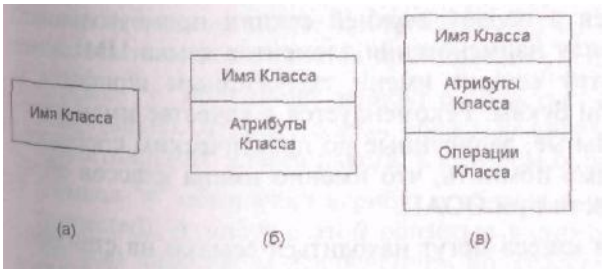
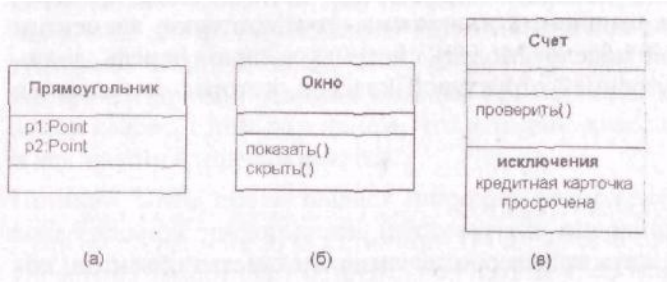
Диаграмма классов

Диаграмма классов (**class diagram**) служит для представления *статической структуры модели системы* в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать, в частности, различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений

Класс (class) в языке UML служит для обозначения множества объектов, которые обладают одинаковой структурой, поведением и отношениями с объектами из других классов. В разделах могут указываться имя класса, *атрибуты* (переменные) и *операции* (методы).



Обязательным элементом обозначения класса является его имя. На начальных этапах разработки диаграммы отдельные классы могут обозначаться простым прямоугольником с указанием только имени соответствующего класса.



Имя класса должно быть уникальным в пределах пакета, который описывается некоторой совокупностью диаграмм классов (возможно, одной диаграммой). Должно начинаться с заглавной буквы. Рекомендуется в качестве имен классов использовать *существительные*, записанные по *практическим соображениям* без пробелов.

Примеры имен классов: "Сотрудник", "Компания", "Руководитель", "Клиент", "Продавец", "Менеджер", "Офис". Имена имеют непосредственное отношение к моделируемой предметной области и функциональному назначению проектируемой системы.

Класс может не иметь экземпляров или объектов. В этом случае он называется *абстрактным* классом, а для обозначения его имени используется наклонный шрифт (курсив).

Атрибуты класса

Во второй секции прямоугольника класса записываются его *атрибуты* (attributes) или свойства. Каждому атрибуту класса соответствует отдельная строка текста:

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

<тип атрибута> = <исходное значение>{строка-свойство}

Операция

Операция (**operation**) представляет собой некоторый сервис, предоставляющий каждый экземпляр класса по определенному требованию. Каждой операции класса соответствует отдельная строка:

<квантор видимости><имя операции>(список параметров):
<выражение типа возвращаемого значения>{строка-свойство}

Отношения между классами

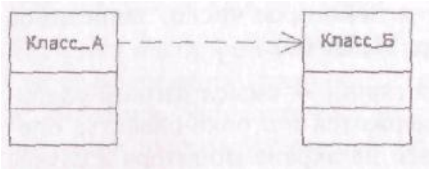
Базовыми отношениями или связями между классами в языке UML являются:

- Отношение зависимости (*dependency relationship*)
- Отношение ассоциации (*association relationship*)
- Отношение обобщения (*generalization relationship*)
- Отношение реализации (*realization relationship*)

Каждое из этих отношений имеет собственное графическое представление на диаграмме, которое отражает взаимосвязи между объектами соответствующих классов.

Отношение зависимости

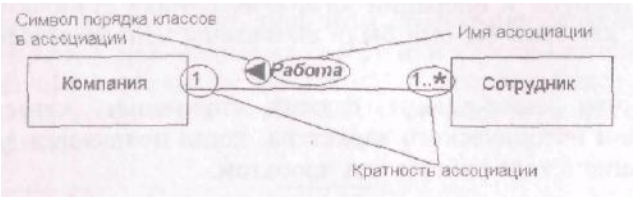
Отношение зависимости в общем случае указывает некоторое семантическое отношение между двумя элементами модели или двумя множествами таких элементов, которое не является отношением ассоциации, обобщения или реализации.



