WEB-ТЕХНОЛОГИИ - ЛЕКЦИИ

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

http//vikchas.ru

Тема «Web- технологии и платформы их создания»

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

Екатеринбург 2025

В 50-ые годы 20-ого века началось производство **Ц**ЭВМ. В 1956 г. в СССР академиком Сергей Алексеевичем Лебедевым была создана самая быстродействующая ЭВМ – МЭСМ, далее БЭСМ, в настоящее время - ЭЛЬБРУС.

По состоянию на 2025 год в Мире более 21 миллиарда ЦЭВМ

Связь нового поколения в конце 50-х годов разработал академик Глушков В.М., сеть для всей страны ОГАС с использованием спутниковых и проводных каналов для этого он разработал лучший, для того времени, компьютер «Мир».

ОГАС это Российский Интернет, а МИР- персональный компьютер (выпущено порядка 300000 штук, патент купили американцы). Глушков В.М. разработал концепцию безбумажной технологии в управлении (монография издана в 1962 г.).

Прошло чуть менее четверти XXI века, и за это время свершилось несколько технологических прорывов.

Наибольшее развитие происходит в информационной сфере (IT), появились: Интернет, социальные сети, беспилотные летательные аппараты (направление БЛА), мобильный интернет, виртуальная и дополненная реальность, искусственный интеллект, электрические и беспилотные автомобили, искусственные органы и протезирование.

ИНТЕРНЕТ - это множество компьютеров, объединённых в сеть.

Первые в Мире сети ЭВМ появились в СССР, в 50-е года, в ПВО.

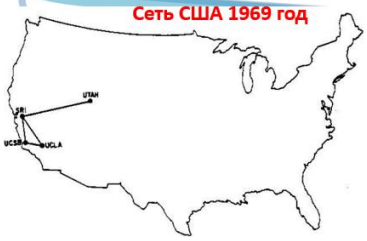
В США первая сеть создана военными **29 октября 1969 года**, вначале соединили Калифорнийском и Стэнфордском университете, далее подключили еще 2 университета. Компьютерная сеть была названа ARPANET, дальнейшее развитие её превратилась в ИНТЕРНЕТ.

Предлагаемый вариант Интернет (Российский) и персональная ЭВМ для этой сети

ЕОМ «Мир» (1964)

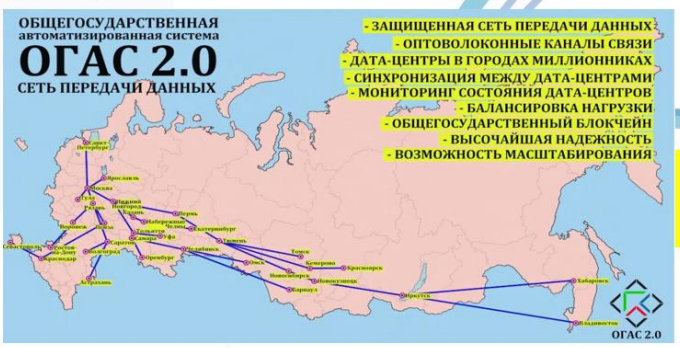
Сеть В.М. Глушкова, проект на конференции 1960 год, описание в монографии 1962 год и предложена Политбюро в 1962 году

Сеть США 1969 год



Проект 1980 года

Современный проект сети передачи данных 2024 г.

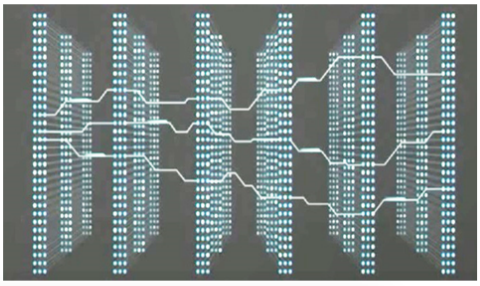


Современный Big Data РФ

Современность

Языковые модели GPT (генеративный предварительно обученный трансформер)

GPT-3 - 17,5 миллиардов параметров
GPT- 4 - 1,76 триллиона параметров
WuDao 2.0 - 1,76 триллиона параметров
GPT-5 - 17,5 триллиона параметров
суперкомпьютера Stargate с поддержкой ИИ $100 млрд, до 5 гигаватт энергии, запланировано пять таких суперкомпьютеров на территории США
DeepSeek-V3 - 671 миллиардов параметров



DeepSeek-V3 обладает возможностями, сопоставимыми с такими западными моделями, как GPT-4, однако тесты показывают, что она подвергается строгой государственной цензуре. Модель DeepSeek-V3 была представлена 26 декабря 2024 года, но обучение было выполнено до октября 2023 года.

