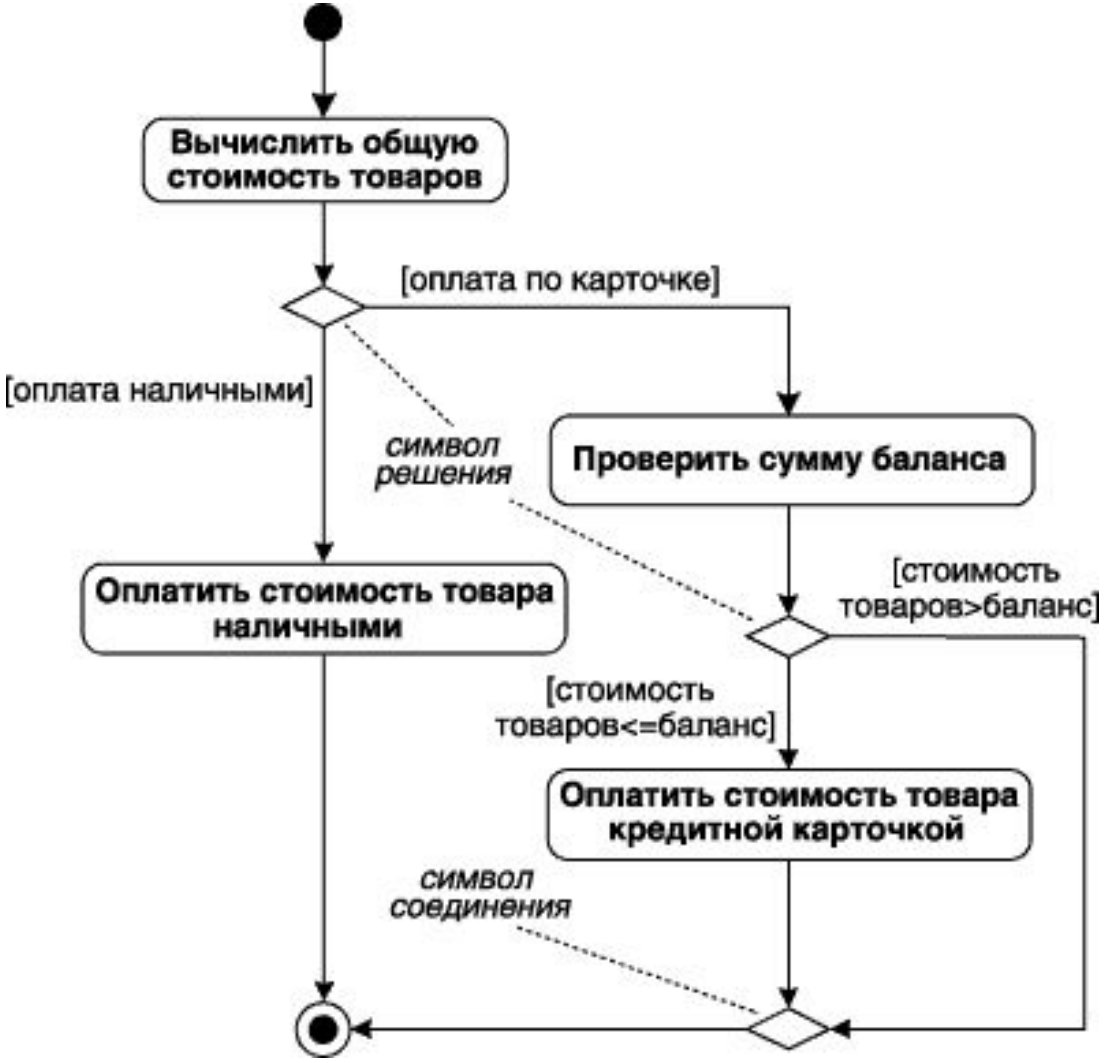
Состояние под-деятельности

Состояние под-деятельности (subactivity state) - состояние в графе деятельности, которое служит для представления неатомарной последовательности шагов процесса.

