

University of Buea

Faculty of Engineering and
Technology

Department of Computer
Engineering



Universite de Buea

Faculte d'ingenieurie et de
Technologie

Genie Informatique

CEF 440: INTERNET AND MOBILE PROGRAMMING

A REVIEW ON MOBILE PROGRAMMING: APPLICATIONS, FRAMEWORKS, PROGRAMMING LANGUAGES, REQUIREMENT ANALYSIS AND COST EVALUATION

- TENDONGFAC GEORGE NJIMO FE20A116
- DONGHO NONGNI GUIROLE FRANCIS FE20A230
- ARVELLA CHRISTIE PETTOUN TCHOUAKWE
FE20A011
- MBUNGAI MICHAEL BERNARD FOMO FE20A064
- NGUEMGNE FOTSO VINY ANGELE FE20A075

Supervisor:

Dr. NKEMENI Valery

TABLE OF CONTENTS

I. MOBILE APPLICATIONS	
Definition.....	3
Types of Mobile Apps.....	3
Differences between Native, Web, Hybrid and Apps.....	6
II. MOBILE PROGRAMMING LANGUAGES	
III. MOBILE APP DEVELOPMENT FRAMEWORKS	
Advantages and Disadvantages of Mobile Frameworks.....	10
Famous Mobile Application Frameworks Comparison.....	12
IV. COLLECTION AND ANALYSIS OF REQUIREMENTS FOR MOBILE APPS	
Collection of Requirements.....	13
Analysis of Requirements.....	13
V. MOBILE APP DEVELOPMENT COST	
What affects the cost?	15
App specification.....	15
Estimation Process.....	16
Price Models.....	17
Conclusion.....	18

MOBILE APPLICATIONS

Definition

Mobile application software is an app that runs on a smartphone or tablet and is characterized by its portability. Built and developed specifically for small devices such as smartphones and tablets, they have limited capacity compared to desktop apps.

They work the same as business apps. They use specific programming languages to perform a task or solve a problem for the end user.

Types of Mobile Applications

Members of the mobile app development community agree that there are three main types of mobile apps which are: native apps, web apps, and hybrid apps.

1. Native Mobile Apps

They work on only one specific mobile platform or operating system(OS). Native applications for Google Android OS can only work on Android devices. Similarly, a native app on Apple iOS can only work on iPhones and iPads.

To build apps on specific OS, app developers use various programming languages. The most commonly used are: **Objective-C, Java, Kotlin and Swift.**

Advantages of Native Apps

- **Better performance:** Native apps are very fast and responsive because they are built for a specific OS and are compiled using the platform's core programming languages and API. As a result, the app is much more efficient.
- **More secure:** Since native apps are built for a specific platform, the data is encrypted within a single infrastructure, which significantly reduces security-associated risks.
- **User friendliness:** A native app is directly linked to the device OS. As a result, it is equipped with different functionalities such as camera, GPS, calendar and microphone. Native apps effectively use these features to offer enhanced user experience.
- **Availability:** Users can easily find native apps from App Store and Play Store. Even afterwards the apps can be downloaded easily with just one click.
- **Instant Updating:** When Android and iOS roll out software upgrades, IT teams can immediately implement the latest features since they've

got quick and easy access to new SDKs that help modify the application.

Disadvantages of Native Apps

- **Cost:** The overall cost involved in the development and maintenance of a native app is considerably high. This is due to the fact that there should be separate versions of the same application.
- **Time Consumption:** Since native apps are developed for multiple platforms, it requires more time. They may require significant amount of time for making compared to their counterparts.
- **Development:** Developing a native app is a difficult process since separate developers are needed for each platform.

Examples of Native Apps are: **Spotify, WhatsApp, LinkedIn, Facebook.**

2. Web Apps

Web apps use a web browser as their UI, thus they need an internet connection. They can run on desktop computers, smartphones, and tablets with web browsers with no need of installing them on the hardware.

The most commonly used programming languages used are: **CSS, HTML5, JavaScript.**

Advantages of Web Apps

- **Cost-effective:** It simply consists of creating links between the RL and the application, and since that is relatively simpler to do, it takes less time for development. Thus, making it overall a cost-effective affair for the owner.
- **Free from downloading needs:** Since by using a web browser a user can directly interact with the app, these do not have to be installed or downloaded directly from different platforms.
- **Runs easy:** They are designed such that, as long as the web browser is in place, their interface with various screen sizes allows for them to easily adapt to iOS, Android, or windows.

