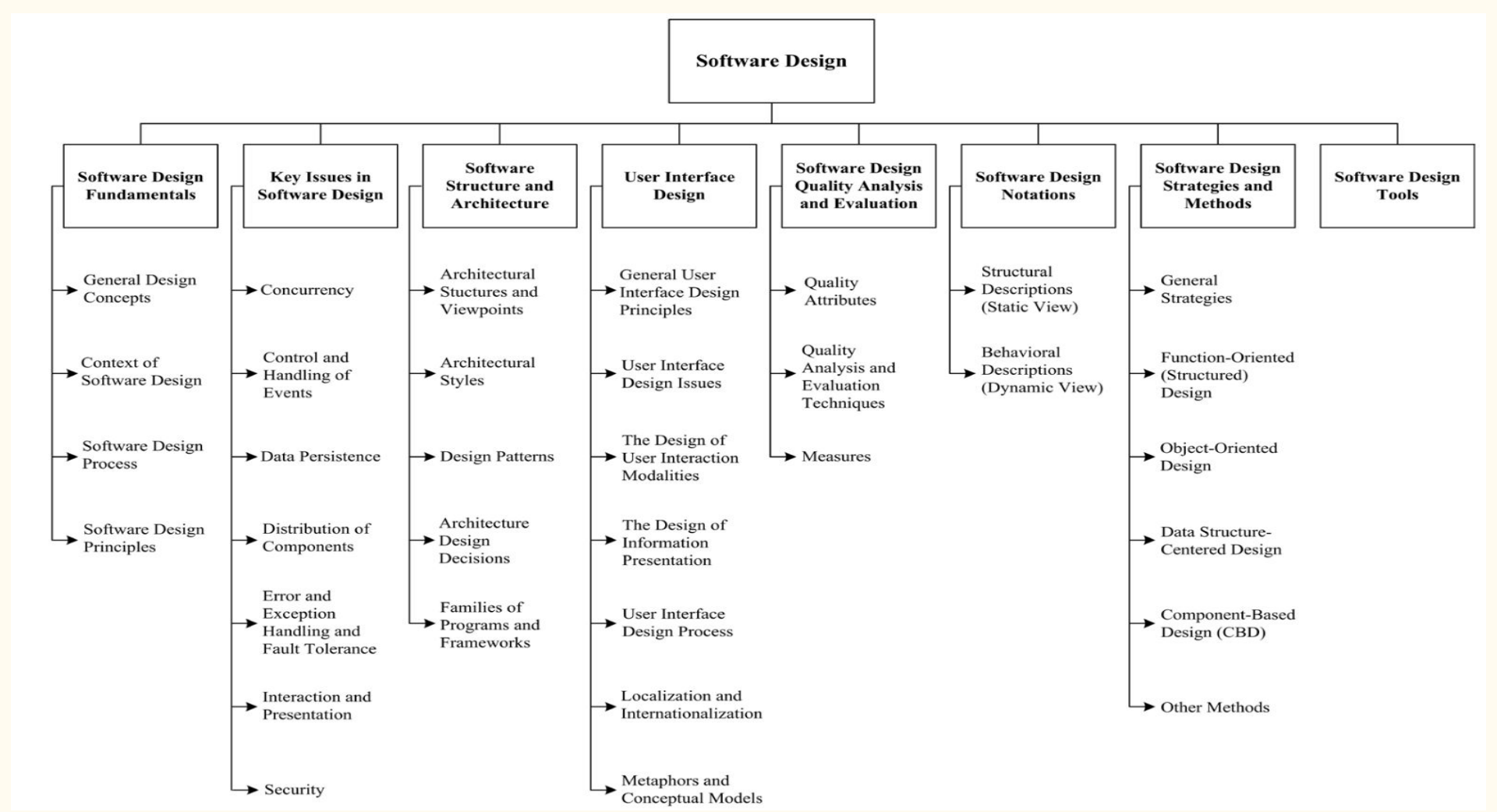
Software Design

Karan Kumar Singh
(173050014)

Sourabh Pal
(173050054)


Software Design
Software Design Fundamentals
General Design Concepts
Context of Software Design
Software Design Process
Software Design Principles
Key Issues in Software Design
Concurrency
Control and Handling of Events
Data Persistence
Distribution of Components
Error and Exception Handling and Fault Tolerance
Interaction and Presentation
Security
Software Structure and Architecture
Architectural Stuctures and Viewpoints
Architectural Styles
Design Patterns
Architecture Design Decisions
Families of Programs and Frameworks
User Interface Design
General User Interface Design Principles
User Interface Design Issues
The Design of User Interaction Modalities
The Design of Information Presentation
User Interface Design Process
Localization and Internationalization
Metaphors and Conceptual Models
Software Design Quality Analysis and Evaluation
Quality Attributes
Quality Analysis and Evaluation Techniques
Measures
Software Design Notations
Structural Descriptions (Static View)
Behavioral Descriptions (Dynamic View)
Software Design Strategies and Methods
General Strategies
Function-Oriented (Structured) Design
Object-Oriented Design
Data Structure-Centered Design
Component-Based Design (CBD)
Other Methods
Software Design Tools

Introduction

The process of defining the architecture, components, interfaces, and other characteristics of a system or component.

Describes the software architecture how software is decomposed
and organized into components and the interfaces between those components.

Content

- Software Design Fundamentals
- Key Issues in Software Design
- Software Design Notations
- Software Design Tools

Software design fundamentals

General Design Concepts

- form of problem solving
- understanding the limits of design

Software Design Process

- Architectural design: how software is organized into components
- Detailed design: Desired behaviour of these components

Software Design Principles

- Abstraction
- Coupling and Cohesion
- Decomposition and modularization
- Separation of interface and implementation

Key issues of software design

Concurrency - decompose into process and threads with issues of efficiency, sync and scheduling.

Control and Handling of Events - control flow and organize data
Data Persistence - handle long lived data

Distribution of Components - component communication, middleware
Error and Exception Handling and Fault Tolerance

Interaction and Presentation - interaction with user and info
Security - prevent unauthorized disclosure, change, deletion, or denial of access to information,etc. Tolerate security-related attacks by limiting damage, speeding repair and recovery

Software design notion

UML

- UML is general purpose, developmental, modelling language
- Used for visualizing, specifying, constructing and documenting the components
- UML gives design a meaning
- There are two broad categories of diagrams:
    - Structural Diagrams
    - Behavioural Diagrams

Structural description (Static view)

-Class diagrams

-Object diagrams

-Deployment diagrams

-Component diagrams

Behavioural description (Dynamic view)

-Activity diagrams

-Use case diagrams

-State machine diagrams

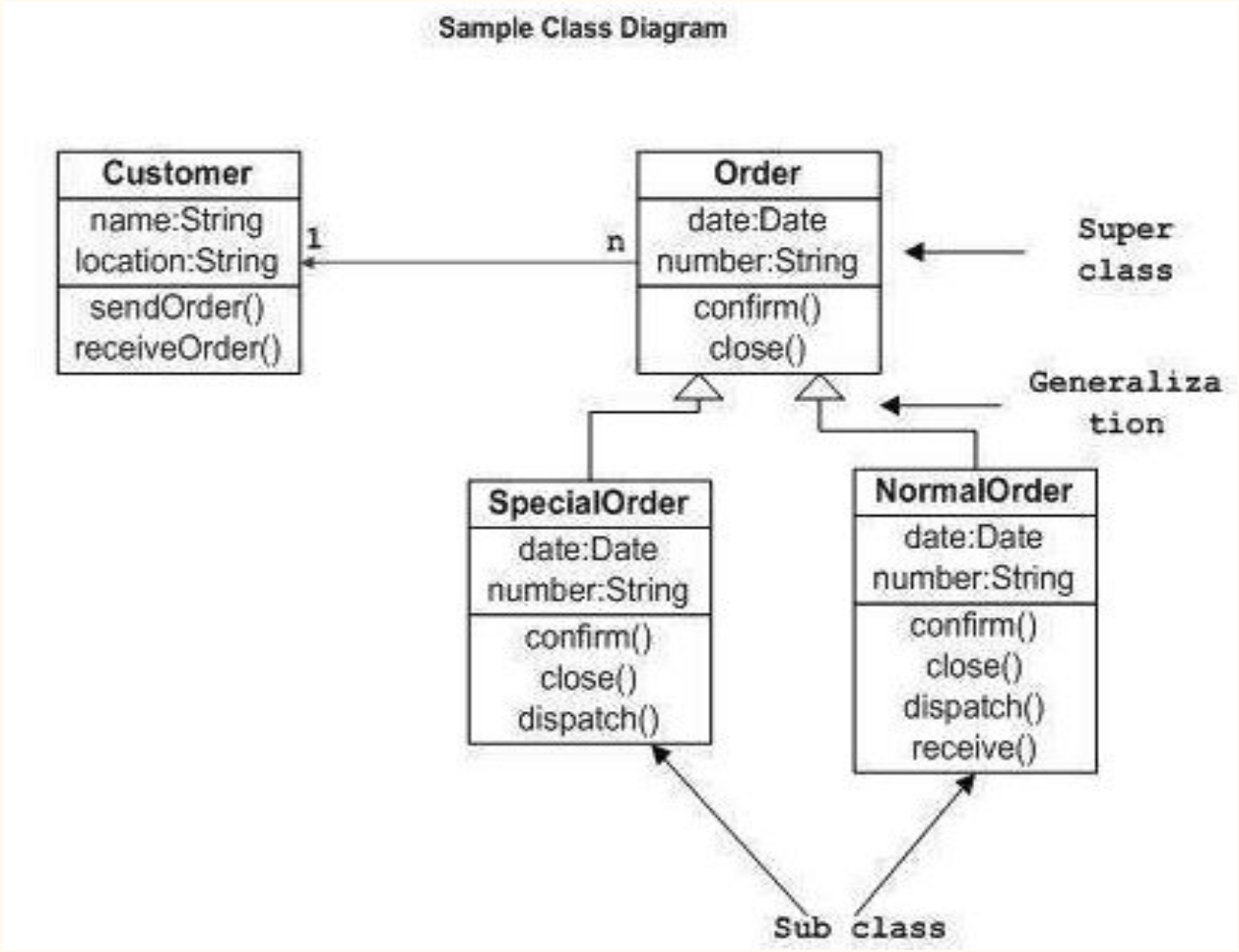
-Interaction diagram

Structural diagrams

- Represent the static aspect of the system
- Static aspects represent those parts of a diagram, which forms the main structure and are therefore stable.
- There are four structural diagrams:
    - Class diagram
    - Object diagram
    - Component diagram
    - Deployment diagram

Class diagrams

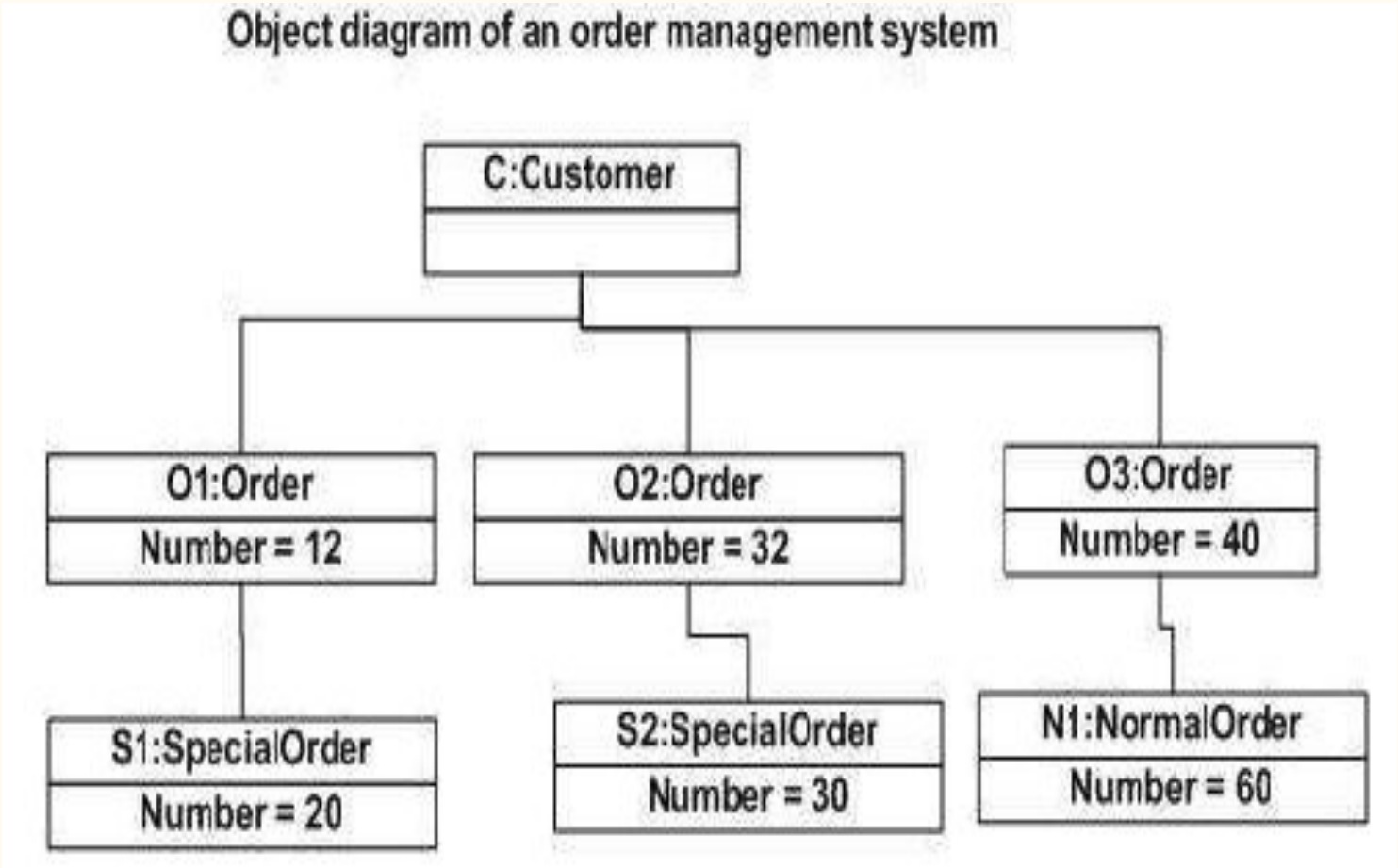
- Represents the static view of an application
- Shows a collection of classes, interfaces, associations, collaborations, and constraints
- Only diagram which can be directly mapped with object-oriented languages
- Describe responsibilities of a system
- Base for component and deployment diagrams
- Forward and reverse engineering.


