ОСНОВЫ WEB ПРОГРАММИРОВАНИЯ

02.03.03 -Математическое обеспечение и администрирование информационных систем, направленность (профиль) -разработка и администрирование информационных систем

http//vikchas.ru

Лекция 2. «Инструменты и методы проектирования и дизайна информационной системы, большие фундаментальные модели ИИ»

Часовских Виктор Петрович
доктор технических наук, профессор кафедры ШИиКМ, ФГБОУ ВО «Уральский государственный экономический университет

Екатеринбург 2026

Информационная система: понятие и общая характеристика

Информационная система, или **ИС**, — это совокупность программных, технических, организационных и информационных средств, предназначенных для сбора, хранения, обработки, передачи и анализа данных.

Информационные системы используются в бизнесе, образовании, медицине, государственном управлении, промышленности, банковской сфере и других областях.

Информационная система на базе больших фундаментальных моделей ИИ — это система, в которой ключевые функции обработки информации выполняются с использованием: больших языковых моделей, LLM; мультимодальных моделей для текста, изображений, аудио и видео; embedding-моделей для семантического поиска; генеративных моделей; моделей классификации, извлечения и анализа данных; агентных ИИ-компонентов.

Такая система может выполнять: поиск и анализ информации; генерацию текстов и документов; интеллектуальную поддержку принятия решений; автоматизацию бизнес-процессов; обработку естественного языка; анализ изображений и документов; диалоговое взаимодействие с пользователем.

Большая языковая модель (LLM, Large Language Model) — это фундаментальная модель искусственного интеллекта, обученная на огромных объёмах текстовых и/или мультимодальных данных и способная понимать, генерировать, преобразовывать и анализировать естественный язык.

Примеры LLM: GPT-4 / GPT-4o; Claude; Gemini; Llama; Mistral; Qwen; DeepSeek; GigaChat; YandexGPT.

LLM — это нейросетевая модель, которая предсказывает наиболее вероятное продолжение текста на основе контекста. Несмотря на простую формулировку, современные LLM способны выполнять сложные задачи: отвечать на вопросы; писать тексты; переводить; резюмировать документы; классифицировать информацию; извлекать факты; писать программный код; анализировать данные; вести диалог; работать как интеллектуальный помощник; управлять внешними инструментами через API.

Модель называется большой, потому что она имеет: огромное число параметров — от миллионов до сотен миллиардов; большой объём обучающих данных; сложную архитектуру; высокую вычислительную стоимость обучения и эксплуатации.

Параметры модели — это числовые коэффициенты нейронной сети, которые настраиваются во время обучения. Чем больше параметров, тем потенциально выше способность модели усваивать сложные закономерности языка и знаний.

Большинство современных LLM основаны на архитектуре **Transformer**.

Ключевая идея Transformer — механизм **self-attention**, или самовнимания. Он позволяет модели учитывать связи между словами и частями текста независимо от их расстояния друг от друга.

Например, в предложении:

«Анна положила книгу на стол, потому что она была тяжёлой»

модель должна понять, что «она» относится к «книге», а не к «Анне» или «столу». Механизм внимания помогает учитывать такие связи.

Инструменты и методы проектирования и дизайна информационной системы на основе больших фундаментальных моделей искусственного интеллекта

Научные достижения Кащенко Михаила Петровича профессора Уральского государственного лесотехнического университета

Ответ ChatGPT 4

Кащенко Михаил Петрович — профессора Уральского государственного лесотехнического университета, известный своими научными достижениями в области лесоведения, лесного хозяйства и охраны окружающей среды. Его работы охватывают широкий спектр тем, включая изучение экологических процессов, технологии лесоводства и устойчивое управление лесными ресурсами.
Некоторые ключевые направления его научной деятельности:
1. Лесоведение: Исследования в области биологии и экологии лесных экосистем, включая изучение растительности и её взаимодействия с окружающей средой.
2. Устойчивое управление лесами: Разработка методов и подходов для рационального использования лесных ресурсов с минимальным воздействием на экосистему.

