

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

**Доценко Валерія Вікторівна**

Допускається до захисту:

завідувач кафедри

інформаційних технологій,

доктор технічних наук, доцент

\_\_\_\_\_ Т. В. Нескородева

« \_\_\_\_ » \_\_\_\_\_ 20\_\_ р.

**ТЕМА**

**Розробка та оптимізація універсальних веб-інтерфейсів для мобільних додатків**

Спеціальність 122 «Комп'ютерні науки»

**Кваліфікаційна (бакалаврська) робота**

Керівник:

Мартянова Т.А., старший викладач кафедри

інформаційних технологій,

старший викладач

Оцінка:

\_\_\_\_ / \_\_\_\_ / \_\_\_\_

(бали за шкалою ЄКТС/за національною шкалою)

Голова ЕК:

\_\_\_\_\_

(підпис)

## АНОТАЦІЯ

**Доценко В.В. Розробка та оптимізація універсальних веб-інтерфейсів для мобільних додатків.** Спеціальність 122 «Комп'ютерні науки», освітня програма

«Сучасні інформаційні технології та програмування». Донецький національний університет імені Василя Стуса, Вінниця 2022.

У кваліфікаційній (бакалаврській) роботі описано основні концепції розробки додатків для мобільних пристроїв. Поглиблено поняття, пов'язані з розробкою багатоплатформних мобільних додатків, а також досліджуються різні методології та інструменти. Проведено експерименти з розробки крос-платформних мобільних додатків та аналіз отриманих результатів.

Ключові слова: додаток, крос-платформеність, розробка, Web, iOS, Androud, платформа.

68 с., 18 рис., 32 джерело.

## ABSTRACT

Dotsenko V.V. Development and optimization of universal web interfaces for mobile applications. Specialty is 122 "Computer science", educational program

"Modern information technologies and programming". Vasyl' Stus Donetsk National University, Vinnytsia 2022.

In the qualification (to the bachelor) work the basic concepts of application development for mobile devices are described. The concepts connected with development of multiplatform mobile applications are deepened, and also various methodologies and tools are investigated. Experiments on the development of cross-platform mobile applications and analysis of the results.

Keywords: application, cross-platform, development, Web, iOS, Android, platform.

68 p., 21 pict., 32 source.

## ЗМІСТ

ВСТУП .....	5
РОЗДІЛ 1. МОБІЛЬНІ ТЕХНОЛОГІЇ .....	6
1.1 Загальні відомості про мобільні технології .....	6
1.2 Еволюція протоколів для мереж зв'язку .....	10
1.3 Сьогодення .....	11
РОЗДІЛ 2. РОЗРОБКА НАТИВНИХ МОБІЛЬНИХ ДОДАТКІВ .....	15
2.1 Розробка програм для Android .....	17
2.1.1 Еволюція .....	17
2.1.2 Процес розробки .....	24
2.2 Розробка нативних програм в iOS .....	26
2.2.1 Еволюція .....	26
2.2.2 Процес розробки .....	28
2.3 Нативна розробка додатків на Windows Phone .....	30
2.3.1 Еволюція .....	30
2.3.2 Процес розробки .....	31
2.4 Технічні відмінності між Android, iOS та Windows Phone .....	32
РОЗДІЛ 3. РОЗРОБКА КРОСПЛАТФОРМНИХ МОБІЛЬНИХ ДОДАТКІВ .....	34
3.1 Мобільні веб-додатки .....	35
3.1.1 Спеціальний ексклюзивний веб-додаток для мобільних пристроїв .....	36
3.1.2 Веб-додаток з адаптивним дизайном .....	36
3.2 Гібридні програми .....	37
3.2.1 PhoneGap .....	38
3.2.2 CocoonJS .....	39
3.2.3 Ionic .....	40
3.2.4 Sencha Touch .....	40
3.3 Інтерпретовані додатки .....	40
3.3.1 Appcelerator Titanium .....	41
3.3.2 NativeScript .....	41



3.4 Програми, створені за допомогою крос-компіляції.....	42
3.4.1 Xamarin.....	43
3.4.2 Embarcadero Delphi 10 Seattle.....	44
3.4.3 RubyMotion .....	45
РОЗДІЛ 4. ЕКСПЕРИМЕНТ .....	46
4.1 Мобільний додаток для платформи E-Learning WebUNLP .....	46
4.1.1 Опис задачі .....	46
4.1.2 Аналіз .....	47
4.1.3 Дизайн.....	47
4.1.4 Розробка.....	48
4.1.4.1 Нативний додаток для Android .....	48
4.1.4.2 Нативний додаток для iOS .....	49
4.1.4.3 Мобільний веб-додаток .....	49
4.1.4.4 Гібридний додаток, що розробляється за допомогою PhoneGap і jQuery Mobile .....	50
4.1.4.5 Гібридна програма, що розробляється за допомогою Sencha Touch .....	50
4.1.4.6 WebUNLP Interpreted Application з Appcelerator Titanium 3 .....	50
4.1.4.7 Програма, яка створюється за допомогою перехресної компіляції WebUNLP з використанням Xamarin/Visual Studio ..	51
4.1.4.8 Програма, що створюється за допомогою перехресної компіляції WebUNLP за допомогою Delphi 10 Seattle.....	52
4.1.5 Висновки експерименту WebUNLP .....	52
4.2 Підходи до розробки: Порівняльний аналіз ефективності.....	54
4.2.1 Опис задачі .....	54
4.3 Порівняльний аналіз підходів до розробки мобільних додатків ..	56
ВИСНОВКИ.....	63
СПИСОК ЛІТЕРАТУРИ	

## ВСТУП

Існують різні підходи до розробки кроссплатформних мобільних додатків. Виходячи з цього, метою цієї заключної роботи є розробка порівняльного аналізу переважаючих підходів та проведення експериментів із розробкою конкретних мобільних додатків для кожного аналізованого елемента.

Мета курсової роботи полягає у проведенні порівняльного аналізу між підходами до розробки багатоплатформних мобільних додатків. Здійснити розробку конкретного мобільного додатка в кожному багатоплатформному підході до розробки. Проаналізувати переваги та недоліки підходів відповідно до експерименту. Проаналізувати продуктивність додатків, які створюються з кожним підходом до багатоплатформної розробки. Вивчити вплив, що виникає в процесі розробки програмного забезпечення відповідно до підходу, який буде використано.

Робота складається із 4 розділів, 18 рисунків, загальний обсяг роботи 72 сторінок.

## РОЗДІЛ 1

### МОБІЛЬНІ ТЕХНОЛОГІЇ

#### 1.1 Загальні відомості про мобільні технології

Мобільні обчислення можна визначити як фізично мобільне обчислювальне середовище. Користувач мобільного обчислювального середовища матиме доступ до даних, інформацію або інші логічні об'єкти з будь-якого пристрою в будь-якій мережі в той час як знаходиться в русі [1]. Ці пристрої мають відмінні фізичні характеристики, серед яких виділяються їх розмір, вага, розмір екрану, механізм введення даних і можливість розширення. Крім того, важливу роль відіграють технічні аспекти, зокрема потужність обробки, простір пам'яті, автономність акумулятора, операційна система тощо.

Розробка програмного забезпечення для мобільних пристроїв ставить перед собою нові завдання, що впливають із унікальних характеристик цієї діяльності. Необхідність роботи з різними мережевими платформами, стандартами, протоколами та технологіями. Можливості пристроїв обмежені, але постійно розвиваються, та вимоги ринку у часі – це лише деякі з проблем, з якими потрібно мати справу[2].

Мобільні додатки створюються в динамічному та невизначеному середовищі. Як правило, вони невеликі, не критичні, але не менш важливі. Вони розраховані на велику кількість кінцевих користувачів і випускаються у швидких версіях для задоволення потреб ринку [3].

Незважаючи на все вищесказане, розробка програмного забезпечення для мобільних пристроїв суттєво відрізняється від традиційної [4] і супроводжує ріст і еволюцію програмної інженерії як дисципліни.

Щоб максимізувати свою присутність на ринку, програмний продукт повинен працювати на якомога більшій кількості пристроїв. Одне з рішень складається з нативної розробки програми на кожній із існуючих платформ з використанням інтегрованого середовища розробки (IDE), мови та інструментів



кожної платформи [5]. Однак, оскільки неможливо повторно використовувати вихідний код між різними платформами, зусилля збільшуються, а витрати на розробку, оновлення та розповсюдження нових версій зростають [6].

Міжплатформна розробка, на відміну від нативної розробки, зосереджена на повторному використанні коду. Прикладом такого підходу є створення мобільних веб-додатків. Однак обмеження, пов'язані з його виконанням у браузері, спонукали розробників програмного забезпечення звернути свою увагу на інші типи багатоплатформних додатків, результати яких ближчі до нативних рішень. У цьому контексті існують різні підкласифікації [5] [7] [8], і представляється інтерес проаналізувати характеристики, властиві кожній з них, шляхом побудови експериментального прототипу.

Був час, коли мобільні телефони не мали сенсорного екрану, вони не дозволяли записувати відео, не могли навіть підключитися до Інтернету. Вони також не могли надсилати та отримувати повідомлення. Вони просто дозволили базову функцію розмови по телефону.

Два десятиліття тому мобільні пристрої були великими і важкими пристроями. Лише декілька людей могли отримати доступ до цих пристроїв, оскільки вони вважалися предметами розкоші. Але цей сценарій поступово змінювався; що призводить до того, що мобільний телефон стає невід'ємним елементом нашого життя. В Аргентині в 2003 році було 4 мільйони активних ліній. Нині цифра сягає 37 млн. активних ліній і 62 млн. заявлених [10].

Друга світова війна створила потребу спілкуватися на відстані, тому Motorola створила військовий комплект під назвою H12-16 Handie Talkie для зв'язку за допомогою радіохвиль.

Його стрибок до цивільних систем відбувся наприкінці 40-х років з аналоговими радіосистемами в основному на частотах FM і послугами в діапазонах KB і УКХ, які пропонувала американська компанія Bell.

У 1955 році Ericsson випустила на ринок мобільний телефон системи А (MTA) Phone, телефон, який важив 40 кілограмів і встановлювався в автомобілях зображено на рис. 1.1. Цікаво, що до 1967 року він мав 125 користувачів.



Рис. 1.1. Мобільна телефонна система А (МТА)

Перший мобільний дзвінок датується 3 квітня 1973 року, коли Мартін Купер, тодішній інженер з електроніки Motorola, зателефонував з вулиці в Нью-Йорку зображено на рис. 1.2. Дзвінок був здійснений за допомогою прототипу Motorola DynaTAC 8000х. Вважається першим мобільним телефоном у світі, він важив близько кілограма і мав розміри 33х4,5х8,9 сантиметрів, його акумулятор мав автономність лише одну годину під час розмови, а для зарядки потрібно було десять годин. Його вартість становила 4000 доларів США.



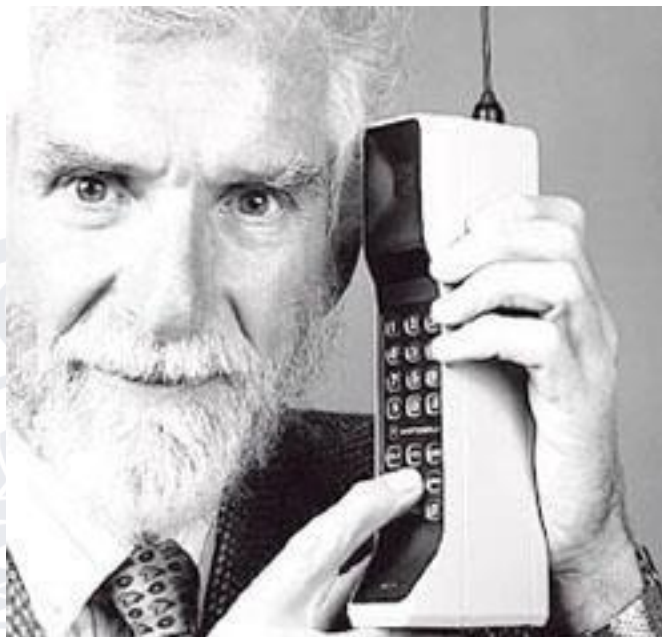


Рис. 1.2. Мартін Купер, автор першого мобільного дзвінка

Справжнє популярне знання про мобільний телефон з'явилося лише в 90-х роках, з другим поколінням цих пристроїв.

Цифровий зв'язок дозволив підвищити якість прослуховування голосу, зменшити розміри та вагу використовуваних пристроїв і, перш за все, збільшити кількість операційних компаній, які поступилися місцем більш конкурентному та доступному ринку.

Користувачі стали свідками дивовижних досягнень, таких як ідентифікація абонента або текстові повідомлення, які стали новою формою спілкування.

Запам'ятливо, мобільний телефон перетворився з того, що вважали робочим інструментом, до узагальнення в усіх прошарках суспільства.

У 1990-х роках мобільні телефони адаптувалися до цих подій, включаючи все більші екрани і, у найдосконаліших, виключаючи антену, яку потрібно було розгорнути, щоб здійснювати зв'язок у відповідних умовах.

Поступово в пристрої почали інтегрувати все більше і більше функцій: вібраційні оповіщення, ігри, bluetooth, інфрачервоний діапазон, політони, а вже на рубежі століть кольорові екрани та камери. Пізніше з'явилися відеозапис, доступ до Інтернету, сенсорні екрани та інше [11].

## 1.2 Еволюція протоколів для мереж зв'язку

Піднесення мобільних телефонів було б неможливим без стандартизації, вдосконалення та розвитку протоколів для мереж зв'язку та їх підтримки операторами. Таким чином, після перших комунікацій через радіохвилі зі смугою частот нижче 600 кГц, наступного зв'язку AM і FM, служб Bell і Ericsson у 50-х і 60-х роках, надійшов той перший дзвінок у 1973 році, який популяризував все в цьому секторі.

Перше покоління 1G було відповідальністю Ericsson із системою Nordic Mobile Telephony (NMT) і продовжувало використовувати аналогові канали. У 1986 році компанія модернізувала систему, що працює на частотах вище 900 МГц, що дозволило обслуговувати більшу кількість користувачів. На додаток до NMT, у 1980-х роках були розроблені інші системи мобільного телефону, наприклад Advanced Mobile Phone System (AMP), розроблена Bell Laboratories.

Друге покоління 2G з'явилося в 1990-х роках з такими системами, як Глобальна система мобільного зв'язку (GSM), внутрішній стандарт 136 (IS-136), інтегрована цифрова розширена мережа (iDEN) і внутрішній стандарт 95 (IS-95). GSM був найбільш актуальною розробкою, оскільки він був європейським стандартом для цифрової мобільної телефонії. У проєкті взяли участь 26 європейських телекомунікаційних компаній, а в 1992 році були запуснені перші європейські мережі GSM-900 і перші мобільні телефони GSM. Крім Європи, GSM також закріпився в Азії, Латинській Америці, Океанії та частині Північної Америки.

Потреба у більш високих швидкостях передачі даних і більших потужностях, які дозволяли б нові послуги, поступилася місцем третьому поколінню (3G), але не раніше, ніж 2.5G, який забезпечував загальну службу пакетної радіопередачі (GPRS). Європейським стандартом є універсальна мобільна телекомунікаційна система (UMTS), заснована на технології W-CDMA, і керується організацією 3GPP, яка також відповідає за GSM, GPRS та покращені швидкості передачі даних для розвитку GSM (EDGE).

Четверте покоління (4G) змінює технології 2G та 3G і пропонує, серед інших

покращень, більшу безпеку та якість обслуговування (QoS), а також набагато вищу швидкість доступу, ніж попередні (більше 100 Мбіт/с у русі та на 1 Гбіт/с в відпочинок). Він повністю заснований на протоколі IP, будучи системою систем і мережею мереж, що досягається завдяки конвергенції між дротовими і бездротовими мережами. За словами Vodafone, «4G нарешті має причину, і це розвага» [12]. Стандарт Long Term Evolution (LTE) є найпоширенішим, хоча і не єдиним з існуючих.

