# Определение понятия «архитектура программного обеспечения».

Архитектура программного обеспечения (software architecture) — это структура программы или вычислительной системы, которая включает программные компоненты (элементы), видимые снаружи свойства этих компонентов, а также отношения (взаимодействия) между ними. Архитектура затрагивает не только структуру и поведение, но также использование, функциональность, и другие аспекты. Этот термин также относится к документированию архитектуры программного обеспечения. Документирование архитектуры ПО упрощает процесс коммуникации между стейкхолдерами (stakeholders – лица заинтересованные в проекте), позволяет зафиксировать принятые на ранних этапах проектирования решения о высокоуровневом дизайне системы и позволяет использовать компоненты этого дизайна и шаблоны повторно в других проектах.

# Архитектурная концептуальная схема Дж. Захмана.

Особое место среди архитектурных концептуальных образцов занимает концептуальная схема Дж. Захмана [7] (1987 год). Схема изначально нацеливалась на системное представление задач информатизации, в которых приходится учитывать специфику ПО.

С момента своего создания схема Дж. Захмана постоянно усовершенствовалась, оставаясь определенного рода «вопросником» о существенных аспектах ПО. Схема построена в форме табличной структуры, для формирования и использования которой необходимо построить систему ответов, каждый из которых заполняет определенную позицию таблицы. Ответ может быть как простым, так и сложным. Одной из наиболее распространенных форм ответов является визуализируемая концептуальная модель.



Рисунок 3. Архитектурная концептуальная схема Дж. Захмана

Использование этой схемы предполагает, что лица, отвечающие за архитектуру системы, заполнят «клетки» подходящим содержанием. Разумеется, содержание «клеток» включает спецификации (на уровне архитектуры) тех решений, которые по этой «клетке» приняты. В результате будет образована целостная архитектурная картина представляемой системы.

Пример конкретной реализации схемы Захмана приведен на рис. 4.



Рисунок 4. Пример реализации схемы Захмана

То содержание, которое вкладывается в каждый ряд и каждый столбец схемы, представлено в таблице 1 и таблице 2.

Таблица 1. Описание назначения строк схемы.

|  |  |
| --- | --- |
| **Ряд схемы** | **Содержание** |
| 1. Цели и границы (Вид планировщиков) | Определяет основные цели и границы, в рамках которых должны приниматься архитектурные решения |
| 2. Модель предприятия, системы (Вид владельцев, ответственных за систему) | Определяет сущность предприятия (системы), включая структуры, процессы и организационные структуры |
| 3. Модель базового понимания (информационного представления, Вид архитектора) | Определяет в более строгих терминах содержание ряда 2 |
| 4. Технологическая модель ( Вид проектировщиков) | Определяет, как и какая технология должна быть использована для того, чтобы запросы третьего ряда были удовлетворены |
| 5. Детализированное представление (Вид разработчика) | Определяет детали проектирования, применяемые языковые средства, структуры данных и другие средства |
| 6. Функциональности системы (Вид пользователей) | Определяет, как должны работать системы в рамках предприятия ( в контексте окружения) |

Таблица 2. Описание назначения столбцов схемы.

|  |  |
| --- | --- |
| **Столбец** | **Содержание** |
| 1. Данные ( Структура, Что?) | Фокусирует внимание на сущностях, объектах, компонентах и отношениях между ними |
| 2. Функции ( Активность, Как?) | Фокусирует внимание на той деятельности, которая осуществляется организацией и поддерживается автоматизированными системами |
| 3. Сеть работ (Месторасположение, Где?) | Фокусирует внимание на географическом (физическом) распределении активностей |
| 4. Люди ( Кто?) | Фокусирует внимание на тех лицах, которые вовлечены в бизнес-процессы |
| 5. Время ( Когда?) | Фокусирует внимание на эффектах, связанных со временем ( планирование, события) |
| 6. Мотивации ( Почему?) | Фокусирует внимание на целях, стратегиях и ограничениях, специфических для реализации системы |

*Здесь и далее используется термин "ВИД" (View). Понимать этот термин следует по аналогии с чертежами и конструкторской документацией: вид на деталь справа, слева, сверху, оксонометрия детали, описание применяемого материала... Для программирования, например, вид с точки зрения пользователя - внешние интерфейсы, вид с точки зрения программиста - структуры данных и алгоритмы работы программы и т.п.*

Заметим, что Дж. Захман разрабатывал свою схему для предприятий, использующих любые информационные технологии, но схема примененима как в теории так и в практике программирования.

Развитие концептуальной схемы Дж. Захмана до трех измерений за счет перехода к многослойной структуре системы моделей показано на рис. 5.



