УДК004.9

д.т.н., проф. **Шинкарук О.М.**
(Державний університет внутрішніх справ, Львів)
к.т.н., с.н.с. **Мірошніченко О.В.** (ВІКНУ)
к.т.н., доц. **Яшина О.М.** (ХмНУ)

DOI: https://doi.org/10.17721/2519-481X/2021/70-10

ДО ПИТАННЯ УПРАВЛІННЯ ЯКІСТЮ ПРОГРАМНИХ ВЕБ-СИСТЕМ ЗАСОБАМИ РОЗРОБКИ

У даній статті наведено результати досліджень існуючих засобів і визначення теоретичних аспектів застосування сучасних фреймворків для розробки програмного забезпечення, а також обґрунтування доцільності та впливу варіативності вибору на якість розроблюваних програмних додатків. Основними результатами дослідження є обгрунтування вибору фреймворків для розробки програмного забезпечення. Потрібно відмітити, що не зважаючи на велику кількість фреймворків, структурований аналіз здійснений досить незначною кількістю дослідників, що і обумовлює наукову новизну даного питання. Вибір відповідного фреймворка є однією з основних засад, що визначає якість майбутнього сайту. Кожну з технологій, що наведені у даній статті можна вважати вичерпною у всіх відношеннях тому, що їх використовують, як у практиці перетворення в графічний інтерфейс (frontend), так і для розробки варіанту архітектури програмного забезпечення (backend). Тому, слід обирати платформу у відповідності до зручностей щодо мов програмування.

Отже, Agile найбільше підходить для проектів із «відкритим кінцем», наприклад, запуск інтернет-сервісів, розробка комп'ютерних ігор, операційних систем. Однак, гнучкість може приводити до втрати фокусу та втрати передбачуваності. Дуже важливо відокремлювати помилки застосування гнучкого підходу від недоліків самої методології. Перш, ніж будуть реалізовані всі переваги підходу, потрібно буде деякий час на адаптацію до реалій конкретних завдань. Практична значущість полягає у можливості застосування тих чи інших фрейморків відповідно до потреб замовника та розробника для досягнення певних результатів при розробці програмного забезпечення.

Ключові слова. Фрейморк, CSS, програмне забезпечення, програмний продукт, мова програмування, управління якістю.

Вступ. На сучасному етапі розвитку засобів програмування в галузі веб-розробки кількість веб-фреймворків (ВФК) різко зросла. Всі вони володіють певною низкою переваг та недоліків, а тому актуальним є питання вибору правильного та відповідного фреймворку або їх набору для кожного конкретного проекту, адже від правильного вибору залежить якість, час реалізації та надійність кожного розроблюваного програмного продукту.

Взагалі кажучи, веб-фреймворки однозначно змінили та покращили своєю появою можливості програмування і стали невід'ємною частиною процесу розробки. Існує велика кількість інформації, що стосується ВФК, але часто ця інформація містить тільки визначення та складні терміни, які не дають ніякого уявлення про дані засоби. Саме тому питання обґрунтування варіативного вибору фреймворків та їх вплив на якість програмного забезпечення є актуальним завданням на сучасному етапі розвитку програмування.

Аналіз останніх досліджень і публікацій. Методологічною основою даного дослідження є праці таких дослідників як Jacek Schae, МейертД.О., Масляк Т.А., Колесникова Т.А., Агалаков С.А., Лаврищева Е.М., Петрухин Е.М., Куликов С.С., Сем Канер, Джек Фолк, Енг Нгуен [1-4] та багато інших. Деякими загальними аспектами використання фреймворків займались Татур Ю.Л., Білоконна К.В. Використання *CSS*-фреймворків у своїх роботах описують Масляк Т.А., Колесникова Т.А.Загалом. Разом із тим, в теперішній час, виконується велика кількість різноманітних досліджень, присвячених застосуванню тих чи інших ВФК та їх доцільності щодо розробки певного програмного продукту. Однак, потрібно відмітити, що проблема варіативності вибору веб-фреймворків при створенні програмних продуктів займає незначне місце у вітчизняних та зарубіжних роботах.