5G нова версія системи бездротового мережевого підключення, у таких країнах, як Південна Корея, а також у Європі. Як і 4G, вища швидкість передачі стане вашим найбільшим проривом зі швидкістю завантаження до 10 Гбіт/с (1,25 Гб/с) [13] подано на рис.1.3.



Рис. 1.3. Еволюція протоколів для мереж мобільного зв'язку

### 1.3 Сьогодення

Багато дій, які деякий час тому раніше виконувались з комп'ютера, наприклад, перевірка електронної пошти, прослуховування музики, планування нової зустрічі в календарі, обмін фотографіями в соціальній мережі, серед багатьох інших, тепер виконуються природним чином з мобільні пристрої. На



рис.1.4 показано, як мобільні пристрої завоювали переваги над командами традиційний так званий «настільний стіл».

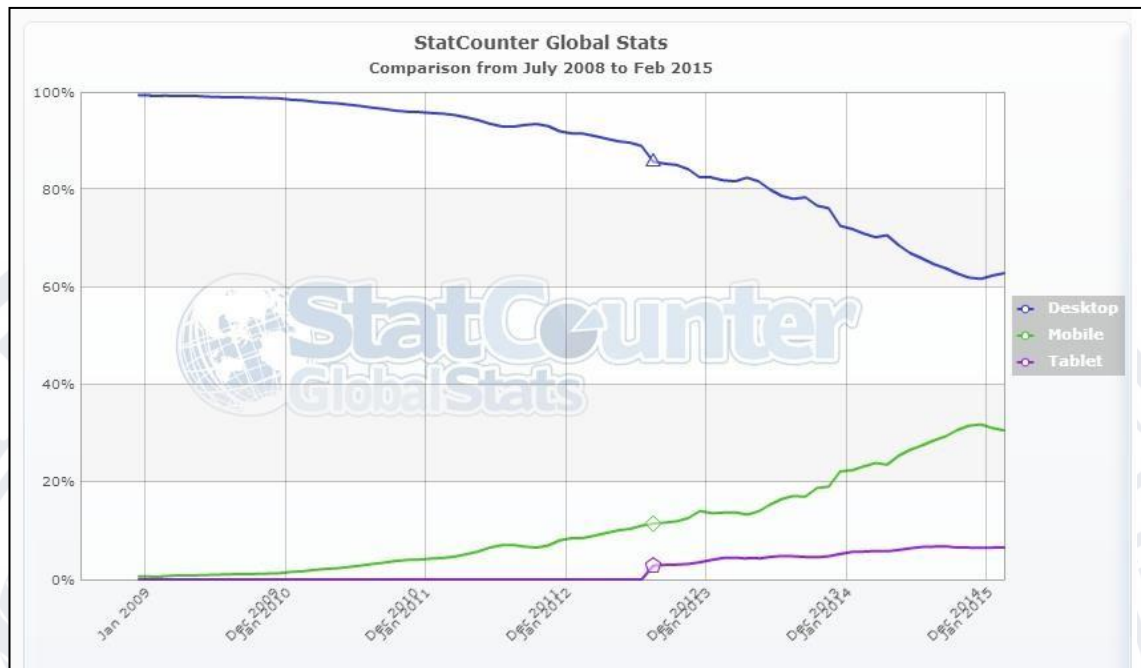


Рис. 1.4. Порівняння між платформами [14]

В даний час існують сотні різних моделей мобільних пристроїв і виробники продовжують впроваджувати інновації, намагаючись виділитися серед конкурентів. Сьогодні ми не дивуємося, коли знаходимо мобільні пристрої з чотирьохядерними процесорами 2,2 ГГц. Навіть прототипи пристроїв на вісім або десять ядер уже починають відомі.

Поруч із процесорами оперативна пам'ять мобільних пристроїв також значно зросла, досягнувши пристроїв з 3 ГБ пам'яті.

Камера пристроїв — це ще одна з можливостей, яка прогресувала найбільше, досягнувши сенсорів на 20 Мп, а також можливість запису відео у форматі UHD 4K (3840 x 2160).

Розмір екранів та їх роздільна здатність були ще однією з характеристик, які змінилися найбільше. Можна знайти пристрої з 5-дюймовими екранами, здатними відображати 16 мільйонів кольорів і з роздільною здатністю Quad HD (2560 x 1440).

Технологія батарей — це одна з областей, де не було досягнуто жодного

значного прогресу в галузі, і не очікується серйозних успіхів у короткостроковій перспективі. Навіть прибуття чіпсів більш потужні, великі екрани або 4G ще більше зменшать автономність батарей.

Останнім аспектом, про який варто згадати, є можливості пристрою. На додаток до вже класичних датчиків глобальної системи позиціонування (GPS), можливості підключення через bluetooth, інфрачервоний порт та WIFI, сьогодні смартфони можуть мати різноманітні варіанти.

Що стосується підключення, то все частіше зустрічаються термінали з механізмами зв'язку ближнього поля (NFC) або WIFI Direct.

Що стосується датчиків, то в даний час існують пристрої, які мають акселерометри, гіроскопи та магнітометри, які пропонують розробникам додатків можливість керувати положенням пристрою, прискоренням, яке зазнає пристрій по відношенню до сили тяжіння тощо. Використання датчиків наближення також поширене, наприклад, для вимкнення екрана під час дзвінка. Датчик яскравості дозволяє вимірювати яскравість навколишнього освітлення. Таким чином, програмне забезпечення телефону використовує ці дані для автоматичного регулювання яскравості екрану, роблячи його світлішим або темнішим.

Новітні смартфони високого класу також здатні вимірювати атмосферний тиск за допомогою барометра. Дані, виміряні пристроєм, використовуються для визначення висоти телефону над рівнем моря, що призводить до покращення точності GPS.

Існують навіть пристрої, які мають датчик вологості повітря, який дозволяє контролювати рівень вологості, щоб, наприклад, допомогти людям з проблемами дихання.

Наче всього цього було недостатньо, є моделі з крокоміром - для точного вимірювання кроків користувача - пульсометром - для вимірювання пульсації по кровоносних судинах пальців - і навіть датчиком відбитків пальців для проблем пов'язані з безпекою.

Завдяки вищезгаданому розробники додатків для пристроїв

Сучасні мобільні телефони стикаються з проблемою можливості маніпулювати за допомогою своїх програм всі описані можливості.





## РОЗДІЛ 2

### РОЗРОБКА НАТИВНИХ МОБІЛЬНИХ ДОДАТКІВ

Хоча розробка мобільних додатків сягає принаймні 10 років тому, з моменту відкриття магазину додатків iPhone у липні 2008 року спостерігається експоненційний ріст. Відтоді виробники пристроїв створили магазини додатків. Додатки для інших мобільних пристроїв, зокрема Android, BlackBerry, Nokia Ovi, Windows Phone, серед інших [15].

В даний час розробка додатків для мобільних пристроїв є сферою, що розвивається, з великим економічним і науковим інтересом. Підтвердженням цього є те, що до липня 2014 року в основних віртуальних магазинах додатків було доступно понад 3 мільйони додатків. І тільки в 2013 році в усьому світі було завантажено 101 мільярд мобільних додатків, з яких 9 мільярдів були платними і 92 мільярди безкоштовними [16].

У зв'язку з зростанням кількості мобільних платформ розробка мобільних додатків стала дуже важкою для бізнесу, оскільки їм потрібно розробляти однакові програми для кожної цільової платформи.

Розробка нативних програм — це природний спосіб розгортання мобільних додатків. Нативні програми розроблені для роботи на певній платформі, тобто потрібно враховувати тип пристрою, операційну систему та її версію.

Нативні програми розробляються з використанням інтегрованого середовища розробки (IDE), яке надає необхідні інструменти розробки для створення та налагодження програм. Вихідний код компілюється у виконуваний код, процес, подібний до традиційних настільних програм.

Коли програма буде готова до розповсюдження, її потрібно перенести в магазини додатків (магазини додатків) для кожної операційної системи. Ці магазини мають процес аудиту, щоб оцінити, чи відповідає програма вимогам

платформа для роботи. Після завершення цього кроку програма стає доступною для користувачів.

Основна перевага цього типу додатків — можливість взаємодії з усіма можливостями пристрою (камера, GPS, акселерометр, порядок денний тощо). Крім того, не обов'язково мати доступ до Інтернету. Його виконання швидке, його можна виконувати у фоновому режимі та повідомляти користувача про подію, яка потребує їхньої уваги.

Очевидно, що ці переваги мають вищу вартість розробки як аналог, оскільки залежно від платформи необхідно використовувати іншу мову програмування. Тому, якщо ви хочете охопити кілька платформ, ви повинні створити додаток для кожної з них. Це призводить до підвищення витрат на розробку, оновлення та поширення нових версій [9] [17] [18].

В даний час існує велика кількість операційних систем для мобільних пристроїв. На рис 2.1 показано використання операційних систем за останні роки в усьому світі, а на рис.2.2 показано те саме порівняння в Аргентині. В обох випадках рейтинг очолює Android, далі йде iOS і на третьому місці Windows Phone [19].

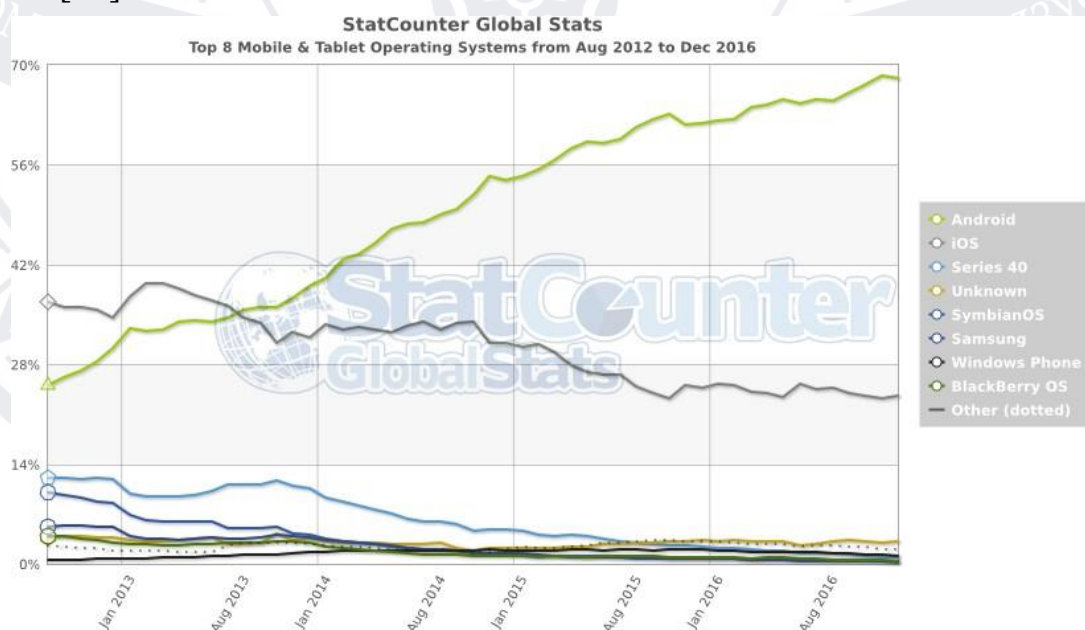


Рис.2.1. Використання мобільних операційних систем у світі

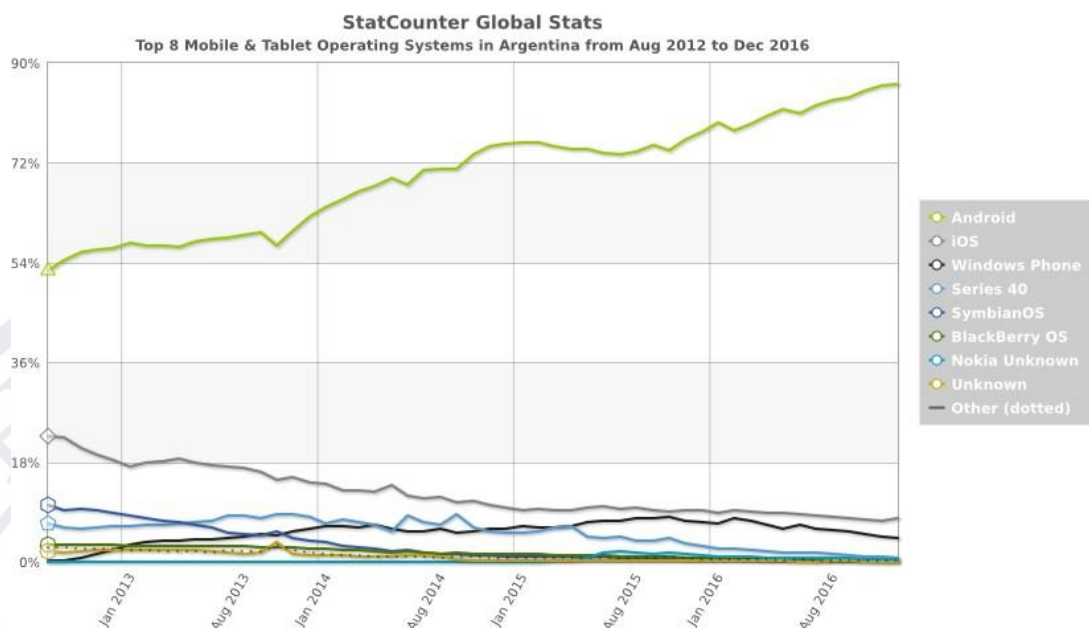


Рис.2.2. Використання мобільних операційних систем в Аргентині

Далі буде проаналізовано процес розробки нативних мобільних додатків у найактуальніших операційних системах сучасності; а потім у таблиці 1 буде порівняно технічні аспекти цих платформ.

## 2.1 Розробка програм для Android

Android є найбільш широко використовуваною операційною системою сьогодні. Його можна використовувати на смартфонах, планшетах, смарт-годинниках, телевізорах та автомобілях.

Він заснований на Linux і підтримується Google.

### 2.1.1 Еволюція

Спочатку розробкою Android займалася компанія Android Inc., яка була заснована в 2003 році і отримала фінансову підтримку від Google. Роками пізніше, точніше в липні 2005 року, Google придбала Android і в 2007 році породила першу версію операційної системи: Android 1.0 Apple Pie1, розроблену на ядрі Linux 2.6. У своїй презентації він підкреслив, що це повністю безкоштовна операційна система з відкритим кодом для мобільних пристроїв, Різниця iOS.

З моменту випуску першої версії операційної системи до теперішнього часу



Google дотримується традиції давати кожній із своїх версій назву десерту в алфавітному порядку.

Перша комерційна версія мала багато можливостей для вдосконалення і майже не турбувала конкуренцію, але вона вже вводила деякі концепції, які через роки стали стандартом для мобільних операційних систем:

- Розкривне меню сповіщень
- Віджети робочого столу
- Android Market, магазин додатків (у ньому не було системи оплати для користувачів. Весь каталог був безкоштовним)
- Інтеграція з Google Mail, Kontakтами та Календарем
- Браузер, Карти, Google Talk, програвач YouTube і підтримка камер.

У лютому 2009 року, всього через 3 місяці після виходу Android 1.0 Apple Pie, приходить перше оновлення, Android 1.1 Banana Bread. Він не додав значних нових функцій, але виправив численні помилки, виявлені в першій версії, і представив концепцію, яку на той час мало використовували конкуренти, яка прагнула полегшити життя користувачам: автоматичні оновлення, за допомогою яких він був дуже просто підтримувати все програмне забезпечення в актуальному стані.