Рисунок 5. Трехмерная схема Захмана

В трехмерную структуру введены следующие дополнения, детализирующие содержание артефактов, которые приписываются «клеткам схемы Дж. Захмана»:

− слой разработки, раскрывающий «Что?» должно быть сделано для каждого артефакта, приписанного к «клетке» схемы Дж. Захмана;

− слой метаданных, включающий описание информации, которая должна быть собрана и «Как?» она должна быть организована;

− слой документирования, раскрывающий типовые формы документов, в которых (Где?) будет зафиксирована информация;

− слой ответственности, фиксирующий, «Кто?» из команды разработчиков будет исполнять требуемые работы;

− слой жизненного цикла, показывающий, «Где?» в процессе разработки будет создан соответствующий артефакт;

− слой целей, описывающий, «Почему?» следует включать артефакт в состав архитектурного описания.

Для предлагаемого расширения автором трехслойной модели разработаны детали для каждого слоя (и для каждой клетки слоя).

# Архитектурные стили. Клиент—сервер.

# Паттерн фабричный метод. Назначение. Схема. Достоинства и недостатки.

# Выгоды, получаемые от построения и использования архитектуры.

Перечислим положительные факторы, получаемые от построения и использования архитектуры ПО. Архитектура:

- вносит вклад в формирование системы требований к ПО;

- показывает совокупность ранних проектных решений;

- описывает организационную структуру ПО;

- является общим базисом для линеек программных продуктов;

- позволяет учитывать, согласовывать и предсказывать характеристики качества, в том числе и по результатам ее исследования;

- дает информацию о распределении работ и их календарных планах;

- дает возможность для более точной оценки стоимости и расписания работ;

- снижает риски;

- является первой формой существования ПО, которая может быть проверена (испытана) как целое;

- предоставляет возможность для переноса или повторного использования стилей и архитектурных каркасов;

- приносит выгоду в ограничении «словаря» альтернатив проекта или его частей;

- помогает в эволюционном прототипировании;

- оформляет концептуальную целостность ПО и процесса его разработки;

- обслуживает понимание и взаимопонимание в индивидуальной и коллективной работе;

- предоставляет возможность рассуждать о потенциальных изменениях по ходу разработки ПО и вносить вклад в управление такими изменениями;

- может быть базисом для изучения системы.

# Архитектурная концептуальная схема DoDAF.

***Архитектурная концептуальная схема DoDAF*** (Department of Defense Architecture Framework), создана по заказу Департамента обороны США.

Архитектурное описание ПО в DoDAF строится на базе модели данных ядра архитектуры (Core Architecture Data Model, CADM). CADM представляет собой стандартизованную систему понятий для определения видов и их составляющих в базе данных[8].

Вид понимается традиционно и представляет определенный объем операций, систем и технических стандартов. Каждый вид является хорошо определенным множеством элементов данных в составе CADM. Архитектурные артефакты документируются с использованием языка UML.

Виды объединены в четыре группы. Первая группа называется «Все виды» (All Views) и содержит обзор, обобщения и интегрированный словарь по архитектуре DoDAF. Остальные группы – это «Операционные виды» (*Operational Views*), «Системные виды» (System Views) и «Технические виды» (*Technical Views*). Обобщенная картина видов представлена на рис. 6.



Рисунок 6. Архитектурная концептуальная схема DoDAF

Операционные виды описывают специфику деятельности и операции, узлы операций, узлы соединений, информационный обмен, организационные отношения, правила операций, последовательности событий и логическую модель данных. Описания, содержащиеся в этой группе, распределены по 24 подвидам. (каждый подвид является видом по определению IEEE-1471).

Системные виды описывают систему и её компоненты, системные интерфейсы, коммуникативное взаимодействие, матрицу отношений на множестве систем и подсистем, функциональности, матрицу связности (traceability), эволюцию, использование и технологические предсказания. Группа состоит из подвидов.

Технические виды (группа из 12 подвидов) описывают текущий стандартный профиль и предсказания по его изменениям.

Архитектурная концептуальная схема DoDAF разработана для военных нужд и в этом плане она является предметно-зависимой. Следует отметить, что DoDAF предоставляет ограниченные возможности по моделировании конфигурации ПО и моделированию нефункциональных требований.

Ближайшим родственником архитектурной схемы DoDAF является схема MoDAF, разработанная европейскими союзниками США. В число основных потребностей, приводящих к отличиям схемы MoDAF от схемы DoDAF, входят следующие потребности:

1. Потребность решать и моделировать задачи приобретения разного рода комплектующих аппаратного и программного типов.

2. Потребность моделировать трансформационные программы и их взаимозависимости.