Не зважаючи на те, що дослідження в галузі застосування фреймворків постійно проводяться, наукових робіт, присвячених варіативності вибору та доцільності їх використання в тій чи іншій сфері розробки та впровадження програмного забезпечення недостатньо, що і обумовило виявлення до даного завдання підвищеного наукового та практичного інтересу.

Мета дослідження. Визначення теоретичних аспектів застосування сучасних фреймворків для розробки програмного забезпечення, а також обґрунтування доцільності та впливу варіативності вибору на якість розроблюваних програмних додатків.

Виклад матеріалу дослідження. З урахуванням того, що веб-фреймворк є інструментом полегшення написання і запуску веб-додатків, розробнику не потрібно самостійно прописувати велику кількість коду та витрачати час на пошук потенційних прорахунків і помилок [3,5,6]. В сучасних умовах існує можливість виброру з наявних мов для веб-програмування відповідного фреймворку, як для статичних, так і для динамічних сторінок. Так, в залежності від поставленого завдання, необхідним є вибір одного ВФК, що здатний задовільнити всі потреби, або поєднання декількох.

Слід приймати до уваги, що у ВФК є дві основні функції: робота на серверній стороні (бекенд) і робота на клієнтській стороні (фронтенд). При цьому, фронтенд-фреймворки пов'язані із зовнішньою частиною програми, тобто відповідають за інтерфейс, а бекенд-фреймворки відповідають за внутрішню чатину додатку, тобто логікудодатку.

Як показує практика, серверні веб-фреймворки (СВФК) не дають можливості створення веб-додатків з насиченим інтерфейсом. Вони обмежені в своїй функціональності, однак забезпечують створення простих сторінок і різних форм. Також, СВФК здатні забезпечити формуваня вихідних даних і відповідати за безпеку в разі атак. Все це, безумовно, може спростити процес розробки. Серверні фреймворки, в основному, відповідають за окремі, але критично важливі частини програми, без яких неможливе стале функціонування. Найбільш поширеними фреймворками такого типу і мови програмування, з якими вони працюють, слід виділити наступні:

- *Django – Python;*
- *Zend – PHP;*
- *Express.js – JavaScript;*
- *Ruby on Rails – Ruby.*

На відміну від серверних, клієнтські веб-фреймворки (КВФК) ніяк не пов'язані з логікою програми. Цей тип фреймворків працює в браузері. З їх допомогою можна поліпшити і впровадити нові користувальницькі інтерфейси. Фронтенд-фреймворки дозволяють створювати різні анімації і односторінкові додатки. Всі клієнтські фреймворки відрізняються по функціональності і використанню. Найбільш поширеними фреймворками такого типу слід виділити наступні:

- *Backbone+Marionette*;
- *Angular*;
- *Ember.js;*
- *Vue.js.*

Всі ці фреймворки використовують мову програмування - *JavaScript*.

Разом з типами фреймворків, що приведені вище, набули широкого використання для вирішення окремих завдань програмування й багатофункціональні фреймворки. *Meteor* відомий як фул-стек веб-фреймворк. Це означає, що він здатний задовільнити майже всі потреби як з боку клієнта, так і з боку сервера, що робить *Meteor* надзвичайно популярним. Його використання забезпечує заощадження часу, наприклад, на налагодження взаємодії між двома ВФК через *RESTAPI*. Використання, при цьому, ВФК *Meteor* забезпечить виконання відповідних процедур і прискорить процес розробки. Оскільки обидві сторони – серверна та клієнтська – працюють на одній мові, то існує можливість створення і використання для них одного коду. Одна з особливостей даного ВФК – «режим реального часу» – внесення змін в

одному інтерфейсі, обумовлює зміни і в інших. Наприклад, при додаванні коментарів або зміні змісту документу або таблиці із загальним доступом, інші користувачі це також будуть бачити.

Окрему увагу слід приділити й розмірам (масштабам) фреймворків. Розрізняють ВФК, що здатні забезпечити рішення для всіх завдань, і більш легкі варіанти, які спеціалізуються на вирішенні конкретних завдань – такі фреймворки називаються мікрофреймворками. Останні не здатні забезпечити вирішення всіх завдань, проте іноді краще розкласти функціональність на кілька підходів (фреймворки, мікрофреймворки, бібліотеки). При цьому, слід зауважити, що розширення функціональності мікрофреймворків можливе за допомогою сторонніх додатків і створенням невеликих проектів на їх основі, або шляхом поєднання мікрофреймворку з основним «великим» фреймворком.