Отношение ассоциации

Отношение ассоциации соответствует наличию некоторого отношения между классами. Могут указываться имя ассоциации, имена и кратность классов-ролей ассоциации.

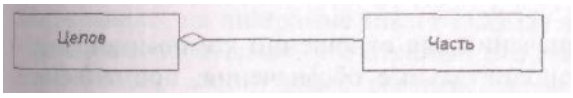
Простой случай — бинарная ассоциация. Она связывает в точности два класса и, как исключение, может связывать класс с самим собой.



Отношение агрегации

Отношение агрегации имеет место, если один из классов представляет собой некоторую сущность, включающую в себя в качестве составных частей другие сущности.

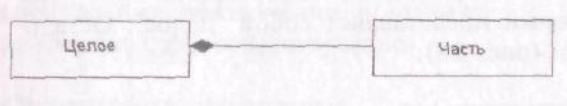
Применяется для представления системных взаимосвязей типа "часть-целое".



Отношение композиции

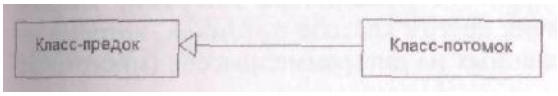
Отношение композиции является частным случаем отношения агрегации. Это специальная форма отношения "часть-целое", при которой составляющие части в некотором смысле находятся *внутри целого*. Специфика взаимосвязи — части не могут

выступать в отрыве от целого. С уничтожением целого уничтожаются и все его составные части.



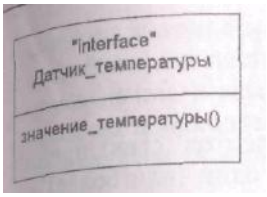
Отношение обобщения

Отношение обобщения является обычным таксономическим отношением между более общим элементом (родителем или предком) и более частным или специальным элементом (дочерним или потомком). Применительно к диаграмме классов данное отношение описывает иерархическое строение классов и наследование их свойств и поведения.



Интерфейсы

При построении, диаграммы классов отдельные интерфейсы могут уточняться. Для их изображения используется прямоугольник класса с ключевым словом или стереотипом "interface". Секция атрибутов отсутствует. Указывается только секция операций.



Объекты

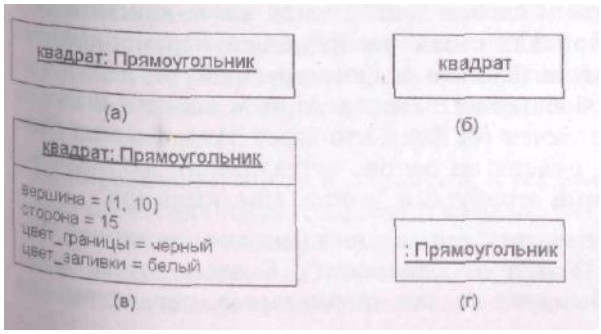
Объект (object) является отдельным экземпляром класса, который создается на этапе выполнения программы.

Для графического изображения объектов используется такой же символ прямоугольника, что и для классов. При указании имен объектов, они обязательно подчеркиваются. Запись имени объекта представляет собой строку текста "имя объекта:имя класса", разделенную двоеточием.

Имя объекта может отсутствовать.

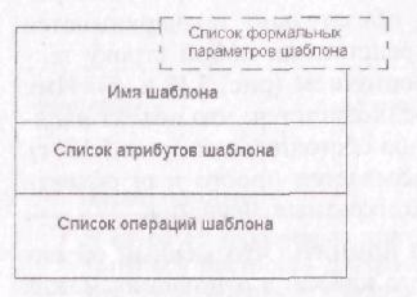
Отсутствовать может и имя класса. Тогда указывается просто имя объекта.

Атрибуты объектов принимают конкретные значения.



Шаблоны или параметризованные классы

Шаблон (template) или *параметризованный класс* (parametrized class) предназначен для обозначения класса, который имеет один или более нефиксированный формальный параметр.



