25. What Are Linked Data and Linked Open Data. URL: https://www.ontotext.com/knowledgehub/fundamentals/linked-data-linked-open-data/

Tkachenko O.I., Shyian Ya.A.

MARKETING ACTIVITY MODELING: ONTOLOGICAL APPROACH

The problems of modeling complex systems are considered on the example of modeling marketing activity, in general, and marketing research and marketing management, in particular. Some aspects, stages and principles of modeling marketing activity and its various spheres are analyzed. The main requirements for modeling tools for marketing research and marketing management for solving a wide range of theoretical and practical problems of marketing activity and its individual areas are defined. Modeling based on the ontological approach (analysis and appropriate classification of the elements of such a model) was chosen as the way to solve the problems.

The ontological approach is based on the understanding of ontology as a universal form of knowledge presentation. Ontology is an effective tool for modeling various subject areas, objects, processes and information resources. Ontological modeling of the subject area "Marketing activity" will contribute to the solution of practical and theoretical problems thanks to the consideration of factors affecting different levels of the ontological model and its elements. The main provisions and tools for building a meta-level ontology for defining the conceptual ontological model of the subject area under consideration are given. The toolkit is the OWL ontology description language and the Protégé ontology editor.

In the subject area of the study, the expediency of using ontological modeling as a method capable of reflecting such a complex subject area as "Marketing activity" at the macro level was substantiated. The methodology of ontological modeling of marketing concepts at the macro level was developed. The developed ontological model of the subject area "Marketing activity" can be used to optimize the marketing structure of a specific enterprise (when realizing its capabilities in full and under specific conditions), as well as to evaluate the effectiveness of marketing research and marketing management. The proposed ontological model is a dynamic model in accordance with the principles of the logic of increasing the depth of coverage of related concepts, accounting for dependencies and regularities.

Keywords: *ontology, ontological model, marketing activity, marketing research, marketing management, subject area, OWL, Protégé, ontograph.*

УДК 004. 004.4, 004.9, 004.8 doi.org/10.33298/2226-8553.2023.2.38.34

Ткаченко О.А., Болячевець Я.Ю.

ДЕЯКІ АСПЕКТИ РОЗРОБКИ ТА ВИКОРИСТАННЯ СУЧАСНИХ ВЕБДОДАТКІВ

Використання вебдодатків у різних сферах людської діяльності (економіці, освіті, науці, дозвіллі) обумовило стрімке зростання технологій, методів та засобів розробки вебдодатків, підвищило рівень складності користувацьких інтерфейсів. В статті відзначено, що зі збільшенням інформації, з якою користувач взаємодіє у вебдодатку, збільшилася і кількість

користувачів зі своїми особливостями сприйняття інформації та вимогами щодо комфортності й інтуїтивній зрозумілості інтерфейсів

Вебдодатки використовують комбінацію серверних і клієнтських сценаріїв для виконання завдань користувача. Проаналізовано використання фреймворків та бібліотек для програмування клієнтської частини вебдодатку, зокрема, такі як: Reactjs, Create React App, Babel, Webpack. Програмування серверної частини підтримується, зокрема, Node.js (наприклад, при розробці online-чату, вебсайтів для потокового відео). Розглянуто основні причини поширення вебдодатків, зокрема: залучення споживачів, кросплатформеність, використання централізованих даних, забезпечення безпеки конфіденційної інформації, простота обслуговування, динамічне розширення та оновлення, доступність та підтримка клієнтів (користувачів вебдодатку). Наведено основні положення та інструментарій розробки сучасних вебдодатків. Розглянуто сучасні програмні засоби розробки вебдодатків та моделювання користувацьких інтерфейсів.

Розглянуто класифікацію сучасних вебдодатків, до якої, зокрема, було віднесено: вебсайти, орієнтовані на документи, інтерактивні вебдодатки, транзакційні вебдодатки, вебдодатки (вебсервіси) на основі робочого процесу, спільні вебдодатки, вебдодатки, орієнтовані на портал, універсальні вебдодатки, вебдодатки на основі знань, традиційні вебдодатки, розширені вебдодатки мають динамічний вміст, містять велику кількість інформації, їх легко інтегрувати та прогресивні вебдодатки.

В результаті проведеного аналізу вебдодатків та технологій їх створення було визначено, що вони мають запропоновувати постійну високоякісну продуктивність незалежно від розміру екрана, щільності пікселів і пристрою, який використовується для доступу до програми. Крім того, коли користувач впроваджує сенсорну взаємодію та адаптивний дизайн у процес розробки, то можна отримати зручність, що є важливим, бо власники вебдодатків намагаються досягти того, щоб користувачі були задоволені, а так звана «видимість» можливостей вебдодатку була найвищою.

Ключові слова: *вебдодаток, вебсайт, вебсторінка, HTML, CSS, JavaScript, фреймворк, клієнт-серверна технологія, користувацький інтерфейс, моделювання користувацького інтерфейсу.*

Вступ. Сучасна інформатизація кожного дня отримує активний імпульс до інноваційного розвитку [1 – 5]. Сучасний світ (бізнес, наука, освіта, дозвілля, держава і суспільство) вже потребує нових інструментів і технологій. В наш час майже всі компанії обмінюються інформацією в Інтернеті, а система освіти здійснює процес навчання. На сьогоднішній день вебдодатки (вебпрограми, вебзастосунки) є настільки поширеними, що іноді залишаються непоміченими. Сучасне суспільство використовує вебдодатки в різних областях своєї діяльності.