30 квітня 2009 року вони випустили нове оновлення надзвичайної важливості з багатьма новими функціями з точки зору зручності використання: Android 1.5 Cupcake. Ця нова версія представила такі функції:

- Екранна сенсорна QWERTY-клавіатура з передбаченням тексту та словником користувача для користувацьких слів.
- Запис і відтворення у форматах MPEG-4 та 3GP
- Віджет Google на робочому столі для прямого пошуку
- SDK для розробки віджетів на робочому столі сторонніх розробників
- Розширені функції буфера обміну
- Покращений інтерфейс для запису та відтворення відео
- Можливість автоповороту.

Android 1.6 Donut з'явився у вересні 2009 року з деякими додатковими

новинами:

- Підтримка CDMA/EVDO, 802.1x, VPN, що розширило ринки, доступні для Android
- Підтримка різних дозволів екрана. Підтримка WVGA
- Оновлення та новий дизайн Android Market
- Універсальна пошукова утиліта в Інтернеті та на одному пристрої
- Галерея, камера та відеокамера з кращою інтеграцією та швидким доступом до камери
- Багатомовний механізм синтезу мовлення, який дозволяє будь-якому додатку Android «промовляти» текстовий рядок
- Покращення текстового та голосового пошуку

Всього через два місяці, в листопаді 2009 року, була випущена Android 2.0 Eclair, одна з найбільш суттєвих змін, які зазнала Android як з точки зору дизайну, так і внутрішньої архітектури. Це була версія, орієнтована на більші пристрої в той час, коли виробники почали диверсифікувати свою пропозицію. Версія 2.1 зберігала ту саму номенклатуру та виправляла лише кілька помилок, але її використання було вищим серед виробників, ніж попередня версія. Ось деякі з нових функцій:

- Розширена синхронізація облікових записів, що дозволяє користувачам додавати кілька облікових записів на пристрій для синхронізації електронної пошти та контактів
- Google Maps Navigation, безкоштовна система навігації GPS
- Сумісність з Microsoft Exchange
- Оптимізація швидкості обладнання та оновлений графічний інтерфейс.
- Підтримка більших розмірів і роздільної здатності екрана з кращим коефіцієнтом контрастності.
- Оновлений браузер, підтримка HTML5 та адресного рядка та уніфікований пошук
- Функція перетворення мовлення в текст, щоб писати тексти за допомогою голосу

- Новий екран розблокування.
- Нові функції камери, включаючи підтримку спалаху, цифрове масштабування, сюжетний режим, баланс білого, колірний ефект і макрофокус.
  - Додавання анімованих шпалер, що дозволяють анімацію фонових зображень головного екрана показувати рух.

20 травня 2010 року з'явилося нове оновлення операційної системи Android 2.2 Froyo. Він привніс численні зміни, деякі скопійовані з інших ROM, а інші з урахуванням комерційного використання:

- Повністю перероблений головний екран із 5 панелями замість 3
- Нова галерея зображень
- Підтримка роботи в якості точки доступу<sup>2</sup> для інших пристроїв, приєднання даних<sup>3</sup>
- Підтримка Flash 10.1
- Покращена функція копіювання та вставки в Google Mail
- Новий альтернативний екран розблокування PIN-коду
- Запис відео у форматі 720p
- Оптимізація швидкості, пам'яті та продуктивності. JIT-компілятор
- Інтеграція движка JavaScript V8 Chrome у браузер.
- Підтримка служби Android Cloud to Device Messaging (C2DM), що дозволяє надсилати push-повідомлення

2010 рік мав завершитися випуском надзвичайно популярної версії, яка панувала протягом тривалого часу: Android 2.3 Gingerbread. Ця версія мала такі особливості:

- Оновлений дизайн інтерфейсу користувача з підвищеною швидкістю та простотою
- Новий дизайн екранної цифрової клавіатури
- Підтримка надзвичайно великих розмірів і роздільної здатності екрана (WXGA і більше)
- Покращена функція копіювання та вставки з підтримкою окремих символів замість текстових полів



2 місце, яке пропонує доступ до Інтернету через бездротову мережу та маршрутизатор, підключений до постачальника послуг Інтернету

3 процес, за допомогою якого мобільний пристрій із підключенням до Інтернету діє як шлюз для надання доступу до мережі іншим пристроям

- Підтримка NFC
- Покращені інструменти візуалізації споживання та використання акумулятора
- Підтримка фронтальних камер
- Низький рівень доступу для розробників ігор
- Одночасний збірник сміття для підвищення продуктивності.
- Заміна файлової системи YAFFS на ext4
- Вбудована підтримка більшої кількості датчиків (таких як гіроскоп і барометр).
- Підтримка відео або голосового чату за допомогою Google Talk.

У лютому 2011 року Google випускає Android 3.0 Honeycomb. Це специфічне оновлення для планшетів, несумісне з телефонами, яке представило основні лінії інтерфейсу в майбутньому. Версії 3.1 і 3.2 зберігали ту саму назву і були в основному набором виправлень. Нижче наведено суттєві зміни, які відбулися в цих версіях:

- Перероблений головний екран
- Включення синіх тонів в інтерфейс на шкоду традиційному зеленому
- Нові функції для розміщення та використання віджетів
- Кінець фізичних кнопок. Автоматична адаптація ОС відповідно до пристрою
- Додана системна панель із функціями для швидкого доступу до сповіщень, станів і кнопок плавної навігації, доступних у нижній частині екрана.
- Додано панель дій, що забезпечує доступ до контекстних параметрів, навігації, віджетів або іншого типу вмісту у верхній частині екрана.
- Спрощена багатозадачність – торкнувшись «Останні програми» на

системній панелі, користувачі зможуть переглядати знімки виконуваних завдань і швидко переходити між додатками.

- Апаратне прискорення графіки
- Оптимізація візуалізації 3D-графіки
- Підтримка периферійних пристроїв USB

Жовтень 2011 року став місяцем випуску Android 4.0 Ice Cream Sandwich, заснованого на ядрі Linux 3.0.1. У цій новій версії внесено такі покращення:

- Новий інтерфейс Holo та шрифт Roboto
- Покращена система управління повідомленнями
- Покращена багатозадачність
- Android Beam, функція передачі даних між двома пристроями за допомогою NFC
- Функція розблокування обличчям
- Нові функції для перегляду та керування споживанням даних
- Нові програми пошти та календаря
- Можливість доступу до програм безпосередньо з заблокованого екрана.
- Інтегрований інструмент для зніmkів екрана
- Підтримка MKV
- Апаратне прискорення інтерфейсу користувача
- Підтримка стилуса (сенсорна ручка)

У липні 2012 року представлена Android 4.1 Jelly Bean, яка включає наступні нові функції:

- Оптимізована система виявлення сенсорного введення
- Запуск Google Now, інтелектуальної служби голосового помічника Google
- Браузер Google Chrome
- Покращений голосовий пошук
- Нові можливості для інтерактивних сповіщень на робочому столі
- Автономний голосовий диктант
- Flash Player більше не підтримується

Через кілька місяців, у листопаді 2012 року, була випущена версія Android 4.2 Jelly Bean, яка включає регулярні виправлення, покращення продуктивності та конкретні зміни в деяких програмах.

Android 4.3 Jelly Bean анонсується 24 липня 2013 року. Однією з цілей цієї версії була спроба консолідувати Android як операційну систему, здатну запускати ігри. Ця нова версія має такі нові функції:

- Покращена підтримка кількох користувачів і профілів
- Підтримка OpenGL ES 3.0
- Розумний Bluetooth
- Платформа Google Games
- Розширені служби визначення місцезнаходження Wi-Fi
- Більше не потрібно натискати піктограму мікрофона, щоб виконати голосовий пошук. Вам просто потрібно сказати «OK Google», а потім замовити команді те, що потрібно.

Android 4.4 KitKat анонсується в жовтні 2013 року. Ця нова версія намагається виправити проблему фрагментації: існує багато версій, і виробники мають труднощі з адаптацією своїх продуктів до вимог кожної нової версії. Основні моменти цієї версії:

- Зменшені вимоги до обладнання для виправлення фрагментації версії
- Сумісний з терміналами з 512 МБ ОЗП
- Зменшення споживання батареї за рахунок оптимізації датчиків
- Включення офісного пакету QuickOffice
- Інтегровані служби онлайн-сховища: Google Drive, Box...
- Підтримка інфрачервоного випромінювання. Використовуйте мобільний телефон як пульт від телевізора
- Повноекранні програми
- Скріншот на відео

У жовтні 2014 року виходить Android 5 Lollipop, який підтримує інші пристрої, такі як Android TV, розумні годинники, автомобілі тощо. Ця нова версія включає:



- Новий дизайн на основі Material Design, який забезпечує більш плавний робочий процес
- Інтерфейс, який адаптується до будь-якого розміру пристрою
- Оновлена система розумних сповіщень
- Цікавий режим багатозадачності, який показує шари з різними відкритими програмами
- Функція Android Smart Lock, яка дозволяє підключати один пристрій Android до іншого, будь то розумний годинник або автомобіль.
- Режим «Гість», щоб ви могли позичити пристрій без доступу інших користувачів до особистої інформації.
- Підтримка 64-розрядних систем

У жовтні 2015 року було анонсовано Android 6 Marshmallow, який включає такі нові функції:

- Підтримка аутентифікації за відбитком пальця.
- Швидке видалення програм з головного екрана.
- Нова схема енергоменеджменту під назвою Doze.
- Довший час роботи від акумулятора, коли пристрій неактивний.
- Додаток Android Pay, який використовуватиме чіп NFC.
- Покращення в Google Асистент.
- Більший контроль над дозволами, які вимагають програми.

### **2.1.2 Процес розробки**

Нативні програми для цієї платформи реалізовані за допомогою мови програмування Java.

Стандарт пропонує використовувати офіційну IDE Android Studio , хоча можна використовувати й інші середовища зображено на рис.2.3. Незалежно від використовуваної IDE, вам потрібен комплект розробки програмного забезпечення (SDK) для версії Android, з якою ви хочете працювати. SDK надає всі інструменти, необхідні для розробки, компіляції, налагодження та моделювання програм.

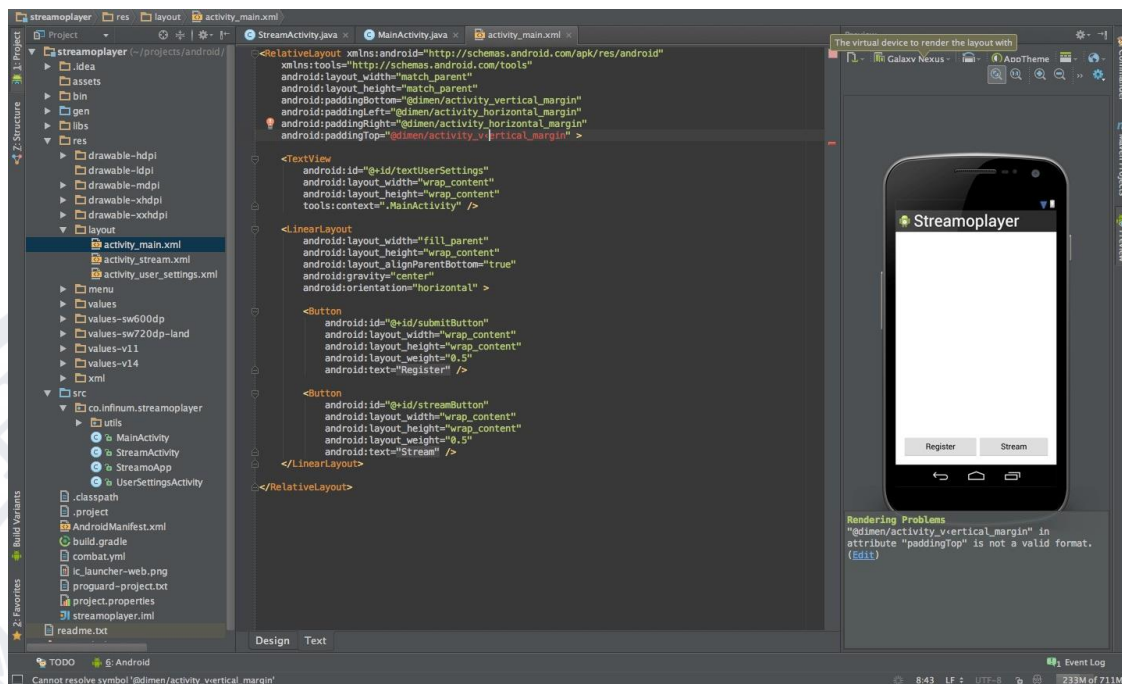
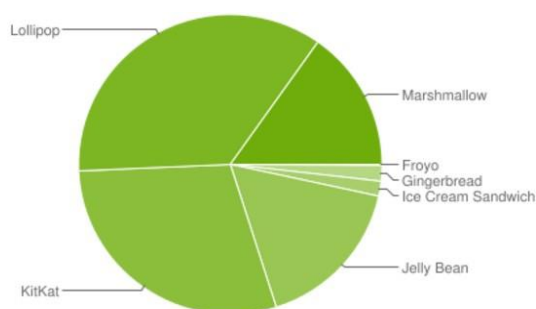


Рис 2.3. Середовище розробки Android Studio

Одне з питань, що виникають при створенні програми, — з якою версією програмного інтерфейсу програм Android (API) працювати. Необхідно враховувати, що при роботі з конкретною версією пристрої, які будуть підтримувати створюваний додаток, обмежені. Не рекомендується використовувати старий API, оскільки багато з досягнень платформи не будуть використані, а також останній на ринку, оскільки мало користувачів зможуть ним користуватися. На рис.2.4. показано розподіл версій за активними пристроями, станом на грудень 2016 року [20]. З цієї інформації можна зробити висновок, що якщо вони виберуть працювати з API 16 (Jelly Bean), 96,6% активних мобільних телефонів зможуть використовувати створену програму.

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.6%
4.1.x	Jelly Bean	16	6.0%
4.2.x		17	8.3%
4.3		18	2.4%
4.4	KitKat	19	29.2%
5.0	Lollipop	21	14.1%
5.1		22	21.4%
6.0	Marshmallow	23	15.2%



Дані зібрані протягом 7 днів до 1 серпня 2016 року. Версії з поширенням менше 0,1% не відображаються

Рисю2.4. Розповсюдження версій Android – грудень 2016 року

## 2.2 Розробка нативних програм в iOS

iOS — операційна система для мобільних пристроїв від транснаціональної компанії Apple. Хоча спочатку вона була розроблена для пристрою iPhone, згодом операційна система охопила інші пристрої, такі як iPod Touch iPad. На відміну від інших операційних систем для мобільних пристроїв, iOS працює лише на пристроях, створених компанією Apple, але не на іншому сторонньому обладнанні.

### 2.1.1 Еволюція

Мобільна операційна система iOS є похідною від операційної системи OS X, яка базується на UNIX.

Перша версія операційної системи була представлена в січні 2007 р. На відміну від наступників, перша версія називалася не iOS, а iPhone OS. Це була досить замкнута платформа, оскільки єдиними доступними програмами були ті, які були інтегровані в операційну систему та створені Apple, наприклад, Maps, Mail, Photos, Calendar та інші.

Розробка додатків сторонніх розробників не відбувалася до березня 2008



року, коли Apple випустила перший набір для розробки.

Через кілька місяців виходить друга версія операційної системи: iPhone OS 2. Цього разу вони представляють App Store, який дозволяє завантажувати сторонні програми. Крім того, програма «Карти» отримала підтримку push-повідомлень GPS та Mail.

