Диаграмма Компонентов (Component Diagram)

Основное назначение логического представления состоит в анализе структурных и функциональных отношений между элементами модели системы. Для создания *конкретной физической системы* необходимо реализовать все элементы логического представления в *конкретные материальные сущности*. Другой аспект модельного представления — физическое представление модели.

Система считается *реализованной*, если она способна выполнять функции своего целевого предназначения. Программный код системы должен быть реализован в форме элементов физического представления системы:

- исполняемых модулей
- библиотек классов и процедур
- стандартных графических интерфейсов
- файлов баз данных.

Полный проект программной системы — совокупность согласованных моделей логического и физического представлений.

Физическое представление моделей систем — *диаграммы реализации* (implementation diagrams), включающие: *диаграмму компонентов* и *диаграмму развертывания*.

Диаграмма компонентов описывает особенности физического представления системы. Определяет архитектуру разрабатываемой системы. Устанавливает зависимости между программными компонентами: исходный, бинарный и исполняемый код. Во многих средах разработки модуль или компонент соответствует файлу.

Основные графические элементы — компоненты, интерфейсы и зависимости между ними.

Диаграмма компонентов разрабатывается для следующих целей:

- Визуализации общей структуры исходного кода программной системы.
- Спецификации исполнимого варианта программной системы.
- Обеспечения многократного использования отдельных фрагментов программного кода.
- Представления концептуальной и физической схем баз данных.

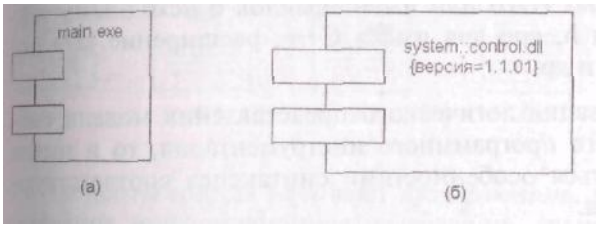
В разработке диаграмм компонентов участвуют системные аналитики, архитекторы и программисты. Диаграмма компонентов

- обеспечивает согласованный переход от логического представления к конкретной реализации проекта в форме программного кода;
- отражает общие зависимости между компонентами, рассматривая последние в качестве классификаторов.

1. Компоненты

Для представления физических сущностей в языке UML применяется специальный термин — *компонент* (component). *Компонент* реализует некоторый набор интерфейсов и служит для общего обозначения элементов физического представления модели.

Компонент может иметь также свои собственные свойства, такие как атрибуты и операции.



Имя компонента

Имя компонента состоит из любого числа букв, цифр и некоторых знаков препинания. Компонент может быть представлен на уровне типа или на уровне экземпляра. Графическое изображение в обоих случаях одинаковое. Правила записи имени компонента отличаются:

- если компонент представляется на уровне типа, то в качестве его имени записывается только имя типа с *заглавной буквы*;
- если компонент представляется на уровне экземпляра, то в качестве имени записывается <имя компонента '**:**' имя типа>. Вся стока имени подчеркивается.

В качестве простых имен принято использовать:

- имена исполняемых файлов (с указанием расширения **exe** после точки-разделителя);
- имена динамических библиотек (расширение **dll**);
- имена Web-страниц (расширение **html**);
- имена текстовых файлов (расширения **txt** или **doc**) или файлов справки (**hlp**);
- имена файлов баз данных (**DB**);
- имена файлов с исходными текстами программ (расширения **h**, **cpp** для языка C++, расширение **java** для языка **Java**);
- скрипты (**pl**, **asp**)
- и др.

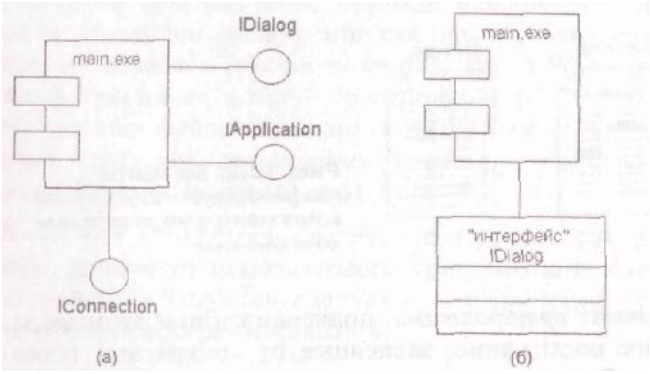
К простому имени компонента может быть добавлена информация об имени объемлющего пакета и о конкретной версии реализации данного компонента (см. рис.).

Виды компонентов

Компонент представляет отдельный модуль кода. Иногда его комментируют с указанием дополнительных графических символов, иллюстрирующих конкретные особенности его реализации. В языке UML выделяют три вида компонентов.

2. Интерфейсы

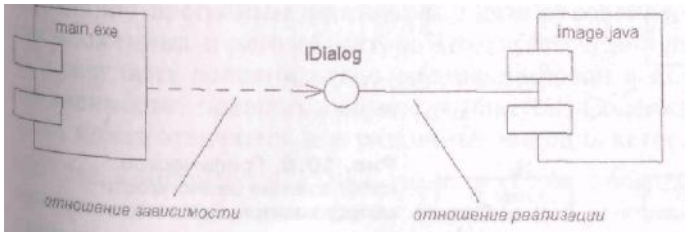
Интерфейс графически изображается окружностью, которая соединяется с компонентом отрезком линии без стрелок (см. рис.). Имя интерфейса должно начинаться с заглавной буквы "I". Записывается рядом с окружностью. Семантически линия означает реализацию интерфейса, а наличие интерфейсов у компонента означает, что данный компонент реализует соответствующий набор интерфейсов.



