“Implementation of multithreaded calculations in educational web applications”

AUTHORS	Viktor Molchanov https://orcid.org/0000-0002-8003-2402
ARTICLE INFO	Viktor Molchanov (2019). Implementation of multithreaded calculations in educational web applications. *Development Management*, *17*(2), 1-7. doi:10.21511/dm.17(2).2019.01
DOI	http://dx.doi.org/10.21511/dm.17(2).2019.01
RELEASED ON	Friday, 26 July 2019
RECEIVED ON	Friday, 29 March 2019
ACCEPTED ON	Thursday, 06 June 2019
LICENSE	This work is licensed under a Creative Commons Attribution 4.0 International License
JOURNAL	"Development Management"
ISSN PRINT	2413-9610
ISSN ONLINE	2663-2365
FOUNDER	Simon Kuznets Kharkiv National University of Economics

NUMBER OF REFERENCES **15**

NUMBER OF FIGURES **2**

NUMBER OF TABLES **0**

Viktor Molchanov (Ukraine)

IMPLEMENTATION OF MULTITHREADED CALCULATIONS IN EDUCATIONAL WEB APPLICATIONS

Abstract

The complication of the logic of educational web applications raises the issue of the effectiveness of the organization of their implementation. At the same time, efficiency, including pedagogical one, is connected among other factors with the technology of implementation of programs.
When using as the main browser program, it is necessary to take into account its features, in particular, one-flow mode of execution of programs (scripts). Implementation of more complex algorithms in web applications delays the response of the application interface to user actions. This creates a discomfort for the user and, as a result, reduces the effectiveness of his work.
Expanding the range of devices from which users access the Internet leads to the fact that mobile devices are more and more often used for learning as well. Therefore, another side of the problem is the impact on the quality of connection to the server. It is necessary to ensure the work of the program in case of interruptions in connection or reduce their impact.
A solution to the problem may be the implementation of part of the calculations in the background. The article deals with the use of calculations in the background streams of the browser and caching control for educational web applications. Various ways of creating such streams and the peculiarities of their use are analyzed.

Keywords WEB-application training, WEB-portal, computing flow, browser API, Dedicated Workers, Shared Workers, Service Workers

JEL Classification C88

В.П. Молчанов (Україна)

РЕАЛІЗАЦІЯ БАГАТОПОТОЧНИХ ОБЧИСЛЕНЬ В НАВЧАЛЬНИХ WEB-ДОДАТКАХ

Анотація

Ускладнення логіки навчальних WEB-додатків робить актуальною проблему ефективності організації їх виконання. При цьому ефективність в тому числі і педагогічна пов'язана серед інших чинників з технологією реалізації програм.
При використанні в якості основної програми браузера доводиться враховувати його особливості, зокрема однопотоковий режим виконання програм (скриптів). Реалізація в WEB-додатках більш складних алгоритмів вносить затримки в реакцію інтерфейсу додатку на дії користувача. Це створює для користувача певний дискомфорт і, як наслідок, знижується ефективність його роботи.
Розширення номенклатури пристроїв, з яких користувачі виходять в Інтернет, призводить до того, що і для навчання все частіше застосовуються мобільні пристрої. Тому ще однією стороною проблеми є вплив на ефективність якості зв'язку з сервером. Потрібно забезпечити роботу програми при перервах в зв'язку або зменшити їх вплив.
Рішенням проблеми може стати виконання частини обчислень у фоновому режимі. У статті розглядаються питання використання обчислень в фонових потоках браузера і управління кешуванням для навчальних WEB-додатків. Аналізуються різні способи створення таких потоків і особливості їх використання.

