

**Fortune Newspapers Addis Ababa**

**Ethiopia**

**------------------------**

**Reading assignment**

**Ermias Mulugeta**

**\*IT\* Student in AAIT(Addis Ababa institute of technology).**

Submitted To: Mr. Abel Seyoum Araya

March 2020

Table of Contents

[1 Miro-services 2](#_Toc36029883)

[1.1 module, modularity 2](#_Toc36029884)

[1.2monolith vs microservice 3](#_Toc36029885)

[2 Distributed system 4](#_Toc36029886)

[2.1 Service discovery 4](#_Toc36029887)

[2.3Virtual machines 4](#_Toc36029888)

[2.4 Containers 4](#_Toc36029889)

[2.4.2kubernetes 5](#_Toc36029890)

[3 linux container 6](#_Toc36029891)

**ACKNOWLEDGEMENT**

First of all I would like to thank Mr. Abel for giving me this chance. Its such a pleasure to do this reading assignment, I learn a lot of thing form it. its an opportunity for knowing about this software architecture, how to develop a software with the big scope and to make the software with the long term of period.

# 

# 1 Miro-services

## 1.1 Module, Modularity

A software developer develops an application then he must think about the future of the softawer, it depends on the developer scope of view.If the developer is breave, he develop his application for the long term and it has to be.So in that space that product development space relly turns to module, module systems modularity as aprinciple to help the developer to achive.modularity helps to do such things.Modularity is abit more concrete in the scope of software-development.There are three tentes of modularity theat the developer must keep in mind when he thinking about software development.The first one is all about strong and capsulation, these two are all about hiding staff.when the developer thinks about hiding, it has to be seted, because all components musnt hide, also the developer must contians some api’s , contract service, definations that are explicitly defined. <strong>It doesn’t only apply to library development</strong> this also applies to the large applications, so these well-defined interfaces allows thses modules that have strongly encapulated parts to still work together in a usefull way. The last one is explicit dependencies, if the developer of creating modules that have encapulated parts and that publicly exported api’s then the developer end up with an application that consists of small parts.

Modularity is the ultimate agile tool because if there is module and modular part in the system

It means that the developer can easily replace internal so the developer can replace iplementaions as long as the developer adhere to these well-defined interfaces.

Microservice is a peace of software component independently deployeble implementing a business, its just not little technical services these are domian services usauly capability doing one thing, and doing it well communicating over a network.microservice are quite difficult to implement, because it needs a buch of people and a resource.so it needs an assumption to use the microsercice when and not to use. now days Microservice is used across the world because of:

**manage complexity**:- large systems are too complex to manage too complex to evolve some how to manage thiese complexity by chopng into a peaces, that’s how it works.

**Scaling** :-- the scaling concept is used for scaling imdependet services and resilience in the arctechture.

## 

## 1.2monolith vs microservice

In a software enginnering a monolithhic application describe a single-tiered software application in which the user interface and data access code are combined into a single program from a single platform. A monolith application is self-contained, independent from the other computing application and also its responsiple not just for a particular task,can perform every step nedded to complete aparticular function.

Today some personal finance applications are monolithic in the sence that thay help the user carry out a complete task ,end to end. But because of all these implicit dependecies, if the programmer tries to fix in some area it could be it falls on the ohters. But on the microservice all these messes are fixed.microservice allows the developer to reuse and repaire parts of the application,but development tools are required to perform these maintenance functions(the application may be need to recompied).compared to the monolith microservice provides to create systems with more features that’s better interms of additional costs,bur for the startup developers mololith is choosable because there isnt much to cost off.

# 2 Distributed system

## 2.1 Service discovery

As it listed before moduarity provides a bunch of modules which used that each contains everything necessary to excute only one aspect of the desired functionality.Then those modules have to connect eachohter, for that there are some solutions like <strong> consules,etcd,netflix-EUREKA</strong>.there are also network fallacies like envoy, zipkin, netflix-hystrix.

#### 2.2 Deployment

Deployment means moving a system from the testing phase to the production phase and its includes all the process required for preparing a software application to run and operate in a specific environment it involves installation,configaration,testing and making changes to optimize the performace of the software.

