Диаграмма компонентов. Диаграмма кооперации. Диаграмма развертывания.

Содержание

1. Диаграмма компонентов
2. Диаграмма кооперации
3. Диаграмма развертывания

Диаграмма компонентов

статическая структурная диаграмма, которая показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и т. п.

Диаграмма компонентов разрабатывается для следующих целей:

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

Базовые элементы

«component» (компонент) - реализует некоторый набор интерфейсов и служит для общего обозначения элементов физического представления модели.

Имя компонента подчиняется общим правилам именования элементов модели в языке UML и может состоять из любого числа букв, цифр и некоторых знаков препинания.

Базовые элементы продолжение

Отдельный компонент может быть представлен на уровне типа или на уровне экземпляра.

Если компонент представляется на уровне типа, то в качестве его имени записывается только имя типа с заглавной буквы. Если же компонент представляется на уровне экземпляра, то в качестве его имени записывается <имя компонента>':'<имя типаX>. При этом вся строка имени подчеркивается.

В качестве простых имен принято использовать имена исполняемых файлов (с указанием расширения exe после точки-разделителя), динамических библиотек (расширение dll), Web-страниц (расширение html), текстовых файлов (расширения txt или doc) или файлов справки (hip), файлов баз данных (DB) или файлов с исходными текстами программ (расширения h, cpp для языка C++, расширение java для языка Java), скрипты (pi, asp) и другие.

Базовые элементы продолжение

В языке UML выделяют три вида компонентов:

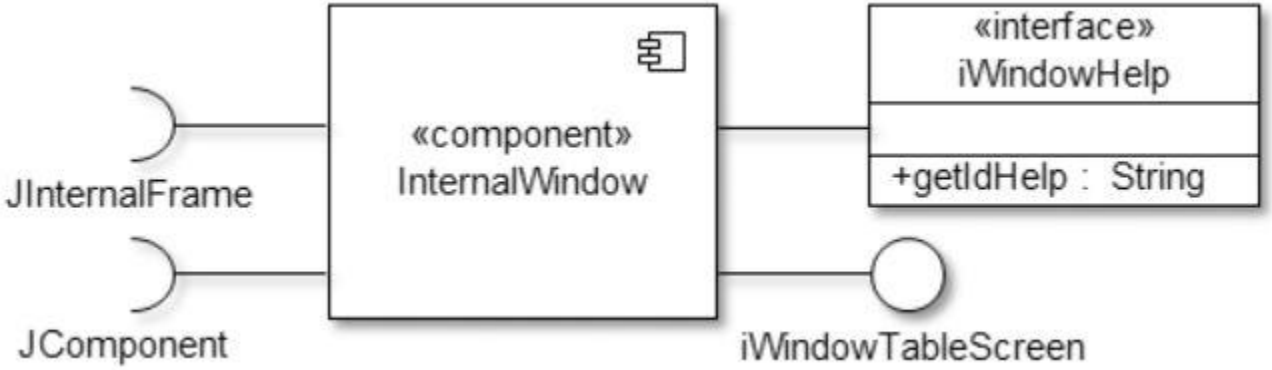
- развертывания, которые обеспечивают непосредственное выполнение системой своих функций. Такими компонентами могут быть динамически подключаемые библиотеки с расширением dll, Web-страницы на языке разметки гипертекста с расширением html и файлы справки с расширением hlp;
- рабочие продукты. Как правило, это файлы с исходными текстами программ, например, с расширениями h или cpp для языка C++;
- исполнения, представляющие собой исполняемые модули - файлы с расширением exe.

Базовые элементы продолжение

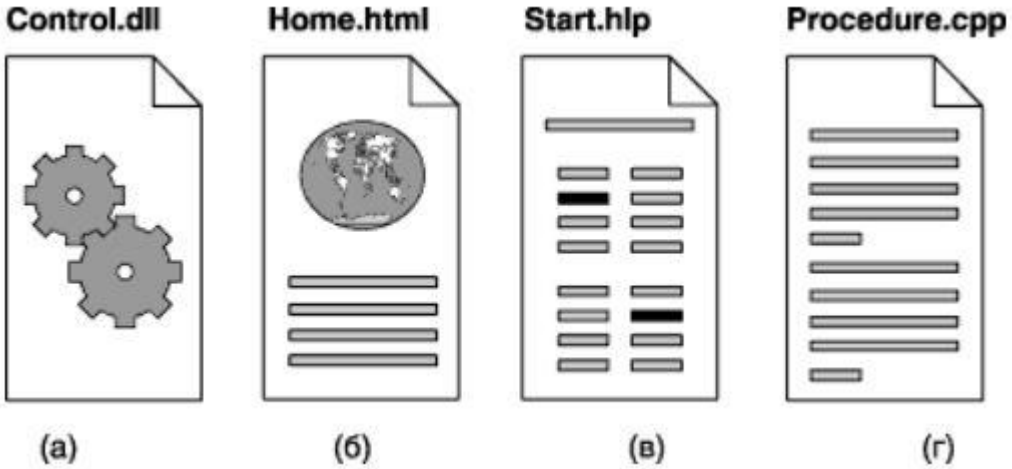
Интерфейс (англ. interface) – это внешне видимый, именованный набор операций, который класс, компонент или подсистема может предоставить другому классу, компоненту или подсистеме, для выполнения им своих функций.

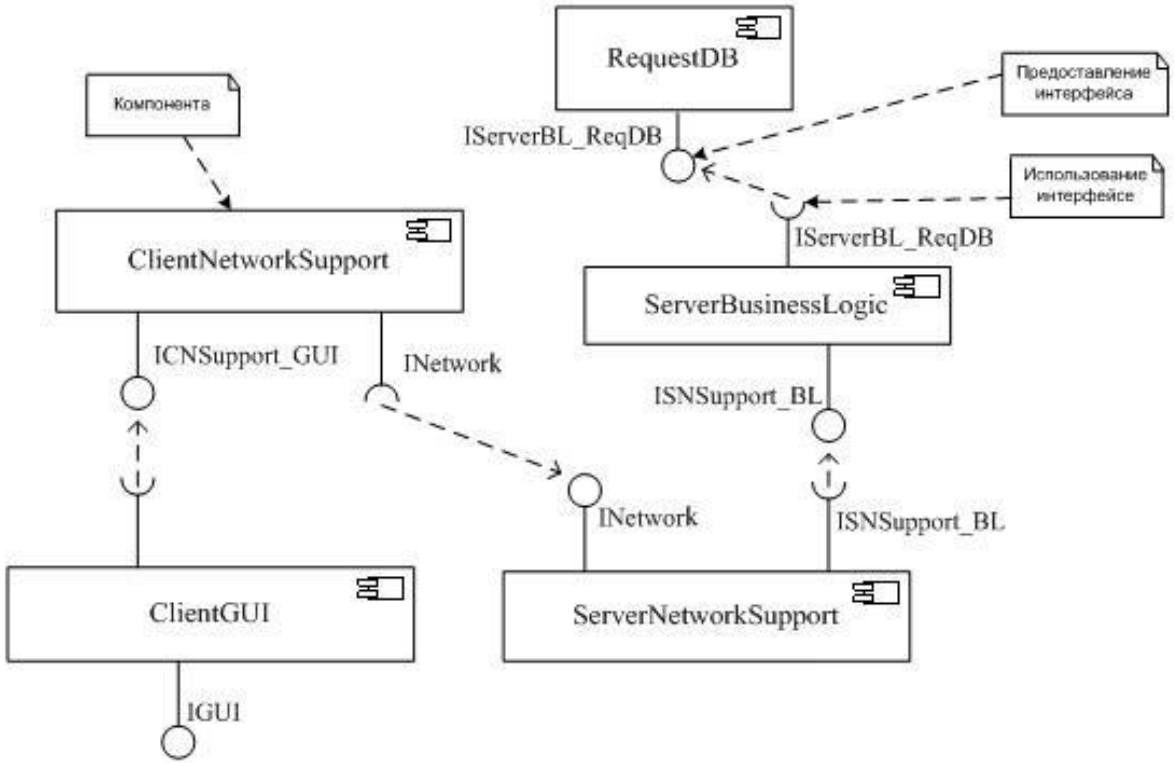
Отношение ассоциации отображается между компонентами и их интерфейсами. **Отношение зависимости** означает зависимость реализации одних компонентов от реализации других. Такое возможно в следующих случаях:

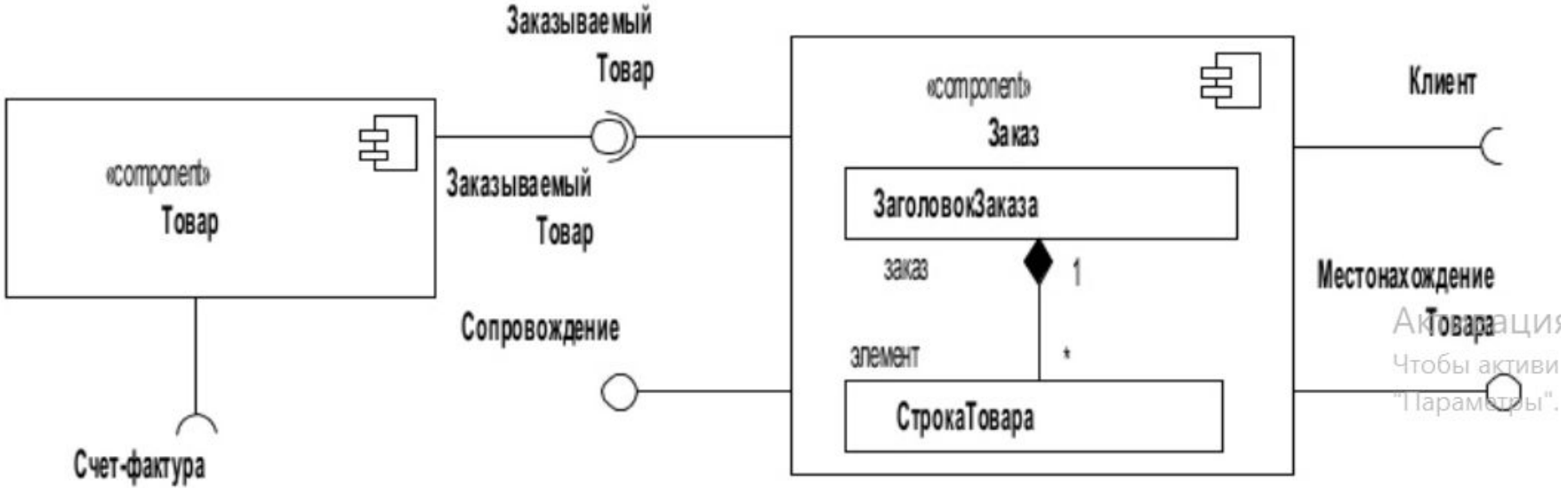
- в методах классов одного компонента (зависимого) осуществляется вызов методов или обращение к атрибутам классов другого компонента (независимого);
- компонент состоит из других компонентов (например, при сборке исполняемого файла из файлов с исходными кодами);
- компонент осуществляет чтение или запись данных в другой компонент;
- связь между таблицами БД;
- и т.д.