Sample Class Diagram
Customer
name:String
location:String
sendOrder()
receiveOrder()
1
n
Order
date:Date
number:String
confirm()
close()
Super class
Generalization
SpecialOrder
date:Date
number:String
confirm()
close()
dispatch()
NormalOrder
date:Date
number:String
confirm()
close()
dispatch()
receive()
Sub class

Object diagrams

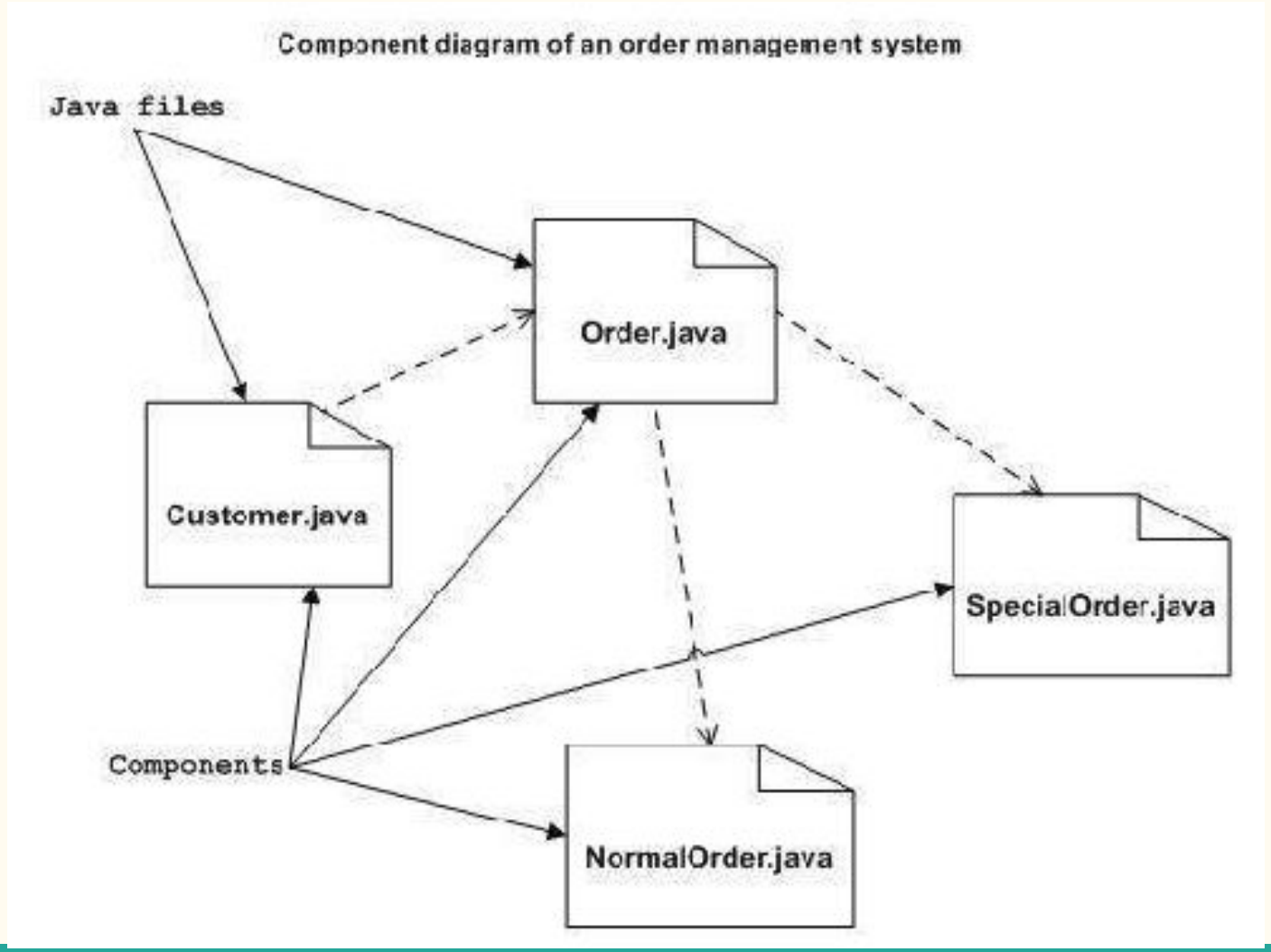
- Represent an instance of a class diagram
- Reverse engineering.
- Object relationships of a system
- Static view of an interaction.
- Understand object behaviour and their relationship from practical perspective
- Modeling complex data structures.

Object diagram of an order management system



Component diagrams

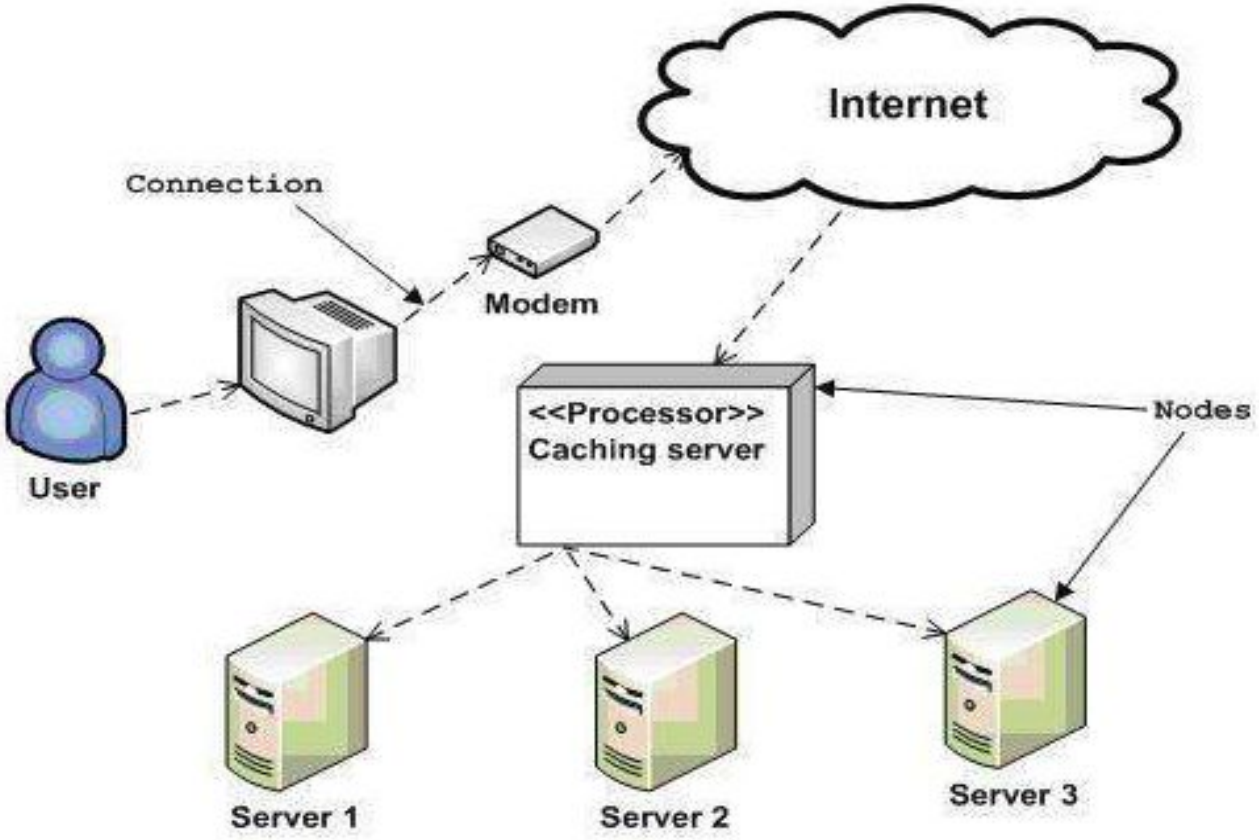
- Visualize the components of a system such as libraries, packages, files, etc.
- Construct executables by using forward and reverse engineering
- Describe the organization and relationships of the components.
- Used during the implementation phase of an application
- Used to model the database schema


Component diagram of an order management system
Java files
Order.java
Customer.java
SpecialOrder.java
Components
NormalOrder.java

Deployment diagrams

- Visualize the hardware topology of a system
- Describe the hardware components used to deploy software components
- Consist of nodes and relationship

Deployment diagram of an order management system



Behaviour diagram

- Behaviour diagram emphasize on what must happen in system being modelled
- Behaviour diagrams show the interaction and flow of action of a working system
- Behaviour diagram visualize and document the dynamic aspects of the system

Activity diagram

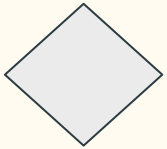
- Activity diagram represent the workflow with support for choice, iterations and concurrency
- Flow can be sequential, branch or concurrent
- Used for business process modeling
- Activity diagram have no message part

Elements of activity diagram

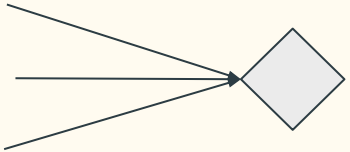
Start point

Action state

Action Flow(Edge)

