04 апреля 2023

ИТ-архитектура

Создание архитектуры в организации с нуля может показаться сложной задачей. Она требует тщательного планирования, исследований и анализа, чтобы убедиться, что система отвечает потребностям компании. Эффективная ИТ-архитектура должна быть разработана для поддержки целей и задач бизнеса, обеспечивать масштабируемость для будущего роста и безопасность.

Создание прочной ИТ-основы: как разработать эффективную ИТ-архитектуру для вашей организации?

1. Разработка ИТ-стратегии

Стратегия должна включать в себя видение и миссию, а также цели и задачи по использованию технологий в организации. Она также должна определять масштаб проекта и выявлять заинтересованные стороны и их роли. Конечным результатом этого процесса должен стать всеобъемлющий план, который описывает, как технология позволит организации достичь своих целей.

На этом этапе необходимо понимать, чем живет ИТ-компания для которой мы пишем стратегию. Изучить зрелость существующих информационных и

бизнес-процессов. Составить стратегическую карту, описать планы проектов и задач, расписать необходимые расходы. Информационная стратегия создается с опорой на бизнес-цели, их необходимо декомпозировать до ИТ и сформировать задачи.

Следующий шаг – составление модели, к которой планируем прийти. Чтобы работа выполнялась более эффективно, добавьте конкретные цели в KPI руководителей и сотрудников на проекте.

Составляем план действий и список необходимых ресурсов. Наибольшее внимание стоит акцентировать на критичной архитектуре, именно она является основой в области операционной устойчивости и обеспечивает непрерывность и постоянство эксплуатации.

2. Разработка иерархического реестра

После того, как организация разработала свою IT-стратегию, следующий шаг - разработка иерархического реестра всех ИТ-систем и ИТ-платформ организации. Этот реестр должен включать информацию о назначениях, типе приложения, требованиях к аппаратному обеспечению, потоке данных, точках интеграции с другими устройствами, мерах безопасности. Эта информация служит основой для дальнейшего развития архитектуры организации.

Самые популярные подходы и стандарты для описания архитектуры:

- TOGAF,
- ITIL,

- схема Захмана,
- COBIT,
- IT4IT.

Расскажем про некоторые из них подробнее:

1. **TOGAF** — это открытая организационная структура, которая предоставляет организациям инструменты и ресурсы, необходимые для разработки, планирования, внедрения и управления ИТ-архитектурой уровня предприятия. Она широко используется в ИТ-индустрии и стала стандартом де-факто для архитектуры предприятия. В основе TOGAF лежат четыре основных принципа: модульность, масштабируемость, расширяемость и гибкость. Он обеспечивает комплексный подход к проектированию архитектуры, которая может быть адаптирована к изменяющимся потребностям организации или ее клиентов. Концепция также содержит руководство по разработке архитектурных моделей и документов, которые можно использовать в качестве опорных точек для будущих проектов разработки.

   Используя эту структуру в качестве руководства для разработки своей архитектуры, организации могут сократить расходы, связанные с разработкой сложных систем с нуля. Кроме того, она помогает обеспечить соответствие архитектуры всем нормативным требованиям, обеспечивая при этом максимальную гибкость для удовлетворения запросов клиентов или изменений в технологии. TOGAF разработан таким образом, чтобы быть простым в использовании и понятным как техническому, так и не техническому персоналу.

   Для любой организации, которая ищет комплексный подход к разработке эффективной архитектуры, TOGAF, безусловно, заслуживает внимания, поскольку он предоставляет все необходимые инструменты и

при этом достаточно прост для быстрого понимания.

2. **Схема Закмана** существует с 1987 года и не теряет актуальности по сей день. Она основана на шести различных перспективах: план, данные, сеть, процесс, приложение и технология. Каждая перспектива далее делится на пять компонентов: кто (люди), что (продукты или услуги), где (местоположение), когда (временные рамки) и почему (мотивы). Это обеспечивает комплексный взгляд на все предприятие с разных сторон.

   Красота этой системы заключается в ее гибкости — она может быть применена к любой организации, независимо от ее размера или отрасли. Она также обеспечивает масштабируемость, позволяя организациям добавлять дополнительные перспективы по мере необходимости. Кроме того, она обеспечивает эффективный способ выявления пробелов в существующих системах и процессах, чтобы их можно было быстро и эффективно устранить. Схема Захмана выдержала испытание временем благодаря своей простоте и в то же время эффективности в обеспечении общего представления системы архитектуры предприятия.

В разных подходах архитектурный каркас информационных систем отличается. Мы рассмотрели несколько самых популярных.