слева от компонента необходимые для работы интерфейсы, справа - предоставляемые




RequestDB
Предоставление интерфейса
Компонента
IServerBL_ReqDB
Использование интерфейса
IServerBL_ReqDB
ClientNetworkSupport
ServerBusinessLogic
ICNSupport_GUI
INetwork
ISNSupport_BL
INetwork
ISNSupport_BL
ClientGUI
ServerNetworkSupport
IGUI



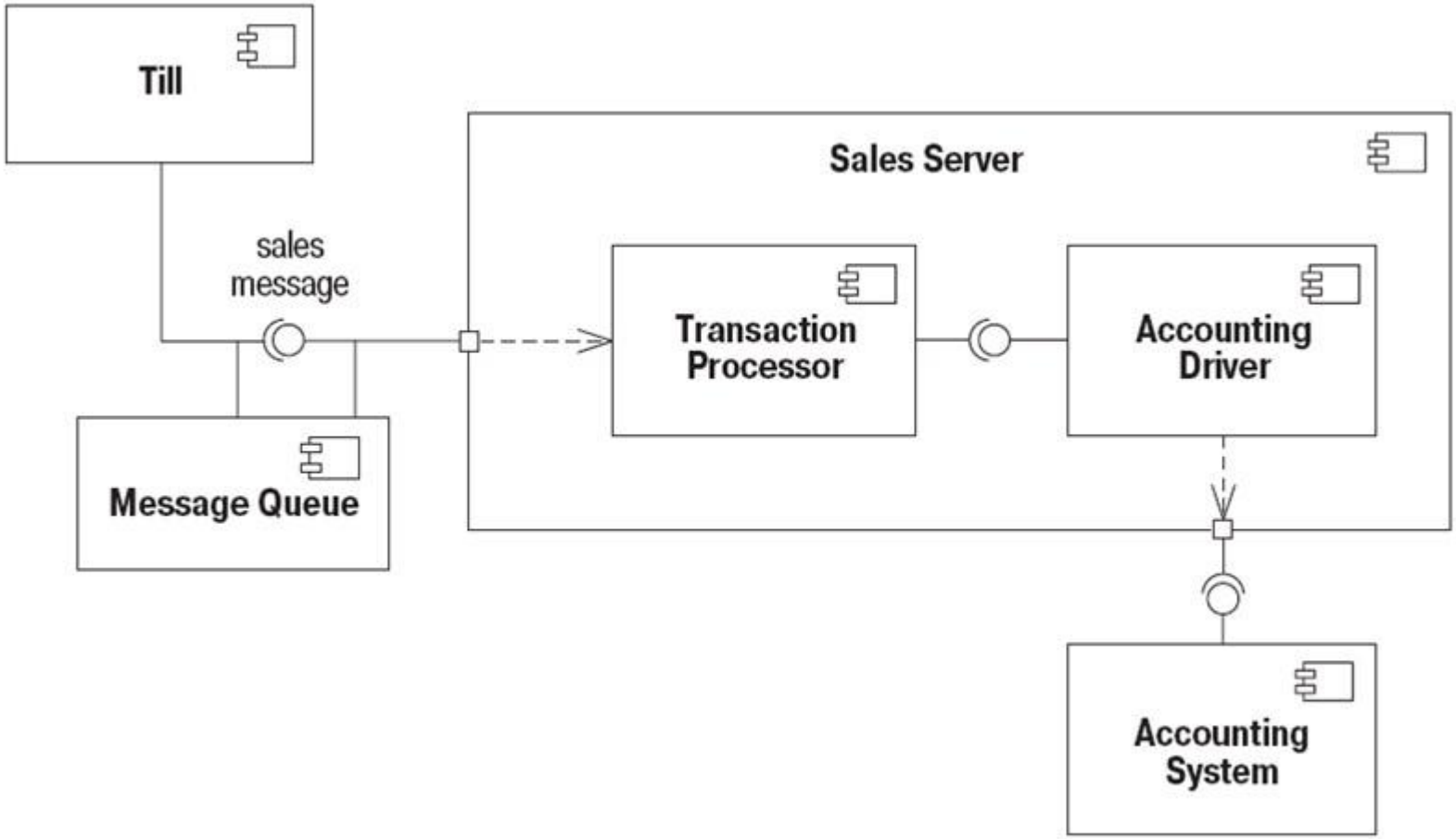
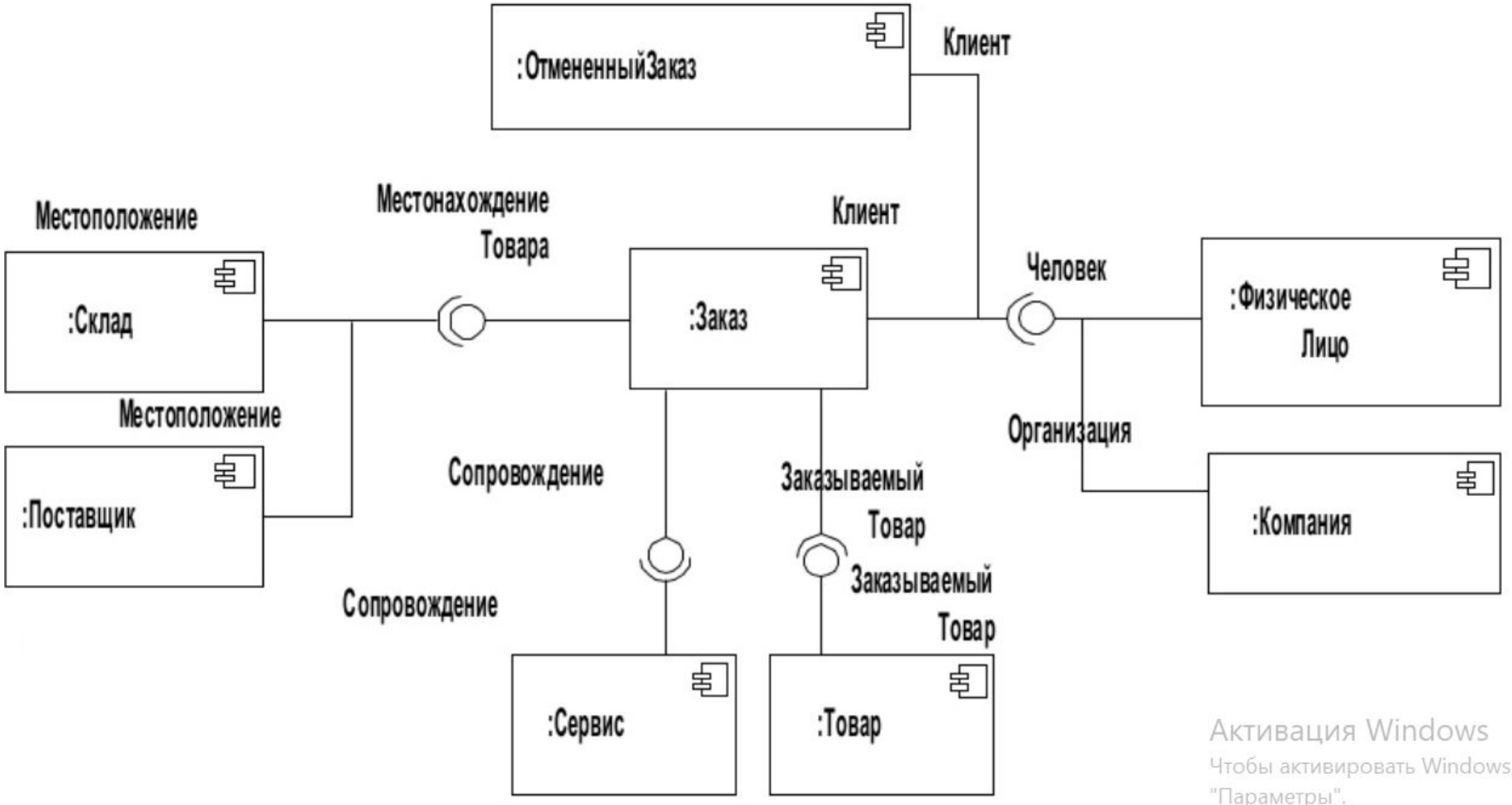
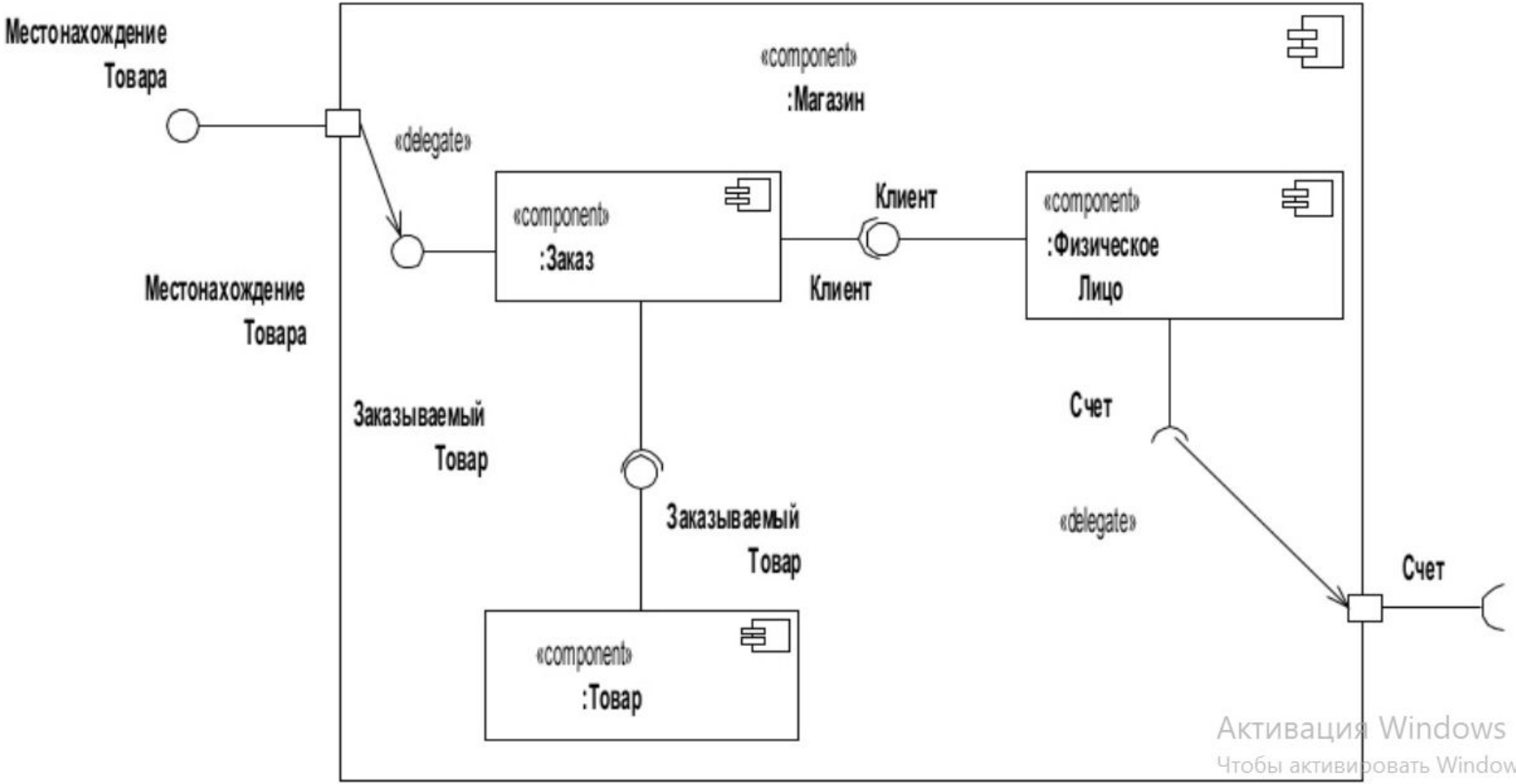