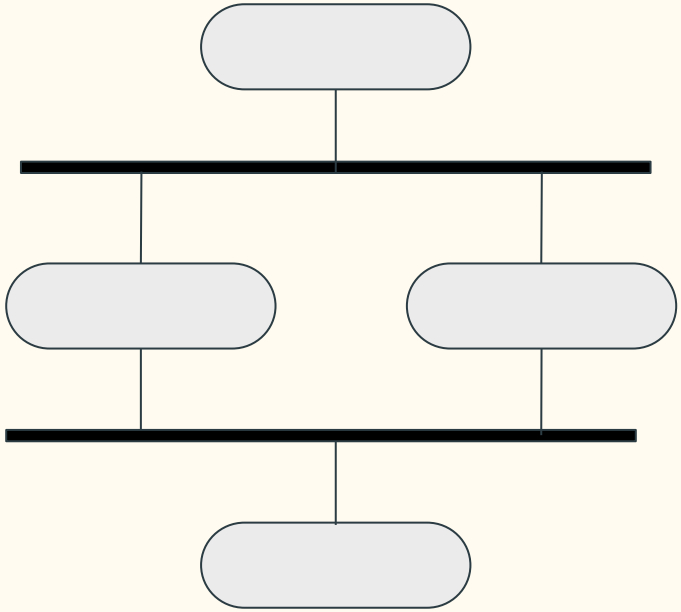


Decision



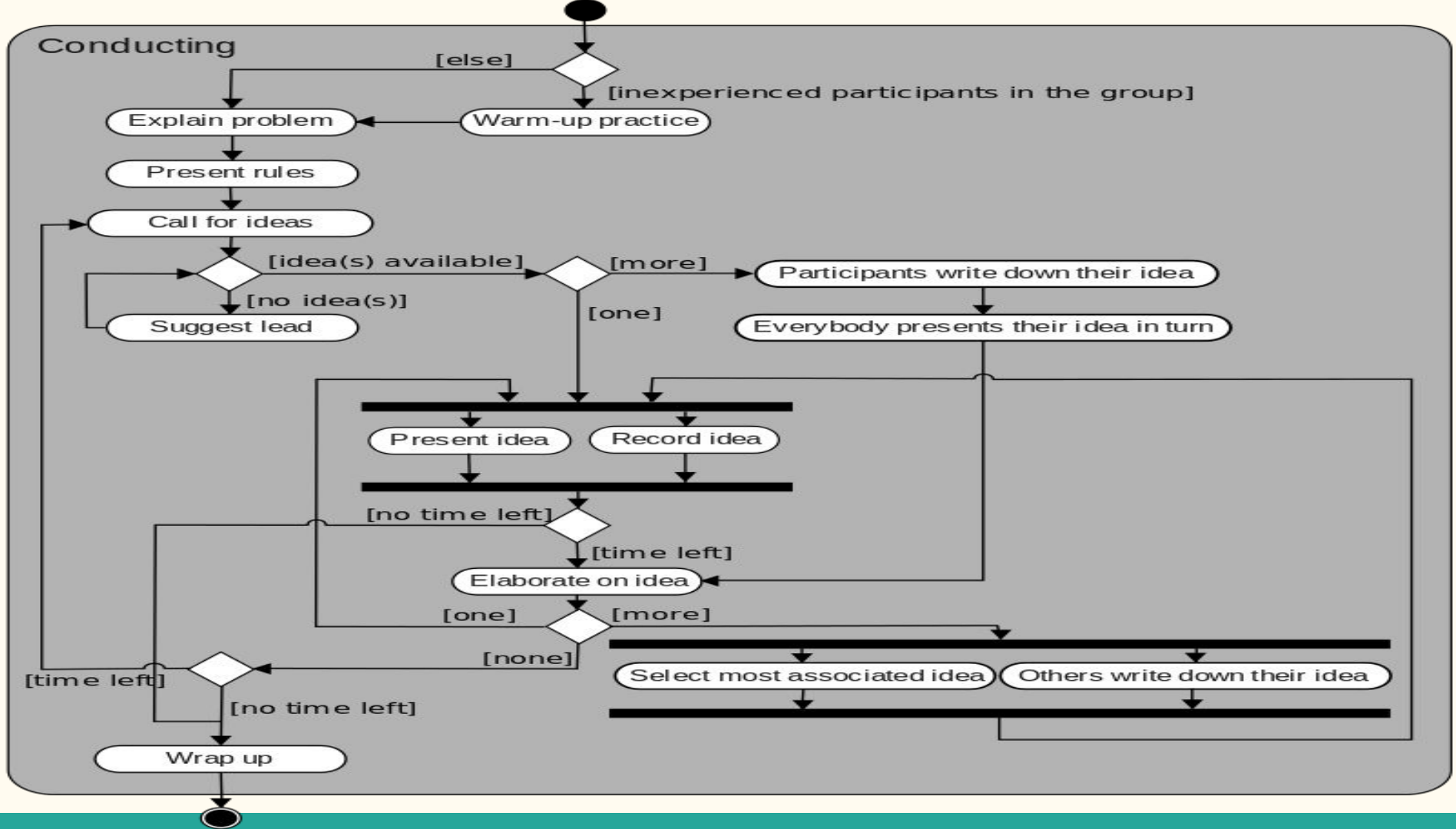
Merge event

Elements of activity diagram



Fork Node

Join Node

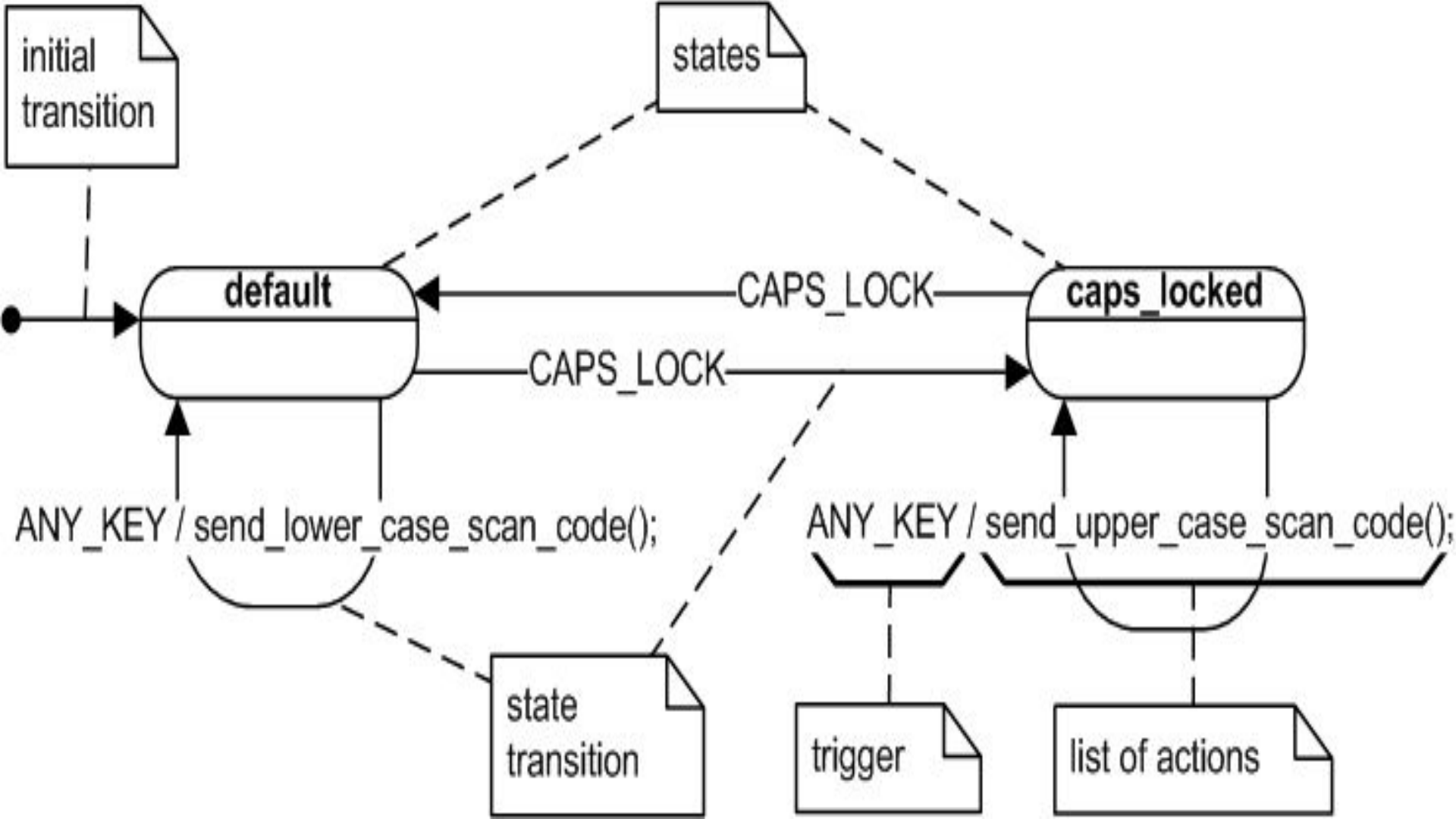

Conducting
[else]
[inexperienced participants in the group]
Explain problem
Warm-up practice
Present rules
Call for ideas
[idea(s) available]
[no idea(s)]
Suggest lead
[more]
[one]
Participants write down their idea
Everybody presents their idea in turn
Present idea
Record idea
[no time left]
[time left]
Elaborate on idea
[one]
[more]
[none]
[time left]
[no time left]
Select most associated idea
Others write down their idea
Wrap up

State machine diagram

- UML State diagrams is a enhanced realization of concept of finite state automata
- State chart diagram describe the flow of control from one state to another triggered by some action
- State diagrams are used to model reactive systems
- Also used for forward and reverse engineering
- State diagrams are basically directed graphs where nodes represent states and connectors represent transitions

Elements of state machine

- EVENTS: Any trigger or stimuli
- STATES: Well defined behaviour of any event
- GUARD CONDITIONS: Condition based flow
- ACTIONS: Response to the events
- TRANSITIONS: Switch from one state to another on some trigger


initial transition
states
default
CAPS_LOCK
caps_locked
CAPS_LOCK
ANY_KEY / send_lower_case_scan_code();
ANY_KEY / send_upper_case_scan_code();
state transition
trigger
list of actions

Use case diagram

- Use case diagrams depict what are the actions that system can perform in collaboration of external user(s)
- Use case diagrams consist of actors, use cases, and their relationships
- Use case diagrams describes the events and their flows, but never describes how they are implemented. It is like a black box

Elements of use case diagram

Use case

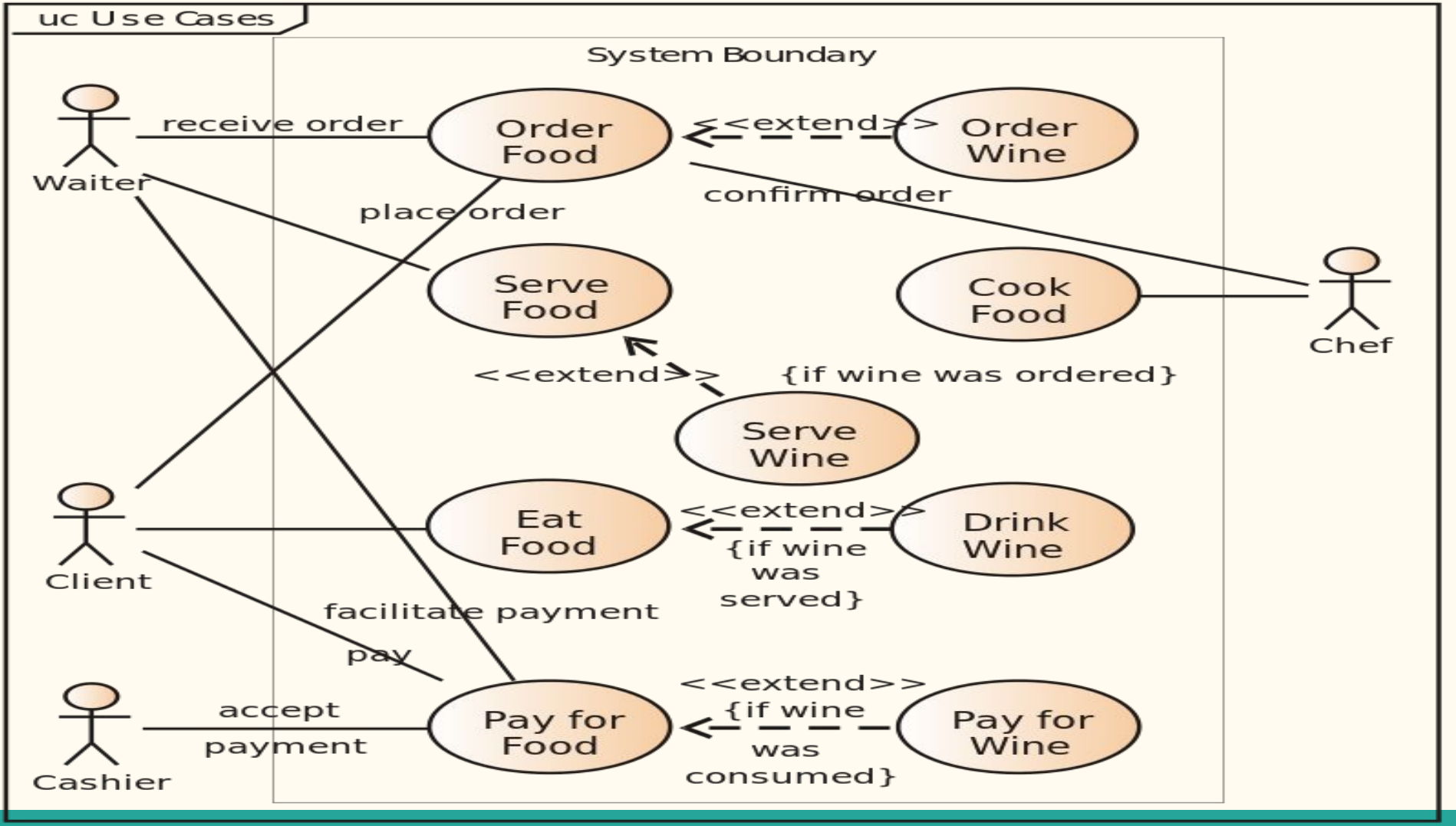
Black box of functionality

Actors

Humans, Internal or external influence

Relationships

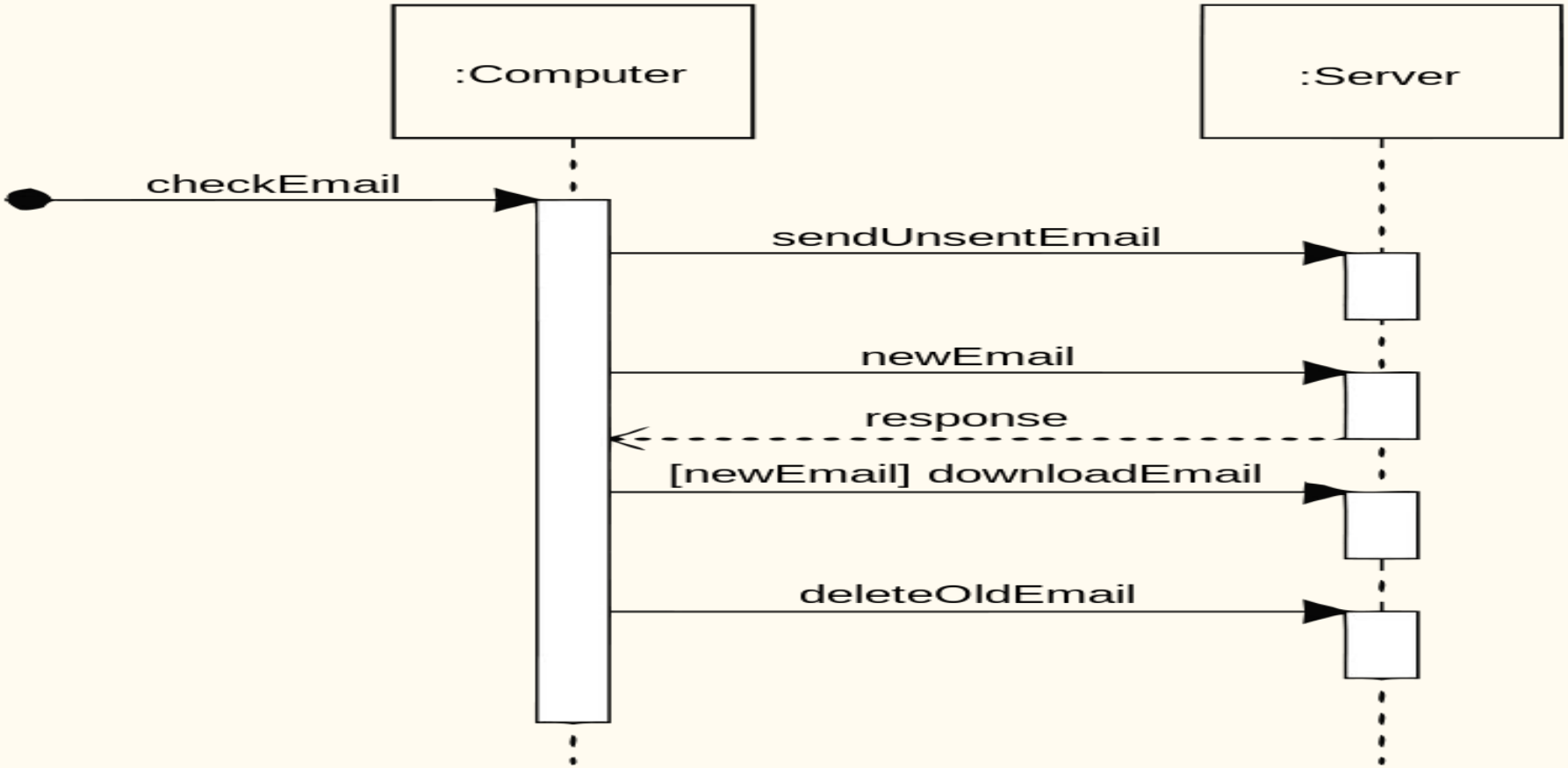
Connection between use cases


uc Use Cases
System Boundary
Waiter
receive order
Order Food
<<extend>>
Order Wine
confirm order
place order
Serve Food
Cook Food
Chef
<<extend>>
{if wine was ordered}
Serve Wine
Client
Eat Food
<<extend>>
Drink Wine
{if wine was served}
facilitate payment
pay
Cashier
accept payment
Pay for Food
<<extend>>
{if wine was consumed}
Pay for Wine