У 2009 році Apple випустила iPhone OS 3, а версія 3.2 була перейменована в iOS 3.2. iOS 3 принесла чудові нові функції: пошук Spotlight, кнопки копіювання та вставки, запис відео, редагування відео, MMS-повідомлення, голосові команди, альбомний режим та багато інших програм. Крім того, переваги push-повідомлень продовжували використовуватися, а його SDK продовжував розвиватися, надаючи розробникам додатків більше можливостей.

iOS 4 з'явилася в середині 2010 року, принісши з собою нові зміни на платформі. Найбільш помітними були включення шпалер і можливість відкривати кілька програм одночасно. Крім того, програми можна згрупувати за папками, щось просте, але корисне.

iOS 5 була випущена в червні 2011 року. Ця нова версія запропонувала чудові нові функції: центр сповіщень, Siri - віртуальний помічник, який задав тенденцію останніх років, iMessage, iCloud, Game Center, кіоск, нагадування, соціальну інтеграцію для Twitter, і рідна програма для перегляду веб-сторінок, Safari, була значно покращена.

iOS 6 була представлена в червні 2012 року і принесла в якості новинок включення Apple Maps (замінивши Google Maps), покращення Siri з набагато більш точною інформацією, інтеграцію Facebook і Twitter в систему, режим «Не турбувати», підтримку LTE, серед інших.

Користувачі цієї операційної системи отримали одне з найбільших оновлень у червні 2013 року, коли була випущена iOS 7. Ця нова версія мала повну переробку інтерфейсу користувача на основі прозорості та градієнтів. Крім того, були представлені Центр керування, AirDrop, оновлений додаток для фотографій, iTunes Radio та CarPlay, покращення перемикача додатків, уповільнені відео, серед іншого.

У червні 2014 року стала доступна iOS 8, нова версія операційної системи, яка намагається вирішити деякі помилки iOS 7 і додати певні покращення. Розширення та можливість підключати додатки один до одного через меню спільного доступу зробили iOS 8 набагато продуктивнішою. Він також містить віджети.

Центр сповіщень або інтерактивні сповіщення. Крім того, реалізовано улюблені контакти, клавіатури сторонніх розробників, функція Reachability і утиліта Continuity, а також програми Apple Pay, Health, HandOff, QuickType, Family Sharing, iCloud Drive і Apple Music.

У 2015 році була представлена iOS 9, яка внесла повну зміну у використання системи завдяки сумісності з 3D Touch iPhone 6s і iPhone 6s Plus. Нотатки були оновлені новими функціями, Кіоск було замінено на Новини, покращено Passbook, який було перейменовано в Wallet, а Карти з маршрутами та трафіком були покращені. Додані комбінації клавіш розділеного перегляду, зображення в картинці та сторонніх розробників. Крім того, тепер система може краще використовувати автономність пристрою завдяки режиму економії батареї, який обмежує підключення та функції системи, щоб не споживати стільки енергії.

Нарешті, у червні 2016 року стала доступна iOS 10, версія, яка характеризується відкритістю для розробників. За допомогою цієї нової версії ви можете отримати доступ до Siri, 3D Touch і прямих програм з операційної системи. Крім того, центр віджетів тепер більш інтуїтивно зрозумілий, а екран блокування — корисніший. Як зазвичай, покращення можна побачити в різних програмах операційної системи, а також можна видалити програми Apple, які не використовуються користувачами.

### **2.2.2 Процес розробки**

Мобільні додатки для платформи iOS розробляються з використанням мови програмування Objective C. Офіційним середовищем розробки є платформа Xcode [21], яка працює спільно з Interface Builder — графічним інструментом для створення інтерфейсів користувача.

На даний момент розробляються дві основні версії мобільної операційної

системи, як видно на рис.2.5. Примітно, що останню версію операційної системи підтримують 63% активних пристроїв. Очевидно, що внутрішня фрагментація набагато менша, ніж в Android.

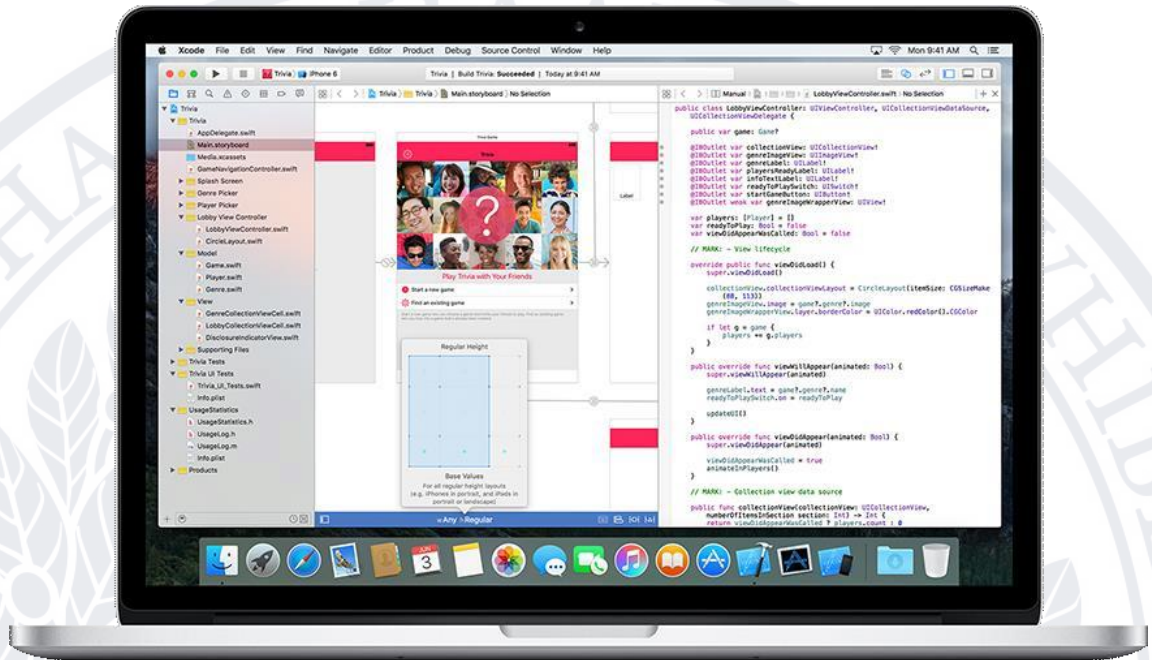
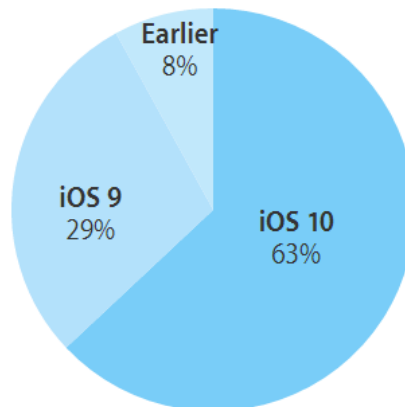


Рис.2.5. Середовище розробки Xcode

63% of devices are using iOS 10.



As measured by the App Store on November 27, 2016.

Рис.2.6.Розповсюдження версій iOS – листопад 2016 року



## 2.3 Нативна розробка додатків на Windows Phone

Мобільна операційна система Windows Phone була розроблена компанією Microsoft і прийшла на зміну Windows Mobile. На відміну від свого попередника, він орієнтований на споживчий ринок, а не на ринок бізнесу.

### 2.3.1 Еволюція

Її перша версія, випущена у вересні 2010 року, називалася Windows Phone 7. Вона була представлена як нова операційна система, що включає функції інтеграції із сервісами Xbox Live і Zune. Інтерфейс, відомий як «Metro», був повністю перероблений і має схожі візуальні характеристики з інтерфейсом пристрою Zune HD.

Windows Phone 7.5 — це оновлення для Windows Phone, випущене у вересні 2011 року. Ця нова версія принесла багато функцій, таких як включення Internet Explorer 9, підтримка медіазапитів CSS3 та підтримка використання GPS під час роботи з мобільними додатками. серед інших.

У січні 2013 року представлено Windows Phone 7.8, оновлення, яке вносить такі покращення, як новий інтерфейс користувача та користувацькі фони для екрана блокування. Це було останнє велике оновлення для Windows Phone 7, оскільки Microsoft зосередилася на Windows Phone 8.

Windows Phone 8 — це наступна версія операційної системи, яка була представлена наприкінці 2012 року. Нові функції включають:

- Нові більш налаштовані домашні екрани та екрани блокування
  - Дитячий куточок: ексклюзивний і контрольований простір для дітей.
  - Гаманець: для зберігання карт лояльності та кредитних карток
  - NFC
  - Internet Explorer 10
  - Інтеграція зі Skype
  - Нове ядро Windows NT з підтримкою багатоядерних процесорів.
- ядра
- Скріншот

Його наступна версія — Windows Phone 8.1 і була представлена в квітні 2014 року.

Серед його найбільш релевантних характеристик можна виділити:

- Центр сповіщень
- Голосовий помічник, відомий під ім'ям Кортана
- Датчики (Wi-Fi, дані та акумулятор)
- Програми, які входять у пакет інсталяції, такі як: Здоров'я та фітнес, Їжа та напої, Подорожі та Карты (останнє конкурує з Here Maps, програмою карт Nokia).
- Покращення головного екрана: можливість додавати шпалери та третій стовпець настроюваної мозаїки (плитки).

У середині 2014 року Microsoft вирішила змінити свою стратегію, намагаючись уніфікувати всі платформи, такі як ПК, планшети, смартфони, Xbox One та інші. Windows 10 (мобільна) була публічно представлена під час події 21 січня 2015 року. Як і його аналог, Windows 10 Mobile використовує Microsoft Edge як браузер за замовчуванням, замінюючи Internet Explorer з Windows Phone 8.1, оскільки він не підходить для нових ліній дизайну операційної системи.

### **2.3.2 Процес розробки**

Найпопулярнішим середовищем розробки для розробки програм Windows Phone є Visual Studio рис.2.7. В даний час є можливість використовувати різні мови програмування: VB.NET і C#.NET, C++ і JavaScript.

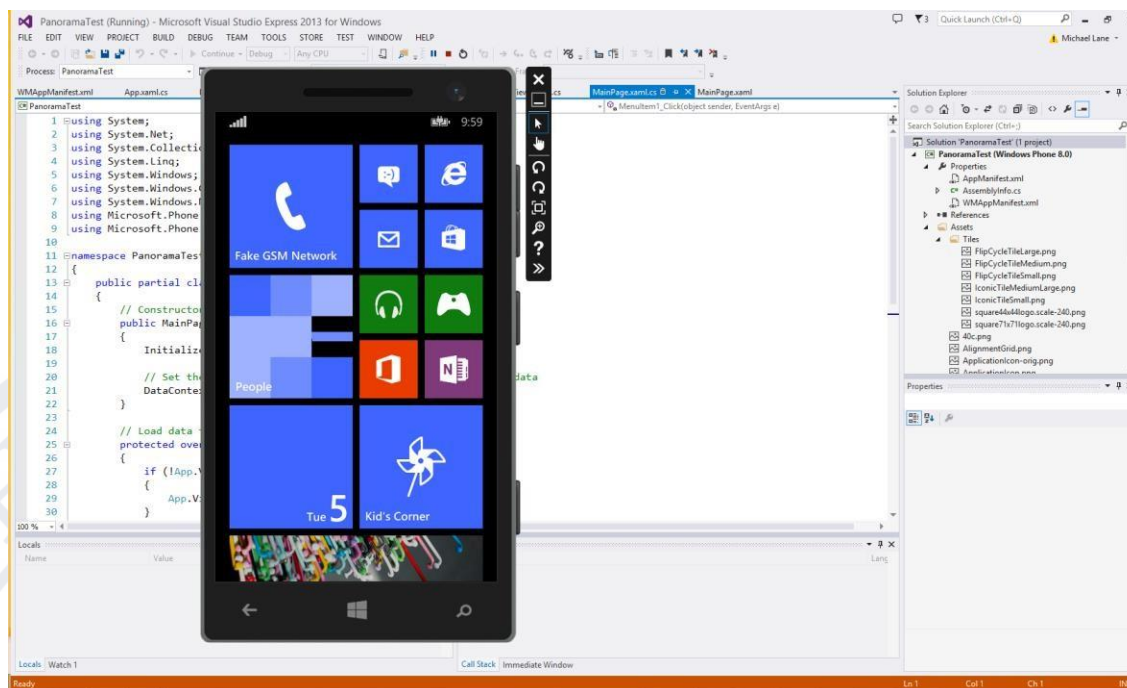


Рис.2.7. Середовище розробки Visual Studio

## 2.4 Технічні відмінності між Android, iOS та Windows Phone

Розробка мобільних додатків для трьох найпопулярніших операційних систем передбачає знання певних технічних аспектів кожної платформи. Деякі з цих технічних аспектів суттєво відрізняються для кожної операційної системи.

Однією з характеристик, яку слід враховувати, є те, чи працюють мобільні додатки всередині спеціально розробленої віртуальної машини, чи вони працюють безпосередньо над операційною системою.

Інший аспект, який розробники повинні враховувати, - це мова програмування для створення рідних програм. Цей аспект важливий, оскільки впливає на інші аспекти, такі як управління пам'яттю.

На додаток до мови програмування, яка буде використовуватися, важливо знати технологію, необхідну для розробки інтерфейсу користувача.

Аналогічно, розробники повинні ознайомитися з різними IDE залежно від платформи, на якій вони хочуть розробляти, а також з різними магазинами додатків, де розміщувати кінцеві продукти.

Нарешті, апаратні технології, необхідні для підтримки операційної системи, також відрізняються залежно від платформи.

У таблиці 2.1. порівнюється кожен із цих аспектів у мобільних операційних



системах Android, iOS та Windows Phone.

Таблиця 2.1 Технічні відмінності між платформами Android, iOS та Windows Phone

Операційна система	Віртуальна машина	Мова програмування си	Інтерфейс користувача	Управ ління пам'яттю	IDE	Платф а розробки
Android	Dalvik VM	Java	XML files	Garbag e collector	Andr oid Studiorma	Multip
iOS	No	Objective- C	Cocoa Touch	Refere nce counting	XCo de	Mac C
Windows Phone	CLR	C# and .Net	XAML files	Garbag e collector	Visu al Studio	Windo Vista/7

### РОЗДІЛ 3.

#### РОЗРОБКА КРОСПЛАТФОРМНИХ МОБІЛЬНИХ ДОДАТКІВ

В останні роки ринок мобільних пристроїв, особливо смартфонів, демонструє значне зростання як в Аргентині, так і в усьому світі. Зокрема, в нашій країні найбільше зросли платформи Android та iOS [14,20].

В даний час велика частина індустрії програмного забезпечення зосереджена на розробці рішень для мобільних пристроїв, особливо на нативних додатках.

Вбудовані програми, як згадувалося в попередньому розділі, пропонують можливість доступу до всіх можливостей пристрою (камера, GPS, акселерометр і порядок денний, серед інших), їх продуктивність висока, доступ до Інтернету не є строго необхідним і вони можуть працювати в фон, який сповіщає користувача, коли потрібна його увага. Ці програми можна поширювати/продавати через відповідні інтернет-магазини.

Основним завданням для постачальників додатків є забезпечення рішень для всіх платформ, але це тягне за собою високу вартість [6]: неможливо повторно використовувати вихідний код між різними платформами, примножуючи зусилля та підвищуючи витрати на розробку, оновлення та поширення нові версії.

Розробка між платформою спрямована на оптимізацію співвідношення витрат і користі шляхом спільного використання однакового кодування між версіями для різних платформ. Серед інших переваг виділяються: менший час і вартість розробки; вбудовані функції з доступом до апаратного забезпечення пристрою та наявністю потужних середовищ розробки (Delphi, Visual Studio тощо) або; натомість використання технологій (HTML5, Javascript і CSS), добре відомих веб-розробникам, які можуть передати свої знання та досвід у мобільну парадигму. Однак продуктивність програм та їхніх користувацьких інтерфейсів

може вплинути на роботу користувача.

Міжплатформні програми можна класифікувати на: мобільні, гібридні, інтерпретовані та крос-компільовані веб-додатки [5]. Кожна з цих класифікацій буде розглянута в наступних розділах.

