

## Разработка и сборка компонентов

По оценкам экспертов, 75 % работ по программированию в мире дублируются (например, программы складского учета, начисления зарплаты, расчета затрат на производство продукции и т.п.). Большинство из этих программ типовые, но каждый раз находятся особенности, которые влияют на их повторное разработку.

Компонентное проектирование сложных программ из готовых компонентов является наиболее производительным.

Переход к компонентам происходил эволюционно: от подпрограмм, модулей, функций. При этом усвершенствовались элементы, методы их композиции и накопления для дальнейшего использования.

Компонентный подход дополняет и расширяет существующие подходы в программировании, особенно ООП. Объекты рассматриваются на логическом уровне проектирования программной системы, а компоненты – это непосредственная физическая реализация объектов.

Один компонент может быть реализацией нескольких объектов или даже некоторой части объектной системы, полученной на уровне проектирования. Компоненты конструируются как некоторая абстракция, которая состоит из трех частей: информационной, внешней и внутренней.

Информационная часть представляет собой информацию о компоненте: назначение, дата изготовления, условия применения (ОС, среда, платформа и т.п.); уровень повторного использования; контекст или окружение; способ взаимодействия между собою компонентов.

Элемент композиции	Описание элемента	Схема взаимодействия	Представление, хранение	Результат композиции
Процедура, подпрограмма, функция	Идентификатор	Непосредственное обращение, оператор вызова	Библиотеки подпрограмм и функций	Программа
Модуль	Паспорт модуля, связи	Вызов модулей, интеграция модулей	Банк, библиотеки модулей	Программа с модульной структурой

Объект	Описание класса	Создание экземпляров классов, вызов методов	Библиотеки классов	Объектно–ориентированная программа
Компонент	Описание логики (бизнес), интерфейсов (APL,IDL), схемы развертывания	Удаленный вызов в компонентных моделях (COM, CORBA, OSF, ...)	Репозиторий компонентов, серверы и контейнеры компонентов	Распределенное компонентно – ориентированное приложение
Сервис	Описание бизнес–логики и интерфейсов сервиса (XML, WSDL, ...)	Удаленный вызов (RPC, HTTP, SOAP)	Индексация и каталогизация сервисов (XML, UDDI)	Распределенное сервисо–ориентированное приложение

Внешняя часть определяет взаимодействие компонента со средой и с платформой, на которой он будет выполняться. Эта часть имеет следующие основные характеристики:

- интероперабельность как способ взаимодействия с другими компонентами, с клиентом или сервером, а также обеспечения переносимости на другую платформу;
- способ интеграции (композиции) компонентов;
- нефункциональные сведения (безопасность, аутентификация, надежность и др.);
- технология проектирования (например, объектно–ориентированная среда и т.п.) и повторное использования компонентов.

Внутренняя часть – это некоторый артефакт (кластер, системная или абстрактная структура, фрагмент кода и др.) и вид его представления: проектный компонент, проектная спецификация, вычисляемая часть, код и др.

Внутренняя часть компонента состоит из: интерфейса (interfaces), реализации (implementation), схемы развертки (deployment).

Характеристики		
Интерфейс	Реализация	Схемы развертывания
<ul style="list-style-type: none"> <li>• Один или несколько;</li> <li>• Уникальность именования в пределах системы;</li> <li>• Клиентский или серверный (входной или выходной);</li> <li>• определенная сигнатура;</li> <li>• описание методов взаимодействия</li> </ul>	<ul style="list-style-type: none"> <li>• одна или несколько;</li> <li>• ориентация на конкретную платформу и операционное окружение</li> <li>• выбор конкретной реализации;</li> <li>• поддержка интерфейсов компонента</li> </ul>	<ul style="list-style-type: none"> <li>• типовость процедуры развертывания;</li> <li>• управляемость;</li> <li>• настраиваемость на операционную среду;</li> <li>• модифицируемость</li> </ul>

Рисунок — Основные составные элементы компонента

Интерфейсы отображают взгляд пользователя на компонент, то есть что компонент будет делать, когда к нему обращаются.

Реализация – это код, который будет использоваться при обращении к операциям, которые определены в интерфейсах компонента. Компонент может иметь несколько реализаций, например, в зависимости от операционной среды или от модели данных и соответствующей системы управления базами данных, которая необходима для функционирования компонента.

Развертка – это физический файл или архив, готовый к выполнению, который передается пользователю и содержит все необходимые способы для создания, настройки и функционирования компонента.

Компонент описывается в языке программирования, не зависит от операционной среды (например, от среды виртуальной машины JAVA) и от реальной платформы (например, от платформ в системе CORBA), где он будет функционировать.