3. Охрана окружающей среды: Вклад в исследования, касающиеся проблем охраны лесов и защиты биоразнообразия.
4. Технологии лесоводства: Исследования новых технологий для повышения эффективности лесного хозяйства, улучшения качества лесных насаждений и восстановления поврежденных лесов.
Научные публикации и проекты, связанные с его именем, внесли значительный вклад в развитие лесной науки и практики, и он продолжает активно участвовать в научной и образовательной деятельности, подготовке новых специалистов в области лесного хозяйства.

LLM получает входной текст, называемый **prompt** или **промпт**, и генерирует ответ по токенам.

Токен — это часть текста: слово, часть слова, символ или знак препинания.

Пример:

Вход: «Столица Франции —»
Выход: «Париж»

На каждом шаге модель оценивает вероятности следующих токенов и выбирает наиболее подходящий.

Пример машинного обучения.

Допустим, у нас есть маленький обучающий контент:

кот ест рыбу
кот пьет молоко
собака ест мясо
собака пьет воду

Словарь модели:
[***кот, собака, ест, пьет, рыбу, молоко, мясо, воду***]

Представим, что модель обучалась **предсказывать** следующее слово.
После обучения у неё могла бы получиться такая матрица весов переходов:

Текущее слово ↓ / Следующее слово →	**кот**	**собака**	**ест**	**пьет**	**рыбу**	**молоко**	**мясо**	**воду**
кот	0.01	0.01	0.45	0.40	0.03	0.03	0.03	0.04

Текущее слово ↓ / Следующее слово →	**кот**	**собака**	**ест**	**пьет**	**рыбу**	**молоко**	**мясо**	**воду**
собака	0.01	0.01	0.43	0.42	0.03	0.03	0.04	0.03
ест	0.01	0.01	0.01	0.01	0.47	0.02	0.45	0.02
пьет	0.01	0.01	0.01	0.01	0.02	0.46	0.02	0.46
рыбу	0.12	0.12	0.12	0.12	0.13	0.13	0.13	0.13
молоко	0.12	0.12	0.12	0.12	0.13	0.13	0.13	0.13
мясо	0.12	0.12	0.12	0.12	0.13	0.13	0.13	0.13
воду	0.12	0.12	0.12	0.12	0.13	0.13	0.13	0.13

Эта матрица показывает, насколько вероятно одно слово после другого.

Например:

- после слова «кот» модель чаще выбирает: ест — 0.45; пьет — 0.40.
- после слова «ест» модель чаще выбирает: рыбу — 0.47; мясо — 0.45.
- после слова «пьет»: молоко — 0.46; воду — 0.46.

То есть если модель получила на входе:

кот ест
она может продолжить:
кот ест рыбу
или реже:

кот ест мясо

В настоящих LLM матрицы весов намного сложнее, но всегда после обучения **веса** отражают закономерности из обучающего контента.

Основные возможности LLM

Генерация текста

LLM может создавать: статьи; письма; отчёты; инструкции; описания товаров; учебные материалы; сценарии; юридические черновики.

Анализ текста

Модель может: находить ключевые идеи; делать краткое резюме; извлекать сущности; определять тональность; классифицировать обращения; сравнивать документы.

Ответы на вопросы

LLM может отвечать на вопросы пользователя, используя: знания, полученные при обучении; контекст, переданный в запросе; внешние базы знаний через RAG; инструменты и API и т.л.

Обучение LLM

Обычно обучение LLM включает несколько этапов.

Предобучение

Модель обучается на больших массивах данных: книги; статьи; сайты; документация; код; научные публикации; диалоги.

На этом этапе модель учится языковым закономерностям, фактам и структурам.

Дообучение

После пред обучения модель может до обучаться на специализированных данных: медицинских; юридических; финансовых; технических; корпоративных.

Это повышает качество работы в конкретной предметной области.

Типы LLM

Закрытые коммерческие модели

Доступны через API или облачные платформы.

Примеры: GPT-4o; Claude; Gemini; Mistral Large; GigaChat; YandexGPT.

Преимущества: высокое качество; простая интеграция; поддержка; масштабируемость.

Недостатки: стоимость API; зависимость от провайдера; вопросы конфиденциальности; ограниченный контроль над моделью.

Открытые модели

Можно развернуть локально или в собственной инфраструктуре.