Частным случаем между шаблоном и формируемым от него классом является отношение обобщения с наследованием свойств шаблона.

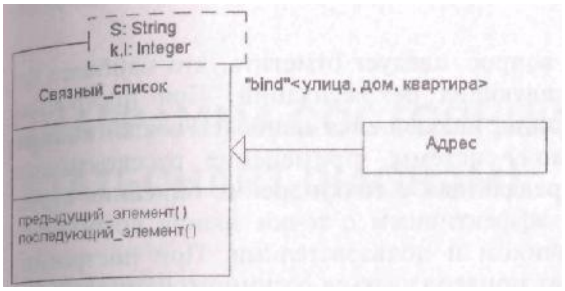


Диаграмма Деятельности (Activity Diagram)

Для моделирования процесса выполнения операций в языке UML используются *диаграммы деятельности*. На диаграмме деятельности отображается логика или последовательность перехода от одной деятельности к другой, при этом внимание фиксируется на результате деятельности.

Состояние действия

Состояние действия (action state) является специальным случаем состояния с некоторым входным действием и, по крайней мере, одним выходящим состояния переходом. Этот переход неявно предполагает, что входное действие уже завершилось.

Графически состояние действия изображается фигурой, напоминающей прямоугольник, боковые стороны которого заменены выпуклыми дугами. Внутри фигуры записывается *выражение действия* (action-expression).

Каждая диаграмма деятельности должна иметь единственное начальное и единственное конечное состояния.

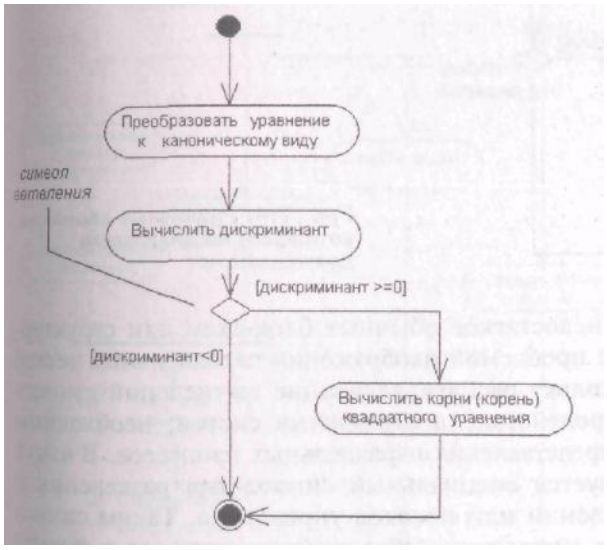
Диаграмму деятельности принято располагать таким образом, чтобы действия следовали сверху вниз.

Переходы

При построении диаграммы деятельности используются только *нетриггерные* переходы, т.е. такие, которые срабатывают сразу после завершения деятельности или выполнения соответствующего действия.

Единственный переход может быть никак не помечен.

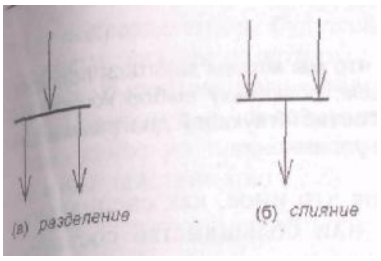
Ветвление на диаграмме деятельности обозначается небольшим *ромбом* без текста. В ромб может входить только одна стрелка. Выходящих стрелок может быть две или более. Для каждой явно указывается соответствующее сторожевое условие в форме булевского выражения.



Особенности параллельных процессов изображаются отрезком толстой горизонтальной или вертикальной линии.

Переход *разделение* (concurrent fork) имеет один входящий переход и несколько выходящих.

Переход *слияние* (concurrent join) имеет несколько входящих переходов и один выходящий.



Пример. Диаграмма деятельности процесса приготовления напитка.