3. Создание комплексного обзора ИТ-систем и модулей

Для того, чтобы убедиться, что все системы правильно интегрированы в общую архитектуру, необходимо заполнить детальные параметры для каждой отдельной системы или модуля. Карточка должна содержать подробные и исчерпывающие данные. В будущем на основе этой информации можно

создавать отчеты и принимать решения о внесении изменений.

Какие параметры она будет включать:

1. Название системы.
2. Краткое описание ее назначения и основных функций.
3. Технические характеристики, например, используемые технологии, платформы и так далее.
4. Производительность и сведения об интеграции с другими ИТ-системами.
5. Информация о безопасности, включая применяемые методы защиты от взломов и утечек данных.
6. Данные о сопровождении и технической поддержке.
7. Данные о поставщике, лицензионном соглашении, дополнительных модулях и плагинах.

4. Разработка архитектурных моделей

Архитектурные модели представляют, как различные компоненты взаимодействуют друг с другом в рамках общей архитектуры.

5. Разработка графических моделей по взаимодействиям ИТ-систем

После разработки архитектурных моделей, которые представляют, как различные компоненты взаимодействуют друг с другом в рамках общей архитектуры, необходимо создать графические модели этой архитектуры. Важно подробно изобразить основные связи данных между ИТ-системами и базами данных, ИТ-системами организации, как внутри, так и с внешними субъектами. Дополнительно можно отразить данные о сервисах, протоколах и так далее.

6. Разработка каталога баз данных всех типов и параметров

Для обеспечения оптимальной производительности любых операций, связанных с базами данных, важна разработка иерархического каталога, который содержит все базы данных, используемые различными приложениями, а также их соответствующие параметры:

- емкость хранилища,
- уровень шифрования,
- параметры репликации и так далее.

Хорошо организованный каталог гарантирует, что все базы данных настроены правильно в соответствии с их требованиями, что помогает повысить эффективность и сократить количество ошибок, связанных с неправильной конфигурацией.

7. Создаем техническую архитектуру и разрабатываем дополнительные модели

Техническая архитектура отвечает за эффективное использование вычислительных ресурсов и определяет необходимые инфраструктурные ресурсы и сервисы, которые обеспечат правильную работу приложений и передачу данных между ними.

При описании технической архитектуры рассматриваются следующие аспекты:

- Компьютерная инфраструктура, включающая базовые инфраструктурные сервисы, серверное и пользовательское оборудование, хранилища данных, серверные платформы и ЦОД,

системное ПО;
- Сетевая инфраструктура, включающая локальные сети, корпоративные сети передачи данных, телефонию, видеоконференции, подключение к Интернету;
- Инженерная инфраструктура.

Основное назначение технической архитектуры – это гарантия стабильных сервисов на территории всего предприятия.

8. Объединение ИТ и бизнес-архитектуры для повышения эффективности

Чтобы убедиться, что инвестиции в технологии осуществляются эффективно, важно установить связи между техническими основами и бизнес-архитектурой. Также этот шаг помогает снизить операционные затраты, поскольку любые избыточные инвестиции, возникающие из-за отсутствия четкой видимости, теперь становятся очевидными, что позволяет избежать их на будущих этапах планирования.

Установить отношения между всеми элементами помогают специальные решения. Некоторые из популярных инструментов для моделирования архитектуры предприятия включают:

- Archi,
- Modelio,
- UML,
- BPMN,
- ERD (диаграммы сущность-связь),
- SysML,
- Enterprise Architect,
- ARIS или Business Studio.

Для начала подойдут открытые и бесплатные решения — Archi и Modelio.

Есть платные варианты для моделирования архитектуры предприятия с более широким набором возможностей:

- Enterprise Architect,
- ARIS,
- Business Studio.

Для того, чтобы их использовать нужен опыт. Решения поддерживают бесшовную связь с другими системами и ориентированы на использование в крупных проектах.

9. Обеспечение безопасности бизнеса

Любое успешное внедрение требует учета операционных рисков, которые связаны с использованием новых технологий. Также мер безопасности, необходимых для защиты конфиденциальных данных. Разработка должна проводиться с учетом обоих этих аспектов. Управление рисками позволяет организации определить, в какой степени потенциальные события повлияют на достижение её целей. Перечень рисков для каждого проекта уникальный и в среднем состоит из 10-20 различных факторов. Пять основных рисков: внутренние погрешности планирования, низкая производительность, нарушение спецификации, изменения требований и перемены в кадрах. Оценка рисков и продолжительность работы с ними зависят от размера и длительности проекта.

10. Разработка нормативных документов

После того, как были созданы протоколы безопасности, необходимо разработать нормативные документы, которые соответствуют требованиям законодательства. Тщательно проработанная документация снижает риски.

