approach for its users. This process is complicated and costly because today there are many platforms out there, such as: iPhone, Android, Windows and Blackberry, which require different software development environment (Halidovic & Karli, 2014; Forni & Meulen, 2016). Thus, many companies struggle to develop applications to reach out to users across platforms. Thus, the need for hybrid application development has become necessary in order to ease the process by allowing to create cross-platforms mobile applications to be developed with a single codebase.

Initially, cross platform development started with HTML5, CSS and JavaScript, but this development approach was not able to support many features required by end users including camera, geolocation and notifications. Subsequently, developers found a better approach for cross-platform development using third-party support, which creates a container inside native application and allows web applications to be embedded inside this container. Presently, cross-platform development is used widely by many software companies. This development approach uses one codebase for multiple platforms which makes the development process shorter and low cost.

Several studies have been conducted to identify the best approach for creating cross-platform applications with a single base code. Positive improvements have been made with the support of new frameworks introduced by large companies such as Microsoft, Google and Facebook. The attention towards cross-platform developments have been increasing and software companies have realised the benefits of investing on this development technology. However, frameworks have their own benefits and drawbacks. The following section evaluates the latest frameworks native verses hybrid development that are significant to achieve reliable cross-platform applications.

2. Cross Platform Development (Hybrid vs Native)

Mobile applications may be categorized as native, web application, or hybrid. Native applications are developed using individual native languages and usually target one specific platform. For instance, Java programming language is used to develop Android applications while Swift based objective C is used to develop iOS applications. This approach allows creating compatible mobile application for only one target platform and is the most reliable way to develop mobile applications (Amatya & Kurti, 2013). However, using native development approach is not suitable solution for organizations that seek to exchange data among cross-platforms.

Table 1. Native Applications Development Requirement (Latif et al., 2016)

	iOS	**Android**	**Windows Phone**	**Blackberry OS**
Languages	Objective C, Swift	Java, (some C, C++)	C# and VB.Net	Java

Tools	xCode	Android SDK/ Android studio	Visual Studio	BB Java Eclipse
Format	.app	.apk	.xap	.cod
Hosting	App Store/ i-Tune	Google Play Store	Windows Phone	Blackberry World

Table 1 displays the programming languages, tools, formats, and the hosting environments for development of native mobile applications. Creating native applications is certainly complex since each platform requires unique way of development. The limitation to one platform is not convenient for companies that want to reach all users cross-platforms. They invest huge amount of money to create mobile applications for Android, iOS, Windows and website development on the other hand. The challenge is also in terms of cost, effort, team development skills, technology applied and time to build the application.

By contrast, web applications are developed using HTML5, CSS and Java Script. This development approach allows applications to run across web browsers using a single code base (Allen, Graupera & Lundrigan, 2010). However, HTML5 web applications do not support many functions in mobile devices including camera, geolocation, user interface and offline support (Heitkötter, Hanschke, Majchrzak, 2012).

As a result, hybrid mobile applications were introduced which are essentially web applications embedded inside a native container enabling the HTML5 pages to be displayed in mobile device in same behaviour as native applications (Halidovic, & Karli, 2014). Mobile hybrid applications are developed with HTML5, CSS and JavaScript and supported by frameworks (Halidovic, Dhupia & Rubab, 2015). Hybrid applications are developed in such a way that have look and feel of native applications. Generally, end users find it hard to distinguish the difference between hybrid and native applications. However, in the development side it is a huge difference due to the tools and technologies used in the development. Hybrid application have been facing issues in performance and user interface design.

All three development formats have their own benefits and drawbacks. Much discussion is ongoing about which approach is better. With the increased competition among companies, the demand for having mobile applications with low cost and fast development has also increased. There is no doubt that native application development is still preferred for creating reliable applications. However, in long term, hybrid applications appear to be a better solution since they are able to run across platforms. As an example, many applications had been developed to run on the Blackberry platform. However, in the recent years Blackberry brand does not have new mobile version and company is almost out of the market. This means, that applications should not be developed only for a particular platform but instead applications need to be cross-platform to sustain in long run. Business organizations are looking for applications that are 'developed once and run everywhere' (Charkaoui, Adraoui & Benlahmar, 2014).

