



{ПРОГРАММИРОВАНИЕ}



**UNIFIED  
MODELING**

*LANGUAGE*

# Урок №2

Диаграммы в UML,  
диаграммы вариантов  
использования

## Содержание

<b>1. Понятие диаграммы в UML.....</b>	<b>3</b>
<b>2. Краткий обзор существующих видов диаграмм .....</b>	<b>4</b>
2.1. Диаграмма вариантов использования (use case diagram) .....	4
2.2. Диаграмма классов (class diagram) .....	5
2.3. Диаграмма состояний (statechart diagram).....	5
2.4. Диаграмма деятельности (activity diagram).....	6
2.5. Диаграмма последовательности (sequence diagram) .....	7
2.6. Диаграмма кооперации (collaboration diagram)...	8
2.7. Диаграмма компонентов (component diagram)...	8
2.8. Диаграмма развертывания (deployment diagram).....	9
<b>3. Инструментарий для построения диаграмм .....</b>	<b>10</b>

# 1. Понятие диаграммы в UML

---

Итак, после первого урока вы владеете некоторым объемом информации о UML. Продолжим наше знакомство и разберем понятие диаграммы.

Что такое диаграмма? Знакомо ли вам данное понятие? Да, конечно, безусловно, воскликнете вы. Наверняка вы сможете привести примеры диаграмм из разных областей знаний (математики, экономики, физики, химии, ...). Например, диаграммы Фейнмана, диаграмма состояния и многие другие. Тот или иной вид диаграммы используется для демонстрации некоторого факта, взаимодействия между объектами, принципов работы. Какие цели ставятся перед диаграммами в UML? Диаграммы в UML используются для отображения связей, состояний, сценариев использования и поведения приложений, и так далее. То есть фактически цель набора диаграмм описать некоторое приложение (программный комплекс) с разных сторон.

В этом уроке мы проведем обзор типов UML диаграмм, проанализируем средства для построения UML диаграмм.

## 2. Краткий обзор существующих видов диаграмм

Целью данного раздела является краткий обзор существующих видов UML диаграмм. Важно сразу понимать, что у каждого вида диаграмм есть своя особая цель, ради которой данный вид диаграммы и был разработан.

### 2.1. Диаграмма вариантов использования (use case diagram)

Диаграмма вариантов использования (диаграмма прецедентов) — это вид диаграммы, который предназначен для того чтобы дать возможность обсуждать, анализировать поведение и функциональность приложения (программного проекта, системы) различным заинтересованным лицам (заказчик, конечный пользователь, программист, тестер).

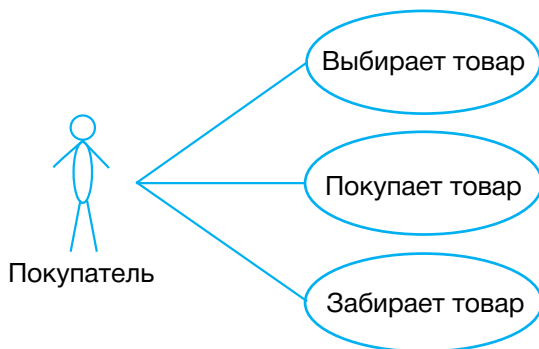


Рисунок 1.

Этот вид диаграммы очень прост и понятен любому человеку в силу применения самых простых обозначений. Фактически этот вид диаграмм отображает сценарий использования приложения (*use case*) в обычных человеческих терминах. Приведем пример типичной диаграммы (рис. 1).

## 2.2. Диаграмма классов (class diagram)

Диаграмма классов — диаграмма, которая описывает структуру приложения (программного проекта, системы). Цель данной диаграммы показать классы, их свойства (атрибуты, методы), связи между классами, интерфейсы и т. д. Приведем пример типичной диаграммы (рис. 2).

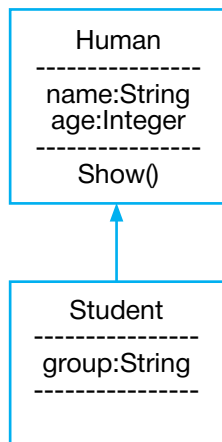


Рисунок 2.

