# Практическая работа №7

**Построение диаграммы активности**

Цель работы: освоение технологии проектирования ИС с помощью UML диаграмм

Задание: Ознакомиться с теоретическим материалом, разработать диаграмму активности в любом редакторе

Теоретический материал:

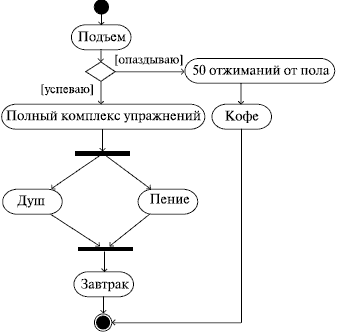
Д*иаграммы активностей* (*Activity Diagrams*) являются представлением алгоритмов неких действий (активностей), выполняющихся в системе.

Можно построить несколько диаграмм деятельности для одной и той же системы, причем каждая из них будет фокусироваться на разных аспектах системы, показывать различные действия, выполняющиеся внутри ее. Говоря более формально, диаграммы активности, в общем-то, не имеют монополии на описание поведенческих особенностей динамических частей системы. Для этой же цели могут использоваться еще *диаграммы прецедентов*, последовательности, кооперации и состояний. Почему же мы говорим именно о диаграмме активности? Нет, не только потому, что так называется эта лекция.

Именно на диаграмме деятельности представлены переходы потока управления от одной деятельности к другой. Это, *по* сути, разновидность диаграммы состояний, где все или большая часть состояний являются некоторыми деятельностями, а все или большая часть переходов срабатывают при завершении определенной деятельности и позволяют перейти к выполнению следующей. Как мы уже говорили (повторение - мать учения), *диаграмма* деятельности может быть присоединена к любому элементу модели, имеющему динамическое поведение. Кстати, исходя из вышесказанного, логичнее говорить не "*диаграмма* деятельности", а "*диаграмма* деятельностей" - во множественном числе. А еще мы предполагаем, что читатель понимает смысл понятий "*деятельность*", "переход" и "*объект*". Об объектах как об экземплярах классов мы уже говорили ранее. Понятия же деятельности (*activity*) как протяженного во времени составного (неатомарного) вычисления (действия, action) и перехода как передачи контроля, надеемся, понятны интуитивно, без дополнительных объяснений.

Диаграммы деятельности позволяют моделировать сложный **жизненный цикл объекта**, с переходами из одного состояния (деятельности) в другое. Но этот вид диаграмм может быть использован и для описания динамики совокупности объектов. Они применимы и для детализации некоторой конкретной *операции*, причем, как мы увидим далее, предоставляют для этого больше возможностей, чем "классическая" *блок-схема*. Диаграммы деятельности описывают переход *от одной деятельности к другой*, в отличие от диаграмм взаимодействия, где акцент делается на переходах потока управления *от объекта к объекту*.

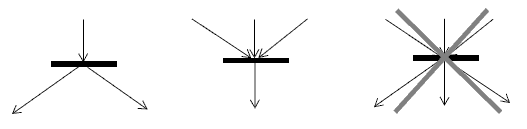
Как говорится, лучше один раз увидеть, чем сто раз услышать. Мы достаточно разрекламировали диаграммы деятельностей. Пора взглянуть на пример.



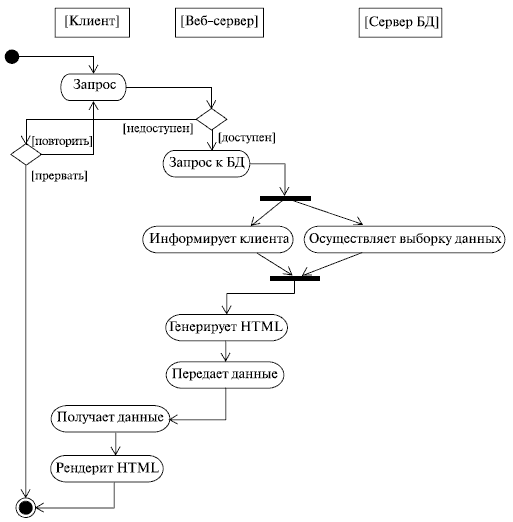
Эта *диаграмма* довольно точно описывает ежеутреннюю последовательность действий автора этих строк (до момента ухода на работу). Как видим, все очень просто и понятно. Действия показаны скругленными прямоугольниками, как в блок-схеме, - мы узнаем даже ромбик символа принятия решения с обозначениями условий возле переходов. Да, отличия от блок-схемы не так уж сильны. Более того, эти отличия выглядят как логичное расширение нотации блок-схем. Обратим внимание на то, что начало и конец уже не изображаются одинаковым безликим кружком. Начало теперь закрашено, а конец изображен в виде символа, напоминающего кошачий глаз (кстати, это образное название - "кошачий глаз" - уже намертво въелось в жаргон архитекторов и аналитиков).