Вебдодаток дозволяє виконувати завдання через Інтернет. Це програма, яка запускає запити та функціонує через вебтехнології та веббраузери. Вебсайт – сукупність вебсторінок, інформаційного характеру. Вебсайт містить контент (текст, зображення, аудіо- та відеофайли, тощо), видає користувачеві доступні для перегляду готові HTML-сторінки, аутентифікація не є обов'язковою. Такі сайти часто називають статичними. Можна сказати, що вебдодаток – програмне забезпечення, яке працює на основі клієнт-серверної технології.

Вебдодатки (вебзастосунки, вебсайти, веб-орієнтовані системи) стали в наш час невід’ємною частиною нашої діяльності: бізнесу, освіти, дозвілля, тощо. Важливою та актуальною проблемою постає зручність як розробки, так і використання вебдодатків.

Розробники потребують сучасних технологій розробки вебдодатків, зокрема, таких, що забезпечують швидке, ефективне та зручне проектування, моделювання інтерфейсу, тестування вебдодатків.

Користувачі надають перевагу швидким вебдодаткам, які швидко завантажуються та не потребують наявності великих ресурсів від відповідного пристрою (наприклад, комп’ютера,

мобільного телефона). При цьому користувачі звертають увагу на швидкість завантаження вебдодатку та його надійність.

Наприклад, підприємства електронної комерції (Інтернет-магазинів, online-магазинів) можуть дуже швидко втратити як вже наявних, так і не залучити потенційних користувачів своїх вебдодатків (чи відповідних вебсайтів), якщо вони стикаються з проблемою, зокрема, довгого очікування відповіді на свій запит через недостатньо високу швидкість завантаження та функціонування вебдодатку.

Постановка проблеми. Широке впровадження вебдодатків у різні сфери людської діяльності спричинило стрімке зростання технологій, методів та засобів розробки вебдодатків, підвищило рівень складності користувацьких інтерфейсів. Одночасно зі збільшенням інформації, з якою користувач взаємодіє у вебдодатку, збільшилася кількість самих користувачів зі своїми особливостями сприйняття інформації та вимогами щодо зрозумілості, комфортності та інтуїтивності інтерфейсів.

Також зросла кількість апаратних засобів, на яких повинен функціонувати вебдодаток. Зручність використання та користувацький досвід можуть стати вирішальними факторами успіху або невдачі вебдодатків та вебсайтів.

Тому актуальність проблеми дослідження та проведення аналізу сучасних технологій розробки та використання вебдодатків (в тому числі й їхніх інтерфейсів). У результаті гостро постала необхідність моделювання користувацьких інтерфейсів, які б надавали можливість ефективної взаємодії користувача зі складною системою, при цьому не обмежуючи її функціональні можливості.

Аналіз останніх досліджень та публікацій. Розробка та використання різних типів вебдодатків досліджувалася різними вченими, як Сотнік С., Ляшенко В. [1, 2], Ашраф С. [5], Ткаченко О.А. [9]. Зокрема, Башовий, В.М., Стаценко В.В., Стаценко Д.В [7] проводили аналіз швидкодії вебдодатків, розроблених за допомогою різних сучасних фреймворків.

Мета дослідження. Основною метою роботи, що пропонується, є дослідження деяких аспектів розробки (технологій, методів, засобів, підходів, мов програмування) та використання сучасних вебдодатків, моделювання користувацького інтерфейсу, опис класифікації сучасних вебдодатків.

Основний матеріал дослідження. Технології розробки вебдодатків є важливим аспектом подальшого успіху/невдачі використання цих додатків. Вибір технологій розробки має високий рівень значущості особливо для підприємств малого бізнесу чи приватних осіб, що обумовлено, зокрема, невеликим бюджетом, який вони можуть спрямувати на таку розробку. Великі компанії мають більше можливостей ретельно дослідити сучасні технології, звертаючи увагу крім фінансової складової на показники кількості користувачів, що одночасно можуть працювати з їхнім вебдодатком.

У веброзробок є: клієнтська (інтерфейс) та серверна сторони, тобто для будь-якого вебдодатку характерним є наявність клієнта та сервера.

Клієнт (*frontend*) відповідає за інтерфейс та взаємодію з користувачем, в той час як сервер (*backend*) зберігає в собі інформацію на публічних або приватних API та ділиться нею в залежності від запитів користувача. Клієнт відправляє свої запити на сервер та очікує від нього відповіді.

Вебдодатки використовують комбінацію серверних і клієнтських сценаріїв для виконання завдань користувача [27]. З огляду на те, що мільйони компаній використовують Інтернет як свою маркетингову стратегію, вебдодаток має надавати інформацію, зберігати дані, обробляти транзакції та взаємодіяти з потенційними споживачами.

Розробка на стороні сервера включає програму, базу даних і сервер [27]. Сервер відповідає за обробку запитів клієнта, включає в себе бізнес-логіку, зберігання та оновлення різних баз даних. Для структурованих баз даних використовують SQL, MySQL, для слабоструктурованих підійде NoSQL, MongoDB [28, 29], для роботи з кешем Redis [17]. Наведемо деякі основні компоненти інтерфейсу:

– мова розмітки гіпертексту (HTML) та каскадні таблиці стилів (CSS) [18]. HTML повідомляє браузеру, як відображати контент вебсторінок, тоді як CSS надає стилістичне відображення цього контенту [27, 29, 30];

– JavaScript (JS) [19] робить вебсторінки інтерактивними. Існує багато бібліотек JavaScript та фреймворків для швидшої та простішої веброзробки (наприклад, jQuery, React.js, Angularjs, Vue.js [1, 2, 10, 11, 26]).

З серверною стороною взаємодія відбувається за допомогою запитів. Сучасні ферймворки та мови програмування (наприклад, Ruby on Rails (Ruby) [27], Django (Python) [27], Spring (Java) [20], Nestjs (Node.js) [20]) доступні для реалізації серверної частини.