Table 2 displays similarities and differences between native, hybrid and HTML5 web applications. The table shows the benefits of cross-platform development compare to web applications and native applications development. However, hybrid approach has issues with performance and user interface. The responsiveness of hybrid applications is slower compare to native applications. Nevertheless, choosing the right development framework enhances the hybrid performance, giving more powerful features with native feel and look. On the contrary, Table 2 shows that hybrid approach has extra benefits compare to native such as application flexibility, optimization, and multiplatform coverage.

Table 2. Comparison between Native, HTML5 and Hybrid (IBM Corporation, 2012; Korf & Oksman, 2016)

App Features	**Native**	**HTML5/ Web apps**	**Hybrid**
Development Languages	Only Native	Web only	Native and Web
Code portability and optimization	None	High	High
Access Device Specific Features	High	Low	Medium
Leverage existing knowledge	Low	High	High
Graphics	Native API's/ High	HTML, Canvas, SVG/ Medium	HTML, Canvas, SVG/ Medium
Upgrade flexibility	Low	High	Medium
Performance	fast	Slow	Medium
Native look and feel	Native	Emulated	Emulated/ closer to native
Distribution	App Store	Web	Play Store, App Store
Device Access			
Camera	Yes	No	Yes
Notifications	Yes	No	Yes
Contacts, Calendar	Yes	No	Yes
Offline storage	Secure file storage	Shared SQL	Secure file system, shared SQL
Geolocation	Yes	Yes	Yes
Gestures			
Swipe	Yes	Yes	Yes
Touch and Pinch	Yes	No	Yes
Connectivity	Online and Offline	Mostly online	Online and offline

2.1 Challenges of Cross-Platform Development

Developers face several challenges when creating cross-platform applications. Usually problems encountered are related to performance, functionality and design. These issues create uncertainty among business organisations and software companies when choosing cross-platform over native development. Cross-platform development is still improving and does not appear to be yet completely stable. According to IBM Corporation (2012), most common issues identified in hybrid development are HTML5 limitations, user interface design, lack support of animation and slow performance.

HTML5 has limitation on accessing (DOM) Document Object Models in web pages (Eisenman, 2016). DOM is an application programming interface (API) which support programmers to build documents, define logical structure of HTML files allowing to add, modify and delete elements. According to Halidovic & Karli (2014) the use of plugins in developments of hybrid applications may not be compatible with the new versions of frameworks. Hence, switching to the latest framework version will affect the application, so developers need to constantly solve issues occurred whenever new update is applied.

In many instances, the user interface design needs to be adjusted based on the target platform. This usually takes extra time for the development team to test the interface design for multiple platforms ensuring application works good for different mobile screen sizes. Yet, another issue is the slow responsiveness of application when using camera and animations. This can cause performance issues including slow response, battery drain, and freezing phone (Korf & Oksman, 2016). Another, issue is the lag on touch screen which may delay response time up to 300ms compared to native. As observed by Natili (2013), HTML5 also has weakness when displaying long list of data, causing delay or freezing on data loading. However, this issue may be resolved with the support of plugins.

Table 3: HTML5 features supported on WebView

HTML5 Features	**Android WebView**	**iPhone WebView**
Elements of media: video and audio	Yes	Yes
Input types: search, TEL, number and so on	Yes	Yes
Semantic elements: footer, header, articles, summary, meter etc.	Mostly	Mostly
Web Storage	Yes	Yes
Application catches	Yes	Yes
SVG and Canvas Graphics	Yes	Yes
Attributes of CSS3	Mostly	Mostly
Math ML	No	Mostly
Drag and drop	No	No

The issues discussed are the main reasons why software companies and business organizations struggle to decide when choosing cross-platform development approach. Latest frameworks that support cross-platform development claim their capabilities are strong enough to compete with native development. This research investigates the main frameworks that support cross-platform development and attempts to identify whether the frameworks are capable to minimize or solve the issues faced by developers when developing cross-platforms.

Table 3 contains some of the HTML5 features that are supported by Android and iOS WebView. Although many HTML5 features can be implemented, programmers cannot rely on HTML5 development, because limitations affect the overall development of cross-platform applications.