Наприклад, якщо додаток заснований на ВФК *Django* і потрібні веб-сокети, то є доцільним використання мікрофреймворку *aiohttp*. Або, якщо додаток не дуже великий і потрібна лише проста маршрутизація *URL* і шаблони з нескладним контекстом, то можливе використання ВФК *Flask* замість *Django*.

Незважаючи на те, що існуючі ВФК відрізняються один від одного, їїх вибір може бути складним завданям, тому, слід навести кілька ознак, які є загальними для них усіх. Це архітектура і особливості, які важливі так, як і функції [7-9].

Архітектура всіх ВФК заснована на декомпозиції декількох окремих прошарків (додатки, модулі і інш.), що означає, можливість розширення функціональності в залежності від потреб і використання зміненої версії разом з кодом ВФК або використання сторонніх додатків. Така гнучкість вважається ще однією ключовою перевагою фреймворків. Існує безліч додатків з відкритим вихідним кодом (*open source*), спільнот і комерційних організацій, які створюють програми або їх розширюють для популярних фреймворків, наприклад, *Django REST Framework, ng-bootstrap* і інш.

Таким чином, зміст кожного веб-фреймворка формується трома складовими: модель, представлення, контролер (model, view,controller), скорочено MVC, що подано на рис. 1.

Рисунок 1 – Зміст веб-фреймворка

Модель містить всі дані і рівні бізнес-логіки, її правила і функції.

Представлення відповідає за візуальне відображення даних, наприклад, діаграми, графіки і т.д.

Контролер просто трансформує дані для команд попередніх двох складових.

Вони невіддільні один від одного, тому важливо як слід у всьому розібратися, щоб уникнути помилок під час роботи програми.

Важливого значення набувають особливості фреймворків, основні з них подані на рис. 2, які роблять їх багатофункціональними і зручними на практиці.

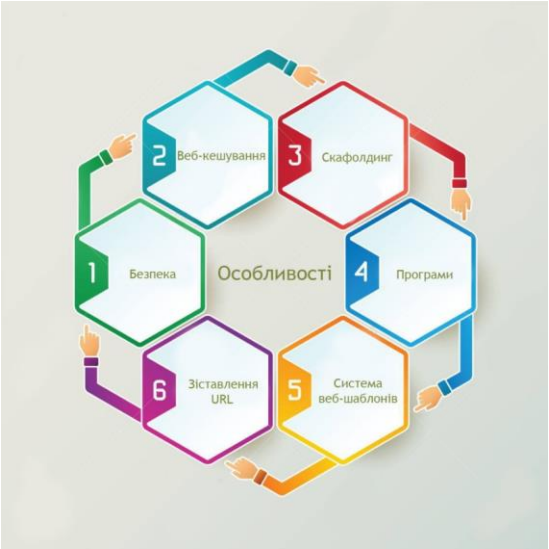

Рисунок 2 – Особливості веб-фреймворків

Веб-кешування забезпечує збереження різних документів і дозволяє уникнути перевантаження сервера. Тому наявна можливість його використання в різних системах за певних умов. Також веб-кешування працює на серверній стороні. Наприклад, існують кешовані веб-сторінки на сторінці пошукової видачі *Google*.

Скафолдинг – це ще одна технологія, що полягає в автоматичному генеруванні типових частин програми або струкутри проекту. Це дозволяє істотно збільшити швидкість розробки і стандартизує кодову базу.

Система веб-шаблонів являє собою набір різних методологій і програмного забезпечення, реалізованих для створення і розгортання веб-сторінок. Для обробки веб-шаблонів використовуються шаблонізатори, що вважаються інструментом фреймворку, який відповідає за веб-публікацію.

Зіставлення *URL* – це функція фреймворків, що сприяє спрощенню доступу до сторінок веб – сайту, індексаціїйого пошуковими рушіями, створюючи привабливу назву.

Велика кількість типів веб-додатків підтримуються веб-фреймворками, що застосовуються, в основному, для створення таких додатків, як блоги, форуми, *CMS* і інші [9-11].