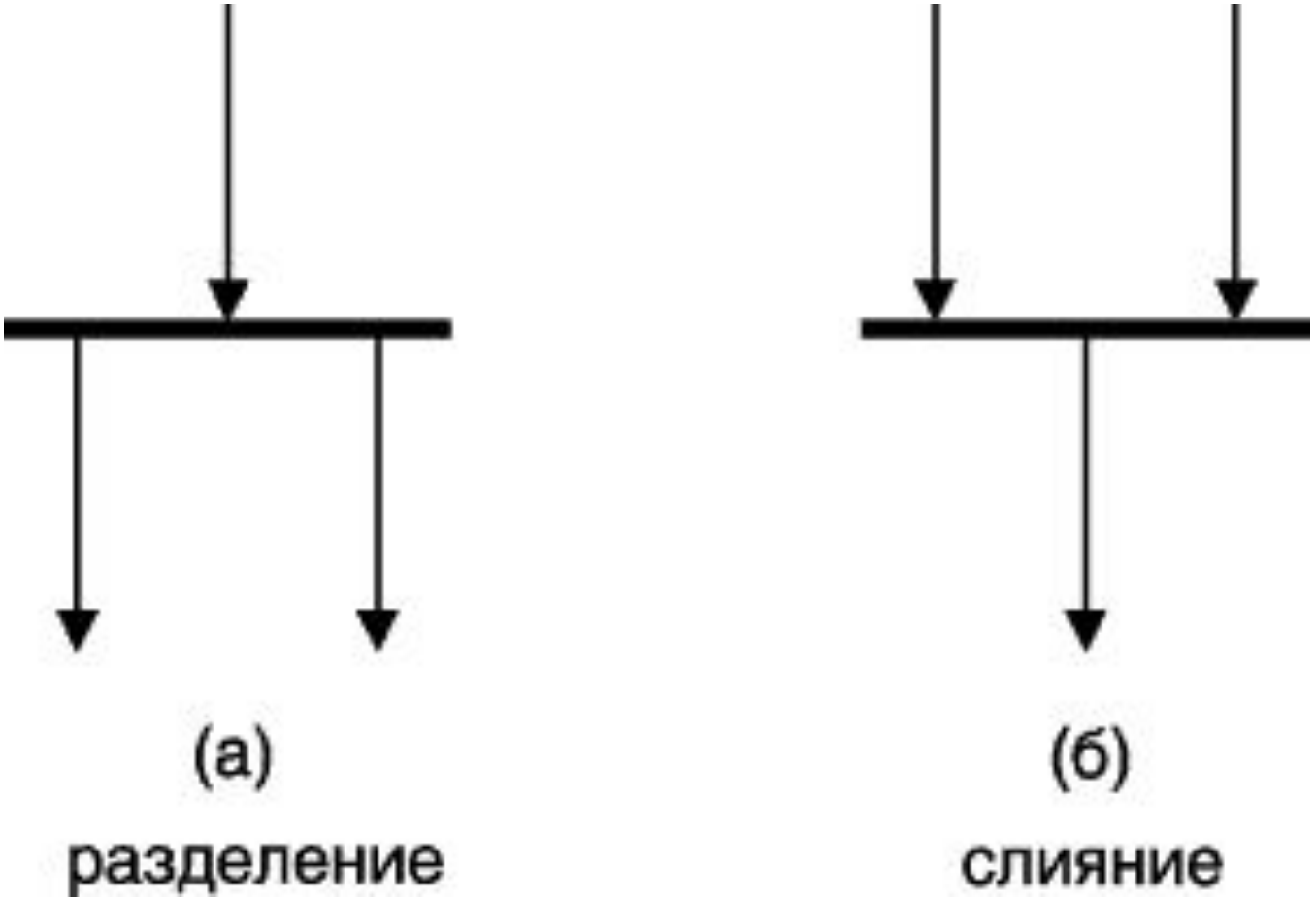
Состояние под-деятельности

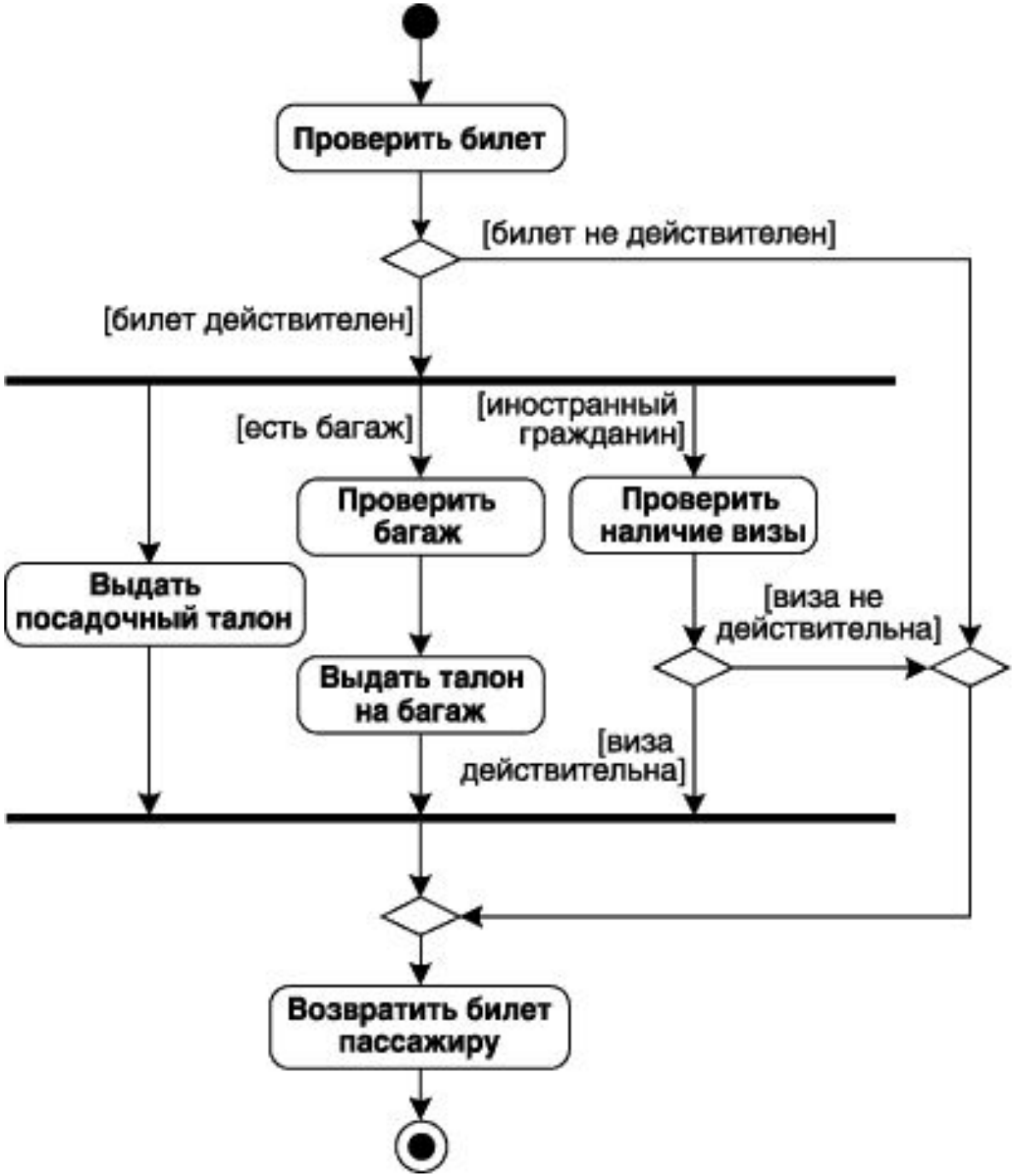


Ветвление и соединение



Разделение и слияние параллельных потоков

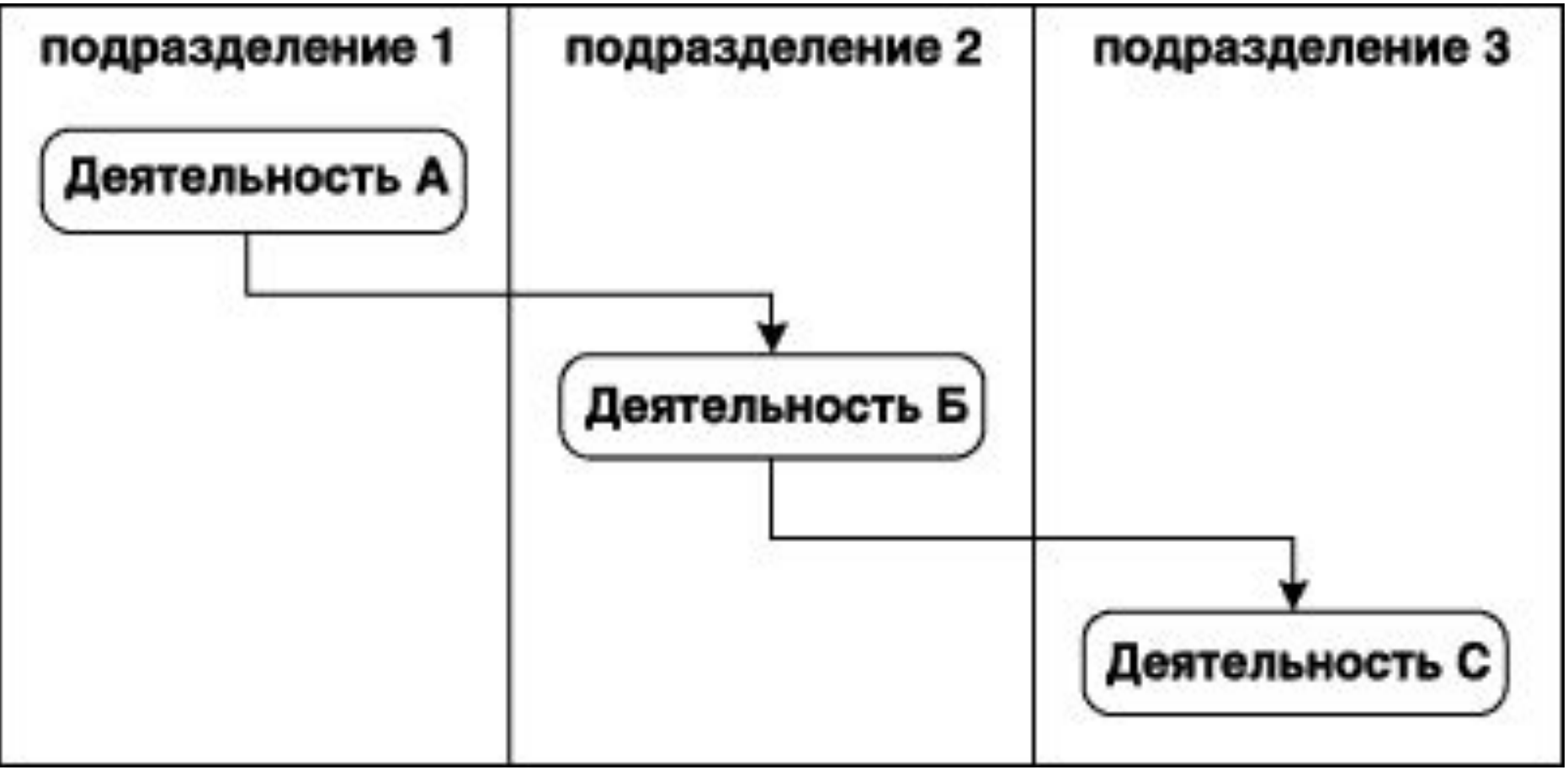


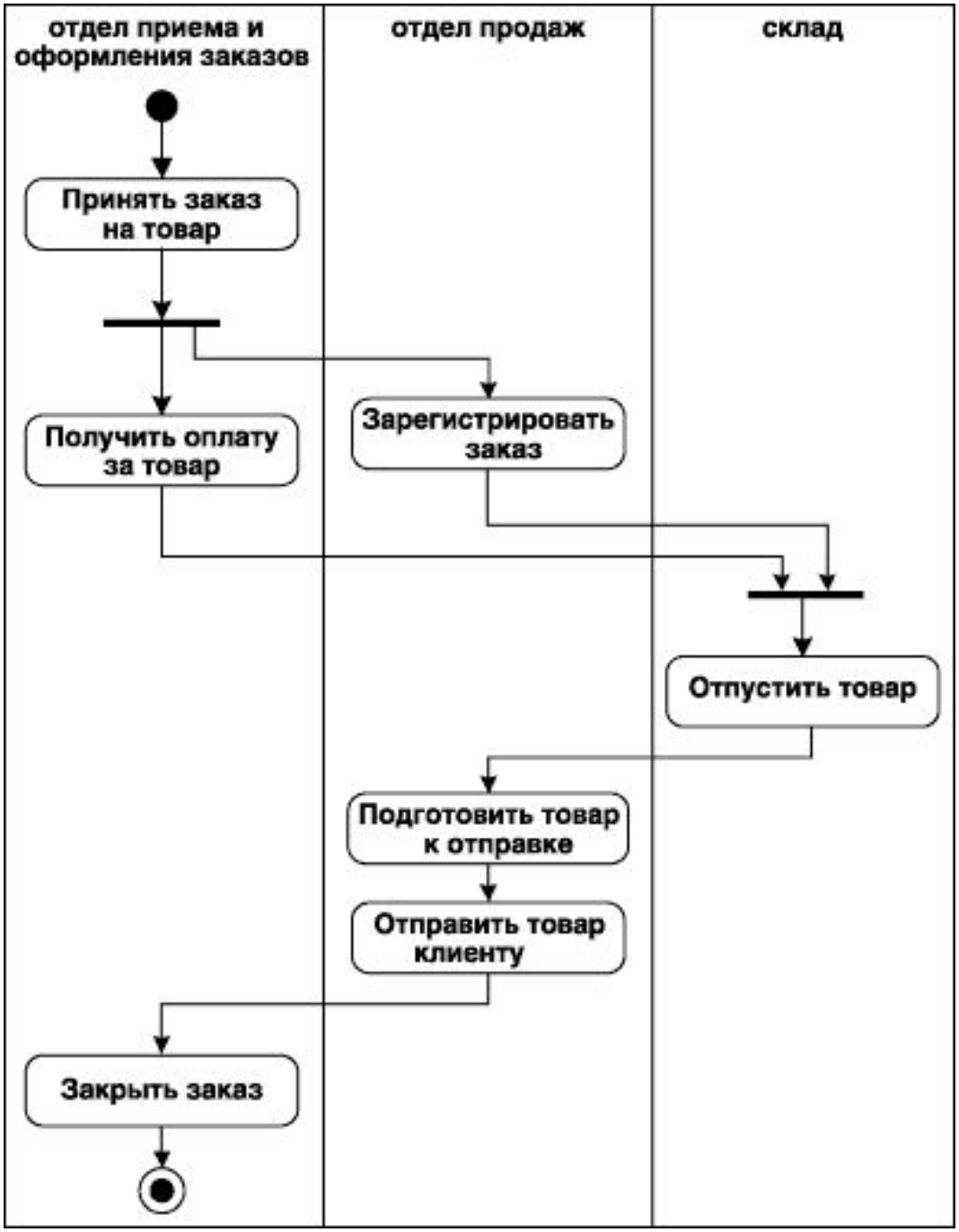

Проверить билет
[билет не действителен]
[билет действителен]
[есть багаж]
[иностранный гражданин]
Проверить багаж
Проверить наличие визы
Выдать посадочный талон
[виза не действительна]
Выдать талон на багаж
[виза действительна]
Возвратить билет пассажиру

Дорожки

Дорожка (swimlane) - графическая область диаграммы деятельности, содержащая элементы модели, ответственность за выполнение которых принадлежит отдельным подсистемам.

Дорожки


подразделение 1
подразделение 2
подразделение 3
Деятельность А
Деятельность Б
Деятельность С


отдел приема и оформления заказов
отдел продаж
склад
Принять заказ на товар
Получить оплату за товар
Зарегистрировать заказ
Отпустить товар
Подготовить товар к отправке
Отправить товар клиенту
Закрыть заказ