### **3.1 Мобільні веб-програми**

Мобільні веб-додатки, розроблені для роботи в браузері, розроблені з використанням стандартних веб-технологій (HTML, CSS і JavaScript) і мають ряд переваг: їм не потрібно адаптуватися до будь-якого операційного середовища, вони незалежні від платформи та запуск швидкий і простий.

З іншого боку, час їх відповіді зменшується через взаємодію клієнт-сервер. У той же час вони виявляються менш привабливими, ніж рідні програми, оскільки не встановлені на пристрої, що передбачає спочатку доступ до браузера. Крім того, обмеження безпеки, накладені на виконання коду через браузер, обмежують доступ до всіх можливостей пристрою [22]. Так само неможливо розробити веб-програми, які працюють у фоновому режимі або навіть офлайн-програми, оскільки для їх роботи потрібне підключення до Інтернету.

Процес розробки мобільного веб-додатка повинен враховувати ряд характеристик, властивих середовищу виконання. Як вже було сказано, доступ до мобільних веб-додатків можна отримати з будь-якого мобільного пристрою, який має браузер і доступ до Інтернету, але це не означає, що навігація в додатку зручна або оптимальна. Наприклад, веб-додаток з меню, що складається з 20 пунктів, може означати правильне використання з ПК. На відміну від цього, для мобільного пристрою з обмеженим екраном користувачеві недоцільно мати таку кількість опцій у меню. На цьому прикладі мається намір показати, що для того, щоб програма була доступною з різних мобільних пристроїв, може знадобитися створити принаймні два різних дизайну презентації: дизайн для ПК та інший дизайн для мобільних пристроїв з організаційною структурою. застосування більш спрощено, по відношенню до традиційного дизайну. Наприклад, деякі функції програми для ПК можуть бути вилучені для мобільної версії або представлені інакше.



Щоб розробити різні презентації одного веб-додатка, наразі існують дві різні стратегії, обидві дійсні залежно від контексту. Кожен з них буде розглянуто в наступних розділах.

### **3.1.1 Спеціальний ексклюзивний веб-додаток для мобільних пристроїв**

Ця методологія була стандартною до 2012 року. У ній URL і HTML-код мобільної версії відрізняються від настільної версії. URL-адреси зазвичай мають формат `m.sitename.com`.

Деякі з переваг цього методу:

- Загалом завантаження відбувається швидше, а перегляд зручніший.
- Можна легше адаптувати зміст розділів.

Недоліки цього методу:

- Більш дороге обслуговування. Ви повинні підтримувати дві різні версії одного сайту.
- Труднощі з позиціонуванням сайту в пошукових системах.
- Ці розробки, зрештою, мають тенденцію бути дуже скороченими версіями Інтернету для ПК, що призводить до розчарування користувача і віддає перевагу перегляду оригінальної версії Інтернету.

### **3.1.2 Веб-додаток з адаптивним дизайном**

За допомогою цієї методології веб-додаток адаптується до пристрою, з якого до нього здійснюється доступ. HTML-код та URL-адреса програми є унікальними. Адаптивний дизайн, або адаптивний дизайн рис.3.1. виник у 2008 році, коли W3C обговорив і описав його цілі. Починаючи з 2012 року, коли Google настійно рекомендував використовувати його [23], він зростає і, можливо, стане стандартом у короткостроковій перспективі.

Завдяки адаптивному дизайну всі елементи Інтернету змінюються по ширині та висоті, адаптуючись до розміру вашого екрана. Можна навіть приховати розділи при доступі з мобільного пристрою.



Рис 3.1. Адаптивний дизайн

Основні переваги розробки веб-додатків з адаптивним дизайном

є:

- Обслуговування є менш дорогим, оскільки вам потрібно підтримувати лише одну версію програми.
- Спрощено процес позиціонування програми в пошукових системах.

З іншого боку, адаптивний дизайн має ряд недоліків:

- Потрібний більш високий технічний рівень для свого розвитку.
- Вміст додатків із адаптивним дизайном може бути більшим, ніж вміст конкретних веб-додатків для мобільних пристроїв, що впливає на час завантаження при використанні не дуже потужного типу з'єднання та генерує більше споживання даних.

В даний час існує велика кількість фреймворків, які спрощують розробку веб-додатків з адаптивним дизайном, фреймворки Bootstrap [24] і Foundation [25], які виділяються своєю популярністю та великою документацією.

### 3.2 Гібридні програми

Гібридні програми використовують веб-технології (HTML, Javascript і CSS), але не виконуються браузером. Натомість вони працюють у веб-контейнері (webview) як частина рідної програми, яка встановлена на пристрої мобільному.

З гібридної програми можна отримати доступ до можливостей пристрою через різні API.

Гібридні програми пропонують великі переваги, дозволяючи повторне використання коду на різних платформах, доступ до апаратного забезпечення пристрою та розповсюдження через магазини додатків. З іншого боку, є два недоліки гібридних додатків по відношенню до рідного випадку: i) користувацький досвід погіршується через невикористання рідних компонентів в інтерфейсі, i ii) виконання сповільнюється навантаженням, пов'язаним з веб-контейнером.

На щастя, існує безліч фреймворків, які дозволяють розробляти гібридні додатки. Нижче наведено короткий виклад найпопулярніших.

### **3.2.1 PhoneGap**

PhoneGap [26] — це безкоштовний фреймворк з відкритим вихідним кодом, створений у 2008 році компанією Nitobi Software, що дозволяє створювати мобільні додатки за допомогою стандартних веб-технологій: HTML, JavaScript і CSS. У жовтні 2011 року Adobe купує компанію Nitobi і надає код PhoneGap до Apache Software Foundation, щоб почати новий проект над PhoneGap під назвою Apache Cordova [27]. PhoneGap став одним із кількох дистрибутивів Apache Cordova рис.3.2., надавши додаткові послуги, такі як Adobe PhoneGap Build, хмарну службу побудови, надану Adobe Creative Cloud, PhoneGap Desktop та PhoneGap Developer.

PhoneGap дозволяє компілювати програми для наступних платформ: Android, IOS, Amazon Fire OS, Windows Phone, Windows 8, Ubuntu, Tizen, Firefox OS, Blackberry 10.



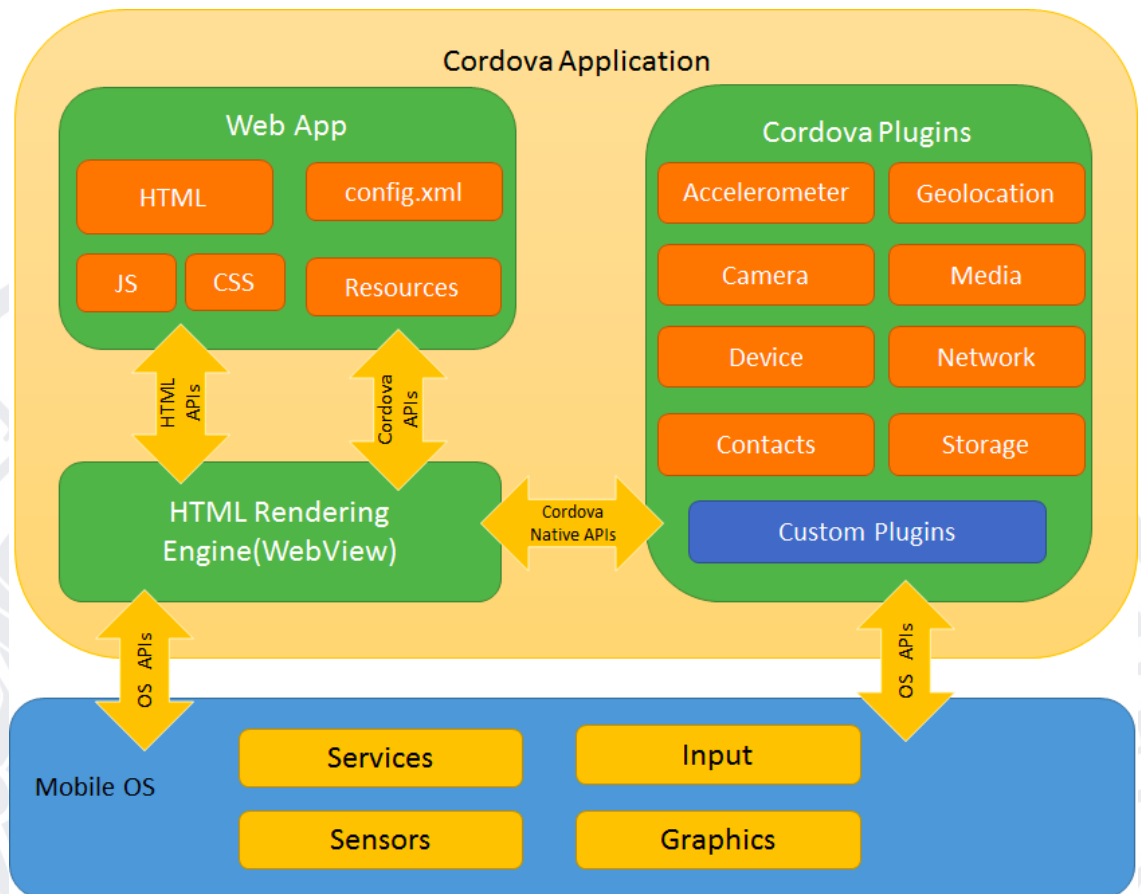


Рис.3.2: Операція Apache Cordova

### 3.2.2 CocoonJS

CocoonJS — це фреймворк для розробки мобільних додатків, розроблений Ludei. Наразі він має власну ліцензію та пропонує різні типи платних та безкоштовних планів. Вищезгаданий фреймворк заснований на Apache Cordova, як і PhoneGap, але він відрізняється від останнього тим, що робить більший акцент на продуктивності отриманих програм. CocoonJS пропонує два типи веб-перегляду:

- **Canvas+**, легкий веб-перегляд, оптимізований для відеоігор. Він надає лише підмножину функцій HTML5 (канва, аудіо, рух, геолокація тощо), необхідних для розробки відеоігор, щоб забезпечити найкращу можливу продуктивність.
- **Webview+**, потужний і повний веб-перегляд, заснований на веб-перегляді проекту Chromium, який має кращу продуктивність порівняно зі стандартним

веб-переглядом Apache Cordova.

CocoonJs дозволяє створювати програми як для iOS, так і для Android.

### **3.2.3 Ionic**

Ionic — це повноцінний безкоштовний пакет SDK з відкритим вихідним кодом для розробки гібридних додатків, вироблений Максом Лінчем, Беном Сперрі та Адамом Бредлі з компанії Drifty Co в 2013 році. За відносно короткий час він досяг неабиякої популярності. Він заснований на Apache Cordova, що дозволяє розробляти програми за допомогою веб-технологій. На відміну від інших фреймворків для розробки гібридних додатків, Ionic надає платформу інтерфейсу користувача та візуальні інструменти для проектування інтерфейсів.

За допомогою Ionic можна створювати програми для Android, IOS, Windows Phone та Firefox OS.

### **3.2.4 SenchaTouch**

Sencha Touch — це безкоштовний JavaScript-фреймворк Model-View-Controller (MVC), побудований на основі системи класів Ext JS, спеціально розроблений для розробки мобільних веб-додатків для сенсорних пристроїв [28].

Як і PhoneGap, Sencha Touch дозволяє розробникам створювати мобільні додатки з веб-розробки за допомогою HTML5, CSS і Javascript. На відміну від PhoneGap, Sencha не вимагає ніякої сторонньої бібліотеки для створення інтерфейсів, схожих на рідні, оскільки фреймворк надає велику кількість готових до використання візуальних компонентів.

Sencha пакує код проекту за допомогою PhoneGap, щоб потім ви могли створювати програми на всіх платформах, які підтримує PhoneGap.

## **3.3 Інтерпретовані додатки**

Інтерпретовані додатки реалізуються за допомогою базової мови, більшість з якої перекладається на рідний код, а решта інтерпретується під час виконання. Ці програми реалізовані незалежним від платформи способом, використовуючи різні технології та мови, такі як JavaScript, Java, Ruby та XML, серед інших.

Однією з головних переваг такого типу застосування являються повністю рідні інтерфейси користувача. Однак розробники відчують повну залежність

від вибраного середовища розробки.

Далі будуть представлені деякі з найбільш відомих фреймворків для розробки інтерпретованих додатків.

### **3.1.1 Appcelerator Titanium**

Розроблений компанією Appcelerator Inc., Appcelerator Titanium є фреймворком з відкритим кодом, який дозволяє створювати мобільні додатки для платформ iOS та Android. Ця структура складається з Titanium Studio, середовища розробки з відкритим кодом для кодування багатоплатформних мобільних додатків, і Titanium SDK, серії інструментів для розробки, тестування, аналізу, налагодження та компіляції додатків.

Програми, розроблені за допомогою Appcelerator Titanium, кодуються за допомогою мови Javascript, яка оцінюється під час виконання інтерпретатором Javascript, що працює в операційній системі пристрою. Під час запуску програми код Javascript виконується з рідного коду, а через Titanium API кожен елемент коду Javascript відображається один за одним із рідними елементами. Таким чином, Titanium API діє як міст, забезпечуючи користувацькі інтерфейси, що складаються з вбудованих елементів керування. Це середовище розробки використовує фреймворк Alloy, розроблений для гнучкої розробки мобільних додатків. Alloy заснований на архітектурі MVC і підтримує використання популярних технологій, таких як Backbone.js і Underscore.js.

Творці Titanium відзначають, що завдяки цьому фреймворку можна повторно використовувати від 60% до 90% коду між різними платформами.

### **3.3.2 NativeScript**

5 березня 2015 року компанія Telerik запустила NativeScript, проект з відкритим кодом, який дозволяє генерувати нативні програми за допомогою JavaScript. Аналогічно, можна розробляти програму за допомогою мови TypeScript, вільної мови з відкритим вихідним кодом, розробленої Microsoft, яка розширює JavaScript, по суті додаючи статичну типізацію та об'єкти на основі класів. У цьому сенсі під час компіляції програми виконується переклад коду TypeScript у код JavaScript.



NativeScript надає кросплатформний модуль, який дозволяє отримувати власні програми з коду JavaScript. Цей модуль узгоджено розкриває можливості пристрою та базової платформи та дозволяє отримати до них доступ із коду JavaScript. Аналогічно інтерфейси користувача можна визначити за допомогою коду JavaScript, HTML-документів і файлів CSS, абстрагуючись від справжніх компонентів.

Коли додаток компілюється, частина кросплатформного коду транслюється в рідний код, а решта інтерпретується під час виконання зображено на рис.3.3

На даний момент NativeScript дозволяє створювати програми для Android та iOS, а також планується підтримка Windows Phone. NativeScript відтворює власні елементи керування без використання WebView, досягаючи продуктивності та користувацького досвіду, подібних до нативних програм.

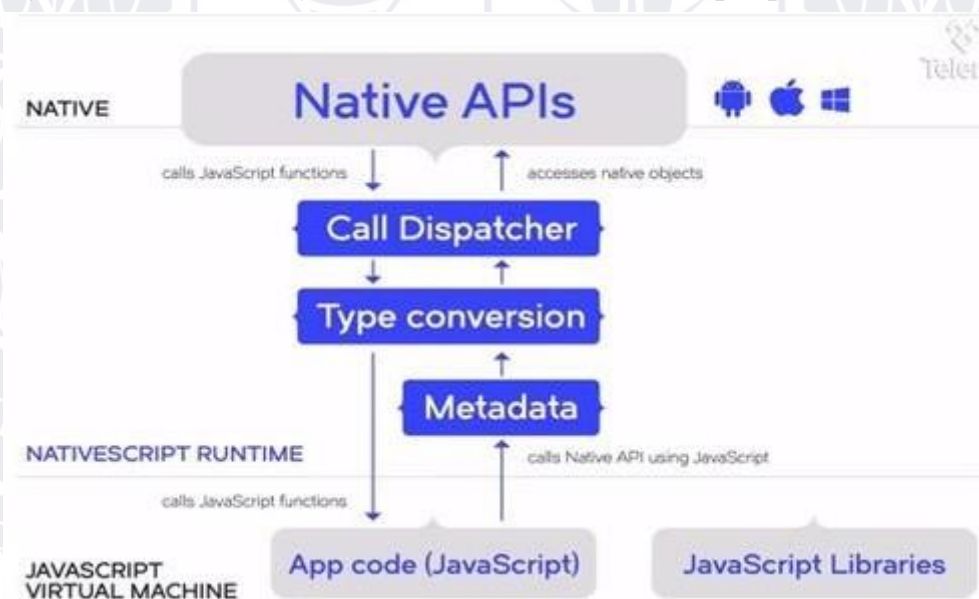


Рис.3.3. Процес інтерпретації за допомогою NativeScript

### 3.4 Програми, створені за допомогою крос-компіляції

Ці додатки компілюються, створюючи конкретну високопродуктивну версію для кожної цільової платформи. Прикладами середовищ розробки для крос-компіляції програм є Applause, Xamarin, Embarcadero Delphi 10 Seattle та RubyMotion.