Наведена інформація щодо функціональності властива всім ВФК. Разом із тим, широкий асортимент фреймворків, як правило, ускладнює їх вибір.Тому, для полегшення розробки варто використовувати критерії, що обумовлюють кращий інструмент. Наприклад, ВФК *Angular*на мові програмування *JavaScript*– це фреймворк від *Google*, розроблений спеціально для створення динамічних веб-додатків, в точу числі інтерфейсних додатків без необхідності застосовування інших плагінів або фреймворків. Його структура утримуєнизку цікавих функцій. Зокрема, використовуючи шаблони, відкривається можливість відображення інформації з моделі і контролера. При цьому, ВФК *Angular* підтримує архітектуру *MVC*, що відкриває можливість розділення додатку на *MVC* компоненти. При цьому, все інше управляється засобами фреймворка.

Всі ці функції є частиною платформи, яка сприяє розробці ефективного веб-сайту. Ось лише деякі приклади майданчиків, розроблених за допомогою цього фреймворка: *Netflix, Freelancer.com, GoodFIlms* і інш.

Фреймворк *RubyonRails* створений мовою програмування *Ruby*. Однією з його переваг вважається докорінне спрощення і прискорення розробки веб-додатків за рахунок повторного використання коду. Такий підхід дозволяє додавати деякі додаткові функції. Серед популярних веб-сайтів, написаних на *Ruby on Rails*, можна виділити *Basecamp, Ask.fm, GitHub, 500px*і інш. Але, основною перевагою цього фреймворку слід вважати швидку розробку з

меншою кількістю написаного коду та помилок. До інших його переваг доцільно віднести гнучкість, що полягає в можливості широкого кола застосування (наприклад, від управління проектами до будівництва), швидкість, що полягає в здатності скорочення часу розробки веб-додатків приблизно на *30-40%*,і можливість вносення змін в код. Саме тому фреймворк *RoR* ідеально підходить для довготривалих проектів.

Фреймворк *YII* є платформою з відкритим вихідним кодом, що вбудована в *PHP5*. Завдяки зрозумілому дизайну і зручному інтерфейсу, цей фреймворк забезпечує надзвичайно просту і швидку веб-розробку. Він оптимізований по продуктивності, що дозволяє його використання для будь-яких проектів. Крім того, дана платформа утримує інструменти, які володіють здатністю налагодження і тестування додатку. Цікавою особливістю платформи слід вважати можливість увімкнення класів та об'єктів, лише, за потреби, що прискорює завантаження додатків. Дані функції сприяють створенню високопродуктивної структури, що сприяє розробці ефективних веб-сайтів (наприклад, *TACC, Craftcms, Hum Hub* і інш.).

Фреймворк *Meteor JS* розроблений на платформі *Node.js* дозволяє створювати різні *real-time* веб-додатки. Одна з основних особливостей *Meteor JS* — належна основа для створення простих сайтів особистого користування.

Meteor JS це ізоморфний *JavaScript* веб-фреймворк з відкритим вихідним кодом, що сприяє швидкому завантаженню сторінок. Крім того, завдяки інтегрованому стеку *JavaScript*, який простягається від бази даних кінцевого користувача до екрану, існує можливість виконання в *10* рядках коду те, що, зазвичай, розтягується на *1000* рядків.

Одна з особливостей даного фреймворку полягає в можливості використання одного коду при розробці під операційні системи *iOS, web, Android* або *desktop*. Також існує можливість використання різних популярних фреймворків та інструментів для створення функцій.

Фреймворк *Express.js* на платформі *Node.js* покриває цілий ряд важливих функцій плагінами, тому доцільним вважається використання для швидкої розробки веб-додатоку і прикладного програ́много інтерфе́йсу(*API*). Також даний фреймворк придатний до використання для створення мобільних додатків.

По суті, *Express.js* складається з *Angular* і бази даних *MongoDB*. Це означає, що для розробки веб-сайтів достатньо знання таких мов, як: *HTML, CSS* і *JavaScript*. А використовуючи модулі *npm*, відкривається можливість розширення функціоналу додатків. Також *Express.js* ідеальний для створення простих веб-сервісів.