2.2 Cross-Platform Development Frameworks

Cross-platform frameworks have made the development process way easier by providing APIs and Libraries which support the implementation of native functionalities. Those frameworks support the development of applications a 'native look and feel'. There are several open source frameworks available for use. Large companies such as Facebook and Microsoft are investing on this technology to provide better approach for cross-platform development. Due to the large number of frameworks, this section is dedicated to evaluation of three well known cross-platform frameworks: Apache Cordova, Xamarin and React Native. React Native is relatively new open source framework. It was launched in 2016 and is widely used in many software development companies.

2.2.1 Apache Cordova Framework (PhoneGap)

Apache Cordova is a well-known open source framework developed for supporting hybrid mobile applications using web technologies such as HTML5, CSS3 and JavaScript. Development with this approach requires WebView embedded, structure API to access functions of native applications, plugins and file storage (Bosnic, Papp & Novak, 2016).

This framework is widely adapted by many software companies because it overcomes the limitations of HTML5 and has a native feel and look (Pardeshi & Shirvas 2013). Applications developed with this framework are supported by Android, iPhone, Palm, Windows Phone, Symbian and Blackberry (Allen, Graupera & Lundrigan, 2010). Apache Cordova provides access to camera, GPS, file system and accelerometers with the embedded HTML5 code which is then displayed in mobile view just like native applications. Apache Cordova also supports multiple plugins and libraries which allow extending functionalities with JavaScript codebase. Under this framework many tools for development are supported including Ionic SDK, Adobe PhoneGap, Visual Studio allowing developers to have multiple selection of tools for developments (Francese et al. 2013; Natili, 2013).

Despite many benefits, this framework tends to have a slower performance compare to native especially when many plugins are used. The graphic user interface (GUI) and back end functionalities can be implemented similar to native. However, extra effort is required to implement the graphic user interface for different mobile screen sizes. A study from Heitkötter, Hanschke & Majchrzak (2012) was conducted to evaluate Cordova, Titanium, and Web applications with the aim to compare with native development using several criteria. The result from this study indicate that Cordova is similar to native approach in terms of the look and feel. In addition, users may not be able to differentiate between hybrid applications developed with Cordova and native applications.

A survey by Dalmasso at al. (2013) shows that Apache Cordova have less consumption of memory, power, and CPU, because it does not have a dedicated user interface component. In addition, Rieger & Majchrzak (2016) state that hybrid mobile can have native appearance with the APIs which support the development of cross-platform application to access majority features similar to native applications. However, performance may still be an issue depending on the browser capabilities.

2.2.2 Xamarin Framework

Using C# codebase and Microsoft tools, Xamarin allows creating cross-platform applications for Android, iOS and Window with native user interface. This framework is used by millions of developers because it provides good environment for development of cross-platform applications. The application development for native Android, iOS and Windows is simplified by using a single codebase, existing teams with same development skills for all platforms. After completing development, Xamarin enables applications to be compiled into native platform such as iOS, Android or Window, meaning become purely native. The advantage of using Xamarin is code reuse. Xamarin.foms are powerful for mapping the native code and the user interface allowing to reuse the same forms. The performance is better compare to Apache Cordova since the application is compiled to native after development. However, the use of many plugins results in a slower performance. The decision lies on the development team, who should carefully select plugins and libraries when developing with Xamarin framework. As this framework is relatively new, many bugs have been reported from developers and testers. Boushehrinejadmoradi et al. (2015), built a testing tool called X-Checker to test the process of Xamarin when compiling applications to native Windows and Android. They detected 47 bugs related to inconsistency of translation between the Android and Window Phone APIs. Despite the reported bugs, Xamarin new version updates gradually fixes majority of bugs detected. However, in certain cases Xamarin framework updates may affect the development of previous applications. This requires consideration whether the application that has been developed previously should be upgraded as new releases are available.

2.2.3 React Native Framework

React native is an open source framework that was developed under Facebook. React Native currently supports only iOS and Android platforms. This framework is soon expected to be compatible with other platforms as well. In 2013, Facebook that working with HTML5 is not good solution to create compatible cross-platform applications, instead Facebook focused on creating a framework that was developed with HTML5, JavaScript based, and includes native programming languages such as object C and Java to support development. Apart from HTML5, Css and JavaScript, this framework requires understanding of the react.js as well as additional scripting languages including JSX, XML, Xcode and ECMA Script (Eisenman, 2016). Although this framework is new, it is widely adapted by many organizations. The issues of mobile cross-platform developments can be minimized using React Native framework because the application development itself allows the native programming languages to overtake at any time when complex requirements need to be implemented. The benefit of using this framework is the reuse of components which are compiled to native within the framework itself without requiring WebView components. This will directly increase the performance including speed, features, native feel and look. In React Native framework the Document Object Model (DOM) components are rendered using in-memory DOM instead of WebView allowing to render the minimal use of DOM only when necessary (Eisenman, 2016).

