Лабораторная работа №3. Проектирование архитектуры инфокоммуникационной системы

Цель работы

Для выбранного варианта инфокоммуникационной системы (основываясь на результатах первых двух работ) разработать и обосновать архитектуру прикладной инфокоммуникационной системы с учётом уровней, компонентов, взаимодействий и используемых стандартов, сформировать представление о подходах UML и C4 для визуального моделирования системной архитектуры.

Задание на выполнение лабораторной работы

Задание 1. Разработка модели архитектуры согласно методологии Zachman Framework

Zachman Framework — это одна из классических и наиболее известных методологий для систематизации и структурирования архитектуры информационных систем. Был разработан Джоном Закманом в 1987 году и представляет собой матричную модель, которая описывает систему с разных точек зрения и уровней детализации.

Структура Zachman Framework

Результаты работы по методологии Zachman Framework представляются таблицей:

Вопросы (Что? Как? Где? Кто? Когда? Почему?)	**Описание столбцов**
Что? (Data)	Какие данные используются в системе?
Как? (Function)	Какие процессы/функции выполняются?
Где? (Network)	Где происходит обработка/хранение данных?
Кто? (People)	Кто участвует в системе (роли, организации)?
Когда? (Time)	Когда происходят события/процессы (временные рамки)?
Почему? (Motivation)	Почему эти процессы реализуются (цели, правила)?
Уровни абстракции (строки)	**Описание строк**
Scope (План)	Перспектива бизнес-контекста, обзор высшего уровня
Business Model (Бизнес-модель)	Описание бизнес-процессов и правил
System Model (Системная модель)	Модель системы с точки зрения проектирования
Technology Model (Технологии)	Выбор технологий и архитектуры
Detailed Representations (Детали)	Подробные спецификации и схемы
Functioning Enterprise (Работающая система)	Реализованная система

Требуется заполнить предлагаемую ниже таблицу, дать максимально конкретные ответы по каждой ячейке матрицы:

Уровень/Вопрос	**Что? (Data)**	**Как? (Function)**	**Где? (Network)**	**Кто? (People)**	**Когда? (Time)**	**Почему? (Motivation)**
Scope (Область)						
Business Model (Бизнес-модель)						
System Model (Системная модель)						
Technology Model (Технологии)						
Detailed Representation (Детализация)						
Functioning Enterprise (Рабочая система)						

Рекомендации по разработке модели архитектуры

Архитектурные решения — это выбор и обоснование ключевых структурных характеристик системы, которые определяют её основу:

- Структуру компонентов и взаимодействие между ними,
- Технологический стек,
- Модели развертывания и интеграции.

Пример:

- Выбор архитектурного стиля: микросервисы, монолит, событийно-ориентированная архитектура.
- Определение способов масштабирования и отказоустойчивости.
- Выбор протоколов взаимодействия: REST, gRPC, AMQP.

Согласно **Zachman Framework** формируется матрица представлений системы (6 вопросов × 6 уровней абстракции), где архитектурные решения в основном находятся на уровнях "Архитектор" (System Model) и частично на уровне "Дизайнер" (Technology Model). Это помогает системно рассмотреть архитектуру с разных точек зрения:

- **Что** (данные, информационная модель)
- **Как** (функции, процессы)
- **Где** (сеть, расположение компонентов)
- **Кто** (пользователи, роли)
- **Когда** (временные аспекты, синхронизация)
- **Почему** (мотивация, цели, бизнес-правила)

Например, если проектируется IoT-платформа для управления транспортной сетью, то Zachman поможет:

- На уровне **"Архитектор"** зафиксировать:
  - Что: модель данных о транспортных узлах, датчиках, потоках.
  - Как: распределённая обработка данных, потоковая аналитика.
  - Где: облачная инфраструктура + локальные узлы.
- На уровне **"Дизайнер"** перейти к проектным деталям:
  - Конкретные протоколы (MQTT, gRPC),
  - Выбор СУБД (TimeSeries DB для телеметрии),
  - Схемы деплоймента в Kubernetes.

Таким образом, гарантируется, что архитектурные решения будут полными и согласованными, так как каждая ячейка матрицы заставляет проектировщика подумать о конкретном аспекте. Но следует учитывать, что сам по себе Zachman — это не метод проектирования, а метод классификации знаний о системе, он не указывает, как делать архитектуру, а лишь помогает убедиться, что ничего не забыто.

Zachman — это хороший инструмент, чтобы систематизировать и проверить полноту архитектурных решений, но для их проработки лучше использовать **UML, SysML, C4** или методологии типа **RUP, TOGAF**.