Zend — це *opensource* фреймворк, що розроблений на мові *PHP*. Він орієнтований на розробку сучасних, надійних і безпечних веб-сервісів. При цьому застосовуються різні професійні пакети *PHP*, які роблять розробку веб-сайтів найвищого класу значно простіше і швидше. Крім того, ВФК використовує архітектуру *MVC*, що відокремлює базу даних і бізнес-логіку від представницького рівня. Це сприяє отриманню більш зрозумілого і чистого коду. Веб-фреймворк *Zend* базуєтьсяна концепціях об'єктно-орієнтованого програмування, що відкриває можливість розширення різних компонент фреймворку. Також слід відзначити маршрутизацію, яка виконує свою роботу бездоганно, і функції кеша.

Django- найпопулярніший фреймворк для веб-розробки, написаний на мові програмування *Python* і використовує архітектуру *MVC*. Ключовими особливостями цієї платформи слід вважати наступні.

Швидкість. Головна мета фреймворка-допомогти розробникам зробити додаток, як можна швидше. Причому, на всіх етапах розробки – від ідеї до релізу. Ефективність і економічність – саме так можна сформулювати девіз *Django*.

Безпека. Як правило, деякі безпекові помилки, що пов'язані з «ін'єкціями» мови програмування структурованих запитів *SQL*, а також підробкою міжсайтових запитів і міжсайтовим скриптингом. *Django* ефективно управляє наявними іменами користувача і паролями, а система аутентифікації користувача, як відомо, відіграє вирішальну роль.

Масштабованість. Більшість бізнес-сайтів використовують ВФК *Django* для швидкого задоволення потреб трафіку.

Багатофункціональність. Фреймворк включає в себе додаткові опції для допомоги з картами сайту, аутентифікацією користувачів, адмініструванням контенту, *RSS*-канали і багато іншого. Кожна з них надає істотну допомогу в процес веб-розробки.

ВФК *Laravel* вважається одним з кращих *PHP* фреймворків. Він здатний забезпечити легку розробку, як веб, так і мобільних додатків для невеликих сайтів і великого бізнесу. При цьому, ВФК *Laravel* відрізняється наявністю багатьох цікавих функцій, таких як, наприклад, техніка авторизації, об'єктно-орієнтовані бібліотеки, підтримка *MVC*, міграції баз даних, міжсайтові запити і інш.

До основних переваг цього фреймворку можна віднести:

–Можливість збільшення трафіку сайту, що обумовлює застосування до будь-якого браузера та пристрою.

–Гнучкість. Фреймворк має модульну структуру, що допомагає спростити сам веб-сайт і процес його розробки.

–*PHP* не потребує спеціальних способів обслуговування, що пов'язано з автоматичним завантаженням об'єкта, який міститься в програмному забезпеченні.

–Фреймворк *Laravel* може створювати унікальні *URL*-адреси, так як реалізує різні маршрути з однією і тією ж назвою.

Як показує досвід, кожен фреймворк володіє низкою переваг і недоліків, що проявляються при вирішенні певних задач. При виборі фреймворку необхідно враховувати декілька чинників, а саме:

1) можливості фреймворку повинні чітко відповідати завданням проекту. Зайвий функціонал не потрібен, але при цьому ніколи не повинно виникати обмежень при доопрацюванні проекту;

2) фреймворк повинен сприяти зменшенню часурозробки сайту;

3) для зручності використання фреймворку іншим розробником він повинен бути поширеним;

4) безпека фреймворка-один з найважливіших його показників.

У випадках прийняття рішення щодо вибору фреймворку, зазвичай порівнюється, з одного боку, заощаджений час за рахунок рішень, пропонованих фреймворком, а з другого боку, час, що витрачений на вивчення нової платформи. За таких умов вибір доцільно спрямовувати на технологію, що дозволяє в мінімальні терміни вирішити потрібне завдання. Разом із тим слід відмітити, що вибір до застосування нової технології може стати дерелом таких ризиків, як: внутрішні помилки, слабка підтримка і документація від розробника, неопрацьованість рішень під конкретні задачі.

