

Анализ и проектирование на UML

Направление подготовки
“Информационные системы и технологии”

Максим Валерьевич Хлопотов,
старший преподаватель кафедры ИС

Общие механизмы

В UML имеются общие правила и механизмы, которые относятся ко всему языку в целом.

Выделяют следующие общие механизмы:

- внутреннее представление модели;
- дополнения;
- подразделения;
- механизмы расширения.

Внутреннее представление

Модель имеет внутреннее представление

Для графов используются способы представления: матрица смежности, списки смежности и др.

Разработчики инструментов для моделирования на UML придумать свое (что обычно и делается).

У каждого элемента модели есть «оборотная сторона», где записаны все свойства, даже те, которые в данном контексте не нужно или нельзя показывать на картинке.

Внутреннее представление

Внутреннее представление содержит список стандартных свойств, определенных для каждого элемента модели. Такое внутреннее представление может быть однозначно переведено во внешнее представление.

Внутреннее представление может быть переведено в текст в формате XMI (конкретное приложение XML) без потери информации.

Текстовое представление моделей UML используется инструментами моделирования, например, для обмена моделями.

Дополнения

Базовая графическая нотация может быть расширена путем использования дополнительных текстовых и/или графических объектов, присоединяемых к базовой нотации.

Такие дополнительные объекты называются **дополнения**.

Дополнения позволяют показать на диаграмме те части внутреннего представления модели, которые не отображаются с помощью базовой графической нотации.

Дополнения (пример)

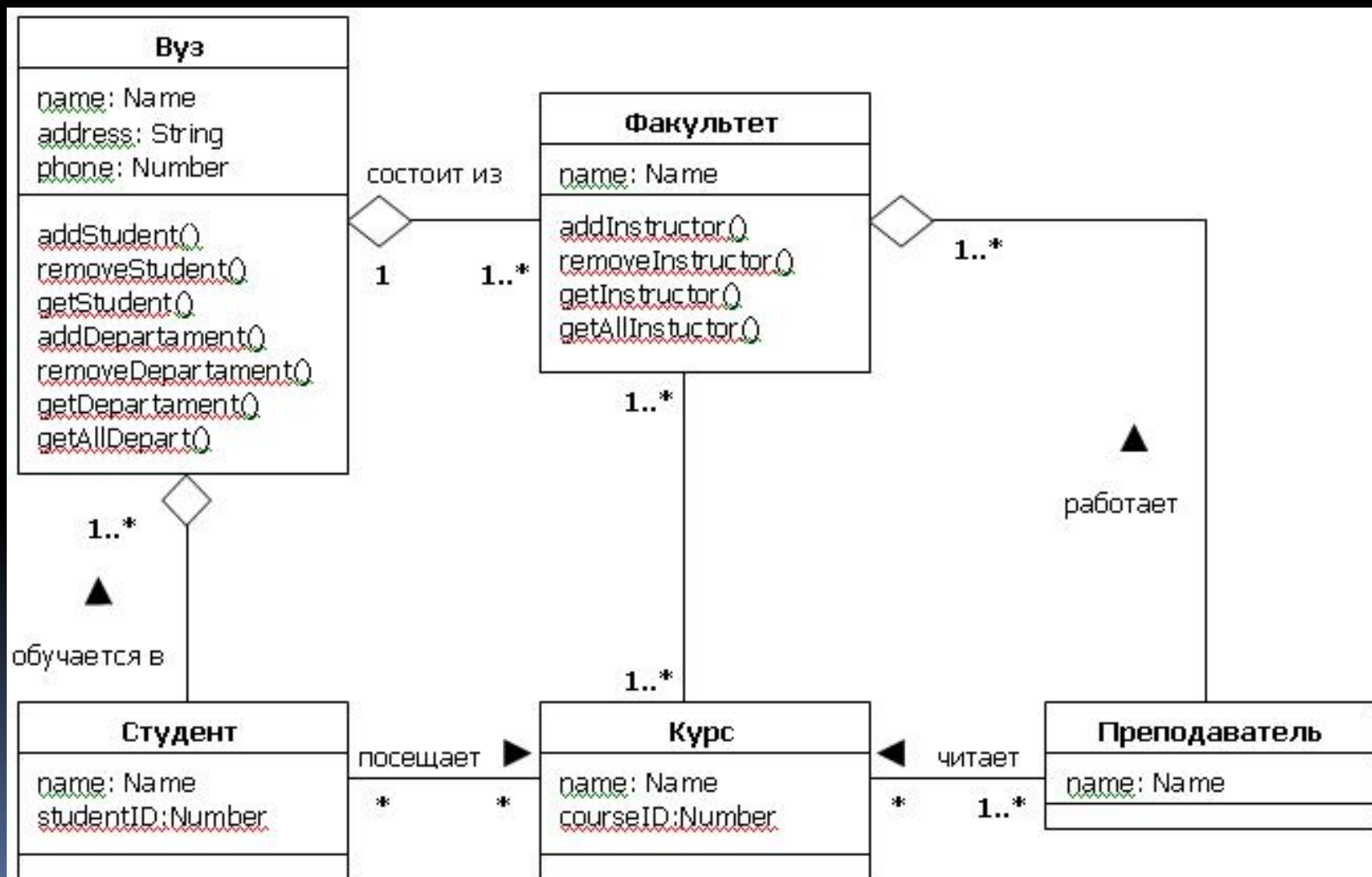



РИС 3.9



Подразделения

UML является объектно-ориентированным языком, поэтому базовые понятия объектно-ориентированного подхода имеют в языке сквозное действие.