Примеры: Llama; Mistral; Qwen; ***DeepSeek***; Gemma.

Преимущества: контроль над данными; гибкая настройка; возможность локального запуска; отсутствие зависимости от одного провайдера.

Недостатки: нужны вычислительные ресурсы; сложнее эксплуатация; требуется настройка; качество может уступать ведущим закрытым моделям.

LLM в информационных системах

В информационных системах LLM используется как интеллектуальный компонент.

Типовые сценарии: корпоративный чат-бот; поиск по документам; автоматизация поддержки клиентов; генерация отчётов; анализ обращений; интеллектуальный помощник сотрудника; ассистент программиста; автоматическое заполнение форм; проверка договоров; анализ регламентов; генерация SQL-запросов; голосовой ассистент; агентная система для выполнения задач.

Ограничения LLM

Несмотря на мощные возможности, LLM имеет ограничения.

Галлюцинации

Модель может уверенно сгенерировать неверный ответ.

Ограниченный контекст

У каждой модели есть максимальный размер входного текста.

Зависимость от качества данных

Если данные неточные, устаревшие или неполные, ответы будут хуже.

Отсутствие настоящего понимания

LLM ***оперирует статистическими закономерностями и представлениями, а не человеческим пониманием в строгом смысле***.

Большая языковая модель (LLM) — это программа для ЭВМ нейросетевой модели искусственного интеллекта, обученная на больших объёмах текстовых данных и предназначенная для понимания, генерации и обработки естественного языка. LLM основана преимущественно на архитектуре Transformer и используется в информационных системах для автоматизации коммуникаций, поиска знаний, анализа документов, генерации текстов, поддержки принятия решений и взаимодействия с пользователями в естественной языковой форме.

ТРАДИЦИОННЫЕ ЭТАПЫ

Основные этапы проектирования ИИ-ориентированной информационной системы

Анализ требований

На этом этапе определяются:

- цели системы;
- пользователи и их роли;
- бизнес-процессы;
- источники данных;
- требования к точности и качеству ответов;
- требования к безопасности;
- требования к масштабируемости;
- ограничения по стоимости и времени отклика.

Методы:

- интервью с заказчиком;
- анализ предметной области;
- моделирование бизнес-процессов;
- user stories;
- use case modeling;
- анализ рисков;
- формирование требований к данным и ИИ-компонентам.

Инструменты:

- Jira;
- Confluence;
- Miro;
- Notion;
- Figma;
- Draw.io;
- Enterprise Architect

Архитектурное проектирование

Типовая архитектура системы

Информационная система на основе фундаментальных моделей обычно включает:

Пользовательский интерфейс

- веб-приложение;
- мобильное приложение;
- чат-интерфейс;
- голосовой интерфейс.

Прикладной слой

- бизнес-логика;
- API;
- сервис авторизации;
- управление сессиями;
- маршрутизация запросов.

ИИ-слой

- LLM;
- RAG-модуль;
- embedding-модель;
- агентная логика;
- классификаторы;

- модуль постобработки ответов.

Слой данных

- реляционные базы данных;
- объектные хранилища;
- векторные базы данных;
- графовые базы знаний;
- журналы и события.

Интеграционный слой

- REST API;
- GraphQL;
- message broker;
- ETL/ELT-пайплайны;
- коннекторы к внешним системам.

Слой мониторинга и безопасности

- логирование;
- аудит;
- контроль доступа;
- observability;
- оценка качества ИИ-ответов;
- защита от prompt injection.

Инструменты проектирования информационных систем

Инструменты проектирования ИС — это программные средства, которые помогают аналитикам, архитекторам, разработчикам и дизайнерам описывать будущую систему, моделировать её структуру, процессы, данные, интерфейсы и взаимодействия.

CASE-средства

CASE-средства — это инструменты автоматизированного проектирования информационных систем.

CASE расшифровывается как **Computer-Aided Software Engineering**, то есть автоматизированная разработка программного обеспечения.

CASE-средства позволяют:

- строить диаграммы;
- описывать бизнес-процессы;
- проектировать базы данных;
- моделировать архитектуру системы;
- документировать требования;
- формировать техническую документацию;
- иногда автоматически генерировать программный код или SQL-скрипты.