Задание 2. Архитектурное проектирование

Требуется, основываясь на результатах первого задания, разработать архитектуру системы. Для решения этой задачи заполните следующую таблицу:

Этап	**Что требуется**
1. Постановка задачи	Описание системы, функций, пользователей, ключевых требований
2. Определение компонентов	Перечислите основные логические и физические компоненты
3. Уровневая архитектура	Опишите 3–5 уровней архитектуры: функциональный, логический и др.
4. Взаимодействие компонентов	Составьте **UML-диаграммы**: ● Sequence (последовательности) – **системную** диаграмму последовательности ● Component (компонентов) ● Deployment (развертывания) и диаграммы **нотации C4** (Level 1–3): ● Context Diagram (Level 1) ● Container Diagram (Level 2) ● Component Diagram (Level 3) Сравните подходы по следующим критериям: ● Уровень детализации

Этап	Что требуется
	● Понятность для разных ролей (разработчик, аналитик, заказчик и др.) ● Простота создания и поддержки ● Влияние на документацию и сопровождение проекта ● Вывод о применимости нотаций для вашей задачи (условия и ограничения)
5. Технологический стек	Обоснуйте выбор протоколов, интерфейсов, форматов данных
6. Безопасность	Укажите основные угрозы и подходы к защите
7. Обоснование архитектуры	Выполните анализ trade-off (компромиссных) решений, сформулируйте выводы и рекомендации дальнейшего развития

Задание 3. Анализ архитектурных и проектных решений

Цель данного задания: осознанно различать архитектурный уровень и уровень проектирования, суть принимаемых архитектурных и проектных решений.

Требуется выявить решения на разработку вашей системы и указать, к какому типу оно относится: архитектурное или проектные, обосновать свой выбор.

Архитектурные решения были определены в первом задании.

Проектные решения — это детализация реализации внутри выбранной архитектуры. Они описывают, как именно будут реализованы функциональные элементы, соответствуя архитектурным принципам.

Пример:

- Определение структуры конкретных классов, модулей, алгоритмов.
- Выбор библиотеки для обработки видео в системе видеонаблюдения.
- Оптимизация структуры БД под конкретные сценарии использования.

Разница между архитектурными и проектными решениями в том, что архитектура отвечает на вопрос "Что и в какой общей структуре", а проектирование — "Как именно конкретное архитектурное решение реализовать".

Результаты анализа можно оформить в виде таблицы:

Решение	Архитектурное или проектное?	Обоснование

Например, для системы «интернет-магазин…» решения могут быть разделены на архитектурные и проектные следующим образом:

Решение	Архитектурное или проектное?	Обоснование
Выбор микросервисной архитектуры	Архитектурное	Стратегический выбор стиля
Использование шаблона Repository	Проектное	Конкретная реализация доступа к данным

REST между сервисами	Архитектурное	Способ взаимодействия
Введение DTO между слоями	Проектное	Техническое проектное решение
Авторизация через JWT	Архитектурное (может быть на границе)	Общее решение безопасности
Выбор PostgreSQL	Архитектурное	Выбор технологии на уровне компонентов
Разделение логики по модулям catalog / order	Проектное	Структурирование внутри выбранной архитектуры

✔ Укажите свои решения! Нельзя просто так взять и переписать пример!

В выводах укажите ответы на следующие вопросы:

1. Почему выбор архитектуры нельзя менять в ходе проекта без значительных последствий, или все-таки можно?

2. Какие проектные решения зависят от архитектурных?

3. Какие проектные решения можно адаптировать в разных архитектурах?

4. Какие решения показались вам пограничными, почему?

Задание 4. Трассировка требований

Свяжите требования с компонентами архитектуры с помощью диаграммы трассировки.

Отчетность по результатам занятия

По результатам работы необходимо оформить и защитить отчет.

В отчете должны быть приведены результаты выполнения заданий (для графического представления рекомендуется использовать результаты практического занятия №3).

Литература

1. *Ларман К.* Применение UML и шаблонов проектирования. 2-е издание.: Пер. с англ. — М. : Издательский дом “Вильямс”, 2004. – 624 с.
2. The C4 Model Website (официальный сайт) [Электронный ресурс] Режим доступа: https://c4model.com.
3. *Браун Симон (Simon Brown)* Software Architecture for Developers [Электронный ресурс] Режим доступа: https://leanpub.com/software-architecture-for-developers
4. Structurizr DSL (язык описания архитектуры в формате C4). [Электронный ресурс] Режим доступа: https://github.com/structurizr/dsl
5. PlantUML C4 Extensions (расширение для PlantUML с C4-нотацией) [Электронный ресурс] Режим доступа: https://github.com/plantuml-stdlib/C4-PlantUML