Disadvantages of Web Apps

- **Internet reliance:** A reliable internet is a must at all times to browse through the website and run the app.
- **Website dependency:** A web app is completely based on its web browser. If the website happens to fail or go unresponsive, the app fails to function too.
- **Restricted Functionality:** As web apps are not native, they cannot sometimes effectively collaborate with all the hardware and OS of the specific device being used.

Examples of Web Apps are: **Amazon, Canva, Google Docs, Microsoft Office 365.**

3. Hybrid Apps

A hybrid mobile app combines the elements of a native app(an application developed for a specific platform like Android or iOS) and a web app(an app that can be accessed on the internet via a browser).

Unlike native apps, hybrid apps have cross-platform compatibility.

The programming languages used are: **CSS, HTML5, Ionic, JavaScript, Objective-C, python, swift.**

Advantages of Hybrid Apps

- **Less development time:** Cross-platform frameworks are the quickest way to bring a product to life and ready for the google play and Apple App stores. It allows the app to be in the hands of every user faster.
- **Lower cost:** Less development time = fewer billable hours. As long as hybrid apps take few hours to create, they will remain cheaper than building native apps.
- **Wider audience:** Once a hybrid app is good to go, it can live on Android and iOS app stores and be used by anyone who is interested, instead of restricting the audience.
- **Easier bug fixes and maintenance:** Hybrid apps let you send out one patch and bug fix to repair issues across all devices. This is easier than solving the problem for iOS and Android each.

Disadvantages of Hybrid Apps

- **Operating Slower:** Hybrid apps are relatively slower as they add a layer between the source code and the mobile target platform. They usually have longer launch times and more crashes. The UI also might feel sluggish if the app involves vast amount of data manipulations.
- **Longer testing process:** A hybrid app has to get the green light for multiple platforms. This means a longer testing process to make sure coding bugs are rolled out to all platforms and the apps are as close to perfect as possible.
- **Less complex functionality:** If you have a complex app that requires high performance or one that relies heavily on 3D graphics and design, then it could be worth the time to forgo hybrid apps, as they don't compete as well in this category as native apps.

Examples of Hybrid Apps are: **Instagram, Twitter, Uber, Facebook, and Yelp.**

Differences between Web Apps, Native Apps, and Hybrid Apps

Characteristics	Web App	Hybrid App	Native App
Usage	Users can access directly from a browser	Users have to install the app on their device of choice	Users have to install the app on their device of choice
Access	Limited by browser and network connectivity	One-step access with offline features	One-step access with offline features
Performance	Slower and less responsive	Faster but may consume more battery power	Performance can be optimized to device
Development	Cost-efficient, faster time to market	Cost-efficient, faster time to market	Expensive, slower time to market
User experience	Inconsistent and dependent on the browser being used	Consistent and engaging	Consistent and engaging
Internal working	Client code in the browser communicates with remote server-side code and databases	Client code and browser code wrapped in a native shell or container	Client code written in technology and language specific to the device or platform it will be installed on.

MOBILE PROGRAMMING LANGUAGES

Mobile programming, also known as **mobile application development**, is the process of creating software solutions (applications) that run on a mobile device. The specificity of mobile programming is that it takes advantage of the specific device features for which it is conceived. For example a health app will take advantage of a smart watch's temperature sensor or a game will take advantage of the phones accelerometer.

With the rapid expansion of technology we have witnessed the birth of a plethora of programming languages all having strong points and setbacks. This now brings the question of what programming language is best for a given project.

In this review we have selected some criteria that should be considered in the selection of a programming language for a project:

1. Type of application

The first thing to consider is the type of application you want to build as this factor will definitely be a major decisive factor in the decision of what programming language to use.

2. Identifying the framework to be used

The framework your app will be running on is an equally important consideration in choosing the right programming language for the project.

3. Maintainability

You should consider the long-term maintainability of the project you are embarking on. The language that you choose should be up to speed on the current programming technology. It should be noted that no matter the programming language used, if the app is not built in a conventional manner, maintenance could be a nightmare.

4. Scalability

As with any project, the plan with a mobile application is to grow and become more popular. Your app will probably therefore attract more users over time. As this happens, your app should be able to handle the increasing influx of data. This feature greatly depends on the framework that is used.

5. Security

Whatever the kind of application, security is of significant importance, and the programming language you use plays a vital role in protecting the user data.

6. Community support

The size of the community of developers using a particular technology is a great indicator of the effectiveness of a programming language. This will greatly influence how fast you can get help to solve a problem if you face one (and trust me you will). So you might want to look out for this in the choice of your programming language.