Подразделения


Пример 1. Четко различается о чём идет речь: об общем описании некоторого множества однотипных объектов (т. е. о классе) или о конкретном объекте из некоторого множества однотипных объектов (т. е. об экземпляре класса).

Это различие передается единообразно: если это конкретный объект, то его имя подчеркивается; если это класс, то оно не подчеркивается.



Подразделения

Пример 2. Если абстрактный интерфейс, то при записи имени используется курсивное начертание шрифта, если конкретная реализация — используется прямое начертание.




Механизмы расширения

Механизмы расширения — это встроенный в язык способ изменить язык. Авторы UML при унификации различных методов постарались включить в язык как можно больше различных средств (удерживая объем языка в рамках разумного), так чтобы язык оказался применимым в разных контекстах и предметных областях. Но вполне могут возникать и возникают случаи, когда стандартных элементов моделирования не хватает или они не вполне адекватны.



Механизмы расширения

Механизмы расширения позволяют определять новые элементы модели на основе существующих управляемым и унифицированным способом. Таких механизмов три:

- помеченные значения;
 - ограничения;
 - стереотипы.
- 

Механизмы расширения

Помеченное значение — это пара: имя свойства и значение свойства, которую можно добавить к любому стандартному элементу модели.

К любому элементу модели можно добавить любое помеченное значение, которое будет храниться также, как и все стандартные свойства данного элемента. Способ обработки помеченного значения, определенного пользователем, стандартом не определяется и отдается на откуп инструменту.

Механизмы расширения

Помеченные значения записываются в модели в виде строки текста, имеющей следующий синтаксис: в фигурных скобках указывается пара: имя и значение, разделенные знаком равенства. Можно указывать сразу список пар, разделяя элементы списка запятыми.



Механизмы расширения

Ограничение — это логическое утверждение относительно значений свойств элементов модели. Логическое утверждение может иметь два значения: истина и ложь, то есть задаваемое им условие либо выполняется, либо не выполняется. Указывая ограничение для элемента модели, мы расширяем его семантику, требуя, чтобы ограничение выполнялось. Ограничение может относиться к отдельному элементу или к совокупности элементов модели или к совокупности элементов модели.

Механизмы расширения

Ограничения записываются в виде строки текста, заключенной в фигурные скобки. Это может быть неформальный текст на естественном языке, логическое выражение языка программирования, поддерживаемого инструментом или выражение на некотором другом формальном языке, специально включенного для этой цели в UML.



Механизмы расширения

Стереотип — это определение нового элемента моделирования в UML на основе существующего элемента моделирования.

Определение стереотипа производится следующим образом. Взяв за основу некоторый существующий элемент модели, к нему добавляют новые помеченные значения (расширяя тем самым внутреннее представление), новые ограничения (расширяя семантику) и дополнения, то есть новые графические элементы (расширяя нотацию).