JavaScript був створений, щоб зробити вебсторінки більш динамічними. Програми на JavaScript – скрипти – можуть вбудовуватися в HTML і виконуватися при завантаженні вебсторінки [29].

Сьогодні JavaScript [19] може виконуватися не тільки в браузері, а й на сервері або на будь-якому іншому пристрої [28], який має спеціальну програму, що називається «рушій» JavaScript. У браузерів є власні рушії (так звані «віртуальні машини JavaScript»). В Chrome і Opera – рушій V8, в Firefox – SpiderMonkey.

В наш час з'являються нові стандарти та виклики у створенні вебдодатків. JavaScript є мовою з динамічною типізацію [19, 27], тому Microsoft презентувала нову мову програмування TypeScript [21] над рівнем JavaScript. TypeScript поєднує перевірку типу і статичний аналіз, явні інтерфейси та найкращі практики.

Перегляд основної вебсторінки передбачає виконання наступних кроків (рис.1) [27– 30]:

1. Користувач вводить адресу вебсторінки.
2. Браузер посилає запит до DNS (Domain Name System – система доменних імен) [29].
3. DNS [29] робить запит до Root Server (кореневого серверу) [22].
4. Root Server [27, 29] повертає IP-адресу TLD (Top-Level Domain) Server [23].
5. DNS [29] робить запит до TLD Server.
6. TLD Server [23] повертає IP-адресу Authoritative Name Server [24].
7. DNS робить запит до Authoritative Name Server.
8. Authoritative Name Server повертає IP-адресу сервера вебдодатку.
9. Браузер робить запит до IP-адресу сервера вебдодатку.
10. Сервер вебдодатку повертає контент користувачеві.

Root Server – сервер у ланцюгу DNS відповідає за роботу кореневої зони DNS, тобто відповідні за трансляцію IP-адрес доменів верхнього рівня: а саме UA, COM, ORG і т.д. [27].

TLD Server – сервер найвищого рівня доменних імен (наприклад UA).

Authoritative Name Server – сервер, який повертає IP-адресу вебдодатку, щодо якого був запит користувача.

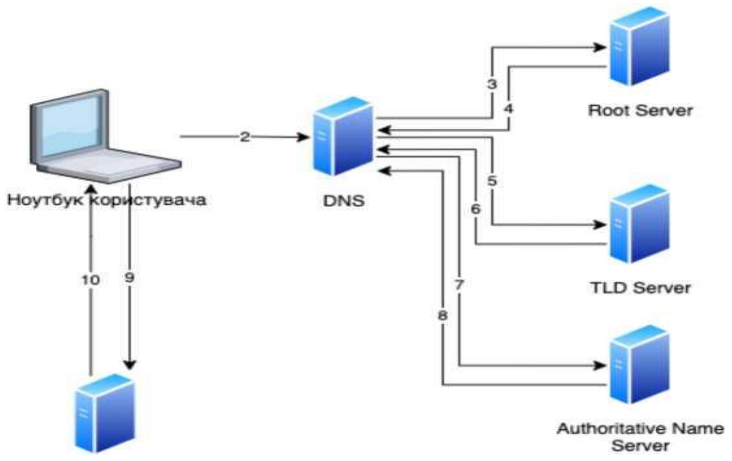

Рисунок 1 – Перегляд вебсайту вебдодатку

Існує багато фреймворків та бібліотек для програмування клієнтської частини вебдодатку, зокрема, такі як:

– React (React.js) [25] – бібліотека JavaScript з відкритим кодом, яка використовується для побудови інтерфейсів (веб та мобільних) [27, 29, 30]. React дозволяє розробникам створювати вебдодатки, які можуть змінювати дані на екрані користувача без перезавантаження вебсторінки. Ця бібліотека легко масштабується, проста у використанні та швидка у дії. Прикладами вебдодатків, побудованими за допомогою React, зокрема, є: facebook.com, dropbox.com, Airbnb.com, Netflix.com, reddit.com, bbc.com.

– Create React App – інструмент, який допомогає створювати вебдодатки, використовуючи React, що не потребує від розробника трудомісткого налаштування. Create React App не обробляє бізнес-логіку, не використовує бази даних, він лише створює шаблон React-додатку, тому розробник може використовувати його з будь-якою серверною базою. Create React app використовує Babel та Webpack [11, 27, 29].

– Babel – набір інструментів, який використовується для перетворення коду ECMAScript 2015+ у сумісну версію JavaScript у старих браузерах або середовищах [12].

– Webpack – конструктор модулів, який працює під час розробки [13, 29]. За допомогою Webpack розробник використовує конфігурацію, щоб налаштувати завантаження конкретних модуліі і файлів.

Програмування серверної частини підтримується, зокрема, Node.js [20] – платформою на основі JavaScript, яка використовується при розробці додатків, наприклад, для online-чату, вебсайтів для потокового відео, що є відкритою і повністю безкоштовною, тому Node.js користуються тисячі розробників. Це робить Node.js кращим вибором, ніж інші платформи на стороні сервера, такі як Java або PHP.

API-бібліотеки Node.js є асинхронними (тобто не блокуються), тому сервер на базі Node.js не чекає, поки API поверне дані. Сервер Node.js реагує неблокуючим способом, що робить його дуже масштабованим на відміну від традиційних серверів, які створюють обмежені потоки для обробки запитів [27, 29, 30].

Основними причинами поширення вебдодатків, зокрема, є:

– *Залучення споживачів*. Для виходу компанії на світовий ринок слід донести всю потрібну інформацію до цільової аудиторії. Сервери вебдодатків діють як платформа для демонстрації продуктів/послуг і залучення споживчої бази компанії.