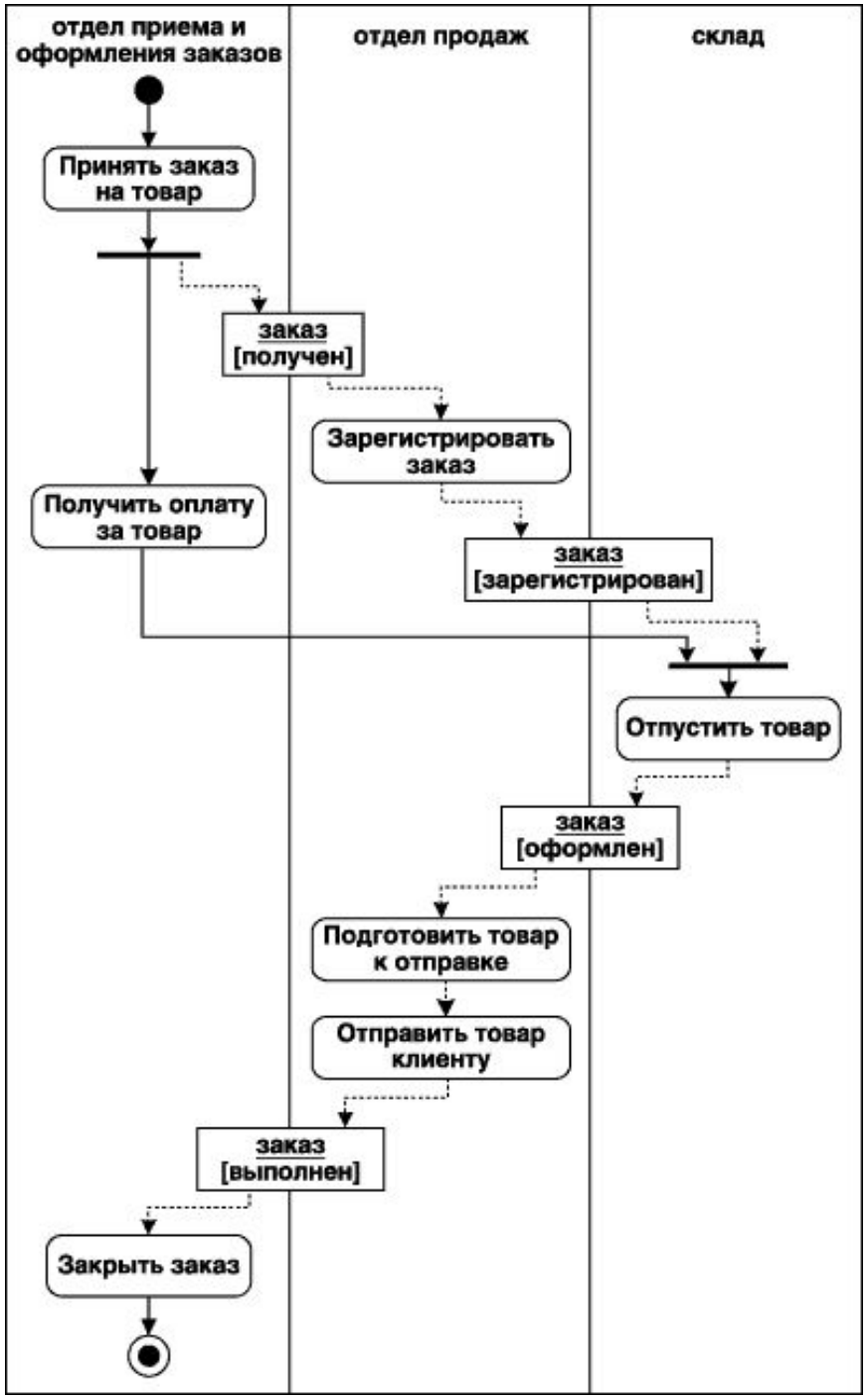

отдел приема и оформления заказов
отдел продаж
склад
Принять заказ на товар
заказ [получен]
Зарегистрировать заказ
Получить оплату за товар
заказ [зарегистрирован]
Отпустить товар
заказ [оформлен]
Подготовить товар к отправке
Отправить товар клиенту
заказ [выполнен]
Закрыть заказ

ДИАГРАММА КОМПОНЕНТОВ

Диаграммы компонентов

Диаграммы компонентов (*component* diagrams) – модель иерархии подсистем, отражает физическое *размещение* баз данных, приложений и интерфейсов ИС.

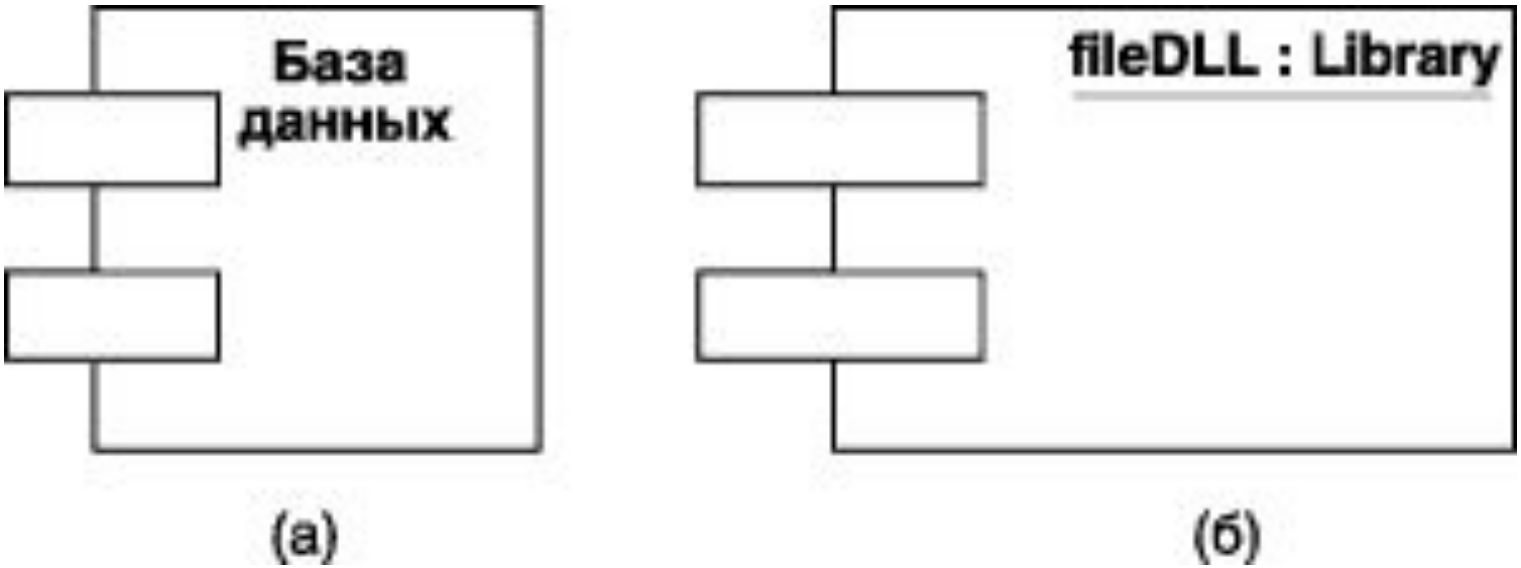
Компонент

Компонент (component) — физически существующая часть системы, которая обеспечивает реализацию классов и отношений, а также функционального поведения моделируемой программной системы.

Компонент предназначен для представления физической организации ассоциированных с ним элементов модели.

Компонентом может быть *исполняемый код* отдельного *модуля*, командные файлы или файлы, содержащие интерпретируемые скрипты.