Площадь – 200га

Охлаждение – 2 мл. л

Электро – 16 Гват

В марте 2025 года появилось сообщение о GPT-4 на русском языке

Нейросеть OpenAI ChatGPT4 на русском

Написано более 850 млн текстовых ответов, сгенерировано >300 млн изображений.

Компания OpenAI — владелец нейросети **ChatGPT**, в том числе её версии **ChatGPT-4.**

OpenAI —частная организация, права на использование и распространение модели принадлежат компании. Среди инвесторов **OpenAI** — технологические гиганты, в том числе Microsoft.

По состоянию на март 2025 года штаб-квартира компании **OpenAI** находится в **Сан-Франциско, Калифорния, США.**

Странная забота США о России

Моё «общение» с GPT-4

Общение с GPT-4

Вопрос - Известные примеры искусственного интеллекта
Ответ ChatGPT-4
1. IBM Watson:
2. Google Assistant / Apple Siri / Amazon Alexa:
3. DeepMind AlphaGo:
4. **ChatGPT**:
предназначена для генерации текста и общения на естественном языке. Они могут отвечать на вопросы, писать статьи, вести беседы и выполнять другие задачи, связанные с текстом.
5. **Autonomous Vehicles**:

Вопрос - Для функционирования GPT 4 что необходимо
Ответ ChatGPT-4
Для функционирования GPT-4 необходимы следующие элементы:
1. **Электронно-вычислительная машина (ЭВМ).**
2. **Программы.**
3. **Данные.**

Вопрос - В искусственном интеллекте есть разум
Ответ ChatGPT 4
В искусственном интеллекте на данный момент **нет разума**

Вопрос - Искусственный интеллект является программой для ЭВМ?
Ответ ChatGPT-4
Искусственный интеллект можно рассматривать как технологию или метод, реализованный с помощью **программного обеспечения** и выполняемого на ЭВМ

Общение с GPT-4

Вопрос - Искусственный интеллект может исказить сказки Пушкина

Ответ ChatGPT-4

Искусственный интеллект действительно может интерпретировать и адаптировать сказки Пушкина, внося в них новые элементы или изменяя сюжет, чтобы соответствовать современным трендам или различным стилям.

Вопрос: Можно доверять нейросетям в передаче фактической информации

Ответ ChatGPT-4

Доверять нейросетям в передаче фактической информации нужно с осторожностью. Хотя они могут быть полезными инструментами для поиска и обработки данных. Сложность в том, что нейросети, как и любой инструмент, могут быть полезными, но их результаты должны быть критически оценены и проверены.

Вопрос: Можно доверять искусственному интеллекту в передаче фактической информации

Ответ ChatGPT-4

Доверять искусственному интеллекту в передаче фактической информации можно, но с определёнными оговорками. В общем, ИИ может быть ценным инструментом для поиска и обработки информации, но всегда полезно использовать его результаты как отправную точку и проверять их с помощью надежных источников.

Новое в ЦЭВМ на июнь 2025 года

ЭВМ Российские ученые разработали технологию (***аналогов нет***) для создания процессоров нового поколения. Она базируется на инновационном методе формирования логических элементов с точностью до 0,2 ангстрема (0,02 нм). Описание разработки опубликовано в ***Science Advances*** («Научные достижения» ***Smirnov et al., Sci. Adv. 11, eads9744 (2025), 7 May 2025***). Технологию запатентовали в России, и в настоящий момент ведется ее патентование в других странах.

Быстродействие ЭВМ резко увеличиться.