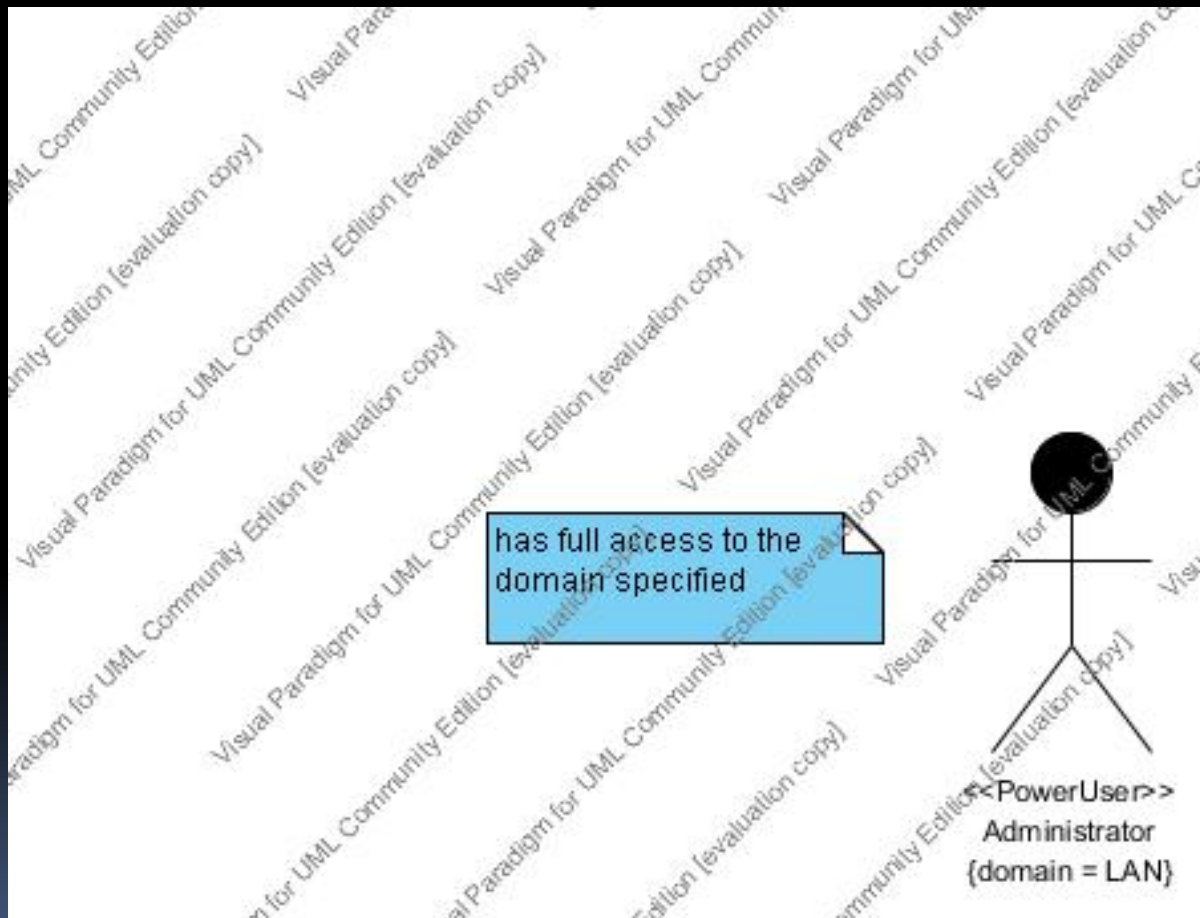
Механизмы расширения

После того, как стереотип определен, его можно использовать как элемент модели нового типа. Если при создании стереотипа не использовались дополнения и графическая нотация взята от базового элемента модели, на основе которого определен стереотип, то стереотип элемента обозначается при помощи имени стереотипа, заключенного в двойные угловые скобки (типографские кавычки), которое помещается перед именем элемента модели. Если же для стереотипа определена своя нотация, например, новый графический символ, то указывается этот символ.

Механизмы расширения

В UML имеется большое количество predetermined стереотипов. Стереотипы используются очень часто, поэтому примеры их применения рассредоточены по всей книге. Стереотипы являются мощным механизмом расширения языка, однако им следует пользоваться умеренно. Определяя большое число нестандартных стереотипов, легко сделать модель непонятной никому, кроме автора.

Механизмы расширения



Общие свойства модели

Модель в целом может обладать (или не обладать) важными свойствами, которые оказывают значительное влияние на ее практическую применимость. Исчерпывающим образом описать эти свойства во вступительном обзоре невозможно — их детализация рассредоточена по всей книге, но назвать и обозначить необходимо.

- правильность;
- непротиворечивость;
- полнота;
- вариации семантики.

Общие свойства модели

Прежде всего, модель должна удовлетворять формальным требованиям к описанию сущностей, отношений и их комбинаций. Т.е. модель должна быть синтаксически *правильной*.

Например, отношение (ребро в графе модели) всегда определяется между сущностями, на диаграмме линия должна начинаться и заканчиваться в фигуре, иначе это синтаксическая ошибка.

Общие свойства модели

В некоторых случаях даже синтаксически правильная модель может содержать такие конструкции, семантика которых не определена или неоднозначна.

Такая модель называется *противоречивой* (ill formed), а модель, в которой все в порядке и семантика всех конструкций определяется однозначно, называется *непротиворечивой* (well formed).

Общие свойства модели

Например, пусть мы определим в модели, что класс А является подклассом класса В, класс В — подкласс С, а класс С — подкласс А. Каждое из этих отношений обобщения в отдельности допустимо и синтаксически правильно, а все вместе они образуют противоречие.

Ответственность за непротиворечивость модели лежит на ее авторе.

Общие свойства модели

Модель не создается мгновенно — она появляется в результате многочисленных итераций и на каждой из них не полна.

В некоторых случаях оказывается достаточно одной диаграммы использования, а в других необходимы диаграммы всех типов, прорисованные до мельчайших деталей. Все зависит от прагматики, т. е. от того, для чего составляется модель.

Полнота – критерий субъективный.

Общие свойства модели

В описании семантики UML определено некоторое количество точек *вариации семантики*.

По сути авторы стандарта говорят: «мы понимаем это так-то и так-то, но допускаем, что другие могут это понимать иначе».

При реализации языка в конкретном инструменте разработчики в точке вариации семантики вправе выбрать альтернативный вариант, если он не противоречит семантике остальной части языка.

Рядовому пользователю точки вариации семантики не заметны и он может о них не думать.

Представления

Все аспекты моделируемой системы не удастся описать с единой точки зрения.

Моделировать сложную систему следует с нескольких различных точек зрения, каждый раз принимая во внимание один аспект моделируемой системы и абстрагируясь от остальных.

Этот тезис является одним из основополагающих принципов UML.

Представления

Выделим три представления:

- представление использования (что делает система полезного?);
- представление структуры (из чего состоит система?);
- представление поведения (как работает система?).

Представление поведения

Представление поведения призвано отвечать на вопрос: как работает система.

Определяющим признаком для отнесения элементов модели к представлению поведения является явное использования понятия времени, в частности, в форме описания последовательности событий/действий, то есть в форме алгоритма.

Описывается диаграммами состояний и деятельности, а также диаграммами взаимодействия в форме диаграмм кооперации и/или последовательности.

Представление поведения

- Модель поведения должна быть достаточно детальной для того, чтобы послужить основой для составления компьютерной программы.
- Модель поведения должна быть компактной и обозримой, чтобы служить средством общения между людьми в процессе разработки системы.
- Модель поведения не должна зависеть от особенностей реализации конкретных компьютеров, средств программирования и технологий.
- Средства моделирования поведения в UML должны быть знакомы и привычны большинству пользователей языка и не должны противоречить требованиям наиболее ходовых парадигм программирования.

Представление поведения

В UML предусмотрено несколько различных средств для описания поведения.

Выбор того или иного средства диктуется типом поведения, которое нужно описать.

Например, для описания жизненного цикла конкретного объекта используется конечный автомат в форме **диаграммы состояний**.

При этом состояния конечного автомата соответствуют состояниям объекта (различным наборам значений атрибутов), а переходы соответствуют выполнению операций.