Примеры CASE-средств:

- **Enterprise Architect**;
- **Visual Paradigm**;
- **IBM Rational Rose**;
- **Erwin Data Modeler**;
- **Oracle SQL Developer Data Modeler**;
- **PowerDesigner**.

Инструменты UML-моделирования

UML, или Unified Modeling Language, — это унифицированный язык моделирования, который используется для описания структуры и поведения программных систем.

С помощью UML можно представить систему в виде диаграмм.

Основные UML-диаграммы:

- диаграмма вариантов использования;
- диаграмма классов;
- диаграмма объектов;
- диаграмма последовательности;
- диаграмма активности;
- диаграмма состояний;

- диаграмма компонентов;
- диаграмма развёртывания.

Инструменты UML-моделирования:

- StarUML;
- PlantUML;
- Visual Paradigm;
- Enterprise Architect;
- Draw.io / diagrams.net;
- Lucidchart.

UML-инструменты помогают формализовать требования и упростить коммуникацию между заказчиками, аналитиками и разработчиками.

Инструменты BPMN-моделирования

BPMN, или Business Process Model and Notation, — это нотация для моделирования бизнес-процессов.

BPMN используется, когда нужно описать последовательность действий в организации или системе.

С помощью BPMN можно моделировать:

- обработку заказа;
- согласование документа;
- регистрацию пользователя;
- процесс оплаты;
- обработку заявки;
- выдачу товара со склада.

Инструменты BPMN-моделирования:

- **Bizagi Modeler**;
- **Camunda Modeler**;
- **ARIS**;
- **Draw.io / diagrams.net**;
- **Visual Paradigm**;
- **ELMA BPM**.

BPMN особенно полезен при проектировании корпоративных систем, систем документооборота, CRM, ERP и workflow-систем.

Инструменты проектирования баз данных

База данных является важной частью большинства информационных систем. Поэтому на этапе проектирования необходимо определить структуру данных, таблицы, связи, ключи и ограничения.

Инструменты проектирования баз данных позволяют:

- строить ER-диаграммы;
- описывать сущности и атрибуты;
- задавать первичные и внешние ключи;
- проектировать связи между таблицами;
- нормализовать структуру данных;
- генерировать SQL-скрипты;
- документировать модель данных.

Примеры инструментов:

- **Erwin Data Modeler**;
- **Oracle SQL Developer Data Modeler**;
- **MySQL Workbench**;
- **pgModeler**;
- **DBeaver**;
- **dbdiagram.io**;
- **Vertabelo**.

Инструменты прототипирования интерфейсов

Перед разработкой интерфейса часто создают прототипы экранов. Это помогает заранее увидеть, как пользователь будет взаимодействовать с системой.

Инструменты управления требованиями

Требования описывают, что должна делать информационная система. Для управления требованиями используются специализированные системы.

Они позволяют:

- фиксировать требования;
- отслеживать изменения;
- связывать требования с задачами;
- контролировать выполнение;
- хранить историю согласований.

Примеры инструментов:

- **Jira**;
- **Confluence**;
- **YouTrack**;
- **Trello**;
- **Azure DevOps**;
- **IBM Engineering Requirements Management DOORS**.

Инструменты управления проектом и командной работой

При создании ИС важно управлять задачами, сроками, командой и документацией.

Примеры инструментов:

- Jira;
- YouTrack;
- Trello;
- Asana;
- Notion;
- Confluence;
- MS Project;
- Redmine.

Они позволяют распределять задачи между участниками проекта, контролировать сроки, фиксировать проблемы и отслеживать ход выполнения работ.

Методы проектирования информационных систем

Методы проектирования ИС — это подходы и принципы, которые определяют, как именно анализировать предметную область, формировать требования, строить модели и разрабатывать архитектуру будущей системы.

Структурное проектирование

Структурное проектирование основано на декомпозиции системы на функции и процессы.

Главная идея — разложить сложную систему на более простые функциональные блоки.

При структурном подходе используются:

- функциональные модели;
- диаграммы потоков данных;
- иерархические схемы функций;
- описание входных и выходных данных.