Programming language	Type of app	Supported framework	Maintainability	Scalability	Security	Community
Java	-Hybrid	-ATG -React -Bootstrap -jQuery -Xamarin	Maintainable	High scalability	High security features	Large community support
Python	-Web based -Hybrid	-Kivy -PyMob -Beeware -Django	Highly maintainable	High scalability	High level of security	Large community support
Objective-C	-Native iOS	Foundation	Highly maintainable	High scalability	High level of security	Not very large community of users
Kotlin	-Native Android	-Android studio	Easily maintainable	Easily scalable	Secured	Large community
Dart	-Hybrid	-Flutter	Easily maintainable	Easily scalable	Secured	Large community support
C#	-Hybrid	-Xamarin	Easily maintainable	Good degree of scalability	Secured	
HTML, CSS, JavaScript	-Web based	-Apache Cordova	Easily maintainable	Easily scalable	Secured	Large community of users
Swift	-Native (iOS)	-Cocoa -Cocoa Touch -SwiftUI	Highly maintainable	High scalability	Secured	Not very large community

MOBILE APP DEVELOPMENT FRAMEWORKS

A mobile app framework is a software creation platform that includes tools and templates, compilers, debugging tools, interfaces among other things that is used to design mobile applications or support mobile app development. Thus, a developer creates the application's source code and uses the framework to generate the application for the mobile devices. Frameworks support different programming languages and are divided into three categories which are:

- **Native Application Frameworks:** These are frameworks used to generate mobile applications meant for a single operating system. They can be IOS, Android, or windows, but they will be developed for a specific Operating system and specific purposes. You can use various programming tools and languages according to your requirements and for which OS you want to develop your mobile application. Example **Android Studio** which is used to develop android apps.
- **Web Application frameworks:** Web applications are those applications that take data from websites but they look like native mobile applications, web app development framework is a set of tools that are used to develop and maintain web applications. Example **JQuery mobile, WordPress**.
- **Cross-platform application frameworks:** Cross-platform apps are those applications that can be run on windows, mac, android, and IOS with a single code source. They operate based on the WORA principle Write Once Run Anywhere. You can develop one application using the cross-platform framework and then convert it to different platforms without changing the whole code. Example **Ionic, Xamarin, React Native, flutter**.

If you are running a business then a cross-platform app development framework is necessary for you because it helps you reach a maximum audience, and reduces the cost and time of developing native apps for different platforms.

Advantages and Disadvantages of mobile Frameworks

a) Native application frameworks

➤ Advantages

- The design process is simplified because of the services and support of offered by the Operating System.
- They produce very fast applications because they are built for a specific operating system.

➤ Disadvantages

- Single platform development focus would result in losing 50% of potential users.
- Development time is high because it would require the application source to be recreated from scratch across each individual platform.

b) Web application frameworks

➤ Advantages

- Ease debugging and Application maintenance
- Security is reinforced
- Code length reduced or inexistent

➤ Disadvantages

- Slow performance
- Learning the Framework and not the language itself
- Lack of community Support.

c) Cross platform application frameworks

➤ Advantages






- Very fast deployment time

- Reusable code with the WORA (write once run anywhere) principle
- Simplified maintenance

➤ **Disadvantages**

- Possible design mismatch on the various platforms
- Limited User experience

Famous Mobile Application Frameworks Comparison

		 <i>React</i>	 Android Studio	 Xamarin	
Programming Languages	Dart	JavaScript	Java Kotlin	C#	Javascript Html Css
Performance	Amazing	Good	Good	Moderate	Amazing
Deployment Time	Very short	short	long	short	short
Cost	Open Source + Free	Open source +Free +paid plans	Not Open Source + Free	Open Source + Purchased	Opens source + Purchased
User-friendliness	Very User-friendly	Complicated	User-friendly	User-friendly	Very User-friendly
Community support	Yes, Very Large	Yes	Yes	Yes	Yes
Category	Cross Platform Framework	Cross platform Framework	Native Application Framework	Cross platform Framework	Web application Framework
Platforms Supported	Android Jelly Bean iOS 8+	Android 4.1+ iOS 8+	All android Versions	Android 4.0.3 + iOS 8+	N/A
Code Reusability	90% Reusable	90% Reusable	Not Reusable	96% Reusable	Not Reusable
Application Example	eBay	Instagram Uberbeats	Microsoft Office Mobile	Olo Storyo	Babcares.com Bubble.io

COLLECTION AND ANALYSIS OF REQUIREMENTS FOR MOBILE APPS

Collection of Requirements

The requirements of a mobile application are collected in a step by step process that is, it follows a chronological order so as to give a more concrete shape to the owner's business idea with a desired output. The collection and analysis aim to produce an important document known as the **Software Requirement Specification (SRS)**.