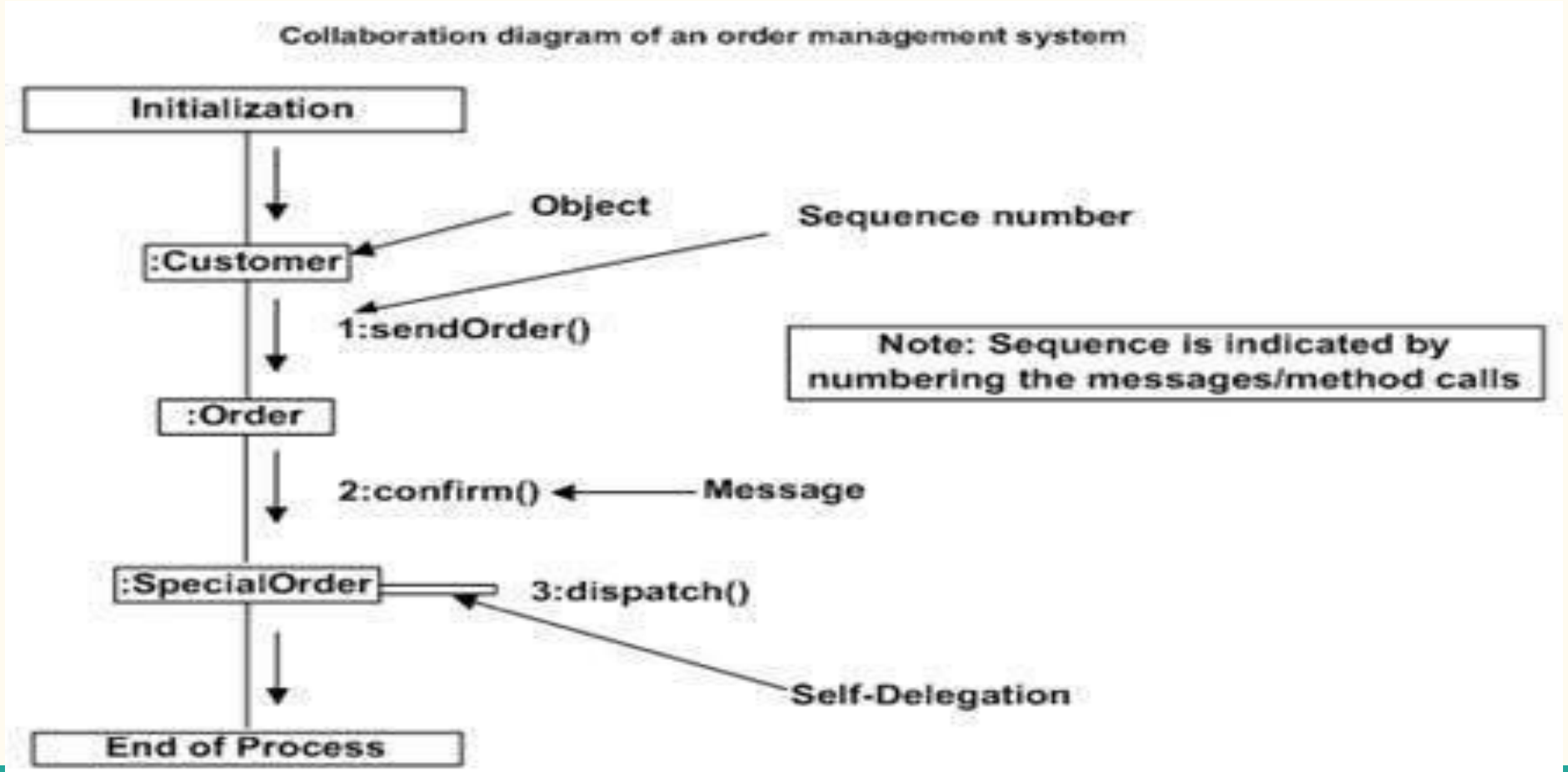
Interaction diagram

- Interaction diagrams are used when we want to understand the sequence of control flow
- It can be subcategorized into:
    - Sequence diagram
    - Collaboration diagram

Sequence diagram



Collaboration diagram



Uses of Behavioural diagrams

- **Activity Diagrams** : For concurrent events
- **State Diagrams** : When states and actions are well defined
- **Use case Diagrams** : For high level and practical understanding
- **Interaction Diagrams** : When sequence and organisation is important

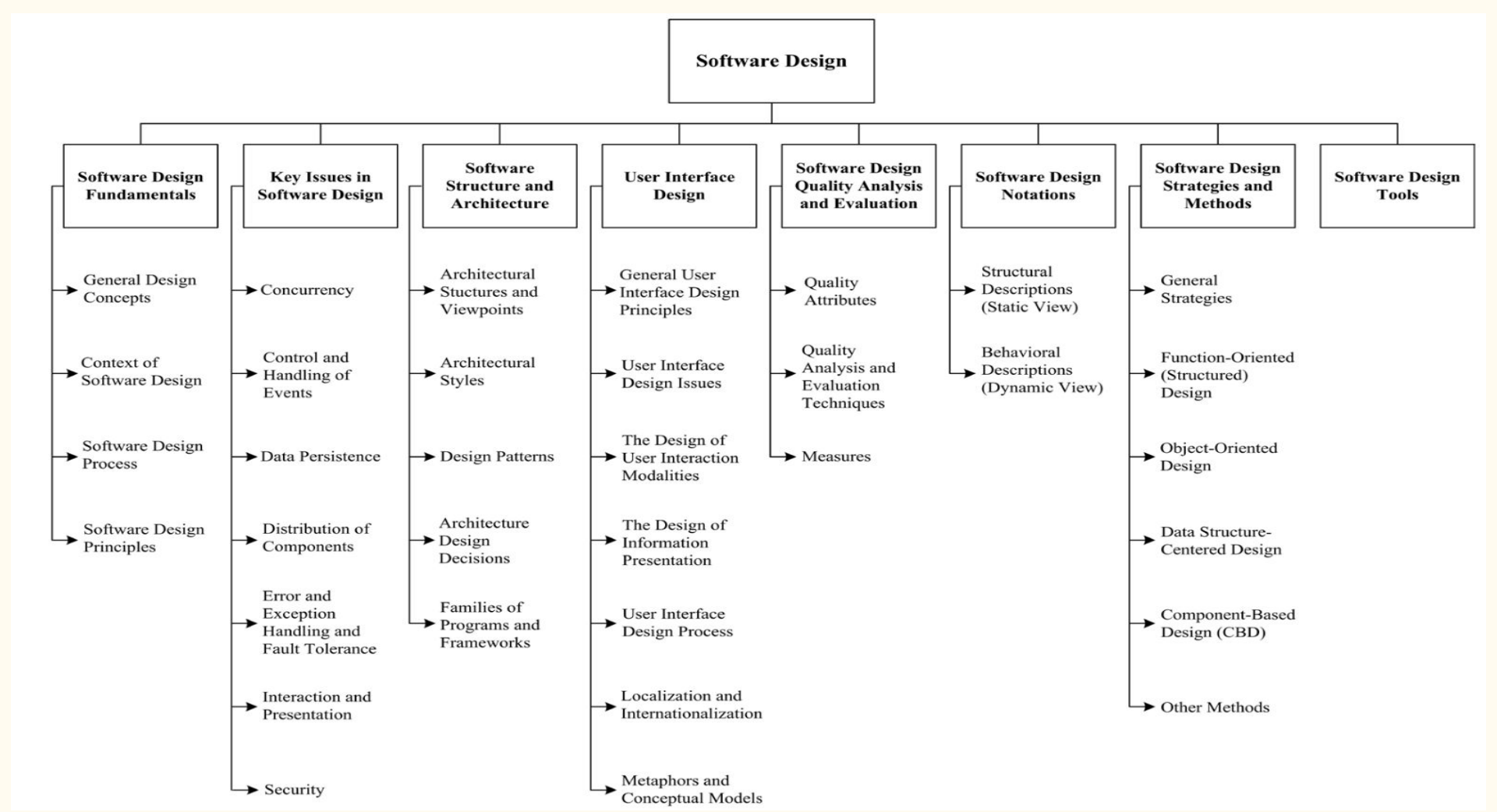
Tools for UML diagram design

- Visual Paradigm
- StarUML
- ArgoUML
- Umbrello UML
- MagicDraw
- Papyrus

Software Design

Karan Kumar Singh
(173050014)

Sourabh Pal
(173050054)


Software Design
Software Design Fundamentals
General Design Concepts
Context of Software Design
Software Design Process
Software Design Principles
Key Issues in Software Design
Concurrency
Control and Handling of Events
Data Persistence
Distribution of Components
Error and Exception Handling and Fault Tolerance
Interaction and Presentation
Security
Software Structure and Architecture
Architectural Stuctures and Viewpoints
Architectural Styles
Design Patterns
Architecture Design Decisions
Families of Programs and Frameworks
User Interface Design
General User Interface Design Principles
User Interface Design Issues
The Design of User Interaction Modalities
The Design of Information Presentation
User Interface Design Process
Localization and Internationalization
Metaphors and Conceptual Models
Software Design Quality Analysis and Evaluation
Quality Attributes
Quality Analysis and Evaluation Techniques
Measures
Software Design Notations
Structural Descriptions (Static View)
Behavioral Descriptions (Dynamic View)
Software Design Strategies and Methods
General Strategies
Function-Oriented (Structured) Design
Object-Oriented Design
Data Structure-Centered Design
Component-Based Design (CBD)
Other Methods
Software Design Tools

Containers

1. What are Containers?

3. How it works?

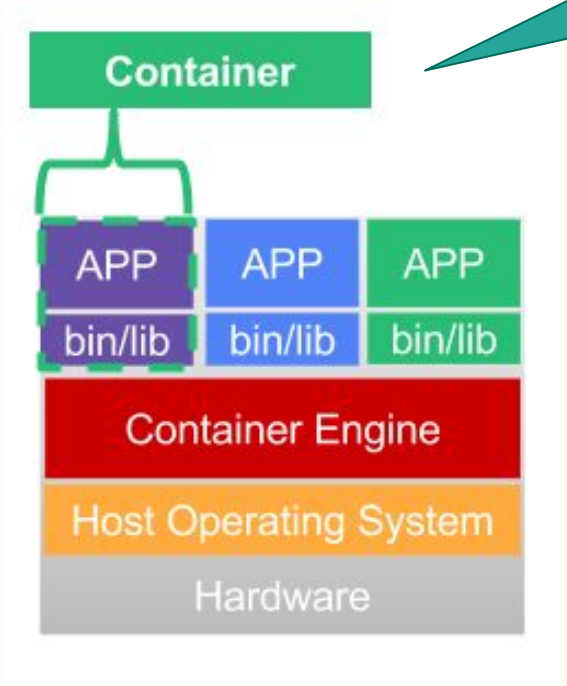
4. Docker example

2. Why use Containers?

5. Containers vs VMs

CONTAINERS

Partitions, jails or virtual environment

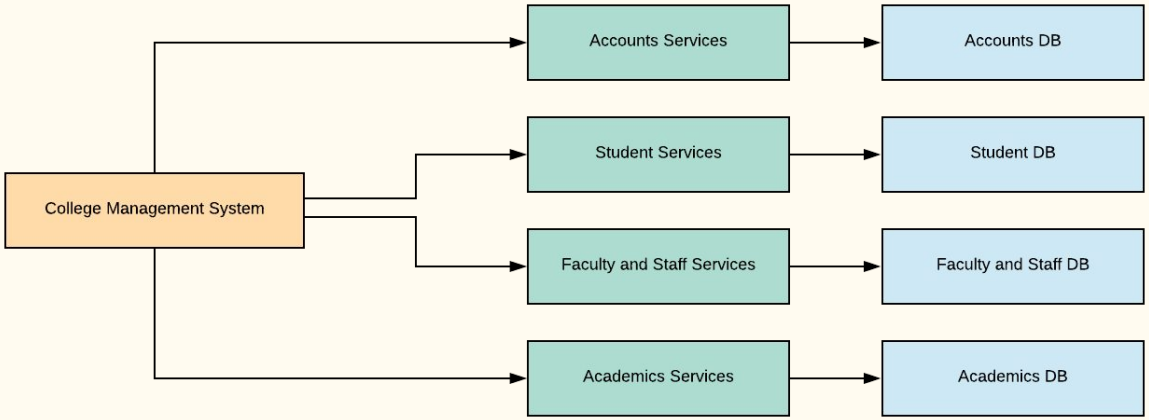


- It is an isolated user space in which computer programs run directly on the host operating system's kernel but have access to a restricted subset of its resources
- It is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