3. Потребность моделировать компетентность и другие виды способностей лиц, вовлечённых в разработку и использование АС.

4. Потребность моделировать необходимые для разработки АС кадровые ресурсы, подобно моделированию технических ресурсов.

5. Потребность объединять информационные представления в традиционные модели архитектуры, чтобы обеспечить необходимой информацией работу архитекторов предприятия.

Средства DoDAF согласованы с концептуальной схемой Захмана. Объём пересечений между схемами (отмеченный овалами) отражён на Рис. 7. В версиях DoDAF указывается, что они согласованы со стандартом IEEE-1471.

# Архитектурные стили. Одноранговые сети.

# Паттерн абстрактная фабрика. Назначение. Схема. Достоинства и недостатки.

# Основные цели и задачи стандарта IEEE 1471-2000.

Стандарт позиционирован его разработчиком (IEEE Architecture Planning Group) как «рекомендуемая практика». Он не является стандартом архитектуры, процесса построения архитектуры или метода разработки архитектуры, он предоставляет разработчикам ПО рекомендации для описания архитектуры (Architectural Description, AD). Эти рекомендации используют модальности «должен» (без обязательности) и «можно» (как вариант). Но рекомендации базируются на большом опыте успешных разработок ПО и приняты мировыми лидерами индустрии разработки ПО.

Цели и задачи стандарта IEEE 1471-2000:

- нацелен на архитектурноге описание (АО) систем, интенсивно использующих программное обеспечение, в которых программная составляющая оказывает существенное влияние на проектирование, конструирование, развертывание и эволюцию системы как целого;

- предполагает, что организации и предприятия, решившие использовать стандарт, должны сами решать, в каком объеме и как он будет внедряться в разработки ПО;

- дает широкие рамки интерпретации архитектуры, как средства, пригодного для процесса разработки ПО;

- исходит из того, что описание архитектуры может быть согласовано широким кругом лиц, вовлеченных в разработку ПО, в то время как система, проект, процесс и организация работ такой возможности обычно не предоставляют (из-за специфики разнородной профессиональной компетенции и языка);

- определяет концептуальную структуру (систему понятий) для описания архитектуры и ее использования;

- устанавливает термины и понятия для архитектурного мышления;

- устанавливает концептуальный каркас и словарь для рассуждений об архитектурной форме существования ПО в виде архитектурного описания;

- обеспечивает средства для рассуждений об архитектурном описании в контексте лиц, вовлеченных в процесс разработки ПО, жизненного цикла ПО и использования АО;

- служит как базис для развития ПО на ранних этапах разработки, когда доступно только общее представление о проекте;

- вводит в системы требований для разработки ПО специальный раздел «архитектурных требований»;

- идентифицирует и устанавливает тождество в описаниях архитектур и пропагандирует озвученное в архитектурной практике;

- открывает возможность для эволюции таких практик, как достигших достаточной инструментально-технологической зрелости;

- служит как базис для развития предметной области, в которой существует только общая терминология;

- вносит надежду разработчикам ПО, что следование его рекомендациям внесет в разработку такие позитивы деятельности как «быстрее», «лучше» и «дешевле».

# 2. Архитектурная концептуальная схема TOGAF.

***Архитектурная концептуальная схема TOGAF*** *1995 г.*

С момента ее первого опубликования в 1995 году архитектурная концептуальная схема TOGAF (The Open Group Architecture Framework) развивается и используется. TOGAF позиционируется ее авторами не как некоторая эталонная модель, а как "средство для разработки архитектур информационных систем". Основное назначение – ускорить и облегчить процесс разработки архитектуры конкретной организации, обеспечивая при этом возможность будущего развития. В декабре 2003 года была опубликована версия 8.1 этой модели.

Основным полем для применения TOGAF является, прежде всего, программная инфраструктура информационной системы (в противоположность таким типам архитектур, как бизнес-архитектура, архитектура данных и приложений). Таким образом, она в наилучшей мере подходит для описания интеграционных компонент, использующихся для поддержки широкого спектра корпоративных приложений, прежде всего, критичных для бизнеса (mission-critical). Поскольку эта интеграционная архитектура сильно зависит от принимаемых решений в остальных областях, то в рамках TOGAF в необходимой степени рассматриваются и эти смежные области.

В состав модели TOGAF входят две основные компоненты – методика ADM (Architecture Development Method), определяющая процесс разработки архитектуры, и Базовая Архитектура (Foundation Architecture). Она дополняется соответствующей базой данных ресурсов, включающей описания архитектурных принципов, примеров реализации, а также специализированный язык ADML. Заметим, что в описании TOGAF добавлен специальный документ, поясняющий соответствие между понятиями TOGAF и моделью Захмана.