Рис. 14.2. Пример диаграммы компонентов





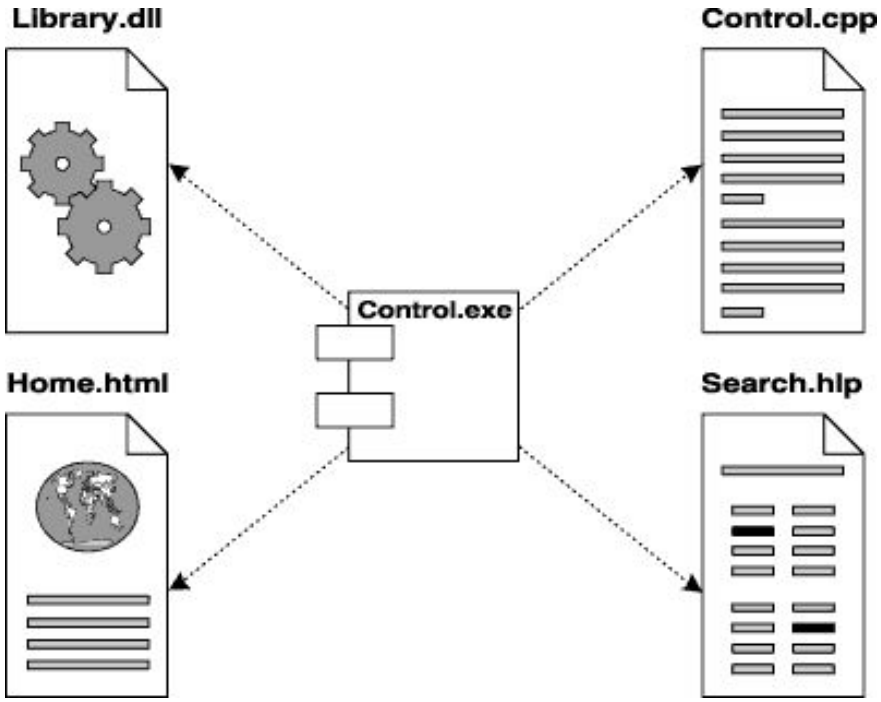

Library.dll
Control.cpp
Control.exe
Home.html
Search.hlp

Диаграмма кооперации

диаграмма, на которой изображаются взаимодействия между частями композитной структуры или ролями кооперации. В отличие от диаграммы последовательности, на диаграмме коммуникации явно указываются отношения между объектами, а время как отдельное измерение не используется (применяются порядковые номера вызовов).

Цель самой кооперации состоит в том, чтобы специфицировать особенности реализации отдельных вариантов использования или наиболее значимых операций в системе. Кооперация определяет структуру поведения системы в терминах взаимодействия участников этой кооперации.

Кооперация может быть представлена на двух уровнях:

- На уровне спецификации - показывает роли классификаторов и роли ассоциаций в рассматриваемом взаимодействии.
- На уровне примеров - указывает экземпляры и связи, образующие отдельные роли в кооперации.