Before Containers

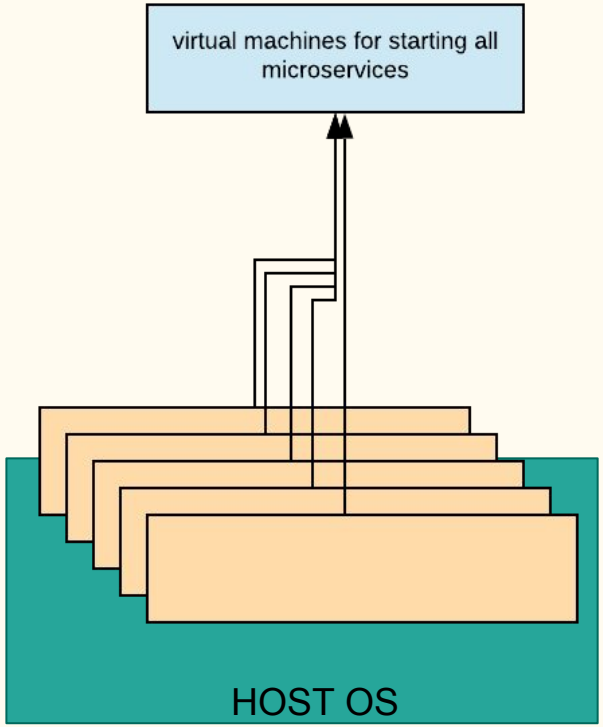
Microservices

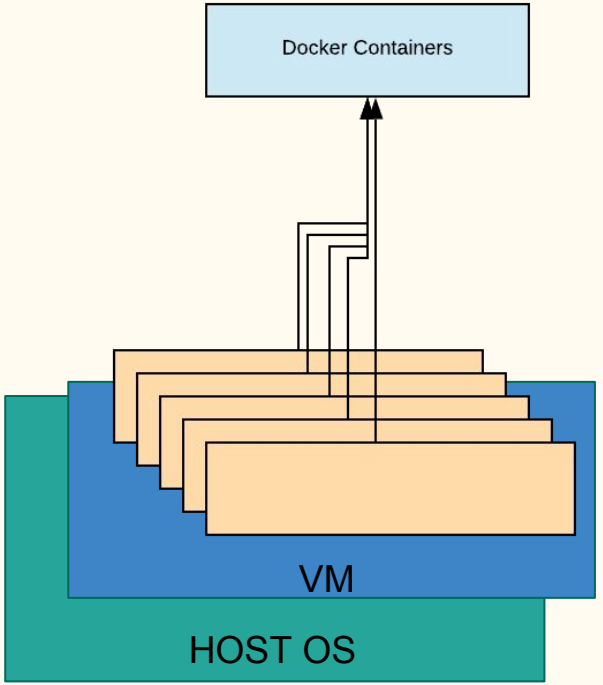
Application can be easier to built and maintain when broken into smaller services. Each component developed separately and application will be sum of its constituent components.
Ex: College management system



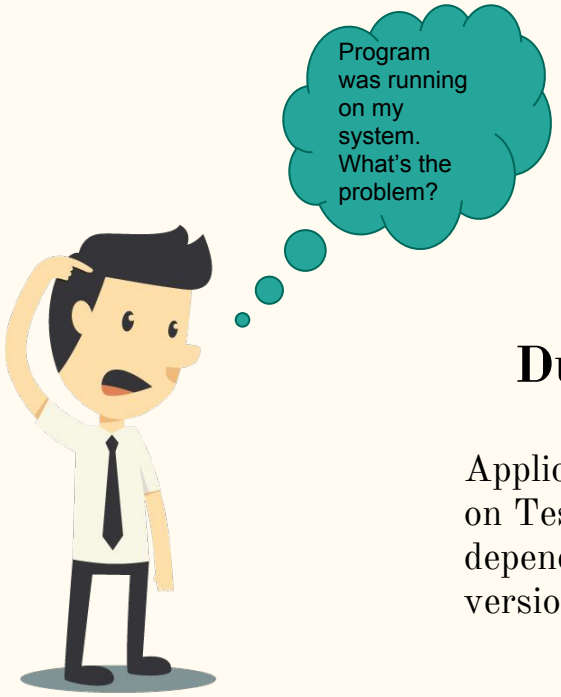
Problem solved

1. Updating module
2. If a service breaks down, whole service affected.


virtual machines for starting all microservices
HOST OS


Docker Containers
VM
HOST OS

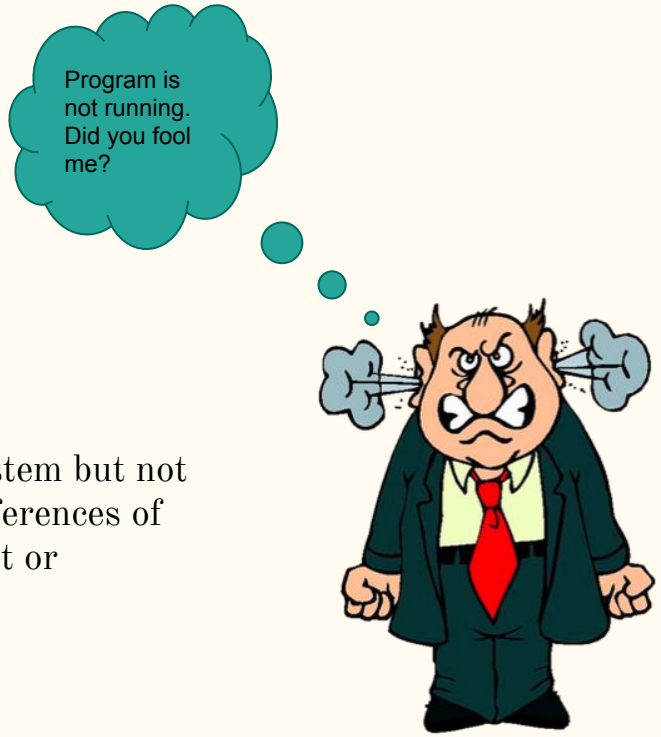
Before Containers



Developer

During SDLC

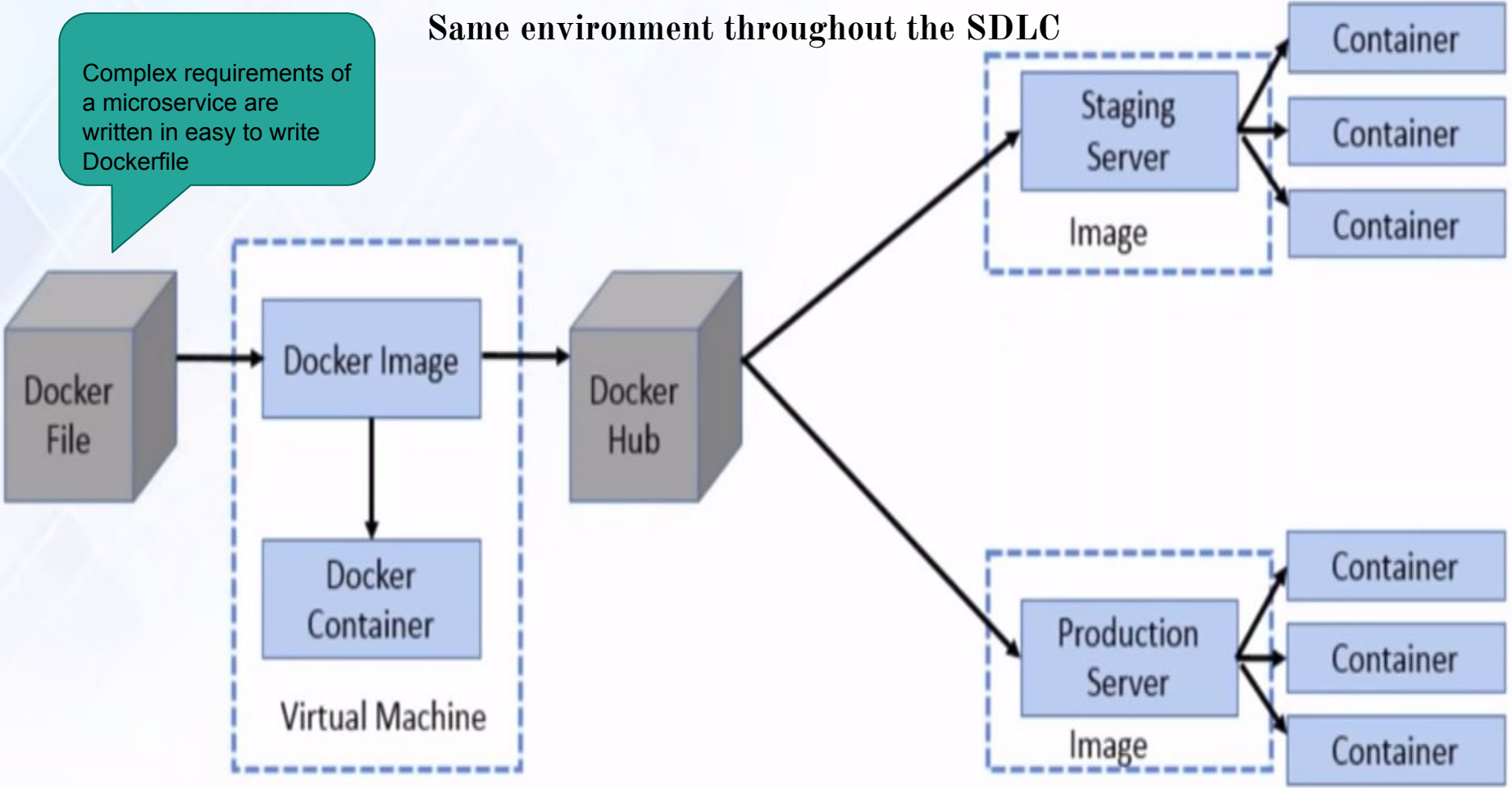
Application run on dev's system but not on Test and Prod due to differences of dependencies or environment or versions



Tester

Same environment throughout the SDLC

Complex requirements of a microservice are written in easy to write Dockerfile



Container Platforms

Linux VServers

- Linux-VServer is a sort of jail mechanism
- It can be used to securely partition resources on a computer system (such as the file system, CPU time, network addresses and memory)
- Unable to implement live migration and check point

OpenVZ

- OpenVZ container has its own isolated set of resources
- Each container act as a standalone server
- Each container has its own semaphores, shared memory segment, message, network stack, firewall rules etc.
- Unlike Linux VServers, check points and live migration is possible in OpenVZ


LXC


Windows Server 2016


rkt

Docker

Docker

- Dockers are lightweight containers that packages your application and all its dependencies together in the form of a docker container to ensure that your application works seamlessly in any environment.
- In a nutshell, dockers are lightweight container to provide process isolation
- Docker is a tool that is designed to benefit both developers and system administrators(DevOps)
- Docker gives flexibility and potentially reduces the number of systems needed because of its small footprint and lower overhead.

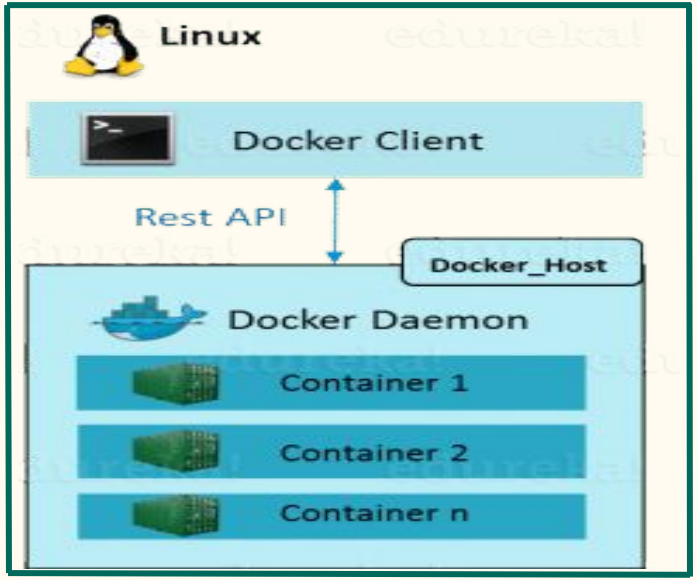
An Example

A company needs to develop a Java Application. In order to do so the developer will setup an environment with tomcat server installed in it. Once the application is developed, it needs to be tested by the tester. Now the tester will again setup tomcat environment from the scratch to test the application. Once the application testing is done, it will be deployed on the production server. Again the production needs an environment with tomcat installed on it, so that it can host the Java application. If you see the same tomcat environment setup is done thrice.

Docker Terminology

Docker Engine

Docker engine or Docker is a client-server application that builds and executes containers using Docker components. Docker engine creates and run dockers.



Docker Image

Images are like templates.

It let docker create and share the softwares. It's like a blueprint.

Docker Container

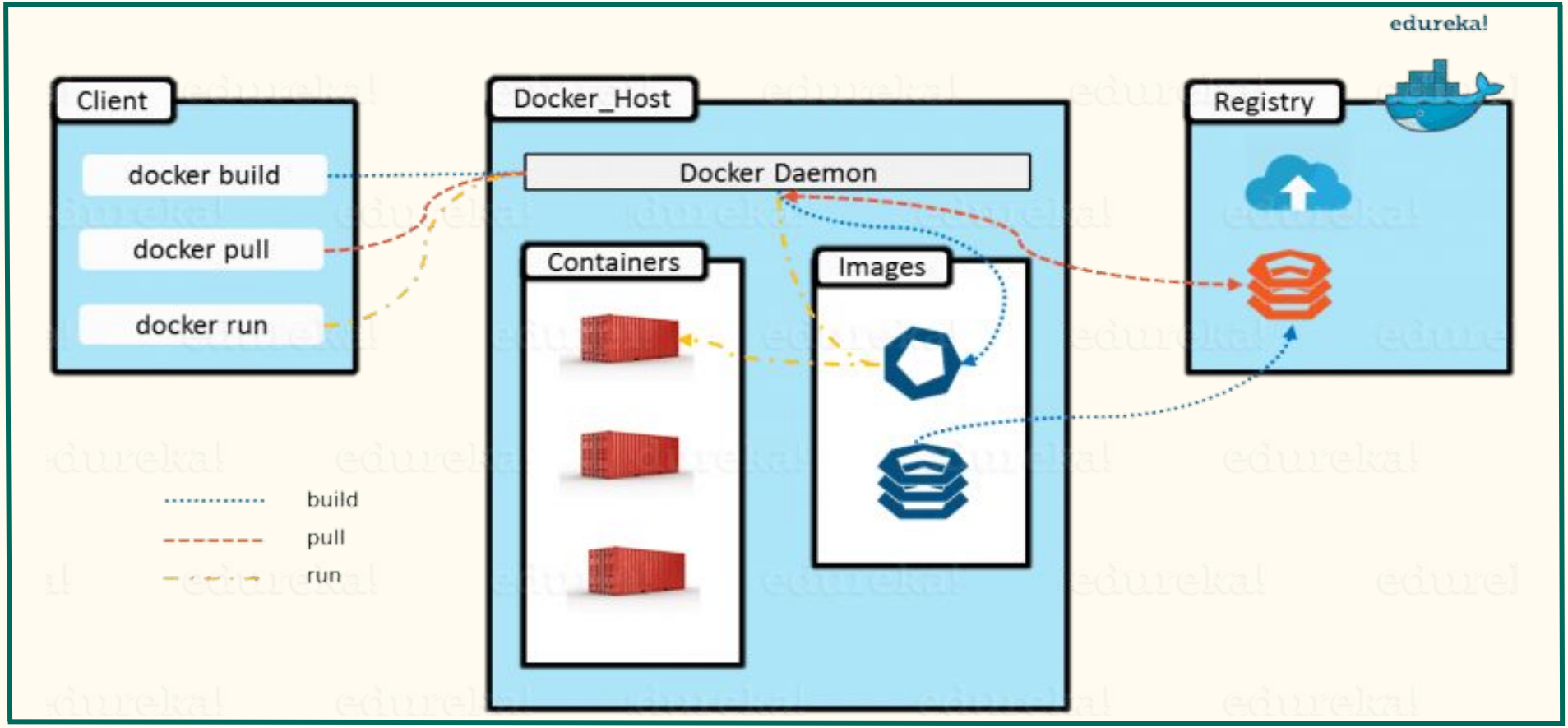
It is the running instance of the docker image and holds the entire package needed to run the application.