Ключові слова навчальний WEB-додаток, WEB-портал, потік обчислень, API браузера, Dedicated Workers, Shared Workers, Service Workers

Класифікація JEL C88

S. KUZNETS KHNUE

Founder

Simon Kuznets Kharkiv National University of Economics, Nauky avenue, 9-A, Kharkiv, 61166, Ukraine
http://www.hneu.edu.ua/

Received on: 29th of March, 2019
Accepted on: 06th of June, 2019

Viktor Molchanov, Candidate of Technical Sciences, Associate Professor, Simon Kuznets Kharkiv National University of Economics, Ukraine

http://dx.doi.org/10.21511/dm.17(2).2019.01

ВСТУП

Навчальні системи у самих різних формах все більше використовують сервіси мережі Інтернет. Електронна пошта, пересилання файлів, WWW і різні навчальні засоби на їх основі тісно інтегруються з різноманітними ресурсами: соціальними мережами, хмарними сервісами, сховищами даних, потокової трансляції аудіо і відео-контенту. Основним фактором, що визначає таку тенденцію, є все більша поширеність і доступність мережі Інтернет, розвиток її технологій.

Прикладом створення навчального середовища на базі інтернет-технологій може служити корпоративний освітній портал [14]. Корпоративний освітній портал - це інтегрований ресурс, створений для надання можливості колективної роботи користувачам в єдиному інформаційному просторі та накопичення створюваних ними даних в структурованому вигляді. На порталі вони можуть навчатися на електронних курсах, спілкуватися з колегами, брати участь у проведених корпоративних заходах, дізнаватися про новини та події. Освітній портал надає своїм користувачам різноманітні види внутрішньої комунікації (форуми, опитування, голосування, конкурси, механізми електронного документообігу тощо).

З точки зору розробки і технологій створення портал - це WEB-додаток, який забезпечує користувачам єдину точку доступу до інформаційних ресурсів і функцій. Як і будь-який WEB-додаток, він має певну архітектуру і створюється з використанням існуючих технологій. На даний момент - це дворівнева архітектура клієнт-сервер з різними варіантами розподілу функцій: з тонким клієнтом, товстим клієнтом і тому подібне [2]. Сама взаємодія може бути синхронною або асинхронною.

Оскільки основною клієнтською програмою навчальних WEB-додатків є браузер, то усі функції на стороні клієнта реалізуються або в скриптах, або в апплетах. І частіше за все у скриптах.

1. ЛІТЕРАТУРНИЙ ОГЛЯД

Основне завдання скриптів клієнтської частини розподіленого додатка полягає в забезпеченні інтерфейсу з користувачем. Однак особливості окремих додатків [11] призводять до того, що виявляється ефективним ускладнення логіки клієнтської частини програми. Обмеженням тут служать можливості клієнта. При використанні в якості клієнта браузера це особливості та обмежені можливості використовуваної мови (JavaScript) і технологічних засобів розробки. До основних обмежень можна віднести, в першу чергу, його однопоточний режим. У тому випадку, якщо браузер виконує обчислення, які пов'язані з логікою програми, інші дії стають неможливими до їх завершення. Таким чином можуть виникати затримки в реакції на дії користувача і інші затримки, які знижують ефективність використання додатку.

При тому ефективність процесу навчання оцінюється з урахуванням педагогічного аспекту. Ці питання досить широко висвітлені в літературі як з точки зору особливостей створення засобів у цілому [3; 7], так і відповідного інтерфейсу [4; 15]. Розглянуто також питання впливу їх використання на ефективність навчання [8; 9; 13]. При розгляді передбачається, що методики і педагогічні сценарії можуть бути реалізовані в повній мірі [1; 6]. Технічної реалізації та режимам використання, які можуть вплинути на ефективність процесу навчання набагато менші [10].

Частково ця проблема вирішується за рахунок організації асинхронних запитів до сервера, але обробка відповідей також може бути досить складною. Іноді пропонують розбиття завдань на частини за таймером, але такий підхід виглядає досить штучним.