Объекты

объект является отдельным экземпляром класса, который создается на этапе выполнения программы

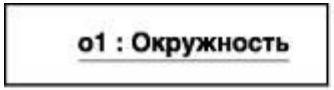
<Имя объекта>'/' <Имя роли классификатора> ':' <Имя классификатора>

роль классификатора - описание множества объектов

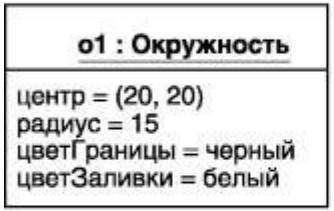
классификатор - ограничение типа объектов

[':' <Имя классификатора >]*

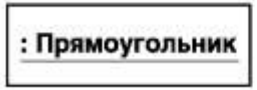
двоеточие всегда должно стоять перед именем класса, а косая черточка - перед именем роли



(а)

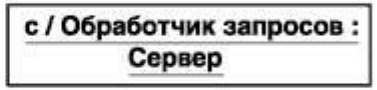


(б)



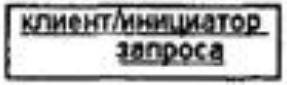
(в)

(г)

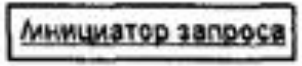


(д)

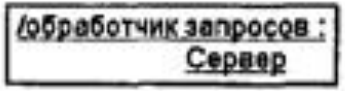
(е)



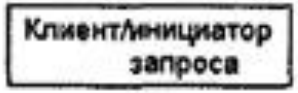
(а)



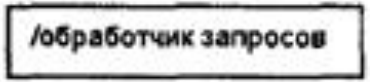
(б)



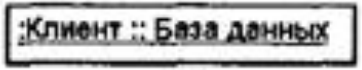
(в)



(г)



(д)

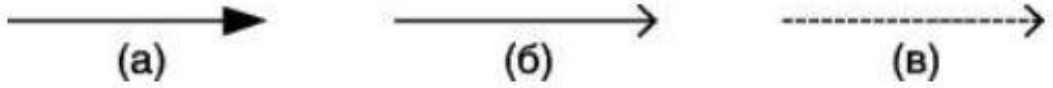


(е)

варианты записи строки текста в прямоугольнике объекта

- : C - анонимный объект, образуемый на основе класса C.
- / R - анонимный объект, играющий роль R.
- / R : C - анонимный объект, образуемый на основе класса C и играющий роль R.
- O / R - объект с именем O, играющий роль R.
- O : C - объект с именем O, образуемый на основе класса C.
- O / R : C - объект с именем O, образуемый на основе класса C и играющий роль R.
- O или - объект с именем O.
- O : - "объект-сирота" с именем O.
- / R - роль с именем R
- : C - анонимная роль на базе класса C.
- / R : C - роль с именем R на основе класса C.

Связи

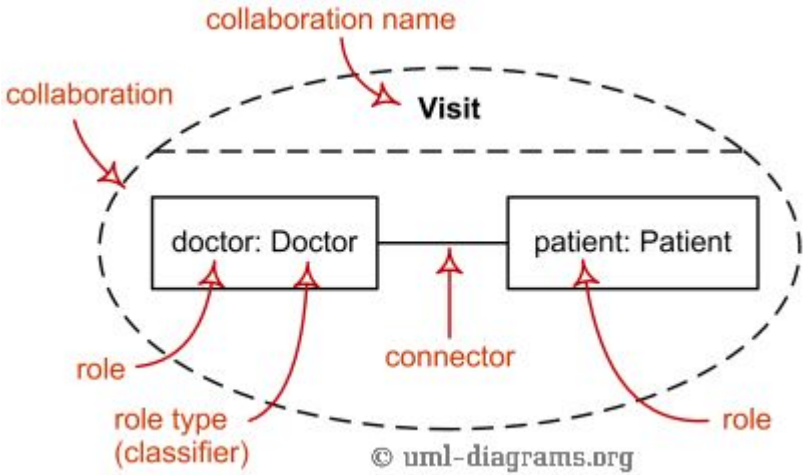


Сплошная линия с треугольной стрелкой (рис. а) обозначает вызов процедуры (операции) или передачу потока управления. Сообщения этого типа могут быть использованы параллельно активными объектами, когда один из них передает сообщение этого типа и ожидает, пока не закончится некоторая последовательность действий, выполняемая вторым объектом. Обычно все такие сообщения синхронны, т. е. инициируются по завершении деятельности или при выполнении определенного условия.

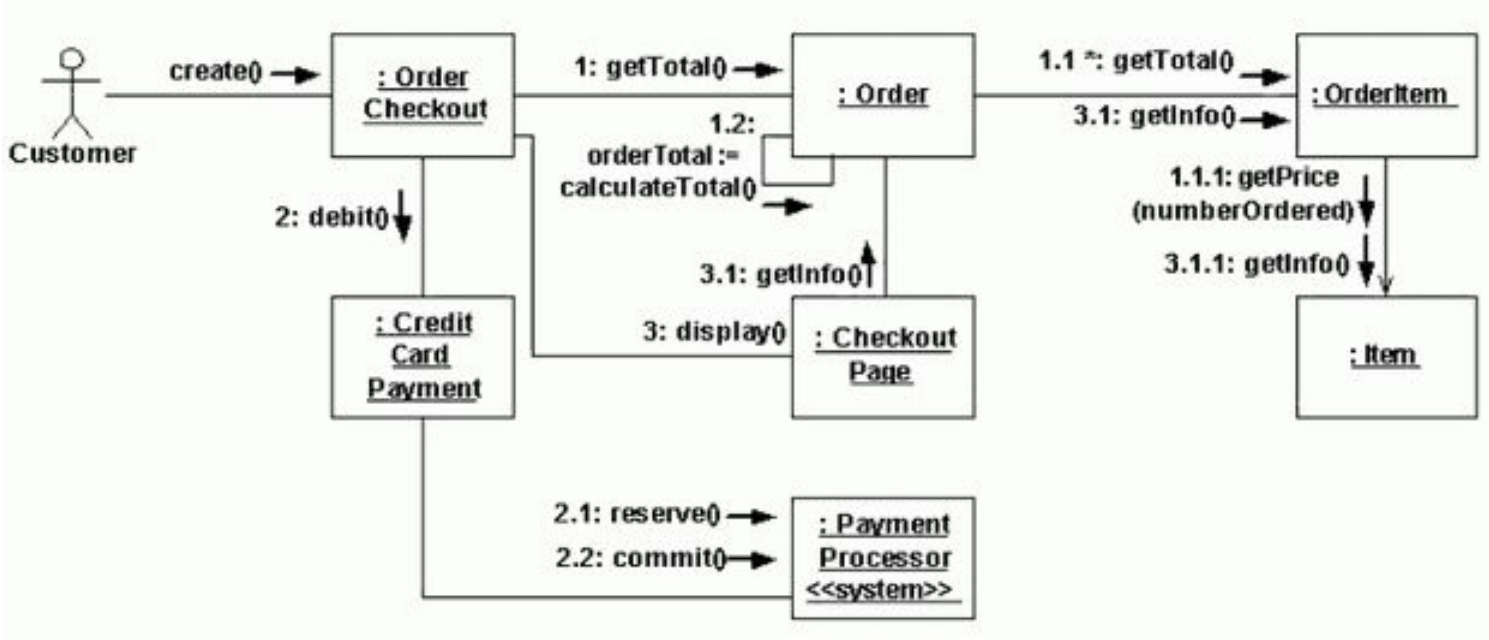
Сплошная линия с V-образной стрелкой (рис. б) обозначает асинхронное сообщение в простом потоке управления. В этом случае клиент передает асинхронное сообщение и продолжает выполнять свою деятельность, не ожидая ответа от клиента.

