АРХИТЕКТУРНЫЕ СТИЛИ В РАЗРАБОТКЕ WEB-ПРИЛОЖЕНИЙ И ОБЛАСТЬ ИХ ПРИМЕНЕНИЯ

Горюнова М.П. Email: Goryunova1134@scientifictext.ru

Горюнова Мария Павловна – заместитель генерального директора,
ОАО «Научный центр прединвестиционных исследований», г. Москва

Аннотация: *в статье описываются основные архитектурные стили или шаблоны, применяемые при разработке программного обеспечения, в частности web-приложений. Описаны такие архитектурные стили, как: компонентная архитектура, многоуровневая архитектура, клиент/сервер архитектура, N-уровневая/3-уровневая архитектура, сервисно-ориентированная архитектура (COA, SOA), объектно-ориентированная архитектура, шина сообщений. В статье представлены основные характеристики этих архитектурных стилей, область их применения и примеры приложений, при построении которых рекомендуется их использование.*

Ключевые слова: *архитектура программного обеспечения, архитектура web-приложений, архитектурные стили, архитектурные шаблоны, компонентная архитектура, многоуровневая архитектура, клиент/сервер архитектура, сервисно-ориентированная архитектура, COA, объектно-ориентированная архитектура, шина сообщений, выбор архитектуры.*

ARCHITECTURAL STYLES IN THE DEVELOPMENT OF WEB-APPLICATIONS AND THE FIELD OF THEIR APPLICATION

Goryunova M.P.

Goryunova Maria Pavlovna - Deputy General Director,
JSC "Scientific Center for Pre-investment Studies", Moscow

Abstract: *the article describes the main architectural styles or templates used in the development of software, in particular in the web applications. Such architectural styles as: component architecture, multi-level architecture, client/server architecture, N-tier/3-tier architecture, service-oriented architecture (SOA), object-oriented architecture, message bus are described in the article. The article presents the main characteristics of the architectural styles, the scope of their application and examples of type of application where their use is recommended.*

Keywords*: software architecture, architecture of web applications, architectural styles, architectural templates, component architecture, multi-level architecture, client / server architecture, service-oriented architecture, SOA, object-oriented architecture, message bus.*

УДК 004.42

Обобщенные принципы и шаблоны, используемые при создании и проектировании ПО, обычно называют архитектурными стилями или шаблонами. В литературе можно встретить определение архитектурного стиля как [1]:

«Архитектурный стиль, таким образом, определяется как семейство систем с точки зрения схемы организации структуры. Более конкретно, архитектурный **стиль** определяет совокупность компонентов и соединений, которые могут использоваться в экземплярах этого стиля, вместе с набором ограничений на их комбинирование. Они могут включать топологические ограничения архитектурных решений (например, отсутствие циклов). Другие ограничения, например, связанные с необходимостью обрабатывать семантику выполнения». К основным архитектурным стилям относятся, например, следующие стили.

1. **Компонентная архитектура**, в которой описывается подход к разработке и проектированию системы, используя методы проектирования ПО. В этом подходе разделение дизайна на отдельные логические или функциональные компоненты, относящиеся к четко определенным интерфейсам, содержащим методы свойства и события, представляется особо важным и создает более высокий уровень абстракции, при сравнении с объектно-ориентированным стилем, также не концентрируется внимание на вопросах общего состояния или протоколах связи. Основные принципы - это использование компонентов, обладающих такими характеристиками, как: замещаемость; возможность повторного использования; расширяемость; независимость от среды и контекста; независимость от других компонентов; инкапсуляция. Применяется чаще всего при создании компонентов пользовательского интерфейса; также при создании ресурсоемких компонентов, доступ к которым осуществляется не часто, а активация выполняется «на лету»; и для создания компонентов с очередью вызовов методов, которые могут асинхронно выполняться благодаря применению очереди сообщений, для пересылки и хранения. К преимуществам подхода можно отнести: простоту разработки; простоту развертывания; упрощение

системы с технической точки зрения; меньшую стоимость разработки и обслуживания; возможность повторного использования.

2. **Многоуровневая архитектура**, которая группирует связанную функциональность ПО в различных слоях, выстраивая их вертикально друг над другом, функциональность объединяется по общей ответственности или роли. При этом слои слабо связаны между собой и между ними осуществляют обмен данными. Слои приложения могут физически располагаться на одном компьютере или на разных. Общие принципы: инкапсуляция; абстракция; возможность повторного использования слоя; высокая связанность внутри слоя; слабая связанность между слоями; четкое разделение функциональности. Примеры таких приложений: системы бухгалтерского учета, веб-приложения или веб-сайты, приложения, использующие централизованные сервера приложений для бизнес-логики.

3. **N-уровневая/3-уровневая архитектура**, описывающая деление функциональности на сегменты, как и в многослойной архитектуре, но в данном случае эти сегменты могут располагаться на различных компьютерах, их называют уровнями. Как правило, для связи используются методы платформы, а не сообщений. К характеристикам N-уровневой архитектуры ПО можно отнести: сервисные компоненты и их распределенное развертывание (обеспечивает масштабируемость и доступность), эффективность и управляемость использования ресурсов функциональную декомпозицию приложения. Независимость уровней от остальных, за исключением тех, с которыми он непосредственно соприкасается. N-му уровню необходимо только знать, как обработать запрос от n+1 уровня, как передать этот запрос на n-1 уровень, и как обработать результаты запроса. Связь между уровнями, как правило, асинхронная, для лучшей масштабируемости. К преимуществам можно отнести: масштабируемость; гибкость; доступность; удобство поддержки. Например: Веб-приложение с высокими требованиями к безопасности, насыщенный клиент.