Ще однією стороною проблеми є ускладнення інтегрованого середовища навчання, що призводить до ускладнення налаштувань і процесу відновлення стану середовища після перерв у роботі. При цьому причиною переривання роботи можуть бути як дії користувача (пауза в роботі), так і порушення зв'язку з сервером. Внаслідок важливим є надання навчальному додатку властивості, що позначається

http://dx.doi.org/10.21511/dm.17(2).2019.01

в літературі як Progressive Web Apps (PWA). Якщо програма є PWA, то переривання зв'язку з сервером для неї є некритичним за рахунок збереження даних в кеші [11; 12]. Однак, декларативні рішення [12] на основі API Application Cache, визнані негнучкими і недостатньо ефективними. Рішенням може бути використання окремого незалежного потоку обчислень, що забезпечує використання локальної пам'яті.

Таким чином, при створенні навчального середовища на базі WEB-технологій виникає проблема ефективності організації виконання скриптів в браузері. Її проявами є затримки у завантаженні ресурсів, "підвисання" інтерфейсу і втрата даних і стану середовища при проблемах в з'єднанні з сервером.

2. МЕТА ДОСЛІДЖЕННЯ

Рішенням сформульованої проблеми може стати створення декількох асинхронних потоків для реалізації клієнтської частини функціоналу на основі нових API HTML5, які реалізовані в більшості сучасних браузерів, включаючи їх мобільні версії. Тому метою дослідження був вибір засобів, що дозволяють підвищити комфортність роботи користувача в середовищі, визначення ступеня впливу різних способів створення потоків на функціонування WEB-додатків, визначення ефекту від використання фонових потоків при організації клієнтської частини WEB-додатків і вироблення рекомендацій щодо організації багатопоточних обчислень в клієнтської частини WEB-додатків на основі API браузерів.

3. МЕТОДИ ДОСЛІДЖЕННЯ

Проведений аналіз функціонування корпоративних навчальних порталів показує, що основний вплив на роботу користувача надають затримки в роботі, викликані сповільненою реакцією середовища на дії користувача та складність у відновленні стану навчального середовища після перерви у роботі. Користувачеві, який працює з таким ресурсом, доводиться не тільки терпіти затримки різної тривалості, а й витрачати час на налаштування і відновлення середовища після втрат зв'язку з сервером. Крім того, виникає питання про збереження даних, якщо такі відправлялися на сервер. Наприклад, результати виконання завдань.

Головною причиною затримок є ускладнення логіки клієнтської частини, яка веде до істотного завантаження процесора, однопотоковий режим виконання скриптів в браузері та порушення зв'язку з сервером. У той же час, наявність в більшості випадків багатопроцесорної (багатоядерної) архітектури в сучасних комп'ютерах створює передумови для прискорення обчислень за рахунок їх розпаралелювання. Однак скрипти, за допомогою яких реалізується функціонал клієнтської частини WEB-додатки, виконуються браузером в одному потоці.

В якості вирішення сформульованих проблем при реалізації логіки додатка на клієнтській стороні запропоновано розглянути використання нових технологічних можливостей в рамках стандарту HTML 5. Стандарт передбачає цілу низку нових API, велика частина яких вже реалізована в останніх версіях браузерів.

Серед нових технологічних можливостей браузерів найбільш підходящими для цього є API Workers. Загальна ідея полягає в запуску скриптів в окремих потоках з окремим простором імен. Залежно від способу зв'язку з основним потоком, набору подій і можливостей є три варіанти запуску (три типи): виділені (Dedicated Workers), колективні (Shared Workers) і сервісні (Service Workers). Перші два типи схожі з точки зору можливостей, їх призначення - виконання фонових обчислень, управляються вони з основного потоку. А третій тип спеціально орієнтований на виконання незалежно від потоку, що його запускає . Сторінка закриється, а процес залишиться. Для кожного типу є відповідний об'єкт. Код для виконання поміщається в окремому файлі. Оскільки пріоритет основного потоку вище, ніж у того, що запускається, то ці потоки можна вважати фоновими.