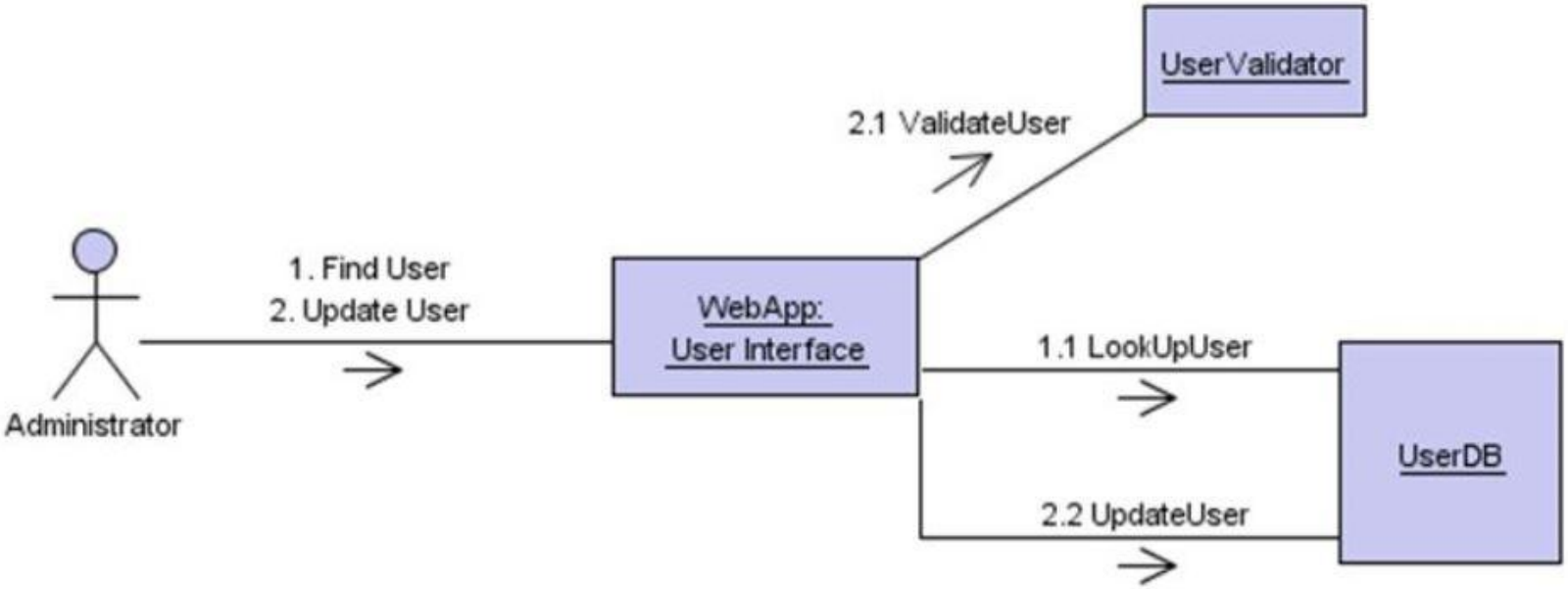
Пунктирная линия с V-образной стрелкой (рис. в) обозначает возврат из вызова процедуры. Стрелки этого типа зачастую отсутствуют на диаграммах кооперации, поскольку неявно предполагается их существование после окончания процесса выполнения операции или деятельности.

- На уровне спецификации


collaboration name
collaboration
Visit
doctor: Doctor
patient: Patient
role
role type
(classifier)
connector
role
© uml-diagrams.org

- На уровне примеров




UserValidator
2.1 ValidateUser
1. Find User
2. Update User
WebApp:
User Interface
1.1 LookUpUser
Administrator
UserDB
2.2 UpdateUser


С : Компания
1. Создать
2. Выдать задание
3. Расформировать
2.3. Сдать отчет
4. Принять
5. Перевести в другой отдел
6. Уволить
2.1. Выдать задание
2.2. Принять работу
: Отдел
: Сотрудник
2.1.1. Выполнить

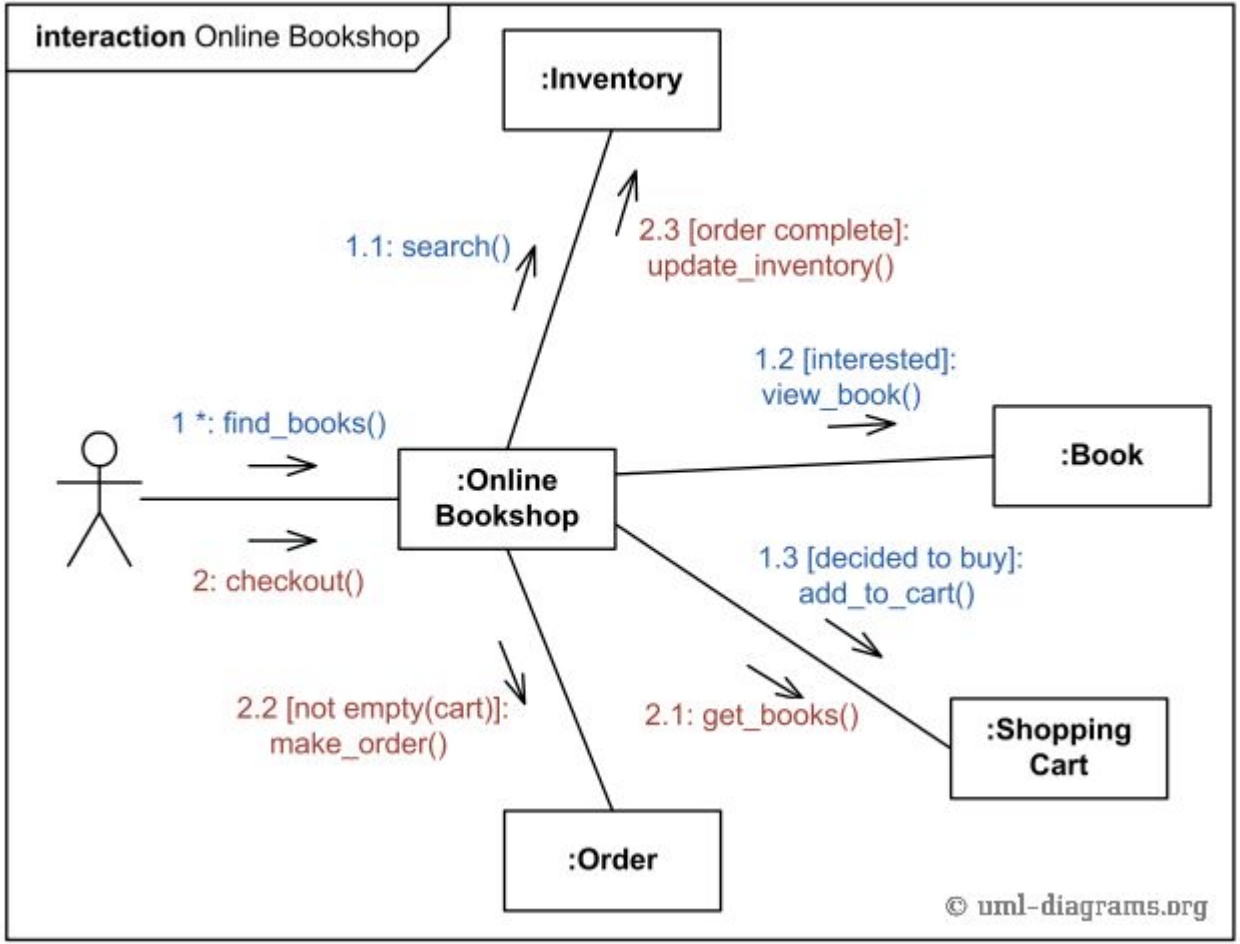

interaction Online Bookshop
:Inventory
1.1: search()
2.3 [order complete]:
update_inventory()
1.2 [interested]:
view_book()
1 *: find_books()
:Online
Bookshop
:Book
2: checkout()
1.3 [decided to buy]:
add_to_cart()
2.2 [not empty(cart)]:
make_order()
2.1: get_books()
:Shopping
Cart
:Order
© uml-diagrams.org

Диаграмма развертывания

- моделирует физическое развертывание артефактов на узлах
- Используется для представления физического расположения системы, показывая, на каком физическом оборудовании запускается та или иная составляющая программного обеспечения

Главными элементами диаграммы являются узлы, связанные информационными путями.

Узел (node) – это то, что может содержать программное обеспечение.

Узлы бывают двух типов:

1. Устройство (device) – это физическое оборудование: компьютер или устройство, связанное с системой.
2. Среда выполнения (execution environment) – это программное обеспечение, которое само может включать другое программное обеспечение, например операционную систему или процесс контейнер.

Продолжение

Узлы могут содержать артефакты (artifacts), которые являются физическим олицетворением программного обеспечения; обычно это файлы.

Такими файлами могут быть исполняемые файлы (такие как файлы .exe, двоичные файлы, файлы DLL, файлы JAR, сборки или сценарии) или файлы данных, конфигурационные файлы, HTML-документы и т. д. Перечень артефактов внутри узла указывает на то, что на данном узле артефакт разворачивается в запускаемую систему.

Артефакты можно изображать в виде **прямоугольников классов** или **перечислять их имена внутри** узла. Если вы показываете эти элементы в виде прямоугольников классов, то можете добавить значок документа или ключевое слово «artifact». Можно сопровождать узлы или артефакты значениями в виде **меток**, чтобы указать различную информацию об узле, например поставщика, операционную систему, местоположение.