These steps are as follows:

- **Clear idea description:** The idea is meant to tackle or solve a problem and the solution to the problem should be clear enough with no ambiguity that is the development team should be able to understand the purpose, usability and whom to target.
- **Determine basic navigation patterns:** The development team uses creativity and imagination to devise all possible actions and sequences that users will take while using the app such as logging in, errors made by the users.
- **Collect market information:** Detailed examination of competitors as well as existing applications to work on different approaches towards a target audience such as if they prefer iOS or Android phones, which gender is prevalent, what is the average age, income level.
- **Choose what features an application should have:** After brainstorming on the different features of the application, to avoid redundancy the “**MoSCoW**” approach can be used which says that **all characteristics can be categorized in four groups that is Most, Should, Could and Won't**. It means that only important features related to the idea should be used.
- **Prepare a functional specification:** This section comprises of use case diagrams as well as class diagrams scenarios of the accepted features after using the “**MoSCoW**” approach. The various case scenarios should be related to each other with clearly stated attributes.
- **Provide wireframes to complement the text:** Wireframe refers to rough work representation of the different interfaces showing the flow between the screen and the functional representation informed by the business idea showing all the features.

Analysis of Requirements

The analysis of the requirements for a mobile application are done in two phases that is: **functional and non-functional requirements** together make up a document known as **Software Requirements Specification (SRS)**.

- **Non-functional requirements**

Non-functional requirements refer to the overall properties of the system meaning how the system goes about executing specific functions. Non-functional requirements do not have an impact on the functionality of the program; However, it affects the performance of the system. Non-functional requirements include the following:

- ✓ **Speed:** Speed here simply refers to how fast is the system in executing certain user activities such as time taken for a page to load, speed within which certain requests must be met.
- ✓ **Availability:** It answers to how often is the system accessible to users that is does it operate every day or it is seasonal.
- ✓ **Scalability:** This requirement tells us how many users the system can handle and also try to accommodate more users as time goes by.
- ✓ **User-friendly:** When new users try to access the app's features is the process smooth or the user finds difficulty in accessing any facility or service pertaining to the app.

- **Functional requirements**

Functional requirements defines what a product must do, that is, the functions of the features in order to meet with the user's needs or requirements. It focuses on the practical use of software from the point of view of the user: It's features, performance, ease of use, absence of defects.

For Example mobile banking app a user should be able to create an account, withdraw money, deposit money, check the balance.

When the user creates an account there should be a mechanism that can verify the user's authentication. Also in case the user wants to withdraw a certain amount of money there should be a checking process to ensure that only the owner of the account has access to the account.

In addition, in a situation where the user wants to check his/her account balance, only him/her can see the account balance and the balance should be correct with some accountability.

MOBILE APP DEVELOPMENT COST

What affects the cost?

The price of building an app depends highly on its functionality and the wages of those who are going to build it. Every feature takes a number of hours to program, and the more complex the app, the higher will be the price of building it and maintaining it after release.

- **Features:** Every feature of an app has a more or less fixed price, which depends on the number of hours it takes to program it. The more complex the feature, the more expensive it is going to be.
- **Supported platforms:** If you plan to develop an app specifically for one platform, native development is the best choice in terms of performance, stability and price. However, if you want your app to be present in both Playstore and App Store, consider building the app with a cross-platform framework that is, a hybrid app.
- **Maintenance:** Every app requires updates to maintain complete compatibility with new devices and OS, as well as fixing bugs (there are always some). Usually, these expenses amount to about 15% of the initial development cost per year. Also, since almost every app stores user data, you will have to rent a secure server or cloud storage.
- **Development team size:** How many people develop your app should be determined according to the project complexity. Generally, a common setup includes:
 - 2 developers (iOS and Android or cross-platform).
 - QA (Quality Assurance) Engineer reviews all specifications and technical documents, as well as creates and plans all testing activities.
 - Backend developer oversees data storage, payment systems and logic of app operation.
 - UX/UI Designer creates a user-friendly interface for the app.
 - Project-manager manages the development process and communicates directly with the client.

App Specification

Before estimating mobile app development, you'll need to review all ideas and concepts and turn them into a detailed technical specification. The more detailed it is, the easier it will be to assess the costs. Software requirements specification usually include:

- **Overview:** This part briefly describes the product and the app's concept. A list of all app features is made along with a list of type users, their roles and classes. An overview also includes information on hardware and software

requirements of the app, design & implementation constraints and a list of user documents to be released.