Всі питання, пов'язані з синхронізацією взаємодії, ведення черги запитів до ресурсу (фонового потоку), вирішуються у браузері і не відображаються на основних потоках. Розподіл ресурсів процесора між потоками також організовується браузером. Це може призводити до неоднакових результатів у різних браузерах. І в цілому можна очікувати, що ефект від фонових обчислень буде залежати від запасу продуктивності (завантаження процесора), числа ядер і інших характеристик процесора.

Крім того, потоки різних типів мають свої особливості і по різному підходять для вирішення сформульованих проблем.

Виділені потоки запускаються і управляються з основного потоку сторінки браузера. Методи і властивості представлені об'єктом var worker = new Worker ('task.js'). Взаємодія між основним і запущеним потоками забезпечується шляхом обміну повідомленнями. При цьому використовується подія message і метод worker.postMessage(). Передаються рядки або об'єкти JSON. Запущений потік може бути зупинений як всередині по команді self.close(), так і з потоку сторінки, з якої він був запущений, worker.terminate(). У тексті скрипта, що виконується в запущеному потоці, self і this посилаються на власний глобальний простір імен.

Схема взаємодії потоків наведена на Рисунку 1.

Кожна сторінка може запускати кілька виділених потоків для виконання різних завдань асинхронно у фоновому режимі. При цьому основний потік залишається більш пріоритетним, завдяки чому не відчуваються затримки у взаємодії з користувачем. Слід зазначити, що з метою безпеки для скриптів, що виконуються у фоновому режимі, введений ряд обмежень. Зокрема, неможливо використання об'єктів DOM, недоступні об'єкти window і document. Передача даних здійснюється лише копіюванням, неможливо передавати по посиланню. Доступними залишаються об'єкти navigator, location, такі об'єкти JavaScript як Object, Array, Date, Math, String тощо. Також доступний XML Http Request, методи setTimeout () і setInterval (), кеш додатка.

Цей вид потоку можна вважати базовим, з його допомогою на клієнті забезпечується виконання досить складних обчислень без затримок призначеного для користувача інтерфейсу. В цьому потоці можливо виконання обчислень, які пов'язані з рендерингом (прорисовкою) тривимірних зображень, шифруванням даних, перевіркою правопису, підказками при введенні даних. У такий потік можна поміщати і асинхронні запити до сервера, якщо відповіді вимагають складної обробки.