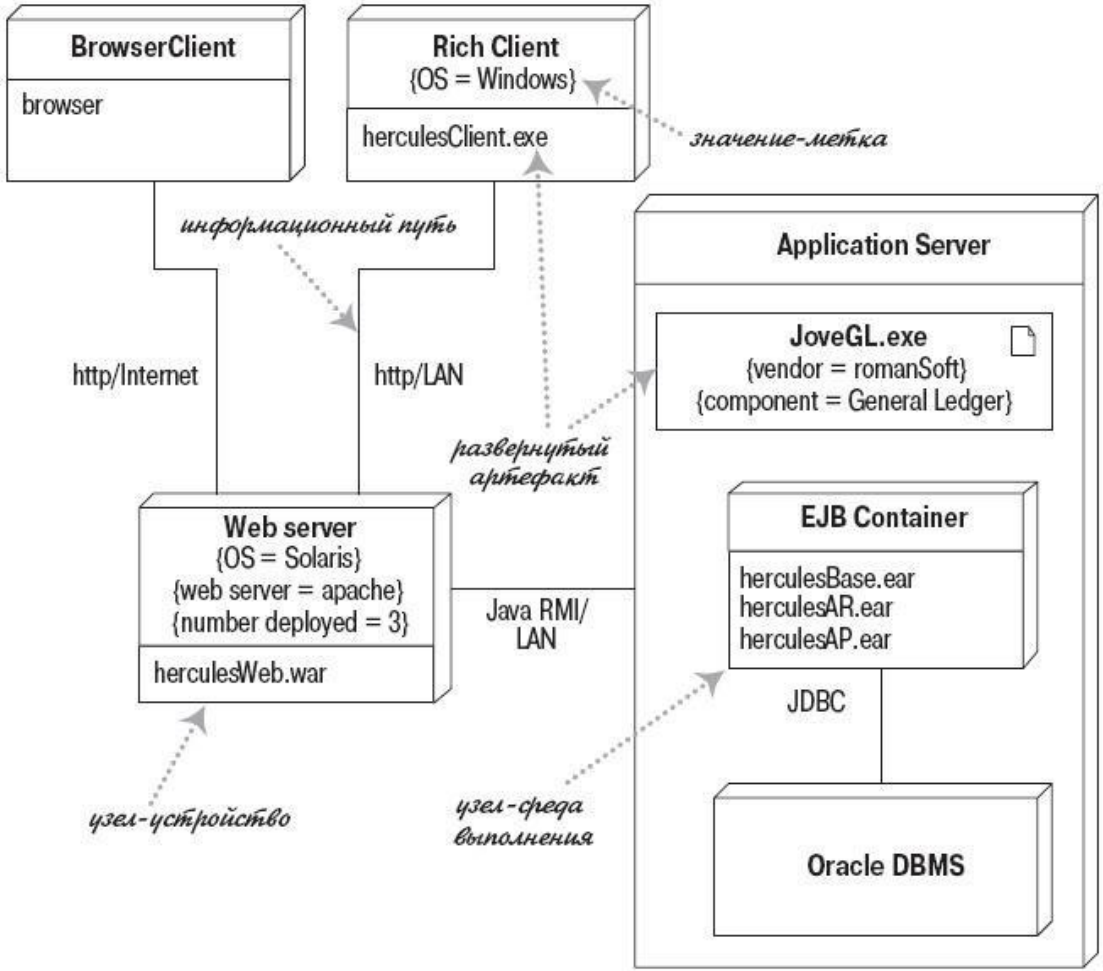
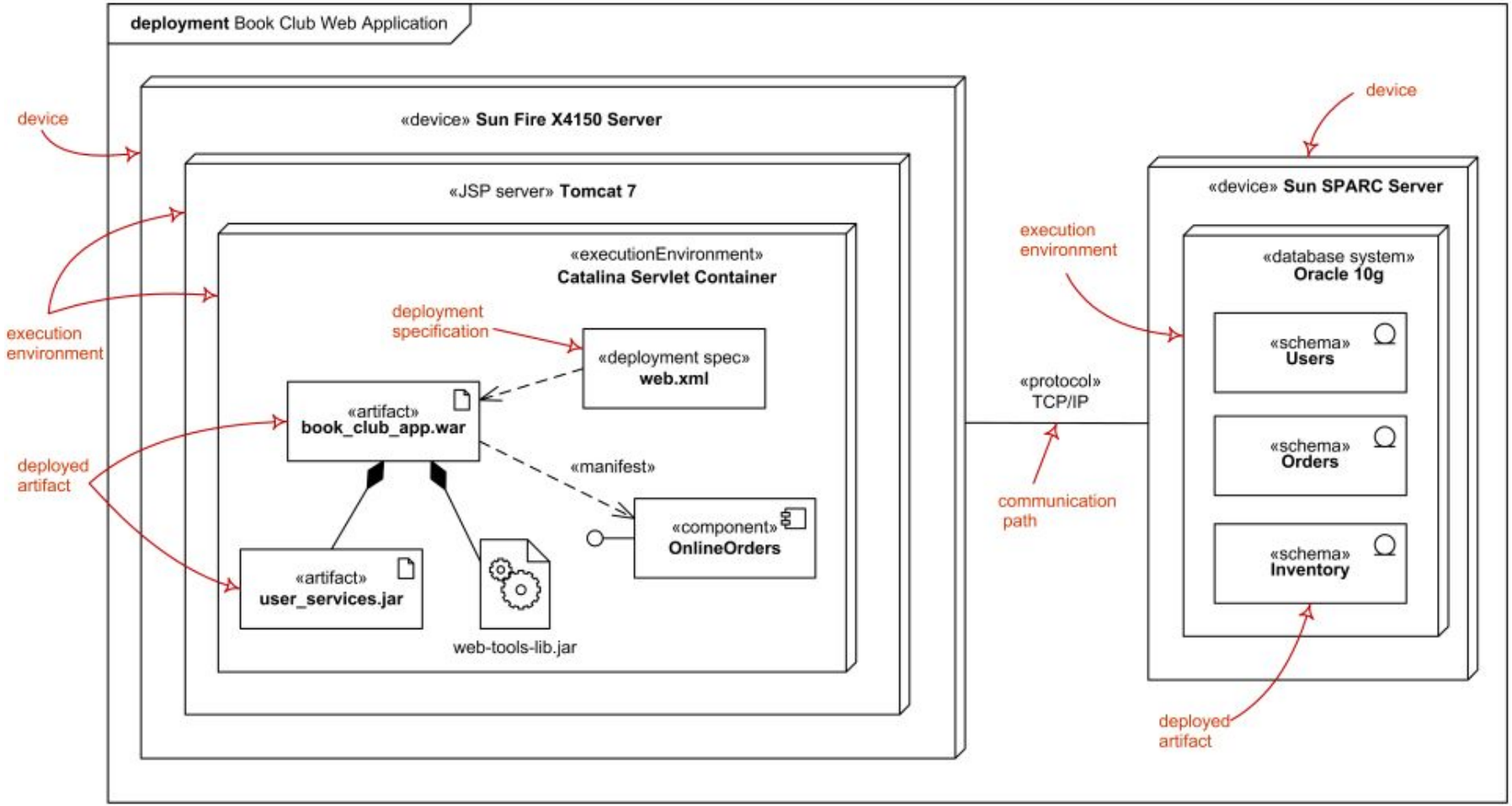
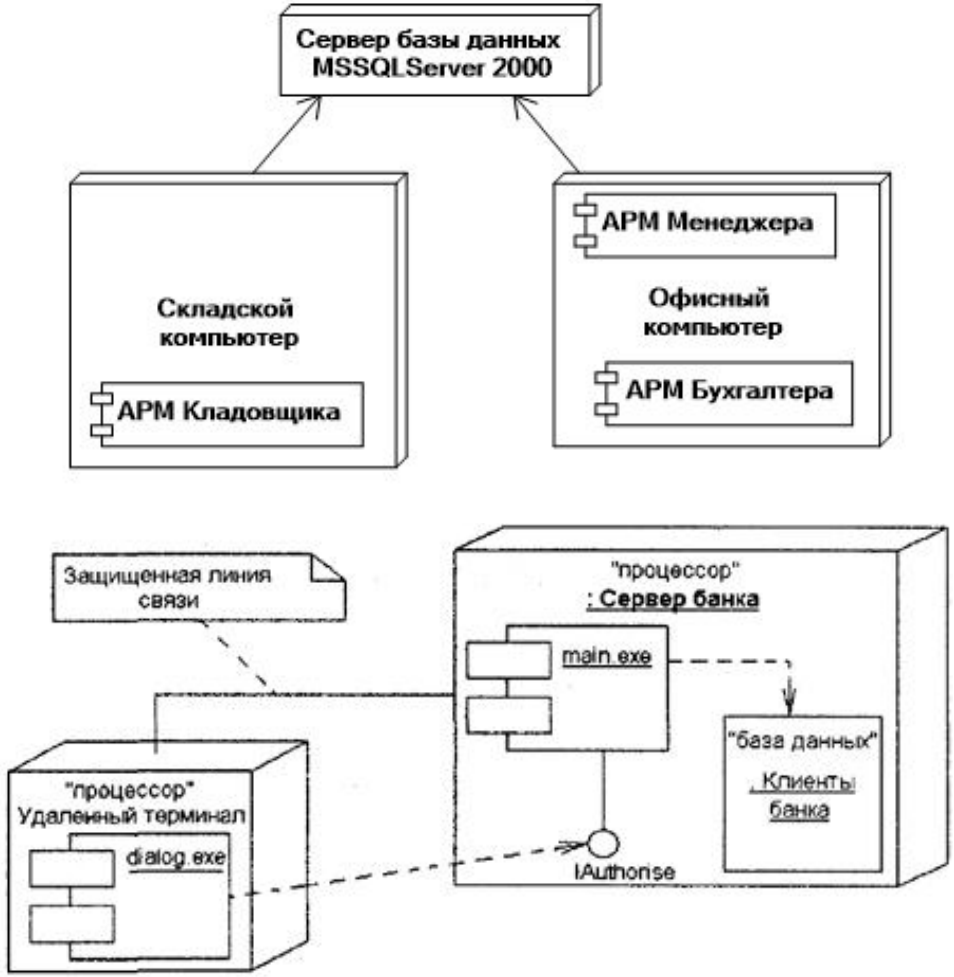
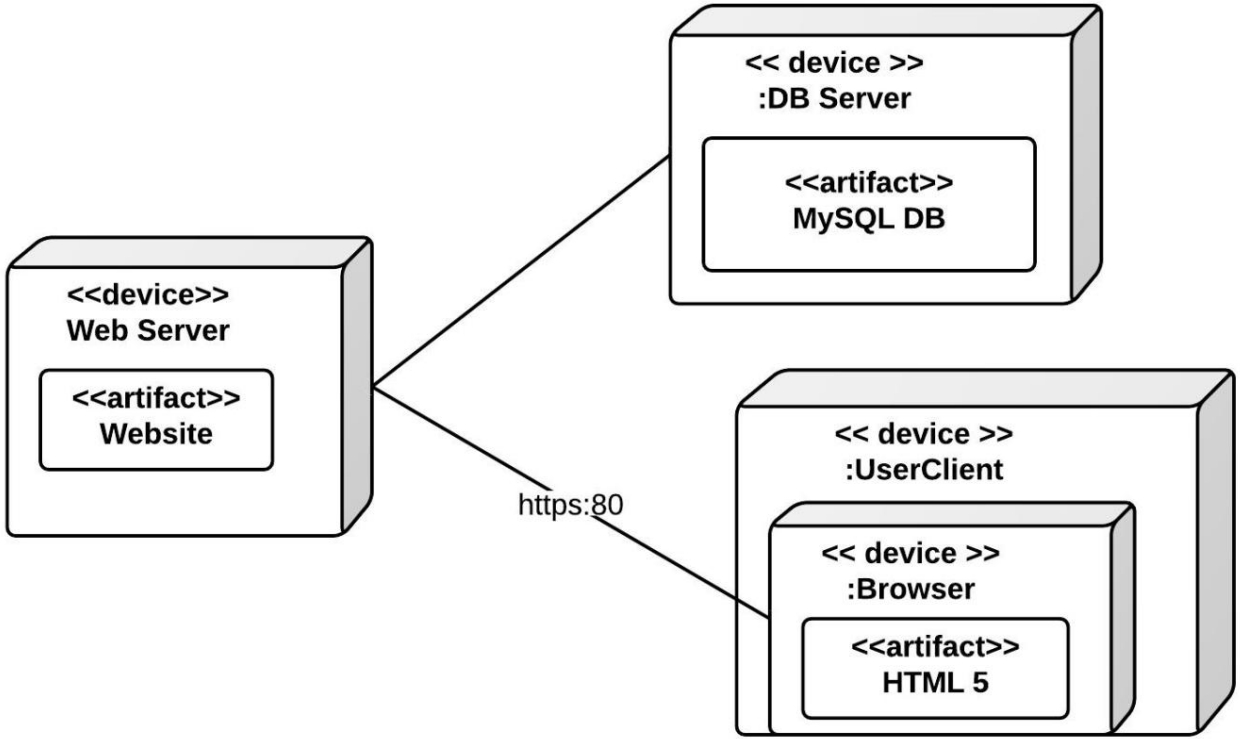