Графическое изображение компонентов

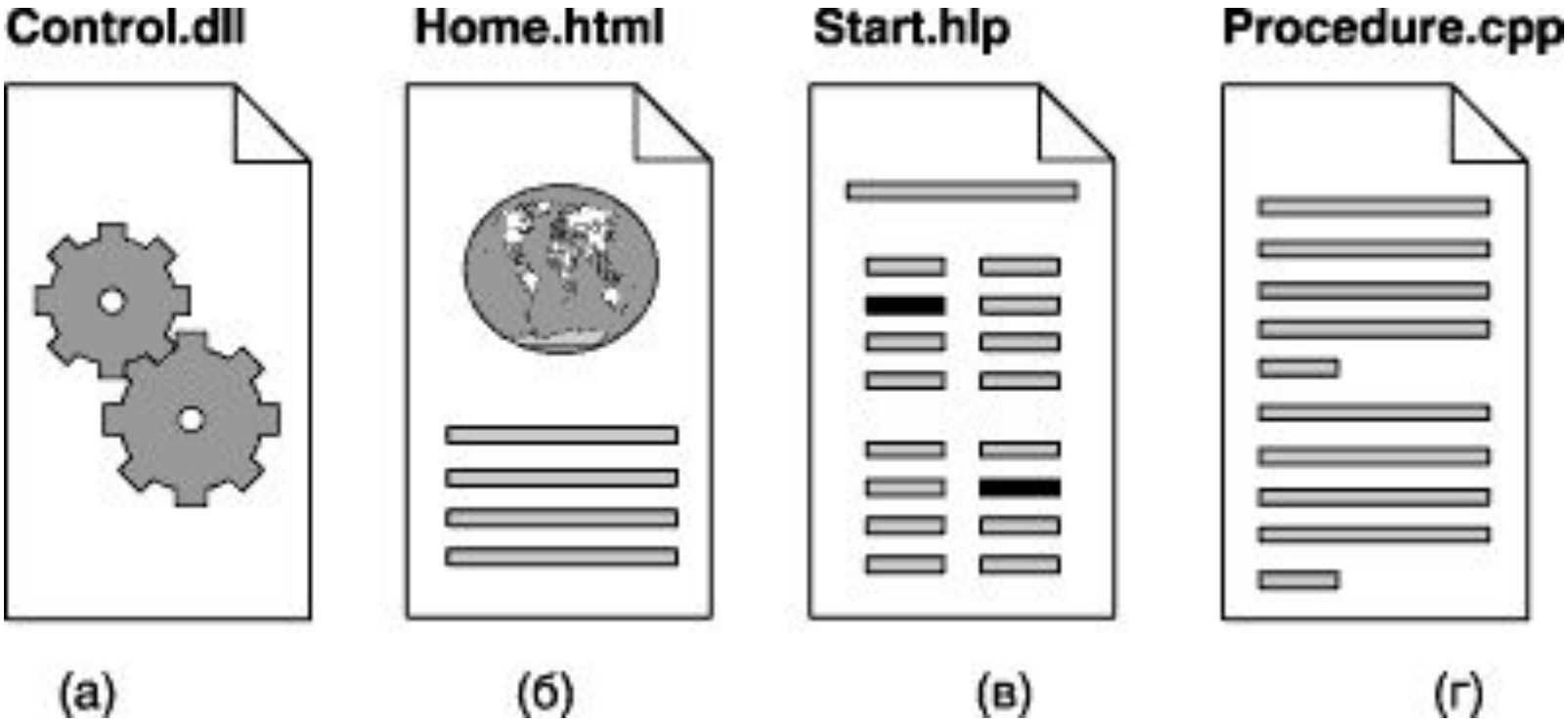


(а) (б)

Модуль

Модуль (module) — часть программной системы, требующая памяти для своего хранения и процессора для исполнения.

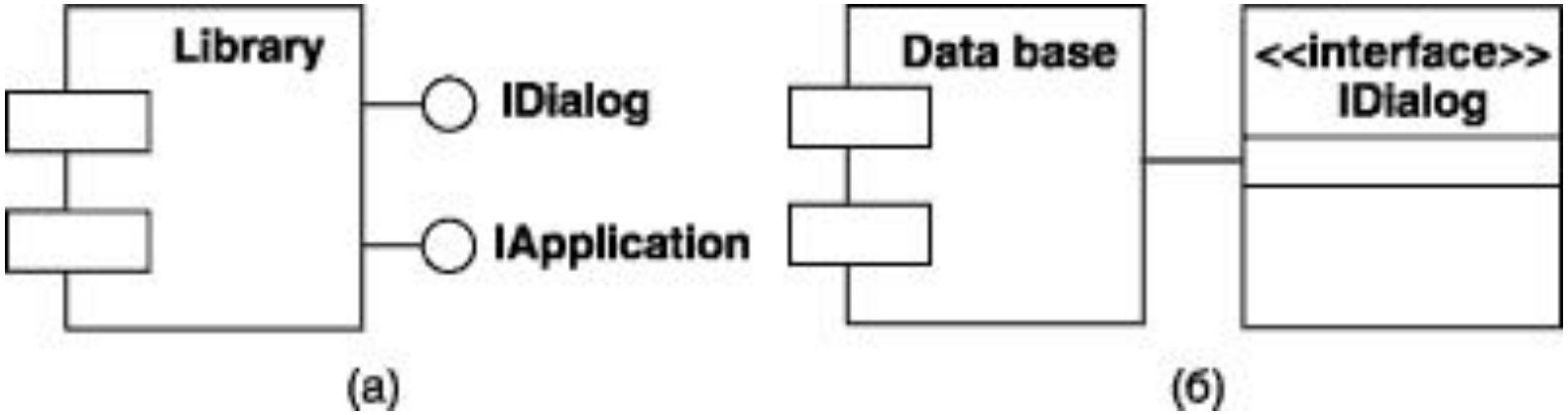
Обозначение физической реализации компонентов



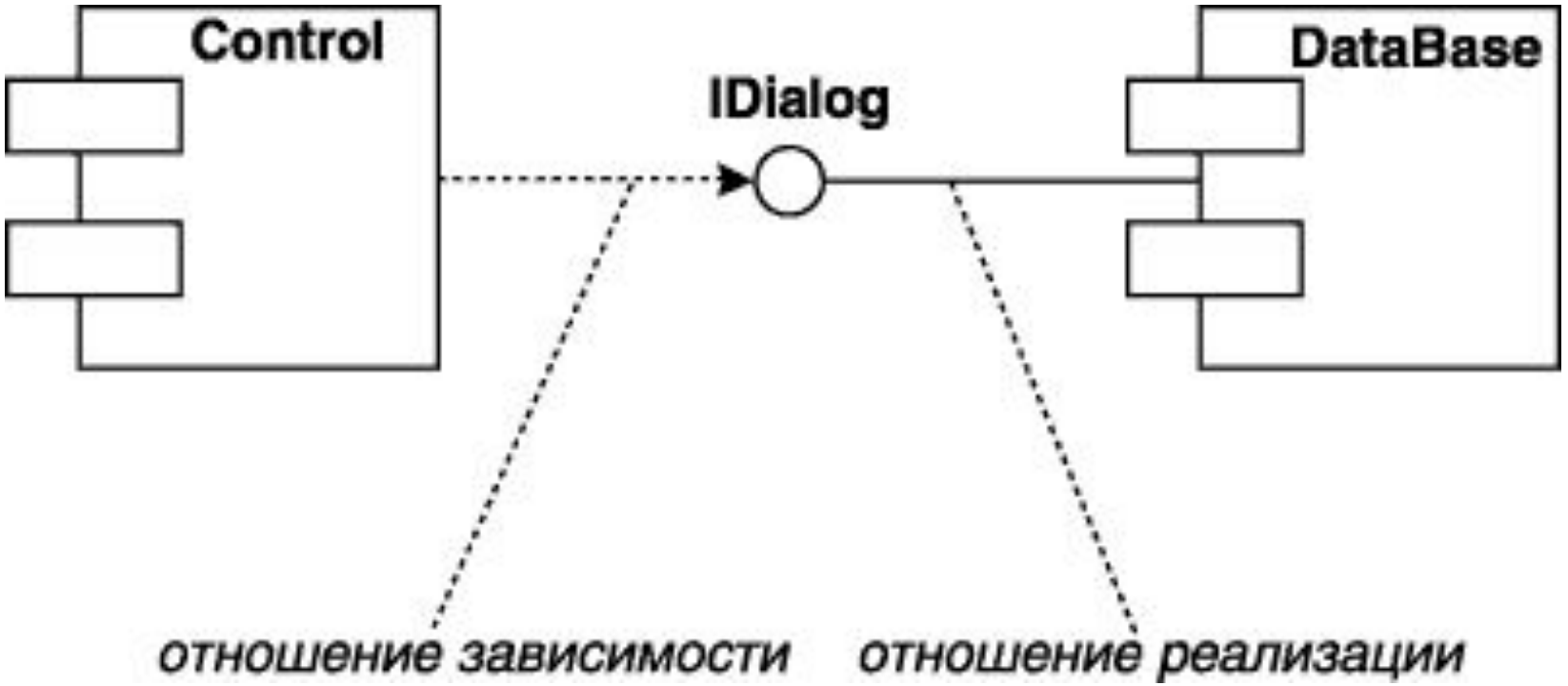
Стереотипы компонентов

- <<file>> (файл) – определяет наиболее общую разновидность *компонента*, который представляется в виде произвольного физического файла.
- <<executable>> (исполнимый) – определяет разновидность компонента-файла, который является исполнимым файлом и может выполняться на компьютерной платформе.
- <<document>> (документ) – определяет разновидность компонента-файла, который представляется в форме документа произвольного содержания, не являющегося исполнимым файлом или файлом с исходным текстом программы.
- <<library>> (библиотека) – определяет разновидность компонента-файла, который представляется в форме динамической или статической библиотеки.
- <<source>> (источник) – определяет разновидность компонента-файла, представляющего собой файл с исходным текстом программы, который после компиляции может быть преобразован в исполнимый файл.
- <<table>> (таблица) – определяет разновидность *компонента*, который представляется в форме таблицы базы данных.