Представление поведения

Диаграммы состояний можно составить не только для программных объектов — экземпляров отдельных классов, но и для более крупных конструкций, в частности, для всей модели приложения в целом или для более мелких — отдельных операций.

Для описания последовательности выполняемых элементарных шагов при выполнении отдельной операции или реализации сложного варианта использования, удобно использовать диаграммы деятельности.

Представление поведения

Взаимодействие нескольких программных объектов между собой описывается диаграммами взаимодействия в одной из двух эквивалентных форм (диаграммы кооперации и диаграммы последовательности).

Для объектно-ориентированной программы поведение прежде всего определяется взаимодействием объектов, поэтому диаграммы данного типа имеют столь важное значение при моделировании поведения в UML.

Диаграммы UML

Диаграмма состояний — это основной способ детального описания поведения в UML. В сущности, диаграммы состояний представляют собой граф состояний и переходов конечного автомата, нагруженный множеством дополнительных деталей и подробностей. Диаграммы состояний в UML являются реализацией основной идеи использования конечных автоматов как средства описания алгоритмов.

Диаграмма состояний



Конечный автомат (state machine) - 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.

Вершинами графа *конечного автомата* являются *состояния*. Дуги графа служат для обозначения *переходов* из *состояния* в *состояние*.

Диаграмма состояний



Диаграммы состояний UML более наглядны и выразительны по сравнению с классическими представлениями автоматов, но их применение требует большей подготовленности пользователя и предъявляет более высокие требования к "сообразительности" и "внимательности" инструментов моделирования.

Состояние



- На диаграммах состояний применяется всего один тип сущностей — состояния, и всего один тип отношений — переходы. Совокупность состояний и переходов между ними образует машину состояний.
- Машина состояний — термин, принятый в англоязычной литературе, но обозначающий тоже самое, что конечный автомат.

Состояние



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

Имя состояния

(a)

Имя состояния

список внутренних
действий в данном
состоянии

(б)

Диаграмма состояний



Состояния бывают: простые, составные, специальные

- Каждый тип состояний имеет дополнительные подтипы и различные составляющие элементы.

Переходы бывают простые и составные, и каждый переход может содержать :

- исходное состояние,
- событие перехода,
- сторожевое условие,
- действие на переходе,
- целевое состояние.

Состояние



- Простое состояние имеет следующую структуру:
- имя;
- действие при входе;
- действие при выходе;
- внутренняя активность.

Пример состояния с действиями



Аутентификация клиента

entry / получение пароля

do / проверка пароля

exit / отобразить меню опций

Состояние



- **Имя состояния является обязательным. Все остальные составляющие простого состояния не являются обязательными.**

Состояние



- Действие при входе (обозначается при помощи ключевого слова `entry`) — это указание атомарного действия, которое должно выполняться при переходе автомата в данное состояние.
- Действие при выходе (обозначается при помощи ключевого слова `exit`) — это указание атомарного действия, которое должно выполняться при переходе автомата из данного состояния. Действие при выходе выполняется до всех других действий, предписанных переходом, выводящим автомат из данного состояния.

Состояние



- Внутренняя активность (обозначается при помощи ключевого слова `do`) — это указание деятельности, которая начинает выполняться при переходе в данное состояние после выполнения всех действий, предписанных переходом, включая действие на входе. Внутренняя активность либо заканчивается по завершении, либо прерывается в случае выполнения перехода (в том числе и внутреннего перехода). В классической модели конечный автомат, находясь в некотором состоянии, ничего не делает: он находится в состоянии ожидания перехода.

Переход



- Простой переход всегда ведет из одного состояния в другое состояние.
- Существует несколько ограничений для специальных состояний: для начального состояния не может быть входящих переходов, а для заключительного — исходящих. В остальном переходы между состояниями могут быть определены произвольным образом.

Переход



- Прочие составляющие — событие перехода, сторожевое условие и действия на переходе не являются обязательными.
- Если они присутствуют, то изображаются в виде текста в определенном синтаксисе рядом со стрелкой, изображающей переход. Синтаксис описания перехода следующий:
- Событие [Сторожевое условие] / Действие

Переход



- Событие перехода — это тот входной символ (стимул), который вкупе с текущим состоянием автомата определяет следующее состояние.
- UML допускает наличие переходов без событий — такой переход называется переходом по завершении.

Переход



- Сторожевое условие — это логическое выражение, которое должно оказаться истинным для того, чтобы возбужденный переход сработал.
- Для каждого возбужденного перехода сторожевое условие проверяется ровно один раз, сразу после того, как переход возбужден и до того, как в системе произойдут какие-либо другие события.
- Если сторожевое условие ложно, то переход не срабатывает и событие теряется. Даже если впоследствии сторожевое условие станет истинным, переход сможет сработать только если повторно возникнет событие перехода.

Переход



- В UML предусмотрены синтаксические средства, до некоторой степени облегчающие семантически правильное построение сторожевых условий за счет более наглядного их изображения.
- Таковыми являются:
- сегментированные переходы;
- символы ветвления;
- переходные состояния;
- предикат else.