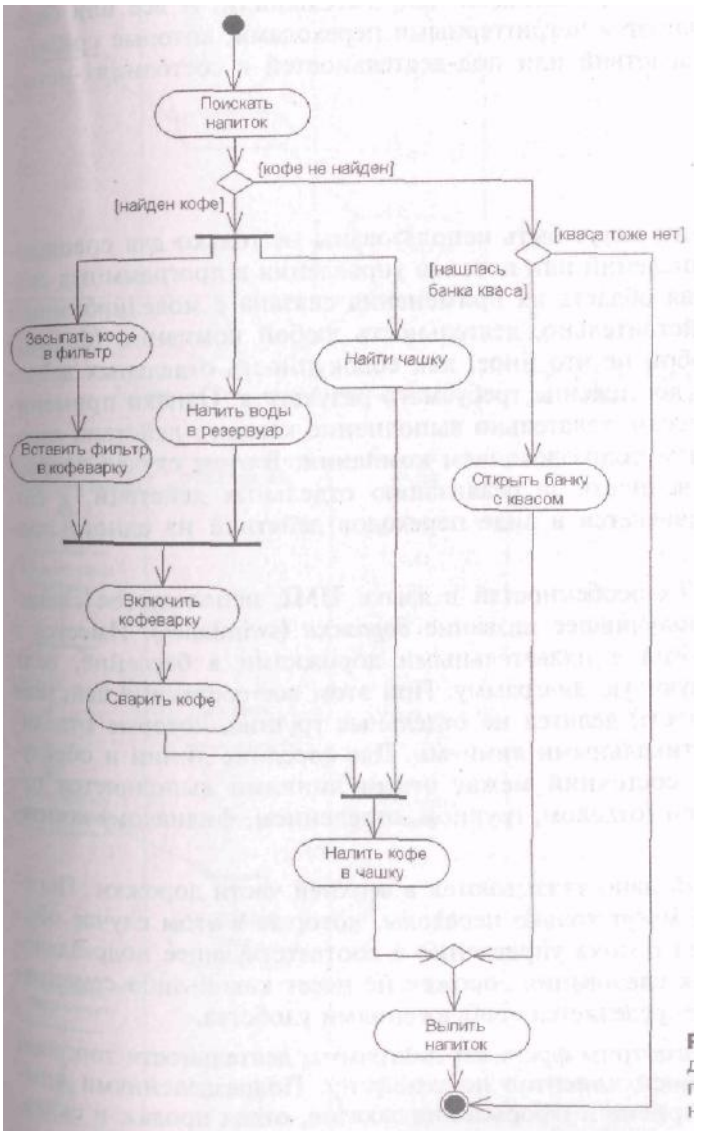


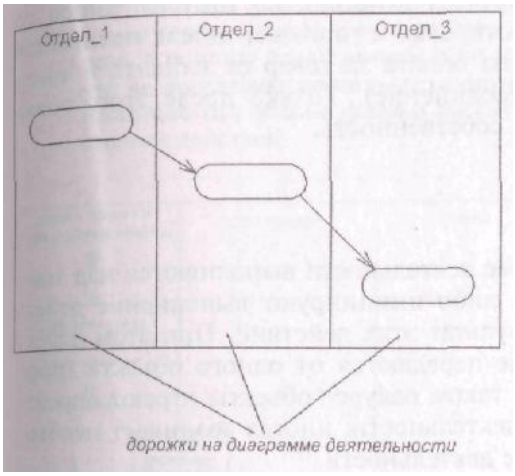
Диаграмма деятельности — специальный случай диаграммы состояний.

Дорожки

Применительно к бизнес-процессам желательно выполнение каждого действия ассоциировать с конкретным подразделением компании. В этом случае подразделение несет ответственность за реализацию отдельных действий, а сам бизнес-процесс представляется в виде переходов действий из одного подразделения к другому. То же можно применить для моделирования распределения функций между подсистемами программы.

Для моделирования этих особенностей в языке UML используется специальная конструкция — *дорожки* (swimlanes).

Названия подразделений явно указываются в верхней части дорожки. Пересекать линию дорожки могут только переходы. Порядок следования дорожек определяется соображениями удобства.



дорожки на диаграмме деятельности

Объекты

Действия на диаграмме деятельности выполняются над теми или иными объектами. Эти объекты либо инициируют выполнение действий, либо определяют некоторый результат этих действий. Действия специфицируют вызовы, которые передаются от одного объекта графа деятельности к другому. Иногда возникает необходимость явно указать их на диаграмме деятельности.