## 2.3. Диаграмма состояний (statechart diagram)

Что такое состояние? Наверняка, вы знаете ответ на столь очевидный вопрос. Например: «Как твоё

состояние?», — это один из самых распространенных вопросов, который задают больному человеку. Естественно состояние обычно не константная величина, и оно меняется в зависимости от различных факторов. В случае больного состояние может улучшиться после приёма лекарств.

В UML диаграмма состояний показывает, как объект переходит из одного состояния в другое. Причем отображается процесс изменения состояний только конкретного объекта некоторого класса, в результате реакции на событие. Приведем пример типичной диаграммы (рис. 3).

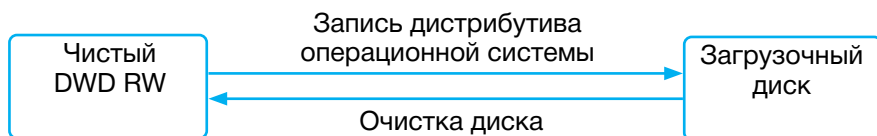


Рисунок 3.

## 2.4. Диаграмма деятельности (activity diagram)

В рамках курса «Язык программирования Си» вы познакомились с понятием блок-схемы. Основная цель блок-схемы состоит в наглядном изображении алгоритма решения некоторой задачи.

Диаграмма деятельности очень схожа с уже знакомыми вам блок-схемами. Этот вид диаграмм является частным случаем диаграмм состояний. Как и блок-схемы, они применяются для визуализации работы алгоритма, по которому запрограммирован некоторый класс.

Приведем пример типичной диаграммы (рис. 4).

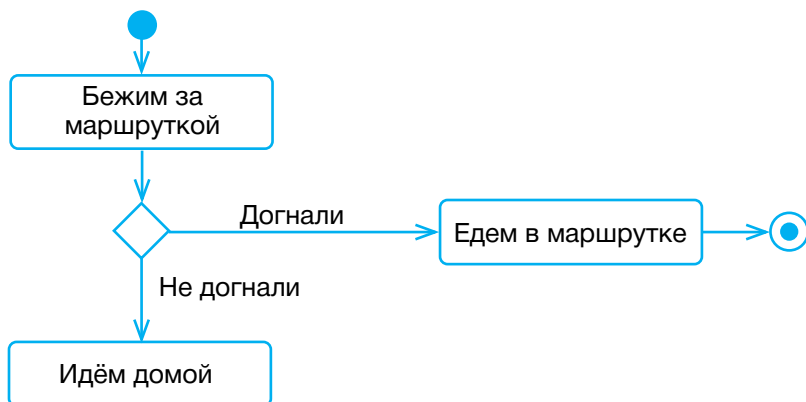


Рисунок 4.

## 2.5. Диаграмма последовательности (sequence diagram)

Этот вид диаграмм используется для отображения, упорядоченного во времени взаимодействия объектов. На ней показываются взаимодействующие объекты, а также последовательность сообщений, которыми они обмениваются. Приведем пример типичной диаграммы: (рис. 5).

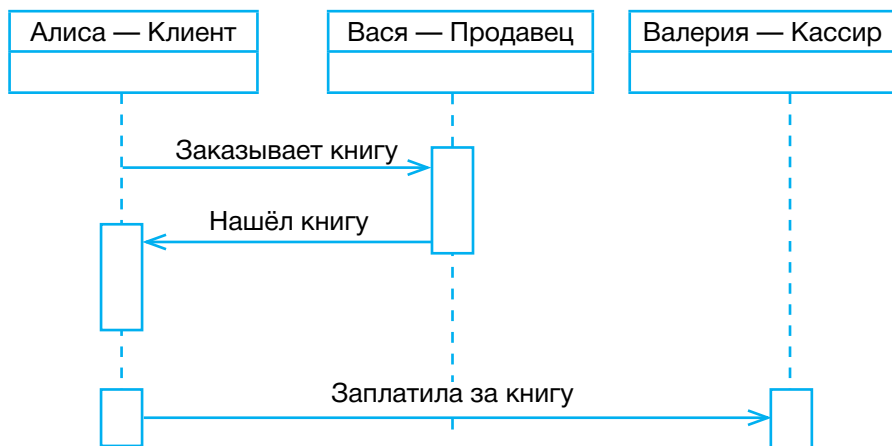


Рисунок 5.