Список документов может включать:

1. Положение об ИТ-архитектуре и всех связях,
2. Порядок передачи ПО в эксплуатацию и вывода из нее,
3. Порядок установки, модификации и обслуживания объектов ИТ-инфраструктуры,
4. Регламент доработки, тестирования и внесения изменений в ИТ-системы,
5. Положение о распределении прав доступа к ИТ-системам,
6. Положение об информационной безопасности и многое другое.

11. Аудит ИТ-архитектуры по чек-листу

Один из последних этапов - проведение аудита информационной архитектуры, согласно предварительно созданному контрольному списку. Необходимо разработать и заполнить чек-лист с детальными требованиями к ИТ-архитектуре. На его основе проводить анализ выполненной работы, также включать туда задачи на следующий период.

12. Сбор новых задач от департаментов

Последнее, но не менее важное организации должны постоянно собирать новые требования. После сдачи основных задач, появляются новые. Чтобы ничего не упустить, действовать эффективно, необходимо сгруппировать и ранжировать их по важности и срочности. Срочные задачи могут оперативно выполняться в рамках методологий гибкого управления Agile (Scrum, Kanban). Крупные глобальные задачи включаются в план развития ИТ.

Метод ADD

Attribute-Driven Design (ADD) — представляет собой пошаговый метод проектирования программной архитектуры системы с использованием программного обеспечения.

Метод проектирования программной архитектуры системы состоит из следующих элементов:

- **Plan (планирование)** — на основе атрибутов качества устанавливаются архитектурные элементы;
- **Do (выполнение)** — архитектурные элементы реализуются для проверки атрибутов качества;
- **Check (проверка)** — проверяются результаты работы предыдущего пункта.

Этот процесс повторяется до тех пор, пока не будут сохранены все архитектурно значимые элементы.

Шаг 1. Процесс анализа введенных данных

На этом этапе важно собрать запросы к архитектуре. Источник — данные анкетирования руководителей проектов, анализ бизнес-целей и истории использования.

Условно вопросы можно разделить на три раздела:

1. **Производительность** (количество пользователей, которые будут работать с установкой; пиковые нагрузки, время ответа).
2. **Безопасность** (шифрование хранимых данных, двухфакторная аутентификация).
3. **Доступность** (допустимо ли временное отключение).

Важно четко выяснить, что необходимо заказчику. Например, не просто «безопасный сайт», а настроить «двухфакторную аутентификацию».

Шаг 2. Изложение необходимых условий к подсистеме

Далее необходимо оценить значимость требований, исходя из их важности для бизнеса и влияния на архитектуру, используя шкалу H (высокий), M (средний) и L (низкий). Каждое требование будет обозначаться двухбуквенным кодом, соответствующим его уровню важности, таким как HH, HM, HL, MH, MM. Большое количество требований с обозначением HH указывает на высокий риск неудачи проекта.

В конечном итоге, цель проекта должна быть ясно определена, все данные и требования собраны, приоритизированы, корректны и реализуемы.

Шаг 3. Определяем фрагмент системы для проектирования

Определяем наиболее важные направления на основе нескольких факторов, включая риски и сложность реализации, значимость для бизнеса, критический путь и точность проработки требований.

Шаг 4. Выбираем подходящий вариант

Пункт, который требует от ИТ-архитектора глубоких знаний. Проанализировать преимущества и недостатки можно с помощью простой

таблицы.

Для опоры используйте следующие вопросы:

- Если ли уступки при использовании?
- Как соотносятся модели друг с другом?
- Можно ли использовать любой шаблон А или шаблон Б, но не оба (являются ли они взаимоисключающими)?

	Паттерн 1		**Паттерн 2**	
	Плюсы	Минусы	Плюсы	Минусы
Требование 1				
Требование 2				

Шаг 5. Закрепляем компоненты, которые были выбраны в рамках концепции, определяем обязанности и интерфейсы между ними

Создаем по одному инстансу продукта, сервиса или компонента. Один инстанс — одно требование. Определяем области ответственности для каждого элемента, описываем связи между ними и выбираем интерфейс, который будет использоваться для взаимодействия между компонентами. Выявляем ресурсы, которые потребуются для выполнения задачи.

Шаг 6. Создаем схему решения с небольшим описанием

Создаем документ, где описываем все модули, подсистемы и их связь между собой.

Шаг 7. Анализируем проделанную работу

На этом этапе важно убедится, что все необходимые требования выполняются, атрибуты покрыты. Если все прошло удачно, то конвертируем результат в дизайн проекта. Если нет, то повторяем шаги 1-7.

Вывод

Данная методология необходима для сложных проектов. Она обеспечивает снижение затрат на разработку в будущем, за счет выбора правильных решений и инструментов еще на этапе проработки концепции.