Docker Registry

Docker Registry is where the Docker Images are stored.

Registry can be either a user's local repository or a public repository like a Docker Hub allowing multiple users to collaborate in building an application.

Docker Architecture



- To build a Docker Image, we can use the CLI (client) to issue a build command to the Docker Daemon (running on Docker_Host). The Docker Daemon will then build an image based on our inputs and save it in the Registry, which can be either Docker hub or a local repository
- If we do not want to create an image, then we can just pull an image from the Docker hub, which would have been built by a different user
- Finally, if we have to create a running instance of my Docker image, we can issue a run command from the CLI, which will create a Docker Container.

Docker Implementation

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  android-libadb android-libbase android-libcutils android-liblog
  ca-certificates-mono cli-common libgdiplus libglade2.0-cil libglib2.0-cil
  libgsoap8 libgtk2.0-cil libllvm4.0 libmono-cairo4.0-cil
  libmono-corlib4.5-cil libmono-i18n-west4.0-cil libmono-i18n4.0-cil
  libmono-posix4.0-cil libmono-security4.0-cil
  libmono-system-configuration4.0-cil libmono-system-drawing4.0-cil
  libmono-system-security4.0-cil libmono-system-xml4.0-cil
  libmono-system4.0-cil libqt4-opengl libvncserver1 linux-headers-4.10.0-28
  linux-headers-4.10.0-28-generic linux-headers-4.10.0-32
  linux-headers-4.10.0-32-generic linux-headers-4.10.0-35
  linux-headers-4.10.0-35-generic linux-headers-4.10.0-37
  linux-headers-4.10.0-37-generic linux-headers-4.10.0-38
  linux-headers-4.10.0-38-generic linux-headers-4.10.0-40
  linux-headers-4.10.0-40-generic linux-headers-4.10.0-42
  linux-headers-4.10.0-42-generic linux-headers-4.13.0-26
  linux-headers-4.13.0-26-generic linux-headers-4.13.0-32
  linux-headers-4.13.0-32-generic linux-headers-4.13.0-36
  linux-headers-4.13.0-36-generic linux-headers-4.13.0-37
  linux-headers-4.13.0-37-generic linux-headers-4.13.0-38
  linux-headers-4.13.0-38-generic linux-headers-4.13.0-39
  linux-headers-4.13.0-39-generic linux-headers-4.13.0-41
  linux-headers-4.13.0-41-generic linux-headers-4.13.0-43
  linux-headers-4.13.0-43-generic linux-headers-4.15.0-29
  linux-headers-4.15.0-29-generic linux-headers-4.15.0-30
  linux-headers-4.15.0-30-generic linux-headers-4.15.0-32
  linux-headers-4.15.0-32-generic linux-headers-4.15.0-33
  linux-headers-4.15.0-33-generic linux-headers-4.15.0-34
  linux-headers-4.15.0-34-generic linux-headers-4.15.0-36
  linux-headers-4.15.0-36-generic linux-headers-4.15.0-39
  linux-headers-4.15.0-39-generic linux-headers-4.15.0-42
  linux-headers-4.15.0-42-generic linux-headers-4.15.0-43
  linux-headers-4.15.0-43-generic linux-image-4.10.0-28-generic
  linux-image-4.10.0-32-generic linux-image-4.10.0-35-generic
  linux-image-4.10.0-37-generic linux-image-4.10.0-38-generic
  linux-image-4.10.0-40-generic linux-image-4.10.0-42-generic
  linux-image-4.13.0-26-generic linux-image-4.13.0-32-generic
  linux-image-4.13.0-36-generic linux-image-4.13.0-37-generic
  linux-image-4.13.0-38-generic linux-image-4.13.0-39-generic
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker info
Containers: 0
 Running: 0
 Paused: 0
 Stopped: 0
Images: 0
Server Version: 18.09.2
Storage Driver: overlay2
 Backing Filesystem: extfs
 Supports d_type: true
 Native Overlay Diff: true
Logging Driver: json-file
Cgroup Driver: cgroupfs
Plugins:
 Volume: local
 Network: bridge host macvlan null overlay
 Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
Swarm: inactive
Runtimes: runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 9754871865f7fe2f4e74d43e2fc7ccd237edcbce
runc version: 09c8266bf2fcf9519a651b04ae54c967b9ab86ec
init version: v0.18.0 (expected: fec3683b971d9c3ef73f284f176672c44b448662)
Security Options:
 apparmor
 seccomp
  Profile: default
Kernel Version: 4.15.0-46-generic
Operating System: Ubuntu 16.04.3 LTS
OSType: linux
Architecture: x86_64
CPUs: 4
Total Memory: 3.784GiB
Name: sourabh-HP-Pavilion-15-Notebook-PC
ID: UZCN:3O5Z:WW2L:3FZN:DBTQ:3543:XVTM:5RQQ:NUIK:Z7HS:ONTP:4NOS
Docker Root Dir: /var/lib/docker
Debug Mode (client): false
Debug Mode (server): false
Registry: https://index.docker.io/v1/
Labels:
Experimental: false
Insecure Registries:
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't
have a Docker ID, head over to https://hub.docker.com to create one.
Username: 173050054
Password:
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't
have a Docker ID, head over to https://hub.docker.com to create one.
Username: 173050054
Password:
WARNING! Your password will be stored unencrypted in /home/sourabh/.docker/confi
g.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:2557e3c07ed1e38f26e389462d03ed943586f744621577a99efb77324b0fe535
Status: Downloaded newer image for hello-world:latest
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker image

Usage:  docker image COMMAND

Manage images

Commands:
  build       Build an image from a Dockerfile
  history     Show the history of an image
  import      Import the contents from a tarball to create a filesystem image
  inspect     Display detailed information on one or more images
  load        Load an image from a tar archive or STDIN
  ls          List images
  prune       Remove unused images
  pull        Pull an image or a repository from a registry
  push        Push an image or a repository to a registry
  rm          Remove one or more images
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
hello-world         latest              fce289e99eb9        3 months ago
1.84kB
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
27833a3ba0a5: Pull complete
e83729dd399a: Pull complete
ebc6a67df66d: Pull complete
Digest: sha256:c8a861b8a1eeef6d48955a6c6d5dff8e2580f13ff4d0f549e082e7c82a8617a2
Status: Downloaded newer image for nginx:latest
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
nginx               latest              2bcb04bdb83f        6 days ago
109MB
ubuntu              latest              94e814e2efa8        3 weeks ago
88.9MB
hello-world         latest              fce289e99eb9        3 months ago
1.84kB
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker search nginx
NAME                                              DESCRIPTION                                     STARS               OFFICIAL
AUTOMATED
nginx                                             Official build of Nginx.                        11159               [OK]
jwilder/nginx-proxy                               Automated Nginx reverse proxy for docker con…   1575
[OK]
richarvey/nginx-php-fpm                           Container running Nginx + PHP-FPM capable of…   698
[OK]
jrcs/letsencrypt-nginx-proxy-companion            LetsEncrypt container to use with nginx as p…   495
[OK]
kitematic/hello-world-nginx                       A light-weight nginx container that demonstr…   124
webdevops/php-nginx                               Nginx with PHP-FPM                              123
[OK]
zabbix/zabbix-web-nginx-mysql                     Zabbix frontend based on Nginx web-server wi…   93
[OK]
bitnami/nginx                                     Bitnami nginx Docker Image                      65
[OK]
linuxserver/nginx                                 An Nginx container, brought to you by LinuxS…   58
1and1internet/ubuntu-16-nginx-php-phpmyadmin-mysql-5   ubuntu-16-nginx-php-phpmyadmin-mysql-5     50
[OK]
tobi312/rpi-nginx                                 NGINX on Raspberry Pi / armhf                   24
[OK]
nginx/nginx-ingress                               NGINX Ingress Controller for Kubernetes         17
schmunk42/nginx-redirect                          A very simple container to redirect HTTP tra…   13
[OK]
nginxdemos/hello                                  NGINX webserver that serves a simple page co…   13
[OK]
wodby/drupal-nginx                                Nginx for Drupal container image                12
[OK]
blacklabelops/nginx                               Dockerized Nginx Reverse Proxy Server.          12
[OK]
centos/nginx-18-centos7                           Platform for running nginx 1.8 or building n…   10
centos/nginx-112-centos7                          Platform for running nginx 1.12 or building …   7
nginxinc/nginx-unprivileged                       Unprivileged NGINX Dockerfiles                  4
1science/nginx                                    Nginx Docker images that include Consul Temp…   4
[OK]
mailu/nginx                                       Mailu nginx frontend                            3
[OK]
travix/nginx                                      NGinx reverse proxy                             2
[OK]
toccoag/openshift-nginx                           Nginx reverse proxy for Nice running on same…   1
[OK]
ansibleplaybookbundle/nginx-apb                   An APB to deploy NGINX                          0
```

Creating a Docker

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't
have a Docker ID, head over to https://hub.docker.com to create one.
Username: 173050054
Password:
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't
have a Docker ID, head over to https://hub.docker.com to create one.
Username: 173050054
Password:
WARNING! Your password will be stored unencrypted in /home/sourabh/.docker/confi
g.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

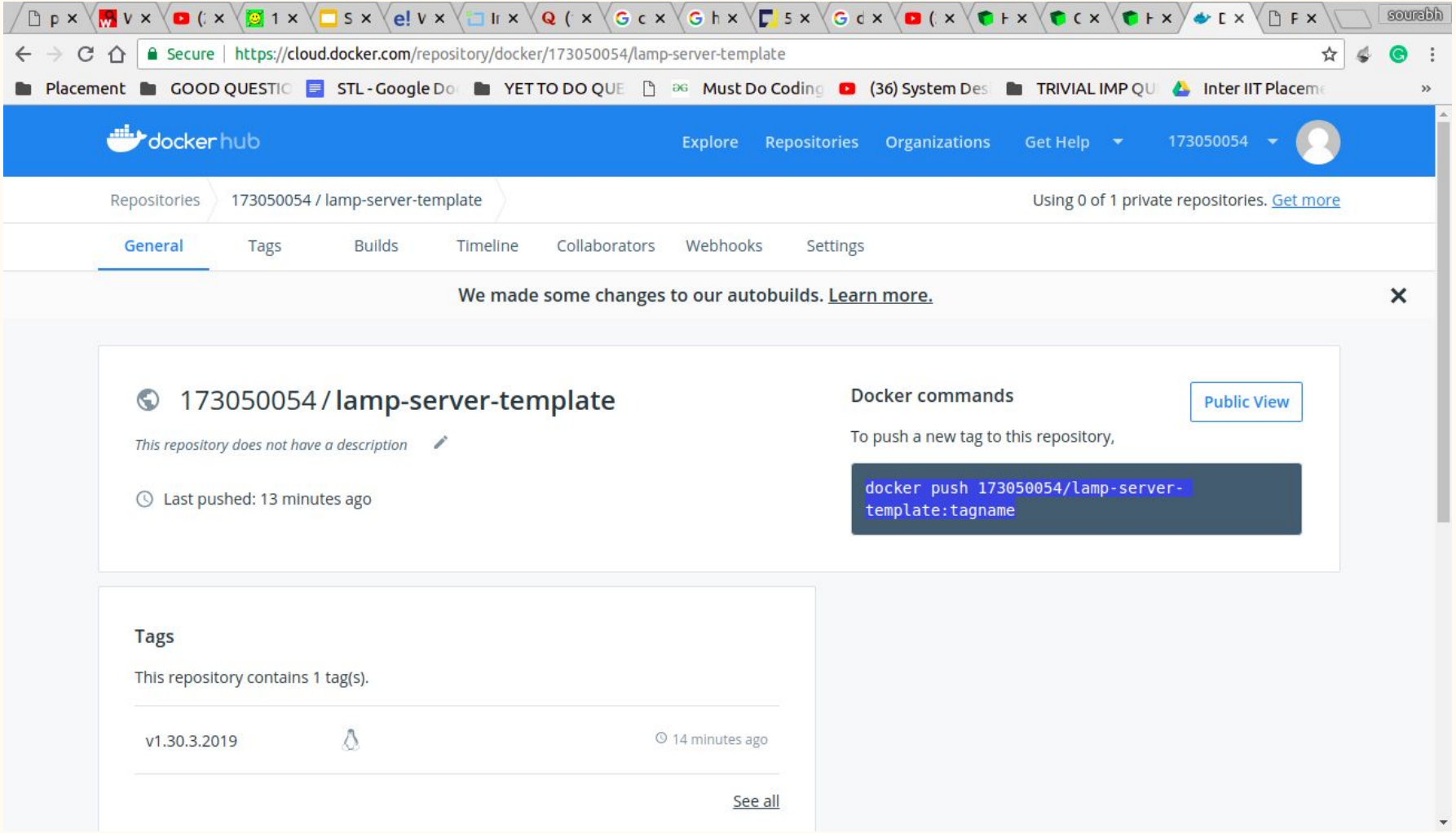
```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
898c46f3b1a1: Pull complete
63366dfa0a50: Pull complete
041d4cd74a92: Pull complete
6e1bee0f8701: Pull complete
Digest: sha256:017eef0b616011647b269b5c65826e2e2ebddbe5d1f8c1e56b3599fb14fabec8
Status: Downloaded newer image for ubuntu:latest
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker run --name lamp-server
-template -it ubuntu:latest bash
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
nginx               latest              2bcb04bdb83f        6 days ago
109MB
ubuntu              latest              94e814e2efa8        3 weeks ago
88.9MB
hello-world         latest              fce289e99eb9        3 months ago
1.84kB
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS                      PORTS               NAMES
3d8391010584        ubuntu:latest       "bash"              2 minutes ago
Exited (100) About a minute ago                 Sourabh
2468f5f49038        ubuntu:latest       "bash"              9 minutes ago
Exited (100) 8 minutes ago                      lamp-server-template
18a5532c3afd        hello-world         "/hello"            42 minutes ago
Exited (0) 42 minutes ago                       trusting_chaplygin
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS                      PORTS               NAMES
3d8391010584        ubuntu:latest       "bash"              2 minutes ago
Exited (100) About a minute ago                 Sourabh
2468f5f49038        ubuntu:latest       "bash"              9 minutes ago
Exited (100) 8 minutes ago                      lamp-server-template
18a5532c3afd        hello-world         "/hello"            42 minutes ago
Exited (0) 42 minutes ago                       trusting_chaplygin
```