Основные элементы структурного проектирования:

- процесс;
- поток данных;
- хранилище данных;
- внешняя сущность.

Пример: система обработки заказов может быть разделена на функции:

- приём заказа;
- проверка наличия товара;
- расчёт стоимости;
- оформление оплаты;
- передача заказа в доставку.

Преимущества структурного метода:

- понятность;
- удобство описания бизнес-процессов;
- возможность пошаговой детализации;

- подходит для систем с чётко определёнными функциями.

Недостатки:

- хуже подходит для сложных объектно-ориентированных систем;
- затрудняет повторное использование компонентов;
- изменения в одной части могут влиять на другие части.

Объектно-ориентированное проектирование

Объектно-ориентированное проектирование, или ООП-проектирование, рассматривает систему как совокупность объектов, которые имеют свойства и поведение.

Объект — это сущность предметной области или программной системы.

Примеры объектов:

- пользователь;
- заказ;
- товар;
- платёж;
- документ;
- счёт;
- клиент.

Каждый объект имеет:

- атрибуты;
- методы;
- связи с другими объектами.

Основные принципы объектно-ориентированного проектирования:

- **инкапсуляция** — объединение данных и методов внутри объекта;
- **наследование** — создание новых классов на основе существующих;
- **полиморфизм** — возможность использовать единый интерфейс для разных типов объектов;
- **абстракция** — выделение существенных характеристик объекта.

Для объектно-ориентированного проектирования часто используются UML-диаграммы:

- диаграмма классов;
- диаграмма объектов;
- диаграмма последовательности;
- диаграмма состояний;
- диаграмма компонентов.

Преимущества ООП-проектирования:

- высокая гибкость;
- удобство сопровождения;
- повторное использование кода;
- близость к современным языкам программирования.

Недостатки:

- требует хорошей квалификации проектировщиков;
- может быть сложнее для понимания заказчиком;
- при неправильном применении приводит к избыточной сложности.

Процессный подход

Процессный подход рассматривает деятельность организации как совокупность бизнес-процессов.

Бизнес-процесс — это последовательность действий, направленных на достижение определённого результата.

Примеры бизнес-процессов:

- оформление заказа;
- согласование договора;
- обработка заявки;
- приём сотрудника на работу;
- выдача товара со склада;
- регистрация пациента.

Процессный подход используется для:

- анализа текущей деятельности организации;

- выявления проблем и узких мест;
- автоматизации процессов;
- оптимизации работы сотрудников;
- построения workflow-систем.

При процессном проектировании часто применяются BPMN-диаграммы.

Преимущества процессного подхода:

- хорошо показывает реальную работу организации;
- понятен бизнес-пользователям;
- помогает улучшить процессы до автоматизации;
- подходит для корпоративных ИС.

Моделирование данных

Моделирование данных — это метод проектирования структуры информации, которая будет храниться и обрабатываться в системе.

Обычно выделяют три уровня модели данных:

Концептуальная модель

Описывает основные сущности предметной области без технических деталей.

Пример сущностей:

- клиент;
- заказ;
- товар;
- поставщик;
- платёж.

Логическая модель

Описывает атрибуты сущностей, связи между ними, ключи и ограничения, но ещё не привязана к конкретной СУБД.

Физическая модель

Описывает конкретную реализацию базы данных:

- таблицы;
- поля;

- типы данных;
- индексы;
- ограничения;
- связи;
- представления.

Моделирование данных помогает избежать дублирования информации, ошибок хранения и противоречий.

Итерационный и инкрементный методы

При итерационном подходе система создаётся постепенно. Разработка делится на циклы — итерации.

На каждой итерации команда:

- уточняет требования;
- проектирует часть системы;
- реализует функциональность;
- тестирует результат;
- получает обратную связь.

Инкрементный подход предполагает добавление новых функций по частям.

Например, интернет-магазин можно разрабатывать так:

- каталог товаров;
- регистрация пользователей;
- корзина;
- оформление заказа;
- оплата;
- доставка;
- личный кабинет;
- аналитика для администратора.

Преимущества:

- быстрый выпуск первых версий;

- возможность учитывать изменения требований;
- снижение рисков;
- постоянное улучшение продукта.