Рис. 8.1. Пример диаграммы развертывания


deployment Book Club Web Application
device
«device» Sun Fire X4150 Server
«JSP server» Tomcat 7
«executionEnvironment»
Catalina Servlet Container
execution environment
deployment specification
«deployment spec»
web.xml
«artifact»
book_club_app.war
deployed artifact
«manifest»
«component»
OnlineOrders
«artifact»
user_services.jar
web-tools-lib.jar
«protocol»
TCP/IP
communication path
device
«device» Sun SPARC Server
execution environment
«database system»
Oracle 10g
«schema»
Users
«schema»
Orders
«schema»
Inventory
deployed artifact


Сервер базы данных
MSSQLServer 2000
Складской
компьютер
АРМ Кладовщика
АРМ Менеджера
Офисный
компьютер
АРМ Бухгалтера
Защищенная линия
связи
"процессор"
: Сервер банка
main.exe
"база данных"
Клиенты
банка
IAuthorise
"процессор"
Удаленный терминал
dialog.exe


<< device >>
:DB Server
<<artifact>>
MySQL DB
<<device>>
Web Server
<<artifact>>
Website
https:80
<< device >>
:UserClient
<< device >>
:Browser
<<artifact>>
HTML 5

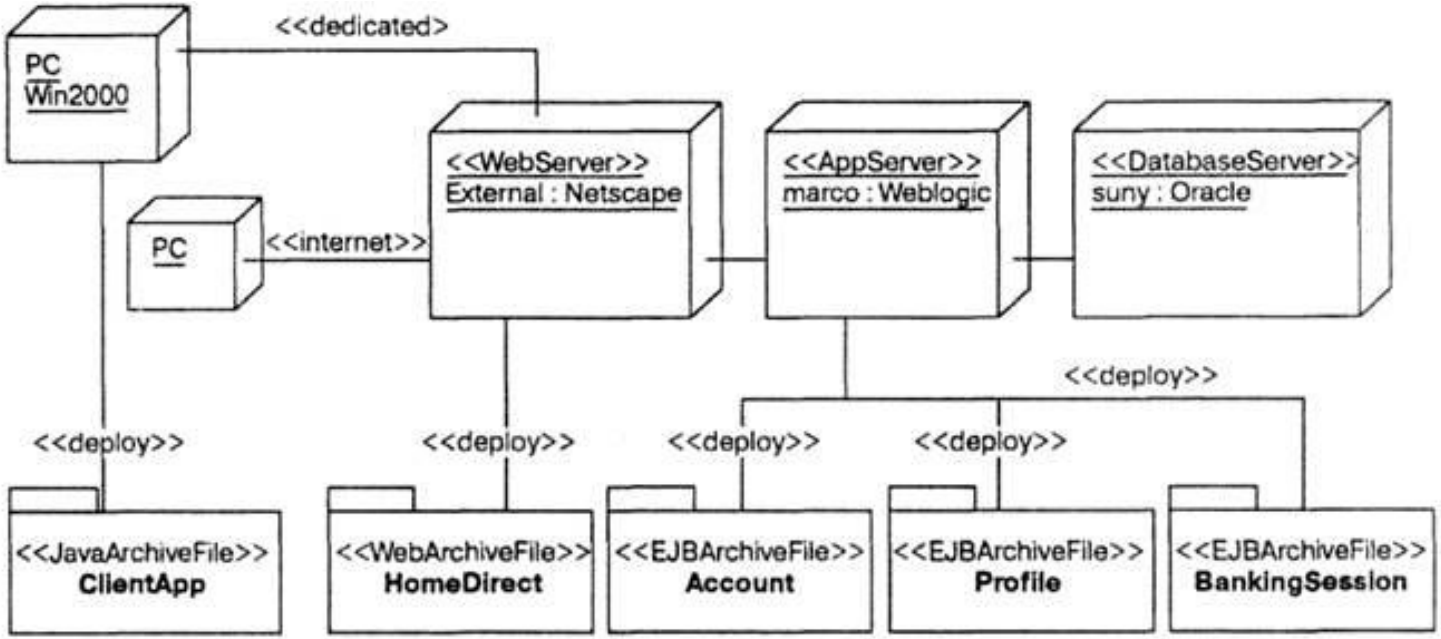
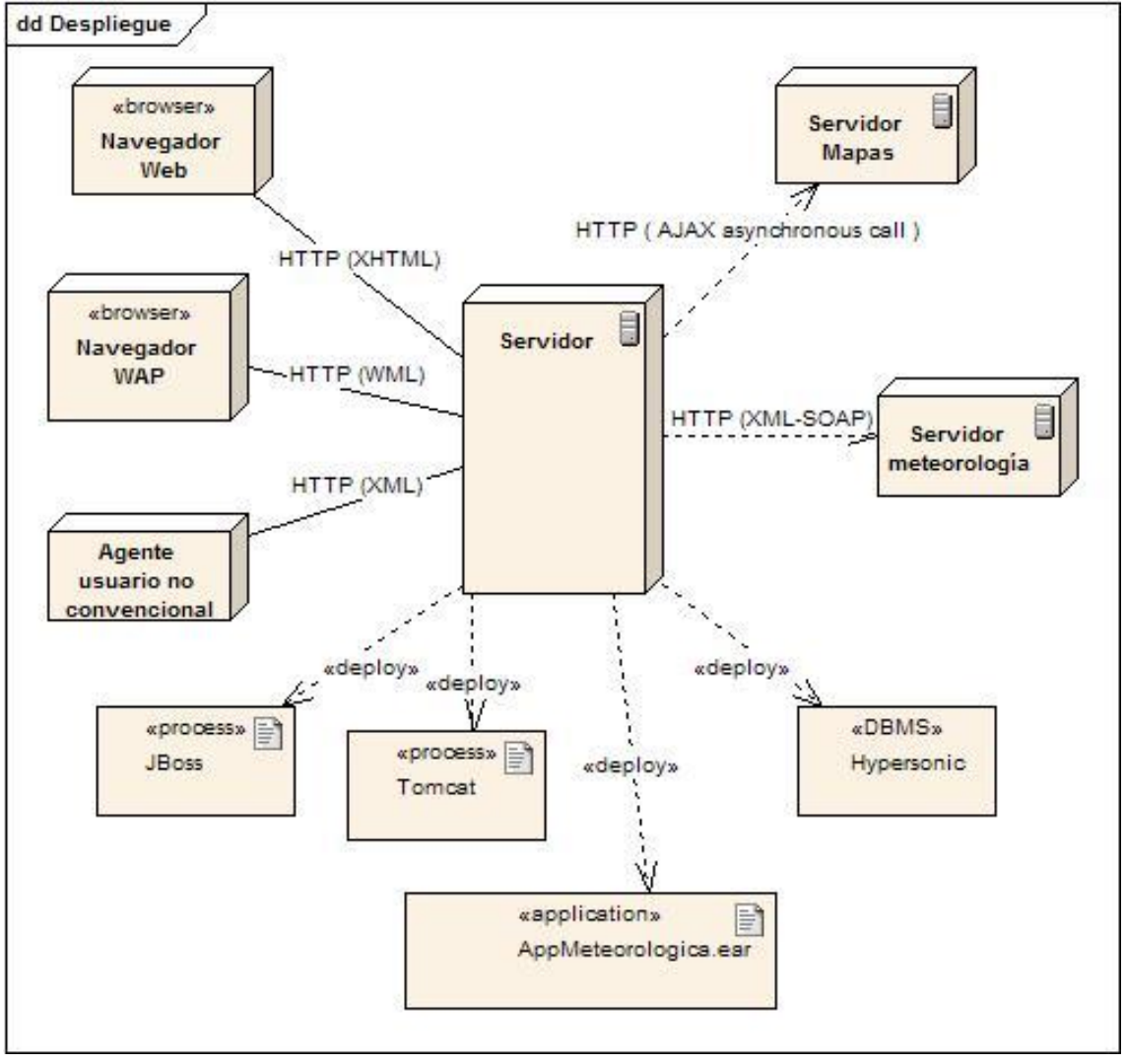
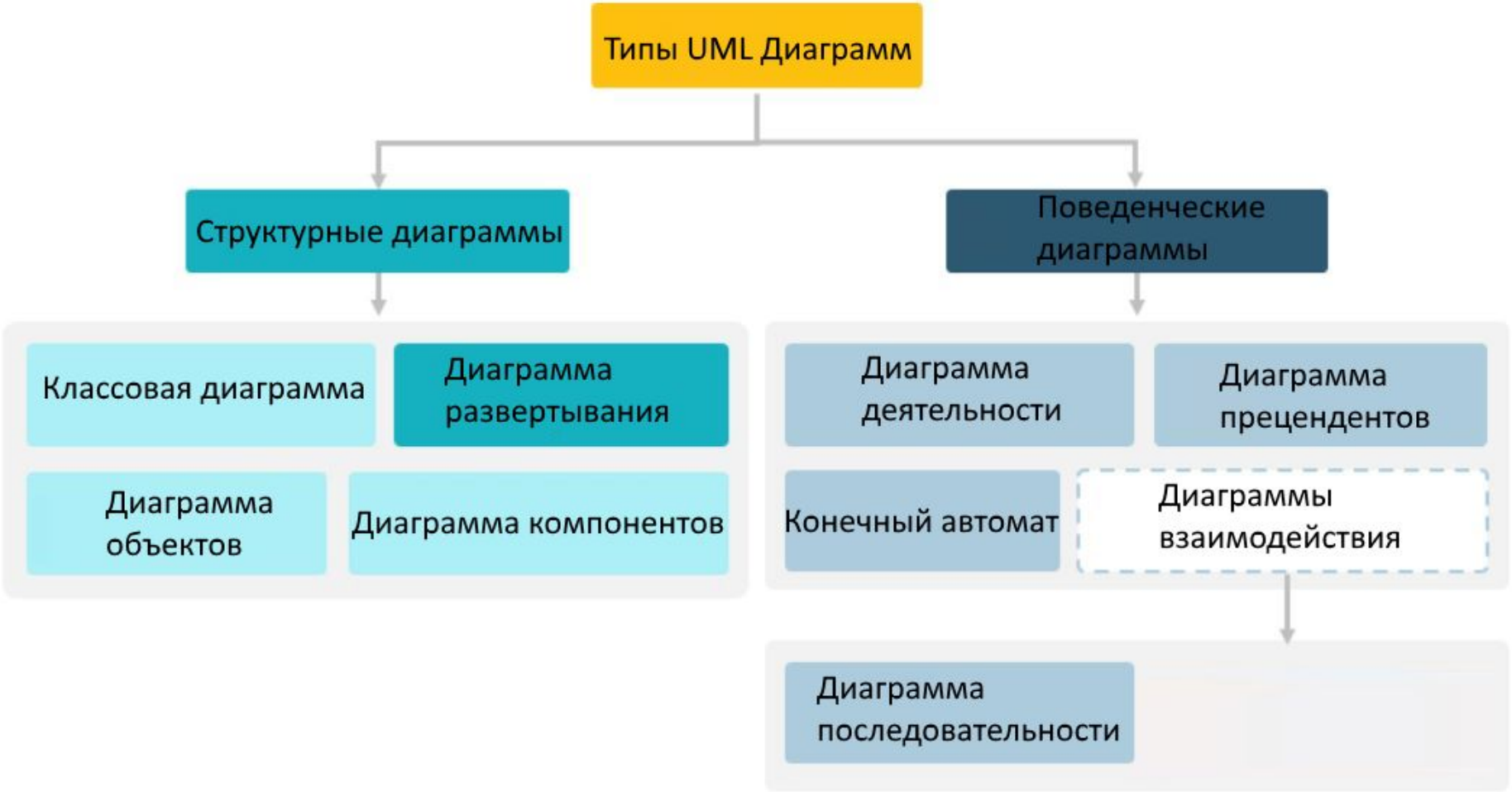


Рис. 15.7. Моделирование развертывания приложений J2EE


dd Despliegue
«browser»
Navegador
Web
«browser»
Navegador
WAP
Agente
usuario no
convencional
Servidor
Mapas
Servidor
Servidor
meteorologia
HTTP (XHTML)
HTTP (WML)
HTTP (XML)
HTTP (AJAX asynchronous call)
HTTP (XML-SOAP)
«deploy»
«deploy»
«deploy»
«deploy»
«process»
JBoss
«process»
Tomcat
«DBMS»
Hypersonic
«application»
AppMeteorologica.ear


Типы UML Диаграмм
Структурные диаграммы
Поведенческие диаграммы
Классовая диаграмма
Диаграмма развертывания
Диаграмма объектов
Диаграмма компонентов
Диаграмма деятельности
Диаграмма прецендентов
Конечный автомат
Диаграммы взаимодействия
Диаграмма последовательности