## 2.6. Диаграмма кооперации (collaboration diagram)

Диаграмма кооперации (диаграмма взаимодействия) — отображает поток сообщений между объектами, составляющими систему, а также демонстрирует связи между ними. Эта диаграмма является аналогом диаграммы последовательностей.

Приведем пример типичной диаграммы (рис. 6).

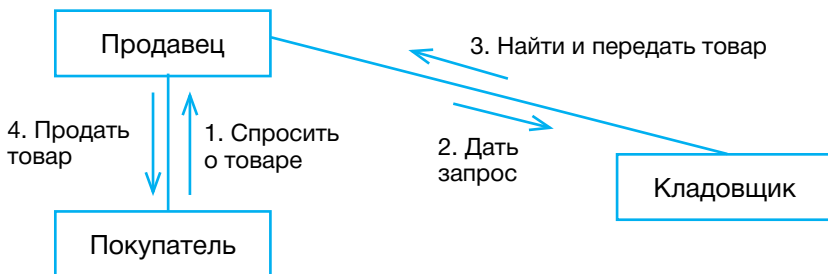


Рисунок 6.

## 2.7. Диаграмма компонентов (component diagram)

Данный вид диаграмм отображает разбиение программного проекта (системы) на множество структурных компонентов и связей между ними. В качестве компонентов могут выступать библиотеки, файлы, модули и так далее. Приведем пример типичной диаграммы (рис. 7).

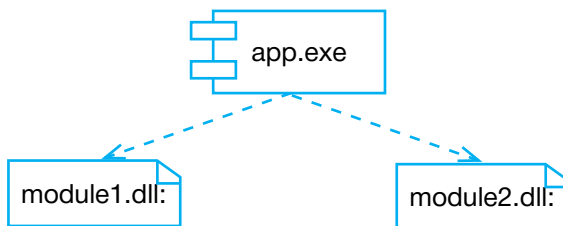


Рисунок 7.



## 2.8. Диаграмма развертывания (deployment diagram)

Из названия этой диаграммы уже понятно для чего она необходима. Когда говорят о развертывании, обычно описывают, что требуется выполнить для того, чтобы некоторое приложение могло работать. Диаграмма развертывания отображает конфигурацию узлов, производящих обработку информации, и описание компонентов размещенных в узлах.

Приведем пример типичной диаграммы (рис. 8).

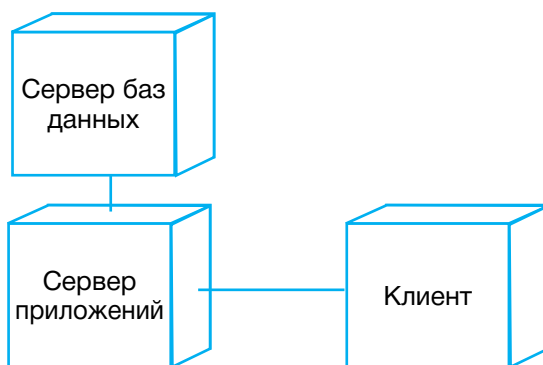


Рисунок 8.

Это был лишь краткий обзор существующих диаграмм. С каждой из них вы познакомитесь более детально в ближайшем будущем.

### 3. Инструментарий для построения диаграмм

Для построения диаграмм можно пользоваться различными пакетами. Среди них есть как платные, так и бесплатные решение. Например, компания Microsoft предоставляет замечательный продукт Visio, который можно использовать для разработки диаграмм разных типов, в частности и диаграмм UML.