Искусственные нейронные сети. В настоящее время модели нейронных сетей базируются на модели 50-годов 20-го века. Перцептрон (1957). Сети MLP.

В апреле 2024 года была представлена новая архитектура нейронных сетей - **сети Колмогорова-Арнольда** (KAN). Эти сети способны выполнять практически все те же задачи, что и обычные нейронные сети, но их работа гораздо более прозрачна.

Веб-технологии: история, развитие и текущее состояние

Зарождение (1980-е – начало 1990-х)

- 1989 год – Тим Бернерс-Ли предложил концепцию Всемирной паутины (World Wide Web)
- 1991 год – создание первого веб-сайта (info.cern.ch)
- 1993 год – появление браузера NCSA Mosaic, сделавшего веб доступным для широкой аудитории
- 1994 год – основание W3C (World Wide Web Consortium) для стандартизации веб-технологий
- 1995 год – появление JavaScript и технологии CSS

Развитие (1995-2005)

- Появление Flash для создания интерактивного контента
- Развитие языков **серверного** программирования (PHP, **ASP**, JSP)
- 1996 год – создание XML
- Бум доткомов (бизнес-модель основывается на работе в сети Интернет) и рост коммерческого использования веб
- 2000-2002 годы – появление концепции веб-сервисов (SOAP, WSDL)

Веб 2.0 (2005-2010)

- Развитие интерактивных приложений и социальных сетей
- AJAX (Asynchronous JavaScript and XML) для создания динамических страниц
- Рост популярности веб-фреймворков (Ruby on Rails, Django)
- Становление REST API как основной архитектуры веб-сервисов
- 2009 год – HTML5 (первые черновики)

Современный веб (2010-настоящее время)

- 2014 год – официальный релиз **HTML5**
- Развитие мобильного веба и **адаптивного** дизайна
- Появление SPA (Single Page Applications) и JavaScript-фреймворков (Angular, React, Vue.js)

- Развитие веб-компонентов и прогрессивных веб-приложений (PWA)
- Web Assembly для высокопроизводительных веб-приложений

Ключевые технологии и их развитие

HTML (Hyper Text Markup Language)

- HTML 1.0 (1993) – базовая разметка
- HTML 4.0 (1997) – поддержка **CSS**, таблиц, форм
- XHTML (2000) – более строгий синтаксис
- HTML5 (2014) – поддержка мультимедиа, canvas, WebSocket, хранилища и т.д.

CSS (Cascading Style Sheets)

- CSS1 (1996) – базовое форматирование
- CSS2 (1998) – позиционирование, z-index, медиа-запросы
- CSS3 (2001-настоящее время) – анимации, гибкая верстка (flexbox, grid), переменные

JavaScript

- 1995 – создание Бренданом Эйхом
- 1997 – стандартизация (ECMAScript 1)
- 2009 – ES5 с важными улучшениями
- 2015 – ES6/ES2015 с классами, модулями, стрелочными функциями
- Ежегодные обновления стандарта с 2015 года

Текущее состояние веб-технологий

Фронтенд (*отображение информации и взаимодействие с пользователем***)**

- **Фреймворки**: React, Angular, Vue.js, Svelte
- **Инструменты сборки**: Webpack, Vite, Rollup
- **CSS-фреймворки**: **Bootstrap**, Tailwind CSS
- **Типизация**: TypeScript
- **Тестирование**: Jest, Cypress, Playwright

Бэкенд

- **Языки и фреймворки**: Node.js, C# (.Net) Python (Django, Flask), Ruby on Rails, PHP (Laravel), Java (Spring), Go

- **API**: REST, GraphQL
- **Микросервисы и серверлесс архитектура**
- **Контейнеризация**: Docker, Kubernetes

Тенденции

- **JAMstack** архитектура (JavaScript, API, Markup)
- **Headless CMS** системы
- **Низкокодовые/бескодовые** платформы
- **Edge Computing**– выполнение кода ближе к пользователю
- **Web Assembly** для высокопроизводительных вычислений
- **Искусственный интеллект** интегрированный в веб-приложения
- **Мета вселенные** и WebXR (AR/VR в браузере)
- **Web3**– децентрализованные приложения на блокчейне

Безопасность и производительность

- HTTPS повсеместно
- Content Security Policy (CSP)
- CORS (Cross-Origin Resource Sharing)
- Core Web Vitals как метрики пользовательского опыта
- HTTP/3 и QUIC протоколы
- **ASP.Net Core Identity**

Веб-технологии прошли огромный путь от простых статических страниц до сложных интерактивных приложений.

