CS 131: Containerization Support Languages

Wilsen Kosasih – University of California, Los Angeles

Abstract

Docker is a software container platform that provides a method to pack, ship, and run software applications. It was originally built on top Linux Container and is fairly fast and lightweight. It is implemented in Go, which is a relatively new programming language. This paper will compare the feasibility of Go in the implementation of Docker against other programming languages: Java, OCaml, and Rust.

Introduction

Containerization encapsulates an application on its own operating environment. Containerization provides the benefit of standardization, which means that the application can be run anywhere without worry of dependencies [1].

The goal of Docker is to automate software deployment to be as lightweight and quick as possible. It was build on top of the Linux Container concept.

In this report, we will compare the use of Go programming language on the implementation of Docker compared to other programming languages.

1. Go

Go is a new language developed by Google. It is a statically compiled language. This is a huge advantage because this makes programs written by Go be tested and can be built on multiple platform rather easily. Since Go needs a programming language that transition really well between systems, Go is really good for this reason.

In addition, Go uses Duck Typing, which means that the development of programs using Go can be tested easily. Additionally, programming languages that use duck typing are typically easy to write.

However, there are some problems with Go. First, because it is a relatively new programming language, many people may not be familiar with it and may face many problems compared to when they are using well-known languages such Java or C++.

Additionally, Go sacrifices safety for speed. Go does not have compile-time generics and is not threads safe.

2. Java

Java is a great and a pretty well-known programming language that is used in many applications, such as web applications and androids.

It has static type checking that makes it easier for developers to catch errors and debug the program early on the development. It is also a relatively a fast language, which will greatly benefit Docker.

However, Java has no duck-typing feature, which will make it harder to write the program compared to GO.

Additionally, dynamically compiled languages like java usually makes it hard for compilers to figure out dependencies, which is not what Docker want.

Another thing that may cause problems is that Java lacks LXC (Linux Containers) support.

3. OCaml

Ocaml is a strong statically typed language and it uses type inference. This means that developers do not need to specify the type of data, as long as they are handled properly within functions.

Additionally, like Java, OCaml is typically fast and it can even be as fast as C++. Additionally, the byte code generated by OCaml is very portable, which is really great for

implementing Docker. It also has a large variety of useable libraries [2].

However, as with Java, there is no LXC support for OCaml. Additionally, learning OCaml can be harder than languages like Java or C++

4. Rust

Same as OCAML, Rust uses type inference, which makes it faster for developer to write the code. Additionally, the syntax of Rust language seems to be largely similar to those of C++/Java, which will make learning the language less hard.

Additionally, it is speed it comparative to C/C++. However, it seems like there are still many things changing in Rust as it is still evolving [3].

In this regard, it does not seem like using Rust might be a good idea, as debugging might be hard.

Another plus point for Rust is that it works for both iOS and Android. However, it requires a bit work to implement it, which does not seem to align with the goal of Decker[3].

5. Conclusion

In conclusion, despite being a relatively new programming language, Go is a really good fit for Docker. The only problem it had is that it is not threads safe.

Meanwhile, I believe that Java would not be a good match to Docker due to the lack of duck typing and LXC support.

On the other hand, OCaml might be a good match for Docker due to its speed and portability.

Also, It seems like Rust is not a mature enough language to be used as it lacks documentations and is still evolving.

References

[1]Stroud, Forrest. *containerization.* Retrieved From http://www.webopedia.com/TERM/C/containerization.html

[2] *Why OCaml?.* Retrieved From http://www2.lib.uchicago.edu/keith/ocaml-class/why.html

[3] Retrieved From https://www.rust-lang.org/en-US/faq.html#libraries