```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC: ~
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker commit 2468f5f49038 la
mp-server-template:v1.30.3.2019
sha256:d4d953fde2de42a0a9b1ae30ce2bc5f451a50df7d0cfad60cd4ece5e00a94777
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker images
REPOSITORY              TAG                 IMAGE ID            CREATED
   SIZE
lamp-server-template    v1.30.3.2019        d4d953fde2de        14 seconds ago
   88.9MB
lamp-server-template    latest              bb3b64efd265        4 minutes ago
   88.9MB
nginx                   latest              2bcb04bdb83f        6 days ago
   109MB
ubuntu                  latest              94e814e2efa8        3 weeks ago
   88.9MB
hello-world             latest              fce289e99eb9        3 months ago
   1.84kB
```

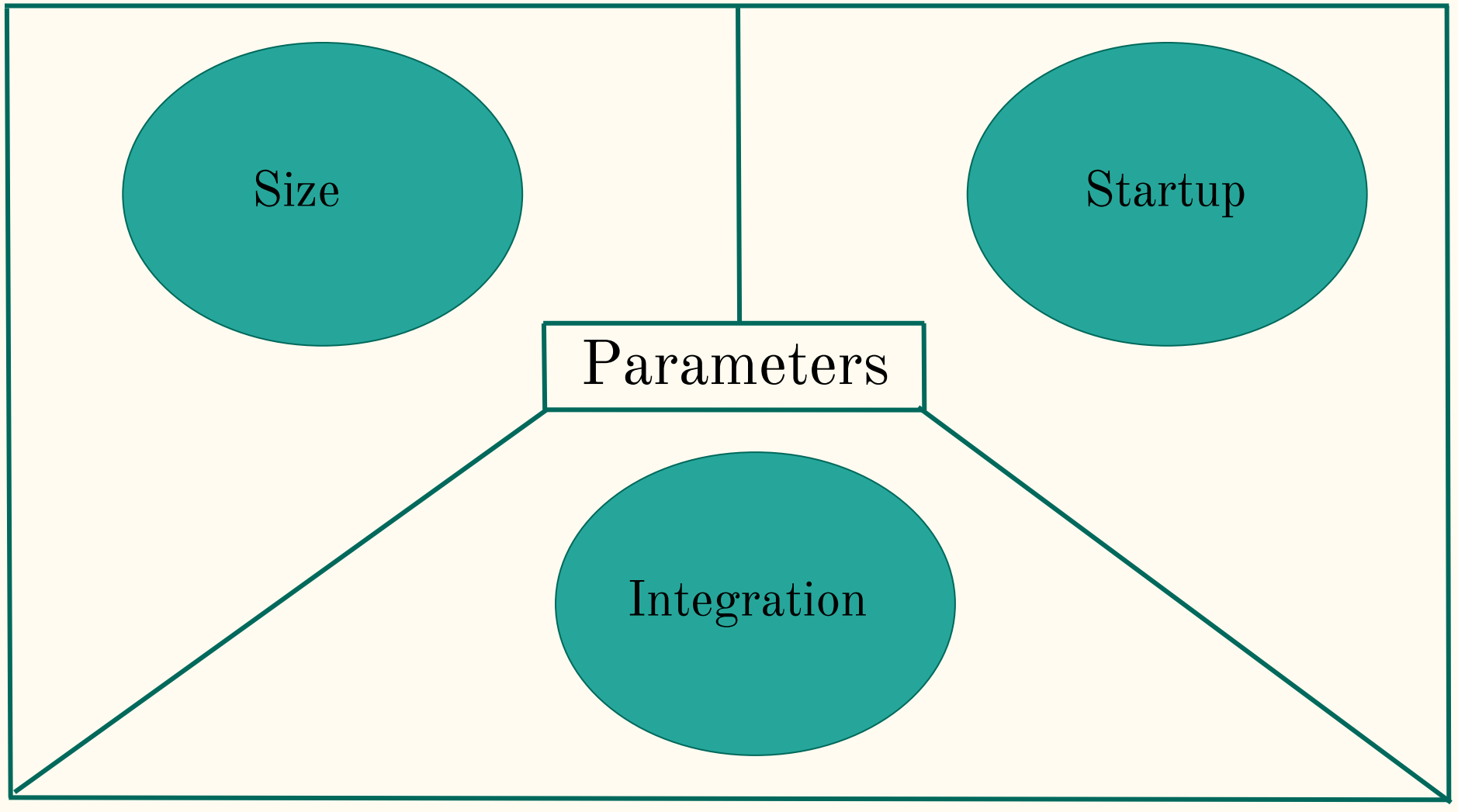
```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker push lamp-server-templ
ate:v1.30.3.2019
The push refers to repository [docker.io/library/lamp-server-template]
f919699c34cf: Preparing
b57c79f4a9f3: Preparing
d60e01b37e74: Preparing
e45cfbc98a50: Preparing
762d8e1a6054: Preparing
denied: requested access to the resource is denied
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker tag lamp-server-templa
te:v1.30.3.2019 173050054/lamp-server-template:v1.30.3.2019
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker push 173050054/lamp-se
rver-template:v1.30.3.2019
The push refers to repository [docker.io/173050054/lamp-server-template]
f919699c34cf: Pushed
b57c79f4a9f3: Pushed
d60e01b37e74: Pushed
e45cfbc98a50: Pushed
762d8e1a6054: Pushed
v1.30.3.2019: digest: sha256:d43dfd28bed82279f2d1fe6e4eb4e071685364557f799d418c3
77ddb43c4d4d6 size: 1357
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$
```


Secure | https://cloud.docker.com/repository/docker/173050054/lamp-server-template
Placement
GOOD QUESTIO
STL - Google Do
YET TO DO QUE
Must Do Coding
(36) System Des
TRIVIAL IMP QU
Inter IIT Placeme
docker hub
Explore
Repositories
Organizations
Get Help
173050054
Repositories
173050054 / lamp-server-template
Using 0 of 1 private repositories. Get more
General
Tags
Builds
Timeline
Collaborators
Webhooks
Settings
We made some changes to our autobuilds. Learn more.
173050054 / lamp-server-template
This repository does not have a description
Last pushed: 13 minutes ago
Docker commands
Public View
To push a new tag to this repository,
docker push 173050054/lamp-server-template:tagname
Tags
This repository contains 1 tag(s).
v1.30.3.2019
14 minutes ago
See all

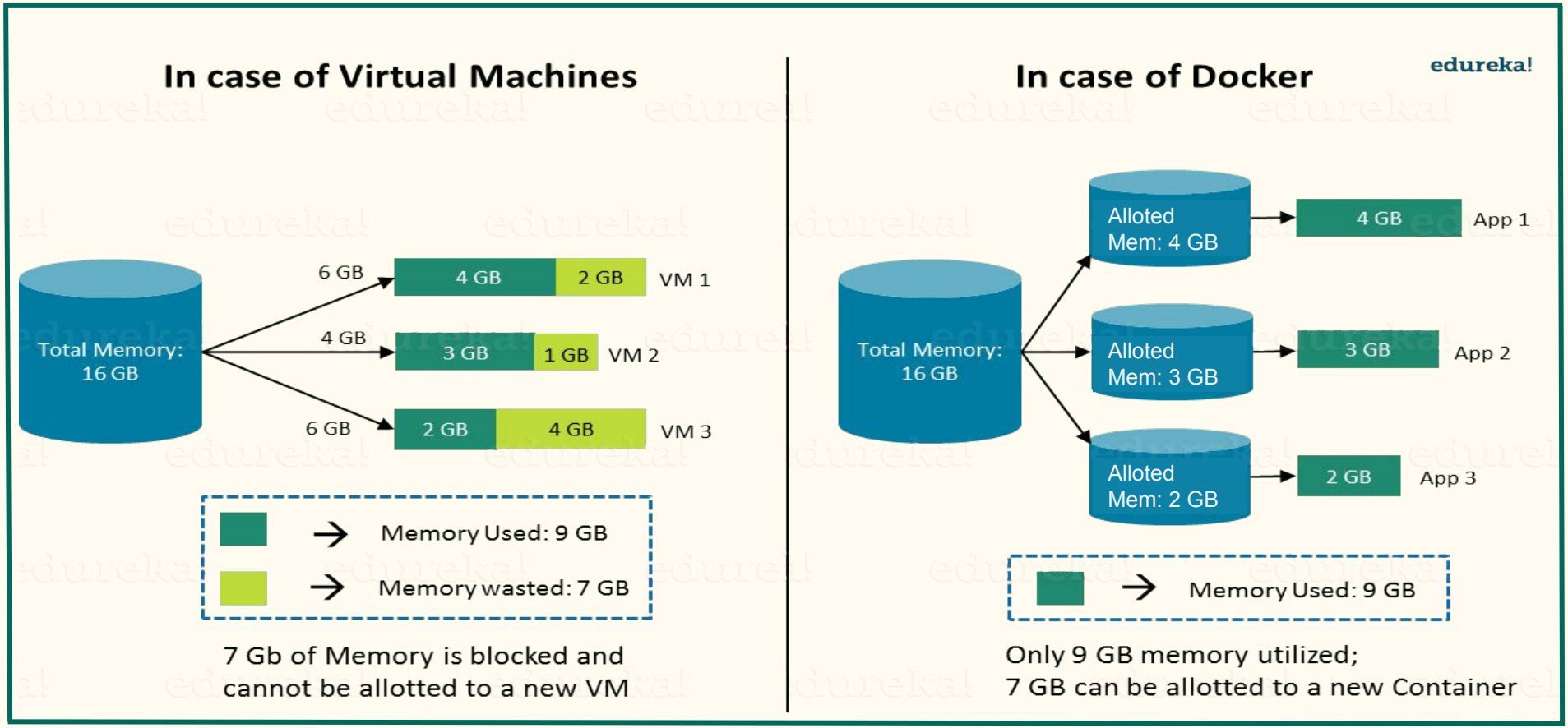
```
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker run 173050054/lamp-ser
ver-template
Unable to find image '173050054/lamp-server-template:latest' locally
docker: Error response from daemon: manifest for 173050054/lamp-server-template:
latest not found.
See 'docker run --help'.
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker pull 173050054/lamp-se
rver-template
Using default tag: latest
Error response from daemon: manifest for 173050054/lamp-server-template:latest n
ot found
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$ sudo docker pull 173050054/lamp-se
rver-template:v1.30.3.2019
v1.30.3.2019: Pulling from 173050054/lamp-server-template
Digest: sha256:d43dfd28bed82279f2d1fe6e4eb4e071685364557f799d418c377ddb43c4d4d6
Status: Image is up to date for 173050054/lamp-server-template:v1.30.3.2019
sourabh@sourabh-HP-Pavilion-15-Notebook-PC:~$
```