– *Кросплатформеність*. Сучасні вебдодатки працюють на кількох платформах на відміну від мобільних додатків, розроблених лише для певної платформи.

– *Централізовані дані*. Дані відіграють важливу роль у діяльності підприємства бізнесу, наукової чи освітньої установи, пересічного користувача. Вебдодати мають більш низький рівень обмежень щодо даних, доступних кожному.

– *Безпека*. Вебдодатки, які розгортаються на виділених серверах і контролюються адміністраторами серверів, забезпечують високий рівень безпеки та конфіденційності online-даних, щоб запобігти можливому витоку конфіденційної інформації.

– *Обслуговування.* Оскільки вебдодатки використовують веббраузери, то вони займають мінімальні об’єми пам’яті для зберігання даних. Оновлення розгортаються та працюють у фоновому режимі (при мінімальному втручанні людини).

– *Розширення та оновлення*. Online-дані потребують постійного оновлення (наприклад, для підприємства електронної комерції вебдодаток повинен постійно мати оновленні дані про запаси та нові продукти). У вебдодатках розробникам потрібно додавати лише оновлений новий вміст на серверах замість внесення змін у весь вихідний файл.

– *Доступність*. Ціна створення вебдодатку має бути економічною обгрунтованою та збалансованою. Для вебдодатків електронної комерції перевага online-продажів полягає в тому, що вони доступні в будь-який час, в будь-якій частині світу, в будь-який час.

– *Конкретна перевага*. Компанії досягають конкретної переваги за допомогою вебдодатків. Створюючи налаштований вебдодаток, компанії можуть розробляти індивідуальне програмне забезпечення, операційні системи та рішення для своїх клієнтів.

– *Уникнення обмежень*. Вебдодатки уникають членства в магазинах програмних продуктів і обмежень, дозволяючи компаніям випускати свої версії. Існують різні види хмарних інструментів, які використовуються компаніями для розширення сховища.

– *Підтримка клієнтів (користувачів)*. Вебдодатки розробляються так, щоб служба підтримки клієнтів була доступна в будь-який час і можна було підвищувати лояльність клієнтів (користувачів).

Сучасні користувачі висувають дуже високі вимоги до сучасних вебдодатків, зокрема, на їхній погляд, вебдодатки повинні:

– бути доступними 24/7 у будь-якій частині світу;

– мати можливість бути використаними з будь-якого пристрою з будь-яким розміром екрана та іншими характеристиками;

– забезпечувати безпеку даних користувачів, захищаючи їх від несанкціонованого доступу та кібератак;

– бути гнучкими та масштабованими;

– мати клієнто-орієнтований інтерфейс.

Розглядаючи особливості сучасних вебдодатків, слід виділити такі, як:

– Наявність характеристик, які сприяють вирішенню проблеми будуть чи ні користувачі використовувати той чи інший вебдодаток для вирішення своїх проблем, отримання потрібної інформації, товару чи послуги, і якщо будуть, то як довго вони будуть залишатися на вебсайті таких додатків і користуватися ними потім.

– нативність вебдодатків, коли користувач відразу може знайти потрібну інформацію (отримати відповідь на свій запит).

– кросплатформеність, яка передбачає і використання різних операційних систем, і різних пристроїв, тощо.

Розглянемо класифікацію сучасних вебдодатків та їхні характеристики:

– *Вебсайти, орієнтовані на документи* – статичні html-документи, що зберігаються на ве-сервері та надсилаються безпосередньо клієнту згідно з їхнім запитом. Вебсайти оновлюються користувачами (в цьому випадку – розробниками) власноруч за допомогою відповідних інструментів. Ці програми є статичними, простими, стабільними та потребують небагато часу для формування відповіді на запит користувача. Ціна обслуговування цих програм є високою (особливо при необхідності оновлення). Крім того такі програми мають проблему неузгодженості через статичність та відсутність своєчасного динамічного оновлення інформації.

– *Інтерактивні вебдодатки*, що пропонують CGI, HTML Forms [18]. Вони містять перемикачі, меню вибору, форми тощо. Ці вебдодатки є простими та з високим рівнем швидкодії. У такому додатку відповідні вебсторінки та посилання (гіперпосилання) генеруються згідно з введеними користувачами даними (в тому числі і в їхніх запитах).

– *Транзакційні вебдодатки* мають можливість модифікувати користувачів, вони є більш інтерактивними та підтримують структуровані запити до своїх баз даних, в яких здійснюється обробка даних послідовно та ефективно.

– *Вебдодатки* (вебсервіси) *на основі робочого процесу* здатні передавати робочий процес (точніше інформацію, яка необхідна для функціонування того чи іншого робочого процесу (це можуть бути, наприклад, фінансові, логістичні, фінансові та інші бізнес-процеси) між компаніями, приватними, громадськими або державними органами. Вебсервіси включені для сумісності. Такі вебдодатки потужні, надійні та гнучкі, щоб використовуватися для керування робочим процесом при автономії компаній. Одним з найкращих прикладів таких застосувань є рішення електронної комерції B2B.

– *Спільні вебдодатки*, які, в основному, використовуються як групові програмні продукти, де важливим є групове спілкування. Прикладами таких вебдодатків, зокрема, є: чати, online-форуми, вебсайти online-навчання або вебсайти (наприклад, Вікіпедія), де користувачі обмінюється різноманітною інформацією з можливістю її подальшого використання та редагування.

– *Вебдодатки, орієнтовані на портал*, в яких єдина точка доступу існує для розділення різних джерел інформації та послуг. Прикладами таких вебдодатків, зокрема, є: пошукові системи, портали різних спільнот (наприклад, навчальних, за інтересами, професійних, релігійних тощо).