Взагалі, при виборі найбільш ефективного ВФК доцільно проаналізувати завдання, чітко його формалізувати, вибрати «топ-5» відповідних варіантів, написати на них невеликий додаток і обрати найкращий.

Також, існує потреба оцінювання ступеню відповідності функціоналу, з переліку претендентів, вимогам проекту, а також тривалість застосування готового функціоналу. При цьому,обов'язково потрібно звернути увагу на можливість зворотньої сумісності, якщо фреймворк змінюється до версії *API*. За відсутності даної можливості такий ВФК використовувати не доцільно.

Як на думку авторів, обрання з відомого набору ВФК вважається нескладним завданням тому, що при проектуванні відразу видно потребу в тих чи інших технологіях. Далі постає завдання збільшення відомих фреймворків, тому, в даному випадку, варто, в першу чергу, вивчити фреймворки, які охоплюють різні технології і найбільш відрізняються один від одного. Наприклад, будь-який класичний *MVC*-фреймворк та ВФК іншої інфраструктури.

Також, слід відмітити, що вибір ВФК, як будь-якого інструменту, залежить від термінів проекту. Якщо проект короткотривалий, тослід обирати, наприклад, з групи *Bootstrap*, або щось нове і паралельно його вивчати. Якщо проект довготривалий, то краще формувати окрему бібліотеку структур.

Сучасні ВФК містять велику кількість готового функціоналу і помітно прискорюють розробку проектів. Тому, сьогодні більш актуальним вважається завдання вибору фреймворків, а не мов програмування. Зазвичай, вбудований готовий функціонал дозволяє зібрати досить ефективну бібліотеку структур. Важливою перевагою готового функціоналу, безперечно, залишається уникнення рутинного програмування. Разом із тим, одним з основних ризиків застосування різних ВФК і їх адаптації до конкретних задач, є «людський чинник». Як тільки завдання виходять за межі певного фреймворка або виникає необхідність у використанні іншого, часто виникає потреба залучення інших фахівців.Якщо сегментація технічних рішень поглиблюватиметься, то так і буде. Але в даний момент більшість фреймворків підтримують схожий функціонал, і сегментація не надто висока. Специфічні рішення є тільки для наймасовіших видів завдань.

Також, слід зауважити, що мови і фреймворки, особливо на пізніх стадіях розвитку, це ортогональні речі. У розвинених екосистемах фреймворки, як правило, мають *API* для роботи з декількома мовами і навпаки: мови з великою екосистемою мають низку фреймворків, готових для використання.В сучасних умовах, при старті проекту, вибір фреймворку, зазвичай, стоїть на першому місці, ще до вибору мови. Якщо ж, наприклад, дістався *legacy*-проект, то, як правило, вибір мови відсутній. Мікросервиси, один з головних архітектурних трендів сьогоднішнього дня, намагається вирішити в тому числі і цю проблему — «звільнити» розробників від використання конкретної мови або фреймворку, щоб кожна частина проекту розроблялася на найбільш адекватних для неї технологіях.

Слід виділити важливі чинники, що дають гарантію надійності і стабільності фреймворку. Гарантія надійності фреймворка – його відмовостійкість. Правильна розробка, оптимізація запитів, ефективне кешування – ось запорука успіху. Наявність стабільних релізів, велика кількість користувачів, наявність автоматизованих тестів, наявність активності, *opensource* (відкритий код).

Приймаючи до уваги активність застосування ВФК слід систематизувати проведені дослідження напрямів їх розвитку. Так, проведений аналіз свідчить про те, що фреймворки будуть рухатися по шляху «полегшення» розробки. Типові рішення, готові модулі та інше – все для того, щоб непідготовлена людина змогла розгорнути сайт.

Для багатьох популярних платформ період нарощування функціональності пройшов. На даний момент розвиток сфокусовано на збільшенні швидкодії. Це справедливо і для *frontend*-, і для *backend*-фреймворків. В якості часткового рішення розглядається поділ фреймворка на логічні модулі. Для проекту повинен збиратися комплект бібліотек, що вирішують тільки необхідні завдання. А одним з рішень слід вважати щільніше з'єднання клієнтської та серверної частин фреймворків з єдиним синтаксисом та логікою роботи із контентом. Принцип «спрощення», який описано вище, призводить до того, що фреймворки стають складнішими за змістом, але все простішими для користувача.