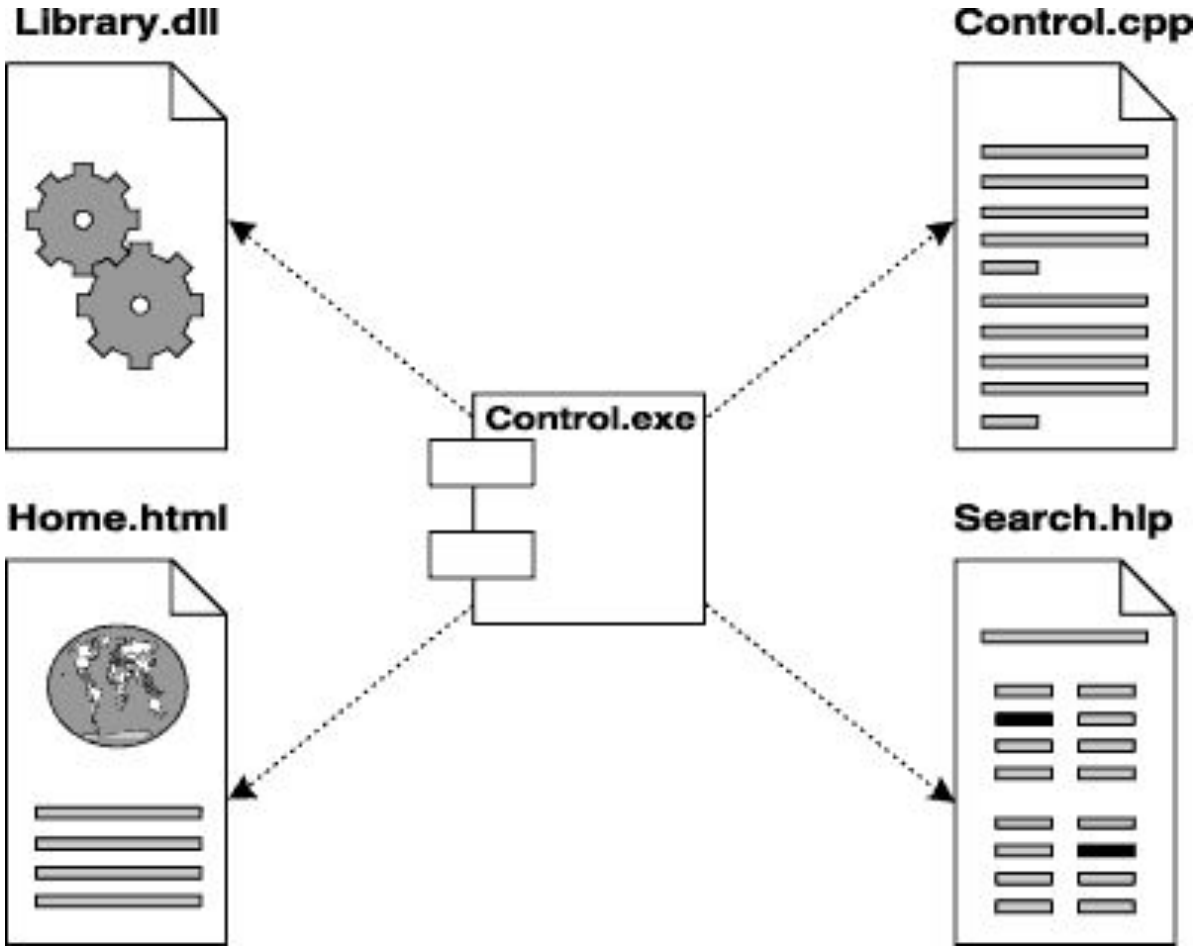
Интерфейсы


Library
IDialog
IApplication
(а)
Data base
<<interface>>
IDialog
(б)

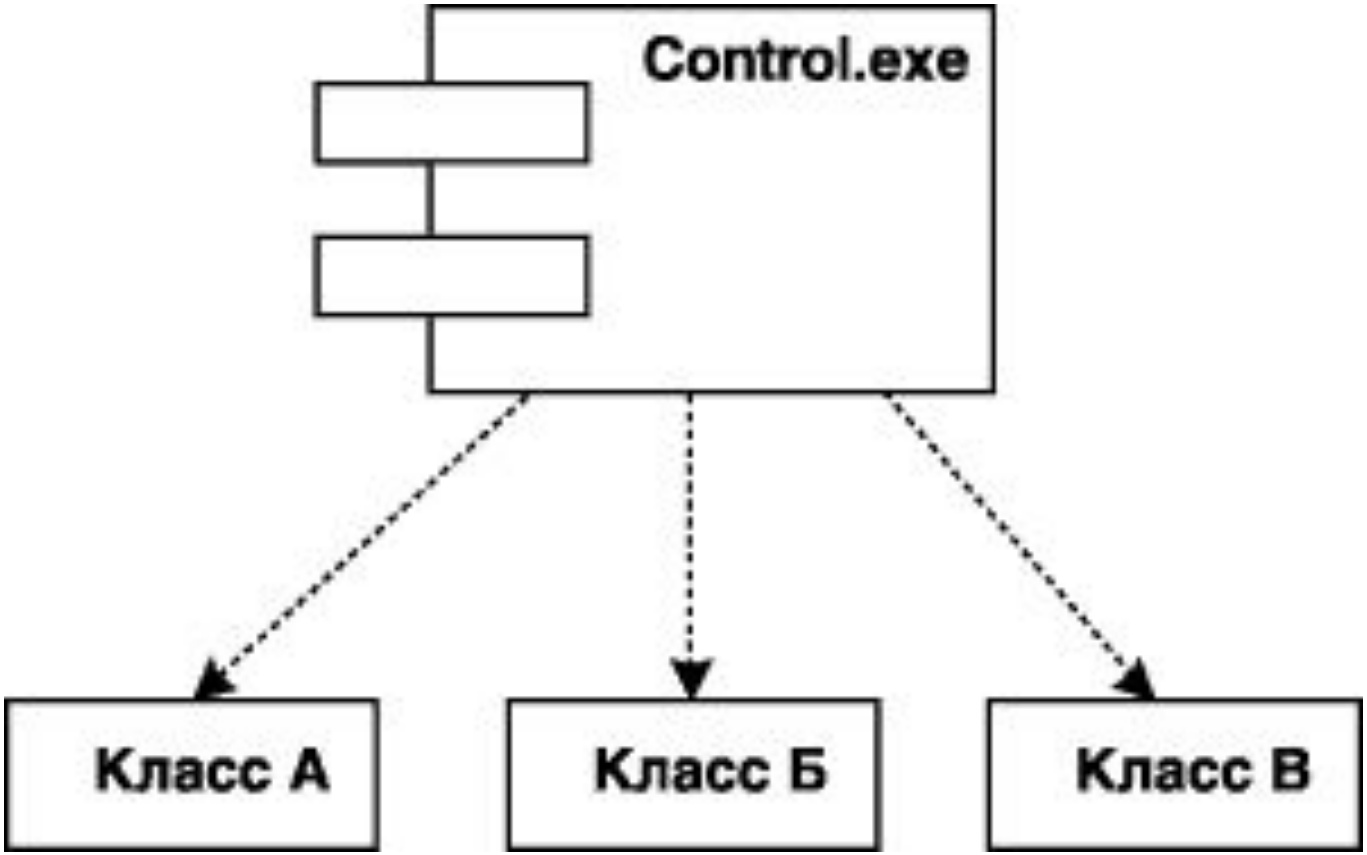
Отношения между интерфейсами и компонентами