http://www.intuit.ru/EDI/23_04_17_1/1492899714-28128/tutorial/356/objects/4/files/04_02.gif

Без пояснений понятен также смысл символа, предшествующего принятию душа и пению и следующего за ними - он означает *распараллеливание*, а затем опять слияние воедино ( *синхронизацию* ) потоков управления, т. е. *операции* "пение" и "душ" выполняются *одновременно*. *Нотация* проста: несколько потоков управления сливаются в один или один *поток* разделяется на несколько. Третьего не дано.

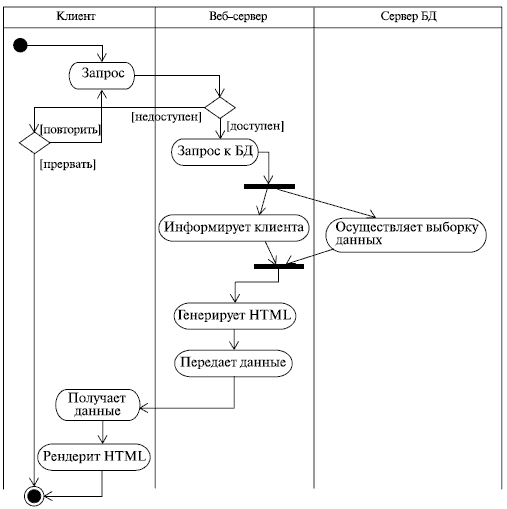


Конечно, это не единственные отличия *диаграммы активностей* от блок-схемы. На диаграмме деятельностей можно не только показать параллельно выполняемые действия, но и указать состояния объектов (так же, как и на представлениях конечных автоматов, о которых нам так много говорили в университетах), также есть возможность показывать распределение ролей и т. д. Вот еще пример, подтверждающий, что *диаграмма* активностей - это нечто большее, чем *блок-схема*.



Смысл диаграммы вполне понятен и без дополнительных объяснений. Как вы уже, конечно, догадались, на ней показана работа с веб-приложением, которое решает некую задачу в удаленной базе данных. Привлекает внимание странное расположение активностей на этой диаграмме: они как бы разбросаны *по* трем беговым дорожкам, каждая из которых соответствует поведению одного из трех объектов - клиента, веб-сервера и сервера баз данных. Благодаря этому легко определить, каким из объектов выполняется каждая из активностей, и неожиданно приходит понимание того, что "странность" этой диаграммы, оказывается, очень упрощает ее восприятие.

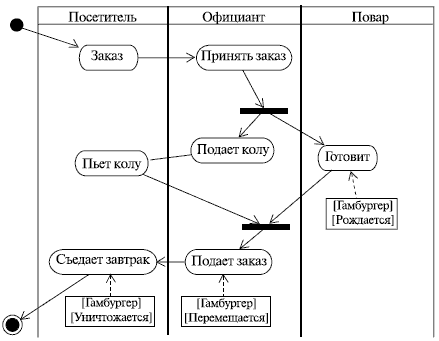
*Аналогия* с дорожками действительно очень удачна. Именно таково официальное название элемента нотации *UML*, позволяющего указать распределение ролей на диаграмме активностей. Только дорожки это не беговые, а плавательные - они так и называются: *swimlanes*. Более формально, дорожка - часть области диаграммы деятельности, на которой отображаются только те деятельности, за которые отвечает конкретный *объект*.