Переход

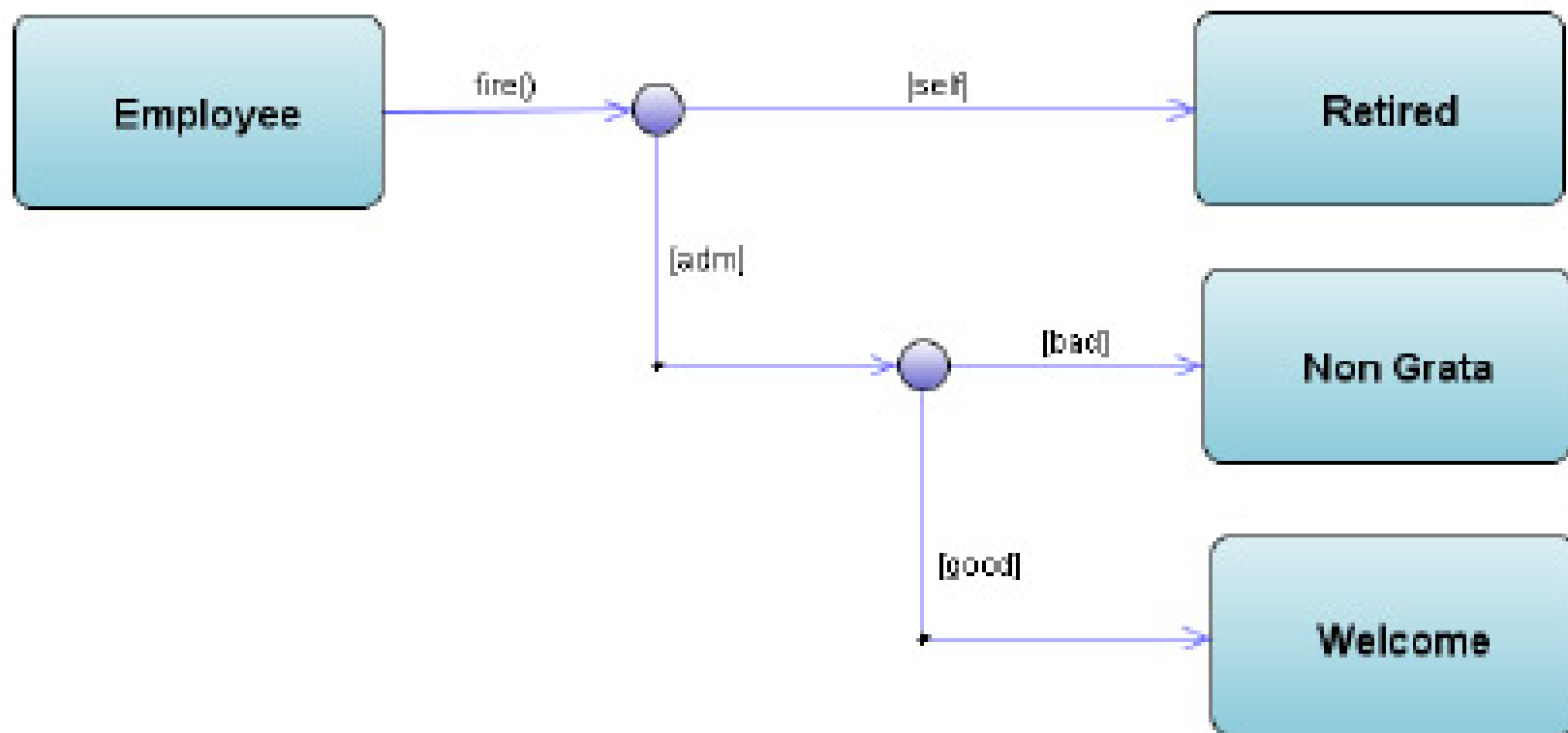


- Линия перехода может быть разбита на части, называемые сегментами.
- Сегменты перехода — части, на которые может быть разбита линия перехода.

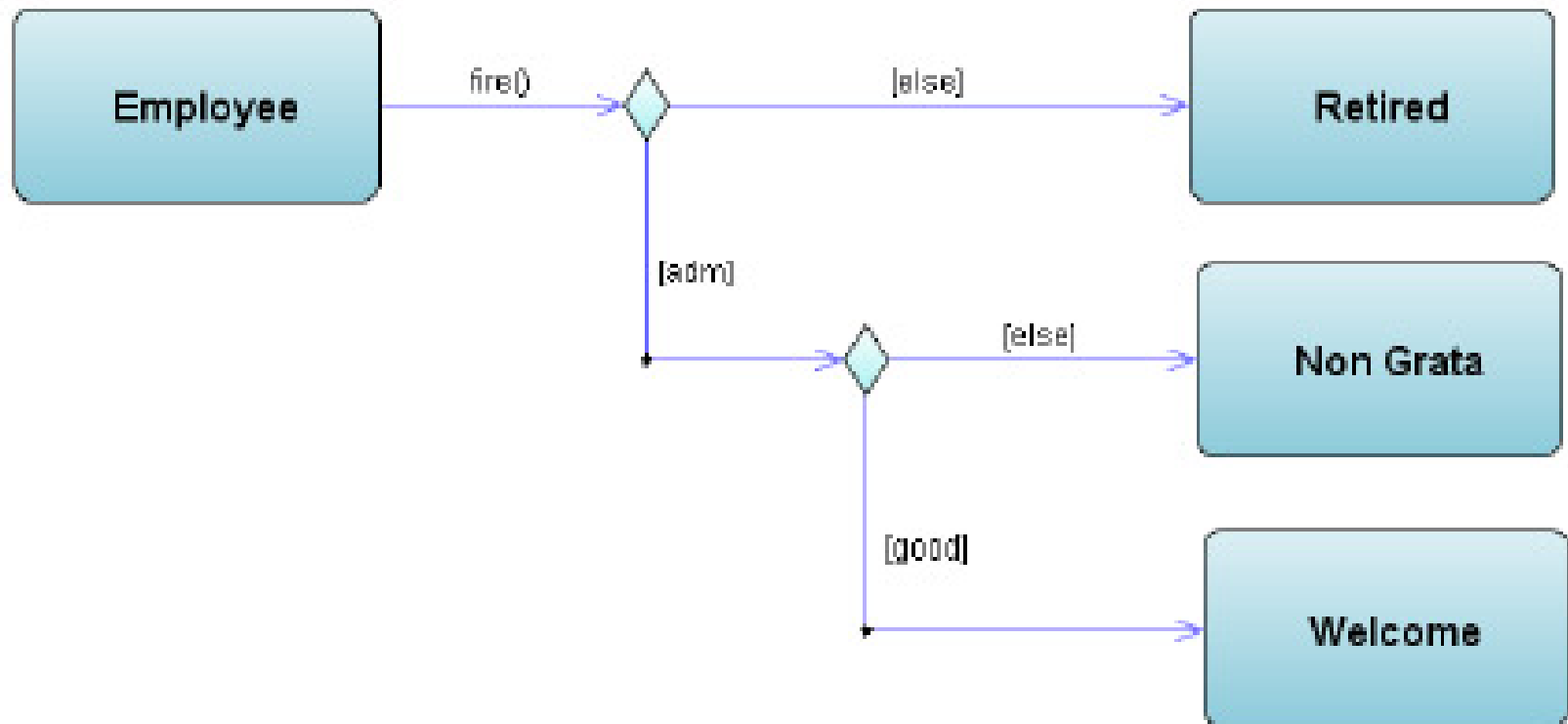
Разбивающими элементами являются следующие фигуры:

- переходное состояние (изображается в виде небольшого кружка);
- ветвление (изображается в виде ромба).

Примеры



Примеры



Составное состояние



- *Составное состояние — это состояние, в которое вложена машина состояний. Глубина вложенности в UML неограниченна, т. е. состояния вложенной машины состояний также могут быть составными.*

Составное состояние



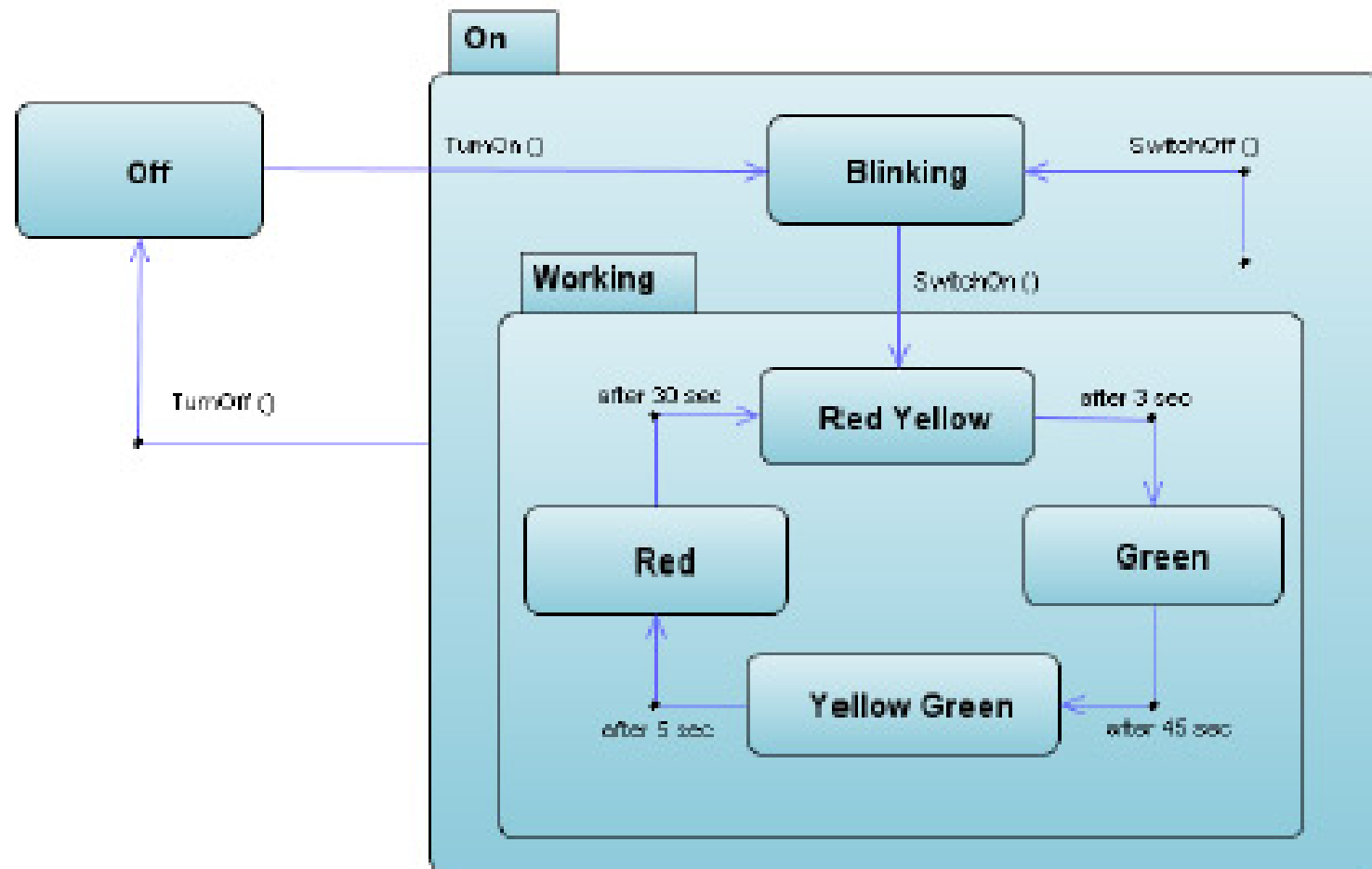
- Рассмотрим все известный прибор: светофор.
- Он может находится в двух основных состояниях:
- **Off** — вообще не работает — выключен или сломался, как слишком часто бывает;
- **On** — работает. Но работать светофор может по-разному:
- **Blinking** — мигающий желтый, дорожное движение не регулируется;
- **Working** — работает по-настоящему и регулирует движение.