Другим способом представления интерфейса на диаграмме компонентов является его изображение в виде прямоугольника класса со стереотипом "интерфейс" и возможными секциями атрибутов и операций (см. рис.). Этот вариант обозначения используется для представления внутренней структуры интерфейса, которая может быть важна для реализации.

3. Зависимости

Зависимость не является ассоциацией, а служит для представления только факта наличия такой связи, когда изменение одного элемента модели оказывает влияние или приводит к изменению другого элемента модели.



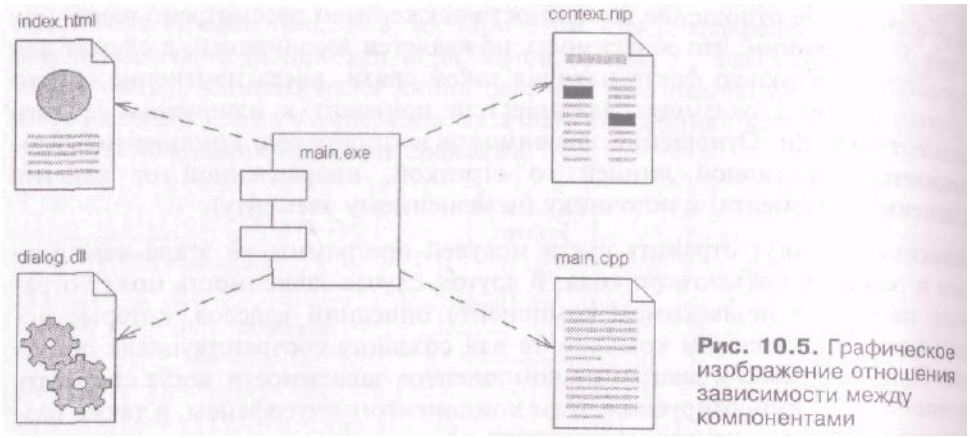


Рис. 10.5. Графическое изображение отношения зависимости между компонентами

На диаграмме компонентов могут быть представлены отношения зависимости между компонентами и реализованными в них классами для обеспечения согласования логического и физического представлений модели системы.

Диаграмма Развертывания (Deployment Diagram)

Физическое представление программной системы не может быть полным, если отсутствует информация о том, на какой платформе и на каких вычислительных средствах она реализована.

Первой из диаграмм физического представления является диаграмма компонентов. Второй формой физического представления программной системы является *диаграмма развертывания* (синоним — *диаграмма размещения*). Она применяется для представления общей конфигурации и топологии распределенной программной системы и содержит распределение компонентов по отдельным узлам системы. Кроме того, диаграмма развертывания показывает наличие физических соединений — маршрутов передачи информации между аппаратными устройствами, задействованными в реализации системы.

Диаграмма развертывания предназначена для визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения (run-time). При этом представляются только компоненты-экземпляры программы являющиеся *исполнимыми файлами* или *динамическими библиотеками*.

Диаграмма развертывания содержит графические изображения процессоров, устройств, процессов и связей между ними. *Диаграмма развертывания* является единой для системы в целом.

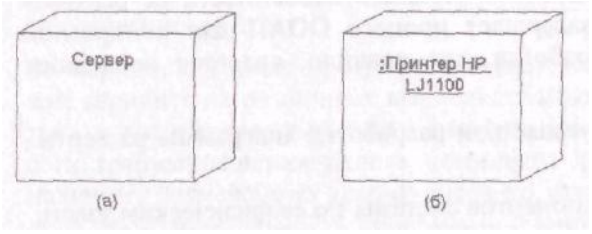
Цели, преследуемые при разработке диаграммы развертывания:

- Определить распределение компонентов системы по ее физическим узлам.
- Показать физические связи между всеми узлами реализации системы на этапе ее исполнения.
- Выявить узкие места системы и реконфигурировать ее топологию для достижения требуемой производительности.

Диаграмма развертывания разрабатывается совместно системными аналитиками, сетевыми инженерами и системотехниками.

1. Узел

Узел (**node**) представляет собой некоторый физически существующий элемент системы, обладающий некоторым вычислительным ресурсом.

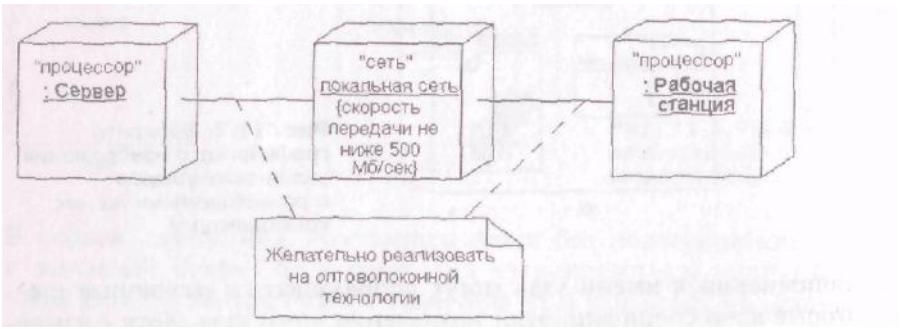


Изображения узлов могут расширяться, чтобы включить некоторую дополнительную информацию о спецификации узла.

2. Соединения

Кроме узлов на диаграмме развертывания указываются отношения между ними. В качестве отношений выступают физические соединения между узлами и зависимости между узлами и компонентами.

Соединения являются разновидностью ассоциации и изображаются отрезками линий без стрелок. Наличие такой линии указывает на необходимость организации физического канала для обмена информацией между соответствующими узлами.



Кроме соединений на диаграмме развертывания могут присутствовать