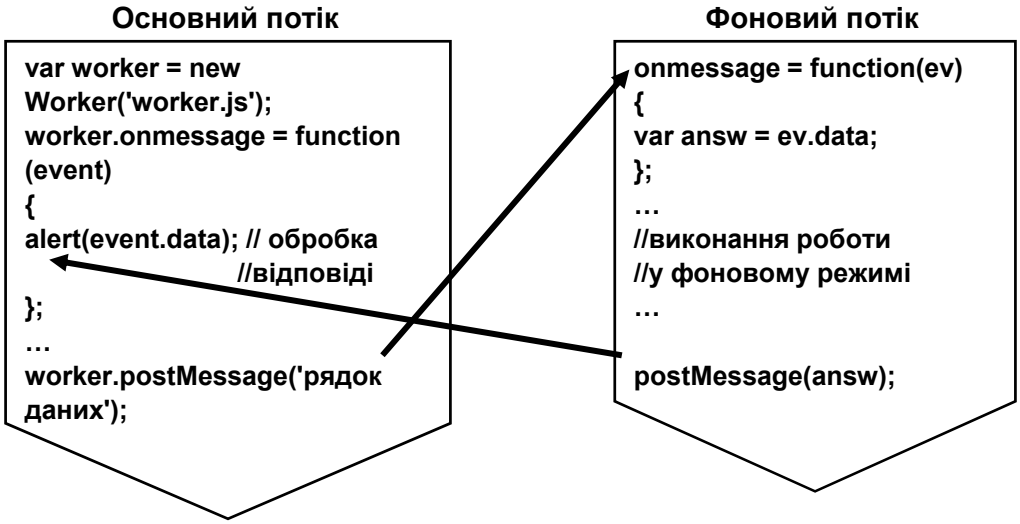

Рисунок 1. Схема взаємодії виділених потоків

http://dx.doi.org/10.21511/dm.17(2).2019.01

Спільні потоки схожі на виділені за способом управління та і за завданнями, які вирішуються з їх допомогою. Відмінність полягає в можливості їх використання з декількох сторінок (з різних вікон, вкладок, фреймів), так як вони доступні з декількох контекстів браузера. Однак ці сторінки повинні мати однакове джерело (протокол, домен, порт). Саме наявність порту для зв'язку і відрізняє їх використання. Для запуску та управління використовується об'єкт, який створюється конструктором Shared Worker(): var Worker = new Shared Worker ("worker.js"). Порт запускається за командою Worker.port.start(). Обмін повідомленнями проводиться за допомогою об'єкта MessagePort у властивості Shared Worker.port. Для обміну повідомленнями служить метод port.postMessage() і подія port.onmessage. Сам код зберігається в єдиному екземплярі і розділяється (тобто використовується по черзі основними потоками). До тих пір, поки виконання програмного коду не закінчитися для одного основного потоку, інші стоять у черзі. Причому їх виконання не зупиняється.

Таким чином основна відмінність між цими типами полягає в тому, що код для виконання в фоновому потоці не дублюється, якщо потрібно виконати одні і ті ж дії. Проте при декількох зверненнях виникають затримки.

Схема взаємодії потоків для цього варіанта наведена на Рисунку 2.

Service Worker - це програмний код, що виконується в окремому потоці браузера, який має відносну самостійність, він не прив'язуються до конкретної сторінці і продовжує виконуватися після закриття сторінки, з якої було здійснено запуск (сторінки джерела). Для управління використовуються об'єкт service Worker і ряд подій (install, activate, fetch). Процес запуску передбачає кілька етапів: реєстрація і активація. Запущений таким чином потік може взаємодіяти з усіма сторінками домену, в якому розташований файл з кодом. Визначається це в момент реєстрації. Взаємодія складається в обміні повідомленнями та перехопленні звернень за ресурсами за подією fetch. Завдяки цьому, з'являється можливість контролю і корекції запитів і переходів сторінок. Service Worker контролюватиме одну або кілька сторінок. Після запуску і реєстрації він має можливість обробляти події, пов'язані зі зверненнями сторінок до сервера. Таким чином можна використовувати ресурси, збережені в кеші, без звернення до сервера.

Слід зазначити, що цей тип потоків з самого початку орієнтований на контроль запитів додатка з метою надати йому властивості PWA. Можна говорити про певний фільтр на взаємодію сервера і клієнта в рамках функціонування ресурсу. Незважаючи на те, що в цілому додаток управляється користувачем, можна коригувати його поведінку і, наприклад, забезпечити використання даних з кеша і збереження їх в кеші до появи з'єднання з сервером.