Составное состояние



- В последнем случае у светофора есть 4 видимых состояния, являющихся предписывающими сигналами для участников дорожного движения:
- **Green** — зеленый свет, движение разрешено;
- **GreenYellow** — состояние перехода из режима разрешения в режим запрещения движения (это настоящее состояние, светофор находится в нём заметное время);
- **Red** — красный свет, движение запрещено;
- **RedYellow** — состояние перехода из режима запрещения в режим разрешения движения (это состояние отличное от **GreenYellow**, светофор подает несколько иные световые сигналы и участники движения обязаны по другому на них реагировать).

Составное состояние



Составное состояние

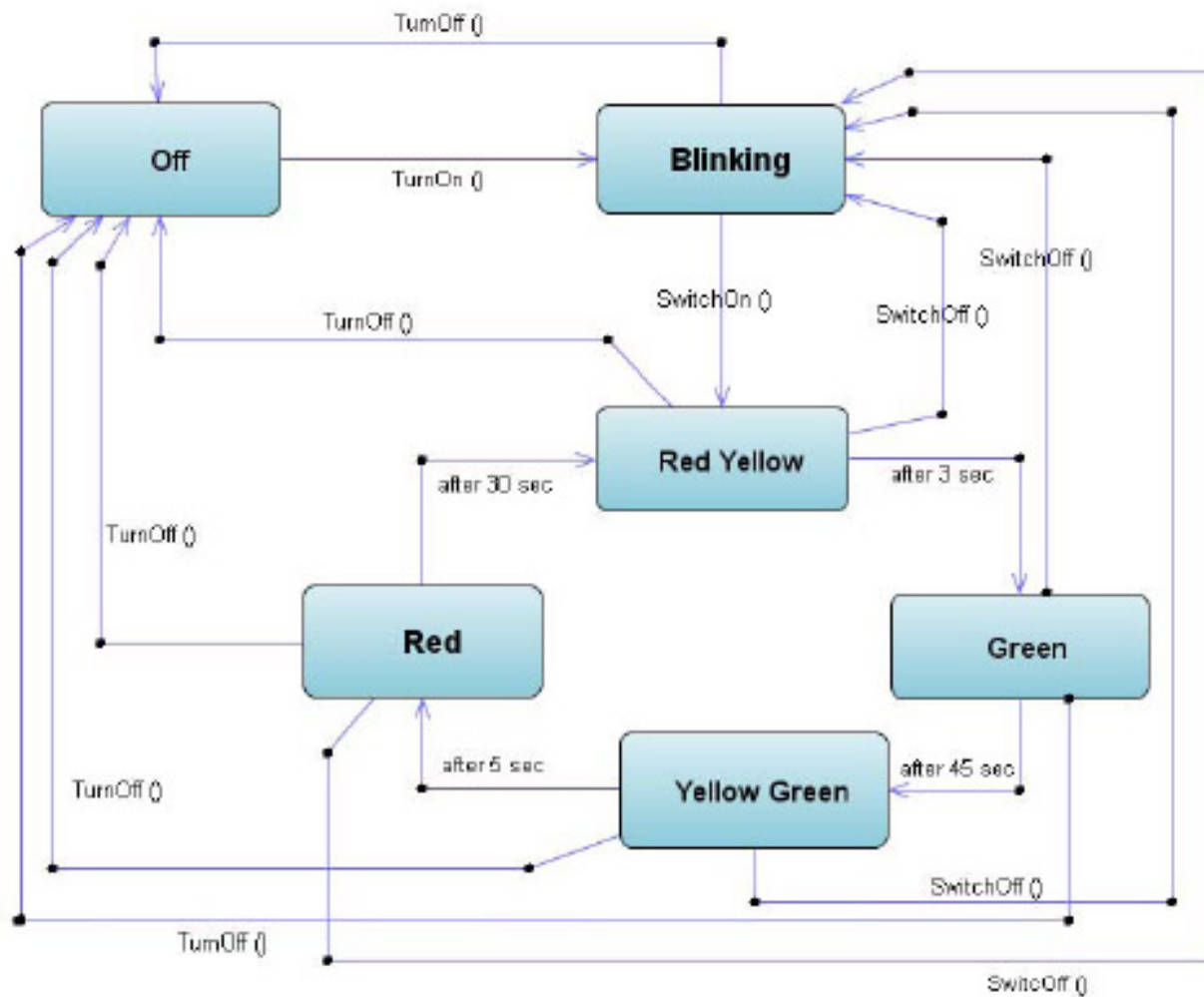
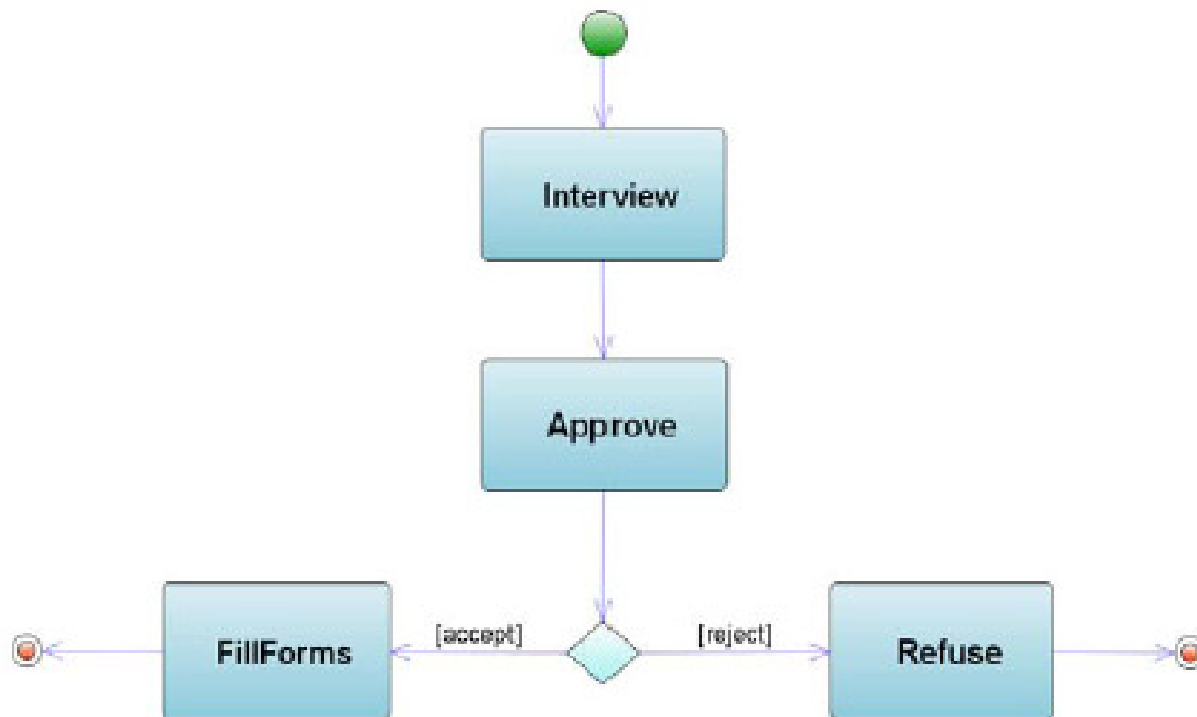