– *Універсальні вебдодатки* надають персоналізовані засоби для будь-якого пристрою з будь-якого місця в будь-який час. Такі вебдодатки мають обмежені можливості взаємодії та підтримують обмежену кількість пристроїв. Для динамічного налаштування цих вебдодатків потрібно мати попередні знання щодо контексту, де конкретний вебдодаток має використовуватися. Прикладом таких вебдодатків є додатки надання послуг на основі місцезнаходження користувача.

– *Вебдодатки на основі знань* використовуються для надання знань як користувачеві, так і відповідному пристрою (персональному комп'ютеру, мобільному телефону, тощо).. Управління знаннями базується на семантичних вебтехнологіях (прикладами таких вебдодатків семантичний web, зв'язування і повторне використання знань).

Традиційні вебдодатки були менш складними, мали, зокрема:

– статичний вміст;
– обмежені сфери використання;
– обмежену функціональність та інтерактивність;
– забезпечення лише мінімального рівня безпеки;
– недостатні можливостей своєчасного оновлення.

Розширені вебдодатки мають динамічний вміст, містять велику кількість інформації, їх легко інтегрувати, розкладати та планувати, підтримують більш ефективне забезпечення безпеки даних [11].

Прогресивні вебдодатки:

– працюють на будь-якій платформі: Windows, macOS, Linux;
– працюють в усіх основних браузерах;
– сумісні зі стільниковими та мобільними пристроями з усіма розмірами екрана;
– автоматично масштабують візуальні елементи.

Прогресивні вебдодатки підтримують: функцію push-сповіщення, роботу в автономному режимі, нативні програми, створені за допомогою SDK для iOS і Android.

Прогресивні вебдодатки дозволяють також використовувати HTML5, CSS разом із фреймворками JavaScript чи Typescript [20] для створення продукту з нуля. Angular і React [25] є одними з найпопулярніших інтерфейсних технологій, які використовуються для створення PWA (Progressive Web Application, прогресивний вебдодаток). Нові версії Angular забезпечують вбудовану підтримку PWA. React є основою для React Native, що дозволяє легко переносити додатки, створені за допомогою React, у нативні додатки. Використання JavaScript для серверної частини не є обов'язковим, можна використовувати будь-який технологічний стек.

Прогресивні вебдодатки є одним з типів вебсайту або вебсторінки. Тому при завантаженні прогресивного вебдодатку як окремого програмного продукту для iPhone або Android через Apple не обов'язкові App Store або Google Play. Замість того, щоб завантажувати його з App Store, користувач може просто завантажити вебдодаток у свій веббраузер (Google Chrome, Safari Opera тощо).

Використання мов програмування при розробці будь-якого типу вебдодатку залежить від функціоналу конкретного вебдодатку та вимог до нього. Найбільш популярною мовою для стартапів на даний момент є JavaScript, поширені також Python, Java, PHP та інші.

JavaScript – мова програмування, яка відповідає за логіку та динаміку вебдодатку, таку як анімація, валідація, взаємодія з формами та з користувачем, в цілому.

Важливою для сучасного додатку є концепція асинхронних запитів AJAX в JavaScript, в результаті чого користувач отримує тільки необхідні дані в форматі JSON [19] або XML [19] та підставляє їх на відповідне місце сторінки. Вебдодатки, що мають одну сторінку та оновлюються відповідно до отриманих даних, називаються SPA (*Single-Page Application*).

При збільшенні проєкту росте і кількість станів інтерфейсу. Для роботи з даними використовують такі бібліотеки керування станом як MobX, Redux, VueX [5, 10, 11]. Вони спрощують оновлення даних та стандартизують цей процес, в результаті зменшується кількість помилок та пришвидшується розробка.

Зі зростанням вебдодатку та збільшенням кількості даних стає актуальним питання підвищення відмовостійкості, тому розумним є використання розподілених баз даних (*Distributed Databases*) які використовують декілька дата-центрів і забезпечують безвідмовну роботу вебдодатку

Сучасні вебдодатки великих розмірів використовують мікросервісну архітектуру, де кожен сервіс відповідає за свою функцію, що спрощує та пришвидшує розгортання і розробку проєкту, підвищує відмовостійкість його [12].

Для створення мікросервісів на Java використовується популярна технологія Spring Boot, для JavaScript підійде Node.js, також використовують Docker для створення обгорток мікросервісів. Щоб забезпечити контроль доступу, зручність та безпечность доступу до різних мікросервісів активно використовують API Gateway.

Компанія Amazon – власник великого, конкурентного, найбільш успішного та технологічного вебдодатку [14].

Вебдодаток, зображений на рис. 2 здебільшого є Single-Page Application, майже всі дії відбуваються на одній HTML-сторінці та динамічно оновлюються без необхідності перезавантаження сторінки. Це робить додаток швидким і зручним для користувача. В деяких випадках зустрічається MPA (*Multi-Page Application*), але це пов’язано з віддаленням від основної області діяльності вебдодатку.

На головній сторінці вебдодатку використано багато компонентів React-бібліотеки, що створює та підтримує динамічний інтерфейс і дає можливість перевикористовувати (повторно використовувати) деякий функціонал в інших місцях вебдодатку, що спрощує та пришвидшує його розробку, а також зменшує кількість можливих помилок.

Рисунок 2 – Головна сторінка Amazon
Джерело: [14]

Рисунок 3 – Архітектура хмарної технології Amazon Web Services
Джерело: [14]

В якості баз даних Amazon використовує власну розробку Amazon Web Services (AWS). Розташування серверів відбувається в різних регіонах світу, що забезпечує швидший доступ до інформації та підвищену відмовостійкість всього вебдодатку. AWS працює на основі розподіленої архітектури і надає велике коло послуг для користувачів (рис 3) у всьому світі, відповідає за інфраструктуру, обробку даних платежів та іншого.

