Oregon State University
College of Engineering

CS Capstone Problem Statement

November 9, 2019

Building a Robot Using Only Philosophy

Prepared for

OSU Department of Geology

Matt Evans _____ Signature _____ Date

Prepared by

Group 36
BLAMO Development Team

Sean Spink _____ Signature _____ Date

Evan Amaya _____ Signature _____ Date

James Trotter _____ Signature _____ Date

Alex Smith _____ Signature _____ Date

Abstract

Our group is building an application to limit the encumbrance of using paper-based data systems. We are building a paperless system to replace the current physical system in place. The digital system will enable users to have a reliable way of recording and transporting data between the field and the lab. Additionally, a stretch goal is to allow the user to seamlessly import the data to an ArcGIS system automatically.

My Job for this tech review is to analyze UI frameworks, Programming languages, and Storage methodologies for data that can be implemented efficiently and reliably in a mobile environment. When analyzing the tools for the BLAMO project, I will be looking at multiple criteria for each category, and describing the benefits of each option. This research will help our team decide which development tools are the best for the project and its implementation.

CONTENTS

1 UI Framework

UI frameworks allow for a simpler Implementation of a user facing Graphical Interface. Since Object Oriented Programming is modular, certain functionality can be imported from libraries like a UI framework. Importable frameworks allows modules to be designed quickly and efficiently without reinventing the wheel.

1.1 Criteria

The User Interface (UI) framework will be front and center to every user of the BLAMO: Bore Hole Logging app. A UI framework provides a simple toolset to establish It is imperative that the framework be Cross Platform (iOS/Android) and that it provides clear and coherent layouts for the users. Another job of the framework is that it provides appealing looks to the user, while supporting inclusive design and wide customization. Some of the criteria we are looking for in a UI frame work is:

- Supported by our chosen language
- Flexible to suit inclusive design
- Cross platform support for iOS/Android
- Easily picked up and implemented
- Lightweight (Low system resource requirements)

After looking around at quite a few UI's the three that fulfilled the above criteria was Flutter, Xamarian, and TonadoFX. The biggest points that the three aforementioned UI's fulfilled was the Cross-platform support, Flexible design, and easily picked up.

1.2 Flutter

Flutter is a UI produced and developed by Google. Google designed flutter to be an all-encompassing mobile UI (supporting both Android and iOS). Google advertises Flutter as "Flutter is Google's UI toolkit for building beautiful, natively compiled applications for mobile, web, and desktop from a single codebase." [1] Flutter supports fast development, a flexible architecture, and full native performance on iOS and Android. All of the attributes of Flutter mean it is a strong, lightweight, responsive, and adaptive option. An added bonus to flutter is it supports desktop applications, so if users use devices like Microsoft's Surface Pro they can easily adapt the mobile BLAMO App to being a Windows application. Flutter is supported by Kotlin/Java, which means it would filter our chosen programming language to those. Additionally, DART is the official cross-platform language for flutter.

1.3 Xamarin

Xamarin is a Microsoft UI framework for Mobile and Desktop applications. Microsoft describes Xamarin as "Xamarin extends the .NET developer platform with tools and libraries specifically for building apps for Android, iOS, tvOS, watchOS, macOS, and Windows." [2] What Xamarin provides is a Multi-platform UI that is supported by C-Sharp/.NET and is open source. Another benefit to Xamarin is that it allows for easy integration with Azure, Microsoft's growing cloud development platform, to support cloud integration. The downside to Xamarin is that it is a bit of learning curve however Microsoft has a thorough library of documentation on its uses.

1.4 TornadoFX

TornadoFX is a translation of the JAVAFX framework to use with Kotlin/Java. The strong points of TornadoFX is great JSON support, IDE hot reloading (With Intellij IDEA plug in), and wide design options to support all types of UI's [3]. TornadoFX is designed for elegant code to match a nice UI with Java development in mind. TornadoFX has support for Mobile and Desktop interfaces, which would be beneficial for future development.

1.5 UI Framework: Final Statement

Between Flutter, Xamarin, and TornadoFX, Flutter seems to be the best suited language. Since flutter supports native performance to both iOS and Android it is a lightweight framework. Flutter is also developed by google, meaning support for our target platform (Android) would most likely be the strongest. Additionally, this decision was made in conjunction with programming language, so Kotlin support became a must.

2 Programming Language

Programming languages very strongly from language to language. Some programming languages are better designed for desktop implementation, some for web-pages, and others for mobile development. Languages very in execution time on their end device because some compilers can produce more efficient machine code than others, depending on their target audience.

2.1 Criteria

Choosing the right programming language is important to our project because it directly effects the quality and efficiency of the development cycle. If we chose something like a Functional Language there would be a lot of hurdles and foreign concepts to learn that would seem like reinventing the wheel. So, when deciding on a programming language the criteria are simple and solid:

- Android Support
- Familiar to All Developers involved (Syntax, Type Casting, OOL)
- iOS Portability
- GUI support
- Easy multi-platform development
- Initial Android implementation
- Simple for Future Developers to pick up

When choosing a programming language, the above criteria must be met. With the criteria in place, the programming languages that are up for consideration are Kotlin, Java, and C-Sharp. Each of the three languages have strengths and weaknesses that need to be taken into account for our project.