Відкрите середовище розробки Applause використовує як вхідну мову предметно-специфічний на основі фреймворку Xtext, явно розроблений для

мобільних додатків, орієнтованих на дані, і генерує вихідний код на Objective C, Java, C# або Python. Функції Xamarin, Embarcadero Delphi 10 Seattle і RubyMotion представлені в наступних розділах.

### 3.4.1 Xamarin

Xamarin — це платформа для розробки мобільних додатків для створення 100% оригінальних програм для iOS, Android і Windows із загальної бази коду C#/.NET для повторного використання коду від 75% до майже 100% між платформами. Програми, написані на Xamarin і C#, мають повний доступ до базових API платформи, а також можливість створювати власні інтерфейси користувача та компілювати в рідний код, тому вплив на продуктивність під час виконання є слабким.

Xamarin — компанія, заснована в травні 2011 року тими ж інженерами, які створили проект Mono, що складається з безкоштовної реалізації платформи розробки .NET для пристроїв Android, iOS і GNU/Linux. Раніше цей проект називався MonoTouch і MonoDroid.

Хоча Xamarin має власну IDE під назвою Xamarin Studio рис.3.4. її можна інтегрувати з Microsoft Visual Studio, і таким чином також створювати програми для Windows, включаючи Windows RT для планшетів і Windows Phone для мобільних телефонів.

У лютому 2016 року Xamarin купує Microsoft, і з цього моменту платформа Xamarin і IDE Xamarin Studio стали безкоштовними.

Xamarin пропонує кросплатформний підхід до розробки, в якому використовується загальне кодування бізнес-логіки. Однак інтерфейси повинні програмуватися незалежно для кожної з цільових платформ. Таким чином, повторне використання коду, за статистичними дослідженнями компанії Xamarin, наближається до 85%.

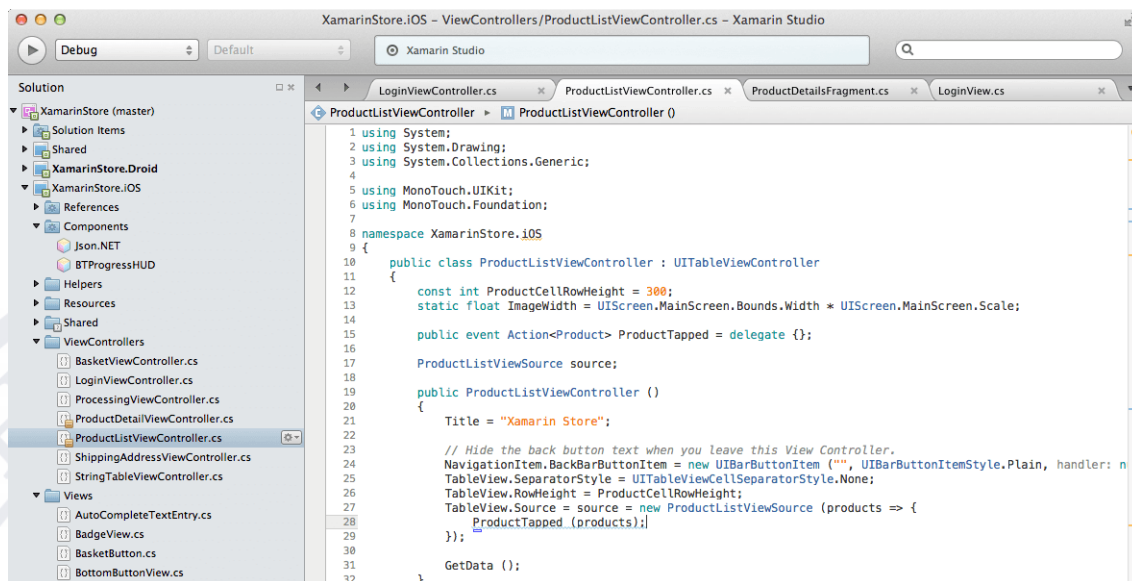


Рис.3.4. Середовище розробки Xamarin Studio

### 3.4.2 Embarcadero Delphi 10 Seattle

Embarcadero Delphi 10 Seattle — це запатентована і небезкоштовна платформа розробки, яка дозволяє розробникам швидко створювати програми за допомогою зручного та інтуїтивно зрозумілого візуального середовища показано на рис.3.5. Розроблений додаток можна скопіювати для кількох платформ, включаючи Windows, Mac, Android та iOS.

Delphi 10 Seattle, як і Xamarin, перевершує інші типи кросплатформних програм зі 100% нативними результатами, маючи повний доступ до датчиків і можливостей пристрою (камера, сповіщення, GPS, акселерометр тощо).

Варто відзначити, що Delphi 10 Seattle через платформу інтерфейсу користувача FireUI Multi-Device Designer дозволяє створювати повністю власні інтерфейси користувача без необхідності розробляти спеціальні інтерфейси для кожної конкретної платформи



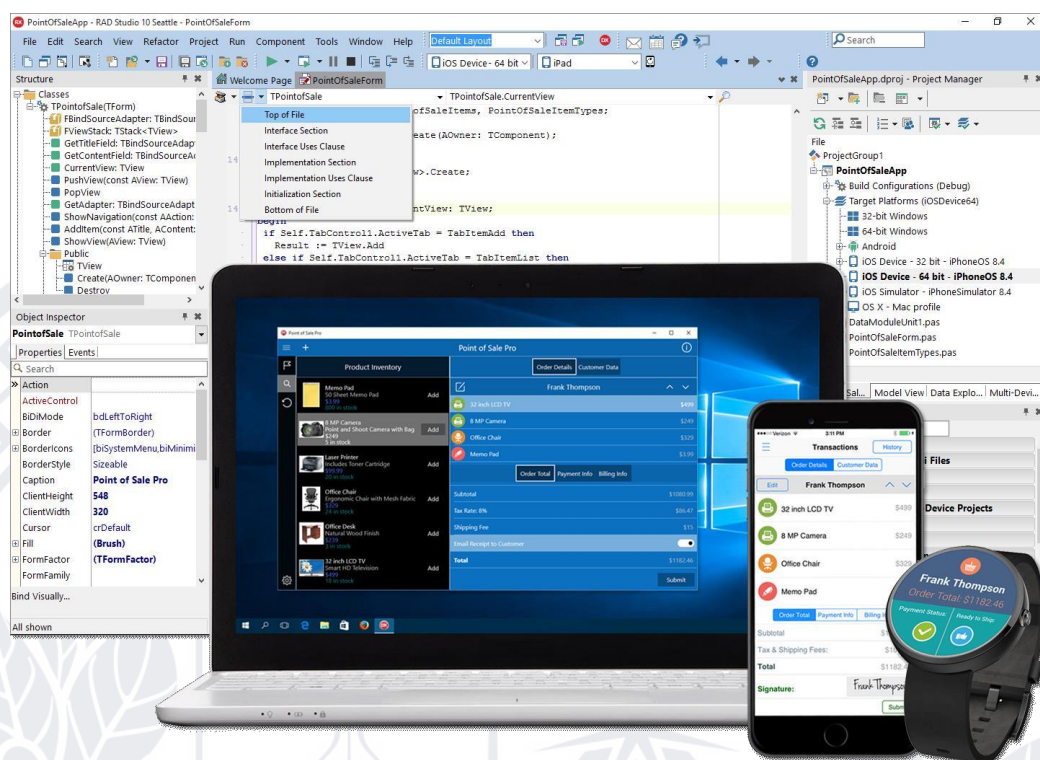


Рис.3.5. Середовище розвитку Delphi 10 Сіетл

### 3.4.3 RubyMotion

У травні 2012 року компанія HipByte запускає RubyMotion, проект, який дозволяє генерувати програми для iOS, Android і OS X, використовуючи мову програмування Ruby. Для створення багатоплатформних додатків RubyMotion має реалізації Ruby на цільових операційних системах (Android та iOS), які надають доступ до всіх функцій, наданих кожною платформою. Інструменти збірки, надані RubyMotion, дбають про вбудовування відповідної реалізації Ruby під час збірки для кожної платформи. Основним недоліком RubyMotion є те, що він працює лише під операційною системою OS X. Ще одним фактором, який слід враховувати, є те, що він не має інтегрованого середовища розробки, що полегшує весь процес для розробника; Це вирішується завдяки тому, що він пропонує доповнення до найбільш використовуваних IDE та редакторів коду.

Нарешті, хоча продукт є платною послугою, він пропонує можливість спробувати його безкоштовно, хоча з підтримкою лише для компіляції до останньої версії кожної цільової платформи.

## ГЛАВА 4.

### ЕКСПЕРИМЕНТ

З метою поглиблення дослідження різних методологій розробки багатоплатформних мобільних додатків одночасно проводилися різні експерименти. Перший експеримент складається з дослідження етапів розробки мобільного додатка з використанням кожної з методологій, вивчених у попередньому розділі, з метою аналізу процесу розробки. Другий експеримент спрямований на аналіз продуктивності мобільних додатків, створених за допомогою досліджуваних методологій, за допомогою математичних розрахунків. Нарешті, представлено порівняльний звіт, який може допомогти у прийнятті рішень про те, яку методологію використовувати в майбутніх розробках.

#### **4.1 Мобільний додаток для платформи E-Learning WebUNLP**

##### **4.1.1 Опис задачі**

WebUNLP – це віртуальне середовище для викладання та навчання, яке дозволяє вчителям виступати посередником у своїх освітніх пропозиціях. Студенти та викладачі можуть зустрічатися в цьому просторі для обміну навчальними матеріалами, спілкування та створення освітнього досвіду віртуально.

Наразі WebUNLP має веб-версію, орієнтовану на настільні чи портативні комп'ютери, але вона не адаптована для використання з мобільних пристроїв.

Додаток можна покращити завдяки розробці яка полягає в розширенні WebUNLP за допомогою побудови мобільного додатка, який надає доступ до певних функцій системи через мобільний пристрій. Запропонований підхід включає аналіз одного і того ж рішення, порівняння нативної, веб-гібридної, інтерпретованої та крос-компільованої розробки, щоб встановити, яка з них підходить.



Як і будь-яка розробка програмного забезпечення, створення мобільного додатка передбачає чітке визначення його призначення та вимог, яким воно має відповідати. Зокрема, у випадку програмного забезпечення для мобільних пристроїв важливо мати більш обмежені цілі, ніж у його настільній версії [29].

#### **4.1.2 Аналіз**

Одним із перших питань, які виникли, був вибір платформи. З точки зору ринку, операційними системами, які переважають, є Android та iOS [15], за допомогою яких було вирішено підтримувати ці дві операційні системи.

Функціональні та нефункціональні вимоги аналізувалися ізольовано від платформи, а потім окремо для кожної з них.

Ось деякі вимоги, яким повинна відповідати програма:

- Користувач повинен мати можливість увійти в програму з тими ж обліковими даними, які використовувалися для доступу до веб-версії для настільних комп'ютерів.
- Користувач повинен мати доступ до рекламних щитів усіх курсів, у яких він бере участь, як викладач, чи студент.
- Користувач повинен отримувати сповіщення на своєму пристрої, коли новина розміщена на білборді. Ця вимога не може бути виконана у веб-версії, доступній із настільних та/або портативних комп'ютерів.
- Користувач повинен мати однаковий досвід роботи на всіх операційних платформах.
- Існуючий веб-додаток має бути синхронізований з мобільним додатком, який розробляється, це означає, що будь-які зміни, внесені з мобільного додатка, мають відображатися у веб-версії і навпаки.

#### **4.1.3 Дизайн**

Щоб задовольнити вимоги, зазначені в попередньому пункті, за винятком сповіщення, дизайн мобільного веб-додатка складається з копії того, що пропонує WEBUNLP, лише адаптуючи інтерфейс до розміру екрана мобільного пристрою.



Для проектування нативних, гібридних, інтерпретованих і крос-компільованих мобільних додатків ситуація складніша. На рис.4.1. показано загальну архітектуру всіх компонентів, які беруть участь у цьому сценарії розробки. Він показує доступ з ПК до WebUNLP і доступ з мобільних пристроїв до інформації WebUNLP через веб-служби. Для розробки веб-сервісів було розроблено Restful API, враховуючи його простоту, масштабованість та взаємосумісність [30].

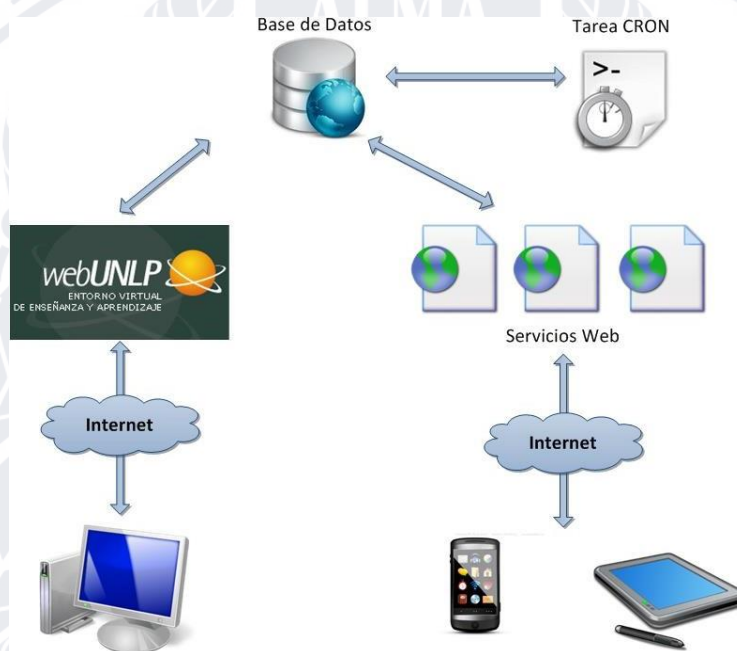


Рис.4.1. Загальна архітектура для нативного та гібридного додатка

Існує також запланована задача (Cron), яка виконується з перервами через регулярні проміжки часу, яка сповіщає відповідні мобільні пристрої, коли новий елемент створюється на рекламному щиті WebUNLP. Cron визначає операційну систему пристрою для сповіщення та створює зазначене сповіщення.

#### 4.1.1 Розробка

##### 4.1.4.1 Нативний додаток для Android

Для розробки додатків для платформи Android потрібен Java Development Kit (JDK) і його середовище програмування, відоме як Android SDK. Останній надає необхідні бібліотеки та інструменти для створення, тестування та налагодження додатків Android.

#### 4.1.4.2 Нативний додаток для iOS

Платформа Apple iOS заснована на власній моделі, тому розробка рідного додатка iOS передбачає наявність Apple Mac під керуванням OS X з встановленим Xcode. Основною мовою програмування є Objective C. Xcode — це середовище розробки Apple для всіх її пристроїв і відповідає за забезпечення iOS SDK необхідними інструментами, компіляторами та фреймворками. Крім того, Xcode інтегрується з симуляторами для пристроїв iOS (iPhone та iPad), які полегшують етапи тестування розробленої системи.