Современный веб характеризуется мощными JavaScript-фреймворками, реактивными интерфейсами, облачными архитектурами и постоянно улучшающейся производительностью.

Будущее веб-технологий движется в сторону большей интеграции с нативными (*оригинальный, подходящий, естественный для определённой ситуации*) возможностями устройств, децентрализации и использования искусственного интеллекта для создания более персонализированного и интуитивного пользовательского опыта.

C# и ASP.NETв WEB-технологиях

Фундаментальные основы

ASP.NET как платформа для веб-разработки

- **Эволюция платформы: От классического ASP.NET Framework до современного ASP.NET Core**
- **Архитектурные модели: WebForms (устаревшая), MVC, Web API, Razor Pages, Blazor**
- **Универсальность применения: От простых веб-сайтов до сложных корпоративных порталов и SPA-приложений**

C# как основной язык веб-разработки Microsoft

- **Синергия с платформой: Язык и платформа развиваются параллельно, обеспечивая оптимальную совместимость**
- **Типобезопасность в веб-разработке: Минимизация ошибок времени выполнения**
- **Современные языковые конструкции: Pattern matching, nullable reference types, records улучшают качество веб-кода**

Современные веб-технологии ASP.NET

ASP.NET Core MVC

- **Паттерн Model-View-Controller: Четкое разделение ответственности в веб-приложениях**
- **Маршрутизация запросов: Гибкие возможности определения URL-структуры**
- **Контроллеры и представления: Разделение бизнес-логики и пользовательского интерфейса**
- **Расширяемость: Фильтры, провайдеры моделей, middleware**

ASP.NET Core Web API

- **REST-архитектура: Создание RESTful API с использованием C#**
- **Форматы данных: Поддержка JSON, XML, custom форматов**
- **Документирование API: Интеграция со Swagger/OpenAPI**
- **Версионирование API: Стратегии и подходы к управлению версиями**

Razor Pages

- **Страничная модель: Упрощенный подход к веб-разработке для небольших проектов**
- **PageModel: Объединение модели данных и логики обработки запросов**
- **Формы и валидация: Встроенные механизмы обработки и проверки форм**

Blazor

- **WebAssembly vs Server: Два подхода к выполнению C# в браузере**
- **Компонентная архитектура: Создание интерактивных интерфейсов на C#**
- **Интероперабельность с JavaScript: Взаимодействие с существующими JS-библиотеками**
- **Прогрессивные веб-приложения (PWA): Создание офлайн-доступных веб-приложений**

Инфраструктура и экосистема

Инструментарий разработчика

- **Visual Studio 22: Интегрированная среда разработки с полной поддержкой веб-технологий**
- **Visual Studio Code: Легковесный редактор с расширениями для C# и ASP.NET**
- **JetBrains Rider: Альтернативная IDE с продвинутыми инструментами для ASP.NET**

Управление зависимостями и пакетами

- **NuGet: Центральный репозиторий для .NET библиотек и пакетов**
- **Популярные веб-пакеты: Entity Framework Core, AutoMapper, FluentValidation, MediatR**
- **Клиентские библиотеки: Интеграция с npm, LibMan, yarn для управления фронтенд-зависимостями**

Развертывание и хостинг

- **IIS: Классический веб-сервер для Windows**

- **Kestrel: Кроссплатформенный веб-сервер, встроенный в ASP.NET Core**
- **Docker контейнеры: Упаковка и изоляция веб-приложений**
- **Облачные платформы: Azure App Service, AWS Elastic Beanstalk, Google Cloud Platform**