Розвиток фреймворків залежить від потреб ринку та можливостей браузерів і призначених для користувача девайсів. Подальше удосконалення фреймворків, безпосередньо, залежить від розвитку і появи нових платформ, як, наприклад, розробка технології *Canvas, WebGL Node.js*–обумовила появу ВФК по роботі з ними.

Висновки. Проаналізувавши ряд наукових робіт, що стосуються використання фреймворків для розробки програмного забезпечення наочним є висновок, що незважаючи на велику популярність даних засобів, існують й протиріччя у їх використанні.

Вибір відповідного фреймворка є однією з основних засад, що визначає якість майбутнього сайту. Кожну з технологій, що наведені у даній статті можна вважати вичерпною у всіх відношеннях тому, що їх використовують, як у практиці перетворення в графічний інтерфейс (*frontend)*, так і для розробки варіанту архітектури програмного забезпечення (*backend*). Тому, слід обирати платформу у відповідності до зручностей щодо мов програмування.

Agile найбільше підходить для проектів із «відкритим кінцем», наприклад, запуск інтернет-сервісів, розробка комп'ютерних ігор, операційних систем. Однак, гнучкість може

приводити до втрати фокусу та втрати передбачуваності. Дуже важливо відокремлювати помилки застосування гнучкого підходу від недоліків самої методології. Перш, ніж будуть реалізовані всі переваги підходу, потрібно буде деякий час на адаптацію до реалій конкретних завдань.

ЛІТЕРАТУРА:

1. Лаврищева Е.М., Петрухин Е.М. Методы и средства инженерии программного обеспечения. Изд. МФТИ – Москва. – 2006. – 304 с.

2. С.С. Куликов. Тестирование программного обеспечения. Изд. «Четыре четверти» - Минск. – 2017. – 312 с.

3. Сем Канер, Джек Фолк, Енг Нгуен. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес приложения. Пер.с анг. – К.: Изд. «Диа Софт», - К. - 2001. – 544 с.

4. Мейерт Д.О. Небольшая книга о HTML /CSS фреймворках / Д.О. Мейерт // Орейли. – 2015. – 30 с.

5. Интернет-аудитория Украины. Статистика 2012-2013 и прогноз на 2014 год [Электронный ресурс] / Netpeak. Режим доступак ресурсу: www/ URL: http://blog.netpeak.ua/internet-auditoriya-ukrainy-statistika-2012-2013- i-prognoz-na-2014- god.

6. Агалаков С.А. Статистические методы анализа данных. - М. - 2018. - 18 с.

7. Білоконна К.В. Новітні технології розробки Web-додатків / Інноваційні та інформаційні технології в бізнесі та освіті [Електронний ресурс] : матеріали міжвузівського студентського вебінару (Вінниця, 21 жовт. 2015 р.) / відп. ред. Л.Б. Ліщинська. – Вінниця: ВТЕІ КНТЕУ, - 2015. – С. 9-10. – Режим доступу: http://www.vtei.com.ua/doc/materialu_vebinary.pdf

8. Лєнков С.В., Огнєвий О.В., Коваль Б.А., Присяжнюк В.В. Аналіз архітектурних особливостей обчислювальних систем з програмованою структурою // Збірник наукових праць Військового інституту Київського університету імені Тараса Шевченка. – Київ, - 2018. - №61. – С. 103 – 114.

9. WRITING ANGULAR 2 IN TYPESCRIPT [Електронний ресурс] – Режим доступу: http://victorsavkin.com/post/123555572351/writing-angular-2-in-typescrip – Назва з екрана.

10. Lenkov, S., Kubyavka, M., Kubiavka, L., Lenkov, Y., Shevchuk, V. Reflex Intellectual text processing systems: Natural language text addressing (2019) CEUR Workshor Proceedings, ISSN: 16130073. - 2386, pp. 85-95.

11. One framework. Mobile and desktop [Електронний ресурс] – Режим доступу: https://angular.io/ – Назва з екрана.

ЛІТЕРАТУРА:

1. Lavrishheva E.M., Petruhin E.M. (2006), Metody i sredstva inzhenerii programmnogo obespechenija [Software Engineering Methods and Tools]. Moscow: MIPT. 304 p.