Вебдодаток (рис. 2) використовує мікросервісну архітектуру, на головній сторінці зустрічаються такі сервіси як авторизації користувача, обробки замовлень, пошуку товарів, корзини, рекомендації, сортування та інші. Програмні продукти Amazon не мають відкритого коду, який є комерційною таємницею компанії, тому важко більш детально проаналізувати використання сучасних технологій використаних при розробці їх програмних продуктів

Поширення вебдодатків та різке збільшення кількості їхніх користувачів загострило проблему розробки адекватного запитам користувачів інтерфейсу. Одним з ефективних підходів до розробки користувацьких інтерфейсів є їх моделювання, яке може бути орієнтовано, зокрема, на використання дизайну (usage-centered design, UCD) [15, 16].

Usage-centered design – використання моделей для проєктування системи, яка підтримує всі завдання, що необхідно виконати користувачам [16]. Цей процес легко масштабується і використовується в проєктах з невеликими ресурсами. Дизайн засновано на таких зв’язаних моделях, як: роль, завдання та контент.

Модель ролі відображає головні характеристики ролей, які грають користувачі по відношенню до вебдодатку, тобто які типи відношень користувачі можуть мати з вебдодатком. Серед можливих аспектів цих відношень можна виділити, зокрема, мету, частоту взаємодії, обсяг і спрямованість інформаційного обміну тощо.

Модель завдання представляє структуру роботи, яку користувачі повинні виконати в результаті взаємодії з вебдодатком.

Модель контенту представляє вміст і організацію користувацького інтерфейсу для підтримки визначених завдань. Перелік завдань формують випадки використання (use cases) – послідовності дій, які включають в себе можливі помилки та взаємодіють з додатком.

Після визначення ролей користувачів майбутнього інтерфейсу та формування випадків використання здійснюється побудова мапи випадків використання (task cases map), яка відображає зв’язки між реальними прикладами завдань та представляє собою діаграму, що показує всі випадки завдання та їх взаємозв’язки.

Розробка варіантів використання завершується побудовою абстрактних прототипів, які дозволяють розробникам проєктувати загальну організацію та архітектуру користувацького інтерфейсу вебдодатку без візуалізації її компонентів або деталізації макета. Абстрактні прототипи складаються з моделі контенту інтерфейсу, що описує вміст контекстів, у яких користувач взаємодіє з вебдодатком, та містить контекстну навігаційну мапу, яка показує, як користувачі переходять від одного контексту до іншого під час виконання задач (task cases).

Використання абстрактних прототипів має ряд переваг, зокрема [16]:

– вони надають повну картину загальної архітектури інтерфейсу;
– абстрактний характер моделей дозволяє відкласти прийняття детальних рішень;
– використання абстрактного прототипу як керівництво до побудови остаточного візуального дизайну забезпечує більш інноваційний користувацький інтерфейс

Системні актори – це програмні та апаратні системні компоненти вебдодатку, з якими повинен взаємодіяти інтерфейс, вони відокремлені від акторів, що мають ролі користувачів.

За таким підходом кожна сторінка, форма чи інший контекст взаємодії відповідає абстрактному прототипу, що підтримує взаємопов’язані випадки задач.

Компоненти користувацького інтерфейсу (кнопки, посилання, таблиці, екрани тощо) походять з абстрактних компонентів, які реалізують конкретні кроки в межах підтримуваних випадків завдання. Варіанти завдань, в свою чергу, підтримують виконання ролей, які користувачі можуть виконувати щодо вебдодатку. Такий підхід до проєктування

користувацького інтерфейсу гарантує, що кінцевий варіант інтерфейсу повністю покриває усі можливі варіанти взаємодії користувача з вебдодатком та надає можливість внесення змін у прототип з мінімальними витратами часу.

Висновки. В результаті проведеного аналізу сучасних вебдодатків та технологій їх створення було визначено, що вебдодатки мають запропоновувати постійну високоякісну продуктивність незалежно від розміру екрана, щільності пікселів і пристрою, який використовується для доступу до програми.

Крім того, коли користувач впроваджує сенсорну взаємодію та адаптивний дизайн у процес розробки, то можна отримати зручність, що є важливим, бо власники вебдодатків намагаються досягти того, щоб користувачі були задоволені, а так звана «видимість» можливостей вебдодатку була найвищою.

Таким чином, у статті проведено аналіз особливостей розробки вебдодатків, розглянуто характеристики вебдодатків та їхня класифікація (типізація).

Розглянуто сучасні програмні засоби розробки вебдодатків та моделювання користувацьких інтерфейсів.

В статті наведено основні причини поширення використання вебдодатків в різних сферах життєдіяльності людини, суспільства, держави.

ЛИТЕРАТУРА

1. Al-Sherrawi M., Saadoon M., Sotnik S., Lyashenko V. Information model of plastic products formation process duration by injection molding method. //International Journal of Mechanical Engineering and Technology, 2018. Vol. 9(3). P. 357-366.

2. Sotnik S., Nevliudova V., Malaya I. Developing the information search system for selecting the moulds forming elements. //Innovative technologies and scientific solutions for industries, 2017. 2(2). P. 86-92.

3. Sotnik S., Belova N., Lyashenko V., Mohammad A. Informational and Structural-Parametric Models of Inductions Micromotors. //Journal of Electrical and Electronics Engineering (IOSR-JEEE), 2018. P. 66-76.

4. Deineko Z., Sotnik S., Lyashenko V. Confidentiality of Information when Using QR-Coding. // IJAISR, 2022. Vol. 6. Issue 9. P. 10-15.