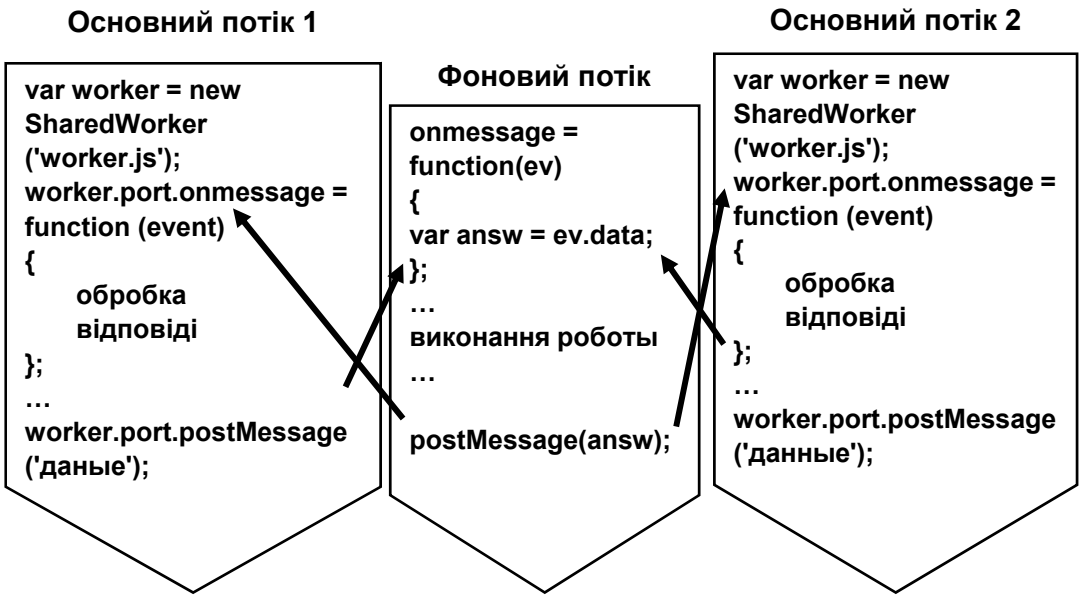

Рисунок 2. Взаємодія потоків при спільному фоновому потоці

4. РЕЗУЛЬТАТИ

Для цілей дослідження були розроблені сторінки з клієнтськими скриптами, що містять фонові обчислення, організовані з використанням різних засобів. Оцінювалися загальні затримки обчислень і ступінь впливу на інтерфейс користувача.

Порівнювалось використання виділених та спільних потоків одночасно кількома сторінками (власний файл для кожної сторінки і загальний файл для декількох сторінок). Оцінювався ефект від створення декількох потоків для однієї сторінки.

У всіх випадках затримки в реакції інтерфейсу користувача відсутні. Затримки в обчисленнях, які виконуються в фонових потоках залежать від завантаження процесора і числа ядер. При числі потоків більшому, ніж число ядер, створюються черги, обслуговування яких створює видимість паралельного виконання, але за більший час. Для виділених потоків обслуговування завжди послідовне (спочатку один потік до закінчення, а лише потім - інший).

Для сервісних потоків перевірялася можливість створення додатків з властивостями PWA. Кешування файлів та їх подальше використання виконується без внесення істотних затримок в завантаження і реакцію інтерфейсу. Більш складні моделі коригування поведінки програми не перевірялися.

ВИСНОВКИ

Розглянуте застосування API Workers дозволяє зробити затримки інтерфейсу, пов'язані з однопоточним режимом браузера, практично непомітними для користувача.

Загальна продуктивність з урахуванням розміщення обчислень в фонових потоках в порівнянні з їх виконанням в одному потоці збільшується тільки для SharedWorker, поки їх число не перевищує числа вільних ядер процесора. В інших випадках затримки залежать від ступеня завантаження ресурсів.

Спостерігаються помітні відмінності в затримках обчислень з використанням фонових потоків в різних браузерах. Це є наслідком особливостей планування та обслуговуванні потоків і їх розпаралелювання при виконанні.