2. Kulikov S.S. (2017), Testirovanie programmnogo obespechenija [Software Testing]. Minsk; Izd. «Chetyre chetverti». 312 p.

3. Sem Kaner, Dzhek Folk, Eng Nguen (2001). Testirovanie programmnogo obespechenija. Fundamental'nye koncepcii menedzhmenta biznes prilozhenija. Per.s ang. [Software testing. Fundamental concepts of business application management. Translated from English]. Kiev: Izd. «Dia Soft». 544 p.

4. Mejert D.O. (2015), Nebol'shaja kniga o HTML /CSS frejmvorkah / D.O. Mejert // Orejli. 30 p.

5. Internet-auditorija Ukrainy. Statistika 2012-2013 i prognoz na 2014 god [Internet audience of Ukraine. Statistics 2012-2013 and forecast for 2014] / Netpeak. URL: http://blog.netpeak.ua/internet-auditoriya-ukrainy-statistika-2012-2013- i-prognoz-na-2014- god.

6. Agalakov S.A. (2018), Statisticheskie metody analiza dannyh [Statistical methods of data analysis]. Moscow. 18 p.

7.Bilokonna K.V. (2015), Novitni tehnologii' rozrobky Web-dodatkiv / Innovacijni ta informacijni tehnologii' v biznesi ta osviti [Elektronnyj resurs] : materialy mizhvuzivs'kogo students'kogo vebinaru (Vinnycja, 21 zhovt. 2015 r.) / vidp. red. L.B. Lishhyns'ka. Vinnycja: VTEI KNTEU/ Pp. 9-10. URL: http://www.vtei.com.ua/doc/materialu_vebinary.pdf

8. Ljenkov S.V., Ognjevyj O.V., Koval' B.A., Prysjazhnjuk V.V. (2018). Analiz arhitekturnyh osoblyvostej obchysljuval'nyh system z programovanoju strukturoju [Analysis of architectural features of computer systems with programmable structure]. *Zbirnyk naukovyh prac' Vijs'kovogo instytutu Kyi'vs'kogo universytetu imeni Tarasa Shevchenka*. Kyiv, no. 61. Pp. 103 – 114.

9. WRITING ANGULAR 2 IN TYPESCRIPT. URL: http://victorsavkin.com/post/123555572351/writing-angular-2-in-typescrip.

10. Lenkov, S., Kubyavka, M., Kubiavka, L., Lenkov, Y., Shevchuk, V. Reflex Intellectual text processing systems: Natural language text addressing (2019) CEUR Workshor Proceedings, ISSN: 16130073. 2386. Pp. 85-95.

11. One framework. Mobile and desktop URL: https://angular.io/.

Doctor of Technical Sciences, Prof. Shynkaruk O.M.,
Ph.D. Miroshnichenko O.V., Ph.D. Yashyna O.M.

ON THE QUESTION OF QUALITY MANAGEMENT OF SOFTWARE WEB SYSTEMS BY DEVELOPMENT TOOLS

This article presents the results of research on existing tools and the definition of theoretical aspects of the use of modern frameworks for software development, as well as justification of the feasibility and impact of variability of choice on the quality of software applications. The main results of the study are the rationale for the choice of frameworks for software development. It should be noted that despite the large number of frameworks, the structured analysis is carried out by a rather small number of researchers, which determines the scientific novelty of this issue. Choosing the right framework is one of the main principles that determine the quality of the future site. Each of the technologies presented in this article can be considered exhaustive in all respects because they are used both in the practice of converting to a graphical interface (frontend) and to develop a variant of software architecture (backend). Therefore, you should choose a platform according to the convenience of programming languages.

So Agile is best suited for open-ended projects, such as launching Internet services, developing computer games, and operating systems. However, flexibility can lead to loss of focus and loss of predictability. It is very important to separate the errors of the flexible approach from the shortcomings of the methodology itself. It will take some time to adapt to the realities of specific tasks before all the benefits of the approach are realized. The practical significance lies in the possibility of using certain frameworks in accordance with the needs of the customer and the developer to achieve certain results in software development.

Keywords. Framework, CSS, software, software product, programming language.