Щодо аспектів, пов'язаних з графічним інтерфейсом та взаємодією з користувачем, необхідно дотримуватись конвенції, запропоновані Apple [31], щоб досягти кращої інтеграції програми з операційною системою та покращити роботу користувача.

Нарешті, щоб задовольнити вимогу сповіщення, Cron повідомляє відповідний пристрій iOS за допомогою служби Apple Push Notification Service (APN).

#### 4.1.4.3 Мобільний веб-додаток

Веб-додаток, який має доступ до рекламного щита WebUNLP, доступний для будь-якого мобільного пристрою, який має браузер, який підтримує функції, що використовуються для розробки: HTML5, CSS3 та Javascript.

Оскільки швидкість передачі/приймання даних мобільного пристрою через Wi-Fi, і зокрема 3G, нижча за швидкість настільного комп'ютера. на робочому столі, веб-версія рекламного щита WebUNLP є легкою, і значна частина вимог реалізується через Ajax [32], щоб уникнути перезавантаження всієї сторінки у разі зміни.

Через обмеження програм, які запускаються у браузері, неможливо реалізувати отримання сповіщення на пристрої, коли новий елемент публікується на білборді.

#### **4.1.4.4 Гібридна програма, що розробляється за допомогою PhoneGap і JQuery Mobile**

Для побудови гібридного додатка WebUNLP необхідно використати фреймворк PhoneGap [26], що дозволяє розробляти мобільні додатки, що використовують технології, загальні для всіх пристроїв: HTML5, CSS та Javascript.

Аналогічно, фреймворк JQuery Mobile Javascript використовувався для досягнення інтерфейсів з узгодженим виглядом і поведінкою на різних мобільних платформах.

Для реалізації шаблону проектування MVC може бути використана бібліотека Backbone.js.

Нарешті, щоб задовольнити вимогу сповіщення, необхідно використати плагін Pushwoosh.

#### **4.1.4.5 Гібридна програма, розроблена за допомогою Sencha Touch**

Розробці WebUNLP за допомогою Sencha Touch сприяє використання шаблону проектування MVC. Це дозволяє отримати більшу гнучкість і читабельність у коді.

Sencha пропонує Sencha Command, міжплатформний інструмент командного рядка, який забезпечує автоматизовані завдання протягом усього життєвого циклу програми. У випадку WebUNLP він використовувався як для генерації проекту, так і для пакування програми.

#### **4.1.4.6 WebUNLP Interpreted Application з Appcelerator Titanium 3**

Для розробки експериментального мобільного додатка за інтерпретованим підходом можна використати безкоштовне середовище розробки з відкритим кодом Appcelerator Titanium 3.

Простота і читабельність контролерів і моделей розробленого додатка виділяються після використання Alloy, Titanium framework, що дозволяє проектувати програми за шаблоном проектування MVC. Для побудови представлень можна вибрати програмування за допомогою Javascript і Titanium API, або за специфікацією XML зі стилями TSS (Titanium Style Sheets). Останній



варіант спрощує процес створення представлення, хоча для його покращення все ще відсутній хороший редактор візуального інтерфейсу.

Щоб задовольнити вимоги щодо сповіщень на пристрої, Titanium надає модуль PushNotifications для платформ Android та iOS.

#### **4.1.4.7 Програма, яка створюється за допомогою перехресної компіляції WebUNLP з використанням Xamarin/Visual Studio**

Як згадувалося в розділі 3.4.1, Xamarin пропонує кросплатформний підхід до розробки, в якому використовується загальне кодування бізнес-логіки. Однак інтерфейси повинні програмуватися незалежно для кожної цільової платформи рис.4.7. Таким чином, повторне використання коду, за статистичними дослідженнями компанії Xamarin, наближається до 85%.



Рис.4.7. Унікальний підхід до розробки Xamarin

Для вивчення крос-компільованих програм мобільний додаток WebUNLP може бути розроблений за допомогою Xamarin, інтегрованого з Microsoft Visual Studio. Дотримуючись найбільш ефективної стратегії роботи в цьому середовищі, необхідно створити унікальне рішення, яке містить три різні проекти. Один з них використовувався для створення програми Android, інший для програми Windows Phone 8 і третій для реалізації Portable Class Library (PCL) з усім спільним кодом. Три проекти розроблялися спільно та паралельно.

Хоча однією з найважливіших переваг Xamarin є повторне використання коду, на це сильно впливає тип розробленої програми. Співвідношення між складністю інтерфейсу та бізнес-логікою має безпосередній вплив на можливість підтримки найбільшої кількості спільно використовуваного коду в PCL. Однак

не варто недооцінювати перевагу використання однакової спільної мови, середовища та набору інструментів при розробці додатків для різних мобільних платформ.

#### **4.1.4.8 Програма, що створюється за допомогою перехресної компіляції WebUNLP за допомогою Delphi 10 Seattle**

З метою поглиблення вивчення крос-компільованих додатків було розглянуто аспекти створення мобільного додатку WebUNLP за допомогою Delphi 10 Seattle. Зокрема, як компоненти, надані Delphi для доступу до служб RESTful, так і середовище візуальної розробки Delphi 10 Seattle були дуже корисними.

Для отримання повідомлень на пристрій новин WebUNLP може використовуватися компонент TPushEvent, підключений до сервісу Kinvey.

#### **4.1.5 Висновки експерименту WebUNLP**

Вибір області для розробки експериментального випадку має свої особливості. Новизна наявності мобільного додатка не є достатньою підставою для виправдання такого розвитку. Необхідно задовольнити набір чітко сформульованих вимог.

WebUNLP – це віртуальне середовище для викладання та навчання, яке використовується різними курсами для студентів і аспірантів у Національному університеті Ла-Плати. Таким чином, кількість користувачів, які отримують вигоду від доступу до цього середовища з будь-якого місця, є значною.

Таким чином, було вирішено вибрати зазначений віртуальний простір для тиражування в мобільній версії, що не тільки дозволяє йому бути ближче до своїх користувачів, але й розширює його функції, в даному випадку спеціально для білборда.

Для розвитку цього експерименту було розглянуто розробку того самого додаток для п'яти типів мобільних підходів до розробки (веб, нативний, гібридний, інтерпретований і створений шляхом перехресної компіляції).

Виділяється величезна простота створення веб-версії, оскільки використовуються ті ж технологічні інструменти, що й для розробки будь-якого

традиційного веб-додатка. Основна відмінність в цьому плані полягає в обмеженні місця на екрані пристрою. Однак найбільший контраст полягає в тому, що неможливо отримати доступ до всіх апаратних можливостей пристрою, що може завадити реалізувати одну з вимог, мабуть, найцікавішу: сповіщення користувача про новини з білборду WebUNLP.

З іншого боку, рідні версії виконують всі вимоги. Хоча найбільшим недоліком є непереносимість, що передбачає конкретну розробку для платформи, яку ви хочете охопити. У цій роботі були представлені аналізи розробки для Android та iOS, найбільш використовуваних операційних систем, з більш високою вартістю розробки та обслуговування.

У випадку гібридного підходу як версія WebUNLP, яка розробляється за допомогою PhoneGap і JQuery Mobile, так і версія, яка створюється за допомогою Sencha Touch, зуміли поєднати простоту веб-розробки з використанням усіх можливостей пристрою. Цей тип підходу має на меті компенсувати недоліки двох попередніх підходів, що позиціонує його як дійсно цікавий вибір, який завжди обумовлюється специфічними вимогами, які необхідно виконати.

З боку інтерпретованого режиму розробки, версія WebUNLP, яка розробляється за допомогою Titanium, змогла вирішити всі вимоги, які були поставлені раніше. Як і в гібридному підході, варто зазначити, що розробка програми за допомогою Titanium є відносно простою і можна повторно використовувати весь кросплатформний код. Що робить його на перший погляд вибором, так це той факт, що ви отримуєте повністю рідні інтерфейси, чого не вдасться досягти за допомогою гібридного підходу.

Нарешті, версії WebUNLP, запропоновано створити за допомогою крос-компіляції з Xamarin і Delphi, які зможуть задовольнити зазначені вимоги. Підкреслюється той факт, що генеруються повністю власні інтерфейси, але я вважаю, що ціна дуже висока: інтерфейси повинні бути розроблені для кожної платформи окремо, а це означає, що весь вихідний код програми не можна використовувати повторно.



## **4.2 Підходи до розробки: Порівняльний аналіз ефективності**

### **4.2.1 Опис задачі**

Всього кілька десятиліть тому використання програмних систем було обмежено невеликою групою спеціалізованих користувачів. Той час різко контрастує з сьогоденням, коли смартфони, маленькі мобільні комп'ютери загального призначення, стали повсякденними та всюдисущими. Ці пристрої можна використовувати для виконання складних і критичних завдань, які вимагають постійного вдосконалення обчислювальних можливостей, доступності, ефективної продуктивності та інших потреб. Бурхлива еволюція цієї технології чинить тиск на адаптацію програмної інженерії.

Стосовно розробки для мобільних пристроїв, слід розглянути ряд характеристик цієї діяльності, яких не було в традиційній розробці програмного забезпечення. Необхідно врахувати, на якому типі пристроїв має виконуватися додаток, що збирається. Різноманітність платформ, мов програмування, засобів розробки, стандартів, протоколів і мережевих технологій; деякі обмеження певних пристроїв та вимоги ринку у часі, серед іншого, становлять деякі проблеми, які необхідно вирішувати.

У більшості випадків успіх програмного продукту для мобільних пристроїв буде залежати від рівня популярності, якого він досягає. Щоб максимізувати свою присутність на ринку, додаток повинен працювати на найбільшій кількості існуючих мобільних платформ, особливо на двох найпопулярніших: Android та iOS. Для досягнення цієї мети є два варіанти:

- 1) Розробляйте конкретні програми для кожної платформи, паралельно просуваючи різні проекти розробки, використовуючи інструменти та мови кожної з них. Ці програми, проаналізовані раніше в цій роботі, називаються нативними програмами.

- 2) Створення програм, які можуть запускатися безпосередньо на більш ніж одній платформі операційної системи з одного проекту розробки. Ці програми, називаються кросплатформними.

В останні роки зріс інтерес співтовариства програмної інженерії до

вивчення розвитку багатоплатформних додатків для мобільних пристроїв.

У [5] представлено порівняльний аналіз підходів до розробки кросплатформних додатків для мобільних пристроїв із запропонованою таксономією: мобільні веб-додатки, гібридні додатки, інтерпретовані програми та крос-компільовані програми.

У [7] та [8] проаналізовано загальні аспекти багатоплатформних фреймворків розробки для мобільних пристроїв.

У порівнюються нефункціональні аспекти для різних підходів до розробки кросплатформних додатків для мобільних пристроїв.

У [9] проаналізовано переваги та недоліки вищезгаданих багатоплатформних підходів до розробки з точки зору інженера-програміста.

Вибір режиму розробки мобільного додатка залежить від кількох факторів. Однією з них, у багатьох випадках важливою, є час виконання. Інтерес до оптимізації часу виконання програмного додатка притаманний дисципліні інформатики. Про це свідчить, серед іншого, еволюція обчислювальної потужності процесорів. Крім того, ефективна продуктивність є одним з атрибутів, яким програмні програми повинні задовольняти відповідно до різних стандартів якості, включаючи ISO/IEC 9126 і ISO/IEC 25010.

Коли справа доходить до розробки мобільних додатків, настав час розглянути час виконання програми з різних причин.

Час виконання програми тісно пов'язане зі споживання, яка обмежена ємністю батареї пристроїв.

Також продуктивність додатків в основному відображається через оцінки користувачів в інтернет-магазинах додатків. Погана продуктивність програм може призвести до незадоволених користувачів, створюючи негативну рекламу [60]. Андре Шарланд і Браян Леру визначають час виконання як одну з основних проблем, які необхідно вирішити при розробці кросплатформних додатків, і заявляють, що кінцеві користувачі стурбовані якістю програмного забезпечення та користувацьким досвідом.

Було проведено порівняльний аналіз продуктивності між нативними та



гібридними додатками, використовуючи фреймворк Phonegap для версії операційної системи Android.

Важливо підкреслити, що не було знайдено жодних досліджень, які б оцінювали та порівнювали продуктивність між різними мультиплатформними режимами розробки, відповідно до таксономії, запропонованої в [5]: мобільні веб-додатки, гібридні додатки, інтерпретовані програми та крос-компільовані програми. Ця таксономія є та, що використовується як посилання в цій роботі.

У цьому розділі виконується порівняльний аналіз продуктивності додатків для мобільних пристроїв, створених за допомогою нативного підходу до розробки та різних підходів багатоплатформної розробки відповідно [5].

#### **4.3 Порівняльний аналіз підходів до розробки мобільних додатків**

Як обговорювалося в попередніх розділах, існують різні підходи до розробки мобільних додатків.

Одним із варіантів є розробка нативних мобільних додатків, які використовують усі можливості мобільного пристрою та забезпечують високий рівень взаємодії з користувачем, оскільки користувацькі інтерфейси складаються з рідних компонентів, подібних до інших інтерфейсів операційної системи. Однак розробка не може бути повторно використана для підтримки інших сімейств операційних систем, що передбачає більш високі витрати на розробку та обслуговування.

На відміну від нативної розробки з'явилася кросплатформна розробка, що дає можливість розробити один продукт і запустити його на різних платформах. У цьому сенсі вивчалися різні підходи: мобільна, гібридна, інтерпретована та крос-компільована веб-розробка. Кожен із цих підходів має певні характеристики, які, природно, роблять їх різними, з перевагами та недоліками.

Було визначено кілька факторів, які можна використовувати для аналізу додатки, створені з використанням власних і кросплатформних підходів до розробки, які докладно описані нижче:

- Досвід користувача: сукупність факторів та елементів, які стосуються рівня загального задоволення користувачів під час використання продукту чи



системи. Результатом є формування позитивного чи негативного сприйняття зазначеної послуги, продукту чи пристрою.

- Розглянуті платформи: у попередніх розділах було показано, що існують різні сімейства операційних систем. Залежно від обраного режиму розробки програми можуть бути розроблені для певної платформи або для кількох.

- Вартість розробки: розробка може бути більшою чи меншою, залежно від того, чи потрібні рішення для кодування спеціально для кожної операційної системи, чи можливе повторне використання коду.

- Вартість обслуговування: для виправлення помилок або додавання нових функцій може знадобитися кодування спеціально для кожної операційної системи. Крім того, необхідно враховувати, чи повинні співіснувати різні версії одного і того ж продукту

- Інтегроване середовище розробки: програмне забезпечення, яке допомагає програмісту створювати програми.

- Інтерфейси користувача: фактор, який аналізує типи компонентів, що використовуються для створення інтерфейсів користувача. Як згадувалося під час аналізу різних підходів до розробки, вибір інструменту для використання впливає на те, чи будуть інтерфейси складатися з використанням рідних компонентів або веб-компонентів, які можна прикрасити для імітації рідних компонентів.

- Повний доступ до пристрою: можливість доступу до всіх можливостей пристрою, таких як камера, датчики, серед іншого, за допомогою інструментів розробки.

- Режим встановлення: стосується того, чи можна встановити програму з магазинів програм, чи доступна вона без встановлення з Інтернет-браузера.

- Продуктивність: коефіцієнт, який аналізує виконання завдань і пов'язаний час у вашій резолюції.