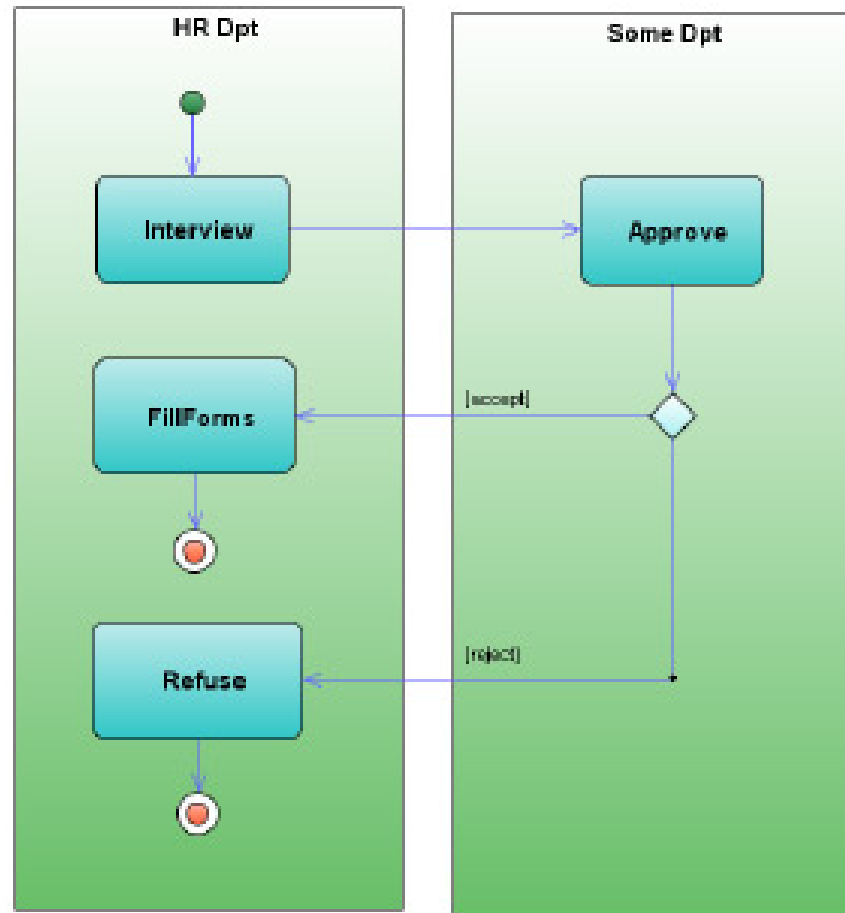
Диаграмма деятельности

Пример (без дорожек)



Диаграммы деятельности

Дорожка — это графический комментарий, позволяющий классифицировать по некоторому признаку сущности на диаграмме деятельности.



Выводы

- Модель поведения — это описание алгоритма работы системы.
- В UML предусмотрено несколько различных средств для описания поведения.
- Выбор того или иного средства диктуется типом поведения, которое нужно описать.
- Для моделирования поведения используются диаграмма состояний, диаграмма деятельности, диаграммы взаимодействия.