Интеграция с фронтенд-технологиями

Работа с современными JavaScript-фреймворками

- **Angular +ASP.NET Core: Типизированный подход к полноценным SPA**
- **React и ASP.NET Core: Компонентная модель и серверный рендеринг**
- **Vue.js интеграция: Легковесная альтернатива для интерактивных интерфейсов**

Гибридные подходы

- **MVC + jQuery: Традиционный подход с частичной интерактивностью**
- **Razor + Alpine.js: Легковесная интерактивность без полноценного SPA**
- **Микрофронтенды: Интеграция различных фронтенд-технологий в рамках одного проекта**

Безопасность веб-приложений

Аутентификация и авторизация

- **ASP.NET Core Identity: Встроенный фреймворк для управления пользователями**
- **JWT-токены: Безопасная аутентификация для Web API**
- **OAuth и OpenID Connect: Федеративная аутентификация и внешние провайдеры**
- **Политики авторизации: Гибкая настройка прав доступа**

Защита от веб-уязвимостей

- **CSRF-защита: Встроенные токены и механизмы предотвращения межсайтовой подделки запросов**
- **XSS-защита: Кодирование вывода и Content Security Policy**

- **Защита от SQL-инъекций: Параметризованные запросы и ORM**
- **Secure cookies: Настройка безопасных cookie-файлов**

Производительность и масштабируемость

Оптимизация производительности

- **Кэширование: Различные уровни и стратегии кэширования (Memory Cache, Distributed Cache)**
- **Асинхронность: Эффективное использование async/await для неблокирующих операций**
- **Компрессия и минификация: Уменьшение размера передаваемых данных**
- **Ленивая загрузка: Отложенная загрузка ресурсов по необходимости**

Масштабируемость веб-приложений

- **Горизонтальное масштабирование: Распределение нагрузки между серверами**
- **Балансировка нагрузки: Равномерное распределение запросов**
- **Сессии без привязки к серверу: Работа в кластерной среде**
- **Микросервисная архитектура: Разделение веб-приложения на независимые сервисы**

Тенденции и будущее

Развитие ASP.NET и C# в веб-разработке

- **Расширение возможностей Blazor: Полноценная платформа для SPA на C#**
- **Интеграция с новыми стандартами: HTTP/3, WebSockets, WebRTC**
- **Минимальные API: Упрощенный синтаксис для микросервисов и небольших API**
- **Интеграция с контейнерами и облачными сервисами: Cloud-native подход к веб-разработке**

Сравнение с конкурирующими технологиями

- **ASP.NETvs Node.js: Производительность, экосистема, типизация**
- **ASP.NETvs Java Spring: Корпоративный подход, масштабируемость**
- **ASP.NETvs PHP/Laravel: Простота разработки, производительность**

- **ASP.NETvs Python/Django: Области применения, экосистема**

Практическое применение

Типовые сценарии использования

- **Корпоративные порталы и информационные системы: Высокая надежность и безопасность**
- **API для мобильных приложений: Бэкенд сервисы с высокой производительностью**
- **Электронная коммерция: Надежные и масштабируемые системы для онлайн-продаж**
- **SaaS-решения: Многопользовательские системы с изоляцией данных клиентов**

Успешные проекты и примеры

- **Stack Overflow: Крупный технический портал наASP.NET**
- **Microsoft Azure Portal: Облачная панель управления наASP.NETи Angular**
- **Множество корпоративных веб-приложений: Банки, страховые компании, логистические системы**

Советы по изучению и развитию

Ресурсы для обучения

- **Официальная документация:learn.microsoft.com**
- **Онлайн-курсы: Pluralsight, Udemy, Coursera**
- **Сообщества: Stack Overflow, Reddit r/dotnet, GitHub**
- **Конференции: .NET Conf, Microsoft Build, NDC**

Путь развития веб-разработчикаASP.NET

- **Начальный уровень: Razor Pages, основы MVC**
- **Средний уровень: Web API, Entity Framework, интеграция с JavaScript**
- **Продвинутый уровень: Архитектура, производительность, микросервисы, Blazor**
- **Экспертный уровень: Глубокое понимание внутренних механизмов, контрибьюция в открытый код**