- **Functional requirements:** This section contains a detailed description of all app's features, which contains: purpose, how a feature works (what triggers it, what it does), priority of implementation and use cases.
- **External interfaces:** This part focuses on all types of interfaces of the app.
 - ✓ **Screens (user interfaces).** Prototypes and descriptions of each app's screen.
 - ✓ **Software Interfaces.** Specifications of APIs, operating systems, database, third-party software.
 - ✓ **Hardware interfaces.** Technical characteristics of how the app interacts with the device's hardware. How it can be controlled via physical controls, what communication protocols are used and differences depending on a device model.
 - ✓ **Communication interfaces.** Descriptions of communication features of the app (email, browser, messaging) and server exchange protocols. Configuration of the app's communication with an external server.
 - ✓ **Other parameters and their specifications.** Usage of geographic data, push notifications, in-app purchases, data security, performance measurement.

Estimation process

When you have the technical specification on hand, it's time for estimation. To be accurate, the process should involve several stages, each of them clarifying development requirements.

Stage 1. Breakdown

First, the provided technical specifications are decomposed into small and easily estimated parts of the app. Usually, the breakdown is done by screens or by functions.

- **By screens.** A wireframe of every screen gets analyzed for present functions (login, record audio) and UI elements (buttons, text fields).
- **By functions.** Every function of the app is analyzed: how a function works, its design in the UI and on what screens it is present.

Then, all features or screens are sorted by priority: must-have items, serving the app's main purpose; 'cool to have' features and low-priority items.

Stage 2. Developer estimation

During the next step, developers assess how long it will take to program each function of the app and present the assessment in the form of a range (minimum to maximum) of hours of development.

Stage 3. Project manager estimation

It is common for developers to slightly overestimate the required amount of hours to develop, and, at this stage, the main priority of the project manager is to find the balance between price and quality. The PM reviews the development team estimates and includes the hours required for design, internal and external communications (team meetings, client demonstrations), focus group tests, code testing and bug fixing. Also, such processes as code merge between developers, refactoring and release are taken into account at this stage.

Stage 4. Final estimation

The project manager confirms his estimation with other members of the team, makes necessary corrections and presents the results to you.

Sometimes the mobile app development cost estimate template might look like an exact cost and time breakdown:

<u>Feature</u>	<u>Time (hours)</u>	<u>Cost (USD)</u>
User login	20	500 to 1000
Push notification	20 to 200	1000 to 10,000
Geolocation	50	1250 to 2500
Chat/messaging	80+	2000 to 4000
Payment integration	50+	2500 to 5000
Streaming videos	30+	1500 or more
Offline mode	40+	1200 to 2000
Phone sensor usage	10+	250 to 500
Search	10+	250 to 500
Data encryption	20	500 to 1000

Price models

When estimating app development, two pricing models are commonly used, that are fixed-price and time & material models.

- **Fixed price:** With a fixed price model, a precise list of features, their realization and costs are determined during the estimation. If any additional works occur during development, they come at an extra cost and require a full agreement process with the client, which leads to a development team downtime, and further increase in the cost and time of development. If any planned works turn out to be unnecessary, the final price does not change for the client. This way, the fixed price model lacks flexibility, bears large

financial and time risks, and can be suitable only for small and short-term projects.

- **Time & material:** Time & material is a flexible model, which is focused not on building the exact list of features, but rather on the end result. The development process is done with monthly reporting and developers are allowed to build additional features without a full agreement process.
 - ✓ The development team provides a rough top-level estimation divided into stages (sprints). The client agrees on the scale of works.
 - ✓ The team starts building the app, presenting the work done to the client at the end of every sprint. The client pays only for what was done during the sprint.
 - ✓ If necessary, the team suggests additional works and, upon the client's agreement, includes them in the next sprint.
 - ✓ In the end, the client receives the finished product, adapted to the current technical requirements and market demands (which may have changed since the estimation).

Because the majority of apps are complex projects, some small details (an additional button on a screen, a logic for an uncommon user flow) are inevitably neglected during the planning, and the time & material model enables developers to quickly and easily include them in the process. Also, if in the fixed price model the estimation of every feature includes potential complications, which may not come true after all, the time & material model reduces such risks for the development team, so it can offer more comfortable rates to the client.

Conclusion

Mobile app estimation is a process that takes into account all aspects of the future project and is based on a detailed technical specification. The process is conducted in several stages, from the project's breakdown by screens or by features, to the estimation by developers and then by a project manager. Although the development team may provide to the client a precise cost estimate for app development and work on the fixed price model, a more rough and top-level estimation together with the time & material price model bears fewer risks for both parties and is more flexible to implement necessary changes when the development has already started.