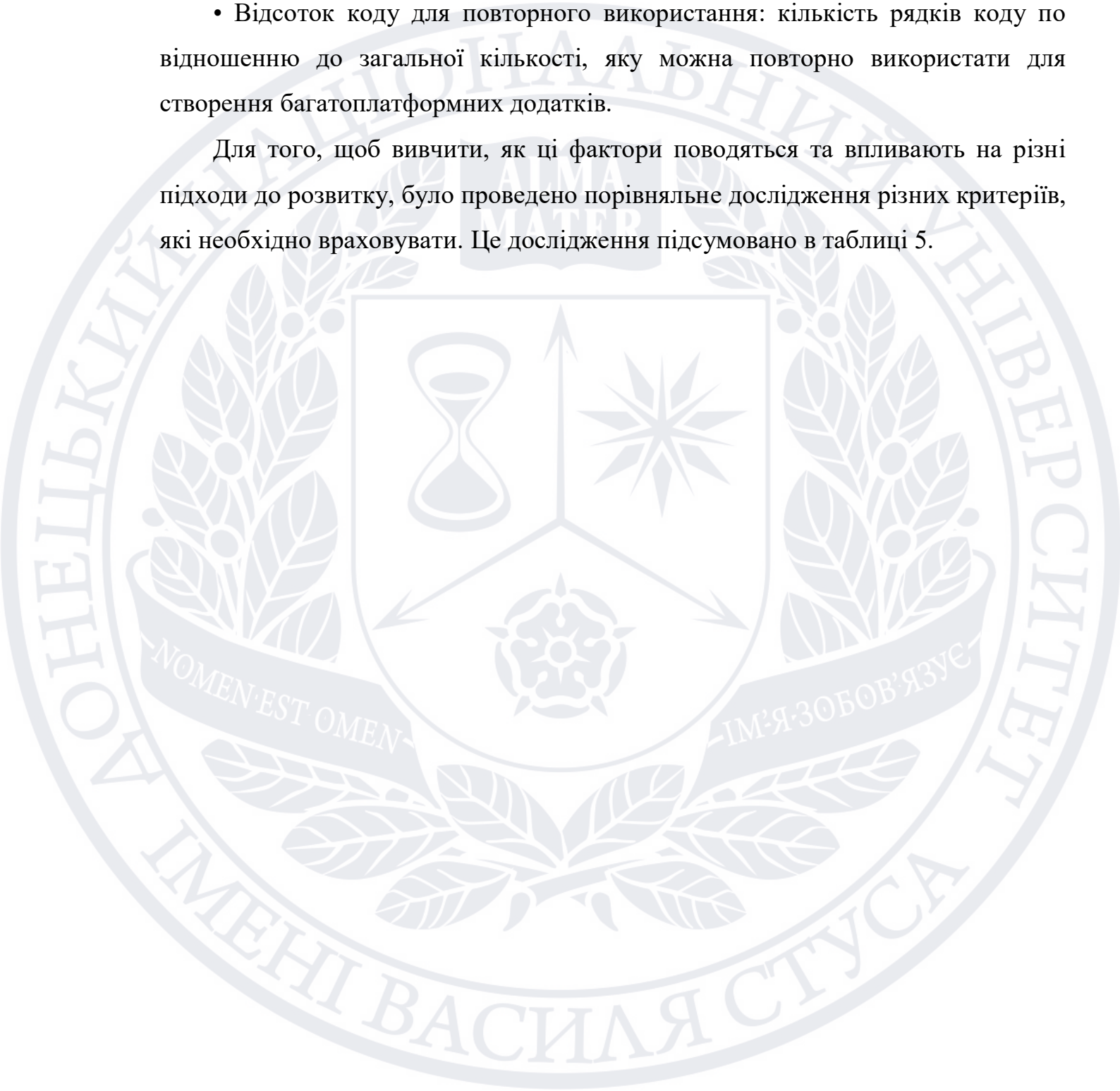
- Використання в автономному режимі: можливість роботи програми без підключення до Інтернету.

- Розповсюдження через магазини: можливість завантажити програму з

магазинів додатків.

- Категорії програм, які будуть розроблені: тип програми, яка буде розроблена.
- Відсоток коду для повторного використання: кількість рядків коду по відношенню до загальної кількості, яку можна повторно використати для створення багатоплатформних додатків.

Для того, щоб вивчити, як ці фактори поведуться та впливають на різні підходи до розвитку, було проведено порівняльне дослідження різних критеріїв, які необхідно враховувати. Це дослідження підсумовано в таблиці 5.



Таблиця 4.4: Порівняльний аналіз підходів до розробки мобільних

Фактор для аналізу	Нативна розробка	Веб-розробка	Гібридний розвиток	Інтерпретовани й розвиток	Створений шляхом перехресної компіляції
Досвід Користувача	Високий	Дуже низький	Низький	Високий	Високий
Досягнуті платформи	Розроблено виключно для цільової платформи	Будь-яка платформа. Все, що вам потрібно, це браузер і доступ до нього Інтернет	Можлива компіляція для Android, iOS, Windows Phone	Можлива компіляція для Android, iOS, Windows Phone	Можлива компіляція для Android, iOS, Windows Phone
Вартість розробки	Дуже висока	Дуже низька	Низька	Середня	Від середнього до високого
Вартість обслуговува	Дуже висока	Дуже низька	Середня	Висока	Від високого до дуже високого



ння					
Інтегроване середовище розробки	Одна IDE для кожної ОС	Кілька варіантів	Загальні плагіни IDE	Деякі рамки мати певну IDE	Одна IDE для кожного фреймворка
інтерфейси Ім'я користувача	Nativas	Web	Web	Nativas	Nativas
Повний доступ до пристрою	Так	Ні	Так	Так	Так
Режим об'єкта	Завантажити з	Немає зручностей	Завантажити з	Завантажити з	Завантажити з

Фактор для аналізу	Нативна розробка	Веб- розробка	Гібридний розвиток	Інтерпретова ний розвиток	Створений розвиток шляхом перехресної компіляції
	магазин додатків і встановле ння		магазин додатків і встановлен ня	магазин додатків і встановленн я	магазин додатків і встановленн я
Продукти вність	Дуже висока	Дуже низька	Від низької до середньої	Висока	Від середньої до високої
Usa offline	Так	Ні	Так	Так	Так
використа ння в автономному	Так	Ні	Так	Так	Так

режимі					
Категорія додатків для розробки	Програми, які вимагають багато ресурсів або які потребують високого досвіду користувача Ім'я користувача	Існуючі веб-сайти	Веб-сайти, інкапсульовані до як програми, які розповсюджуються через магазини	Від простих до складних додатків	Від простих до складних додатків
Відсоток від повторного використання	0%	100%	Близько 100%	Близько 90%	Між 60 у 85%

Таблиця.4.5 Порівняльний аналіз підходів до розробки мобільних додатків



## ВИСНОВКИ

Спочатку мобільні пристрої були продумані та розроблені з певною метою. З роками технологічне зростання дозволило впровадити додаткові функції, що дозволило розширити рамки використання.

Сьогодні обчислювальні потужності, що лежать в основі широкого спектру мобільних пристроїв, створили нові можливості, кидаючи виклик інженерам-програмістам.

Так само, як згадувалося в попередніх розділах, існують різні операційні системи для мобільних пристроїв, найпопулярнішими є Android, iOS і Windows Phone. У зв'язку з тим, що успіх програмного продукту багато в чому пов'язаний з його масовим використанням, виникає необхідність розробки мобільних додатків з урахуванням кожної з цих платформ.

Розробка програмного забезпечення для мобільних пристроїв має свої особливості, які відрізняються від традиційної розробки. Ця нещодавня діяльність, немислима десятиліття тому, знаменує собою до і після еволюції програмної інженерії. Слід очікувати, що ця еволюція триватиме, а отже, підходи до розробки мобільних пристроїв зміняться або будуть замінені іншими. Зрозуміло, що на даний момент існує п'ять можливостей для розробки одного мобільного додатка:

- Нативні програми
- Веб-додатки
- Гібридні програми
- Інтерпретовані додатки
- Програми, створені шляхом перехресної компіляції.

Для повного аналізу кожного підходу до розробки було розроблено серію тематичних досліджень. Перший з них полягав у аналізі розробки мобільного додатка для кожного представленого підходу до розробки. Визначивши цю першу мету, було вирішено розширити віртуальне середовище викладання та

навчання WebUNLP за допомогою мобільного додатка, зосередженого на бюлетені новин курсів.

Додаток телефон повинен має дозволити вхід студентам і викладачам з тими ж обліковими даними, що використовуються у веб-версії, і показувати інформацію, синхронізовану з Інтернетом, про новини, опубліковані на білбордах курсу. Крім того, коли викладачі публікують новини, студенти, які належать до курсу, повинні отримувати сповіщення на свій мобільний пристрій.

Після розробки мобільних додатків для кожного підходу до розробки можна зробити деякі висновки.

Виділяється простота створення мобільної веб-версії, але спостерігаються обмеження, оскільки неможливо отримати доступ до всіх апаратних можливостей пристрою.

Розробка нативних додатків є оптимальною в порівнянні з вимогами, встановленими вчасно, з недоліком, що додатки не є переносними, що передбачає специфічні розробки для кожної платформи, яку потрібно охопити. З іншого боку, розробка гібридних додатків – за допомогою PhoneGap і Sencha – зуміла поєднати простоту веб-розробки з використанням усіх можливостей пристрою, позиціонуючи його як дійсно цікавий вибір. Таким же чином інтерпретований додаток, який розробляється за допомогою Titanium, може вирішити всі вимоги, які були поставлені раніше; підкреслюється, що розробка є відносно простою і можна повторно використовувати весь код між платформами. Факт отримання повністю рідних інтерфейсів позиціонує його як ідеальну альтернативу. Нарешті, програми, створені за допомогою підходу крос-компіляції (Delphi та Xamarin), можуть задовольнити зазначені вимоги; Отримання повністю рідних інтерфейсів — це те, що слід виділити, але воно коштує високої вартості, оскільки інтерфейси повинні бути розроблені для кожної платформи окремо.

Нарешті, було проведено порівняльний аналіз властивостей кожного підходу до розробки. Були визначені певні фактори, що представляють інтерес, які спільнота розробників програмного забезпечення враховує при виборі

технології, а також проаналізовано їх вплив на кожен підхід до розробки.

### СПИСОК ПОСИЛАНЬ

1. A. Talukder, H. Ahmed, and R. Yavagal, Mobile Computing, Technology, Applications, and Service Creation, Tata McGraw-Hill, Ed., 2010.
2. Розробка веб-додатків, мобільних додатків та порталів. URL: <https://ittel.com.ua/informacijni-texnologiyi/rozrobka-mobilnih-dodatktiv/> (дата звернення: 15.03.2022).
3. P. Abrahamsson, "Mobile software development - the business opportunity of today," in Proceedings of the International Conference on Software Development, Reykjavik, 2005.
4. Особливості тестування мобільних додатків. URL: <https://www.quality-assurance-group.com/osoblyvosti-testuvannya-mobilnyh-dodatktiv/> (дата звернення: 20.03.2022).
5. Що таке інтегроване середовище IDE? URL: (20.03.2022) <https://uk.theastrologypage.com/integrated-development-environment>
6. Архітектура та проектування програмного забезпечення URL: <http://dspace.wunu.edu.ua/jspui/bitstream/316497/24194/1/%D0%BE%D0%BF%D0%BE%D1%80%D0%BD%D0%B8%D0%B9%20%D0%BA%D0%BE%D0%BD%D1%81%D0%BF%D0%B5%D0%BA%D1%82%20%D0%BB%D0%B5%D0%BA%D1%86%D1%96%D0%B9.pdf> (21.03.2022)
7. Міньков, Костянтин Павлович. Система захисту мікросервісних систем. MS thesis. Київ, 2019.
8. Кочкар'юв, Сергій Вадимович. Проектування та розроблення WEB-додатків на платформі систем контролювання доступу "Intteks Acs". BS thesis. КПІ ім. Ігоря Сікорського, 2020.
9. Кравченко, Євгеній Олександрович. "Сучасні інформаційні технології та їх використання." (2021).
10. Telam. [Online]. <http://www.telam.com.ar/notas/201504/102073-telefoniamovillineas-activas.html>
11. RTVE. [Online]. <http://www.rtve.es/noticias/20110212/evolucion-del-telefonomovil-del-zapatofono-smartphones/404523.shtml>
12. Vodafone. [Online] <https://www.vodafone.ua/>
13. Muy Canal. [Online]. <http://www.muycanal.com/2014/01/31/futuro-del-telefonomovil>



14. Statcounter. [Online]. <http://gs.statcounter.com/>
15. Яцків, Н. Г. "Мобільні інформаційні системи." (2016).
16. Statista. [Online]. <http://www.statista.com/>
17. Полтавська, Д. С. "Експериментальне дослідження можливості застосування фонових джерел світлової енергії для енергопостачання мобільних пристроїв." (2020).
18. Штогрин, Павло Петрович. Мобільний додаток для моніторингу та прогнозування погодних умов у реальному часі. BS thesis. КПІ ім. Ігоря Сікорського, 2020.
19. Abhinay Puvvala, Amitava Dutta, Rahul Roy, and Priya Seetharaman, "Mobile Application Developers' Platform Choice Model," in 2016 49th Hawaii International Conference on System Sciences (HICSS), Koloa, Hawaii, 2016.
20. Android Dashboard. [Online]. <https://developer.android.com/about/dashboards/index.html>
21. Петрішенко, Сергій Олександрович. "Порівняльний аналіз Swift і Objective-c." Міжнародний науковий журнал 6 (1) (2016): 78-80.
22. Слабов, Д. В. "Дослідження вразливостей мобільних додатків корпоративної мережі." (2021).
23. Копельчук, Сергій Володимирович. Створення інтернет-магазину на MVC фреймворку WP Emegre. Diss. ОДЕКУ, 2018.
24. Bootstrap. [Online]. <http://getbootstrap.com/>
25. Foundation. [Online]. <http://foundation.zurb.com/>
26. PhoneGap. [Online]. <http://phonegap.com/>
27. Apache Cordova. [Online]. <https://cordova.apache.org/>
28. A Kosmaczewski, Sencha Touch 2 Up and Running.: O'Reilly, 2013.
29. Регета, Д. М. "Розробка програмного забезпечення для відтворення файлів мультимедіа для Windows Phone." (2014).
30. Miller, Mark A., et al. "A RESTful API for access to phylogenetic tools via the CIPRES science gateway." Evolutionary Bioinformatics 11 (2015): EBO-S21501.
31. iOS Human Interface Guidelines. [Online]. <https://developer.apple.com/ios/humaninterface-guidelines/overview/design-principles/>

32. Mark Norm Norman Francis Christian Heilmann, Web Development Solutions: Ajax, APIs, Libraries, and Hosted Services Made Easy.: Apress, 2007.



Декларація щодо унікальності текстів роботи  
та невикористання матеріалів інших авторів без посилань

Доценко Валерія Вікторівна

Прізвище, ім'я, по батькові

Факультет інформаційних і прикладних технологій

Факультет

122 Комп'ютерні науки

Шифр і назва спеціальності

Сучасні інформаційні технології та програмування

Освітня програма

**ДЕКЛАРАЦІЯ**

Усвідомлюючи свою відповідальність за надання неправдивої інформації, стверджую, що подана кваліфікаційна (бакалаврська) робота на тему: «РОЗРОБКА ТА ОПТИМІЗАЦІЯ УНІВЕРСАЛЬНИХ ВЕБ-ІНТЕРФЕЙСІВ ДЛЯ МОБІЛЬНИХ ДОДАТКІВ» є написаною мною особисто.

Одночасно заявляю, що ця робота:

- не передавалась іншим особам і подається до захисту вперше;
- не порушує авторських та суміжних прав, закріплених статтями 21-25 Закону України «Про авторське право та суміжні права»;
- не отримувались іншими особами, а також дані та інформація не отримувались у недозволений спосіб.

Я усвідомлюю, що у разі порушення цього порядку моя кваліфікаційна (бакалаврська) робота буде відхилена без права її захисту, або під час захисту за неї буде поставлена оцінка «незадовільно».

дата

підпис