Advantages of deployment:-

1. Time saving (the process of installation becomes faster.)
2. Enchance security (it configures the rols of permission sets.)
3. Moniter user action (it makes easy to moniter user action effectivly.)
4. Effective software update (deployment accurate updates,software maintenance tasks and uninstall can be targeted automatically.)

### 2.3Virtual machines

Virtual machines use so-called “hypervisors” as the emulation layer between the guest and the host oprating system for containers, the rough equivalent is the container engine.

### 2.4 Containers

Containers is used for packages up code and all its dependencies so the application runs quickly and reliably.Continers work very differently.becuase they only conatin the application and the libraries, frame works, etc.they depend on, the developer can put lots of them on a single host oprating system.the only oprating system. The only oprating system on the server is the one host oprating system and the containers small and the iverhead extremely low.

There are a bunch of containers like ducker, kumbernerts ,AWS fargates,IBM cloud kurnets service and so on and The most popular continer is ducker.

2.4.1Doker platform

2.4.1 Kumbernnets engine

2.4.1Linux container

#### 2.4.1 Doker platform

Doker is a set of platform as service products that uses OS-level visualization deliver software in packages called containers. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.

Running Docker on AWS provides developers and admins a highly reliable, low-cost way to build, ship, and run distributed applications at any scale. AWS supports both Docker licensing models: open source.

Docker provides tooling and a platform to manage the lifecycle of your containers:

* Develop your application and its supporting components using containers.
* The container becomes the unit for distributing and testing your application.

Docker uses a client-server architecture. The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers. The Docker client and daemon can run on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface.

### 2.4.2kubernetes

is an open-source container-orchestration system for automating application deployment, scaling, and management. It was originally designed by Google, and is now maintained by the Cloud Native Computing Foundation. It aims to provide a "platform for automating deployment, scaling, and operations of application containers across clusters of hosts.It works with a range of container tools, including Docker. Many cloud services offer a Kubernetes-based platform or infrastructure as a service (PaaS or IaaS) on which Kubernetes can be deployed as a platform-providing service. Many vendors also provide their own branded Kubernetes distributions.

Containers are a good way to bundle and run your applications. In a production environment, you need to manage the containers that run the applications and ensure that there is no downtime. For example, if a container goes down, another container needs to start. Wouldn’t it be easier if this behavior was handled by a system?

That’s how Kubernetes comes to the rescue! Kubernetes provides you with a framework to run distributed systems resiliently. It takes care of scaling and failover for your application, provides deployment patterns, and more. For example, Kubernetes can easily manage a canary deployment for your system.

Kubernetes provides you with:

* **Service discovery and load balancing**  
  Kubernetes can expose a container using the DNS name or using their own IP address. If traffic to a container is high, Kubernetes is able to load balance and distribute the network traffic so that the deployment is stable.
* **Storage orchestration**  
  Kubernetes allows you to automatically mount a storage system of your choice, such as local storages, public cloud providers, and more.

### 

### 3 linux container

Linux containers are technologies that allow you to package and isolate applications with their entire runtime environment—all of the files necessary to run. Its Red Hat OpenShift Container Platform. This makes it easy to move the contained application between environments (dev, test, production, etc.) while retaining full functionality. Containers are also an important part IT-security. By building security into the container pipeline and defending your infrastructure, you can make sure your containers are reliable, scalable, and trusted.

A Linux container is a set of one or more processes that are isolated from the rest of the system. All the files necessary to run them are provided from a distinct image, meaning that Linux containers are portable and consistent as they move from development, to testing, and finally to production. This makes them much quicker than development pipelines that rely on replicating traditional testing environments. Because of their popularity and ease of use containers are also an important part of IT security.

# References

<https://www.youtube.com/watch?v=L1ie8negCjc>

<https://www.youtube.com/watch?v=GYppOyCbM68>

<https://www.youtube.com/watch?v=K5qzEvjIyiE>

<https://www.wikipedia.org/>

<https://www.quora.com/>

<https://stackoverflow.com/>

# 