5. Ashraf S. Avoiding Vulnerabilities and Attacks with a Proactive Strategy for Web Applications. //Advances in Robotics and Mechanical Engineering, 2021. Vol. Issue 2.

6. Jazayeri M., Mesnage C.S., Rose J., Jazayeri M. Modern Web Application Development //Emerging Methods, Technologies, and Process Management in Software Engineering. John Wiley and Sons Inc., 2007. P. 131-147.

7. Башовий, В.М., Стаценко В.В., Стаценко Д.В. Визначення швидкості роботи сучасних фреймворків для створення web-інтерфейсів //Технології та інжиніринг, 2022. №4(9). С. 9 -16.

8. Gohan, N.V., Gohan V.V., Afanasieva I.V. Black and white-box unit testing for web applications // Вісник Національного технічного університету «ХПІ». Серія: Системний аналіз, управління та інформаційні технології, 2022.. № 1 (7). С. 79-83.

9. Ткаченко О.А., Ткаченко О.І. Деякі аспекти ситуаційно-семантичного моделювання складних об’єктів, процесів та систем //Водний транспорт, 2017. Вип.№ 1 (26). С.129-133.

10. Brandon, D.M. (ed.). Software Engineering for Modern Web Applications: Methodologies and Technologies: Methodologies and Technologies. IGI Global, 2008. 402 p.

11. Kleppmann M. Designing Data-Intensive Applications. New York: O’Reilly Media, 2012. 336 p.

12. Most popular technologies. Stackoverflow. URL: https://insights.stackoverflow.com/survey/2021#most-popular-technologies-language

13. Simpson K. Up & Going. New York: O’Reilly Media, 2017. 642 p.

14. Amazon. URL: https://www.amazon.com/

15. Anderson J., Fleek F. Integrating Usability Techniques into Software Development. //IEEE Software, 2017. 18 (1). P. 168-182.

16. Cloyd M.H. Designing User-Centered Web Applications in Web Time. //IEEE Software, 2017. 18 (1). P. 94-105.

17. Comparing Database Management Systems: MySQL, PostgreSQL, MSSQL Server, MongoDB, Elasticsearch, and others. URL: https://www.altexsoft.com/blog/business/comparing-database-management-systems-mysql-postgresql-mssql-server-mongodb-elasticsearch-and-others/

18. HTML For Beginners The Easy Way: Start Learning HTML & CSS Today. URL: https://html.com

19. JavaScript reference. URL: https://devdocs.io/javascript/

20. Introduction to Node.js. URL: https://nodejs.dev/en/learn/

21. What is TypeScript? URL: https://www.typescriptlang.org

22. Root Servers. URL: https://www.iana.org/domains/root/servers

23. What is a TLD Server? URL: https://threat.media/definition/what-is-a-tld-server/

24. What is Authoritative DNS server? URL: https://www.cloudns.net/blog/authoritative-dns-server/

25. React. The library for web and native user interfaces. URL: https://react.dev

26. The Progressive JavaScript Framework. URL: https://vuejs.org

27. Гайтан О.М., Бочкарь В.О. Методи моделювання користувацького інтерфейсу веб-додатків. URL: https://reposit.nupp.edu.ua/bitstream/PoltNTU/10492/1/vol1-365-368.pdf

28. Грінько К.О. Дослідження методів обробки природної мови для створення карти D&D. https://openarchive.nure.ua/bitstream/document/18894/1/2021_M_PI_Grinko_KO.pdf.

29. Шепелєв І.Н. Інформаційна система оцінювання швидкості завантаження веб-додатків. URL: https://essuir.sumdu.edu.ua/bitstream-download/123456789/82567/1/Shepeliev_mag_rob.pdf;jsessionid=FC8C43B60B82159FACB1F82CDA7FAEE4

30. Розробка веб-додатків. URL: https://openarchive.nure.ua/server/api/core/bitstreams/689a8d92-140e-4825-b47f- 3b07b085929d/content

REFERENCES

1. Al-Sherrawi M., Saadoon M., Sotnik S., Lyashenko V. (2018). Information model of plastic products formation process duration by injection molding method. //International Journal of Mechanical Engineering and Technology. Vol. 9(3). P. 357-366.

2. Sotnik S., Nevliudova V., Malaya I. (2017). Developing the information search system for selecting the moulds forming elements. //Innovative technologies and scientific solutions for industries. 2(2). P. 86-92.

3. Sotnik S., Belova N., Lyashenko V., Mohammad A. (2018). Informational and Structural-Parametric Models of Inductions Micromotors. //Journal of Electrical and Electronics Engineering (IOSR-JEEE). P. 66-76.

4. Deineko Z., Sotnik S., Lyashenko V. (2022). Confidentiality of Information when Using QR-Coding. // IJAISR. 2022. Vol. 6. Issue 9. P. 10-15.

5. Ashraf S. (2021). Avoiding Vulnerabilities and Attacks with a Proactive Strategy for Web Applications. //Advances in Robotics and Mechanical Engineering. Vol. Issue 2.

6. Jazayeri M., Mesnage C.S., Rose J., Jazayeri M. (2007). Modern Web Application Development //Emerging Methods, Technologies, and Process Management in Software Engineering. 8(3). P. 131-147.

7. Bashovy, V.M., Statsenko V.V., Statsenko D.V. (2022). Vyznachennya shvydkosti roboty suchasnykh freymvorkiv dlya stvorennya web-interfeysiv [Determining the speed of operation of modern frameworks for creating web interfaces]. // Tekhnolohiyi ta inzhynirynh [Technologies and engineering]. № 4 (9). P. 9-16. (in Ukranian).