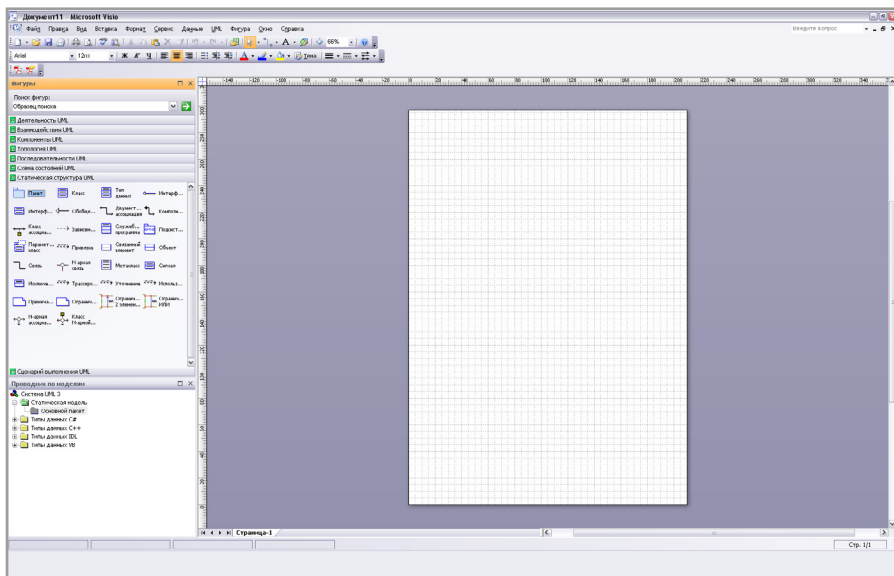


Рисунок 9.

Существует также большое количество OpenSource проектов, помогающих в нелегком деле проектирования. Например, такие проекты как ArgoUML, StarUML.

Также нельзя не упомянуть о культовом средстве разработки Rational Rose. Данный продукт пользуется большим уважением в среде разработчиков.

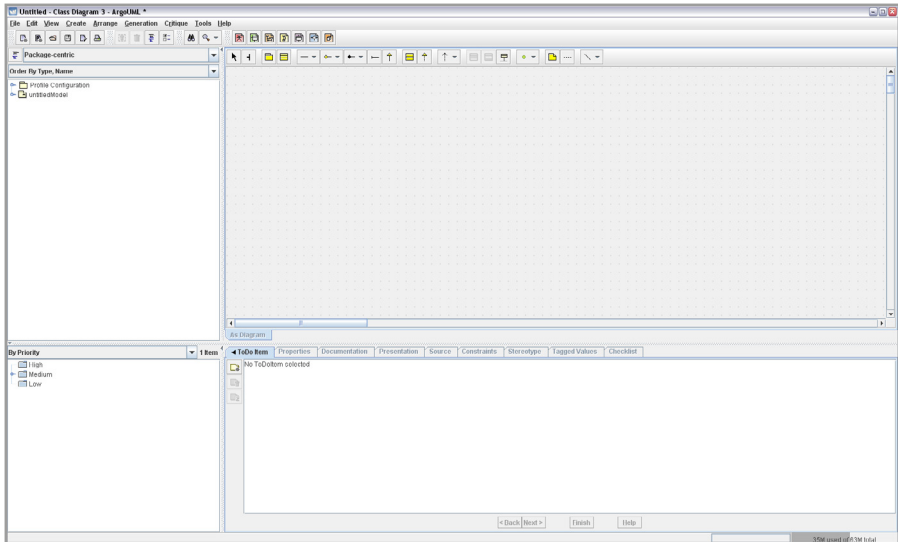


Рисунок 10.



## Урок №2

# Диаграммы в UML, диаграммы вариантов использования

© STEP IT Academy, [www.itstep.org](http://www.itstep.org)

Все права на охраняемые авторским правом фото-, аудио- и видеопроизведения, фрагменты которых использованы в материале, принадлежат их законным владельцам. Фрагменты произведений используются в иллюстративных целях в объёме, оправданном поставленной задачей, в рамках учебного процесса и в учебных целях, в соответствии с законодательством о свободном использовании произведения без согласия его автора (или другого лица, имеющего авторское право на данное произведение). Объём и способ цитируемых произведений соответствует принятым нормам, не наносит ущерба нормальному использованию объектов авторского права и не ущемляет законные интересы автора и правообладателей. Цитируемые фрагменты произведений на момент использования не могут быть заменены альтернативными, не охраняемыми авторским правом аналогами, и как таковые соответствуют критериям добросовестного использования и честного использования. Все права защищены. Полное или частичное копирование материалов запрещено. Согласование использования произведений или их фрагментов производится с авторами и правообладателями. Согласованное использование материалов возможно только при указании источника. Ответственность за несанкционированное копирование и коммерческое использование материалов определяется действующим законодательством.