React Native allows reuse of UI components without rewriting, which allows to switch an application that already exist without rending it (Eisenman, 2016). For instance, an application developed with Cordova can be reused in React Native. Its entire focus is to build solid UI unlike MeteorJS or AngularJs whose main concern is whether the application works. Also, the IU has native approach with JavaScript interaction that is asynchronous with the native development environment. Efficiency in terms of development with rapid product development and native results is another advantage of this framework. React Native has simplified development of cross-platform applications and performance has been increase with DOM abstraction which takes full control over the application without relying on WebView. This approach allows linking of plugins with native, empowering the application to have full access to zoom, rotation and compass, while less memory load is used.

3. Selecting the Right Framework

Deciding between native and hybrid development depends on the complexity of application as well as target platform. For large companies, choosing cross-platform development on in stage is risky since this approach is constantly upgrading and it is not stable. However, companies that need to compete in a digital market or want to explore a digital market within short term results, the cross-platform development approach is better solution. Mercado, Muniana & Meneely (2016) evaluated the difference between native and hybrid application using 787,228 user reviews data from play store and app store by evaluating users rating for particular applications. The focus on the research was on reliability, security, performance and the usability of the applications. The finding from their study indicates that hybrid approach tends to have more user complaints. However, certain applications that have been developed with the latest frameworks, are nearly with native with feel and look and it is impossible for users to know whether application is hybrid or native.

Cross-platform developments are supported by many frameworks which are built on top of HTML5, CSS and JavaScript. The best approach for cross-platform development depends on the application requirements and complexity. Table 4 compares Apache Cordova, React Native and Xamarin frameworks as well as their potential to develop cross-platform applications. Each framework has different approach for creating cross-platform applications.

Table 4: Comparison between three Frameworks

Features	**Apache Cordova**	**Xamarin**	**React Native**
Use of Native Programming Languages	-	C, C++	Object C and Java
Target Platforms	Android, iOS, Windows, Blackberry, web browser etc.	Android, iOS, Windows, Blackberry etc.	Only Android and iOS
User Interface Development	HTML5, CSS3, JavaScript, (AngularJs, Ionic)	HTML5, CSS3, JavaScript	HTML5, CSS3, JavaScript, React Native Components
Reusable code	Yes	Yes	Yes
Native Feel and Look	Moderate – Supported by Ionic SDK	High – Compiled to native app	High – Is built as Native App
Build as Hybrid	Yes	Yes	No
Data Object Model	WebView	WebView	Own Model
Support API's	Yes	Yes	Yes
Able to access data on web	Yes	Yes	Yes

Table 4 indicates that the frameworks have similar objectives, but they use different approach during development. While Apache relay on WebView container and is developed completely hybrid with HTML5, CSS and JavaScript, React Native uses native approach for exchanging data on the application. Also, Xamarin relay mostly on C, C++ programming language and then compile the application to cross-platforms. During compiling applications Xamarin may have several errors and developers need extra work to solve them. React Native allows native programming languages such as Java and C++ to take over at any time when HTML5 and JavaScript do not support certain functions. Also, React Native has better performance compare to Apache Cordova an Xamarin.

However, the limitation of React Native is that it only supports Android and iOS platforms.

The comparison above shows that all frameworks offer good solutions to improve most of the issues when developing cross-platform applications. Most issues are manageable when the development team are professional and they are able to select the right approach for developing cross-platform applications. In addition, the right approach to develop cross-platforms depends on the type of applications that are expected to be developed.

4. Conclusion and Future Research

This study has revealed that cross platform development is the direction for future mobile application development due to its benefits over native development. Cross-platform developments are currently going through a process of redefining the development process in order to be compatible and to overcome the weaknesses of native developments. Presently, selecting cross-platform developments requires consideration of the target platforms, the type of application and availability of the resources in order to develop a reliable application. HTML5 has few limitations that affect the overall performance and user experience. However, much improvements have been made recently with the frameworks introduced by large companies such as Microsoft (Xamarin) and Facebook (React Native). These frameworks are promising due to their ability to create application with a single code base and then compile into native applications.