СПИСОК ЛІТЕРАТУРИ

1. Abramov, O. M. (2012). Про становлення, розвиток та взаємозв'язок стандартів та специфікацій електронного навчання (e-learning) [Pro stanovlennia, rozvytok ta vzaiemozviazok standartiv ta spetsyfikatsii elektronnoho navchannia (e-learning)]. *Visnyk Kharkivskoi derzhavnoi akademii kultury, 37*, 284-293. Retrieved from https://bitly.su/NOL3Sw1G
2. Anatolyev, A. G. (2016). *Компоненты сетевого приложения. Клиент-серверное взаимодействие и роли серверов [Komponenty setevogo prilozheniya. Kliyent-servernoye vzaimodeystviye i roli serverov]*. Retrieved from http://www.4stud.info/networking/lecture5.html
3. Berezovskij, V. S., Stecenko, I. V., & Zavadskij, I. O. (2013). *Створення електронних навчальних ресурсів та онлайнове навчання [Stvorennia elektronnykh navchalnykh resursiv ta onlainove navchannia]* (176 p.). Kyiv: Vydavnycha hrupa BHV.
4. Bulanova, T. V., Starodubcev, V. A., & Shamina, O. B. (2012). Педагогический дизайн информационной учебной среды [Pedagogicheskiy dizayn otkrytoy uchebnoy sredy]. *Problemy informatiki, 5*(17), 208-212. Retrieved from http://www.problem-info.sscc.ru/2012-5/34.pdf
5. Demianenko, M. (2017). *Что такое Progressive Web Apps и какие возможности они открывают для вашего бизнеса [Chto takoye Progressive Web Apps i kakiye vozmozhnosti oni otkryvayut dlya vashego biznesa]*. Retrieved from https://netpeak.net/ru/blog/chto-takoe-progressive-web-apps-i-kakie-vozmozhnosti-oni-otkryvayut-dlya-vashego-biznesa/
6. Gricenko, V. I. (2004). *Дистанционное обучение: теория и практика [Distantsionnoye obucheniye: teoriya i praktika]* (375 p.). Kiev: Naukova Dumka.
7. Klarin, M. V. (1997). *Инновации в обучении. Метафоры и модели [Innovatsii v obuchenii. Metafory i modeli]* (223 p.). Moskva: Nauka.
8. Kuklev, V. A. (2009). *Электронное обучение с помощью мобильных устройств в любое время и в любом месте [Elektronnoye obucheniye s pomoshch'yu mobil'nykh ustroystv v lyuboye vremya i v lyubom meste]* (356 p.). Ulyanovsk: UlGTU.
9. Kurdickaya, O. S. (2007). Оценка эффективности обучения в высокотехнологичных информационных компаниях [Otsenka effektivnosti obucheniya v vysokotekhnologichnykh informatsionnykh kompaniyakh]. *Teoriia i praktyka suchasnoi ekonomiky,* 298-300.

http://dx.doi.org/10.21511/dm.17(2).2019.01

10. Molchanov, V. P. (2016). Анализ реализации новых WEB-стандартов в массовом программном обеспечении [Analiz realizatsii novykh WEB-standartov v massovom programmnom predstavlenii]. *Systemy obrobky informatsii, 4*(141), 226-228. Retrieved from http://nbuv.gov.ua/UJRN/soi_2016_4_44
11. Molchanov, V. P. (2017). Архитектура процессов клиентской части WEB-приложений [Arkhitektura protsessov kliyentskoy chasti WEB-prilozheniy]. *Systemy obrobky informatsii, 2*(148), 229-232. Retrieved from http://nbuv.gov.ua/UJRN/soi_2017_2_44
12. Molchanov, V. P. (2018). Підвищення ефективності навчальних WEB-додатків за рахунок кешування [Pidvyshchennia efektyvnosti navchalnykh WEB-dodatkiv za rakhunok keshuvannia]. *ScienceRise, 3*(44), 31-33. Retrieved from http://journals.uran.ua/sciencerise/article/view/127118
13. Nazarova, O. L. (2003). Новые информационные технологии в управлении качеством образовательного процесса в колледже [Novyye informatsionnyye tekhnologii v upravlenii kachestvom obrazovateľnogo protsessa v kolledzhe]. *Informatika i obrazovaniye, 11*, 79-84.
14. Ponomarenko, V. S., Pushkar, O. I., & Andriushchenko, T. U. (2017*). Педагогічний дизайн засобів електронного навчання на робочому місці [Pedahohichnyi dyzain zasobiv elektronnoho navchannia na robochomu mistsi]* (263 p.). Kharkiv: KhNEU.
15. Pushkar O., & Lepeyko T. (2006). Design of interactive visual tools in the computer multimedia education program (by the example of management disciplines) Yeditepe university. *4tn International Symposium of Interactive Media Design, 30*, 117-125.