4. **Клиент/серверная архитектура**, дающая описание распределенным системам, состоящим из отдельных сервера и клиента и сети, которая их соединяет. Примерами приложения могут быть: веб-приложение, выполняемое во внутренних сетях компании или в Интернет, настольные приложения, работающие с удаленными ресурсами. Основные преимущества: простота обслуживания, большая безопасность, централизованный доступ к данным, к минусам можно отнести сложность расширяемости, масштабирования и зависимость от центрального сервера.

5. **Основанная на шине сообщений архитектура**, описывающая вариант использования программной системы, в которой отправляются и принимаются сообщения по одному или нескольким каналам связи, позволяя взаимодействовать приложениям без детальных знаний друг о друге. Взаимодействие реализуется путем передачи сообщений через шину, как правило, асинхронной. Типично использование маршрутизатора сообщений или шаблона Публикация/Подписка (Publish/Subscribe) и системы обмена сообщениями, такие как Очередь сообщений (Message Queuing). Шина позволяет обеспечить обработку: основанные на сообщениях взаимодействия; сложной логики обработки; изменение логики обработки; интеграцию с различными инфраструктурами. Основные преимущества: гибкость, расширяемость, слабое связывание, масштабируемость, простота приложений, невысокая сложность.

6. **Сервисно-ориентированная архитектура (COA, SOA)**, дающая возможность создавать приложения, использующие программные сервисы, и позволяющая предоставлять функциональность ПО в виде набора сервисов. Сервисы слабо связаны благодаря использованию основанных на стандартах интерфейсов, которые можно вызвать, опубликовать и обнаружить. COA позволяет упаковать бизнес-процессы в сервисы, поддерживающие возможность взаимодействия и использования различных форматов данных и протоколов. Основные принципы: Совместимость основана на политике, сервисы слабо связаны, автономны, совместно используют контракт, схему, но не класс, сервисы могут быть распределены. Преимущества: абстракция; возможность обнаружения и автоматического подключения через интерфейс, согласование предметных областей, рационализация, возможность взаимодействия.

7. **Объектно-ориентированная архитектура**, основанная на разделении ответственности системы или приложения на пригодные для повторного использования объекты, каждый из которых содержит поведение и данные, относящиеся к объекту. При данном подходе система рассматривается как набор взаимодействующих объектов, а не набор подпрограмм и процедурных команд. Основные принципы: наследование, инкапсуляция, композиция, абстракция, отделение, полиморфизм. Стиль используется для описания объектов, имеющих место в реальной жизни, или для описания моделей, поддерживающих сложные финансовые или научные операции. Основные плюсы: возможность повторного использования с помощью полиморфизма или абстракции, понятность, улучшенная тестируемость, расширяемость, высокая связность.

Следует принять во внимание, что различные шаблоны могут использоваться в разных аспектах описания ПО. Некоторые из шаблонов применяются при описании схем развертывания, связи, другие при описании структуры или дизайна. Как правило, при разработке стандартного приложения применяется несколько различных архитектурных шаблонов.

Также особо рекомендуется сочетать архитектурные стили при построении web-приложений, так как это дает возможность эффективно разделить функциональность при применении многослойного архитектурного шаблона. Это дает возможность отделить бизнес-логику от логики доступа к данным. Трех и более уровневое развертывание приложения могут диктоваться требованиями безопасности компании. В этом случае можно разместить в пограничной сети, расположенной между внешней и внутренней сетью компании, уровень представления. В этом случае можно реализовать СОА и сделать связь между сервисом приложений и веб посредством обмена сообщениями. Или использовать в качестве модели взаимодействия на уровне представления такой шаблон, как Модель-представление-контроллер (МПК) (Model-View-Controller (MVC). Использовать СОА для построения экспертных систем на базе веб предлагают работы [3, 4, 5].

Список литературы / References

1. *Garlan D., Shaw M.* An introduction to software architecture // Advances in software engineering and knowledge engineering, 1993. Т.1. № 3.4.
2. *Huimin F., Panpan Y.* Research on Application of Web Services in an Expert System // Industrial Control and Electronics Engineering (ICICEE), 2012 International Conference on. IEEE, 2012. С. 350 - 352.
3. *Kruchten P., Obbink H., Stafford J.* The past, present, and future for software architecture // IEEE software, 2006. Т. 23. № 2. С. 22 - 30.
4. *Pyshkin E., Kuznetsov A.* Approach to building a web-based expert system interface and its application for software provisioning in clouds //Computer Science and Information Systems, 2015. С. 343 - 354.
5. *Tapiador A. et al.* A web collaboration architecture//Collaborative Computing: Networking, Applications and Worksharing, 2006. CollaborateCom 2006. International Conference on. IEEE, 2006. С. 1 - 4.