```
karan@divu:~$ sudo docker pull 173050054/lamp-server-template:v1.30.3.2019
v1.30.3.2019: Pulling from 173050054/lamp-server-template
898c46f3b1a1: Pull complete
63366dfa0a50: Pull complete
041d4cd74a92: Pull complete
6e1bee0f8701: Pull complete
effedca25304: Pull complete
Digest: sha256:d43dfd28bed82279f2d1fe6e4eb4e071685364557f799d418c377ddb43c4d4d6
Status: Downloaded newer image for 173050054/lamp-server-template:v1.30.3.2019
karan@divu:~$ sudo docker images
REPOSITORY                          TAG                 IMAGE ID            CREATED
             SIZE
173050054/lamp-server-template      v1.30.3.2019        d4d953fde2de        About a
n hour ago   88.9MB
```

Docker VS Virtual Machine

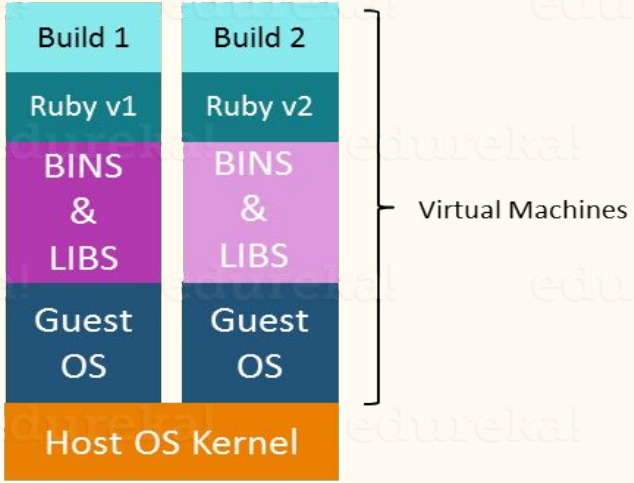

Size
Startup
Parameters
Integration

SIZE


In case of Virtual Machines
Total Memory: 16 GB
6 GB
4 GB
2 GB
VM 1
4 GB
3 GB
1 GB
VM 2
6 GB
2 GB
4 GB
VM 3
Memory Used: 9 GB
Memory wasted: 7 GB
7 Gb of Memory is blocked and cannot be allotted to a new VM
In case of Docker
edureka!
Total Memory: 16 GB
Alloted Mem: 4 GB
4 GB
App 1
Alloted Mem: 3 GB
3 GB
App 2
Alloted Mem: 2 GB
2 GB
App 3
Memory Used: 9 GB
Only 9 GB memory utilized;
7 GB can be allotted to a new Container

STARTUP

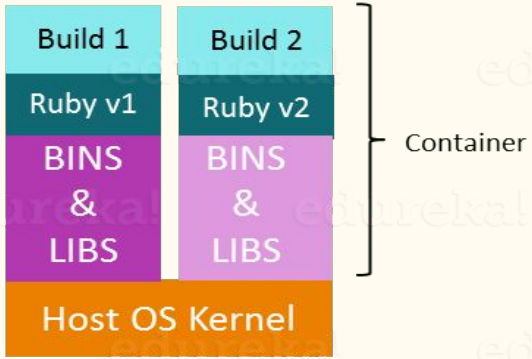
In case of Virtual Machines



New Builds → Multiple OS → Separate Libraries → Heavy → **More Time**

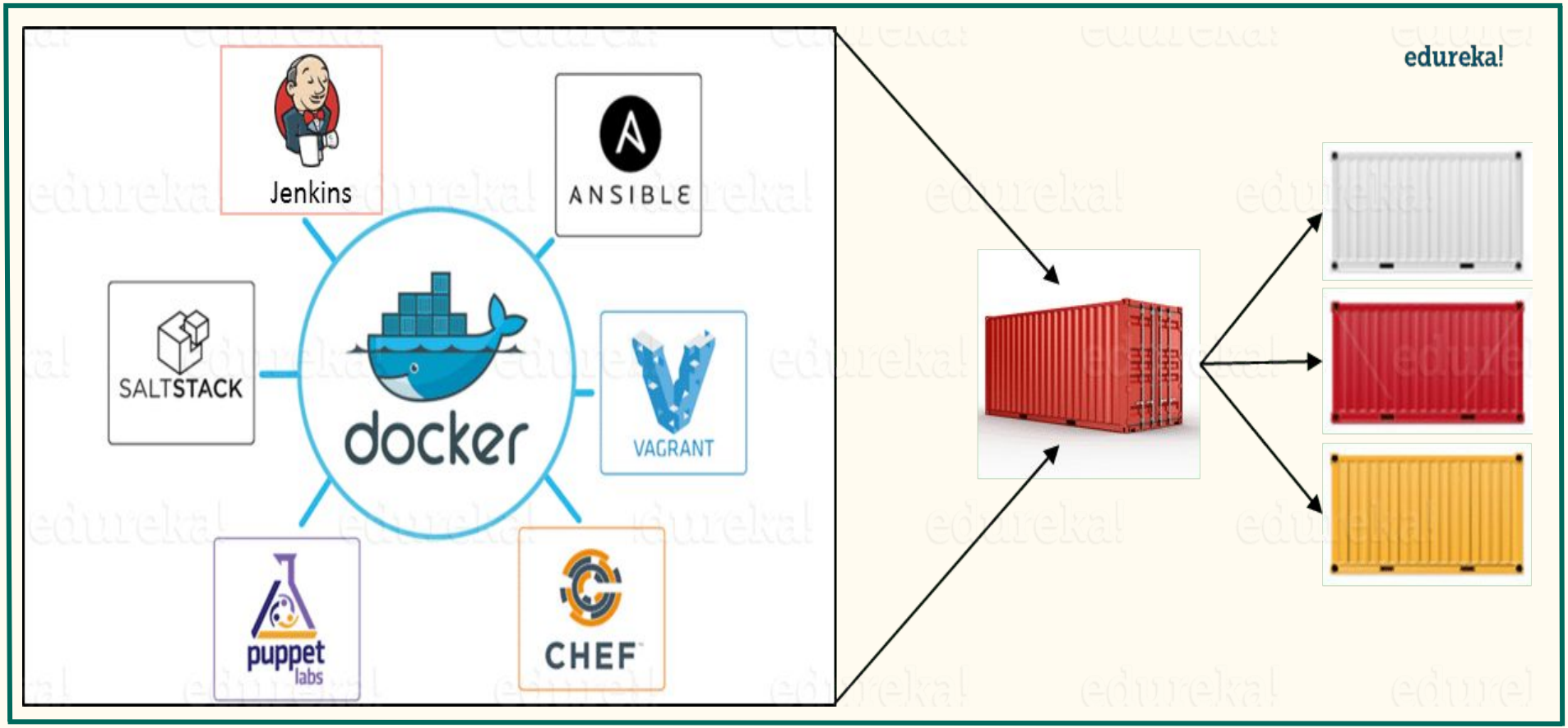
In case of Docker

edureka!



New Builds → Same OS → Separate Libraries → Lightweight → **Less Time**

INTEGRATION



PERFORMANCE ANALYSIS

- For small processes, containers always wins over VMs.
- Is it always true?
- What happen when there are hefty complex process?
- Will container still be better under complex situation?

A Comparative Study of Containers and Virtual Machines in Big Data Environment

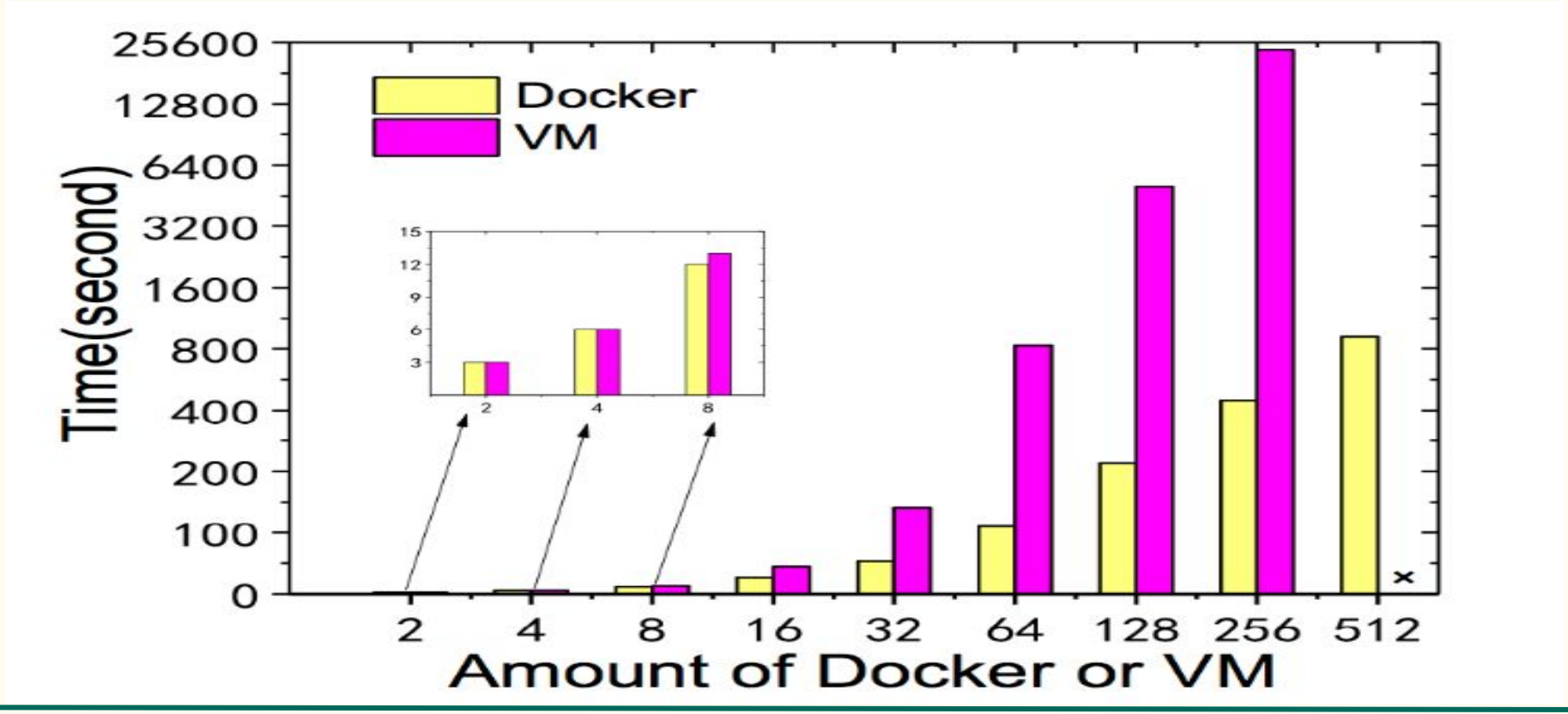
Qi Zhang[1], Ling Liu[2], Calton Pu[2], Qiwei Dou[3], Liren Wu[3], and Wei Zhou[3]

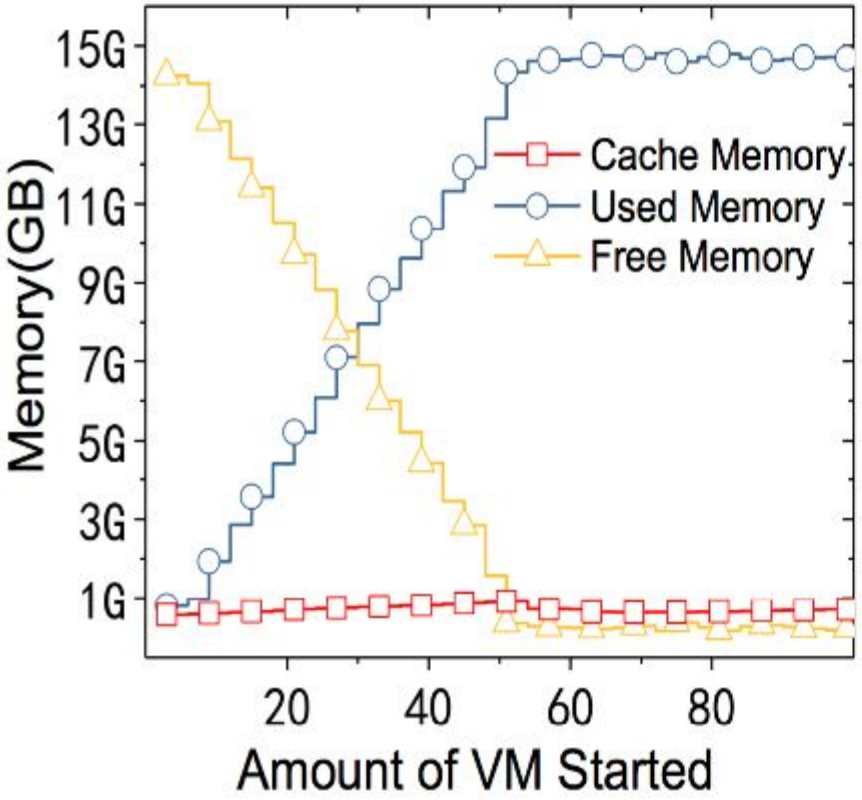
[1]IBM Thomas J. Watson Research, New York, USA
[2]College of Computing, Georgia Institute of Technology, Georgia, USA
[3]Department of Computer Science, Yunnan University, Yunnan, China

Deployment Convenience

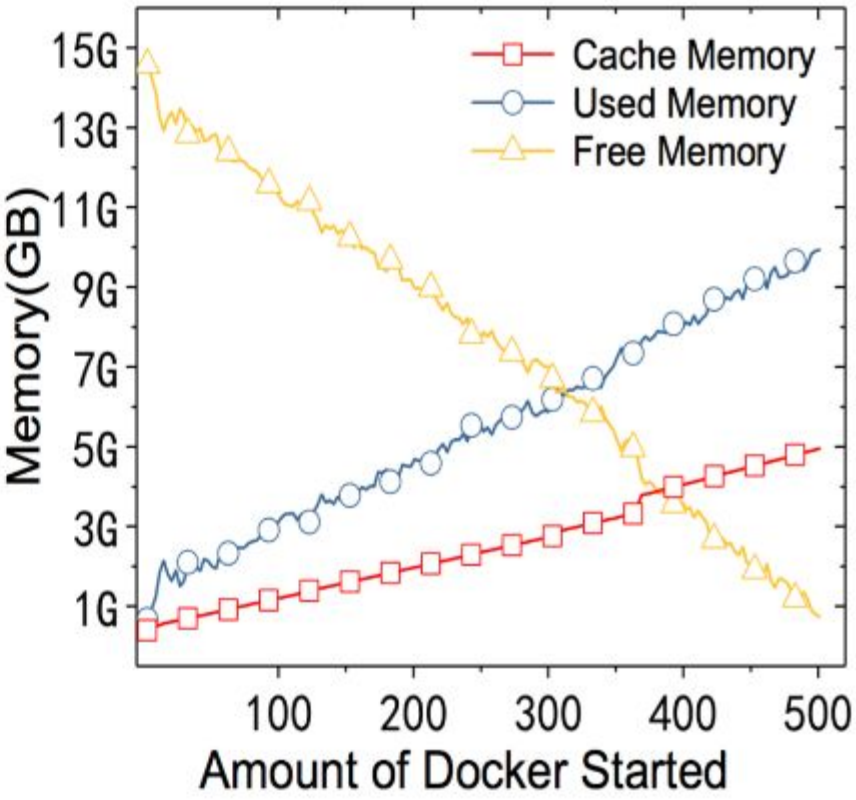
	Total time(in Mins)	Setup time(in Mins)	Start cluster(in Mins)
VM	46	13	5
Container	23	12	5

Bootup Efficiency





(a) VM



(b) Docker container

Drawbacks of Dockers

- Not isolated fully
- Container ecosystem is fractured
- Graphical application does not work well
- Not all applications are benefited by containers

Thank you!!!