Архитектурное проектирование

Архитектурное проектирование определяет общую структуру информационной системы.

На этом этапе выбираются:

- архитектурный стиль;
- способ хранения данных;
- способы взаимодействия компонентов;
- технологии разработки;
- требования к безопасности;
- требования к масштабируемости;
- схема развёртывания.

Распространённые архитектурные подходы:

- монолитная архитектура;
- клиент-серверная архитектура;
- трёхуровневая архитектура;
- микросервисная архитектура;
- сервис-ориентированная архитектура;
- событийно-ориентированная архитектура.

Архитектурное проектирование влияет на производительность, надёжность, стоимость сопровождения и развитие системы.

Дизайн информационных систем

Дизайн ИС — это не только внешний вид интерфейса. В широком смысле дизайн информационной системы включает проектирование пользовательского взаимодействия, структуры экранов, логики навигации, визуального оформления, доступности и удобства использования.

UX-дизайн информационной системы

UX-дизайн, или User Experience Design, — это проектирование пользовательского опыта.

Цель UX-дизайна — сделать систему понятной, удобной и эффективной для пользователя.

UX-дизайнер отвечает на вопросы:

- кто будет пользоваться системой;
- какие задачи выполняет пользователь;
- какие действия он должен совершить;
- какие ошибки могут возникнуть;
- как сократить количество лишних шагов;
- как сделать интерфейс понятным.

Основные элементы UX-дизайна:

- исследование пользователей;
- сценарии использования;
- пользовательские истории;
- CJM — карта пути пользователя;
- информационная архитектура;
- прототипирование;
- usability-тестирование.

UI-дизайн информационной системы

UI-дизайн, или User Interface Design, — это проектирование визуального интерфейса системы.

UI-дизайн включает:

- цветовую схему;
- шрифты;
- кнопки;
- формы;
- таблицы;
- иконки;
- меню;

- модальные окна;
- уведомления;
- расположение элементов на экране.

Цель UI-дизайна — сделать интерфейс визуально понятным, аккуратным и единообразным.

Информационная архитектура

Информационная архитектура определяет, как в системе организована информация.

Она включает:

- структуру разделов;
- меню и навигацию;
- группировку функций;
- иерархию страниц;
- названия экранов;
- связи между разделами.

Пример структуры ИС интернет-магазина:

- Главная страница;
- Каталог;
- Карточка товара;
- Корзина;
- Оформление заказа;
- Личный кабинет;
- История заказов;
- Поддержка;
- Панель администратора.

Хорошая информационная архитектура помогает пользователю быстро находить нужные функции.

Usability и эргономика

Usability — это удобство использования системы.

Хорошая ИС должна быть:

- понятной;
- предсказуемой;
- быстрой в освоении;
- устойчивой к ошибкам пользователя;
- удобной для ежедневной работы;
- доступной для людей с разными возможностями.

Критерии usability:

- эффективность выполнения задач;
- скорость работы пользователя;
- количество ошибок;
- удовлетворённость пользователя;
- простота обучения.

Адаптивный дизайн

Современные информационные системы часто используются на разных устройствах:

- компьютерах;
- ноутбуках;
- планшетах;
- смартфонах;
- терминалах;
- промышленных панелях.

Адаптивный дизайн позволяет интерфейсу корректно отображаться на экранах разных размеров.

Дизайн безопасности

Дизайн ИС должен учитывать безопасность.

Это означает, что интерфейс и логика системы должны помогать защищать данные.

Примеры решений:

- сложные пароли;
- двухфакторная аутентификация;

- разграничение прав доступа;
- подтверждение критических операций;
- автоматический выход из системы;
- отображение предупреждений;
- журналирование действий пользователя.

Дизайн доступности

Доступность означает, что системой могут пользоваться люди с различными ограничениями.

Например:

- пользователи с нарушением зрения;
- пользователи с нарушением моторики;
- пожилые пользователи;
- пользователи с временными ограничениями.

Принципы доступности:

- достаточный контраст;
- читаемые шрифты;
- возможность работы с клавиатуры;
- текстовые описания элементов;
- понятные сообщения об ошибках;
- отсутствие зависимости только от цвета.