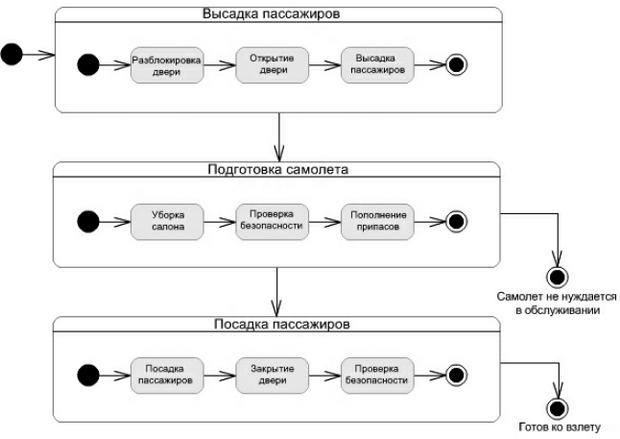
Предназначены они для разбиения диаграммы в соответствии с распределением ответственности за действия. Имя дорожки может означать роль или *объект*, которому она соответствует. При использовании дорожек *нотация* слегка изменяется. Вот как, к примеру, выглядит *диаграмма*из предыдущего примера, перерисованная с использованием дорожек.

Кстати, дорожки могут быть не только вертикальными, но и, если вам как автору так удобнее, горизонтальными. Изображаются горизонтальные дорожки аналогично - просто поверните "обычные" дорожки на 90 градусов против часовой стрелки!

Есть еще один нюанс нотации *диаграмм активностей*, о котором мы пока не говорили: это так называемая *траектория объекта*, или *поток объекта* ( object flow ). Суть его состоит в том, что на диаграмме деятельности можно изобразить и объекты, относящиеся к деятельности. С помощью символа зависимости (пунктирная стрелка, помните?) эти объекты можно соотнести с той деятельностью или переходом, где они создаются, изменяются или уничтожаются. Представим такую ситуацию из повседневной жизни: вы приходите в какой-нибудь фастфуд и заказываете гамбургер с колой. Что, знакомо? Во время приготовления завтрака повар создает новый *объект* - гамбургер. Пока вы нетерпеливо выпиваете колу, официант перемещает этот *объект* (подает ваш заказ). Естественно, во время завтрака вы уничтожаете этот *объект*. Вот как это выглядит на диаграмме .



На этом можно было бы и закончить наш разговор о нотации *диаграмм активностей* и их отличиях от блок-схем. Если бы не одно НО. Мы говорили, что *деятельность* - это протяженное *по* времени составное действие. Составное! То есть *составленное* из более простых действий. Вот эти-то самые простые (атомарные) действия, а вернее, последовательность их выполнения, частенько изображают внутри деятельности в виде маленькой *диаграммы активностей*. Это слегка напоминает матрешку - одна (а часто и не одна) *диаграмма* внутри другой. Мы не будем долго говорить об этом: нашей целью было просто обратить внимание читателя на подобную возможность "вложенных" диаграмм. Мы просто покажем пример, позаимствованный нами из Zicom Mentor



*Диаграмма* описывает высадку пассажиров самолета, достигших пункта назначения, и посадку новых пассажиров. Предлагаем читателю самому внимательно рассмотреть эту диаграмму. Из нее, например, можно почерпнуть, что конечных состояний может быть больше одного. Кстати, кроме начального и конечного состояний есть еще конечное состояние потока (*Flow* final *mode*). От конечного состояния оно отличается вот чем: конечное состояние потока означает завершение одного потока управления, а конечное состояние говорит о завершении всех потоков управления внутри деятельности. Обозначается конечное состояние потока простым символом, напоминающим лампочку накаливания в схемах электрических цепей



**Выводы**

* Диаграммой деятельности можно дополнить любой элемент модели, имеющий динамическое поведение.
* Диаграммы деятельности являются частным случаем диаграммы состояний.
* В отличие от блок-схем, диаграммы деятельности могут отображать одновременно выполняемые действия.
* На диаграммах активности можно использовать плавательные дорожки, распределяющие деятельности в соответствии с ролями (объектами), их выполняющими.
* Траектория объекта позволяет показать объекты, относящиеся к деятельности, и моменты переходов этих объектов из одного состояния в другое.
* Сложные деятельности можно дополнительно детализировать, разбив на действия и изобразив "диаграмму в диаграмме".
* Диаграммы деятельностей можно использовать для проектирования процессов (например, бизнес-процессов) или операций (вычислений). Во втором случае UML выступает в роли визуального языка программирования.

**Контрольные вопросы**

* Какие еще виды диаграмм (кроме *диаграмм активностей*) можно использовать для моделирования динамики системы?
* Чем диаграммы деятельности отличаются от блок-схем? Какие преимущества это сулит разработчикам?
* Что такое траектория объекта?
* Чем конечное состояние потока отличается от конечного состояния деятельности?
* Чем моделирование процессов отличается от моделирования операций?
* Применимы ли диаграммы деятельности безотносительно к ООП?