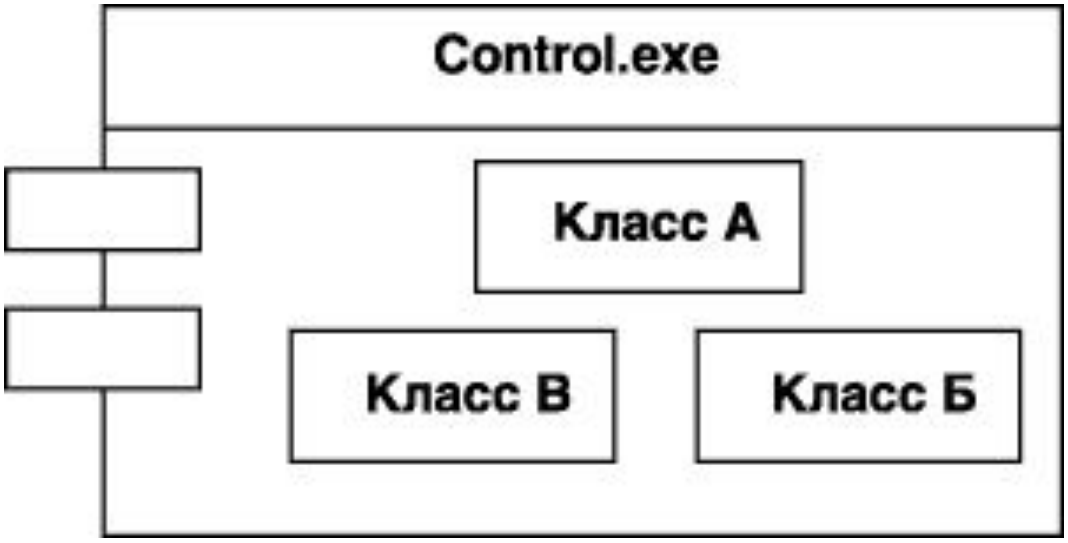
Зависимость между компонентами

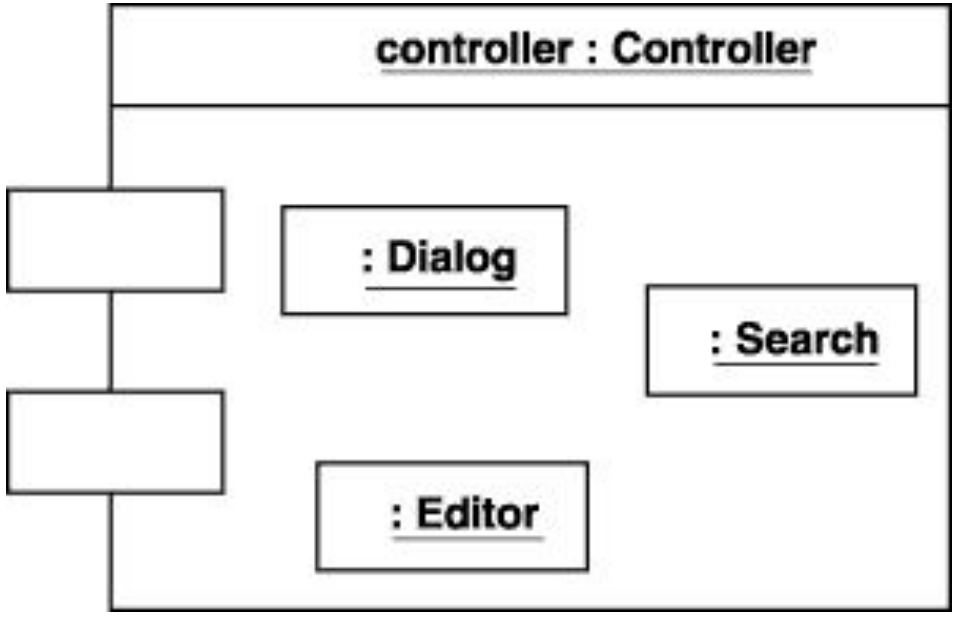


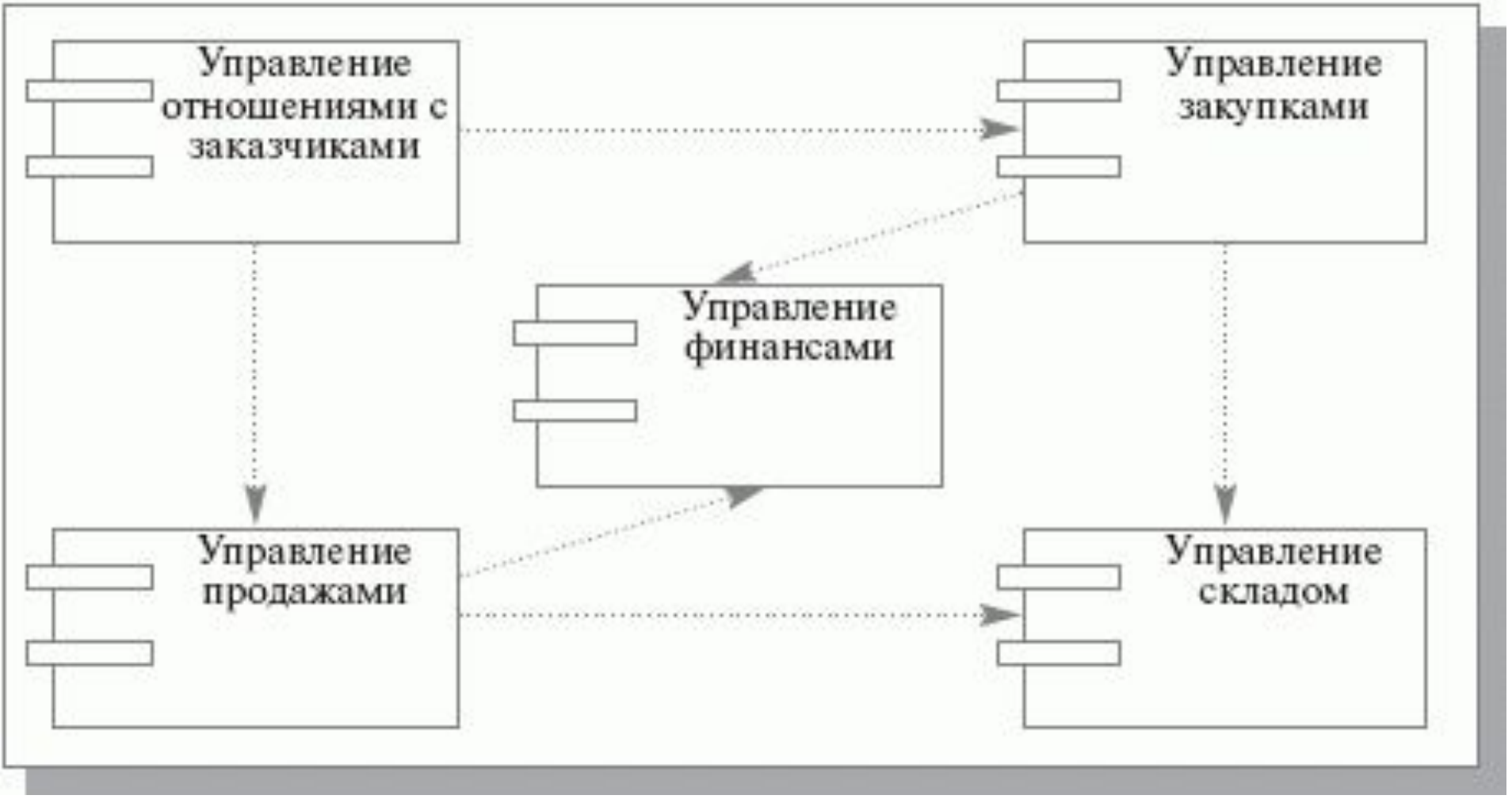
Зависимость между компонентом и классами