The overall benefits of cross-platform development are for end users and organizations. Using this approach brings users updated applications within a short time, while organizations spend less money, and are able to bring innovation solution into the market by targeting multiple platforms.

Further research is needed to test the cross-platform development frameworks since there are many, but none of them appears to be completely stable. Therefore, being able to test these frameworks would contribute better to the improvement of cross-platform development and would establish standardisation for consistent cross-platform application development.

References

Allen, S., Graupera, V. & Lundrigan, L. (2010). Pro smartphone cross-platform development. 1st Ed. New York: Apress.

Bosnic, S., Papp, I. & Novak, S. (2016). The development of hybrid mobile applications with Apache Cordova. *2016 24th Telecommunications Forum* (TELFOR). p.1-4.

Boushehrinejadmoradi, N., Ganapathy, V., Nagarakatte, S. & Iftode, L. (2015). Testing Cross-Platform Mobile App Development Frameworks (T). *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. p.441-451.

Charkaoui, S., Adraoui, Z. & Benlahmar, E. (2014). Cross-platform mobile development approaches. *2014 Third IEEE International Colloquium in Information Science and Technology (CIST)*. p.188-191.

Dalmasso, I. S. K. Datta, K., Bonnet C. & Nikaein, N. (2013). Survey, comparison and evaluation of cross platform mobile application development tools. *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC),* Sardinia. p. 323-328.

Eisenman, B. (2016). *Learning React Native*. 1st ed. United States of America: Rally Media Inc. p. 10-50.

Francese, R., Risi. M., Tortora, G. & Scanniello, G. (2013). Supporting the development of multi-platform mobile applications. Web Systems Evolution (WSE), 2013 15th IEEE International Symposium. p. 87,90, 27-27.

Halidovic, R. & Karli, G. (2014). Cross-Platform Mobile App Development using HTML5 and JavaScript while leveraging the Cloud*. IOSR Journal of Engineering.* 4(2). p.06-11.

Heitkötter, H., Hanschke, S. & Majchrzak, T.A. (2012) Comparing cross-platform development approaches for mobile applications. *In: 8th International Conference on Web Information Systems and Technologies, WEBIST*. p. 299–311

IBM Corporation (2012). Native, web or hybrid mobile-app development. Somers, NY 10589 [Online] Available at: ftp://public.dhe.ibm.com/software/pdf/mobile-enterprise/WSW14182USEN.pdf [Accessed 20 March 2017].

Korf, M. and Oksman, E. (2016). Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options. [online] Salesforce Developers. Available at: https://developer.salesforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options [Accessed 25 March 2017].

Latif, M., Lakhrissi, Y., Nfaoui, E. & Es-sbai, N. (2016). Cross platform approach for mobile application development: a survey. *2016 International Conference on Information Technology for Organizations Development (IT4OD)*. Fez. p. 1-5.

Mercado, I., Muniana M. & Meneely A. (2016). The Impact of Cross-Platform Development Approaches for Mobile Applications from the User's Perspective. *Proceedings of the International Workshop on App Market Analytics - WAMA 2016*. p.43.

Natili, G. (2013). PhoneGap Beginner's Guide. 1st ed. Birmingham: Packt Publishing.

Pardeshi, A. (2013). To Study and Design a Cross-Platform Mobile Application for Student Information System using PhoneGap Framework. *International Journal of Emerging Technology and Advanced Engineering.* 3(9), p.390.

Rieger, Ch. & Majchrzak, T. A. (2016). Weighted Evaluation Framework for Cross-Platform App Development Approaches. *Springer International Publishing AG 2016 S. Wrycza (Ed.):* SIGSAND/PLAIS 2016. LNBIP 264, pp. 18–39.

Sui, L (2016). 44% of World Population will Own Smartphones in 2017. [Online] Available at: https://www.strategyanalytics.com/strategy-analytics/blogs/smart-phones/2016/12/21/44-of-world-population-will-own-smartphones-in-2017#.WgESRGiCzIU [Accessed 20 March 2017]. Strategyanalytics.com [Accessed 28 March 2017].