Общая структура TOGAF [9] показана на рис. 8.

Важно отметить, что TOGAF распространяется свободно и может быть использована бесплатно любой организацией для разработки внутренних проектов. Лицензируется только коммерческое использование.

 В соответствии с методикой ADM, процесс разработки архитектуры включает следующие фазы:

* Подготовка: уточнение модели под особенности организации, определение принципов реализации проекта.
* Фаза A: определение границ проекта, разработка общего представления (Vision) архитектуры; утверждение плана работ и подхода руководством.



Рисунок 8. Общая структура TOGAF

* Фаза B: разработка бизнес-архитектуры предприятия.
* Фаза C: разработка архитектуры данных и архитектуры приложений.
* Фаза D: разработка технологической архитектуры.
* Фаза E: проверка возможности реализации предложенных решений.
* Фаза F: планирование перехода к новой системе.
* Фаза G: формирование системы управления преобразованиями.
* Фаза H: управление изменением архитектуры.

 Каждая фаза, в свою очередь разбивается на подпроцессы (этапы), отдельные работы и так далее. Например, фаза D включает следующие основные подпроцессы:

* Описание существующей технологической архитектуры.
* Обзор бизнес-архитектуры, архитектуры данных и приложений для определения начальных данных и необходимой степени детализации.
* Описание существующей системы с необходимой степенью детализации, которая выбирается для того, чтобы можно было выявить необходимые изменения при формировании целевой архитектуры. Формирование реестра используемых платформ программного и аппаратного обеспечения.
* Выявление и описание элементарных архитектурных блоков – кандидатов на использование в новой архитектуре. Фактически, речь идет о возможных архитектурных шаблонах.
* Разработка черновика технического отчета, резюмирующего основные результаты изучения существующего состояния и возможности использования типовых блоков.
* Направление черновика отчета на рецензирование, анализ комментариев и внесение, при необходимости, поправок.
* Формирование целевой технологической архитектуры.
* Описание существующей системы в терминах TOGAF.
* Определение перспектив (представлений) архитектуры.
* Формирование модели целевой архитектуры.
* Определение ИТ-служб (сервисов).
* Подтверждение учета бизнес-требований.
* Определение архитектуры и используемых блоков (шаблонов).
* Проведение анализа расхождений (gap analysis).

 Для каждого такого подпроцесса определяются решаемые в его ходе задачи, входные и выходные документы. Важно отметить, что процесс предусматривает не обязательную, но возможную адаптацию самого метода к условиям конкретного предприятия, которая осуществляется на предварительной фазе. Это может быть вызвано как необходимостью учета других существующих стандартов предприятия, так и привлечением аутсорсинговых компаний к разработке архитектуры.

Процесс разработки не заканчивается после выбора оптимальной архитектуры и разработки плана миграции. Необходимыми элементами являются задачи, выполняемые на фазах G и H. В частности, для обеспечения практического принятия архитектуры в организации и успеха проекта обязательным является формирование Системы управления реализацией архитектуры (Implementation Governance).

Применение TOGAF осуществляется как итеративный процесс. Итерации TOGAF (которые в руководствах называются циклами) обычно объединяют несколько фаз реализации и обслуживания архитектуры предприятия (Рис. 9).



Рисунок 9. Жизненный цикл TOGAF

# 3. Архитектурные стили. Сервис-ориентированная архитектура.

# 4. Паттерн строитель. Назначение. Схема. Достоинства и недостатки.

# 1. Концептуальный каркас стандарта IEEE 1471-2000.

# 2. Архитектурная концептуальная схема FEAF.

***Архитектурная схема FEAF*** *1998 год.*

Ее специфика (рис. 10) связана с разработкой ПО в рамках системы задач государственного масштаба для США.

Федеральная Архитектура – это концептуальная модель описания деятельности федерального правительства и государственных организаций с функциональной точки зрения, вне зависимости от организационных структур, реализующих соответствующие функции, с целью улучшения их деятельности за счет использования информационных технологий. Подобные задачи в нашей стране намечено решать в рамках «Электронной России».

Федеральная Архитектура – определяет функции государственных организаций, информацию и технологии, необходимые для их реализации, а также процессы преобразований, необходимые для внедрения новых информационных технологий. Отдельные государственные организации должны использовать эту общую модель для описания своих собственных архитектур [10].



Рисунок 10. Архитектурная схема FEAF

# 3. Архитектурные стили. Архитектура, управляемая моделью (MDA).

# 4. Паттерн прототип. Назначение. Схема. Достоинства и недостатки.