8. Gohan, N.V., Gohan V.V., Afanasieva I.V. (2022). Black and white-box unit testing for web

applications // Visnyk Natsional′noho tekhnichnoho universytetu «KHPI». Seriya: Systemnyy analiz, upravlinnya ta informatsiyni tekhnolohiyi. [Bulletin of the National Technical University "KhPI". Series: System analysis, management and information technologies]. № 1 (7). P. 79 - 83.

9. Tkachenko O.A., Tkachenko O.I. (2017). Deyaki aspekty sytuatsiyno-semantychnoho modelyuvannya skladnykh ob″yektiv, protsesiv ta system [Some aspects of situational-semantic modeling of complex objects, processes and systems] // Vodnyy transport [Water transport]. Vyp. № 1 (26). P.129-133. (in Ukranian).

10. Brandon, D.M. (ed.). (2008). Software Engineering for Modern Web Applications: Methodologies and Technologies: Methodologies and Technologies. IGI Global. 402 p.

11. Kleppmann M. (2012). Designing Data-Intensive Applications. New York: O'Reilly Media. 336 p.

12. Most popular technologies. Stackoverflow. URL: https://insights. stackoverflow.com/survey/2021#most-popular-technologies-language

13. Simpson K. (2017). Up & Going. New York: O'Reilly Media. 642 p.

14. Amazon. URL: https://www.amazon.com/

15. Anderson J., Fleek F. (2017). Integrating Usability Techniques into Software Development. //IEEE Software. 18 (1). P. 168-182.

16. Cloyd M.H. (2017). Designing User-Centered Web Applications in Web Time. //IEEE Software. 18 (1). P. 94-105.

17. Comparing Database Management Systems: MySQL, PostgreSQL, MSSQL Server, MongoDB, Elasticsearch, and others. URL: https://www.altexsoft.com/blog/business/comparing-database-management-systems-mysql-postgresql-mssql-server-mongodb-elasticsearch-and-others/

18. HTML For Beginners The Easy Way: Start Learning HTML & CSS Today. URL: https://html.com

19. JavaScript reference. URL: https://devdocs.io/javascript/

20. Introduction to Node.js. URL: https://nodejs.dev/en/learn/

21. What is TypeScript? URL: https://www.typescriptlang.org

22. Root Servers. URL: https://www.iana.org/domains/root/servers

23. What is a TLD Server? URL: https://threat.media/definition/what-is-a-tld-server/

24. What is Authoritative DNS server? URL: https://www.cloudns.net/blog/authoritative-dns-server/

25. React. The library for web and native user interfaces. URL: https://react.dev

26. The Progressive JavaScript Framework. URL: https://vuejs.org

27. Haytan O.M., Bochkar V.O. Metody modelyuvannya korystuvats′koho interfeysu veb-dodatkiv. [Methods of modeling the user interface of web applications]. URL: https://reposit.nupp.edu.ua/bitstream/PoltNTU/10492/1/vol1-365-368.pdf (in Ukranian).

28. Grinko K.O. Doslidzhennya metodiv obrobky pryrodnoyi movy dlya stvorennya karty D&D. [Study of natural language processing methods for creating a map D&D]. URL: https://openarchive.nure.ua/bitstream/document/18894/1/2021_M_PI_Grinko_KO.pdf. (in Ukranian).

29. Shepelev I.N. Informatsiyna systema otsinyuvannya shvydkosti zavantazhennya veb-dodatkiv [Information system for evaluating the speed of loading web applications]. URL: https://essuir.sumdu.edu.ua/bitstream-download/123456789/82567/1/Shepeliev_mag_ rob.pdf;jsessionid=FC8C43B60B82159FACB1F82CDA7FAEE4. (in Ukranian).

30. Rozrobka vebdodatkiv [Development of web applications]. URL: https://openarchive.nure.ua/server/api/core/bitstreams/689a8d92-140e-4825-b47f-3b07b085929d/ content. (in Ukranian).

Tkachenko O.A., Boliachevets Ya.Yu.

SOME ASPECTS OF THE DEVELOPMENT AND USE OF MODERN WEB APPLICATIONS

The use of web applications in various spheres of human activity (economy, education, science, leisure) led to the rapid growth of technologies, methods and means of developing web applications,

and increased the level of complexity of user interfaces. The article noted that with the increase in information with which the user interacts in the web application, the number of users with their own peculiarities of information perception and requirements for comfort and intuitiveness of interfaces also increased.

Web applications use a combination of server-side and client-side scripting to perform user tasks. The use of frameworks and libraries for programming the client part of the web application was analyzed, in particular, such as: Reactjs, Create React App, Babel, Webpack. Server-side programming is supported, in particular, by Node.js (for example, when developing online chat, websites for streaming video). The main reasons for the spread of web applications are considered, in particular: attracting consumers, cross-platform, use of centralized data, ensuring the security of confidential information, ease of maintenance, dynamic expansion and updating, availability and support of customers (users of the web application). The main provisions and tools for the development of modern web applications are presented. Modern software tools for developing web applications and modeling user interfaces are considered.

The classification of modern web applications was considered, which, in particular, included: document-oriented websites, interactive web applications, transactional web applications, workflow-based web applications (web services), shared web applications, portal-oriented web applications, universal web applications, knowledge-based web applications , traditional web applications, rich web applications that have dynamic content, contain a large amount of information, are easy to integrate, and progressive web applications.

As a result of the analysis of web applications and their creation technologies, it was determined that they should offer consistent high-quality performance regardless of screen size, pixel density and the device used to access the application. In addition, when the user introduces touch interaction and responsive design into the development process, convenience can be obtained, which is important because web application owners try to achieve user satisfaction and the so-called "visibility" of the web application's capabilities is the highest.

***Keywords**: web application, website, web page, HTML, CSS, JavaScript, framework, client-server technology, user interface, user interface modeling.*