Реализация классов компонентом






Управление отношениями с заказчиками
Управление закупками
Управление финансами
Управление продажами
Управление складом

ДИАГРАММЫ РАЗВЕРТЫВАНИЯ

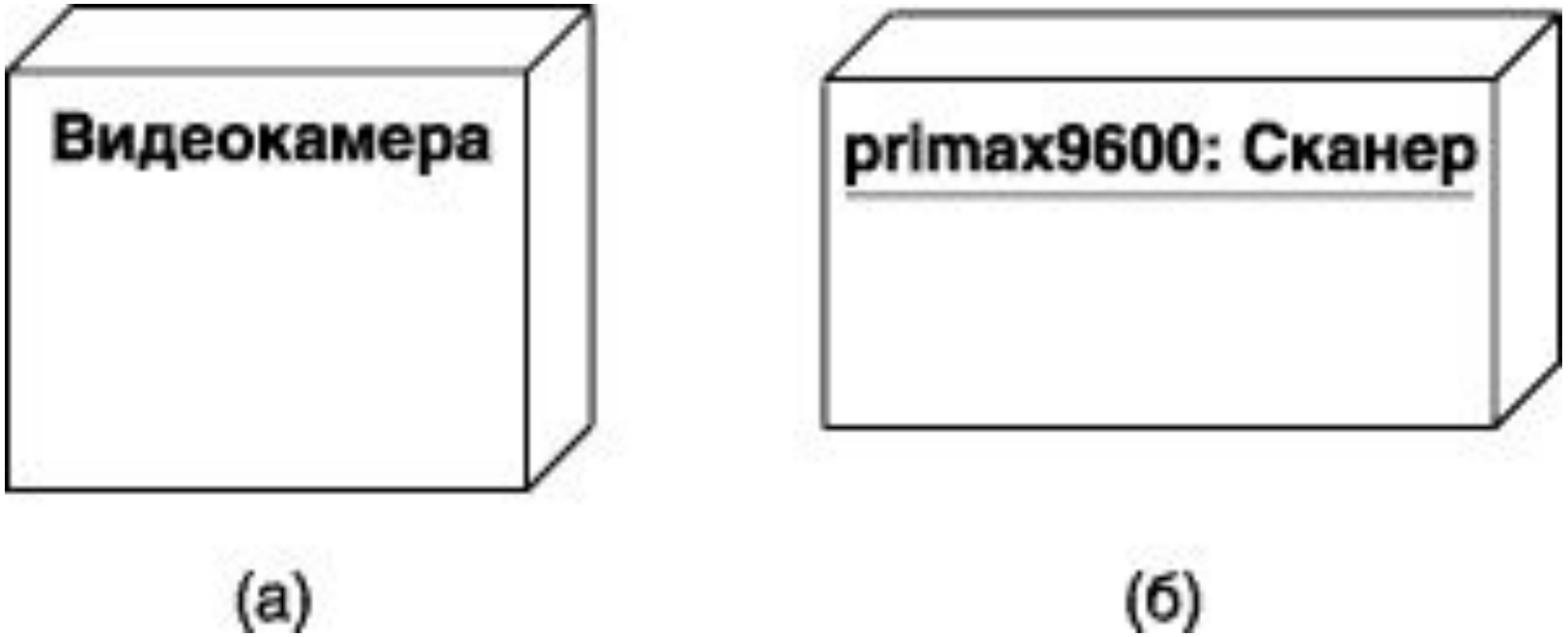
Диаграммы развертывания

Диаграммы развертывания (диаграммы размещения, *deployment* diagrams) – модель физической архитектуры системы, отображает аппаратную конфигурацию ИС.

Узел

Узел (node) представляет собой физически существующий элемент системы, который может обладать вычислительным ресурсом или являться техническим *устройством* .

Узел



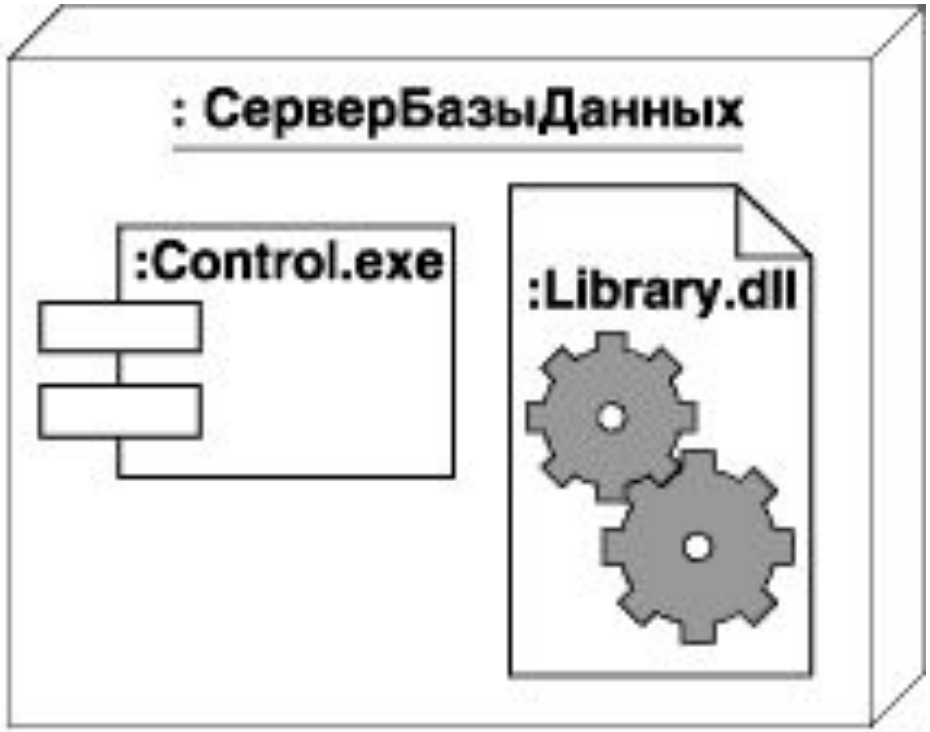
Узел


: МежсетевойЭкран
{допускает удаленное администрирование}
: СерверБазыДанных
{количество процессоров=4 оперативная память=2 Гб}
(а)
(б)

Узел


: СерверБазыДанных
Control.exe
Library.dll

(а)


: СерверБазыДанных
:Control.exe
:Library.dll

(б)

Стереотипы узлов

"*processor*" (*процессор*),

"*sensor*" (датчик),

"*modem*" (*модем*),

"*net*" (*сеть*),

"*printer*" (принтер) и другие

Стереотипы ресурсоемкий узел и устройство

Соединения


<<processor>>
: Сервер Базы Данных
<<net>>
локальная сеть
<<processor>>
: Рабочая станция
{Реализовать на оптоволоконной технологии}

Зависимости

Мобильный доступ к корпоративной БД