2.2 Dart

Dart is a programming language designed around supporting mobile development, and the flutter framework [4]. The premise of Dart is that It is made to be a single code base for a multi-platform implementation, removing the complications of using Java or Objective-C. Another large benefit to Dart is speed of running to near native performance, which means when Dart compiles for a device it will run similarly to if the program was written in Java (for android) or Swift/Objective-C (For iOS). Dart Would provide the easiest cross-platform implementation for the BLAMO project.

2.3 Kotlin

Kotlin is a programming language designed around supporting mobile development. More specifically, Kotlin was designed for Android, JVM, Browser, and native development and was targeted towards the Mobile market [5]. Kotlin has iOS and Android support, fulfilling the cross-platform criteria. Kotlin is built as an easy-to-transition java replacement with more recent programming features. Kotlin also hosts the ability to utilize any java framework and libraries, giving a fresh implementation with the power of legacy. Lastly, Kotlin has a concise implementation which allows for easier readability and maintenance simplicity. Kotlin would support TornadoFX and Flutter as UI frameworks.

2.4 Java

Java is the industry standard for most Android/Computer applications and has a wide array of usability. The legacy, familiarity, and platform support of Java makes it a strong contender for the BLAMO application. Using Java would allow for a wide array of Libraries and Frameworks, efficient development, and no learning curve (as we are all familiar with the language). Java was designed to be a cross-platform language that could run on any device that supports the java platform. Having multi-platform support would benefit the BLAMO app by providing more option for future expansion. Java would support TornadoFX and Flutter as UI frameworks.

2.5 C-Sharp

C-Sharp provides a strong foundation to build on, toting the strength of C with the usability of Java. Utilizing C-Sharp offers a cross-platform between mobile and desktop, familiar language implementation, and a powerful framework called .NET. Another feature of C-Sharp is the excellent documentation provided by both Microsoft and the C-Sharp community. Choosing C-Sharp would restrict our UI framework to Xamarin (Which I discuss in the UI framework section).

2.6 Programming Language: Final Statement

Our final pick out of the three languages is Kotlin. The reason for choosing Kotlin is how easy it is to pick up, efficient compilers, and support for our chosen UI framework, Flutter. The other advantage Kotlin has to the other languages is that it is tailor made for mobile development. Since Kotlin is a fast, safe, and concise language that supports the same frameworks as java it is the choice for our project.

3 Local Storage on Mobile Device

3.1 Criteria

Users of the BLAMO: Bore Hole Logging App will need reliable persistent storage for their data. Having reliable data for data logging maintains data integrity, and eases the users mind, and there are a few ways to implement such a system on Android, External Storage Systems, and Internal File Storage.

3.2 Internal File Storage

Internal file storage: Allows a private memory space for applications to store data that is non-accessible by other applications or outside sources. This method would be good if we were trying to build an app that is security critical, however our application is not severely dependent on security. Since data integrity and access is imperative to the

BLAMO app, it might be more beneficial to have an externally accessible data location in case of damage to the functionality of the device. If a user had recorded a host of data and they drop the tablet, shattering the screen, and removing critical functionality to the device, it would be nice to still be able to retrieve the data.

3.3 External File Storage

External File storage: This is a public shared data space for the device, allowing for access under any user or account. The public shared space is used for documents, photos, and video; however, applications can use it as a storage space for primitive data as well. The advantages to using the external File storage is that since it is publicly accessible it can be accessed even if device functionality is compromised. Having publicly accessible data would be a design decision that would need to consider the users security needs.

3.4 Local Storage: Final Statement

To fulfill the needs of the user, having the data publicly accessible would be a valuable trait, as the users of the BLAMO app are going to be doing mobile field work to log the data. Field work environments are prone to accidents, and mitigating the damages in case of an accident is extremely important, thus the External File Storage would be the best implementation for saving local data. Android allows data access via a hard wire connection, so a user could extract their data from a broken device once they get it to a desktop.

REFERENCES

[1] "Flutter" *Flutter By Google*. [Online]. Available: https://flutter.dev/. [Accessed: 04-Nov-2019]
[2] "Xamarin" *Xamarin by Microsft*. [Online]. Available: https://dotnet.microsoft.com/apps/xamarin/. [Accessed: 03-Nov-2019]
[3] "TornadoFX" *TornadoFX*. [Online]. Available: https://tornadofx.io/. [Accessed: 03-Nov-2019]
[4] "Dart" *Dart by Google*. [Online]. Available: https://dart.dev/. [Accessed: 05-Nov-2019]
[5] "Kotlin" *Kotlin by Jet Brains*. [Online]. Available: https://kotlinlang.org/. [Accessed: 04-Nov-2019]
[6] "Java" *Java by Oracle*. [Online]. Available: https://docs.oracle.com/javase/7/docs/technotes/guides/language/. [Accessed: 03-Nov-2019]
[7] "C-Sharp" *C-Sharp by Microsoft*. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/csharp/. [Accessed: 03-Nov-2019]